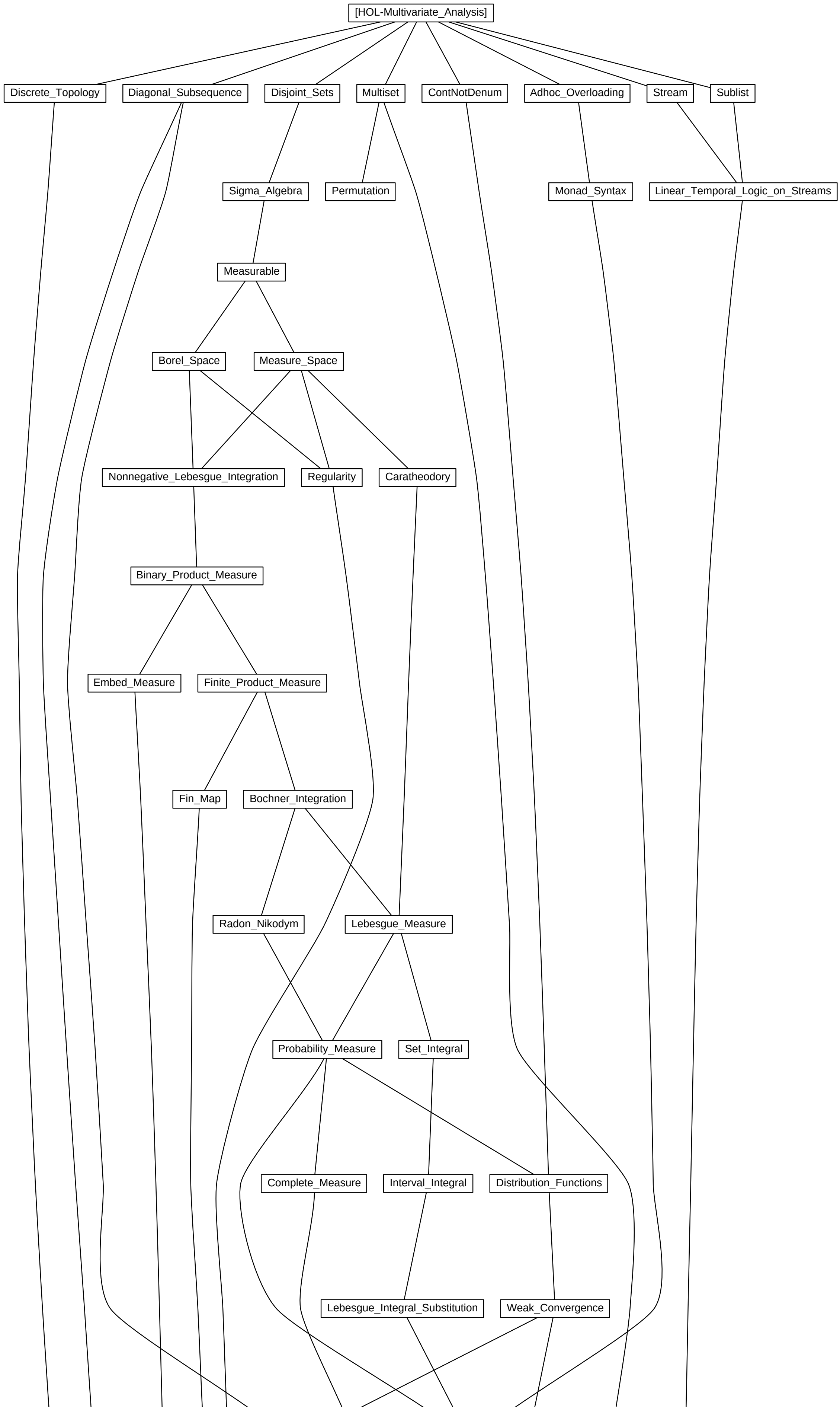
Measure and Probability Theory

April 17, 2016

Contents


[HOL-Multivariate_Analysis]
Discrete_Topology
Diagonal_Subsequence
Disjoint_Sets
Multiset
ContNotDenum
Adhoc_Overloading
Stream
Sublist
Sigma_Algebra
Permutation
Monad_Syntax
Linear_Temporal_Logic_on_Streams
Measurable
Borel_Space
Measure_Space
Nonnegative_Lebesgue_Integration
Regularity
Caratheodory
Binary_Product_Measure
Embed_Measure
Finite_Product_Measure
Fin_Map
Bochner_Integration
Radon_Nikodym
Lebesgue_Measure
Probability_Measure
Set_Integral
Complete_Measure
Interval_Integral
Distribution_Functions
Lebesgue_Integral_Substitution
Weak_Convergence

theory *Discrete-Topology*
imports ~~/*src*/*HOL*/*Multivariate-Analysis*/*Multivariate-Analysis*
begin

Copy of discrete types with discrete topology. This space is polish.

typedef *'a discrete* = *UNIV*::*'a set*
morphisms *of-discrete discrete*
⟨*proof*⟩

instantiation *discrete* :: (*type*) *metric-space*
begin

definition *dist-discrete* :: *'a discrete* ⇒ *'a discrete* ⇒ *real*
where *dist-discrete n m* = (*if n* = *m then 0 else 1*)

definition *uniformity-discrete* :: (*'a discrete* × *'a discrete*) *filter* **where**
(*uniformity*::(*'a discrete* × *'a discrete*) *filter*) = (*INF e*:{*0* <..}. *principal* {(*x*, *y*). *dist x y* < *e*})

definition *open-discrete* :: *'a discrete set* ⇒ *bool* **where**
(*open*::*'a discrete set* ⇒ *bool*) *U* ⟷ (∀ *x*∈*U*. *eventually* (λ(*x'*, *y*). *x'* = *x* ⟶ *y* ∈ *U*) *uniformity*)

instance ⟨*proof*⟩
end

lemma *open-discrete*: *open* (*S* :: *'a discrete set*)
⟨*proof*⟩

instance *discrete* :: (*type*) *complete-space*
⟨*proof*⟩

instance *discrete* :: (*countable*) *countable*
⟨*proof*⟩

instance *discrete* :: (*countable*) *second-countable-topology*
⟨*proof*⟩

instance *discrete* :: (*countable*) *polish-space* ⟨*proof*⟩

end

1 Handling Disjoint Sets

theory *Disjoint-Sets*
imports *Main*
begin

lemma *range-subsetD*: $range\ f \subseteq B \Longrightarrow f\ i \in B$
⟨*proof*⟩

lemma *Int-Diff-disjoint*: $A \cap B \cap (A - B) = \{\}$
⟨*proof*⟩

lemma *Int-Diff-Un*: $A \cap B \cup (A - B) = A$
⟨*proof*⟩

lemma *mono-Un*: $mono\ A \Longrightarrow (\bigcup i{\le}n.\ A\ i) = A\ n$
⟨*proof*⟩

1.1 Set of Disjoint Sets

abbreviation *disjoint* :: $'a\ set\ set \Rightarrow bool$ **where** $disjoint \equiv pairwise\ disjnt$

lemma *disjoint-def*: $disjoint\ A \longleftrightarrow (\forall\, a{\in}A.\ \forall\, b{\in}A.\ a \neq b \longrightarrow a \cap b = \{\})$
⟨*proof*⟩

lemma *disjointI*:
$(\bigwedge a\ b.\ a \in A \Longrightarrow b \in A \Longrightarrow a \neq b \Longrightarrow a \cap b = \{\}) \Longrightarrow disjoint\ A$
⟨*proof*⟩

lemma *disjointD*:
$disjoint\ A \Longrightarrow a \in A \Longrightarrow b \in A \Longrightarrow a \neq b \Longrightarrow a \cap b = \{\}$
⟨*proof*⟩

lemma *disjoint-INT*:
assumes $*$: $\bigwedge i.\ i \in I \Longrightarrow disjoint\ (F\ i)$
shows $disjoint\ \{\bigcap i{\in}I.\ X\ i \mid X.\ \forall\, i{\in}I.\ X\ i \in F\ i\}$
⟨*proof*⟩

1.1.1 Family of Disjoint Sets

definition *disjoint-family-on* :: $('i \Rightarrow 'a\ set) \Rightarrow 'i\ set \Rightarrow bool$ **where**
$disjoint\text{-}family\text{-}on\ A\ S \longleftrightarrow (\forall\, m{\in}S.\ \forall\, n{\in}S.\ m \neq n \longrightarrow A\ m \cap A\ n = \{\})$

abbreviation $disjoint\text{-}family\ A \equiv disjoint\text{-}family\text{-}on\ A\ UNIV$

lemma *disjoint-family-onD*:
$disjoint\text{-}family\text{-}on\ A\ I \Longrightarrow i \in I \Longrightarrow j \in I \Longrightarrow i \neq j \Longrightarrow A\ i \cap A\ j = \{\}$
⟨*proof*⟩

lemma *disjoint-family-subset*: $disjoint\text{-}family\ A \Longrightarrow (\bigwedge x.\ B\ x \subseteq A\ x) \Longrightarrow disjoint\text{-}family$
B
⟨*proof*⟩

lemma *disjoint-family-on-bisimulation*:
assumes $disjoint\text{-}family\text{-}on\ f\ S$

and $\bigwedge n\ m.\ n \in S \Longrightarrow m \in S \Longrightarrow n \neq m \Longrightarrow f\ n \cap f\ m = \{\} \Longrightarrow g\ n \cap g\ m = \{\}$
shows *disjoint-family-on g S*
⟨*proof*⟩

lemma *disjoint-family-on-mono*:
$A \subseteq B \Longrightarrow$ *disjoint-family-on f B* $\Longrightarrow$ *disjoint-family-on f A*
⟨*proof*⟩

lemma *disjoint-family-Suc*:
$(\bigwedge n.\ A\ n \subseteq A\ (Suc\ n)) \Longrightarrow$ *disjoint-family* $(\lambda i.\ A\ (Suc\ i) - A\ i)$
⟨*proof*⟩

lemma *disjoint-family-on-disjoint-image*:
disjoint-family-on A I $\Longrightarrow$ *disjoint* $(A\ \text{`}\ I)$
⟨*proof*⟩

lemma *disjoint-family-on-vimageI*: *disjoint-family-on F I* $\Longrightarrow$ *disjoint-family-on* $(\lambda i.\ f\ -\text{`}\ F\ i)\ I$
⟨*proof*⟩

lemma *disjoint-image-disjoint-family-on*:
assumes *d*: *disjoint* $(A\ \text{`}\ I)$ **and** *i*: *inj-on A I*
shows *disjoint-family-on A I*
⟨*proof*⟩

lemma *disjoint-UN*:
assumes *F*: $\bigwedge i.\ i \in I \Longrightarrow$ *disjoint* $(F\ i)$ **and** $*$: *disjoint-family-on* $(\lambda i.\ \bigcup F\ i)\ I$
shows *disjoint* $(\bigcup i{\in}I.\ F\ i)$
⟨*proof*⟩

lemma *disjoint-union*: *disjoint C* $\Longrightarrow$ *disjoint B* $\Longrightarrow \bigcup C \cap \bigcup B = \{\} \Longrightarrow$ *disjoint* $(C \cup B)$
⟨*proof*⟩

1.2 Construct Disjoint Sequences

definition *disjointed* :: $(nat \Rightarrow 'a\ set) \Rightarrow nat \Rightarrow 'a\ set$ **where**
disjointed $A\ n = A\ n - (\bigcup i{\in}\{0..{<}n\}.\ A\ i)$

lemma *finite-UN-disjointed-eq*: $(\bigcup i{\in}\{0..{<}n\}.$ *disjointed* $A\ i) = (\bigcup i{\in}\{0..{<}n\}.\ A\ i)$
⟨*proof*⟩

lemma *UN-disjointed-eq*: $(\bigcup i.$ *disjointed* $A\ i) = (\bigcup i.\ A\ i)$
⟨*proof*⟩

lemma *less-disjoint-disjointed*: $m < n \Longrightarrow$ *disjointed* $A\ m \cap$ *disjointed* $A\ n = \{\}$
⟨*proof*⟩

lemma *disjoint-family-disjointed*: *disjoint-family* (*disjointed A*)
⟨*proof*⟩

lemma *disjointed-subset*: *disjointed A n* ⊆ *A n*
⟨*proof*⟩

lemma *disjointed-0*[*simp*]: *disjointed A 0* = *A 0*
⟨*proof*⟩

lemma *disjointed-mono*: *mono A* ⟹ *disjointed A* (*Suc n*) = *A* (*Suc n*) − *A n*
⟨*proof*⟩

end

2 Describing measurable sets

theory *Sigma-Algebra*
imports
Complex-Main
~~/src/HOL/Library/Countable-Set
~~/src/HOL/Library/FuncSet
~~/src/HOL/Library/Indicator-Function
~~/src/HOL/Library/Extended-Nonnegative-Real
~~/src/HOL/Library/Disjoint-Sets
begin

Sigma algebras are an elementary concept in measure theory. To measure — that is to integrate — functions, we first have to measure sets. Unfortunately, when dealing with a large universe, it is often not possible to consistently assign a measure to every subset. Therefore it is necessary to define the set of measurable subsets of the universe. A sigma algebra is such a set that has three very natural and desirable properties.

2.1 Families of sets

locale *subset-class* =
fixes Ω :: *'a set* **and** *M* :: *'a set set*
assumes *space-closed*: *M* ⊆ *Pow* Ω

lemma (**in** *subset-class*) *sets-into-space*: *x* ∈ *M* ⟹ *x* ⊆ Ω
⟨*proof*⟩

2.1.1 Semiring of sets

locale *semiring-of-sets* = *subset-class* +
assumes *empty-sets*[*iff*]: {} ∈ *M*
assumes *Int*[*intro*]: ⋀*a b*. *a* ∈ *M* ⟹ *b* ∈ *M* ⟹ *a* ∩ *b* ∈ *M*

assumes *Diff-cover*:
$\bigwedge a\ b.\ a \in M \Longrightarrow b \in M \Longrightarrow \exists\, C{\subseteq}M.\ \mathit{finite}\ C \wedge \mathit{disjoint}\ C \wedge a - b = \bigcup C$

lemma (**in** *semiring-of-sets*) *finite-INT*[*intro*]:
assumes $\mathit{finite}\ I\ I \neq \{\}\ \bigwedge i.\ i \in I \Longrightarrow A\ i \in M$
shows $(\bigcap i{\in}I.\ A\ i) \in M$
⟨*proof*⟩

lemma (**in** *semiring-of-sets*) *Int-space-eq1* [*simp*]: $x \in M \Longrightarrow \Omega \cap x = x$
⟨*proof*⟩

lemma (**in** *semiring-of-sets*) *Int-space-eq2* [*simp*]: $x \in M \Longrightarrow x \cap \Omega = x$
⟨*proof*⟩

lemma (**in** *semiring-of-sets*) *sets-Collect-conj*:
assumes $\{x{\in}\Omega.\ P\ x\} \in M\ \{x{\in}\Omega.\ Q\ x\} \in M$
shows $\{x{\in}\Omega.\ Q\ x \wedge P\ x\} \in M$
⟨*proof*⟩

lemma (**in** *semiring-of-sets*) *sets-Collect-finite-All′*:
assumes $\bigwedge i.\ i \in S \Longrightarrow \{x{\in}\Omega.\ P\ i\ x\} \in M\ \mathit{finite}\ S\ S \neq \{\}$
shows $\{x{\in}\Omega.\ \forall\, i{\in}S.\ P\ i\ x\} \in M$
⟨*proof*⟩

locale *ring-of-sets* = *semiring-of-sets* +
assumes *Un* [*intro*]: $\bigwedge a\ b.\ a \in M \Longrightarrow b \in M \Longrightarrow a \cup b \in M$

lemma (**in** *ring-of-sets*) *finite-Union* [*intro*]:
$\mathit{finite}\ X \Longrightarrow X \subseteq M \Longrightarrow \bigcup X \in M$
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *finite-UN*[*intro*]:
assumes *finite I* **and** $\bigwedge i.\ i \in I \Longrightarrow A\ i \in M$
shows $(\bigcup i{\in}I.\ A\ i) \in M$
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *Diff* [*intro*]:
assumes $a \in M\ b \in M$ **shows** $a - b \in M$
⟨*proof*⟩

lemma *ring-of-setsI*:
assumes *space-closed*: $M \subseteq \mathit{Pow}\ \Omega$
assumes *empty-sets*[*iff*]: $\{\} \in M$
assumes *Un*[*intro*]: $\bigwedge a\ b.\ a \in M \Longrightarrow b \in M \Longrightarrow a \cup b \in M$
assumes *Diff*[*intro*]: $\bigwedge a\ b.\ a \in M \Longrightarrow b \in M \Longrightarrow a - b \in M$
shows *ring-of-sets* Ω M
⟨*proof*⟩

lemma *ring-of-sets-iff*: $\mathit{ring\text{-}of\text{-}sets}\ \Omega\ M \longleftrightarrow M \subseteq \mathit{Pow}\ \Omega \wedge \{\} \in M \wedge (\forall\, a{\in}M.$

$\forall\, b{\in}M.\ a \cup b \in M) \wedge (\forall\, a{\in}M.\ \forall\, b{\in}M.\ a - b \in M)$
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *insert-in-sets*:
assumes $\{x\} \in M$ $A \in M$ **shows** *insert* $x\ A \in M$
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *sets-Collect-disj*:
assumes $\{x{\in}\Omega.\ P\ x\} \in M$ $\{x{\in}\Omega.\ Q\ x\} \in M$
shows $\{x{\in}\Omega.\ Q\ x \vee P\ x\} \in M$
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *sets-Collect-finite-Ex*:
assumes $\bigwedge i.\ i \in S \Longrightarrow \{x{\in}\Omega.\ P\ i\ x\} \in M$ *finite* S
shows $\{x{\in}\Omega.\ \exists\, i{\in}S.\ P\ i\ x\} \in M$
⟨*proof*⟩

locale *algebra* = *ring-of-sets* +
assumes *top* [*iff*]: $\Omega \in M$

lemma (**in** *algebra*) *compl-sets* [*intro*]:
$a \in M \Longrightarrow \Omega - a \in M$
⟨*proof*⟩

lemma *algebra-iff-Un*:
algebra $\Omega\ M \longleftrightarrow$
$M \subseteq Pow\ \Omega\ \wedge$
$\{\} \in M\ \wedge$
$(\forall\, a \in M.\ \Omega - a \in M)\ \wedge$
$(\forall\, a \in M.\ \forall\ b \in M.\ a \cup b \in M)$ (**is** - $\longleftrightarrow$ *?Un*)
⟨*proof*⟩

lemma *algebra-iff-Int*:
algebra $\Omega\ M \longleftrightarrow$
$M \subseteq Pow\ \Omega\ \&\ \{\} \in M\ \&$
$(\forall\, a \in M.\ \Omega - a \in M)\ \&$
$(\forall\, a \in M.\ \forall\ b \in M.\ a \cap b \in M)$ (**is** - $\longleftrightarrow$ *?Int*)
⟨*proof*⟩

lemma (**in** *algebra*) *sets-Collect-neg*:
assumes $\{x{\in}\Omega.\ P\ x\} \in M$
shows $\{x{\in}\Omega.\ \neg\ P\ x\} \in M$
⟨*proof*⟩

lemma (**in** *algebra*) *sets-Collect-imp*:
$\{x{\in}\Omega.\ P\ x\} \in M \Longrightarrow \{x{\in}\Omega.\ Q\ x\} \in M \Longrightarrow \{x{\in}\Omega.\ Q\ x \longrightarrow P\ x\} \in M$
⟨*proof*⟩

lemma (**in** *algebra*) *sets-Collect-const*:

$\{x{\in}\Omega.\ P\} \in M$
⟨*proof*⟩

lemma *algebra-single-set*:
$X \subseteq S \Longrightarrow algebra\ S\ \{\ \{\},\ X,\ S - X,\ S\ \}$
⟨*proof*⟩

2.1.2 Restricted algebras

abbreviation (**in** *algebra*)
$restricted\text{-}space\ A \equiv (op\ \cap\ A)\ `\ M$

lemma (**in** *algebra*) *restricted-algebra*:
assumes $A \in M$ **shows** *algebra A* (*restricted-space A*)
⟨*proof*⟩

2.1.3 Sigma Algebras

locale *sigma-algebra* = *algebra* +
assumes *countable-nat-UN* [*intro*]: $\bigwedge A.\ range\ A \subseteq M \Longrightarrow (\bigcup i{::}nat.\ A\ i) \in M$

lemma (**in** *algebra*) *is-sigma-algebra*:
assumes *finite M*
shows *sigma-algebra* Ω *M*
⟨*proof*⟩

lemma *countable-UN-eq*:
fixes $A :: 'i{::}countable \Rightarrow 'a\ set$
shows $(range\ A \subseteq M \longrightarrow (\bigcup i.\ A\ i) \in M) \longleftrightarrow$
$(range\ (A \circ from\text{-}nat) \subseteq M \longrightarrow (\bigcup i.\ (A \circ from\text{-}nat)\ i) \in M)$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *countable-Union* [*intro*]:
assumes *countable X* $X \subseteq M$ **shows** $\bigcup X \in M$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *countable-UN*[*intro*]:
fixes $A :: 'i{::}countable \Rightarrow 'a\ set$
assumes $A`X \subseteq M$
shows $(\bigcup x{\in}X.\ A\ x) \in M$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *countable-UN'*:
fixes $A :: 'i \Rightarrow 'a\ set$
assumes *X*: *countable X*
assumes *A*: $A`X \subseteq M$
shows $(\bigcup x{\in}X.\ A\ x) \in M$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *countable-UN''*:

$[\![$ *countable* X; $\bigwedge x\ y.\ x \in X \Longrightarrow A\ x \in M\ ]\!] \Longrightarrow (\bigcup x{\in}X.\ A\ x) \in M$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *countable-INT* [*intro*]:
fixes A :: $'i$::*countable* $\Rightarrow$ $'a$ *set*
assumes A: $A\,`X \subseteq M\ X \neq \{\}$
shows $(\bigcap i{\in}X.\ A\ i) \in M$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *countable-INT'*:
fixes A :: $'i \Rightarrow\ 'a$ *set*
assumes X: *countable* $X\ X \neq \{\}$
assumes A: $A\,`X \subseteq M$
shows $(\bigcap x{\in}X.\ A\ x) \in M$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *countable-INT''*:
UNIV $\in M \Longrightarrow$ *countable* $I \Longrightarrow (\bigwedge i.\ i \in I \Longrightarrow F\ i \in M) \Longrightarrow (\bigcap i{\in}I.\ F\ i) \in M$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *countable*:
assumes $\bigwedge a.\ a \in A \Longrightarrow \{a\} \in M$ *countable* A
shows $A \in M$
⟨*proof*⟩

lemma *ring-of-sets-Pow*: *ring-of-sets sp* (*Pow sp*)
⟨*proof*⟩

lemma *algebra-Pow*: *algebra sp* (*Pow sp*)
⟨*proof*⟩

lemma *sigma-algebra-iff*:
sigma-algebra Ω $M \longleftrightarrow$
algebra Ω $M \wedge (\forall A.$ *range* $A \subseteq M \longrightarrow (\bigcup i{::}nat.\ A\ i) \in M)$
⟨*proof*⟩

lemma *sigma-algebra-Pow*: *sigma-algebra sp* (*Pow sp*)
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *sets-Collect-countable-All*:
assumes $\bigwedge i.\ \{x{\in}\Omega.\ P\ i\ x\} \in M$
shows $\{x{\in}\Omega.\ \forall i{::}'i{::}countable.\ P\ i\ x\} \in M$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *sets-Collect-countable-Ex*:
assumes $\bigwedge i.\ \{x{\in}\Omega.\ P\ i\ x\} \in M$
shows $\{x{\in}\Omega.\ \exists i{::}'i{::}countable.\ P\ i\ x\} \in M$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *sets-Collect-countable-Ex'*:
assumes $\bigwedge i.\ i \in I \Longrightarrow \{x{\in}\Omega.\ P\ i\ x\} \in M$
assumes *countable I*
shows $\{x{\in}\Omega.\ \exists\, i{\in}I.\ P\ i\ x\} \in M$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *sets-Collect-countable-All'*:
assumes $\bigwedge i.\ i \in I \Longrightarrow \{x{\in}\Omega.\ P\ i\ x\} \in M$
assumes *countable I*
shows $\{x{\in}\Omega.\ \forall\, i{\in}I.\ P\ i\ x\} \in M$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *sets-Collect-countable-Ex1'*:
assumes $\bigwedge i.\ i \in I \Longrightarrow \{x{\in}\Omega.\ P\ i\ x\} \in M$
assumes *countable I*
shows $\{x{\in}\Omega.\ \exists!i{\in}I.\ P\ i\ x\} \in M$
⟨*proof*⟩

lemmas (**in** *sigma-algebra*) *sets-Collect* =
sets-Collect-imp sets-Collect-disj sets-Collect-conj sets-Collect-neg sets-Collect-const
sets-Collect-countable-All sets-Collect-countable-Ex sets-Collect-countable-All

lemma (**in** *sigma-algebra*) *sets-Collect-countable-Ball*:
assumes $\bigwedge i.\ \{x{\in}\Omega.\ P\ i\ x\} \in M$
shows $\{x{\in}\Omega.\ \forall\, i{::}'i{::}countable{\in}X.\ P\ i\ x\} \in M$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *sets-Collect-countable-Bex*:
assumes $\bigwedge i.\ \{x{\in}\Omega.\ P\ i\ x\} \in M$
shows $\{x{\in}\Omega.\ \exists\, i{::}'i{::}countable{\in}X.\ P\ i\ x\} \in M$
⟨*proof*⟩

lemma *sigma-algebra-single-set*:
assumes $X \subseteq S$
shows *sigma-algebra* $S\ \{\ \{\},\ X,\ S - X,\ S\ \}$
⟨*proof*⟩

2.1.4 Binary Unions

definition *binary* :: $'a \Rightarrow 'a \Rightarrow nat \Rightarrow 'a$
where $binary\ a\ b = \ (\lambda x.\ b)(0 := a)$

lemma *range-binary-eq*: $range(binary\ a\ b) = \{a,b\}$
⟨*proof*⟩

lemma *Un-range-binary*: $a \cup b = (\bigcup i{::}nat.\ binary\ a\ b\ i)$
⟨*proof*⟩

lemma *Int-range-binary*: $a \cap b = (\bigcap i{::}nat.\ binary\ a\ b\ i)$

⟨*proof*⟩

lemma *sigma-algebra-iff2*:
$sigma\text{-}algebra\ \Omega\ M \longleftrightarrow$
$M \subseteq Pow\ \Omega\ \wedge$
$\{\} \in M \wedge (\forall s \in M.\ \Omega - s \in M) \wedge$
$(\forall A.\ range\ A \subseteq M \longrightarrow (\bigcup i{::}nat.\ A\ i) \in M)$
⟨*proof*⟩

2.1.5 Initial Sigma Algebra

Sigma algebras can naturally be created as the closure of any set of M with regard to the properties just postulated.

inductive-set *sigma-sets* :: $'a\ set \Rightarrow 'a\ set\ set \Rightarrow 'a\ set\ set$
for *sp* :: $'a\ set$ **and** *A* :: $'a\ set\ set$
where
Basic[*intro*, *simp*]: $a \in A \Longrightarrow a \in sigma\text{-}sets\ sp\ A$
| *Empty*: $\{\} \in sigma\text{-}sets\ sp\ A$
| *Compl*: $a \in sigma\text{-}sets\ sp\ A \Longrightarrow sp - a \in sigma\text{-}sets\ sp\ A$
| *Union*: $(\bigwedge i{::}nat.\ a\ i \in sigma\text{-}sets\ sp\ A) \Longrightarrow (\bigcup i.\ a\ i) \in sigma\text{-}sets\ sp\ A$

lemma (**in** *sigma-algebra*) *sigma-sets-subset*:
assumes *a*: $a \subseteq M$
shows $sigma\text{-}sets\ \Omega\ a \subseteq M$
⟨*proof*⟩

lemma *sigma-sets-into-sp*: $A \subseteq Pow\ sp \Longrightarrow x \in sigma\text{-}sets\ sp\ A \Longrightarrow x \subseteq sp$
⟨*proof*⟩

lemma *sigma-algebra-sigma-sets*:
$a \subseteq Pow\ \Omega \Longrightarrow sigma\text{-}algebra\ \Omega\ (sigma\text{-}sets\ \Omega\ a)$
⟨*proof*⟩

lemma *sigma-sets-least-sigma-algebra*:
assumes $A \subseteq Pow\ S$
shows $sigma\text{-}sets\ S\ A = \bigcap\{B.\ A \subseteq B \wedge sigma\text{-}algebra\ S\ B\}$
⟨*proof*⟩

lemma *sigma-sets-top*: $sp \in sigma\text{-}sets\ sp\ A$
⟨*proof*⟩

lemma *sigma-sets-Un*:
$a \in sigma\text{-}sets\ sp\ A \Longrightarrow b \in sigma\text{-}sets\ sp\ A \Longrightarrow a \cup b \in sigma\text{-}sets\ sp\ A$
⟨*proof*⟩

lemma *sigma-sets-Inter*:
assumes *Asb*: $A \subseteq Pow\ sp$
shows $(\bigwedge i{::}nat.\ a\ i \in sigma\text{-}sets\ sp\ A) \Longrightarrow (\bigcap i.\ a\ i) \in sigma\text{-}sets\ sp\ A$
⟨*proof*⟩

lemma *sigma-sets-INTER*:
assumes *Asb*: $A \subseteq Pow\ sp$
and *ai*: $\bigwedge i::nat.\ i \in S \Longrightarrow a\ i \in sigma\text{-}sets\ sp\ A$ **and** *non*: $S \neq \{\}$
shows $(\bigcap i{\in}S.\ a\ i) \in sigma\text{-}sets\ sp\ A$
⟨*proof*⟩

lemma *sigma-sets-UNION*:
$countable\ B \Longrightarrow (\bigwedge b.\ b \in B \Longrightarrow b \in sigma\text{-}sets\ X\ A) \Longrightarrow (\bigcup B) \in sigma\text{-}sets\ X\ A$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *sigma-sets-eq*:
$sigma\text{-}sets\ \Omega\ M = M$
⟨*proof*⟩

lemma *sigma-sets-eqI*:
assumes *A*: $\bigwedge a.\ a \in A \Longrightarrow a \in sigma\text{-}sets\ M\ B$
assumes *B*: $\bigwedge b.\ b \in B \Longrightarrow b \in sigma\text{-}sets\ M\ A$
shows $sigma\text{-}sets\ M\ A = sigma\text{-}sets\ M\ B$
⟨*proof*⟩

lemma *sigma-sets-subseteq*: **assumes** $A \subseteq B$ **shows** $sigma\text{-}sets\ X\ A \subseteq sigma\text{-}sets\ X\ B$
⟨*proof*⟩

lemma *sigma-sets-mono*: **assumes** $A \subseteq sigma\text{-}sets\ X\ B$ **shows** $sigma\text{-}sets\ X\ A \subseteq sigma\text{-}sets\ X\ B$
⟨*proof*⟩

lemma *sigma-sets-mono'*: **assumes** $A \subseteq B$ **shows** $sigma\text{-}sets\ X\ A \subseteq sigma\text{-}sets\ X\ B$
⟨*proof*⟩

lemma *sigma-sets-superset-generator*: $A \subseteq sigma\text{-}sets\ X\ A$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *restriction-in-sets*:
fixes $A :: nat \Rightarrow 'a\ set$
assumes $S \in M$
and $*$: $range\ A \subseteq (\lambda A.\ S \cap A)\ `\ M$ (**is** - $\subseteq$ *?r*)
shows $range\ A \subseteq M\ (\bigcup i.\ A\ i) \in (\lambda A.\ S \cap A)\ `\ M$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *restricted-sigma-algebra*:
assumes $S \in M$
shows $sigma\text{-}algebra\ S\ (restricted\text{-}space\ S)$
⟨*proof*⟩

lemma *sigma-sets-Int*:
assumes $A \in$ *sigma-sets sp st* $A \subseteq sp$
shows $op \cap A$ ‘ *sigma-sets sp st* = *sigma-sets A* ($op \cap A$ ‘ st)
⟨*proof*⟩

lemma *sigma-sets-empty-eq*: *sigma-sets A* {} = {{}, A}
⟨*proof*⟩

lemma *sigma-sets-single*[*simp*]: *sigma-sets A* {A} = {{}, A}
⟨*proof*⟩

lemma *sigma-sets-sigma-sets-eq*:
$M \subseteq$ *Pow S* $\Longrightarrow$ *sigma-sets S* (*sigma-sets S M*) = *sigma-sets S M*
⟨*proof*⟩

lemma *sigma-sets-singleton*:
assumes $X \subseteq S$
shows *sigma-sets S* { X } = { {}, X, S − X, S }
⟨*proof*⟩

lemma *restricted-sigma*:
assumes S: $S \in$ *sigma-sets* Ω M **and** M: $M \subseteq$ *Pow* Ω
shows *algebra.restricted-space* (*sigma-sets* Ω M) S =
sigma-sets S (*algebra.restricted-space M S*)
⟨*proof*⟩

lemma *sigma-sets-vimage-commute*:
assumes X: $X \in \Omega \to \Omega'$
shows {X −‘ $A \cap \Omega$ |A. $A \in$ *sigma-sets* Ω' M'}
= *sigma-sets* Ω {X −‘ $A \cap \Omega$ |A. $A \in M'$} (**is** ?L = ?R)
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *UNION-in-sets*:
fixes A:: *nat* ⇒ *'a set*
assumes A: *range A* $\subseteq M$
shows ($\bigcup i \in \{0..<n\}$. A i) $\in M$
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *range-disjointed-sets*:
assumes A: *range A* $\subseteq M$
shows *range* (*disjointed A*) $\subseteq M$
⟨*proof*⟩

lemma (**in** *algebra*) *range-disjointed-sets'*:
range A $\subseteq M \Longrightarrow$ *range* (*disjointed A*) $\subseteq M$
⟨*proof*⟩

lemma *sigma-algebra-disjoint-iff*:
sigma-algebra Ω M $\longleftrightarrow$ *algebra* Ω M $\wedge$

($\forall A.$ *range* $A \subseteq M \longrightarrow$ *disjoint-family* $A \longrightarrow (\bigcup i::nat.\ A\ i) \in M$)
⟨*proof*⟩

2.1.6 Ring generated by a semiring

definition (**in** *semiring-of-sets*)
generated-ring $= \{ \bigcup C \mid C.\ C \subseteq M \wedge$ *finite* $C \wedge$ *disjoint* $C \}$

lemma (**in** *semiring-of-sets*) *generated-ringE*[*elim?*]:
assumes $a \in$ *generated-ring*
obtains C **where** *finite* C *disjoint* C $C \subseteq M$ $a = \bigcup C$
⟨*proof*⟩

lemma (**in** *semiring-of-sets*) *generated-ringI*[*intro?*]:
assumes *finite* C *disjoint* C $C \subseteq M$ $a = \bigcup C$
shows $a \in$ *generated-ring*
⟨*proof*⟩

lemma (**in** *semiring-of-sets*) *generated-ringI-Basic*:
$A \in M \Longrightarrow A \in$ *generated-ring*
⟨*proof*⟩

lemma (**in** *semiring-of-sets*) *generated-ring-disjoint-Un*[*intro*]:
assumes a: $a \in$ *generated-ring* **and** b: $b \in$ *generated-ring*
and $a \cap b = \{\}$
shows $a \cup b \in$ *generated-ring*
⟨*proof*⟩

lemma (**in** *semiring-of-sets*) *generated-ring-empty*: $\{\} \in$ *generated-ring*
⟨*proof*⟩

lemma (**in** *semiring-of-sets*) *generated-ring-disjoint-Union*:
assumes *finite* A **shows** $A \subseteq$ *generated-ring* $\Longrightarrow$ *disjoint* $A \Longrightarrow \bigcup A \in$ *generated-ring*
⟨*proof*⟩

lemma (**in** *semiring-of-sets*) *generated-ring-disjoint-UNION*:
finite $I \Longrightarrow$ *disjoint* $(A$ ‘ $I) \Longrightarrow (\bigwedge i.\ i \in I \Longrightarrow A\ i \in$ *generated-ring*$) \Longrightarrow$ *UNION* $I\ A \in$ *generated-ring*
⟨*proof*⟩

lemma (**in** *semiring-of-sets*) *generated-ring-Int*:
assumes a: $a \in$ *generated-ring* **and** b: $b \in$ *generated-ring*
shows $a \cap b \in$ *generated-ring*
⟨*proof*⟩

lemma (**in** *semiring-of-sets*) *generated-ring-Inter*:
assumes *finite* A $A \neq \{\}$ **shows** $A \subseteq$ *generated-ring* $\Longrightarrow \bigcap A \in$ *generated-ring*
⟨*proof*⟩

lemma (**in** *semiring-of-sets*) *generated-ring-INTER*:
$finite\ I \Longrightarrow I \neq \{\} \Longrightarrow (\bigwedge i.\ i \in I \Longrightarrow A\ i \in generated\text{-}ring) \Longrightarrow INTER\ I\ A \in generated\text{-}ring$
⟨*proof*⟩

lemma (**in** *semiring-of-sets*) *generating-ring*:
$ring\text{-}of\text{-}sets\ \Omega\ generated\text{-}ring$
⟨*proof*⟩

lemma (**in** *semiring-of-sets*) *sigma-sets-generated-ring-eq*: $sigma\text{-}sets\ \Omega\ generated\text{-}ring = sigma\text{-}sets\ \Omega\ M$
⟨*proof*⟩

2.1.7 A Two-Element Series

definition $binaryset :: 'a\ set \Rightarrow 'a\ set \Rightarrow nat \Rightarrow 'a\ set$
where $binaryset\ A\ B = (\lambda x.\ \{\})(0 := A,\ Suc\ 0 := B)$

lemma *range-binaryset-eq*: $range(binaryset\ A\ B) = \{A,B,\{\}\}$
⟨*proof*⟩

lemma *UN-binaryset-eq*: $(\bigcup i.\ binaryset\ A\ B\ i) = A \cup B$
⟨*proof*⟩

2.1.8 Closed CDI

definition *closed-cdi* **where**
$closed\text{-}cdi\ \Omega\ M \longleftrightarrow$
$M \subseteq Pow\ \Omega\ \&$
$(\forall s \in M.\ \Omega - s \in M)\ \&$
$(\forall A.\ (range\ A \subseteq M)\ \&\ (A\ 0 = \{\})\ \&\ (\forall n.\ A\ n \subseteq A\ (Suc\ n)) \longrightarrow$
$(\bigcup i.\ A\ i) \in M)\ \&$
$(\forall A.\ (range\ A \subseteq M)\ \&\ disjoint\text{-}family\ A \longrightarrow (\bigcup i{::}nat.\ A\ i) \in M)$

inductive-set
$smallest\text{-}ccdi\text{-}sets :: 'a\ set \Rightarrow 'a\ set\ set \Rightarrow 'a\ set\ set$
for $\Omega\ M$
where
Basic [*intro*]:
$a \in M \Longrightarrow a \in smallest\text{-}ccdi\text{-}sets\ \Omega\ M$
| *Compl* [*intro*]:
$a \in smallest\text{-}ccdi\text{-}sets\ \Omega\ M \Longrightarrow \Omega - a \in smallest\text{-}ccdi\text{-}sets\ \Omega\ M$
| *Inc*:
$range\ A \in Pow(smallest\text{-}ccdi\text{-}sets\ \Omega\ M) \Longrightarrow A\ 0 = \{\} \Longrightarrow (\bigwedge n.\ A\ n \subseteq A\ (Suc\ n))$
$\Longrightarrow (\bigcup i.\ A\ i) \in smallest\text{-}ccdi\text{-}sets\ \Omega\ M$
| *Disj*:
$range\ A \in Pow(smallest\text{-}ccdi\text{-}sets\ \Omega\ M) \Longrightarrow disjoint\text{-}family\ A$
$\Longrightarrow (\bigcup i{::}nat.\ A\ i) \in smallest\text{-}ccdi\text{-}sets\ \Omega\ M$

lemma (**in** *subset-class*) *smallest-closed-cdi1*: $M \subseteq$ *smallest-ccdi-sets* Ω M
⟨*proof*⟩

lemma (**in** *subset-class*) *smallest-ccdi-sets*: *smallest-ccdi-sets* Ω $M \subseteq$ *Pow* Ω
⟨*proof*⟩

lemma (**in** *subset-class*) *smallest-closed-cdi2*: *closed-cdi* Ω (*smallest-ccdi-sets* Ω M)
⟨*proof*⟩

lemma *closed-cdi-subset*: *closed-cdi* Ω $M \Longrightarrow M \subseteq$ *Pow* Ω
⟨*proof*⟩

lemma *closed-cdi-Compl*: *closed-cdi* Ω $M \Longrightarrow s \in M \Longrightarrow \Omega - s \in M$
⟨*proof*⟩

lemma *closed-cdi-Inc*:
closed-cdi Ω $M \Longrightarrow$ *range* $A \subseteq M \Longrightarrow A\ 0 = \{\} \Longrightarrow (!!n.\ A\ n \subseteq A\ (Suc\ n)) \Longrightarrow (\bigcup i.\ A\ i) \in M$
⟨*proof*⟩

lemma *closed-cdi-Disj*:
closed-cdi Ω $M \Longrightarrow$ *range* $A \subseteq M \Longrightarrow$ *disjoint-family* $A \Longrightarrow (\bigcup i{::}nat.\ A\ i) \in M$
⟨*proof*⟩

lemma *closed-cdi-Un*:
assumes *cdi*: *closed-cdi* Ω M **and** *empty*: $\{\} \in M$
and A: $A \in M$ **and** B: $B \in M$
and *disj*: $A \cap B = \{\}$
shows $A \cup B \in M$
⟨*proof*⟩

lemma (**in** *algebra*) *smallest-ccdi-sets-Un*:
assumes A: $A \in$ *smallest-ccdi-sets* Ω M **and** B: $B \in$ *smallest-ccdi-sets* Ω M
and *disj*: $A \cap B = \{\}$
shows $A \cup B \in$ *smallest-ccdi-sets* Ω M
⟨*proof*⟩

lemma (**in** *algebra*) *smallest-ccdi-sets-Int1*:
assumes a: $a \in M$
shows $b \in$ *smallest-ccdi-sets* Ω $M \Longrightarrow a \cap b \in$ *smallest-ccdi-sets* Ω M
⟨*proof*⟩

lemma (**in** *algebra*) *smallest-ccdi-sets-Int*:
assumes b: $b \in$ *smallest-ccdi-sets* Ω M
shows $a \in$ *smallest-ccdi-sets* Ω $M \Longrightarrow a \cap b \in$ *smallest-ccdi-sets* Ω M
⟨*proof*⟩

lemma (**in** *algebra*) *sigma-property-disjoint-lemma*:
assumes *sbC*: $M \subseteq C$
and *ccdi*: *closed-cdi* Ω C
shows *sigma-sets* Ω $M \subseteq C$
⟨*proof*⟩

lemma (**in** *algebra*) *sigma-property-disjoint*:
assumes *sbC*: $M \subseteq C$
and *compl*: $!!s.\ s \in C \cap$ *sigma-sets* $(\Omega)\ (M) \Longrightarrow \Omega - s \in C$
and *inc*: $!!A.$ *range* $A \subseteq C \cap$ *sigma-sets* $(\Omega)\ (M)$
$\Longrightarrow A\ 0 = \{\} \Longrightarrow (!!n.\ A\ n \subseteq A\ (Suc\ n))$
$\Longrightarrow (\bigcup i.\ A\ i) \in C$
and *disj*: $!!A.$ *range* $A \subseteq C \cap$ *sigma-sets* $(\Omega)\ (M)$
$\Longrightarrow$ *disjoint-family* $A \Longrightarrow (\bigcup i{::}nat.\ A\ i) \in C$
shows *sigma-sets* $(\Omega)\ (M) \subseteq C$
⟨*proof*⟩

2.1.9 Dynkin systems

locale *dynkin-system* = *subset-class* +
assumes *space*: $\Omega \in M$
and *compl*[*intro*!]: $\bigwedge A.\ A \in M \Longrightarrow \Omega - A \in M$
and *UN*[*intro*!]: $\bigwedge A.$ *disjoint-family* $A \Longrightarrow$ *range* $A \subseteq M$
$\Longrightarrow (\bigcup i{::}nat.\ A\ i) \in M$

lemma (**in** *dynkin-system*) *empty*[*intro*, *simp*]: $\{\} \in M$
⟨*proof*⟩

lemma (**in** *dynkin-system*) *diff*:
assumes *sets*: $D \in M$ $E \in M$ **and** $D \subseteq E$
shows $E - D \in M$
⟨*proof*⟩

lemma *dynkin-systemI*:
assumes $\bigwedge A.\ A \in M \Longrightarrow A \subseteq \Omega$ $\Omega \in M$
assumes $\bigwedge A.\ A \in M \Longrightarrow \Omega - A \in M$
assumes $\bigwedge A.$ *disjoint-family* $A \Longrightarrow$ *range* $A \subseteq M$
$\Longrightarrow (\bigcup i{::}nat.\ A\ i) \in M$
shows *dynkin-system* Ω M
⟨*proof*⟩

lemma *dynkin-systemI'*:
assumes *1*: $\bigwedge A.\ A \in M \Longrightarrow A \subseteq \Omega$
assumes *empty*: $\{\} \in M$
assumes *Diff*: $\bigwedge A.\ A \in M \Longrightarrow \Omega - A \in M$
assumes *2*: $\bigwedge A.$ *disjoint-family* $A \Longrightarrow$ *range* $A \subseteq M$
$\Longrightarrow (\bigcup i{::}nat.\ A\ i) \in M$
shows *dynkin-system* Ω M
⟨*proof*⟩

lemma *dynkin-system-trivial*:
shows *dynkin-system* A (*Pow* A)
⟨*proof*⟩

lemma *sigma-algebra-imp-dynkin-system*:
assumes *sigma-algebra* Ω M **shows** *dynkin-system* Ω M
⟨*proof*⟩

2.1.10 Intersection sets systems

definition *Int-stable* $M \longleftrightarrow (\forall\ a \in M.\ \forall\ b \in M.\ a \cap b \in M)$

lemma (**in** *algebra*) *Int-stable*: *Int-stable* M
⟨*proof*⟩

lemma *Int-stableI*:
$(\bigwedge a\ b.\ a \in A \Longrightarrow b \in A \Longrightarrow a \cap b \in A) \Longrightarrow$ *Int-stable* A
⟨*proof*⟩

lemma *Int-stableD*:
Int-stable $M \Longrightarrow a \in M \Longrightarrow b \in M \Longrightarrow a \cap b \in M$
⟨*proof*⟩

lemma (**in** *dynkin-system*) *sigma-algebra-eq-Int-stable*:
sigma-algebra Ω $M \longleftrightarrow$ *Int-stable* M
⟨*proof*⟩

2.1.11 Smallest Dynkin systems

definition *dynkin* **where**
dynkin Ω M = $(\bigcap\{D.$ *dynkin-system* $\Omega\ D \wedge M \subseteq D\})$

lemma *dynkin-system-dynkin*:
assumes $M \subseteq$ *Pow* (Ω)
shows *dynkin-system* Ω (*dynkin* Ω M)
⟨*proof*⟩

lemma *dynkin-Basic*[*intro*]: $A \in M \Longrightarrow A \in$ *dynkin* Ω M
⟨*proof*⟩

lemma (**in** *dynkin-system*) *restricted-dynkin-system*:
assumes $D \in M$
shows *dynkin-system* Ω $\{Q.\ Q \subseteq \Omega \wedge Q \cap D \in M\}$
⟨*proof*⟩

lemma (**in** *dynkin-system*) *dynkin-subset*:
assumes $N \subseteq M$
shows *dynkin* Ω $N \subseteq M$
⟨*proof*⟩

lemma *sigma-eq-dynkin*:
  assumes *sets*: $M \subseteq Pow\ \Omega$
  assumes *Int-stable M*
  shows *sigma-sets* Ω M = *dynkin* Ω M
⟨*proof*⟩

lemma (**in** *dynkin-system*) *dynkin-idem*:
  dynkin Ω $M = M$
⟨*proof*⟩

lemma (**in** *dynkin-system*) *dynkin-lemma*:
  assumes *Int-stable E*
  and *E*: $E \subseteq M$ $M \subseteq$ *sigma-sets* Ω E
  shows *sigma-sets* Ω $E = M$
⟨*proof*⟩

2.1.12 Induction rule for intersection-stable generators

The reason to introduce Dynkin-systems is the following induction rules for σ-algebras generated by a generator closed under intersection.

lemma *sigma-sets-induct-disjoint*[*consumes 3*, *case-names basic empty compl union*]:
  assumes *Int-stable G*
    and *closed*: $G \subseteq Pow\ \Omega$
    and *A*: $A \in$ *sigma-sets* Ω G
  assumes *basic*: $\bigwedge A.\ A \in G \Longrightarrow P\ A$
    and *empty*: $P\ \{\}$
    and *compl*: $\bigwedge A.\ A \in$ *sigma-sets* Ω $G \Longrightarrow P\ A \Longrightarrow P\ (\Omega - A)$
    and *union*: $\bigwedge A.$ *disjoint-family* $A \Longrightarrow$ *range* $A \subseteq$ *sigma-sets* Ω $G \Longrightarrow (\bigwedge i.\ P$
$(A\ i)) \Longrightarrow P\ (\bigcup i{::}nat.\ A\ i)$
  shows *P A*
⟨*proof*⟩

2.2 Measure type

definition *positive* :: *'a set set* ⇒ (*'a set* ⇒ *ennreal*) ⇒ *bool* **where**
  positive M $\mu \longleftrightarrow \mu\ \{\} = 0$

definition *countably-additive* :: *'a set set* ⇒ (*'a set* ⇒ *ennreal*) ⇒ *bool* **where**
  countably-additive M f $\longleftrightarrow (\forall A.$ *range* $A \subseteq M \longrightarrow$ *disjoint-family* $A \longrightarrow (\bigcup i.$
$A\ i) \in M \longrightarrow$
    $(\sum i.\ f\ (A\ i)) = f\ (\bigcup i.\ A\ i))$

definition *measure-space* :: *'a set* ⇒ *'a set set* ⇒ (*'a set* ⇒ *ennreal*) ⇒ *bool* **where**
  measure-space Ω A $\mu \longleftrightarrow$ *sigma-algebra* Ω A $\wedge$ *positive* A μ $\wedge$ *countably-additive*
A μ

typedef *'a measure* = $\{(\Omega{::}'a\ set,\ A,\ \mu).\ (\forall a{\in}{-}A.\ \mu\ a = 0) \wedge$ *measure-space* Ω
A μ }

⟨*proof*⟩

definition *space* :: *'a measure* ⇒ *'a set* **where**
space M = *fst* (*Rep-measure M*)

definition *sets* :: *'a measure* ⇒ *'a set set* **where**
sets M = *fst* (*snd* (*Rep-measure M*))

definition *emeasure* :: *'a measure* ⇒ *'a set* ⇒ *ennreal* **where**
emeasure M = *snd* (*snd* (*Rep-measure M*))

definition *measure* :: *'a measure* ⇒ *'a set* ⇒ *real* **where**
measure M A = *enn2real* (*emeasure M A*)

declare [[*coercion sets*]]

declare [[*coercion measure*]]

declare [[*coercion emeasure*]]

lemma *measure-space*: *measure-space* (*space M*) (*sets M*) (*emeasure M*)
⟨*proof*⟩

interpretation *sets*: *sigma-algebra space M sets M* **for** *M* :: *'a measure*
⟨*proof*⟩

definition *measure-of* :: *'a set* ⇒ *'a set set* ⇒ (*'a set* ⇒ *ennreal*) ⇒ *'a measure*
where
measure-of Ω *A* μ = *Abs-measure* (Ω, *if A* ⊆ *Pow* Ω *then sigma-sets* Ω *A else* {{}, Ω},
λ*a*. *if a* ∈ *sigma-sets* Ω *A* ∧ *measure-space* Ω (*sigma-sets* Ω *A*) μ *then* μ *a else* *0*)

abbreviation *sigma* Ω *A* ≡ *measure-of* Ω *A* (λ*x*. *0*)

lemma *measure-space-0*: *A* ⊆ *Pow* Ω ⟹ *measure-space* Ω (*sigma-sets* Ω *A*) (λ*x*. *0*)
⟨*proof*⟩

lemma *sigma-algebra-trivial*: *sigma-algebra* Ω {{}, Ω}
⟨*proof*⟩

lemma *measure-space-0'*: *measure-space* Ω {{}, Ω} (λ*x*. *0*)
⟨*proof*⟩

lemma *measure-space-closed*:
assumes *measure-space* Ω *M* μ
shows *M* ⊆ *Pow* Ω
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *positive-cong-eq*:
$(\bigwedge a.\ a \in M \Longrightarrow \mu'\ a = \mu\ a) \Longrightarrow$ *positive* $M\ \mu' =$ *positive* $M\ \mu$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *countably-additive-eq*:
$(\bigwedge a.\ a \in M \Longrightarrow \mu'\ a = \mu\ a) \Longrightarrow$ *countably-additive* $M\ \mu' =$ *countably-additive* $M\ \mu$
⟨*proof*⟩

lemma *measure-space-eq*:
assumes *closed*: $A \subseteq$ *Pow* Ω **and** *eq*: $\bigwedge a.\ a \in$ *sigma-sets* $\Omega\ A \Longrightarrow \mu\ a = \mu'\ a$
shows *measure-space* Ω (*sigma-sets* $\Omega\ A$) $\mu =$ *measure-space* Ω (*sigma-sets* Ω A) μ'
⟨*proof*⟩

lemma *measure-of-eq*:
assumes *closed*: $A \subseteq$ *Pow* Ω **and** *eq*: $(\bigwedge a.\ a \in$ *sigma-sets* $\Omega\ A \Longrightarrow \mu\ a = \mu'\ a)$
shows *measure-of* $\Omega\ A\ \mu =$ *measure-of* $\Omega\ A\ \mu'$
⟨*proof*⟩

lemma
shows *space-measure-of-conv*: *space* (*measure-of* $\Omega\ A\ \mu$) $= \Omega$ (**is** *?space*)
and *sets-measure-of-conv*:
sets (*measure-of* $\Omega\ A\ \mu$) = (*if* $A \subseteq$ *Pow* Ω *then sigma-sets* $\Omega\ A$ *else* $\{\{\}, \Omega\}$) (**is** *?sets*)
and *emeasure-measure-of-conv*:
emeasure (*measure-of* $\Omega\ A\ \mu$) =
(λB. *if* $B \in$ *sigma-sets* $\Omega\ A\ \wedge$ *measure-space* Ω (*sigma-sets* $\Omega\ A$) μ *then* $\mu\ B$ *else* *0*) (**is** *?emeasure*)
⟨*proof*⟩

lemma [*simp*]:
assumes *A*: $A \subseteq$ *Pow* Ω
shows *sets-measure-of*: *sets* (*measure-of* $\Omega\ A\ \mu$) = *sigma-sets* $\Omega\ A$
and *space-measure-of*: *space* (*measure-of* $\Omega\ A\ \mu$) $= \Omega$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *sets-measure-of-eq*[*simp*]: *sets* (*measure-of* $\Omega\ M\ \mu$) = M
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *space-measure-of-eq*[*simp*]: *space* (*measure-of* $\Omega\ M\ \mu$) $= \Omega$
⟨*proof*⟩

lemma *measure-of-subset*: $M \subseteq$ *Pow* $\Omega \Longrightarrow M' \subseteq M \Longrightarrow$ *sets* (*measure-of* $\Omega\ M'$ μ) $\subseteq$ *sets* (*measure-of* $\Omega\ M\ \mu'$)
⟨*proof*⟩

lemma *emeasure-sigma*: *emeasure* (*sigma* Ω *A*) = (λx. *0*)
⟨*proof*⟩

lemma *sigma-sets-mono''*:
assumes $A \in$ *sigma-sets C D*
assumes $B \subseteq D$
assumes $D \subseteq$ *Pow C*
shows *sigma-sets A B* $\subseteq$ *sigma-sets C D*
⟨*proof*⟩

lemma *in-measure-of*[*intro*, *simp*]: $M \subseteq$ *Pow* $\Omega \Longrightarrow A \in M \Longrightarrow A \in$ *sets* (*measure-of* Ω *M* μ)
⟨*proof*⟩

lemma *space-empty-iff*: *space N* = {} $\longleftrightarrow$ *sets N* = {{}}
⟨*proof*⟩

2.2.1 Constructing simple *'a measure*

lemma *emeasure-measure-of*:
assumes *M*: *M* = *measure-of* Ω *A* μ
assumes *ms*: $A \subseteq$ *Pow* Ω *positive* (*sets M*) μ *countably-additive* (*sets M*) μ
assumes *X*: $X \in$ *sets M*
shows *emeasure M X* = μ *X*
⟨*proof*⟩

lemma *emeasure-measure-of-sigma*:
assumes *ms*: *sigma-algebra* Ω *M positive M* μ *countably-additive M* μ
assumes *A*: $A \in M$
shows *emeasure* (*measure-of* Ω *M* μ) *A* = μ *A*
⟨*proof*⟩

lemma *measure-cases*[*cases type*: *measure*]:
obtains (*measure*) Ω *A* μ **where** *x* = *Abs-measure* (Ω, *A*, μ) $\forall a \in -A$. μ *a* = *0* *measure-space* Ω *A* μ
⟨*proof*⟩

lemma *sets-le-imp-space-le*: *sets A* $\subseteq$ *sets B* $\Longrightarrow$ *space A* $\subseteq$ *space B*
⟨*proof*⟩

lemma *sets-eq-imp-space-eq*: *sets M* = *sets M'* $\Longrightarrow$ *space M* = *space M'*
⟨*proof*⟩

lemma *emeasure-notin-sets*: $A \notin$ *sets M* $\Longrightarrow$ *emeasure M A* = *0*
⟨*proof*⟩

lemma *emeasure-neq-0-sets*: *emeasure M A* $\neq$ *0* $\Longrightarrow A \in$ *sets M*
⟨*proof*⟩

lemma *measure-notin-sets*: $A \notin$ *sets* $M \Longrightarrow$ *measure* $M\ A = 0$
⟨*proof*⟩

lemma *measure-eqI*:
fixes $M\ N$ :: *'a measure*
assumes *sets* $M =$ *sets* N **and** *eq*: $\bigwedge A.\ A \in$ *sets* $M \Longrightarrow$ *emeasure* $M\ A =$ *emeasure* $N\ A$
shows $M = N$
⟨*proof*⟩

lemma *sigma-eqI*:
assumes [*simp*]: $M \subseteq$ *Pow* Ω $N \subseteq$ *Pow* Ω *sigma-sets* $\Omega\ M =$ *sigma-sets* $\Omega\ N$
shows *sigma* $\Omega\ M =$ *sigma* $\Omega\ N$
⟨*proof*⟩

2.2.2 Measurable functions

definition *measurable* :: *'a measure* $\Rightarrow$ *'b measure* $\Rightarrow$ (*'a* $\Rightarrow$ *'b*) *set* (**infixr** $\rightarrow_M$ *60*) **where**
measurable $A\ B = \{f \in$ *space* $A \rightarrow$ *space* $B.\ \forall y \in$ *sets* $B.\ f$ −' $y \cap$ *space* $A \in$ *sets* $A\}$

lemma *measurableI*:
$(\bigwedge x.\ x \in$ *space* $M \Longrightarrow f\,x \in$ *space* $N) \Longrightarrow (\bigwedge A.\ A \in$ *sets* $N \Longrightarrow f$ −' $A \cap$ *space* $M \in$ *sets* $M) \Longrightarrow$
$f \in$ *measurable* $M\ N$
⟨*proof*⟩

lemma *measurable-space*:
$f \in$ *measurable* $M\ A \Longrightarrow x \in$ *space* $M \Longrightarrow f\,x \in$ *space* A
⟨*proof*⟩

lemma *measurable-sets*:
$f \in$ *measurable* $M\ A \Longrightarrow S \in$ *sets* $A \Longrightarrow f$ −' $S \cap$ *space* $M \in$ *sets* M
⟨*proof*⟩

lemma *measurable-sets-Collect*:
assumes f: $f \in$ *measurable* $M\ N$ **and** P: $\{x \in$ *space* $N.\ P\ x\} \in$ *sets* N **shows** $\{x \in$ *space* $M.\ P\ (f\,x)\} \in$ *sets* M
⟨*proof*⟩

lemma *measurable-sigma-sets*:
assumes B: *sets* $N =$ *sigma-sets* $\Omega\ A$ $A \subseteq$ *Pow* Ω
and f: $f \in$ *space* $M \rightarrow \Omega$
and *ba*: $\bigwedge y.\ y \in A \Longrightarrow (f$ −' $y) \cap$ *space* $M \in$ *sets* M
shows $f \in$ *measurable* $M\ N$
⟨*proof*⟩

lemma *measurable-measure-of*:
assumes B: $N \subseteq Pow\ \Omega$
and f: $f \in space\ M \to \Omega$
and ba: $\bigwedge y.\ y \in N \Longrightarrow (f -`\ y) \cap space\ M \in sets\ M$
shows $f \in measurable\ M\ (measure\text{-}of\ \Omega\ N\ \mu)$
⟨*proof*⟩

lemma *measurable-iff-measure-of*:
assumes $N \subseteq Pow\ \Omega\ f \in space\ M \to \Omega$
shows $f \in measurable\ M\ (measure\text{-}of\ \Omega\ N\ \mu) \longleftrightarrow (\forall A \in N.\ f -`\ A \cap space\ M \in sets\ M)$
⟨*proof*⟩

lemma *measurable-cong-sets*:
assumes *sets*: $sets\ M = sets\ M'\ sets\ N = sets\ N'$
shows $measurable\ M\ N = measurable\ M'\ N'$
⟨*proof*⟩

lemma *measurable-cong*:
assumes $\bigwedge w.\ w \in space\ M \Longrightarrow f\ w = g\ w$
shows $f \in measurable\ M\ M' \longleftrightarrow g \in measurable\ M\ M'$
⟨*proof*⟩

lemma *measurable-cong'*:
assumes $\bigwedge w.\ w \in space\ M$ *=simp=>* $f\ w = g\ w$
shows $f \in measurable\ M\ M' \longleftrightarrow g \in measurable\ M\ M'$
⟨*proof*⟩

lemma *measurable-cong-strong*:
$M = N \Longrightarrow M' = N' \Longrightarrow (\bigwedge w.\ w \in space\ M \Longrightarrow f\ w = g\ w) \Longrightarrow$
$f \in measurable\ M\ M' \longleftrightarrow g \in measurable\ N\ N'$
⟨*proof*⟩

lemma *measurable-compose*:
assumes f: $f \in measurable\ M\ N$ **and** g: $g \in measurable\ N\ L$
shows $(\lambda x.\ g\ (f\ x)) \in measurable\ M\ L$
⟨*proof*⟩

lemma *measurable-comp*:
$f \in measurable\ M\ N \Longrightarrow g \in measurable\ N\ L \Longrightarrow g \circ f \in measurable\ M\ L$
⟨*proof*⟩

lemma *measurable-const*:
$c \in space\ M' \Longrightarrow (\lambda x.\ c) \in measurable\ M\ M'$
⟨*proof*⟩

lemma *measurable-ident*: $id \in measurable\ M\ M$
⟨*proof*⟩

lemma *measurable-id*: $(\lambda x.\ x) \in measurable\ M\ M$
⟨*proof*⟩

lemma *measurable-ident-sets*:
assumes *eq*: $sets\ M = sets\ M'$ **shows** $(\lambda x.\ x) \in measurable\ M\ M'$
⟨*proof*⟩

lemma *sets-Least*:
assumes *meas*: $\bigwedge i{::}nat.\ \{x{\in}space\ M.\ P\ i\ x\} \in M$
shows $(\lambda x.\ LEAST\ j.\ P\ j\ x)$ −‘ $A \cap space\ M \in sets\ M$
⟨*proof*⟩

lemma *measurable-mono1*:
$M' \subseteq Pow\ \Omega \Longrightarrow M \subseteq M' \Longrightarrow$
$measurable\ (measure\text{-}of\ \Omega\ M\ \mu)\ N \subseteq measurable\ (measure\text{-}of\ \Omega\ M'\ \mu')\ N$
⟨*proof*⟩

2.2.3 Counting space

definition *count-space* :: *'a set* ⇒ *'a measure* **where**
$count\text{-}space\ \Omega = measure\text{-}of\ \Omega\ (Pow\ \Omega)\ (\lambda A.\ if\ finite\ A\ then\ of\text{-}nat\ (card\ A)\ else\ \infty)$

lemma
shows *space-count-space*[*simp*]: $space\ (count\text{-}space\ \Omega) = \Omega$
and *sets-count-space*[*simp*]: $sets\ (count\text{-}space\ \Omega) = Pow\ \Omega$
⟨*proof*⟩

lemma *measurable-count-space-eq1*[*simp*]:
$f \in measurable\ (count\text{-}space\ A)\ M \longleftrightarrow f \in A \to space\ M$
⟨*proof*⟩

lemma *measurable-compose-countable'*:
assumes *f*: $\bigwedge i.\ i \in I \Longrightarrow (\lambda x.\ f\ i\ x) \in measurable\ M\ N$
and *g*: $g \in measurable\ M\ (count\text{-}space\ I)$ **and** *I*: *countable I*
shows $(\lambda x.\ f\ (g\ x)\ x) \in measurable\ M\ N$
⟨*proof*⟩

lemma *measurable-count-space-eq-countable*:
assumes *countable A*
shows $f \in measurable\ M\ (count\text{-}space\ A) \longleftrightarrow (f \in space\ M \to A \wedge (\forall a{\in}A.\ f$ −‘ $\{a\} \cap space\ M \in sets\ M))$
⟨*proof*⟩

lemma *measurable-count-space-eq2*:
$finite\ A \Longrightarrow f \in measurable\ M\ (count\text{-}space\ A) \longleftrightarrow (f \in space\ M \to A \wedge (\forall a{\in}A.\ f$ −‘ $\{a\} \cap space\ M \in sets\ M))$
⟨*proof*⟩

lemma *measurable-count-space-eq2-countable*:
fixes f :: $'a$ => $'c$::*countable*
shows $f \in$ *measurable* M (*count-space* A) $\longleftrightarrow$ ($f \in$ *space* $M \to A \wedge (\forall a{\in}A.\ f$ $-`\ \{a\} \cap$ *space* $M \in$ *sets* M))
⟨*proof*⟩

lemma *measurable-compose-countable*:
assumes f: $\bigwedge i$::$'i$::*countable*. ($\lambda x.\ f\ i\ x) \in$ *measurable* $M\ N$ **and** g: $g \in$ *measurable* M (*count-space UNIV*)
shows ($\lambda x.\ f\ (g\ x)\ x) \in$ *measurable* $M\ N$
⟨*proof*⟩

lemma *measurable-count-space-const*:
($\lambda x.\ c) \in$ *measurable* M (*count-space UNIV*)
⟨*proof*⟩

lemma *measurable-count-space*:
$f \in$ *measurable* (*count-space* A) (*count-space UNIV*)
⟨*proof*⟩

lemma *measurable-compose-rev*:
assumes f: $f \in$ *measurable* $L\ N$ **and** g: $g \in$ *measurable* $M\ L$
shows ($\lambda x.\ f\ (g\ x)) \in$ *measurable* $M\ N$
⟨*proof*⟩

lemma *measurable-empty-iff*:
space $N = \{\} \Longrightarrow f \in$ *measurable* $M\ N \longleftrightarrow$ *space* $M = \{\}$
⟨*proof*⟩

2.2.4 Extend measure

definition *extend-measure* $\Omega\ I\ G\ \mu$ =
(*if* ($\exists \mu'.\ (\forall i{\in}I.\ \mu'\ (G\ i) = \mu\ i) \wedge$ *measure-space* Ω (*sigma-sets* Ω ($G`I$)) $\mu') \wedge$ $\neg (\forall i{\in}I.\ \mu\ i = 0)$
then measure-of Ω ($G`I$) (*SOME* $\mu'.\ (\forall i{\in}I.\ \mu'\ (G\ i) = \mu\ i) \wedge$ *measure-space* $\Omega$ (*sigma-sets* $\Omega$ ($G`I$)) μ')
else measure-of Ω ($G`I$) (λ-. *0*))

lemma *space-extend-measure*: $G\ `\ I \subseteq Pow\ \Omega \Longrightarrow$ *space* (*extend-measure* $\Omega\ I\ G\ \mu) = \Omega$
⟨*proof*⟩

lemma *sets-extend-measure*: $G\ `\ I \subseteq Pow\ \Omega \Longrightarrow$ *sets* (*extend-measure* $\Omega\ I\ G\ \mu$) = *sigma-sets* $\Omega$ ($G`I$)
⟨*proof*⟩

lemma *emeasure-extend-measure*:
assumes M: M = *extend-measure* $\Omega\ I\ G\ \mu$
and *eq*: $\bigwedge i.\ i \in I \Longrightarrow \mu'\ (G\ i) = \mu\ i$

and *ms*: G ' $I \subseteq Pow\ \Omega$ *positive* (*sets* M) μ' *countably-additive* (*sets* M) μ'
and $i \in I$
shows *emeasure* M ($G\ i$) = $\mu\ i$
⟨*proof*⟩

lemma *emeasure-extend-measure-Pair*:
assumes M: M = *extend-measure* Ω {(i, j). $I\ i\ j$} (λ(i, j). $G\ i\ j$) (λ(i, j). $\mu\ i\ j$)
and *eq*: $\bigwedge i\ j.\ I\ i\ j \Longrightarrow \mu'\ (G\ i\ j) = \mu\ i\ j$
and *ms*: $\bigwedge i\ j.\ I\ i\ j \Longrightarrow G\ i\ j \in Pow\ \Omega$ *positive* (*sets* M) μ' *countably-additive* (*sets* M) μ'
and $I\ i\ j$
shows *emeasure* M ($G\ i\ j$) = $\mu\ i\ j$
⟨*proof*⟩

2.2.5 Supremum of a set of σ-algebras

definition *Sup-sigma* M = *sigma* ($\bigcup x{\in}M$. *space* x) ($\bigcup x{\in}M$. *sets* x)

syntax
-SUP-sigma :: *pttrn* ⇒ *'a set* ⇒ *'b* ⇒ *'b* ((*3*$\bigsqcup_\sigma$ -∈-./ -) [*0*, *0*, *10*] *10*)

translations
$\bigsqcup_\sigma$ $x{\in}A$. B == *CONST Sup-sigma* ((λx. B) ' A)

lemma *space-Sup-sigma*: *space* (*Sup-sigma* M) = ($\bigcup x{\in}M$. *space* x)
⟨*proof*⟩

lemma *sets-Sup-sigma*: *sets* (*Sup-sigma* M) = *sigma-sets* ($\bigcup x{\in}M$. *space* x) ($\bigcup x{\in}M$. *sets* x)
⟨*proof*⟩

lemma *in-Sup-sigma*: $m \in M \Longrightarrow A \in$ *sets* $m \Longrightarrow A \in$ *sets* (*Sup-sigma* M)
⟨*proof*⟩

lemma *SUP-sigma-cong*:
assumes $*$: $\bigwedge i.\ i \in I \Longrightarrow$ *sets* ($M\ i$) = *sets* ($N\ i$) **shows** *sets* ($\bigsqcup_\sigma$ $i{\in}I$. $M\ i$) = *sets* ($\bigsqcup_\sigma$ $i{\in}I$. $N\ i$)
⟨*proof*⟩

lemma *sets-Sup-in-sets*:
assumes $M \neq \{\}$
assumes $\bigwedge m.\ m \in M \Longrightarrow$ *space* m = *space* N
assumes $\bigwedge m.\ m \in M \Longrightarrow$ *sets* $m \subseteq$ *sets* N
shows *sets* (*Sup-sigma* M) $\subseteq$ *sets* N
⟨*proof*⟩

lemma *measurable-Sup-sigma1*:
assumes m: $m \in M$ **and** f: $f \in$ *measurable* m N

and *const-space*: $\bigwedge m\ n.\ m \in M \Longrightarrow n \in M \Longrightarrow space\ m = space\ n$
shows $f \in measurable\ (Sup\text{-}sigma\ M)\ N$
⟨*proof*⟩

lemma *measurable-Sup-sigma2*:
assumes M: $M \neq \{\}$
assumes f: $\bigwedge m.\ m \in M \Longrightarrow f \in measurable\ N\ m$
shows $f \in measurable\ N\ (Sup\text{-}sigma\ M)$
⟨*proof*⟩

lemma *Sup-sigma-sigma*:
assumes [*simp*]: $M \neq \{\}$ **and** M: $\bigwedge m.\ m \in M \Longrightarrow m \subseteq Pow\ \Omega$
shows $(\bigsqcup_\sigma\ m{\in}M.\ sigma\ \Omega\ m) = sigma\ \Omega\ (\bigcup M)$
⟨*proof*⟩

lemma *SUP-sigma-sigma*:
assumes M: $M \neq \{\}\ \bigwedge m.\ m \in M \Longrightarrow f\ m \subseteq Pow\ \Omega$
shows $(\bigsqcup_\sigma\ m{\in}M.\ sigma\ \Omega\ (f\ m)) = sigma\ \Omega\ (\bigcup m{\in}M.\ f\ m)$
⟨*proof*⟩

2.3 The smallest σ-algebra regarding a function

definition
$vimage\text{-}algebra\ X\ f\ M = sigma\ X\ \{f\ -`\ A \cap X \mid A.\ A \in sets\ M\}$

lemma *space-vimage-algebra*[*simp*]: $space\ (vimage\text{-}algebra\ X\ f\ M) = X$
⟨*proof*⟩

lemma *sets-vimage-algebra*: $sets\ (vimage\text{-}algebra\ X\ f\ M) = sigma\text{-}sets\ X\ \{f\ -`\ A \cap X \mid A.\ A \in sets\ M\}$
⟨*proof*⟩

lemma *sets-vimage-algebra2*:
$f \in X \to space\ M \Longrightarrow sets\ (vimage\text{-}algebra\ X\ f\ M) = \{f\ -`\ A \cap X \mid A.\ A \in sets\ M\}$
⟨*proof*⟩

lemma *sets-vimage-algebra-cong*: $sets\ M = sets\ N \Longrightarrow sets\ (vimage\text{-}algebra\ X\ f\ M) = sets\ (vimage\text{-}algebra\ X\ f\ N)$
⟨*proof*⟩

lemma *vimage-algebra-cong*:
assumes $X = Y$
assumes $\bigwedge x.\ x \in Y \Longrightarrow f\ x = g\ x$
assumes $sets\ M = sets\ N$
shows $vimage\text{-}algebra\ X\ f\ M = vimage\text{-}algebra\ Y\ g\ N$
⟨*proof*⟩

lemma *in-vimage-algebra*: $A \in sets\ M \Longrightarrow f\ -`\ A \cap X \in sets\ (vimage\text{-}algebra\ X$

f M)
⟨*proof*⟩

lemma *sets-image-in-sets*:
assumes *N*: *space N* = *X*
assumes *f*: *f* ∈ *measurable N M*
shows *sets* (*vimage-algebra X f M*) ⊆ *sets N*
⟨*proof*⟩

lemma *measurable-vimage-algebra1*: *f* ∈ *X* → *space M* ⟹ *f* ∈ *measurable* (*vimage-algebra X f M*) *M*
⟨*proof*⟩

lemma *measurable-vimage-algebra2*:
assumes *g*: *g* ∈ *space N* → *X* **and** *f*: (λ*x*. *f* (*g x*)) ∈ *measurable N M*
shows *g* ∈ *measurable N* (*vimage-algebra X f M*)
⟨*proof*⟩

lemma *vimage-algebra-sigma*:
assumes *X*: *X* ⊆ *Pow* Ω′ **and** *f*: *f* ∈ Ω → Ω′
shows *vimage-algebra* Ω *f* (*sigma* Ω′ *X*) = *sigma* Ω {*f* −‘ *A* ∩ Ω | *A*. *A* ∈ *X* }
(**is** *?V* = *?S*)
⟨*proof*⟩

lemma *vimage-algebra-vimage-algebra-eq*:
assumes ∗: *f* ∈ *X* → *Y g* ∈ *Y* → *space M*
shows *vimage-algebra X f* (*vimage-algebra Y g M*) = *vimage-algebra X* (λ*x*. *g* (*f x*)) *M*
(**is** *?VV* = *?V*)
⟨*proof*⟩

lemma *sets-vimage-Sup-eq*:
assumes ∗: *M* ≠ {} ⋀*m*. *m* ∈ *M* ⟹ *f* ∈ *X* → *space m*
shows *sets* (*vimage-algebra X f* (*Sup-sigma M*)) = *sets* (⨆$_\sigma$ *m* ∈ *M*. *vimage-algebra X f m*)
(**is** *?IS* = *?SI*)
⟨*proof*⟩

lemma *vimage-algebra-Sup-sigma*:
assumes [*simp*]: *MM* ≠ {} **and** ⋀*M*. *M* ∈ *MM* ⟹ *f* ∈ *X* → *space M*
shows *vimage-algebra X f* (*Sup-sigma MM*) = *Sup-sigma* (*vimage-algebra X f* ‘ *MM*)
⟨*proof*⟩

2.3.1 Restricted Space Sigma Algebra

definition *restrict-space* **where**
restrict-space M Ω = *measure-of* (Ω ∩ *space M*) ((*op* ∩ Ω) ‘ *sets M*) (*emeasure M*)

lemma *space-restrict-space*: *space* (*restrict-space M* Ω) = Ω ∩ *space M*
⟨*proof*⟩

lemma *space-restrict-space2*: Ω ∈ *sets M* ⟹ *space* (*restrict-space M* Ω) = Ω
⟨*proof*⟩

lemma *sets-restrict-space*: *sets* (*restrict-space M* Ω) = (*op* ∩ Ω) ‘ *sets M*
⟨*proof*⟩

lemma *restrict-space-sets-cong*:
A = B ⟹ *sets M* = *sets N* ⟹ *sets* (*restrict-space M A*) = *sets* (*restrict-space N B*)
⟨*proof*⟩

lemma *sets-restrict-space-count-space* :
sets (*restrict-space* (*count-space A*) *B*) = *sets* (*count-space* (A ∩ B))
⟨*proof*⟩

lemma *sets-restrict-UNIV*[*simp*]: *sets* (*restrict-space M UNIV*) = *sets M*
⟨*proof*⟩

lemma *sets-restrict-restrict-space*:
sets (*restrict-space* (*restrict-space M A*) *B*) = *sets* (*restrict-space M* (A ∩ B))
⟨*proof*⟩

lemma *sets-restrict-space-iff*:
Ω ∩ *space M* ∈ *sets M* ⟹ A ∈ *sets* (*restrict-space M* Ω) ⟷ (A ⊆ Ω ∧ A ∈ *sets M*)
⟨*proof*⟩

lemma *sets-restrict-space-cong*: *sets M* = *sets N* ⟹ *sets* (*restrict-space M* Ω) = *sets* (*restrict-space N* Ω)
⟨*proof*⟩

lemma *restrict-space-eq-vimage-algebra*:
Ω ⊆ *space M* ⟹ *sets* (*restrict-space M* Ω) = *sets* (*vimage-algebra* Ω (λx. x) *M*)
⟨*proof*⟩

lemma *sets-Collect-restrict-space-iff*:
assumes S ∈ *sets M*
shows {x∈*space* (*restrict-space M S*). P x} ∈ *sets* (*restrict-space M S*) ⟷ {x∈*space M*. x ∈ S ∧ P x} ∈ *sets M*
⟨*proof*⟩

lemma *measurable-restrict-space1*:
assumes *f*: *f* ∈ *measurable M N*
shows *f* ∈ *measurable* (*restrict-space M* Ω) *N*
⟨*proof*⟩

lemma *measurable-restrict-space2-iff*:
$f \in$ *measurable* M (*restrict-space* N Ω) $\longleftrightarrow$ ($f \in$ *measurable* M N $\wedge$ $f \in$ *space* $M \to \Omega$)
⟨*proof*⟩

lemma *measurable-restrict-space2*:
$f \in$ *space* $M \to \Omega \Longrightarrow f \in$ *measurable* M $N \Longrightarrow f \in$ *measurable* M (*restrict-space* N Ω)
⟨*proof*⟩

lemma *measurable-piecewise-restrict*:
assumes I: *countable* C
and X: $\bigwedge\Omega.\ \Omega \in C \Longrightarrow \Omega \cap$ *space* $M \in$ *sets* M *space* $M \subseteq \bigcup C$
and f: $\bigwedge\Omega.\ \Omega \in C \Longrightarrow f \in$ *measurable* (*restrict-space* M Ω) N
shows $f \in$ *measurable* M N
⟨*proof*⟩

lemma *measurable-piecewise-restrict-iff*:
countable $C \Longrightarrow (\bigwedge\Omega.\ \Omega \in C \Longrightarrow \Omega \cap$ *space* $M \in$ *sets* $M) \Longrightarrow$ *space* $M \subseteq (\bigcup C) \Longrightarrow$
$f \in$ *measurable* M $N \longleftrightarrow (\forall\,\Omega{\in}C.\ f \in$ *measurable* (*restrict-space* M Ω) $N)$
⟨*proof*⟩

lemma *measurable-If-restrict-space-iff*:
$\{x{\in}$*space* $M.\ P\ x\} \in$ *sets* $M \Longrightarrow$
$(\lambda x.$ *if* $P\ x$ *then* $f\ x$ *else* $g\ x) \in$ *measurable* M $N \longleftrightarrow$
($f \in$ *measurable* (*restrict-space* M $\{x.\ P\ x\}$) $N \wedge g \in$ *measurable* (*restrict-space* M $\{x.\ \neg\ P\ x\}$) N)
⟨*proof*⟩

lemma *measurable-If*:
$f \in$ *measurable* M $M' \Longrightarrow g \in$ *measurable* M $M' \Longrightarrow \{x{\in}$*space* $M.\ P\ x\} \in$ *sets* $M \Longrightarrow$
$(\lambda x.$ *if* $P\ x$ *then* $f\ x$ *else* $g\ x) \in$ *measurable* M M'
⟨*proof*⟩

lemma *measurable-If-set*:
assumes *measure*: $f \in$ *measurable* M M' $g \in$ *measurable* M M'
assumes P: $A \cap$ *space* $M \in$ *sets* M
shows $(\lambda x.$ *if* $x \in A$ *then* $f\ x$ *else* $g\ x) \in$ *measurable* M M'
⟨*proof*⟩

lemma *measurable-restrict-space-iff*:
$\Omega \cap$ *space* $M \in$ *sets* $M \Longrightarrow c \in$ *space* $N \Longrightarrow$
$f \in$ *measurable* (*restrict-space* M Ω) $N \longleftrightarrow (\lambda x.$ *if* $x \in \Omega$ *then* $f\ x$ *else* $c) \in$ *measurable* M N
⟨*proof*⟩

lemma *restrict-space-singleton*: $\{x\} \in sets\ M \Longrightarrow sets\ (restrict\text{-}space\ M\ \{x\}) = sets\ (count\text{-}space\ \{x\})$
⟨*proof*⟩

lemma *measurable-restrict-countable*:
assumes *X*[*intro*]: *countable X*
assumes *sets*[*simp*]: $\bigwedge x.\ x \in X \Longrightarrow \{x\} \in sets\ M$
assumes *space*[*simp*]: $\bigwedge x.\ x \in X \Longrightarrow f\ x \in space\ N$
assumes *f*: $f \in measurable\ (restrict\text{-}space\ M\ (-\ X))\ N$
shows $f \in measurable\ M\ N$
⟨*proof*⟩

lemma *measurable-discrete-difference*:
assumes *f*: $f \in measurable\ M\ N$
assumes *X*: *countable X* $\bigwedge x.\ x \in X \Longrightarrow \{x\} \in sets\ M$ $\bigwedge x.\ x \in X \Longrightarrow g\ x \in space\ N$
assumes *eq*: $\bigwedge x.\ x \in space\ M \Longrightarrow x \notin X \Longrightarrow f\ x = g\ x$
shows $g \in measurable\ M\ N$
⟨*proof*⟩

end

theory *Measurable*
imports
Sigma-Algebra
~~/src/HOL/Library/Order-Continuity
begin

2.4 Measurability prover

lemma (**in** *algebra*) *sets-Collect-finite-All*:
assumes $\bigwedge i.\ i \in S \Longrightarrow \{x{\in}\Omega.\ P\ i\ x\} \in M$ *finite S*
shows $\{x{\in}\Omega.\ \forall\, i{\in}S.\ P\ i\ x\} \in M$
⟨*proof*⟩

abbreviation $pred\ M\ P \equiv P \in measurable\ M\ (count\text{-}space\ (UNIV::bool\ set))$

lemma *pred-def*: $pred\ M\ P \longleftrightarrow \{x{\in}space\ M.\ P\ x\} \in sets\ M$
⟨*proof*⟩

lemma *pred-sets1*: $\{x{\in}space\ M.\ P\ x\} \in sets\ M \Longrightarrow f \in measurable\ N\ M \Longrightarrow pred\ N\ (\lambda x.\ P\ (f\ x))$
⟨*proof*⟩

lemma *pred-sets2*: $A \in sets\ N \Longrightarrow f \in measurable\ M\ N \Longrightarrow pred\ M\ (\lambda x.\ f\ x \in A)$
⟨*proof*⟩

⟨*ML*⟩

declare
pred-sets1 [*measurable-dest*]
pred-sets2 [*measurable-dest*]
sets.sets-into-space[*measurable-dest*]

declare
sets.top[*measurable*]
sets.empty-sets[*measurable* (*raw*)]
sets.Un[*measurable* (*raw*)]
sets.Diff [*measurable* (*raw*)]

declare
measurable-count-space[*measurable* (*raw*)]
measurable-ident[*measurable* (*raw*)]
measurable-id[*measurable* (*raw*)]
measurable-const[*measurable* (*raw*)]
measurable-If [*measurable* (*raw*)]
measurable-comp[*measurable* (*raw*)]
measurable-sets[*measurable* (*raw*)]

declare *measurable-cong-sets*[*measurable-cong*]
declare *sets-restrict-space-cong*[*measurable-cong*]
declare *sets-restrict-UNIV* [*measurable-cong*]

lemma *predE*[*measurable* (*raw*)]:
$pred\ M\ P \Longrightarrow \{x{\in}space\ M.\ P\ x\} \in sets\ M$
⟨*proof*⟩

lemma *pred-intros-imp'*[*measurable* (*raw*)]:
$(K \Longrightarrow pred\ M\ (\lambda x.\ P\ x)) \Longrightarrow pred\ M\ (\lambda x.\ K \longrightarrow P\ x)$
⟨*proof*⟩

lemma *pred-intros-conj1'*[*measurable* (*raw*)]:
$(K \Longrightarrow pred\ M\ (\lambda x.\ P\ x)) \Longrightarrow pred\ M\ (\lambda x.\ K \wedge P\ x)$
⟨*proof*⟩

lemma *pred-intros-conj2'*[*measurable* (*raw*)]:
$(K \Longrightarrow pred\ M\ (\lambda x.\ P\ x)) \Longrightarrow pred\ M\ (\lambda x.\ P\ x \wedge K)$
⟨*proof*⟩

lemma *pred-intros-disj1'*[*measurable* (*raw*)]:
$(\neg\ K \Longrightarrow pred\ M\ (\lambda x.\ P\ x)) \Longrightarrow pred\ M\ (\lambda x.\ K \vee P\ x)$
⟨*proof*⟩

lemma *pred-intros-disj2'*[*measurable* (*raw*)]:
$(\neg\ K \Longrightarrow pred\ M\ (\lambda x.\ P\ x)) \Longrightarrow pred\ M\ (\lambda x.\ P\ x \vee K)$
⟨*proof*⟩

lemma *pred-intros-logic*[*measurable* (*raw*)]:
  $pred\ M\ (\lambda x.\ x \in space\ M)$
  $pred\ M\ (\lambda x.\ P\ x) \Longrightarrow pred\ M\ (\lambda x.\ \neg\ P\ x)$
  $pred\ M\ (\lambda x.\ Q\ x) \Longrightarrow pred\ M\ (\lambda x.\ P\ x) \Longrightarrow pred\ M\ (\lambda x.\ Q\ x \wedge P\ x)$
  $pred\ M\ (\lambda x.\ Q\ x) \Longrightarrow pred\ M\ (\lambda x.\ P\ x) \Longrightarrow pred\ M\ (\lambda x.\ Q\ x \longrightarrow P\ x)$
  $pred\ M\ (\lambda x.\ Q\ x) \Longrightarrow pred\ M\ (\lambda x.\ P\ x) \Longrightarrow pred\ M\ (\lambda x.\ Q\ x \vee P\ x)$
  $pred\ M\ (\lambda x.\ Q\ x) \Longrightarrow pred\ M\ (\lambda x.\ P\ x) \Longrightarrow pred\ M\ (\lambda x.\ Q\ x = P\ x)$
  $pred\ M\ (\lambda x.\ f\,x \in UNIV)$
  $pred\ M\ (\lambda x.\ f\,x \in \{\})$
  $pred\ M\ (\lambda x.\ P'\ (f\,x)\ x) \Longrightarrow pred\ M\ (\lambda x.\ f\,x \in \{y.\ P'\ y\ x\})$
  $pred\ M\ (\lambda x.\ f\,x \in (B\ x)) \Longrightarrow pred\ M\ (\lambda x.\ f\,x \in -\ (B\ x))$
  $pred\ M\ (\lambda x.\ f\,x \in (A\ x)) \Longrightarrow pred\ M\ (\lambda x.\ f\,x \in (B\ x)) \Longrightarrow pred\ M\ (\lambda x.\ f\,x \in (A\ x) - (B\ x))$
  $pred\ M\ (\lambda x.\ f\,x \in (A\ x)) \Longrightarrow pred\ M\ (\lambda x.\ f\,x \in (B\ x)) \Longrightarrow pred\ M\ (\lambda x.\ f\,x \in (A\ x) \cap (B\ x))$
  $pred\ M\ (\lambda x.\ f\,x \in (A\ x)) \Longrightarrow pred\ M\ (\lambda x.\ f\,x \in (B\ x)) \Longrightarrow pred\ M\ (\lambda x.\ f\,x \in (A\ x) \cup (B\ x))$
  $pred\ M\ (\lambda x.\ g\ x\ (f\,x) \in (X\ x)) \Longrightarrow pred\ M\ (\lambda x.\ f\,x \in (g\ x) -`\ (X\ x))$
  ⟨*proof*⟩

lemma *pred-intros-countable*[*measurable* (*raw*)]:
  fixes $P :: 'a \Rightarrow 'i :: countable \Rightarrow bool$
  shows
    $(\bigwedge i.\ pred\ M\ (\lambda x.\ P\ x\ i)) \Longrightarrow pred\ M\ (\lambda x.\ \forall\, i.\ P\ x\ i)$
    $(\bigwedge i.\ pred\ M\ (\lambda x.\ P\ x\ i)) \Longrightarrow pred\ M\ (\lambda x.\ \exists\, i.\ P\ x\ i)$
  ⟨*proof*⟩

lemma *pred-intros-countable-bounded*[*measurable* (*raw*)]:
  fixes $X :: 'i :: countable\ set$
  shows
    $(\bigwedge i.\ i \in X \Longrightarrow pred\ M\ (\lambda x.\ x \in N\,x\,i)) \Longrightarrow pred\ M\ (\lambda x.\ x \in (\bigcap i{\in}X.\ N\,x\,i))$
    $(\bigwedge i.\ i \in X \Longrightarrow pred\ M\ (\lambda x.\ x \in N\,x\,i)) \Longrightarrow pred\ M\ (\lambda x.\ x \in (\bigcup i{\in}X.\ N\,x\,i))$
    $(\bigwedge i.\ i \in X \Longrightarrow pred\ M\ (\lambda x.\ P\ x\ i)) \Longrightarrow pred\ M\ (\lambda x.\ \forall\, i{\in}X.\ P\ x\ i)$
    $(\bigwedge i.\ i \in X \Longrightarrow pred\ M\ (\lambda x.\ P\ x\ i)) \Longrightarrow pred\ M\ (\lambda x.\ \exists\, i{\in}X.\ P\ x\ i)$
  ⟨*proof*⟩

lemma *pred-intros-finite*[*measurable* (*raw*)]:
  $finite\ I \Longrightarrow (\bigwedge i.\ i \in I \Longrightarrow pred\ M\ (\lambda x.\ x \in N\,x\,i)) \Longrightarrow pred\ M\ (\lambda x.\ x \in (\bigcap i{\in}I.\ N\,x\,i))$
  $finite\ I \Longrightarrow (\bigwedge i.\ i \in I \Longrightarrow pred\ M\ (\lambda x.\ x \in N\,x\,i)) \Longrightarrow pred\ M\ (\lambda x.\ x \in (\bigcup i{\in}I.\ N\,x\,i))$
  $finite\ I \Longrightarrow (\bigwedge i.\ i \in I \Longrightarrow pred\ M\ (\lambda x.\ P\ x\ i)) \Longrightarrow pred\ M\ (\lambda x.\ \forall\, i{\in}I.\ P\ x\ i)$
  $finite\ I \Longrightarrow (\bigwedge i.\ i \in I \Longrightarrow pred\ M\ (\lambda x.\ P\ x\ i)) \Longrightarrow pred\ M\ (\lambda x.\ \exists\, i{\in}I.\ P\ x\ i)$
  ⟨*proof*⟩

lemma *countable-Un-Int*[*measurable* (*raw*)]:
  $(\bigwedge i :: 'i :: countable.\ i \in I \Longrightarrow N\ i \in sets\ M) \Longrightarrow (\bigcup i{\in}I.\ N\ i) \in sets\ M$
  $I \neq \{\} \Longrightarrow (\bigwedge i :: 'i :: countable.\ i \in I \Longrightarrow N\ i \in sets\ M) \Longrightarrow (\bigcap i{\in}I.\ N\ i) \in sets\ M$

⟨*proof*⟩

declare
finite-UN[*measurable* (*raw*)]
finite-INT[*measurable* (*raw*)]

lemma *sets-Int-pred*[*measurable* (*raw*)]:
assumes *space*: $A \cap B \subseteq$ *space M* **and** [*measurable*]: *pred M* $(\lambda x.\ x \in A)$ *pred M* $(\lambda x.\ x \in B)$
shows $A \cap B \in$ *sets M*
⟨*proof*⟩

lemma [*measurable* (*raw generic*)]:
assumes *f*: $f \in$ *measurable M N* **and** *c*: $c \in$ *space N* $\Longrightarrow \{c\} \in$ *sets N*
shows *pred-eq-const1*: *pred M* $(\lambda x.\ f\ x = c)$
and *pred-eq-const2*: *pred M* $(\lambda x.\ c = f\ x)$
⟨*proof*⟩

lemma *pred-count-space-const1*[*measurable* (*raw*)]:
$f \in$ *measurable M* (*count-space UNIV*) $\Longrightarrow$ *Measurable.pred M* $(\lambda x.\ f\ x = c)$
⟨*proof*⟩

lemma *pred-count-space-const2*[*measurable* (*raw*)]:
$f \in$ *measurable M* (*count-space UNIV*) $\Longrightarrow$ *Measurable.pred M* $(\lambda x.\ c = f\ x)$
⟨*proof*⟩

lemma *pred-le-const*[*measurable* (*raw generic*)]:
assumes *f*: $f \in$ *measurable M N* **and** *c*: $\{..\ c\} \in$ *sets N* **shows** *pred M* $(\lambda x.\ f\ x \le c)$
⟨*proof*⟩

lemma *pred-const-le*[*measurable* (*raw generic*)]:
assumes *f*: $f \in$ *measurable M N* **and** *c*: $\{c\ ..\} \in$ *sets N* **shows** *pred M* $(\lambda x.\ c \le f\ x)$
⟨*proof*⟩

lemma *pred-less-const*[*measurable* (*raw generic*)]:
assumes *f*: $f \in$ *measurable M N* **and** *c*: $\{..< c\} \in$ *sets N* **shows** *pred M* $(\lambda x.\ f\ x < c)$
⟨*proof*⟩

lemma *pred-const-less*[*measurable* (*raw generic*)]:
assumes *f*: $f \in$ *measurable M N* **and** *c*: $\{c <..\} \in$ *sets N* **shows** *pred M* $(\lambda x.\ c < f\ x)$
⟨*proof*⟩

declare
sets.Int[*measurable* (*raw*)]

lemma *pred-in-If*[*measurable* (*raw*)]:
$(P \Longrightarrow pred\ M\ (\lambda x.\ x \in A\ x)) \Longrightarrow (\neg\ P \Longrightarrow pred\ M\ (\lambda x.\ x \in B\ x)) \Longrightarrow$
$pred\ M\ (\lambda x.\ x \in (if\ P\ then\ A\ x\ else\ B\ x))$
⟨*proof*⟩

lemma *sets-range*[*measurable-dest*]:
$A\ `\ I \subseteq sets\ M \Longrightarrow i \in I \Longrightarrow A\ i \in sets\ M$
⟨*proof*⟩

lemma *pred-sets-range*[*measurable-dest*]:
$A\ `\ I \subseteq sets\ N \Longrightarrow i \in I \Longrightarrow f \in measurable\ M\ N \Longrightarrow pred\ M\ (\lambda x.\ f\ x \in A\ i)$
⟨*proof*⟩

lemma *sets-All*[*measurable-dest*]:
$\forall i.\ A\ i \in sets\ (M\ i) \Longrightarrow A\ i \in sets\ (M\ i)$
⟨*proof*⟩

lemma *pred-sets-All*[*measurable-dest*]:
$\forall i.\ A\ i \in sets\ (N\ i) \Longrightarrow f \in measurable\ M\ (N\ i) \Longrightarrow pred\ M\ (\lambda x.\ f\ x \in A\ i)$
⟨*proof*⟩

lemma *sets-Ball*[*measurable-dest*]:
$\forall i{\in}I.\ A\ i \in sets\ (M\ i) \Longrightarrow i{\in}I \Longrightarrow A\ i \in sets\ (M\ i)$
⟨*proof*⟩

lemma *pred-sets-Ball*[*measurable-dest*]:
$\forall i{\in}I.\ A\ i \in sets\ (N\ i) \Longrightarrow i{\in}I \Longrightarrow f \in measurable\ M\ (N\ i) \Longrightarrow pred\ M\ (\lambda x.\ f$
$x \in A\ i)$
⟨*proof*⟩

lemma *measurable-finite*[*measurable* (*raw*)]:
fixes $S :: 'a \Rightarrow nat\ set$
assumes [*measurable*]: $\bigwedge i.\ \{x{\in}space\ M.\ i \in S\ x\} \in sets\ M$
shows $pred\ M\ (\lambda x.\ finite\ (S\ x))$
⟨*proof*⟩

lemma *measurable-Least*[*measurable*]:
assumes [*measurable*]: $(\bigwedge i{::}nat.\ (\lambda x.\ P\ i\ x) \in measurable\ M\ (count\text{-}space\ UNIV))q$
shows $(\lambda x.\ LEAST\ i.\ P\ i\ x) \in measurable\ M\ (count\text{-}space\ UNIV)$
⟨*proof*⟩

lemma *measurable-Max-nat*[*measurable* (*raw*)]:
fixes $P :: nat \Rightarrow 'a \Rightarrow bool$
assumes [*measurable*]: $\bigwedge i.\ Measurable.pred\ M\ (P\ i)$
shows $(\lambda x.\ Max\ \{i.\ P\ i\ x\}) \in measurable\ M\ (count\text{-}space\ UNIV)$
⟨*proof*⟩

lemma *measurable-Min-nat*[*measurable* (*raw*)]:
fixes $P :: nat \Rightarrow 'a \Rightarrow bool$

assumes [*measurable*]: $\bigwedge i.$ *Measurable.pred M* (*P i*)
shows ($\lambda x.$ *Min* {*i. P i x*}) $\in$ *measurable M* (*count-space UNIV*)
⟨*proof*⟩

lemma *measurable-count-space-insert*[*measurable* (*raw*)]:
$s \in S \Longrightarrow A \in$ *sets* (*count-space S*) $\Longrightarrow$ *insert s A* $\in$ *sets* (*count-space S*)
⟨*proof*⟩

lemma *sets-UNIV* [*measurable* (*raw*)]: $A \in$ *sets* (*count-space UNIV*)
⟨*proof*⟩

lemma *measurable-card*[*measurable*]:
fixes S :: $'a \Rightarrow$ *nat set*
assumes [*measurable*]: $\bigwedge i.$ {$x \in$*space M*. $i \in S\ x$} $\in$ *sets M*
shows ($\lambda x.$ *card* ($S\ x$)) $\in$ *measurable M* (*count-space UNIV*)
⟨*proof*⟩

lemma *measurable-pred-countable*[*measurable* (*raw*)]:
assumes *countable X*
shows
($\bigwedge i.\ i \in X \Longrightarrow$ *Measurable.pred M* ($\lambda x.$ *P x i*)) $\Longrightarrow$ *Measurable.pred M* ($\lambda x.$ $\forall i \in X.$ *P x i*)
($\bigwedge i.\ i \in X \Longrightarrow$ *Measurable.pred M* ($\lambda x.$ *P x i*)) $\Longrightarrow$ *Measurable.pred M* ($\lambda x.$ $\exists i \in X.$ *P x i*)
⟨*proof*⟩

2.5 Measurability for (co)inductive predicates

lemma *measurable-bot*[*measurable*]: *bot* $\in$ *measurable M* (*count-space UNIV*)
⟨*proof*⟩

lemma *measurable-top*[*measurable*]: *top* $\in$ *measurable M* (*count-space UNIV*)
⟨*proof*⟩

lemma *measurable-SUP*[*measurable*]:
fixes F :: $'i \Rightarrow 'a \Rightarrow 'b$::{*complete-lattice*, *countable*}
assumes [*simp*]: *countable I*
assumes [*measurable*]: $\bigwedge i.\ i \in I \Longrightarrow F\ i \in$ *measurable M* (*count-space UNIV*)
shows ($\lambda x.$ *SUP i*:*I*. *F i x*) $\in$ *measurable M* (*count-space UNIV*)
⟨*proof*⟩

lemma *measurable-INF*[*measurable*]:
fixes F :: $'i \Rightarrow 'a \Rightarrow 'b$::{*complete-lattice*, *countable*}
assumes [*simp*]: *countable I*
assumes [*measurable*]: $\bigwedge i.\ i \in I \Longrightarrow F\ i \in$ *measurable M* (*count-space UNIV*)
shows ($\lambda x.$ *INF i*:*I*. *F i x*) $\in$ *measurable M* (*count-space UNIV*)
⟨*proof*⟩

lemma *measurable-lfp-coinduct*[*consumes 1*, *case-names continuity step*]:

fixes F :: $('a \Rightarrow 'b) \Rightarrow ('a \Rightarrow 'b$::{*complete-lattice*, *countable*})
assumes $P\ M$
assumes F: *sup-continuous* F
assumes $*$: $\bigwedge M\ A.\ P\ M \Longrightarrow (\bigwedge N.\ P\ N \Longrightarrow A \in$ *measurable* N (*count-space* *UNIV*)) $\Longrightarrow F\ A \in$ *measurable* M (*count-space* *UNIV*)
shows *lfp* $F \in$ *measurable* M (*count-space* *UNIV*)
⟨*proof*⟩

lemma *measurable-lfp*:
fixes F :: $('a \Rightarrow 'b) \Rightarrow ('a \Rightarrow 'b$::{*complete-lattice*, *countable*})
assumes F: *sup-continuous* F
assumes $*$: $\bigwedge A.\ A \in$ *measurable* M (*count-space* *UNIV*) $\Longrightarrow F\ A \in$ *measurable* M (*count-space* *UNIV*)
shows *lfp* $F \in$ *measurable* M (*count-space* *UNIV*)
⟨*proof*⟩

lemma *measurable-gfp-coinduct*[*consumes 1*, *case-names continuity step*]:
fixes F :: $('a \Rightarrow 'b) \Rightarrow ('a \Rightarrow 'b$::{*complete-lattice*, *countable*})
assumes $P\ M$
assumes F: *inf-continuous* F
assumes $*$: $\bigwedge M\ A.\ P\ M \Longrightarrow (\bigwedge N.\ P\ N \Longrightarrow A \in$ *measurable* N (*count-space* *UNIV*)) $\Longrightarrow F\ A \in$ *measurable* M (*count-space* *UNIV*)
shows *gfp* $F \in$ *measurable* M (*count-space* *UNIV*)
⟨*proof*⟩

lemma *measurable-gfp*:
fixes F :: $('a \Rightarrow 'b) \Rightarrow ('a \Rightarrow 'b$::{*complete-lattice*, *countable*})
assumes F: *inf-continuous* F
assumes $*$: $\bigwedge A.\ A \in$ *measurable* M (*count-space* *UNIV*) $\Longrightarrow F\ A \in$ *measurable* M (*count-space* *UNIV*)
shows *gfp* $F \in$ *measurable* M (*count-space* *UNIV*)
⟨*proof*⟩

lemma *measurable-lfp2-coinduct*[*consumes 1*, *case-names continuity step*]:
fixes F :: $('a \Rightarrow 'c \Rightarrow 'b) \Rightarrow ('a \Rightarrow 'c \Rightarrow 'b$::{*complete-lattice*, *countable*})
assumes $P\ M\ s$
assumes F: *sup-continuous* F
assumes $*$: $\bigwedge M\ A\ s.\ P\ M\ s \Longrightarrow (\bigwedge N\ t.\ P\ N\ t \Longrightarrow A\ t \in$ *measurable* N (*count-space* *UNIV*)) $\Longrightarrow F\ A\ s \in$ *measurable* M (*count-space* *UNIV*)
shows *lfp* $F\ s \in$ *measurable* M (*count-space* *UNIV*)
⟨*proof*⟩

lemma *measurable-gfp2-coinduct*[*consumes 1*, *case-names continuity step*]:
fixes F :: $('a \Rightarrow 'c \Rightarrow 'b) \Rightarrow ('a \Rightarrow 'c \Rightarrow 'b$::{*complete-lattice*, *countable*})
assumes $P\ M\ s$
assumes F: *inf-continuous* F
assumes $*$: $\bigwedge M\ A\ s.\ P\ M\ s \Longrightarrow (\bigwedge N\ t.\ P\ N\ t \Longrightarrow A\ t \in$ *measurable* N (*count-space* *UNIV*)) $\Longrightarrow F\ A\ s \in$ *measurable* M (*count-space* *UNIV*)
shows *gfp* $F\ s \in$ *measurable* M (*count-space* *UNIV*)

⟨*proof*⟩

lemma *measurable-enat-coinduct*:
fixes f :: $'a \Rightarrow enat$
assumes $R\ f$
assumes ∗: $\bigwedge f.\ R\ f \Longrightarrow \exists\, g\ h\ i\ P.\ R\ g \land f = (\lambda x.\ \mathit{if}\ P\ x\ \mathit{then}\ h\ x\ \mathit{else}\ eSuc\ (g\ (i\ x))) \land$
$Measurable.pred\ M\ P \land$
$i \in measurable\ M\ M \land$
$h \in measurable\ M\ (count\text{-}space\ UNIV)$
shows $f \in measurable\ M\ (count\text{-}space\ UNIV)$
⟨*proof*⟩

lemma *measurable-THE*:
fixes P :: $'a \Rightarrow 'b \Rightarrow bool$
assumes [*measurable*]: $\bigwedge i.\ Measurable.pred\ M\ (P\ i)$
assumes I[*simp*]: $countable\ I\ \bigwedge i\ x.\ x \in space\ M \Longrightarrow P\ i\ x \Longrightarrow i \in I$
assumes *unique*: $\bigwedge x\ i\ j.\ x \in space\ M \Longrightarrow P\ i\ x \Longrightarrow P\ j\ x \Longrightarrow i = j$
shows $(\lambda x.\ THE\ i.\ P\ i\ x) \in measurable\ M\ (count\text{-}space\ UNIV)$
⟨*proof*⟩

lemma *measurable-Ex1*[*measurable* (*raw*)]:
assumes [*simp*]: *countable I* **and** [*measurable*]: $\bigwedge i.\ i \in I \Longrightarrow Measurable.pred\ M\ (P\ i)$
shows $Measurable.pred\ M\ (\lambda x.\ \exists!i{\in}I.\ P\ i\ x)$
⟨*proof*⟩

lemma *measurable-Sup-nat*[*measurable* (*raw*)]:
fixes F :: $'a \Rightarrow nat\ set$
assumes [*measurable*]: $\bigwedge i.\ Measurable.pred\ M\ (\lambda x.\ i \in F\ x)$
shows $(\lambda x.\ Sup\ (F\ x)) \in M \rightarrow_M count\text{-}space\ UNIV$
⟨*proof*⟩

lemma *measurable-if-split*[*measurable* (*raw*)]:
$(c \Longrightarrow Measurable.pred\ M\ f) \Longrightarrow (\neg\ c \Longrightarrow Measurable.pred\ M\ g) \Longrightarrow$
$Measurable.pred\ M\ (\mathit{if}\ c\ \mathit{then}\ f\ \mathit{else}\ g)$
⟨*proof*⟩

lemma *pred-restrict-space*:
assumes $S \in sets\ M$
shows $Measurable.pred\ (restrict\text{-}space\ M\ S)\ P \longleftrightarrow Measurable.pred\ M\ (\lambda x.\ x \in S \land P\ x)$
⟨*proof*⟩

lemma *measurable-predpow*[*measurable*]:
assumes $Measurable.pred\ M\ T$
assumes $\bigwedge Q.\ Measurable.pred\ M\ Q \Longrightarrow Measurable.pred\ M\ (R\ Q)$
shows $Measurable.pred\ M\ ((R\ \hat{}\hat{}\ n)\ T)$
⟨*proof*⟩

hide-const (**open**) *pred*

end

3 Measure spaces and their properties

theory *Measure-Space*
imports
Measurable ~~/*src*/*HOL*/*Multivariate-Analysis*/*Multivariate-Analysis*
begin

3.1 Relate extended reals and the indicator function

lemma *suminf-cmult-indicator*:
fixes f :: *nat* ⇒ *ennreal*
assumes *disjoint-family* A $x \in A$ i
shows $(\sum n.\ f\ n * indicator\ (A\ n)\ x) = f\ i$
⟨*proof*⟩

lemma *suminf-indicator*:
assumes *disjoint-family* A
shows $(\sum n.\ indicator\ (A\ n)\ x :: ennreal) = indicator\ (\bigcup i.\ A\ i)\ x$
⟨*proof*⟩

lemma *setsum-indicator-disjoint-family*:
fixes f :: $'d \Rightarrow\ 'e$::*semiring-1*
assumes *d*: *disjoint-family-on* A P **and** $x \in A\ j$ **and** *finite* P **and** $j \in P$
shows $(\sum i{\in}P.\ f\ i * indicator\ (A\ i)\ x) = f\ j$
⟨*proof*⟩

The type for emeasure spaces is already defined in *Sigma-Algebra*, as it is also used to represent sigma algebras (with an arbitrary emeasure).

3.2 Extend binary sets

lemma *LIMSEQ-binaryset*:
assumes f: $f\ \{\} = 0$
shows $(\lambda n.\ \sum i{<}n.\ f\ (binaryset\ A\ B\ i)) \longrightarrow f\ A + f\ B$
⟨*proof*⟩

lemma *binaryset-sums*:
assumes f: $f\ \{\} = 0$
shows $(\lambda n.\ f\ (binaryset\ A\ B\ n))$ *sums* $(f\ A + f\ B)$
⟨*proof*⟩

lemma *suminf-binaryset-eq*:
fixes f :: $'a\ set \Rightarrow\ 'b$::{*comm-monoid-add*, *t2-space*}
shows $f\ \{\} = 0 \Longrightarrow (\sum n.\ f\ (binaryset\ A\ B\ n)) = f\ A + f\ B$

⟨*proof*⟩

3.3 Properties of a premeasure μ

The definitions for *positive* and *countably-additive* should be here, by they are necessary to define *'a measure* in *Sigma-Algebra*.

definition *subadditive* **where**
subadditive M f $\longleftrightarrow$ $(\forall x{\in}M.\ \forall y{\in}M.\ x \cap y = \{\} \longrightarrow f\ (x \cup y) \le f\ x + f\ y)$

lemma *subadditiveD*: *subadditive M f* $\Longrightarrow x \cap y = \{\} \Longrightarrow x \in M \Longrightarrow y \in M \Longrightarrow f\ (x \cup y) \le f\ x + f\ y$
⟨*proof*⟩

definition *countably-subadditive* **where**
countably-subadditive M f $\longleftrightarrow$
$(\forall A.\ range\ A \subseteq M \longrightarrow disjoint\text{-}family\ A \longrightarrow (\bigcup i.\ A\ i) \in M \longrightarrow (f\ (\bigcup i.\ A\ i) \le (\sum i.\ f\ (A\ i))))$

lemma (**in** *ring-of-sets*) *countably-subadditive-subadditive*:
fixes f :: *'a set* $\Rightarrow$ *ennreal*
assumes *f*: *positive M f* **and** *cs*: *countably-subadditive M f*
shows *subadditive M f*
⟨*proof*⟩

definition *additive* **where**
additive M $\mu \longleftrightarrow (\forall x{\in}M.\ \forall y{\in}M.\ x \cap y = \{\} \longrightarrow \mu\ (x \cup y) = \mu\ x + \mu\ y)$

definition *increasing* **where**
increasing M $\mu \longleftrightarrow (\forall x{\in}M.\ \forall y{\in}M.\ x \subseteq y \longrightarrow \mu\ x \le \mu\ y)$

lemma *positiveD1*: *positive M f* $\Longrightarrow f\ \{\} = 0$ ⟨*proof*⟩

lemma *positiveD-empty*:
positive M f $\Longrightarrow f\ \{\} = 0$
⟨*proof*⟩

lemma *additiveD*:
additive M f $\Longrightarrow x \cap y = \{\} \Longrightarrow x \in M \Longrightarrow y \in M \Longrightarrow f\ (x \cup y) = f\ x + f\ y$
⟨*proof*⟩

lemma *increasingD*:
increasing M f $\Longrightarrow x \subseteq y \Longrightarrow x{\in}M \Longrightarrow y{\in}M \Longrightarrow f\ x \le f\ y$
⟨*proof*⟩

lemma *countably-additiveI*[*case-names countably*]:
$(\bigwedge A.\ range\ A \subseteq M \Longrightarrow disjoint\text{-}family\ A \Longrightarrow (\bigcup i.\ A\ i) \in M \Longrightarrow (\sum i.\ f\ (A\ i)) = f\ (\bigcup i.\ A\ i))$
$\Longrightarrow$ *countably-additive M f*
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *disjointed-additive*:
  assumes *f*: *positive M f additive M f* **and** *A*: *range A* $\subseteq$ *M incseq A*
  shows $(\sum i \le n.\ f\ (disjointed\ A\ i)) = f\ (A\ n)$
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *additive-sum*:
  fixes *A*:: *'i* ⇒ *'a set*
  assumes *f*: *positive M f* **and** *ad*: *additive M f* **and** *finite S*
    and *A*: *A'S* $\subseteq$ *M*
    and *disj*: *disjoint-family-on A S*
  shows $(\sum i \in S.\ f\ (A\ i)) = f\ (\bigcup i \in S.\ A\ i)$
  ⟨*proof*⟩

lemma (**in** *ring-of-sets*) *additive-increasing*:
  fixes *f* :: *'a set* ⇒ *ennreal*
  assumes *posf*: *positive M f* **and** *addf*: *additive M f*
  shows *increasing M f*
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *subadditive*:
  fixes *f* :: *'a set* ⇒ *ennreal*
  assumes *f*: *positive M f additive M f* **and** *A*: *A'S* $\subseteq$ *M* **and** *S*: *finite S*
  shows $f\ (\bigcup i \in S.\ A\ i) \le (\sum i \in S.\ f\ (A\ i))$
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *countably-additive-additive*:
  fixes *f* :: *'a set* ⇒ *ennreal*
  assumes *posf*: *positive M f* **and** *ca*: *countably-additive M f*
  shows *additive M f*
⟨*proof*⟩

lemma (**in** *algebra*) *increasing-additive-bound*:
  fixes *A*:: *nat* ⇒ *'a set* **and** *f* :: *'a set* ⇒ *ennreal*
  assumes *f*: *positive M f* **and** *ad*: *additive M f*
    and *inc*: *increasing M f*
    and *A*: *range A* $\subseteq$ *M*
    and *disj*: *disjoint-family A*
  shows $(\sum i.\ f\ (A\ i)) \le f\ \Omega$
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *countably-additiveI-finite*:
  fixes μ :: *'a set* ⇒ *ennreal*
  assumes *finite* Ω *positive M* μ *additive M* μ
  shows *countably-additive M* μ
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *countably-additive-iff-continuous-from-below*:
  fixes *f* :: *'a set* ⇒ *ennreal*

assumes f: *positive M f additive M f*
shows *countably-additive M f* $\longleftrightarrow$
$(\forall A.\ range\ A \subseteq M \longrightarrow incseq\ A \longrightarrow (\bigcup i.\ A\ i) \in M \longrightarrow (\lambda i.\ f\ (A\ i)) \longlonglongrightarrow f\ (\bigcup i.\ A\ i))$
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *continuous-from-above-iff-empty-continuous*:
fixes f :: *'a set* $\Rightarrow$ *ennreal*
assumes f: *positive M f additive M f*
shows $(\forall A.\ range\ A \subseteq M \longrightarrow decseq\ A \longrightarrow (\bigcap i.\ A\ i) \in M \longrightarrow (\forall i.\ f\ (A\ i) \neq \infty) \longrightarrow (\lambda i.\ f\ (A\ i)) \longlonglongrightarrow f\ (\bigcap i.\ A\ i))$
$\longleftrightarrow (\forall A.\ range\ A \subseteq M \longrightarrow decseq\ A \longrightarrow (\bigcap i.\ A\ i) = \{\} \longrightarrow (\forall i.\ f\ (A\ i) \neq \infty) \longrightarrow (\lambda i.\ f\ (A\ i)) \longlonglongrightarrow 0)$
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *empty-continuous-imp-continuous-from-below*:
fixes f :: *'a set* $\Rightarrow$ *ennreal*
assumes f: *positive M f additive M f* $\forall A{\in}M.\ f\ A \neq \infty$
assumes *cont*: $\forall A.\ range\ A \subseteq M \longrightarrow decseq\ A \longrightarrow (\bigcap i.\ A\ i) = \{\} \longrightarrow (\lambda i.\ f\ (A\ i)) \longlonglongrightarrow 0$
assumes A: *range A* $\subseteq$ *M incseq A* $(\bigcup i.\ A\ i) \in M$
shows $(\lambda i.\ f\ (A\ i)) \longlonglongrightarrow f\ (\bigcup i.\ A\ i)$
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *empty-continuous-imp-countably-additive*:
fixes f :: *'a set* $\Rightarrow$ *ennreal*
assumes f: *positive M f additive M f* **and** *fin*: $\forall A{\in}M.\ f\ A \neq \infty$
assumes *cont*: $\bigwedge A.\ range\ A \subseteq M \Longrightarrow decseq\ A \Longrightarrow (\bigcap i.\ A\ i) = \{\} \Longrightarrow (\lambda i.\ f\ (A\ i)) \longlonglongrightarrow 0$
shows *countably-additive M f*
⟨*proof*⟩

3.4 Properties of *emeasure*

lemma *emeasure-positive*: *positive* (*sets M*) (*emeasure M*)
⟨*proof*⟩

lemma *emeasure-empty*[*simp*, *intro*]: *emeasure M* $\{\} = 0$
⟨*proof*⟩

lemma *emeasure-single-in-space*: *emeasure M* $\{x\} \neq 0 \Longrightarrow x \in$ *space M*
⟨*proof*⟩

lemma *emeasure-countably-additive*: *countably-additive* (*sets M*) (*emeasure M*)
⟨*proof*⟩

lemma *suminf-emeasure*:
range A $\subseteq$ *sets M* $\Longrightarrow$ *disjoint-family A* $\Longrightarrow (\sum i.\ emeasure\ M\ (A\ i)) = emeasure\ M\ (\bigcup i.\ A\ i)$

⟨*proof*⟩

lemma *sums-emeasure*:
$disjoint\text{-}family\ F \Longrightarrow (\bigwedge i.\ F\ i \in sets\ M) \Longrightarrow (\lambda i.\ emeasure\ M\ (F\ i))\ sums\ emeasure\ M\ (\bigcup i.\ F\ i)$
⟨*proof*⟩

lemma *emeasure-additive*: $additive\ (sets\ M)\ (emeasure\ M)$
⟨*proof*⟩

lemma *plus-emeasure*:
$a \in sets\ M \Longrightarrow b \in sets\ M \Longrightarrow a \cap b = \{\} \Longrightarrow emeasure\ M\ a + emeasure\ M\ b = emeasure\ M\ (a \cup b)$
⟨*proof*⟩

lemma *setsum-emeasure*:
$F`I \subseteq sets\ M \Longrightarrow disjoint\text{-}family\text{-}on\ F\ I \Longrightarrow finite\ I \Longrightarrow$
$(\sum i{\in}I.\ emeasure\ M\ (F\ i)) = emeasure\ M\ (\bigcup i{\in}I.\ F\ i)$
⟨*proof*⟩

lemma *emeasure-mono*:
$a \subseteq b \Longrightarrow b \in sets\ M \Longrightarrow emeasure\ M\ a \leq emeasure\ M\ b$
⟨*proof*⟩

lemma *emeasure-space*:
$emeasure\ M\ A \leq emeasure\ M\ (space\ M)$
⟨*proof*⟩

lemma *emeasure-Diff*:
assumes *finite*: $emeasure\ M\ B \neq \infty$
and [*measurable*]: $A \in sets\ M\ B \in sets\ M$ **and** $B \subseteq A$
shows $emeasure\ M\ (A - B) = emeasure\ M\ A - emeasure\ M\ B$
⟨*proof*⟩

lemma *emeasure-compl*:
$s \in sets\ M \Longrightarrow emeasure\ M\ s \neq \infty \Longrightarrow emeasure\ M\ (space\ M - s) = emeasure\ M\ (space\ M) - emeasure\ M\ s$
⟨*proof*⟩

lemma *Lim-emeasure-incseq*:
$range\ A \subseteq sets\ M \Longrightarrow incseq\ A \Longrightarrow (\lambda i.\ (emeasure\ M\ (A\ i))) \longrightarrow emeasure\ M\ (\bigcup i.\ A\ i)$
⟨*proof*⟩

lemma *incseq-emeasure*:
assumes $range\ B \subseteq sets\ M\ incseq\ B$
shows $incseq\ (\lambda i.\ emeasure\ M\ (B\ i))$
⟨*proof*⟩

lemma *SUP-emeasure-incseq*:
assumes *A*: *range A* $\subseteq$ *sets M incseq A*
shows (*SUP n. emeasure M* (*A n*)) = *emeasure M* ($\bigcup$*i. A i*)
⟨*proof*⟩

lemma *decseq-emeasure*:
assumes *range B* $\subseteq$ *sets M decseq B*
shows *decseq* (λ*i. emeasure M* (*B i*))
⟨*proof*⟩

lemma *INF-emeasure-decseq*:
assumes *A*: *range A* $\subseteq$ *sets M* **and** *decseq A*
and *finite*: $\bigwedge$*i. emeasure M* (*A i*) $\neq \infty$
shows (*INF n. emeasure M* (*A n*)) = *emeasure M* ($\bigcap$*i. A i*)
⟨*proof*⟩

lemma *emeasure-INT-decseq-subset*:
fixes *F* :: *nat* $\Rightarrow$ *'a set*
assumes *I*: *I* $\neq$ {} **and** *F*: $\bigwedge$*i j. i* $\in$ *I* $\Longrightarrow$ *j* $\in$ *I* $\Longrightarrow$ *i* $\leq$ *j* $\Longrightarrow$ *F j* $\subseteq$ *F i*
assumes *F-sets*[*measurable*]: $\bigwedge$*i. i* $\in$ *I* $\Longrightarrow$ *F i* $\in$ *sets M*
and *fin*: $\bigwedge$*i. i* $\in$ *I* $\Longrightarrow$ *emeasure M* (*F i*) $\neq \infty$
shows *emeasure M* ($\bigcap$*i*$\in$*I. F i*) = (*INF i*:*I. emeasure M* (*F i*))
⟨*proof*⟩

lemma *Lim-emeasure-decseq*:
assumes *A*: *range A* $\subseteq$ *sets M decseq A* **and** *fin*: $\bigwedge$*i. emeasure M* (*A i*) $\neq \infty$
shows (λ*i. emeasure M* (*A i*)) $\longrightarrow$ *emeasure M* ($\bigcap$*i. A i*)
⟨*proof*⟩

lemma *emeasure-lfp'*[*consumes 1*, *case-names cont measurable*]:
assumes *P M*
assumes *cont*: *sup-continuous F*
assumes $*$: $\bigwedge$*M A. P M* $\Longrightarrow$ ($\bigwedge$*N. P N* $\Longrightarrow$ *Measurable.pred N A*) $\Longrightarrow$ *Measurable.pred M* (*F A*)
shows *emeasure M* {*x*$\in$*space M. lfp F x*} = (*SUP i. emeasure M* {*x*$\in$*space M.* (*F ^^ i*) (λ*x. False*) *x*})
⟨*proof*⟩

lemma *emeasure-lfp*:
assumes [*simp*]: $\bigwedge$*s. sets* (*M s*) = *sets N*
assumes *cont*: *sup-continuous F sup-continuous f*
assumes *meas*: $\bigwedge$*P. Measurable.pred N P* $\Longrightarrow$ *Measurable.pred N* (*F P*)
assumes *iter*: $\bigwedge$*P s. Measurable.pred N P* $\Longrightarrow$ *P* $\leq$ *lfp F* $\Longrightarrow$ *emeasure* (*M s*) {*x*$\in$*space N. F P x*} = *f* (λ*s. emeasure* (*M s*) {*x*$\in$*space N. P x*}) *s*
shows *emeasure* (*M s*) {*x*$\in$*space N. lfp F x*} = *lfp f s*
⟨*proof*⟩

lemma *emeasure-subadditive-finite*:
finite I $\Longrightarrow$ *A ` I* $\subseteq$ *sets M* $\Longrightarrow$ *emeasure M* ($\bigcup$*i*$\in$*I. A i*) $\leq$ ($\sum$*i*$\in$*I. emeasure*

M (*A* *i*))
⟨*proof*⟩

lemma *emeasure-subadditive*:
$A \in sets\ M \Longrightarrow B \in sets\ M \Longrightarrow emeasure\ M\ (A \cup B) \leq emeasure\ M\ A + emeasure\ M\ B$
⟨*proof*⟩

lemma *emeasure-subadditive-countably*:
assumes $range\ f \subseteq sets\ M$
shows $emeasure\ M\ (\bigcup i.\ f\ i) \leq (\sum i.\ emeasure\ M\ (f\ i))$
⟨*proof*⟩

lemma *emeasure-insert*:
assumes *sets*: $\{x\} \in sets\ M\ A \in sets\ M$ **and** $x \notin A$
shows $emeasure\ M\ (insert\ x\ A) = emeasure\ M\ \{x\} + emeasure\ M\ A$
⟨*proof*⟩

lemma *emeasure-insert-ne*:
$A \neq \{\} \Longrightarrow \{x\} \in sets\ M \Longrightarrow A \in sets\ M \Longrightarrow x \notin A \Longrightarrow emeasure\ M\ (insert\ x\ A) = emeasure\ M\ \{x\} + emeasure\ M\ A$
⟨*proof*⟩

lemma *emeasure-eq-setsum-singleton*:
assumes $finite\ S\ \bigwedge x.\ x \in S \Longrightarrow \{x\} \in sets\ M$
shows $emeasure\ M\ S = (\sum x{\in}S.\ emeasure\ M\ \{x\})$
⟨*proof*⟩

lemma *setsum-emeasure-cover*:
assumes *finite S* **and** $A \in sets\ M$ **and** *br-in-M*: $B\ `\ S \subseteq sets\ M$
assumes *A*: $A \subseteq (\bigcup i{\in}S.\ B\ i)$
assumes *disj*: *disjoint-family-on B S*
shows $emeasure\ M\ A = (\sum i{\in}S.\ emeasure\ M\ (A \cap (B\ i)))$
⟨*proof*⟩

lemma *emeasure-eq-0*:
$N \in sets\ M \Longrightarrow emeasure\ M\ N = 0 \Longrightarrow K \subseteq N \Longrightarrow emeasure\ M\ K = 0$
⟨*proof*⟩

lemma *emeasure-UN-eq-0*:
assumes $\bigwedge i{::}nat.\ emeasure\ M\ (N\ i) = 0$ **and** $range\ N \subseteq sets\ M$
shows $emeasure\ M\ (\bigcup i.\ N\ i) = 0$
⟨*proof*⟩

lemma *measure-eqI-finite*:
assumes [*simp*]: $sets\ M = Pow\ A\ sets\ N = Pow\ A$ **and** *finite A*
assumes *eq*: $\bigwedge a.\ a \in A \Longrightarrow emeasure\ M\ \{a\} = emeasure\ N\ \{a\}$
shows $M = N$
⟨*proof*⟩

lemma *measure-eqI-generator-eq*:
fixes $M\ N :: 'a\ measure$ **and** $E :: 'a\ set\ set$ **and** $A :: nat \Rightarrow 'a\ set$
assumes *Int-stable* $E\ E \subseteq Pow\ \Omega$
and *eq*: $\bigwedge X.\ X \in E \Longrightarrow emeasure\ M\ X = emeasure\ N\ X$
and *M*: $sets\ M = sigma\text{-}sets\ \Omega\ E$
and *N*: $sets\ N = sigma\text{-}sets\ \Omega\ E$
and *A*: $range\ A \subseteq E\ (\bigcup i.\ A\ i) = \Omega\ \bigwedge i.\ emeasure\ M\ (A\ i) \neq \infty$
shows $M = N$
⟨*proof*⟩

lemma *measure-of-of-measure*: $measure\text{-}of\ (space\ M)\ (sets\ M)\ (emeasure\ M) = M$
⟨*proof*⟩

3.5 μ-null sets

definition *null-sets* $:: 'a\ measure \Rightarrow 'a\ set\ set$ **where**
$null\text{-}sets\ M = \{N \in sets\ M.\ emeasure\ M\ N = 0\}$

lemma *null-setsD1*[*dest*]: $A \in null\text{-}sets\ M \Longrightarrow emeasure\ M\ A = 0$
⟨*proof*⟩

lemma *null-setsD2*[*dest*]: $A \in null\text{-}sets\ M \Longrightarrow A \in sets\ M$
⟨*proof*⟩

lemma *null-setsI*[*intro*]: $emeasure\ M\ A = 0 \Longrightarrow A \in sets\ M \Longrightarrow A \in null\text{-}sets\ M$
⟨*proof*⟩

interpretation *null-sets*: *ring-of-sets space M null-sets M* **for** *M*
⟨*proof*⟩

lemma *UN-from-nat-into*:
assumes *I*: *countable* $I\ I \neq \{\}$
shows $(\bigcup i \in I.\ N\ i) = (\bigcup i.\ N\ (from\text{-}nat\text{-}into\ I\ i))$
⟨*proof*⟩

lemma *null-sets-UN'*:
assumes *countable I*
assumes $\bigwedge i.\ i \in I \Longrightarrow N\ i \in null\text{-}sets\ M$
shows $(\bigcup i \in I.\ N\ i) \in null\text{-}sets\ M$
⟨*proof*⟩

lemma *null-sets-UN*[*intro*]:
$(\bigwedge i{::}'i{::}countable.\ N\ i \in null\text{-}sets\ M) \Longrightarrow (\bigcup i.\ N\ i) \in null\text{-}sets\ M$
⟨*proof*⟩

lemma *null-set-Int1*:
assumes $B \in null\text{-}sets\ M\ A \in sets\ M$ **shows** $A \cap B \in null\text{-}sets\ M$

⟨*proof*⟩

lemma *null-set-Int2*:
assumes $B \in$ *null-sets* M $A \in$ *sets* M **shows** $B \cap A \in$ *null-sets* M
⟨*proof*⟩

lemma *emeasure-Diff-null-set*:
assumes $B \in$ *null-sets* M $A \in$ *sets* M
shows *emeasure* M $(A - B)$ = *emeasure* M A
⟨*proof*⟩

lemma *null-set-Diff*:
assumes $B \in$ *null-sets* M $A \in$ *sets* M **shows** $B - A \in$ *null-sets* M
⟨*proof*⟩

lemma *emeasure-Un-null-set*:
assumes $A \in$ *sets* M $B \in$ *null-sets* M
shows *emeasure* M $(A \cup B)$ = *emeasure* M A
⟨*proof*⟩

3.6 The almost everywhere filter (i.e. quantifier)

definition *ae-filter* :: *'a measure* ⇒ *'a filter* **where**
ae-filter M = (*INF* N:*null-sets* M. *principal* (*space* M − N))

abbreviation *almost-everywhere* :: *'a measure* ⇒ (*'a* ⇒ *bool*) ⇒ *bool* **where**
almost-everywhere M P ≡ *eventually* P (*ae-filter* M)

syntax
-almost-everywhere :: *pttrn* ⇒ *'a* ⇒ *bool* ⇒ *bool* (*AE* - *in* -. - [*0*,*0*,*10*] *10*)

translations
AE x in M. P ⇌ *CONST almost-everywhere M* (λ*x*. *P*)

lemma *eventually-ae-filter*: *eventually* P (*ae-filter* M) ⟷ (∃ N∈*null-sets* M. {x ∈ *space* M. ¬ P x} ⊆ N)
⟨*proof*⟩

lemma *AE-I'*:
N ∈ *null-sets* M ⟹ {x∈*space* M. ¬ P x} ⊆ N ⟹ (*AE x in M. P x*)
⟨*proof*⟩

lemma *AE-iff-null*:
assumes {x∈*space* M. ¬ P x} ∈ *sets* M (**is** *?P* ∈ *sets* M)
shows (*AE x in M. P x*) ⟷ {x∈*space* M. ¬ P x} ∈ *null-sets* M
⟨*proof*⟩

lemma *AE-iff-null-sets*:
N ∈ *sets* M ⟹ N ∈ *null-sets* M ⟷ (*AE x in M.* $x \notin N$)

⟨*proof*⟩

lemma *AE-not-in*:
$N \in$ *null-sets* $M \Longrightarrow$ *AE x in M*. $x \notin N$
⟨*proof*⟩

lemma *AE-iff-measurable*:
$N \in$ *sets* $M \Longrightarrow \{x \in$ *space* $M.\ \neg\ P\ x\} = N \Longrightarrow$ (*AE x in M*. $P\ x$) $\longleftrightarrow$ *emeasure* $M\ N = 0$
⟨*proof*⟩

lemma *AE-E*[*consumes 1*]:
assumes *AE x in M*. $P\ x$
obtains N **where** $\{x \in$ *space* $M.\ \neg\ P\ x\} \subseteq N$ *emeasure* $M\ N = 0$ $N \in$ *sets* M
⟨*proof*⟩

lemma *AE-E2*:
assumes *AE x in M*. $P\ x$ $\{x \in$ *space* $M.\ P\ x\} \in$ *sets* M
shows *emeasure* M $\{x \in$ *space* $M.\ \neg\ P\ x\} = 0$ (**is** *emeasure* M *?P* $= 0$)
⟨*proof*⟩

lemma *AE-I*:
assumes $\{x \in$ *space* $M.\ \neg\ P\ x\} \subseteq N$ *emeasure* $M\ N = 0$ $N \in$ *sets* M
shows *AE x in M*. $P\ x$
⟨*proof*⟩

lemma *AE-mp*[*elim*!]:
assumes *AE-P*: *AE x in M*. $P\ x$ **and** *AE-imp*: *AE x in M*. $P\ x \longrightarrow Q\ x$
shows *AE x in M*. $Q\ x$
⟨*proof*⟩

lemma
shows *AE-iffI*: *AE x in M*. $P\ x \Longrightarrow$ *AE x in M*. $P\ x \longleftrightarrow Q\ x \Longrightarrow$ *AE x in M*. $Q\ x$
and *AE-disjI1*: *AE x in M*. $P\ x \Longrightarrow$ *AE x in M*. $P\ x \lor Q\ x$
and *AE-disjI2*: *AE x in M*. $Q\ x \Longrightarrow$ *AE x in M*. $P\ x \lor Q\ x$
and *AE-conjI*: *AE x in M*. $P\ x \Longrightarrow$ *AE x in M*. $Q\ x \Longrightarrow$ *AE x in M*. $P\ x \land Q\ x$
and *AE-conj-iff*[*simp*]: (*AE x in M*. $P\ x \land Q\ x$) $\longleftrightarrow$ (*AE x in M*. $P\ x$) $\land$ (*AE x in M*. $Q\ x$)
⟨*proof*⟩

lemma *AE-impI*:
($P \Longrightarrow$ *AE x in M*. $Q\ x$) $\Longrightarrow$ *AE x in M*. $P \longrightarrow Q\ x$
⟨*proof*⟩

lemma *AE-measure*:
assumes *AE*: *AE x in M*. $P\ x$ **and** *sets*: $\{x \in$ *space* $M.\ P\ x\} \in$ *sets* M (**is** *?P* $\in$

sets M)
shows *emeasure M* {*x*∈*space M*. *P x*} = *emeasure M* (*space M*)
⟨*proof*⟩

lemma *AE-space*: *AE x in M*. *x* ∈ *space M*
⟨*proof*⟩

lemma *AE-I2*[*simp*, *intro*]:
(⋀*x*. *x* ∈ *space M* ⟹ *P x*) ⟹ *AE x in M*. *P x*
⟨*proof*⟩

lemma *AE-Ball-mp*:
∀ *x*∈*space M*. *P x* ⟹ *AE x in M*. *P x* ⟶ *Q x* ⟹ *AE x in M*. *Q x*
⟨*proof*⟩

lemma *AE-cong*[*cong*]:
(⋀*x*. *x* ∈ *space M* ⟹ *P x* ⟷ *Q x*) ⟹ (*AE x in M*. *P x*) ⟷ (*AE x in M*. *Q x*)
⟨*proof*⟩

lemma *AE-all-countable*:
(*AE x in M*. ∀ *i*. *P i x*) ⟷ (∀ *i*::'*i*::*countable*. *AE x in M*. *P i x*)
⟨*proof*⟩

lemma *AE-ball-countable*:
assumes [*intro*]: *countable X*
shows (*AE x in M*. ∀ *y*∈*X*. *P x y*) ⟷ (∀ *y*∈*X*. *AE x in M*. *P x y*)
⟨*proof*⟩

lemma *AE-discrete-difference*:
assumes *X*: *countable X*
assumes *null*: ⋀*x*. *x* ∈ *X* ⟹ *emeasure M* {*x*} = *0*
assumes *sets*: ⋀*x*. *x* ∈ *X* ⟹ {*x*} ∈ *sets M*
shows *AE x in M*. *x* ∉ *X*
⟨*proof*⟩

lemma *AE-finite-all*:
assumes *f*: *finite S* **shows** (*AE x in M*. ∀ *i*∈*S*. *P i x*) ⟷ (∀ *i*∈*S*. *AE x in M*. *P i x*)
⟨*proof*⟩

lemma *AE-finite-allI*:
assumes *finite S*
shows (⋀*s*. *s* ∈ *S* ⟹ *AE x in M*. *Q s x*) ⟹ *AE x in M*. ∀ *s*∈*S*. *Q s x*
⟨*proof*⟩

lemma *emeasure-mono-AE*:
assumes *imp*: *AE x in M*. *x* ∈ *A* ⟶ *x* ∈ *B*
and *B*: *B* ∈ *sets M*

shows *emeasure M A* $\leq$ *emeasure M B*
⟨*proof*⟩

lemma *emeasure-eq-AE*:
assumes *iff*: *AE x in M*. $x \in A \longleftrightarrow x \in B$
assumes *A*: $A \in$ *sets M* **and** *B*: $B \in$ *sets M*
shows *emeasure M A* = *emeasure M B*
⟨*proof*⟩

lemma *emeasure-Collect-eq-AE*:
AE x in M. *P x* $\longleftrightarrow$ *Q x* $\Longrightarrow$ *Measurable.pred M Q* $\Longrightarrow$ *Measurable.pred M P* $\Longrightarrow$
emeasure M {*x*∈*space M*. *P x*} = *emeasure M* {*x*∈*space M*. *Q x*}
⟨*proof*⟩

lemma *emeasure-eq-0-AE*: *AE x in M*. ¬ *P x* $\Longrightarrow$ *emeasure M* {*x*∈*space M*. *P x*} = *0*
⟨*proof*⟩

lemma *emeasure-add-AE*:
assumes [*measurable*]: $A \in$ *sets M* $B \in$ *sets M* $C \in$ *sets M*
assumes *1*: *AE x in M*. $x \in C \longleftrightarrow x \in A \vee x \in B$
assumes *2*: *AE x in M*. $\neg (x \in A \wedge x \in B)$
shows *emeasure M C* = *emeasure M A* + *emeasure M B*
⟨*proof*⟩

3.7 σ-finite Measures

locale *sigma-finite-measure* =
fixes *M* :: *'a measure*
assumes *sigma-finite-countable*:
∃ *A*::*'a set set*. *countable A* ∧ *A* ⊆ *sets M* ∧ (⋃ *A*) = *space M* ∧ (∀ *a*∈*A*. *emeasure M a* ≠ ∞)

lemma (**in** *sigma-finite-measure*) *sigma-finite*:
obtains *A* :: *nat* ⇒ *'a set*
where *range A* ⊆ *sets M* (⋃ *i*. *A i*) = *space M* ⋀*i*. *emeasure M* (*A i*) ≠ ∞
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *sigma-finite-disjoint*:
obtains *A* :: *nat* ⇒ *'a set*
where *range A* ⊆ *sets M* (⋃ *i*. *A i*) = *space M* ⋀*i*. *emeasure M* (*A i*) ≠ ∞ *disjoint-family A*
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *sigma-finite-incseq*:
obtains *A* :: *nat* ⇒ *'a set*
where *range A* ⊆ *sets M* (⋃ *i*. *A i*) = *space M* ⋀*i*. *emeasure M* (*A i*) ≠ ∞ *incseq A*

⟨*proof*⟩

3.8 Measure space induced by distribution of $op \rightarrow_M$-functions

definition *distr* :: *'a measure* ⇒ *'b measure* ⇒ (*'a* ⇒ *'b*) ⇒ *'b measure* **where**
distr M N f = *measure-of* (*space N*) (*sets N*) (λ*A*. *emeasure M* (*f* −' *A* ∩ *space M*))

lemma
shows *sets-distr*[*simp*, *measurable-cong*]: *sets* (*distr M N f*) = *sets N*
and *space-distr*[*simp*]: *space* (*distr M N f*) = *space N*
⟨*proof*⟩

lemma
shows *measurable-distr-eq1*[*simp*]: *measurable* (*distr Mf Nf f*) *Mf'* = *measurable Nf Mf'*
and *measurable-distr-eq2*[*simp*]: *measurable Mg'* (*distr Mg Ng g*) = *measurable Mg' Ng*
⟨*proof*⟩

lemma *distr-cong*:
M = *K* ⟹ *sets N* = *sets L* ⟹ (⋀*x*. *x* ∈ *space M* ⟹ *f x* = *g x*) ⟹ *distr M N f* = *distr K L g*
⟨*proof*⟩

lemma *emeasure-distr*:
fixes *f* :: *'a* ⇒ *'b*
assumes *f*: *f* ∈ *measurable M N* **and** *A*: *A* ∈ *sets N*
shows *emeasure* (*distr M N f*) *A* = *emeasure M* (*f* −' *A* ∩ *space M*) (**is** - = *?μ A*)
⟨*proof*⟩

lemma *emeasure-Collect-distr*:
assumes *X*[*measurable*]: *X* ∈ *measurable M N Measurable.pred N P*
shows *emeasure* (*distr M N X*) {*x*∈*space N*. *P x*} = *emeasure M* {*x*∈*space M*. *P* (*X x*)}
⟨*proof*⟩

lemma *emeasure-lfp2*[*consumes 1*, *case-names cont f measurable*]:
assumes *P M*
assumes *cont*: *sup-continuous F*
assumes *f*: ⋀*M*. *P M* ⟹ *f* ∈ *measurable M' M*
assumes ∗: ⋀*M A*. *P M* ⟹ (⋀*N*. *P N* ⟹ *Measurable.pred N A*) ⟹ *Measurable.pred M* (*F A*)
shows *emeasure M'* {*x*∈*space M'*. *lfp F* (*f x*)} = (*SUP i*. *emeasure M'* {*x*∈*space M'*. (*F* ^^ *i*) (λ*x*. *False*) (*f x*)})
⟨*proof*⟩

lemma *distr-id*[*simp*]: *distr N N* (λ*x*. *x*) = *N*

⟨*proof*⟩

lemma *measure-distr*:
$f \in measurable\ M\ N \Longrightarrow S \in sets\ N \Longrightarrow measure\ (distr\ M\ N\ f)\ S = measure\ M\ (f\ -`\ S \cap space\ M)$
⟨*proof*⟩

lemma *distr-cong-AE*:
assumes *1*: $M = K$ $sets\ N = sets\ L$ **and**
2: $(AE\ x\ in\ M.\ f\ x = g\ x)$ **and** $f \in measurable\ M\ N$ **and** $g \in measurable\ K\ L$
shows $distr\ M\ N\ f = distr\ K\ L\ g$
⟨*proof*⟩

lemma *AE-distrD*:
assumes *f*: $f \in measurable\ M\ M'$
and *AE*: $AE\ x\ in\ distr\ M\ M'\ f.\ P\ x$
shows $AE\ x\ in\ M.\ P\ (f\ x)$
⟨*proof*⟩

lemma *AE-distr-iff*:
assumes *f*[*measurable*]: $f \in measurable\ M\ N$ **and** *P*[*measurable*]: $\{x \in space\ N.\ P\ x\} \in sets\ N$
shows $(AE\ x\ in\ distr\ M\ N\ f.\ P\ x) \longleftrightarrow (AE\ x\ in\ M.\ P\ (f\ x))$
⟨*proof*⟩

lemma *null-sets-distr-iff*:
$f \in measurable\ M\ N \Longrightarrow A \in null\text{-}sets\ (distr\ M\ N\ f) \longleftrightarrow f\ -`\ A \cap space\ M \in null\text{-}sets\ M \wedge A \in sets\ N$
⟨*proof*⟩

lemma *distr-distr*:
$g \in measurable\ N\ L \Longrightarrow f \in measurable\ M\ N \Longrightarrow distr\ (distr\ M\ N\ f)\ L\ g = distr\ M\ L\ (g \circ f)$
⟨*proof*⟩

3.9 Real measure values

lemma *ring-of-finite-sets*: $ring\text{-}of\text{-}sets\ (space\ M)\ \{A{\in}sets\ M.\ emeasure\ M\ A \neq top\}$
⟨*proof*⟩

lemma *measure-nonneg*[*simp*]: $0 \leq measure\ M\ A$
⟨*proof*⟩

lemma *zero-less-measure-iff*: $0 < measure\ M\ A \longleftrightarrow measure\ M\ A \neq 0$
⟨*proof*⟩

lemma *measure-le-0-iff*: $measure\ M\ X \leq 0 \longleftrightarrow measure\ M\ X = 0$
⟨*proof*⟩

lemma *measure-empty*[*simp*]: *measure M* {} = *0*
⟨*proof*⟩

lemma *emeasure-eq-ennreal-measure*:
emeasure M A ≠ *top* ⟹ *emeasure M A* = *ennreal* (*measure M A*)
⟨*proof*⟩

lemma *measure-zero-top*: *emeasure M A* = *top* ⟹ *measure M A* = *0*
⟨*proof*⟩

lemma *measure-eq-emeasure-eq-ennreal*: *0* ≤ *x* ⟹ *emeasure M A* = *ennreal x* ⟹ *measure M A* = *x*
⟨*proof*⟩

lemma *enn2real-plus*:*a* < *top* ⟹ *b* < *top* ⟹ *enn2real* (*a* + *b*) = *enn2real a* + *enn2real b*
⟨*proof*⟩

lemma *measure-Union*:
emeasure M A ≠ ∞ ⟹ *emeasure M B* ≠ ∞ ⟹ *A* ∈ *sets M* ⟹ *B* ∈ *sets M* ⟹ *A* ∩ *B* = {} ⟹
measure M (*A* ∪ *B*) = *measure M A* + *measure M B*
⟨*proof*⟩

lemma *disjoint-family-on-insert*:
i ∉ *I* ⟹ *disjoint-family-on A* (*insert i I*) ⟷ *A i* ∩ (⋃*i*∈*I*. *A i*) = {} ∧ *disjoint-family-on A I*
⟨*proof*⟩

lemma *measure-finite-Union*:
finite S ⟹ *A*'*S* ⊆ *sets M* ⟹ *disjoint-family-on A S* ⟹ (⋀*i*. *i* ∈ *S* ⟹ *emeasure M* (*A i*) ≠ ∞) ⟹
measure M (⋃*i*∈*S*. *A i*) = (∑*i*∈*S*. *measure M* (*A i*))
⟨*proof*⟩

lemma *measure-Diff*:
assumes *finite*: *emeasure M A* ≠ ∞
and *measurable*: *A* ∈ *sets M B* ∈ *sets M B* ⊆ *A*
shows *measure M* (*A* − *B*) = *measure M A* − *measure M B*
⟨*proof*⟩

lemma *measure-UNION*:
assumes *measurable*: *range A* ⊆ *sets M disjoint-family A*
assumes *finite*: *emeasure M* (⋃*i*. *A i*) ≠ ∞
shows (λ*i*. *measure M* (*A i*)) *sums* (*measure M* (⋃*i*. *A i*))
⟨*proof*⟩

lemma *measure-subadditive*:

assumes *measurable*: $A \in$ *sets M* $B \in$ *sets M*
and *fin*: *emeasure M A* $\neq \infty$ *emeasure M B* $\neq \infty$
shows *measure M* $(A \cup B) \leq$ *measure M A* $+$ *measure M B*
⟨*proof*⟩

lemma *measure-subadditive-finite*:
assumes *A*: *finite I A ‘I* $\subseteq$ *sets M* **and** *fin*: $\bigwedge i.\ i \in I \Longrightarrow$ *emeasure M* $(A\ i) \neq \infty$
shows *measure M* $(\bigcup i{\in}I.\ A\ i) \leq (\sum i{\in}I.$ *measure M* $(A\ i))$
⟨*proof*⟩

lemma *measure-subadditive-countably*:
assumes *A*: *range A* $\subseteq$ *sets M* **and** *fin*: $(\sum i.$ *emeasure M* $(A\ i)) \neq \infty$
shows *measure M* $(\bigcup i.\ A\ i) \leq (\sum i.$ *measure M* $(A\ i))$
⟨*proof*⟩

lemma *measure-eq-setsum-singleton*:
finite S $\Longrightarrow (\bigwedge x.\ x \in S \Longrightarrow \{x\} \in$ *sets M*$) \Longrightarrow (\bigwedge x.\ x \in S \Longrightarrow$ *emeasure M* $\{x\} \neq \infty) \Longrightarrow$
measure M S $= (\sum x{\in}S.$ *measure M* $\{x\})$
⟨*proof*⟩

lemma *Lim-measure-incseq*:
assumes *A*: *range A* $\subseteq$ *sets M incseq A* **and** *fin*: *emeasure M* $(\bigcup i.\ A\ i) \neq \infty$
shows $(\lambda i.$ *measure M* $(A\ i)) \longrightarrow$ *measure M* $(\bigcup i.\ A\ i)$
⟨*proof*⟩

lemma *Lim-measure-decseq*:
assumes *A*: *range A* $\subseteq$ *sets M decseq A* **and** *fin*: $\bigwedge i.$ *emeasure M* $(A\ i) \neq \infty$
shows $(\lambda n.$ *measure M* $(A\ n)) \longrightarrow$ *measure M* $(\bigcap i.\ A\ i)$
⟨*proof*⟩

3.10 Measure spaces with *emeasure M* (*space M*) $< \infty$

locale *finite-measure* = *sigma-finite-measure M* **for** *M* +
assumes *finite-emeasure-space*: *emeasure M* (*space M*) $\neq$ *top*

lemma *finite-measureI*[*Pure.intro*!]:
emeasure M (*space M*) $\neq \infty \Longrightarrow$ *finite-measure M*
⟨*proof*⟩

lemma (**in** *finite-measure*) *emeasure-finite*[*simp*, *intro*]: *emeasure M A* $\neq$ *top*
⟨*proof*⟩

lemma (**in** *finite-measure*) *emeasure-eq-measure*: *emeasure M A* = *ennreal* (*measure M A*)
⟨*proof*⟩

lemma (**in** *finite-measure*) *emeasure-real*: $\exists r.\ 0 \leq r \land$ *emeasure M A* = *ennreal*

r
⟨*proof*⟩

lemma (**in** *finite-measure*) *bounded-measure*: *measure M A* $\leq$ *measure M* (*space M*)
⟨*proof*⟩

lemma (**in** *finite-measure*) *finite-measure-Diff*:
assumes *sets*: $A \in$ *sets M* $B \in$ *sets M* **and** $B \subseteq A$
shows *measure M* $(A - B) =$ *measure M A* $-$ *measure M B*
⟨*proof*⟩

lemma (**in** *finite-measure*) *finite-measure-Union*:
assumes *sets*: $A \in$ *sets M* $B \in$ *sets M* **and** $A \cap B = \{\}$
shows *measure M* $(A \cup B) =$ *measure M A* $+$ *measure M B*
⟨*proof*⟩

lemma (**in** *finite-measure*) *finite-measure-finite-Union*:
assumes *measurable*: *finite S A'S* $\subseteq$ *sets M disjoint-family-on A S*
shows *measure M* $(\bigcup i{\in}S.\ A\ i) = (\sum i{\in}S.$ *measure M* $(A\ i))$
⟨*proof*⟩

lemma (**in** *finite-measure*) *finite-measure-UNION*:
assumes *A*: *range A* $\subseteq$ *sets M disjoint-family A*
shows $(\lambda i.$ *measure M* $(A\ i))$ *sums* (*measure M* $(\bigcup i.\ A\ i))$
⟨*proof*⟩

lemma (**in** *finite-measure*) *finite-measure-mono*:
assumes $A \subseteq B$ $B \in$ *sets M* **shows** *measure M A* $\leq$ *measure M B*
⟨*proof*⟩

lemma (**in** *finite-measure*) *finite-measure-subadditive*:
assumes *m*: $A \in$ *sets M* $B \in$ *sets M*
shows *measure M* $(A \cup B) \leq$ *measure M A* $+$ *measure M B*
⟨*proof*⟩

lemma (**in** *finite-measure*) *finite-measure-subadditive-finite*:
assumes *finite I A'I* $\subseteq$ *sets M* **shows** *measure M* $(\bigcup i{\in}I.\ A\ i) \leq (\sum i{\in}I.$ *measure M* $(A\ i))$
⟨*proof*⟩

lemma (**in** *finite-measure*) *finite-measure-subadditive-countably*:
range A $\subseteq$ *sets M* $\Longrightarrow$ *summable* $(\lambda i.$ *measure M* $(A\ i)) \Longrightarrow$ *measure M* $(\bigcup i.\ A\ i) \leq (\sum i.$ *measure M* $(A\ i))$
⟨*proof*⟩

lemma (**in** *finite-measure*) *finite-measure-eq-setsum-singleton*:
assumes *finite S* **and** $*$: $\bigwedge x.\ x \in S \Longrightarrow \{x\} \in$ *sets M*
shows *measure M S* $= (\sum x{\in}S.$ *measure M* $\{x\})$

⟨*proof*⟩

lemma (**in** *finite-measure*) *finite-Lim-measure-incseq*:
assumes *A*: *range A* $\subseteq$ *sets M incseq A*
shows $(\lambda i.\ measure\ M\ (A\ i)) \longrightarrow measure\ M\ (\bigcup i.\ A\ i)$
⟨*proof*⟩

lemma (**in** *finite-measure*) *finite-Lim-measure-decseq*:
assumes *A*: *range A* $\subseteq$ *sets M decseq A*
shows $(\lambda n.\ measure\ M\ (A\ n)) \longrightarrow measure\ M\ (\bigcap i.\ A\ i)$
⟨*proof*⟩

lemma (**in** *finite-measure*) *finite-measure-compl*:
assumes *S*: $S \in$ *sets M*
shows *measure M* (*space M* $-$ *S*) $=$ *measure M* (*space M*) $-$ *measure M S*
⟨*proof*⟩

lemma (**in** *finite-measure*) *finite-measure-mono-AE*:
assumes *imp*: *AE x in M*. $x \in A \longrightarrow x \in B$ **and** *B*: $B \in$ *sets M*
shows *measure M A* $\leq$ *measure M B*
⟨*proof*⟩

lemma (**in** *finite-measure*) *finite-measure-eq-AE*:
assumes *iff*: *AE x in M*. $x \in A \longleftrightarrow x \in B$
assumes *A*: $A \in$ *sets M* **and** *B*: $B \in$ *sets M*
shows *measure M A* $=$ *measure M B*
⟨*proof*⟩

lemma (**in** *finite-measure*) *measure-increasing*: *increasing M* (*measure M*)
⟨*proof*⟩

lemma (**in** *finite-measure*) *measure-zero-union*:
assumes $s \in$ *sets M* $t \in$ *sets M measure M t* $=$ *0*
shows *measure M* $(s \cup t)$ $=$ *measure M s*
⟨*proof*⟩

lemma (**in** *finite-measure*) *measure-eq-compl*:
assumes $s \in$ *sets M* $t \in$ *sets M*
assumes *measure M* (*space M* $-$ *s*) $=$ *measure M* (*space M* $-$ *t*)
shows *measure M s* $=$ *measure M t*
⟨*proof*⟩

lemma (**in** *finite-measure*) *measure-eq-bigunion-image*:
assumes *range f* $\subseteq$ *sets M range g* $\subseteq$ *sets M*
assumes *disjoint-family f disjoint-family g*
assumes $\bigwedge n :: nat.\ measure\ M\ (f\ n) = measure\ M\ (g\ n)$
shows $measure\ M\ (\bigcup i.\ f\ i) = measure\ M\ (\bigcup i.\ g\ i)$
⟨*proof*⟩

lemma (**in** *finite-measure*) *measure-countably-zero*:
assumes *range c* $\subseteq$ *sets M*
assumes $\bigwedge$ *i. measure M (c i) = 0*
shows *measure M* $(\bigcup i :: nat.\ c\ i) = 0$
⟨*proof*⟩

lemma (**in** *finite-measure*) *measure-space-inter*:
assumes *events*:$s \in$ *sets M* $t \in$ *sets M*
assumes *measure M t = measure M (space M)*
shows *measure M* $(s \cap t)$ = *measure M s*
⟨*proof*⟩

lemma (**in** *finite-measure*) *measure-equiprobable-finite-unions*:
assumes *s*: *finite s* $\bigwedge x.\ x \in s \Longrightarrow \{x\} \in$ *sets M*
assumes $\bigwedge$ *x y*. $[\![x \in s;\ y \in s]\!] \Longrightarrow$ *measure M* $\{x\}$ = *measure M* $\{y\}$
shows *measure M s = real (card s)* $*$ *measure M* $\{SOME\ x.\ x \in s\}$
⟨*proof*⟩

lemma (**in** *finite-measure*) *measure-real-sum-image-fn*:
assumes $e \in$ *sets M*
assumes $\bigwedge$ *x*. $x \in s \Longrightarrow e \cap f\ x \in$ *sets M*
assumes *finite s*
assumes *disjoint*: $\bigwedge$ *x y*. $[\![x \in s\ ;\ y \in s\ ;\ x \neq y]\!] \Longrightarrow f\ x \cap f\ y = \{\}$
assumes *upper*: *space M* $\subseteq (\bigcup i \in s.\ f\ i)$
shows *measure M e* $= (\sum\ x \in s.$ *measure M* $(e \cap f\ x))$
⟨*proof*⟩

lemma (**in** *finite-measure*) *measure-exclude*:
assumes $A \in$ *sets M* $B \in$ *sets M*
assumes *measure M A = measure M (space M)* $A \cap B = \{\}$
shows *measure M B = 0*
⟨*proof*⟩
lemma (**in** *finite-measure*) *finite-measure-distr*:
assumes *f*: $f \in$ *measurable M M′*
shows *finite-measure (distr M M′ f)*
⟨*proof*⟩

lemma *emeasure-gfp*[*consumes 1*, *case-names cont measurable*]:
assumes *sets*[*simp*]: $\bigwedge$*s. sets (M s) = sets N*
assumes $\bigwedge$*s. finite-measure (M s)*
assumes *cont*: *inf-continuous F inf-continuous f*
assumes *meas*: $\bigwedge$*P. Measurable.pred N P* $\Longrightarrow$ *Measurable.pred N (F P)*
assumes *iter*: $\bigwedge$*P s. Measurable.pred N P* $\Longrightarrow$ *emeasure (M s)* $\{x \in$*space N. F P x*$\}$ = *f* (λ*s. emeasure (M s)* $\{x \in$*space N. P x*$\}$) *s*
assumes *bound*: $\bigwedge$*P. f P* $\leq$ *f* (λ*s. emeasure (M s) (space (M s))*)
shows *emeasure (M s)* $\{x \in$*space N. gfp F x*$\}$ = *gfp f s*
⟨*proof*⟩

3.11 Counting space

lemma *strict-monoI-Suc*:
assumes *ord* [*simp*]: ($\bigwedge n.\ f\ n < f\ (Suc\ n)$) **shows** *strict-mono f*
⟨*proof*⟩

lemma *emeasure-count-space*:
assumes $X \subseteq A$ **shows** *emeasure* (*count-space A*) $X =$ (*if finite X then of-nat* (*card X*) *else* ∞)
(**is** - = *?M X*)
⟨*proof*⟩

lemma *distr-bij-count-space*:
assumes *f*: *bij-betw f A B*
shows *distr* (*count-space A*) (*count-space B*) *f* = *count-space B*
⟨*proof*⟩

lemma *emeasure-count-space-finite*[*simp*]:
$X \subseteq A \Longrightarrow$ *finite X* $\Longrightarrow$ *emeasure* (*count-space A*) *X* = *of-nat* (*card X*)
⟨*proof*⟩

lemma *emeasure-count-space-infinite*[*simp*]:
$X \subseteq A \Longrightarrow$ *infinite X* $\Longrightarrow$ *emeasure* (*count-space A*) $X = \infty$
⟨*proof*⟩

lemma *measure-count-space*: *measure* (*count-space A*) *X* = (*if* $X \subseteq A$ *then of-nat* (*card X*) *else 0*)
⟨*proof*⟩

lemma *emeasure-count-space-eq-0*:
emeasure (*count-space A*) *X* = *0* $\longleftrightarrow$ ($X \subseteq A \longrightarrow X = \{\}$)
⟨*proof*⟩

lemma *space-empty*: *space M* = $\{\} \Longrightarrow$ *M* = *count-space* $\{\}$
⟨*proof*⟩

lemma *null-sets-count-space*: *null-sets* (*count-space A*) = $\{\ \{\}\ \}$
⟨*proof*⟩

lemma *AE-count-space*: (*AE x in count-space A. P x*) $\longleftrightarrow$ ($\forall x \in A.\ P\ x$)
⟨*proof*⟩

lemma *sigma-finite-measure-count-space-countable*:
assumes *A*: *countable A*
shows *sigma-finite-measure* (*count-space A*)
⟨*proof*⟩

lemma *sigma-finite-measure-count-space*:
fixes *A* :: *'a::countable set* **shows** *sigma-finite-measure* (*count-space A*)
⟨*proof*⟩

lemma *finite-measure-count-space*:
assumes [*simp*]: *finite A*
shows *finite-measure* (*count-space A*)
⟨*proof*⟩

lemma *sigma-finite-measure-count-space-finite*:
assumes *A*: *finite A* **shows** *sigma-finite-measure* (*count-space A*)
⟨*proof*⟩

3.12 Measure restricted to space

lemma *emeasure-restrict-space*:
assumes Ω ∩ *space M* ∈ *sets M A* ⊆ Ω
shows *emeasure* (*restrict-space M* Ω) *A* = *emeasure M A*
⟨*proof*⟩

lemma *measure-restrict-space*:
assumes Ω ∩ *space M* ∈ *sets M A* ⊆ Ω
shows *measure* (*restrict-space M* Ω) *A* = *measure M A*
⟨*proof*⟩

lemma *AE-restrict-space-iff*:
assumes Ω ∩ *space M* ∈ *sets M*
shows (*AE x in restrict-space M* Ω. *P x*) ⟷ (*AE x in M*. *x* ∈ Ω ⟶ *P x*)
⟨*proof*⟩

lemma *restrict-restrict-space*:
assumes *A* ∩ *space M* ∈ *sets M B* ∩ *space M* ∈ *sets M*
shows *restrict-space* (*restrict-space M A*) *B* = *restrict-space M* (*A* ∩ *B*) (**is** *?l* = *?r*)
⟨*proof*⟩

lemma *restrict-count-space*: *restrict-space* (*count-space B*) *A* = *count-space* (*A* ∩ *B*)
⟨*proof*⟩

lemma *sigma-finite-measure-restrict-space*:
assumes *sigma-finite-measure M*
and *A*: *A* ∈ *sets M*
shows *sigma-finite-measure* (*restrict-space M A*)
⟨*proof*⟩

lemma *finite-measure-restrict-space*:
assumes *finite-measure M*
and *A*: *A* ∈ *sets M*
shows *finite-measure* (*restrict-space M A*)
⟨*proof*⟩

lemma *restrict-distr*:
assumes [*measurable*]: $f \in$ *measurable M N*
assumes [*simp*]: $\Omega \cap$ *space N* $\in$ *sets N* **and** *restrict*: $f \in$ *space M* $\rightarrow \Omega$
shows *restrict-space* (*distr M N f*) Ω = *distr M* (*restrict-space N* Ω) *f*
(**is** *?l* = *?r*)
⟨*proof*⟩

lemma *measure-eqI-restrict-generator*:
assumes *E*: *Int-stable E E* $\subseteq$ *Pow* Ω $\bigwedge X.\ X \in E \Longrightarrow$ *emeasure M X* = *emeasure N X*
assumes *sets-eq*: *sets M* = *sets N* **and** Ω: $\Omega \in$ *sets M*
assumes *sets* (*restrict-space M* Ω) = *sigma-sets* Ω *E*
assumes *sets* (*restrict-space N* Ω) = *sigma-sets* Ω *E*
assumes *ae*: *AE x in M*. $x \in \Omega$ *AE x in N*. $x \in \Omega$
assumes *A*: *countable A* $A \neq \{\}$ $A \subseteq E$ $\bigcup A = \Omega$ $\bigwedge a.\ a \in A \Longrightarrow$ *emeasure M a* $\neq \infty$
shows *M* = *N*
⟨*proof*⟩

3.13 Null measure

definition *null-measure M* = *sigma* (*space M*) (*sets M*)

lemma *space-null-measure*[*simp*]: *space* (*null-measure M*) = *space M*
⟨*proof*⟩

lemma *sets-null-measure*[*simp*, *measurable-cong*]: *sets* (*null-measure M*) = *sets M*
⟨*proof*⟩

lemma *emeasure-null-measure*[*simp*]: *emeasure* (*null-measure M*) *X* = *0*
⟨*proof*⟩

lemma *measure-null-measure*[*simp*]: *measure* (*null-measure M*) *X* = *0*
⟨*proof*⟩

lemma *null-measure-idem* [*simp*]: *null-measure* (*null-measure M*) = *null-measure M*
⟨*proof*⟩

3.14 Scaling a measure

definition *scale-measure* :: *ennreal* ⇒ *'a measure* ⇒ *'a measure*
where
scale-measure r M = *measure-of* (*space M*) (*sets M*) ($\lambda A.\ r *$ *emeasure M A*)

lemma *space-scale-measure*: *space* (*scale-measure r M*) = *space M*
⟨*proof*⟩

lemma *sets-scale-measure* [*simp*, *measurable-cong*]: *sets* (*scale-measure r M*) = *sets M*

⟨*proof*⟩

lemma *emeasure-scale-measure* [*simp*]:
emeasure (*scale-measure r M*) *A* = *r* ∗ *emeasure M A*
(**is** - = *?μ A*)
⟨*proof*⟩

lemma *scale-measure-1* [*simp*]: *scale-measure 1 M* = *M*
⟨*proof*⟩

lemma *scale-measure-0*[*simp*]: *scale-measure 0 M* = *null-measure M*
⟨*proof*⟩

lemma *measure-scale-measure* [*simp*]: *0* ≤ *r* ⟹ *measure* (*scale-measure r M*) *A* = *r* ∗ *measure M A*
⟨*proof*⟩

lemma *scale-scale-measure* [*simp*]:
scale-measure r (*scale-measure r′ M*) = *scale-measure* (*r* ∗ *r′*) *M*
⟨*proof*⟩

lemma *scale-null-measure* [*simp*]: *scale-measure r* (*null-measure M*) = *null-measure M*
⟨*proof*⟩

3.15 Measures form a chain-complete partial order

instantiation *measure* :: (*type*) *order-bot*
begin

definition *bot-measure* :: *′a measure* **where**
bot-measure = *sigma* {} {{}}

lemma *space-bot*[*simp*]: *space bot* = {}
⟨*proof*⟩

lemma *sets-bot*[*simp*]: *sets bot* = {{}}
⟨*proof*⟩

lemma *emeasure-bot*[*simp*]: *emeasure bot* = (*λx. 0*)
⟨*proof*⟩

inductive *less-eq-measure* :: *′a measure* ⇒ *′a measure* ⇒ *bool* **where**
sets N = *sets M* ⟹ (⋀*A. A* ∈ *sets M* ⟹ *emeasure M A* ≤ *emeasure N A*) ⟹ *less-eq-measure M N*
| *less-eq-measure bot N*

definition *less-measure* :: *′a measure* ⇒ *′a measure* ⇒ *bool* **where**
less-measure M N ⟷ (*M* ≤ *N* ∧ ¬ *N* ≤ *M*)

instance
⟨*proof*⟩
end

lemma *le-emeasureD*: $M \leq N \Longrightarrow$ *emeasure* M $A \leq$ *emeasure* N A
⟨*proof*⟩

lemma *le-sets*: $N \leq M \Longrightarrow$ *sets* $N \leq$ *sets* M
⟨*proof*⟩

instantiation *measure* :: (*type*) *ccpo*
begin

definition *Sup-measure* :: *'a measure set* ⇒ *'a measure* **where**
Sup-measure A = *measure-of* (*SUP a:A. space a*) (*SUP a:A. sets a*) (*SUP a:A. emeasure a*)

lemma
assumes *A*: *Complete-Partial-Order.chain op* ≤ *A* **and** *a*: $a \neq bot$ $a \in A$
shows *space-Sup*: *space* (*Sup A*) = *space a*
and *sets-Sup*: *sets* (*Sup A*) = *sets a*
⟨*proof*⟩

lemma *emeasure-Sup*:
assumes *A*: *Complete-Partial-Order.chain op* ≤ *A* $A \neq \{\}$
assumes $X \in$ *sets* (*Sup A*)
shows *emeasure* (*Sup A*) *X* = (*SUP a:A. emeasure a*) *X*
⟨*proof*⟩

instance
⟨*proof*⟩
end

lemma
assumes *A*: *Complete-Partial-Order.chain op* ≤ (*f* ‘ *A*) **and** *a*: $a \in A$ $f\ a \neq bot$
shows *space-SUP*: *space* (*SUP M:A. f M*) = *space* (*f a*)
and *sets-SUP*: *sets* (*SUP M:A. f M*) = *sets* (*f a*)
⟨*proof*⟩

lemma *emeasure-SUP*:
assumes *A*: *Complete-Partial-Order.chain op* ≤ (*f* ‘ *A*) $A \neq \{\}$
assumes $X \in$ *sets* (*SUP M:A. f M*)
shows *emeasure* (*SUP M:A. f M*) *X* = (*SUP M:A. emeasure* (*f M*)) *X*
⟨*proof*⟩

end

4 Borel spaces

theory *Borel-Space*
imports
Measurable
~~/src/HOL/Multivariate-Analysis/Multivariate-Analysis
begin

lemma *sets-Collect-eventually-sequentially*[*measurable*]:
$(\bigwedge i.\ \{x{\in}space\ M.\ P\ x\ i\} \in sets\ M) \Longrightarrow \{x{\in}space\ M.\ eventually\ (P\ x)\ sequentially\} \in sets\ M$
⟨*proof*⟩

lemma *open-Collect-less*:
fixes $f\ g :: 'i{::}topological\text{-}space \Rightarrow 'a :: \{dense\text{-}linorder,\ linorder\text{-}topology\}$
assumes *continuous-on UNIV f*
assumes *continuous-on UNIV g*
shows *open* $\{x.\ f\ x < g\ x\}$
⟨*proof*⟩

lemma *closed-Collect-le*:
fixes $f\ g :: 'i{::}topological\text{-}space \Rightarrow 'a :: \{dense\text{-}linorder,\ linorder\text{-}topology\}$
assumes *f*: *continuous-on UNIV f*
assumes *g*: *continuous-on UNIV g*
shows *closed* $\{x.\ f\ x \le g\ x\}$
⟨*proof*⟩

lemma *topological-basis-trivial*: *topological-basis* $\{A.\ open\ A\}$
⟨*proof*⟩

lemma *open-prod-generated*: $open = generate\text{-}topology\ \{A \times B \mid A\ B.\ open\ A \wedge open\ B\}$
⟨*proof*⟩

definition $mono\text{-}on\ f\ A \equiv \forall r\ s.\ r \in A \wedge s \in A \wedge r \le s \longrightarrow f\ r \le f\ s$

lemma *mono-onI*:
$(\bigwedge r\ s.\ r \in A \Longrightarrow s \in A \Longrightarrow r \le s \Longrightarrow f\ r \le f\ s) \Longrightarrow mono\text{-}on\ f\ A$
⟨*proof*⟩

lemma *mono-onD*:
$[\![mono\text{-}on\ f\ A;\ r \in A;\ s \in A;\ r \le s]\!] \Longrightarrow f\ r \le f\ s$
⟨*proof*⟩

lemma *mono-imp-mono-on*: $mono\ f \Longrightarrow mono\text{-}on\ f\ A$
⟨*proof*⟩

lemma *mono-on-subset*: $mono\text{-}on\ f\ A \Longrightarrow B \subseteq A \Longrightarrow mono\text{-}on\ f\ B$
⟨*proof*⟩

definition *strict-mono-on f A* ≡ ∀ *r s*. *r* ∈ *A* ∧ *s* ∈ *A* ∧ *r* < *s* ⟶ *f r* < *f s*

lemma *strict-mono-onI*:
(⋀*r s*. *r* ∈ *A* ⟹ *s* ∈ *A* ⟹ *r* < *s* ⟹ *f r* < *f s*) ⟹ *strict-mono-on f A*
⟨*proof*⟩

lemma *strict-mono-onD*:
⟦*strict-mono-on f A*; *r* ∈ *A*; *s* ∈ *A*; *r* < *s*⟧ ⟹ *f r* < *f s*
⟨*proof*⟩

lemma *mono-on-greaterD*:
assumes *mono-on g A x* ∈ *A y* ∈ *A g x* > (*g* (*y*::-::*linorder*) :: - :: *linorder*)
shows *x* > *y*
⟨*proof*⟩

lemma *strict-mono-inv*:
fixes *f* :: (′*a*::*linorder*) ⇒ (′*b*::*linorder*)
assumes *strict-mono f* **and** *surj f* **and** *inv*: ⋀*x*. *g* (*f x*) = *x*
shows *strict-mono g*
⟨*proof*⟩

lemma *strict-mono-on-imp-inj-on*:
assumes *strict-mono-on* (*f* :: (- :: *linorder*) ⇒ (- :: *preorder*)) *A*
shows *inj-on f A*
⟨*proof*⟩

lemma *strict-mono-on-leD*:
assumes *strict-mono-on* (*f* :: (- :: *linorder*) ⇒ - :: *preorder*) *A x* ∈ *A y* ∈ *A x* ≤ *y*
shows *f x* ≤ *f y*
⟨*proof*⟩

lemma *strict-mono-on-eqD*:
fixes *f* :: (- :: *linorder*) ⇒ (- :: *preorder*)
assumes *strict-mono-on f A f x* = *f y x* ∈ *A y* ∈ *A*
shows *y* = *x*
⟨*proof*⟩

lemma *mono-on-imp-deriv-nonneg*:
assumes *mono*: *mono-on f A* **and** *deriv*: (*f has-real-derivative D*) (*at x*)
assumes *x* ∈ *interior A*
shows *D* ≥ *0*
⟨*proof*⟩

lemma *strict-mono-on-imp-mono-on*:
strict-mono-on (*f* :: (- :: *linorder*) ⇒ - :: *preorder*) *A* ⟹ *mono-on f A*
⟨*proof*⟩

lemma *mono-on-ctble-discont*:
fixes f :: *real* ⇒ *real*
fixes A :: *real set*
assumes *mono-on f A*
shows *countable* $\{a \in A.\ \neg$ *continuous* (*at a within A*) $f\}$
⟨*proof*⟩

lemma *mono-on-ctble-discont-open*:
fixes f :: *real* ⇒ *real*
fixes A :: *real set*
assumes *open A mono-on f A*
shows *countable* $\{a \in A.\ \neg$*isCont f a*$\}$
⟨*proof*⟩

lemma *mono-ctble-discont*:
fixes f :: *real* ⇒ *real*
assumes *mono f*
shows *countable* $\{a.\ \neg$ *isCont f a*$\}$
⟨*proof*⟩

lemma *has-real-derivative-imp-continuous-on*:
assumes $\bigwedge x.\ x \in A \Longrightarrow$ (*f has-real-derivative f′ x*) (*at x*)
shows *continuous-on A f*
⟨*proof*⟩

lemma *closure-contains-Sup*:
fixes S :: *real set*
assumes $S \neq \{\}$ *bdd-above S*
shows *Sup S* ∈ *closure S*
⟨*proof*⟩

lemma *closed-contains-Sup*:
fixes S :: *real set*
shows $S \neq \{\} \Longrightarrow$ *bdd-above S* ⟹ *closed S* ⟹ *Sup S* ∈ *S*
⟨*proof*⟩

lemma *deriv-nonneg-imp-mono*:
assumes *deriv*: $\bigwedge x.\ x \in \{a..b\} \Longrightarrow$ (*g has-real-derivative g′ x*) (*at x*)
assumes *nonneg*: $\bigwedge x.\ x \in \{a..b\} \Longrightarrow g'\ x \geq 0$
assumes *ab*: $a \leq b$
shows $g\ a \leq g\ b$
⟨*proof*⟩

lemma *continuous-interval-vimage-Int*:
assumes *continuous-on* $\{a$::*real*..$b\}$ g **and** *mono*: $\bigwedge x\ y.\ a \leq x \Longrightarrow x \leq y \Longrightarrow y \leq b \Longrightarrow g\ x \leq g\ y$
assumes $a \leq b$ (c::*real*) $\leq d$ $\{c..d\} \subseteq \{g\ a..g\ b\}$
obtains $c'\ d'$ **where** $\{a..b\} \cap g$ −‘ $\{c..d\} = \{c'..d'\}$ $c' \leq d'$ $g\ c' = c$ $g\ d' = d$
⟨*proof*⟩

4.1 Generic Borel spaces

definition (**in** *topological-space*) *borel* :: *'a measure* **where**
borel = *sigma UNIV* {*S. open S*}

abbreviation *borel-measurable M* ≡ *measurable M borel*

lemma *in-borel-measurable*:
$f \in$ *borel-measurable M* $\longleftrightarrow$
($\forall S \in$ *sigma-sets UNIV* {*S. open S*}. f −' $S \cap$ *space M* $\in$ *sets M*)
⟨*proof*⟩

lemma *in-borel-measurable-borel*:
$f \in$ *borel-measurable M* $\longleftrightarrow$
($\forall S \in$ *sets borel*.
f −' $S \cap$ *space M* $\in$ *sets M*)
⟨*proof*⟩

lemma *space-borel*[*simp*]: *space borel* = *UNIV*
⟨*proof*⟩

lemma *space-in-borel*[*measurable*]: *UNIV* $\in$ *sets borel*
⟨*proof*⟩

lemma *sets-borel*: *sets borel* = *sigma-sets UNIV* {*S. open S*}
⟨*proof*⟩

lemma *measurable-sets-borel*:
⟦$f \in$ *measurable borel M*; $A \in$ *sets M*⟧ $\Longrightarrow$ f −' $A \in$ *sets borel*
⟨*proof*⟩

lemma *pred-Collect-borel*[*measurable* (*raw*)]: *Measurable.pred borel P* $\Longrightarrow$ {*x. P x*} $\in$ *sets borel*
⟨*proof*⟩

lemma *borel-open*[*measurable* (*raw generic*)]:
assumes *open A* **shows** $A \in$ *sets borel*
⟨*proof*⟩

lemma *borel-closed*[*measurable* (*raw generic*)]:
assumes *closed A* **shows** $A \in$ *sets borel*
⟨*proof*⟩

lemma *borel-singleton*[*measurable*]:
$A \in$ *sets borel* $\Longrightarrow$ *insert x A* $\in$ *sets* (*borel* :: *'a::t1-space measure*)
⟨*proof*⟩

lemma *borel-comp*[*measurable*]: $A \in$ *sets borel* $\Longrightarrow$ − $A \in$ *sets borel*
⟨*proof*⟩

lemma *borel-measurable-vimage*:
fixes $f :: 'a \Rightarrow 'x::t2\text{-}space$
assumes *borel*[*measurable*]: $f \in$ *borel-measurable M*
shows $f -` \{x\} \cap$ *space M* $\in$ *sets M*
⟨*proof*⟩

lemma *borel-measurableI*:
fixes $f :: 'a \Rightarrow 'x::topological\text{-}space$
assumes $\bigwedge S.$ *open S* $\Longrightarrow f -` S \cap$ *space M* $\in$ *sets M*
shows $f \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-const*:
$(\lambda x.\ c) \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-indicator*:
assumes *A*: $A \in$ *sets M*
shows *indicator A* $\in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-count-space*[*measurable* (*raw*)]:
$f \in$ *borel-measurable* (*count-space S*)
⟨*proof*⟩

lemma *borel-measurable-indicator'*[*measurable* (*raw*)]:
assumes [*measurable*]: $\{x \in space\ M.\ f\ x \in A\ x\} \in$ *sets M*
shows $(\lambda x.$ *indicator* $(A\ x)\ (f\ x)) \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-indicator-iff*:
(*indicator A* $:: 'a \Rightarrow 'x::\{t1\text{-}space, zero\text{-}neq\text{-}one\}) \in$ *borel-measurable M* $\longleftrightarrow A$
$\cap$ *space M* $\in$ *sets M*
(**is** *?I* $\in$ *borel-measurable M* $\longleftrightarrow$ -)
⟨*proof*⟩

lemma *borel-measurable-subalgebra*:
assumes *sets N* $\subseteq$ *sets M space N* $=$ *space M f* $\in$ *borel-measurable N*
shows $f \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-restrict-space-iff-ereal*:
fixes $f :: 'a \Rightarrow ereal$
assumes Ω[*measurable*, *simp*]: $\Omega \cap$ *space M* $\in$ *sets M*
shows $f \in$ *borel-measurable* (*restrict-space M* Ω) $\longleftrightarrow$
$(\lambda x.\ f\ x * indicator\ \Omega\ x) \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-restrict-space-iff-ennreal*:

fixes f :: $'a \Rightarrow$ *ennreal*
assumes Ω[*measurable*, *simp*]: $\Omega \cap$ *space M* $\in$ *sets M*
shows $f \in$ *borel-measurable* (*restrict-space M* Ω) $\longleftrightarrow$
($\lambda x.$ *f x* $*$ *indicator* Ω *x*) $\in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-restrict-space-iff*:
fixes f :: $'a \Rightarrow$ $'b$::*real-normed-vector*
assumes Ω[*measurable*, *simp*]: $\Omega \cap$ *space M* $\in$ *sets M*
shows $f \in$ *borel-measurable* (*restrict-space M* Ω) $\longleftrightarrow$
($\lambda x.$ *indicator* Ω *x* $*_R$ *f x*) $\in$ *borel-measurable M*
⟨*proof*⟩

lemma *cbox-borel*[*measurable*]: *cbox a b* $\in$ *sets borel*
⟨*proof*⟩

lemma *box-borel*[*measurable*]: *box a b* $\in$ *sets borel*
⟨*proof*⟩

lemma *borel-compact*: *compact* (*A*::$'a$::*t2-space set*) $\Longrightarrow$ *A* $\in$ *sets borel*
⟨*proof*⟩

lemma *borel-sigma-sets-subset*:
A $\subseteq$ *sets borel* $\Longrightarrow$ *sigma-sets UNIV A* $\subseteq$ *sets borel*
⟨*proof*⟩

lemma *borel-eq-sigmaI1*:
fixes *F* :: $'i \Rightarrow$ $'a$::*topological-space set* **and** *X* :: $'a$::*topological-space set set*
assumes *borel-eq*: *borel* = *sigma UNIV X*
assumes *X*: $\bigwedge x.$ $x \in X \Longrightarrow x \in$ *sets* (*sigma UNIV* (*F* ‘ *A*))
assumes *F*: $\bigwedge i.$ $i \in A \Longrightarrow$ *F i* $\in$ *sets borel*
shows *borel* = *sigma UNIV* (*F* ‘ *A*)
⟨*proof*⟩

lemma *borel-eq-sigmaI2*:
fixes *F* :: $'i \Rightarrow$ $'j \Rightarrow$ $'a$::*topological-space set*
and *G* :: $'l \Rightarrow$ $'k \Rightarrow$ $'a$::*topological-space set*
assumes *borel-eq*: *borel* = *sigma UNIV* ((λ(*i*, *j*). *G i j*)‘*B*)
assumes *X*: $\bigwedge i\ j.$ (*i*, *j*) $\in B \Longrightarrow$ *G i j* $\in$ *sets* (*sigma UNIV* ((λ(*i*, *j*). *F i j*) ‘
A))
assumes *F*: $\bigwedge i\ j.$ (*i*, *j*) $\in A \Longrightarrow$ *F i j* $\in$ *sets borel*
shows *borel* = *sigma UNIV* ((λ(*i*, *j*). *F i j*) ‘ *A*)
⟨*proof*⟩

lemma *borel-eq-sigmaI3*:
fixes *F* :: $'i \Rightarrow$ $'j \Rightarrow$ $'a$::*topological-space set* **and** *X* :: $'a$::*topological-space set set*
assumes *borel-eq*: *borel* = *sigma UNIV X*
assumes *X*: $\bigwedge x.$ $x \in X \Longrightarrow x \in$ *sets* (*sigma UNIV* ((λ(*i*, *j*). *F i j*) ‘ *A*))
assumes *F*: $\bigwedge i\ j.$ (*i*, *j*) $\in A \Longrightarrow$ *F i j* $\in$ *sets borel*

shows *borel* = *sigma UNIV* ((λ(*i*, *j*). *F i j*) ‘ *A*)
⟨*proof*⟩

lemma *borel-eq-sigmaI4*:
fixes *F* :: ′*i* ⇒ ′*a*::*topological-space set*
and *G* :: ′*l* ⇒ ′*k* ⇒ ′*a*::*topological-space set*
assumes *borel-eq*: *borel* = *sigma UNIV* ((λ(*i*, *j*). *G i j*)‘*A*)
assumes *X*: ⋀*i j*. (*i*, *j*) ∈ *A* ⟹ *G i j* ∈ *sets* (*sigma UNIV* (*range F*))
assumes *F*: ⋀*i*. *F i* ∈ *sets borel*
shows *borel* = *sigma UNIV* (*range F*)
⟨*proof*⟩

lemma *borel-eq-sigmaI5*:
fixes *F* :: ′*i* ⇒ ′*j* ⇒ ′*a*::*topological-space set* **and** *G* :: ′*l* ⇒ ′*a*::*topological-space set*
assumes *borel-eq*: *borel* = *sigma UNIV* (*range G*)
assumes *X*: ⋀*i*. *G i* ∈ *sets* (*sigma UNIV* (*range* (λ(*i*, *j*). *F i j*)))
assumes *F*: ⋀*i j*. *F i j* ∈ *sets borel*
shows *borel* = *sigma UNIV* (*range* (λ(*i*, *j*). *F i j*))
⟨*proof*⟩

lemma *second-countable-borel-measurable*:
fixes *X* :: ′*a*::*second-countable-topology set set*
assumes *eq*: *open* = *generate-topology X*
shows *borel* = *sigma UNIV X*
⟨*proof*⟩

lemma *borel-eq-closed*: *borel* = *sigma UNIV* (*Collect closed*)
⟨*proof*⟩

lemma *borel-eq-countable-basis*:
fixes *B*::′*a*::*topological-space set set*
assumes *countable B*
assumes *topological-basis B*
shows *borel* = *sigma UNIV B*
⟨*proof*⟩

lemma *borel-measurable-continuous-on-restrict*:
fixes *f* :: ′*a*::*topological-space* ⇒ ′*b*::*topological-space*
assumes *f*: *continuous-on A f*
shows *f* ∈ *borel-measurable* (*restrict-space borel A*)
⟨*proof*⟩

lemma *borel-measurable-continuous-on1*: *continuous-on UNIV f* ⟹ *f* ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *borel-measurable-continuous-on-if*:
A ∈ *sets borel* ⟹ *continuous-on A f* ⟹ *continuous-on* (− *A*) *g* ⟹

(*λx. if x ∈ A then f x else g x*) ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *borel-measurable-continuous-countable-exceptions*:
fixes *f* :: *'a::t1-space* ⇒ *'b::topological-space*
assumes *X*: *countable X*
assumes *continuous-on* (− *X*) *f*
shows *f* ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *borel-measurable-continuous-on*:
assumes *f*: *continuous-on UNIV f* **and** *g*: *g* ∈ *borel-measurable M*
shows (*λx. f* (*g x*)) ∈ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-continuous-on-indicator*:
fixes *f g* :: *'a::topological-space* ⇒ *'b::real-normed-vector*
shows *A* ∈ *sets borel* ⟹ *continuous-on A f* ⟹ (*λx. indicator A x* $*_R$ *f x*) ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *borel-measurable-Pair*[*measurable* (*raw*)]:
fixes *f* :: *'a* ⇒ *'b::second-countable-topology* **and** *g* :: *'a* ⇒ *'c::second-countable-topology*
assumes *f*[*measurable*]: *f* ∈ *borel-measurable M*
assumes *g*[*measurable*]: *g* ∈ *borel-measurable M*
shows (*λx.* (*f x*, *g x*)) ∈ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-continuous-Pair*:
fixes *f* :: *'a* ⇒ *'b::second-countable-topology* **and** *g* :: *'a* ⇒ *'c::second-countable-topology*
assumes [*measurable*]: *f* ∈ *borel-measurable M*
assumes [*measurable*]: *g* ∈ *borel-measurable M*
assumes *H*: *continuous-on UNIV* (*λx. H* (*fst x*) (*snd x*))
shows (*λx. H* (*f x*) (*g x*)) ∈ *borel-measurable M*
⟨*proof*⟩

4.2 Borel spaces on order topologies

lemma [*measurable*]:
fixes *a b* :: *'a::linorder-topology*
shows *lessThan-borel*: {..< *a*} ∈ *sets borel*
and *greaterThan-borel*: {*a* <..} ∈ *sets borel*
and *greaterThanLessThan-borel*: {*a*<..<*b*} ∈ *sets borel*
and *atMost-borel*: {..*a*} ∈ *sets borel*
and *atLeast-borel*: {*a*..} ∈ *sets borel*
and *atLeastAtMost-borel*: {*a*..*b*} ∈ *sets borel*
and *greaterThanAtMost-borel*: {*a*<..*b*} ∈ *sets borel*
and *atLeastLessThan-borel*: {*a*..<*b*} ∈ *sets borel*
⟨*proof*⟩

lemma *borel-Iio*:
borel = *sigma UNIV* (*range lessThan* :: *'a*::{*linorder-topology*, *second-countable-topology*} *set set*)
⟨*proof*⟩

lemma *borel-Ioi*:
borel = *sigma UNIV* (*range greaterThan* :: *'a*::{*linorder-topology*, *second-countable-topology*} *set set*)
⟨*proof*⟩

lemma *borel-measurableI-less*:
fixes *f* :: *'a* ⇒ *'b*::{*linorder-topology*, *second-countable-topology*}
shows (⋀*y*. {*x*∈*space M*. *f x* < *y*} ∈ *sets M*) ⟹ *f* ∈ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurableI-greater*:
fixes *f* :: *'a* ⇒ *'b*::{*linorder-topology*, *second-countable-topology*}
shows (⋀*y*. {*x*∈*space M*. *y* < *f x*} ∈ *sets M*) ⟹ *f* ∈ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurableI-le*:
fixes *f* :: *'a* ⇒ *'b*::{*linorder-topology*, *second-countable-topology*}
shows (⋀*y*. {*x*∈*space M*. *f x* ≤ *y*} ∈ *sets M*) ⟹ *f* ∈ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurableI-ge*:
fixes *f* :: *'a* ⇒ *'b*::{*linorder-topology*, *second-countable-topology*}
shows (⋀*y*. {*x*∈*space M*. *y* ≤ *f x*} ∈ *sets M*) ⟹ *f* ∈ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-less*[*measurable*]:
fixes *f* :: *'a* ⇒ *'b*::{*second-countable-topology*, *dense-linorder*, *linorder-topology*}
assumes *f* ∈ *borel-measurable M*
assumes *g* ∈ *borel-measurable M*
shows {*w* ∈ *space M*. *f w* < *g w*} ∈ *sets M*
⟨*proof*⟩

lemma
fixes *f* :: *'a* ⇒ *'b*::{*second-countable-topology*, *dense-linorder*, *linorder-topology*}
assumes *f*[*measurable*]: *f* ∈ *borel-measurable M*
assumes *g*[*measurable*]: *g* ∈ *borel-measurable M*
shows *borel-measurable-le*[*measurable*]: {*w* ∈ *space M*. *f w* ≤ *g w*} ∈ *sets M*
and *borel-measurable-eq*[*measurable*]: {*w* ∈ *space M*. *f w* = *g w*} ∈ *sets M*
and *borel-measurable-neq*: {*w* ∈ *space M*. *f w* ≠ *g w*} ∈ *sets M*
⟨*proof*⟩

lemma *borel-measurable-SUP*[*measurable* (*raw*)]:
fixes *F* :: - ⇒ - ⇒ -::{*complete-linorder*, *linorder-topology*, *second-countable-topology*}

assumes [*simp*]: *countable I*
assumes [*measurable*]: $\bigwedge i.\ i \in I \Longrightarrow F\ i \in$ *borel-measurable M*
shows (λx. *SUP i:I. F i x*) $\in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-INF*[*measurable* (*raw*)]:
fixes *F* :: - ⇒ - ⇒ -::{*complete-linorder*, *linorder-topology*, *second-countable-topology*}
assumes [*simp*]: *countable I*
assumes [*measurable*]: $\bigwedge i.\ i \in I \Longrightarrow F\ i \in$ *borel-measurable M*
shows (λx. *INF i:I. F i x*) $\in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-cSUP*[*measurable* (*raw*)]:
fixes *F* :: - ⇒ - ⇒ *'a*::{*conditionally-complete-linorder*, *linorder-topology*, *second-countable-topology*}
assumes [*simp*]: *countable I*
assumes [*measurable*]: $\bigwedge i.\ i \in I \Longrightarrow F\ i \in$ *borel-measurable M*
assumes *bdd*: $\bigwedge x.\ x \in$ *space M* $\Longrightarrow$ *bdd-above* ((λi. *F i x*) ‘ *I*)
shows (λx. *SUP i:I. F i x*) $\in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-cINF*[*measurable* (*raw*)]:
fixes *F* :: - ⇒ - ⇒ *'a*::{*conditionally-complete-linorder*, *linorder-topology*, *second-countable-topology*}
assumes [*simp*]: *countable I*
assumes [*measurable*]: $\bigwedge i.\ i \in I \Longrightarrow F\ i \in$ *borel-measurable M*
assumes *bdd*: $\bigwedge x.\ x \in$ *space M* $\Longrightarrow$ *bdd-below* ((λi. *F i x*) ‘ *I*)
shows (λx. *INF i:I. F i x*) $\in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-lfp*[*consumes 1*, *case-names continuity step*]:
fixes *F* :: (*'a* ⇒ *'b*) ⇒ (*'a* ⇒ *'b*::{*complete-linorder*, *linorder-topology*, *second-countable-topology*})
assumes *sup-continuous F*
assumes ∗: $\bigwedge f.\ f \in$ *borel-measurable M* $\Longrightarrow$ *F f* $\in$ *borel-measurable M*
shows *lfp F* $\in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-gfp*[*consumes 1*, *case-names continuity step*]:
fixes *F* :: (*'a* ⇒ *'b*) ⇒ (*'a* ⇒ *'b*::{*complete-linorder*, *linorder-topology*, *second-countable-topology*})
assumes *inf-continuous F*
assumes ∗: $\bigwedge f.\ f \in$ *borel-measurable M* $\Longrightarrow$ *F f* $\in$ *borel-measurable M*
shows *gfp F* $\in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-max*[*measurable* (*raw*)]:
f $\in$ *borel-measurable M* $\Longrightarrow$ *g* $\in$ *borel-measurable M* $\Longrightarrow$ (λx. *max* (*g x*) (*f x*) :: *'b*::{*second-countable-topology*, *linorder-topology*}) $\in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-min*[*measurable* (*raw*)]:
f $\in$ *borel-measurable M* $\Longrightarrow$ *g* $\in$ *borel-measurable M* $\Longrightarrow$ (λx. *min* (*g x*) (*f x*) ::

'b::{second-countable-topology, linorder-topology}) ∈ borel-measurable M
⟨*proof*⟩

lemma *borel-measurable-Min[measurable (raw)]*:
finite I ⟹ (⋀i. i ∈ I ⟹ f i ∈ borel-measurable M) ⟹ (λx. Min ((λi. f i x)'I) :: 'b::{second-countable-topology, linorder-topology}) ∈ borel-measurable M
⟨*proof*⟩

lemma *borel-measurable-Max[measurable (raw)]*:
finite I ⟹ (⋀i. i ∈ I ⟹ f i ∈ borel-measurable M) ⟹ (λx. Max ((λi. f i x)'I) :: 'b::{second-countable-topology, linorder-topology}) ∈ borel-measurable M
⟨*proof*⟩

lemma *borel-measurable-sup[measurable (raw)]*:
f ∈ borel-measurable M ⟹ g ∈ borel-measurable M ⟹ (λx. sup (g x) (f x) :: 'b::{lattice, second-countable-topology, linorder-topology}) ∈ borel-measurable M
⟨*proof*⟩

lemma *borel-measurable-inf[measurable (raw)]*:
f ∈ borel-measurable M ⟹ g ∈ borel-measurable M ⟹ (λx. inf (g x) (f x) :: 'b::{lattice, second-countable-topology, linorder-topology}) ∈ borel-measurable M
⟨*proof*⟩

lemma *[measurable (raw)]*:
fixes *f :: nat ⇒ 'a ⇒ 'b::{complete-linorder, second-countable-topology, linorder-topology}*
assumes *⋀i. f i ∈ borel-measurable M*
shows *borel-measurable-liminf: (λx. liminf (λi. f i x)) ∈ borel-measurable M*
and *borel-measurable-limsup: (λx. limsup (λi. f i x)) ∈ borel-measurable M*
⟨*proof*⟩

lemma *measurable-convergent[measurable (raw)]*:
fixes *f :: nat ⇒ 'a ⇒ 'b::{complete-linorder, second-countable-topology, dense-linorder, linorder-topology}*
assumes *[measurable]: ⋀i. f i ∈ borel-measurable M*
shows *Measurable.pred M (λx. convergent (λi. f i x))*
⟨*proof*⟩

lemma *sets-Collect-convergent[measurable]*:
fixes *f :: nat ⇒ 'a ⇒ 'b::{complete-linorder, second-countable-topology, dense-linorder, linorder-topology}*
assumes *f[measurable]: ⋀i. f i ∈ borel-measurable M*
shows *{x∈space M. convergent (λi. f i x)} ∈ sets M*
⟨*proof*⟩

lemma *borel-measurable-lim[measurable (raw)]*:
fixes *f :: nat ⇒ 'a ⇒ 'b::{complete-linorder, second-countable-topology, dense-linorder, linorder-topology}*
assumes *[measurable]: ⋀i. f i ∈ borel-measurable M*
shows *(λx. lim (λi. f i x)) ∈ borel-measurable M*

⟨*proof*⟩

lemma *borel-measurable-LIMSEQ-order*:
fixes $u :: nat \Rightarrow 'a \Rightarrow 'b::\{complete\text{-}linorder, second\text{-}countable\text{-}topology, dense\text{-}linorder, linorder\text{-}topology\}$
assumes u': $\bigwedge x.\ x \in space\ M \Longrightarrow (\lambda i.\ u\ i\ x) \longrightarrow u'\ x$
and u: $\bigwedge i.\ u\ i \in borel\text{-}measurable\ M$
shows $u' \in borel\text{-}measurable\ M$
⟨*proof*⟩

4.3 Borel spaces on topological monoids

lemma *borel-measurable-add*[*measurable* (*raw*)]:
fixes $f\ g :: 'a \Rightarrow 'b::\{second\text{-}countable\text{-}topology, topological\text{-}monoid\text{-}add\}$
assumes f: $f \in borel\text{-}measurable\ M$
assumes g: $g \in borel\text{-}measurable\ M$
shows $(\lambda x.\ f\ x + g\ x) \in borel\text{-}measurable\ M$
⟨*proof*⟩

lemma *borel-measurable-setsum*[*measurable* (*raw*)]:
fixes $f :: 'c \Rightarrow 'a \Rightarrow 'b::\{second\text{-}countable\text{-}topology, topological\text{-}comm\text{-}monoid\text{-}add\}$
assumes $\bigwedge i.\ i \in S \Longrightarrow f\ i \in borel\text{-}measurable\ M$
shows $(\lambda x.\ \sum i{\in}S.\ f\ i\ x) \in borel\text{-}measurable\ M$
⟨*proof*⟩

lemma *borel-measurable-suminf-order*[*measurable* (*raw*)]:
fixes $f :: nat \Rightarrow 'a \Rightarrow 'b::\{complete\text{-}linorder, second\text{-}countable\text{-}topology, dense\text{-}linorder, linorder\text{-}topology, topological\text{-}comm\text{-}monoid\text{-}add\}$
assumes f[*measurable*]: $\bigwedge i.\ f\ i \in borel\text{-}measurable\ M$
shows $(\lambda x.\ suminf\ (\lambda i.\ f\ i\ x)) \in borel\text{-}measurable\ M$
⟨*proof*⟩

4.4 Borel spaces on Euclidean spaces

lemma *borel-measurable-inner*[*measurable* (*raw*)]:
fixes $f\ g :: 'a \Rightarrow 'b::\{second\text{-}countable\text{-}topology, real\text{-}inner\}$
assumes $f \in borel\text{-}measurable\ M$
assumes $g \in borel\text{-}measurable\ M$
shows $(\lambda x.\ f\ x \cdot g\ x) \in borel\text{-}measurable\ M$
⟨*proof*⟩

notation
eucl-less (**infix** $<e$ *50*)

lemma *box-oc*: $\{x.\ a <e\ x \wedge x \le b\} = \{x.\ a <e\ x\} \cap \{..b\}$
and *box-co*: $\{x.\ a \le x \wedge x <e\ b\} = \{a..\} \cap \{x.\ x <e\ b\}$
⟨*proof*⟩

lemma *eucl-ivals*[*measurable*]:
fixes $a\ b :: 'a::ordered\text{-}euclidean\text{-}space$

shows $\{x.\ x <e\ a\} \in$ *sets borel*
and $\{x.\ a <e\ x\} \in$ *sets borel*
and $\{..a\} \in$ *sets borel*
and $\{a..\} \in$ *sets borel*
and $\{a..b\} \in$ *sets borel*
and $\{x.\ a <e\ x \wedge x \le b\} \in$ *sets borel*
and $\{x.\ a \le x \wedge\ x <e\ b\} \in$ *sets borel*
⟨*proof*⟩

lemma
fixes i :: $'a$::{*second-countable-topology*, *real-inner*}
shows *hafspace-less-borel*: $\{x.\ a < x \cdot i\} \in$ *sets borel*
and *hafspace-greater-borel*: $\{x.\ x \cdot i < a\} \in$ *sets borel*
and *hafspace-less-eq-borel*: $\{x.\ a \le x \cdot i\} \in$ *sets borel*
and *hafspace-greater-eq-borel*: $\{x.\ x \cdot i \le a\} \in$ *sets borel*
⟨*proof*⟩

lemma *borel-eq-box*:
borel = *sigma UNIV* (*range* (λ (a, b). *box a b* :: $'a$ :: *euclidean-space set*))
(**is** - = *?SIGMA*)
⟨*proof*⟩

lemma *halfspace-gt-in-halfspace*:
assumes i: $i \in A$
shows $\{x::'a.\ a < x \cdot i\} \in$
sigma-sets UNIV ((λ (a, i). $\{x::'a::$*euclidean-space*$.\ x \cdot i < a\}$) ‘ (*UNIV* $\times$ A))
(**is** *?set* $\in$ *?SIGMA*)
⟨*proof*⟩

lemma *borel-eq-halfspace-less*:
borel = *sigma UNIV* ((λ(a, i). $\{x::'a::$*euclidean-space*$.\ x \cdot i < a\}$) ‘ (*UNIV* $\times$ *Basis*))
(**is** - = *?SIGMA*)
⟨*proof*⟩

lemma *borel-eq-halfspace-le*:
borel = *sigma UNIV* ((λ (a, i). $\{x::'a::$*euclidean-space*$.\ x \cdot i \le a\}$) ‘ (*UNIV* $\times$ *Basis*))
(**is** - = *?SIGMA*)
⟨*proof*⟩

lemma *borel-eq-halfspace-ge*:
borel = *sigma UNIV* ((λ (a, i). $\{x::'a::$*euclidean-space*$.\ a \le x \cdot i\}$) ‘ (*UNIV* $\times$ *Basis*))
(**is** - = *?SIGMA*)
⟨*proof*⟩

lemma *borel-eq-halfspace-greater*:

borel = *sigma UNIV* ((λ (*a*, *i*). {*x*::'*a*::*euclidean-space*. *a* < *x* · *i*}) ' (*UNIV* × *Basis*))
(**is** - = *?SIGMA*)
⟨*proof*⟩

lemma *borel-eq-atMost*:
borel = *sigma UNIV* (*range* (λ*a*. {..*a*::'*a*::*ordered-euclidean-space*}))
(**is** - = *?SIGMA*)
⟨*proof*⟩

lemma *borel-eq-greaterThan*:
borel = *sigma UNIV* (*range* (λ*a*::'*a*::*ordered-euclidean-space*. {*x*. *a* <*e* *x*}))
(**is** - = *?SIGMA*)
⟨*proof*⟩

lemma *borel-eq-lessThan*:
borel = *sigma UNIV* (*range* (λ*a*::'*a*::*ordered-euclidean-space*. {*x*. *x* <*e* *a*}))
(**is** - = *?SIGMA*)
⟨*proof*⟩

lemma *borel-eq-atLeastAtMost*:
borel = *sigma UNIV* (*range* (λ(*a*,*b*). {*a*..*b*} ::'*a*::*ordered-euclidean-space set*))
(**is** - = *?SIGMA*)
⟨*proof*⟩

lemma *borel-set-induct*[*consumes 1*, *case-names empty interval compl union*]:
assumes *A* ∈ *sets borel*
assumes *empty*: *P* {} **and** *int*: ⋀*a b*. *a* ≤ *b* ⟹ *P* {*a*..*b*} **and** *compl*: ⋀*A*. *A* ∈ *sets borel* ⟹ *P A* ⟹ *P* (−*A*) **and**
un: ⋀*f*. *disjoint-family f* ⟹ (⋀*i*. *f i* ∈ *sets borel*) ⟹ (⋀*i*. *P* (*f i*)) ⟹ *P* (⋃*i*::*nat*. *f i*)
shows *P* (*A*::*real set*)
⟨*proof*⟩

lemma *borel-sigma-sets-Ioc*: *borel* = *sigma UNIV* (*range* (λ(*a*, *b*). {*a* <.. *b*::*real*}))
⟨*proof*⟩

lemma *eucl-lessThan*: {*x*::*real*. *x* <*e* *a*} = *lessThan a*
⟨*proof*⟩

lemma *borel-eq-atLeastLessThan*:
borel = *sigma UNIV* (*range* (λ(*a*, *b*). {*a* ..< *b* :: *real*})) (**is** - = *?SIGMA*)
⟨*proof*⟩

lemma *borel-measurable-halfspacesI*:
fixes *f* :: '*a* ⇒ '*c*::*euclidean-space*
assumes *F*: *borel* = *sigma UNIV* (*F* ' (*UNIV* × *Basis*))
and *S-eq*: ⋀*a i*. *S a i* = *f* −' *F* (*a*,*i*) ∩ *space M*
shows *f* ∈ *borel-measurable M* = (∀ *i*∈*Basis*. ∀ *a*::*real*. *S a i* ∈ *sets M*)

⟨*proof*⟩

lemma *borel-measurable-iff-halfspace-le*:
fixes $f :: 'a \Rightarrow 'c::euclidean\text{-}space$
shows $f \in borel\text{-}measurable\ M = (\forall\, i{\in}Basis.\ \forall\, a.\ \{w \in space\ M.\ f\ w \cdot i \leq a\} \in sets\ M)$
⟨*proof*⟩

lemma *borel-measurable-iff-halfspace-less*:
fixes $f :: 'a \Rightarrow 'c::euclidean\text{-}space$
shows $f \in borel\text{-}measurable\ M \longleftrightarrow (\forall\, i{\in}Basis.\ \forall\, a.\ \{w \in space\ M.\ f\ w \cdot i < a\} \in sets\ M)$
⟨*proof*⟩

lemma *borel-measurable-iff-halfspace-ge*:
fixes $f :: 'a \Rightarrow 'c::euclidean\text{-}space$
shows $f \in borel\text{-}measurable\ M = (\forall\, i{\in}Basis.\ \forall\, a.\ \{w \in space\ M.\ a \leq f\ w \cdot i\} \in sets\ M)$
⟨*proof*⟩

lemma *borel-measurable-iff-halfspace-greater*:
fixes $f :: 'a \Rightarrow 'c::euclidean\text{-}space$
shows $f \in borel\text{-}measurable\ M \longleftrightarrow (\forall\, i{\in}Basis.\ \forall\, a.\ \{w \in space\ M.\ a < f\ w \cdot i\} \in sets\ M)$
⟨*proof*⟩

lemma *borel-measurable-iff-le*:
$(f::'a \Rightarrow real) \in borel\text{-}measurable\ M = (\forall\, a.\ \{w \in space\ M.\ f\ w \leq a\} \in sets\ M)$
⟨*proof*⟩

lemma *borel-measurable-iff-less*:
$(f::'a \Rightarrow real) \in borel\text{-}measurable\ M = (\forall\, a.\ \{w \in space\ M.\ f\ w < a\} \in sets\ M)$
⟨*proof*⟩

lemma *borel-measurable-iff-ge*:
$(f::'a \Rightarrow real) \in borel\text{-}measurable\ M = (\forall\, a.\ \{w \in space\ M.\ a \leq f\ w\} \in sets\ M)$
⟨*proof*⟩

lemma *borel-measurable-iff-greater*:
$(f::'a \Rightarrow real) \in borel\text{-}measurable\ M = (\forall\, a.\ \{w \in space\ M.\ a < f\ w\} \in sets\ M)$
⟨*proof*⟩

lemma *borel-measurable-euclidean-space*:
fixes $f :: 'a \Rightarrow 'c::euclidean\text{-}space$
shows $f \in borel\text{-}measurable\ M \longleftrightarrow (\forall\, i{\in}Basis.\ (\lambda x.\ f\ x \cdot i) \in borel\text{-}measurable\ M)$
⟨*proof*⟩

4.5 Borel measurable operators

lemma *borel-measurable-norm*[*measurable*]: *norm* ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *borel-measurable-sgn* [*measurable*]: (*sgn*::'*a*::*real-normed-vector* ⇒ '*a*) ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *borel-measurable-uminus*[*measurable* (*raw*)]:
fixes *g* :: '*a* ⇒ '*b*::{*second-countable-topology*, *real-normed-vector*}
assumes *g*: *g* ∈ *borel-measurable M*
shows (λ*x*. − *g x*) ∈ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-diff*[*measurable* (*raw*)]:
fixes *f* :: '*a* ⇒ '*b*::{*second-countable-topology*, *real-normed-vector*}
assumes *f*: *f* ∈ *borel-measurable M*
assumes *g*: *g* ∈ *borel-measurable M*
shows (λ*x*. *f x* − *g x*) ∈ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-times*[*measurable* (*raw*)]:
fixes *f* :: '*a* ⇒ '*b*::{*second-countable-topology*, *real-normed-algebra*}
assumes *f*: *f* ∈ *borel-measurable M*
assumes *g*: *g* ∈ *borel-measurable M*
shows (λ*x*. *f x* * *g x*) ∈ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-setprod*[*measurable* (*raw*)]:
fixes *f* :: '*c* ⇒ '*a* ⇒ '*b*::{*second-countable-topology*, *real-normed-field*}
assumes ⋀*i*. *i* ∈ *S* ⟹ *f i* ∈ *borel-measurable M*
shows (λ*x*. ∏ *i*∈*S*. *f i x*) ∈ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-dist*[*measurable* (*raw*)]:
fixes *g f* :: '*a* ⇒ '*b*::{*second-countable-topology*, *metric-space*}
assumes *f*: *f* ∈ *borel-measurable M*
assumes *g*: *g* ∈ *borel-measurable M*
shows (λ*x*. *dist* (*f x*) (*g x*)) ∈ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-scaleR*[*measurable* (*raw*)]:
fixes *g* :: '*a* ⇒ '*b*::{*second-countable-topology*, *real-normed-vector*}
assumes *f*: *f* ∈ *borel-measurable M*
assumes *g*: *g* ∈ *borel-measurable M*
shows (λ*x*. *f x* $*_R$ *g x*) ∈ *borel-measurable M*
⟨*proof*⟩

lemma *affine-borel-measurable-vector*:

fixes f :: $'a \Rightarrow 'x$::*real-normed-vector*
assumes $f \in$ *borel-measurable M*
shows $(\lambda x.\ a + b *_R f\,x) \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-const-scaleR*[*measurable* (*raw*)]:
$f \in$ *borel-measurable M* $\Longrightarrow (\lambda x.\ b *_R f\,x$ ::$'a$::*real-normed-vector*$) \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-const-add*[*measurable* (*raw*)]:
$f \in$ *borel-measurable M* $\Longrightarrow (\lambda x.\ a + f\,x$ ::$'a$::*real-normed-vector*$) \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-inverse*[*measurable* (*raw*)]:
fixes f :: $'a \Rightarrow 'b$::*real-normed-div-algebra*
assumes f: $f \in$ *borel-measurable M*
shows $(\lambda x.$ *inverse* $(f\,x)) \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-divide*[*measurable* (*raw*)]:
$f \in$ *borel-measurable M* $\Longrightarrow g \in$ *borel-measurable M* $\Longrightarrow$
$(\lambda x.\ f\,x\ /\ g\,x$::$'b$::{*second-countable-topology*, *real-normed-div-algebra*}$) \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-abs*[*measurable* (*raw*)]:
$f \in$ *borel-measurable M* $\Longrightarrow (\lambda x.\ |f\,x$ :: *real*$|) \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-nth*[*measurable* (*raw*)]:
$(\lambda x$::*real*^$'n.\ x$ \$ $i) \in$ *borel-measurable borel*
⟨*proof*⟩

lemma *convex-measurable*:
fixes A :: $'a$ :: *euclidean-space set*
shows $X \in$ *borel-measurable M* $\Longrightarrow X$ ‘ *space M* $\subseteq A \Longrightarrow$ *open A* $\Longrightarrow$ *convex-on A q* $\Longrightarrow$
$(\lambda x.\ q\ (X\ x)) \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-ln*[*measurable* (*raw*)]:
assumes f: $f \in$ *borel-measurable M*
shows $(\lambda x.$ *ln* $(f\,x$ :: *real*$)) \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-log*[*measurable* (*raw*)]:
$f \in$ *borel-measurable M* $\Longrightarrow g \in$ *borel-measurable M* $\Longrightarrow (\lambda x.$ *log* $(g\ x)\ (f\,x)) \in$

borel-measurable M
⟨*proof*⟩

lemma *borel-measurable-exp*[*measurable*]:
(*exp*::'*a*::{*real-normed-field*,*banach*}⇒'*a*) ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *measurable-real-floor*[*measurable*]:
(*floor* :: *real* ⇒ *int*) ∈ *measurable borel* (*count-space UNIV*)
⟨*proof*⟩

lemma *measurable-real-ceiling*[*measurable*]:
(*ceiling* :: *real* ⇒ *int*) ∈ *measurable borel* (*count-space UNIV*)
⟨*proof*⟩

lemma *borel-measurable-real-floor*: (λ*x*::*real*. *real-of-int* ⌊*x*⌋) ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *borel-measurable-root* [*measurable*]: *root n* ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *borel-measurable-sqrt* [*measurable*]: *sqrt* ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *borel-measurable-power* [*measurable* (*raw*)]:
fixes *f* :: - ⇒ '*b*::{*power*,*real-normed-algebra*}
assumes *f*: *f* ∈ *borel-measurable M*
shows (λ*x*. (*f x*) ^ *n*) ∈ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-Re* [*measurable*]: *Re* ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *borel-measurable-Im* [*measurable*]: *Im* ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *borel-measurable-of-real* [*measurable*]: (*of-real* :: - ⇒ (-::*real-normed-algebra*)) ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *borel-measurable-sin* [*measurable*]: (*sin* :: - ⇒ (-::{*real-normed-field*,*banach*})) ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *borel-measurable-cos* [*measurable*]: (*cos* :: - ⇒ (-::{*real-normed-field*,*banach*})) ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *borel-measurable-arctan* [*measurable*]: *arctan* ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *borel-measurable-complex-iff*:
f ∈ *borel-measurable M* ⟷
(λ*x*. *Re* (*f x*)) ∈ *borel-measurable M* ∧ (λ*x*. *Im* (*f x*)) ∈ *borel-measurable M*
⟨*proof*⟩

4.6 Borel space on the extended reals

lemma *borel-measurable-ereal*[*measurable* (*raw*)]:
assumes *f*: *f* ∈ *borel-measurable M* **shows** (λ*x*. *ereal* (*f x*)) ∈ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-real-of-ereal*[*measurable* (*raw*)]:
fixes *f* :: ′*a* ⇒ *ereal*
assumes *f*: *f* ∈ *borel-measurable M*
shows (λ*x*. *real-of-ereal* (*f x*)) ∈ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-ereal-cases*:
fixes *f* :: ′*a* ⇒ *ereal*
assumes *f*: *f* ∈ *borel-measurable M*
assumes *H*: (λ*x*. *H* (*ereal* (*real-of-ereal* (*f x*)))) ∈ *borel-measurable M*
shows (λ*x*. *H* (*f x*)) ∈ *borel-measurable M*
⟨*proof*⟩

lemma
fixes *f* :: ′*a* ⇒ *ereal* **assumes** *f*[*measurable*]: *f* ∈ *borel-measurable M*
shows *borel-measurable-ereal-abs*[*measurable*(*raw*)]: (λ*x*. |*f x*|) ∈ *borel-measurable M*
and *borel-measurable-ereal-inverse*[*measurable*(*raw*)]: (λ*x*. *inverse* (*f x*) :: *ereal*) ∈ *borel-measurable M*
and *borel-measurable-uminus-ereal*[*measurable*(*raw*)]: (λ*x*. − *f x* :: *ereal*) ∈ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-uminus-eq-ereal*[*simp*]:
(λ*x*. − *f x* :: *ereal*) ∈ *borel-measurable M* ⟷ *f* ∈ *borel-measurable M* (**is** *?l* = *?r*)
⟨*proof*⟩

lemma *set-Collect-ereal2*:
fixes *f g* :: ′*a* ⇒ *ereal*
assumes *f*: *f* ∈ *borel-measurable M*
assumes *g*: *g* ∈ *borel-measurable M*
assumes *H*: {*x* ∈ *space M*. *H* (*ereal* (*real-of-ereal* (*f x*))) (*ereal* (*real-of-ereal* (*g x*)))} ∈ *sets M*

$\{x \in space\ borel.\ H\ (-\infty)\ (ereal\ x)\} \in sets\ borel$
$\{x \in space\ borel.\ H\ (\infty)\ (ereal\ x)\} \in sets\ borel$
$\{x \in space\ borel.\ H\ (ereal\ x)\ (-\infty)\} \in sets\ borel$
$\{x \in space\ borel.\ H\ (ereal\ x)\ (\infty)\} \in sets\ borel$
shows $\{x \in space\ M.\ H\ (f\ x)\ (g\ x)\} \in sets\ M$
⟨*proof*⟩

lemma *borel-measurable-ereal-iff*:
shows $(\lambda x.\ ereal\ (f\ x)) \in borel\text{-}measurable\ M \longleftrightarrow f \in borel\text{-}measurable\ M$
⟨*proof*⟩

lemma *borel-measurable-erealD*[*measurable-dest*]:
$(\lambda x.\ ereal\ (f\ x)) \in borel\text{-}measurable\ M \Longrightarrow g \in measurable\ N\ M \Longrightarrow (\lambda x.\ f\ (g\ x)) \in borel\text{-}measurable\ N$
⟨*proof*⟩

lemma *borel-measurable-ereal-iff-real*:
fixes $f :: 'a \Rightarrow ereal$
shows $f \in borel\text{-}measurable\ M \longleftrightarrow$
$((\lambda x.\ real\text{-}of\text{-}ereal\ (f\ x)) \in borel\text{-}measurable\ M \wedge f\ -`\ \{\infty\} \cap space\ M \in sets\ M \wedge f\ -`\ \{-\infty\} \cap space\ M \in sets\ M)$
⟨*proof*⟩

lemma *borel-measurable-ereal-iff-Iio*:
$(f::'a \Rightarrow ereal) \in borel\text{-}measurable\ M \longleftrightarrow (\forall a.\ f\ -`\ \{..< a\} \cap space\ M \in sets\ M)$
⟨*proof*⟩

lemma *borel-measurable-ereal-iff-Ioi*:
$(f::'a \Rightarrow ereal) \in borel\text{-}measurable\ M \longleftrightarrow (\forall a.\ f\ -`\ \{a <..\} \cap space\ M \in sets\ M)$
⟨*proof*⟩

lemma *vimage-sets-compl-iff*:
$f\ -`\ A \cap space\ M \in sets\ M \longleftrightarrow f\ -`\ (-\ A) \cap space\ M \in sets\ M$
⟨*proof*⟩

lemma *borel-measurable-iff-Iic-ereal*:
$(f::'a \Rightarrow ereal) \in borel\text{-}measurable\ M \longleftrightarrow (\forall a.\ f\ -`\ \{..a\} \cap space\ M \in sets\ M)$
⟨*proof*⟩

lemma *borel-measurable-iff-Ici-ereal*:
$(f::'a \Rightarrow ereal) \in borel\text{-}measurable\ M \longleftrightarrow (\forall a.\ f\ -`\ \{a..\} \cap space\ M \in sets\ M)$
⟨*proof*⟩

lemma *borel-measurable-ereal2*:
fixes $f\ g :: 'a \Rightarrow ereal$
assumes f: $f \in borel\text{-}measurable\ M$
assumes g: $g \in borel\text{-}measurable\ M$

assumes H: $(\lambda x.\ H\ (ereal\ (real\text{-}of\text{-}ereal\ (f\ x)))\ (ereal\ (real\text{-}of\text{-}ereal\ (g\ x)))) \in$ *borel-measurable M*
$(\lambda x.\ H\ (-\infty)\ (ereal\ (real\text{-}of\text{-}ereal\ (g\ x)))) \in$ *borel-measurable M*
$(\lambda x.\ H\ (\infty)\ (ereal\ (real\text{-}of\text{-}ereal\ (g\ x)))) \in$ *borel-measurable M*
$(\lambda x.\ H\ (ereal\ (real\text{-}of\text{-}ereal\ (f\ x)))\ (-\infty)) \in$ *borel-measurable M*
$(\lambda x.\ H\ (ereal\ (real\text{-}of\text{-}ereal\ (f\ x)))\ (\infty)) \in$ *borel-measurable M*
shows $(\lambda x.\ H\ (f\ x)\ (g\ x)) \in$ *borel-measurable M*
⟨*proof*⟩

lemma [*measurable*(*raw*)]:
fixes *f* :: *'a* ⇒ *ereal*
assumes [*measurable*]: *f* ∈ *borel-measurable M g* ∈ *borel-measurable M*
shows *borel-measurable-ereal-add*: $(\lambda x.\ f\ x + g\ x) \in$ *borel-measurable M*
and *borel-measurable-ereal-times*: $(\lambda x.\ f\ x * g\ x) \in$ *borel-measurable M*
⟨*proof*⟩

lemma [*measurable*(*raw*)]:
fixes *f g* :: *'a* ⇒ *ereal*
assumes *f* ∈ *borel-measurable M*
assumes *g* ∈ *borel-measurable M*
shows *borel-measurable-ereal-diff*: $(\lambda x.\ f\ x - g\ x) \in$ *borel-measurable M*
and *borel-measurable-ereal-divide*: $(\lambda x.\ f\ x\ /\ g\ x) \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-ereal-setsum*[*measurable* (*raw*)]:
fixes *f* :: *'c* ⇒ *'a* ⇒ *ereal*
assumes $\bigwedge i.\ i \in S \Longrightarrow f\ i \in$ *borel-measurable M*
shows $(\lambda x.\ \sum i{\in}S.\ f\ i\ x) \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-ereal-setprod*[*measurable* (*raw*)]:
fixes *f* :: *'c* ⇒ *'a* ⇒ *ereal*
assumes $\bigwedge i.\ i \in S \Longrightarrow f\ i \in$ *borel-measurable M*
shows $(\lambda x.\ \prod i{\in}S.\ f\ i\ x) \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-extreal-suminf*[*measurable* (*raw*)]:
fixes *f* :: *nat* ⇒ *'a* ⇒ *ereal*
assumes [*measurable*]: $\bigwedge i.\ f\ i \in$ *borel-measurable M*
shows $(\lambda x.\ (\sum i.\ f\ i\ x)) \in$ *borel-measurable M*
⟨*proof*⟩

4.7 Borel space on the extended non-negative reals

ennreal is a topological monoid, so no rules for plus are required, also all order statements are usually done on type classes.

lemma *measurable-enn2ereal*[*measurable*]: *enn2ereal* ∈ *borel* $\rightarrow_M$ *borel*
⟨*proof*⟩

lemma *measurable-e2ennreal*[*measurable*]: *e2ennreal* $\in$ *borel* $\to_M$ *borel*
⟨*proof*⟩

lemma *borel-measurable-enn2real*[*measurable* (*raw*)]:
$f \in M \to_M$ *borel* $\Longrightarrow$ (λx. *enn2real* (f x)) $\in M \to_M$ *borel*
⟨*proof*⟩

definition [*simp*]: *is-borel f M* $\longleftrightarrow$ $f \in$ *borel-measurable M*

lemma *is-borel-transfer*[*transfer-rule*]: *rel-fun* (*rel-fun op* = *pcr-ennreal*) *op* = *is-borel is-borel*
⟨*proof*⟩

context
includes *ennreal.lifting*
begin

lemma *measurable-ennreal*[*measurable*]: *ennreal* $\in$ *borel* $\to_M$ *borel*
⟨*proof*⟩

lemma *borel-measurable-ennreal-iff* [*simp*]:
assumes [*simp*]: $\bigwedge x$. $x \in$ *space M* $\Longrightarrow$ $0 \le f\ x$
shows (λx. *ennreal* (f x)) $\in M \to_M$ *borel* $\longleftrightarrow$ $f \in M \to_M$ *borel*
⟨*proof*⟩

lemma *borel-measurable-times-ennreal*[*measurable* (*raw*)]:
fixes *f g* :: *'a* $\Rightarrow$ *ennreal*
shows $f \in M \to_M$ *borel* $\Longrightarrow$ $g \in M \to_M$ *borel* $\Longrightarrow$ (λx. $f\ x * g\ x$) $\in M \to_M$ *borel*
⟨*proof*⟩

lemma *borel-measurable-inverse-ennreal*[*measurable* (*raw*)]:
fixes *f* :: *'a* $\Rightarrow$ *ennreal*
shows $f \in M \to_M$ *borel* $\Longrightarrow$ (λx. *inverse* (f x)) $\in M \to_M$ *borel*
⟨*proof*⟩

lemma *borel-measurable-divide-ennreal*[*measurable* (*raw*)]:
fixes *f* :: *'a* $\Rightarrow$ *ennreal*
shows $f \in M \to_M$ *borel* $\Longrightarrow$ $g \in M \to_M$ *borel* $\Longrightarrow$ (λx. $f\ x$ / $g\ x$) $\in M \to_M$ *borel*
⟨*proof*⟩

lemma *borel-measurable-minus-ennreal*[*measurable* (*raw*)]:
fixes *f* :: *'a* $\Rightarrow$ *ennreal*
shows $f \in M \to_M$ *borel* $\Longrightarrow$ $g \in M \to_M$ *borel* $\Longrightarrow$ (λx. $f\ x - g\ x$) $\in M \to_M$ *borel*
⟨*proof*⟩

lemma *borel-measurable-setprod-ennreal*[*measurable* (*raw*)]:

fixes f :: $'c \Rightarrow 'a \Rightarrow$ *ennreal*
assumes $\bigwedge i.\ i \in S \Longrightarrow f\ i \in$ *borel-measurable M*
shows $(\lambda x.\ \prod i{\in}S.\ f\ i\ x) \in$ *borel-measurable M*
⟨*proof*⟩

end

hide-const (**open**) *is-borel*

4.8 LIMSEQ is borel measurable

lemma *borel-measurable-LIMSEQ-real*:
fixes u :: *nat* $\Rightarrow 'a \Rightarrow$ *real*
assumes u': $\bigwedge x.\ x \in$ *space M* $\Longrightarrow (\lambda i.\ u\ i\ x) \longrightarrow u'\ x$
and u: $\bigwedge i.\ u\ i \in$ *borel-measurable M*
shows $u' \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-LIMSEQ-metric*:
fixes f :: *nat* $\Rightarrow 'a \Rightarrow 'b$ :: *metric-space*
assumes [*measurable*]: $\bigwedge i.\ f\ i \in$ *borel-measurable M*
assumes *lim*: $\bigwedge x.\ x \in$ *space M* $\Longrightarrow (\lambda i.\ f\ i\ x) \longrightarrow g\ x$
shows $g \in$ *borel-measurable M*
⟨*proof*⟩

lemma *sets-Collect-Cauchy*[*measurable*]:
fixes f :: *nat* $\Rightarrow 'a$ => $'b$::{*metric-space*, *second-countable-topology*}
assumes f[*measurable*]: $\bigwedge i.\ f\ i \in$ *borel-measurable M*
shows $\{x{\in}$*space M*. *Cauchy* $(\lambda i.\ f\ i\ x)\} \in$ *sets M*
⟨*proof*⟩

lemma *borel-measurable-lim-metric*[*measurable* (*raw*)]:
fixes f :: *nat* $\Rightarrow 'a \Rightarrow 'b$::{*banach*, *second-countable-topology*}
assumes f[*measurable*]: $\bigwedge i.\ f\ i \in$ *borel-measurable M*
shows $(\lambda x.$ *lim* $(\lambda i.\ f\ i\ x)) \in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-suminf*[*measurable* (*raw*)]:
fixes f :: *nat* $\Rightarrow 'a \Rightarrow 'b$::{*banach*, *second-countable-topology*}
assumes f[*measurable*]: $\bigwedge i.\ f\ i \in$ *borel-measurable M*
shows $(\lambda x.$ *suminf* $(\lambda i.\ f\ i\ x)) \in$ *borel-measurable M*
⟨*proof*⟩

lemma *isCont-borel*:
fixes f :: $'b$::*metric-space* $\Rightarrow 'a$::*metric-space*
shows $\{x.$ *isCont f x*$\} \in$ *sets borel*
⟨*proof*⟩

lemma *isCont-borel-pred*[*measurable*]:
fixes *f* :: ′*b*::*metric-space* ⇒ ′*a*::*metric-space*
shows *Measurable.pred borel* (*isCont f*)
⟨*proof*⟩

lemma *is-real-interval*:
assumes *S*: *is-interval S*
shows ∃ *a b*::*real*. *S* = {} ∨ *S* = *UNIV* ∨ *S* = {..<*b*} ∨ *S* = {..*b*} ∨ *S* = {*a*<..} ∨ *S* = {*a*..} ∨
S = {*a*<..<*b*} ∨ *S* = {*a*<..*b*} ∨ *S* = {*a*..<*b*} ∨ *S* = {*a*..*b*}
⟨*proof*⟩

lemma *real-interval-borel-measurable*:
assumes *is-interval* (*S*::*real set*)
shows *S* ∈ *sets borel*
⟨*proof*⟩

lemma *borel-measurable-mono-on-fnc*:
fixes *f* :: *real* ⇒ *real* **and** *A* :: *real set*
assumes *mono-on f A*
shows *f* ∈ *borel-measurable* (*restrict-space borel A*)
⟨*proof*⟩

lemma *borel-measurable-mono*:
fixes *f* :: *real* ⇒ *real*
shows *mono f* ⟹ *f* ∈ *borel-measurable borel*
⟨*proof*⟩

no-notation
eucl-less (**infix** <*e 50*)

end

5 Lebesgue Integration for Nonnegative Functions

theory *Nonnegative-Lebesgue-Integration*
imports *Measure-Space Borel-Space*
begin

5.1 Simple function

Our simple functions are not restricted to nonnegative real numbers. Instead they are just functions with a finite range and are measurable when singleton sets are measurable.

definition *simple-function M g* ⟷
finite (*g* ‘ *space M*) ∧
(∀ *x* ∈ *g* ‘ *space M*. *g* −‘ {*x*} ∩ *space M* ∈ *sets M*)

lemma *simple-functionD*:
assumes *simple-function M g*
shows *finite* (*g* ‘ *space M*) **and** *g* −‘ *X* ∩ *space M* ∈ *sets M*
⟨*proof*⟩

lemma *measurable-simple-function*[*measurable-dest*]:
simple-function M f ⟹ *f* ∈ *measurable M* (*count-space UNIV*)
⟨*proof*⟩

lemma *borel-measurable-simple-function*:
simple-function M f ⟹ *f* ∈ *borel-measurable M*
⟨*proof*⟩

lemma *simple-function-measurable2*[*intro*]:
assumes *simple-function M f simple-function M g*
shows *f* −‘ *A* ∩ *g* −‘ *B* ∩ *space M* ∈ *sets M*
⟨*proof*⟩

lemma *simple-function-indicator-representation*:
fixes *f* ::′*a* ⇒ *ennreal*
assumes *f*: *simple-function M f* **and** *x*: *x* ∈ *space M*
shows *f x* = (∑ *y* ∈ *f* ‘ *space M*. *y* ∗ *indicator* (*f* −‘ {*y*} ∩ *space M*) *x*)
(**is** *?l* = *?r*)
⟨*proof*⟩

lemma *simple-function-notspace*:
simple-function M (λ*x*. *h x* ∗ *indicator* (− *space M*) *x*::*ennreal*) (**is** *simple-function M ?h*)
⟨*proof*⟩

lemma *simple-function-cong*:
assumes ⋀*t*. *t* ∈ *space M* ⟹ *f t* = *g t*
shows *simple-function M f* ⟷ *simple-function M g*
⟨*proof*⟩

lemma *simple-function-cong-algebra*:
assumes *sets N* = *sets M space N* = *space M*
shows *simple-function M f* ⟷ *simple-function N f*
⟨*proof*⟩

lemma *simple-function-borel-measurable*:
fixes *f* :: ′*a* ⇒ ′*x*::{*t2-space*}
assumes *f* ∈ *borel-measurable M* **and** *finite* (*f* ‘ *space M*)
shows *simple-function M f*
⟨*proof*⟩

lemma *simple-function-iff-borel-measurable*:
fixes *f* :: ′*a* ⇒ ′*x*::{*t2-space*}
shows *simple-function M f* ⟷ *finite* (*f* ‘ *space M*) ∧ *f* ∈ *borel-measurable M*

⟨*proof*⟩

lemma *simple-function-eq-measurable*:
simple-function M f ⟷ *finite* (*f'space M*) ∧ *f* ∈ *measurable M* (*count-space UNIV*)
⟨*proof*⟩

lemma *simple-function-const*[*intro*, *simp*]:
simple-function M (λ*x*. *c*)
⟨*proof*⟩
lemma *simple-function-compose*[*intro*, *simp*]:
assumes *simple-function M f*
shows *simple-function M* (*g* ∘ *f*)
⟨*proof*⟩

lemma *simple-function-indicator*[*intro*, *simp*]:
assumes *A* ∈ *sets M*
shows *simple-function M* (*indicator A*)
⟨*proof*⟩

lemma *simple-function-Pair*[*intro*, *simp*]:
assumes *simple-function M f*
assumes *simple-function M g*
shows *simple-function M* (λ*x*. (*f x*, *g x*)) (**is** *simple-function M ?p*)
⟨*proof*⟩

lemma *simple-function-compose1*:
assumes *simple-function M f*
shows *simple-function M* (λ*x*. *g* (*f x*))
⟨*proof*⟩

lemma *simple-function-compose2*:
assumes *simple-function M f* **and** *simple-function M g*
shows *simple-function M* (λ*x*. *h* (*f x*) (*g x*))
⟨*proof*⟩

lemmas *simple-function-add*[*intro*, *simp*] = *simple-function-compose2*[**where** *h*=*op* +]
and *simple-function-diff*[*intro*, *simp*] = *simple-function-compose2*[**where** *h*=*op* −]
and *simple-function-uminus*[*intro*, *simp*] = *simple-function-compose*[**where** *g*=*uminus*]
and *simple-function-mult*[*intro*, *simp*] = *simple-function-compose2*[**where** *h*=*op* *]
and *simple-function-div*[*intro*, *simp*] = *simple-function-compose2*[**where** *h*=*op* /]
and *simple-function-inverse*[*intro*, *simp*] = *simple-function-compose*[**where** *g*=*inverse*]
and *simple-function-max*[*intro*, *simp*] = *simple-function-compose2*[**where** *h*=*max*]

lemma *simple-function-setsum*[*intro*, *simp*]:

assumes $\bigwedge i.\ i \in P \Longrightarrow$ *simple-function M* $(f\ i)$
shows *simple-function M* $(\lambda x.\ \sum i{\in}P.\ f\ i\ x)$
⟨*proof*⟩

lemma *simple-function-ennreal*[*intro*, *simp*]:
fixes $f\ g :: {'a} \Rightarrow real$ **assumes** *sf*: *simple-function M f*
shows *simple-function M* $(\lambda x.\ ennreal\ (f\ x))$
⟨*proof*⟩

lemma *simple-function-real-of-nat*[*intro*, *simp*]:
fixes $f\ g :: {'a} \Rightarrow nat$ **assumes** *sf*: *simple-function M f*
shows *simple-function M* $(\lambda x.\ real\ (f\ x))$
⟨*proof*⟩

lemma *borel-measurable-implies-simple-function-sequence*:
fixes $u :: {'a} \Rightarrow ennreal$
assumes *u*[*measurable*]: $u \in$ *borel-measurable M*
shows $\exists f.\ incseq\ f \wedge (\forall i.\ (\forall x.\ f\ i\ x < top) \wedge$ *simple-function M* $(f\ i)) \wedge u = (SUP\ i.\ f\ i)$
⟨*proof*⟩

lemma *borel-measurable-implies-simple-function-sequence'*:
fixes $u :: {'a} \Rightarrow ennreal$
assumes *u*: $u \in$ *borel-measurable M*
obtains *f* **where**
$\bigwedge i.$ *simple-function M* $(f\ i)$ *incseq f* $\bigwedge i\ x.\ f\ i\ x < top$ $\bigwedge x.\ (SUP\ i.\ f\ i\ x) = u\ x$
⟨*proof*⟩

lemma *simple-function-induct*[*consumes 1*, *case-names cong set mult add*, *induct set*: *simple-function*]:
fixes $u :: {'a} \Rightarrow ennreal$
assumes *u*: *simple-function M u*
assumes *cong*: $\bigwedge f\ g.$ *simple-function M f* $\Longrightarrow$ *simple-function M g* $\Longrightarrow (AE\ x\ in\ M.\ f\ x = g\ x) \Longrightarrow P\ f \Longrightarrow P\ g$
assumes *set*: $\bigwedge A.\ A \in sets\ M \Longrightarrow P\ (indicator\ A)$
assumes *mult*: $\bigwedge u\ c.\ P\ u \Longrightarrow P\ (\lambda x.\ c * u\ x)$
assumes *add*: $\bigwedge u\ v.\ P\ u \Longrightarrow P\ v \Longrightarrow P\ (\lambda x.\ v\ x + u\ x)$
shows $P\ u$
⟨*proof*⟩

lemma *simple-function-induct-nn*[*consumes 1*, *case-names cong set mult add*]:
fixes $u :: {'a} \Rightarrow ennreal$
assumes *u*: *simple-function M u*
assumes *cong*: $\bigwedge f\ g.$ *simple-function M f* $\Longrightarrow$ *simple-function M g* $\Longrightarrow (\bigwedge x.\ x \in space\ M \Longrightarrow f\ x = g\ x) \Longrightarrow P\ f \Longrightarrow P\ g$
assumes *set*: $\bigwedge A.\ A \in sets\ M \Longrightarrow P\ (indicator\ A)$
assumes *mult*: $\bigwedge u\ c.$ *simple-function M u* $\Longrightarrow P\ u \Longrightarrow P\ (\lambda x.\ c * u\ x)$
assumes *add*: $\bigwedge u\ v.$ *simple-function M u* $\Longrightarrow P\ u \Longrightarrow$ *simple-function M v* $\Longrightarrow (\bigwedge x.\ x \in space\ M \Longrightarrow u\ x = 0 \vee v\ x = 0) \Longrightarrow P\ v \Longrightarrow P\ (\lambda x.\ v\ x + u\ x)$

shows *P u*
⟨*proof*⟩

lemma *borel-measurable-induct*[*consumes 1*, *case-names cong set mult add seq*, *induct set*: *borel-measurable*]:
fixes $u :: 'a \Rightarrow ennreal$
assumes *u*: $u \in borel\text{-}measurable\ M$
assumes *cong*: $\bigwedge f\ g.\ f \in borel\text{-}measurable\ M \Longrightarrow g \in borel\text{-}measurable\ M \Longrightarrow (\bigwedge x.\ x \in space\ M \Longrightarrow f\ x = g\ x) \Longrightarrow P\ g \Longrightarrow P\ f$
assumes *set*: $\bigwedge A.\ A \in sets\ M \Longrightarrow P\ (indicator\ A)$
assumes *mult'*: $\bigwedge u\ c.\ c < top \Longrightarrow u \in borel\text{-}measurable\ M \Longrightarrow (\bigwedge x.\ x \in space\ M \Longrightarrow u\ x < top) \Longrightarrow P\ u \Longrightarrow P\ (\lambda x.\ c * u\ x)$
assumes *add*: $\bigwedge u\ v.\ u \in borel\text{-}measurable\ M \Longrightarrow (\bigwedge x.\ x \in space\ M \Longrightarrow u\ x < top) \Longrightarrow P\ u \Longrightarrow v \in borel\text{-}measurable\ M \Longrightarrow (\bigwedge x.\ x \in space\ M \Longrightarrow v\ x < top) \Longrightarrow (\bigwedge x.\ x \in space\ M \Longrightarrow u\ x = 0 \lor v\ x = 0) \Longrightarrow P\ v \Longrightarrow P\ (\lambda x.\ v\ x + u\ x)$
assumes *seq*: $\bigwedge U.\ (\bigwedge i.\ U\ i \in borel\text{-}measurable\ M) \Longrightarrow (\bigwedge i\ x.\ x \in space\ M \Longrightarrow U\ i\ x < top) \Longrightarrow (\bigwedge i.\ P\ (U\ i)) \Longrightarrow incseq\ U \Longrightarrow u = (SUP\ i.\ U\ i) \Longrightarrow P\ (SUP\ i.\ U\ i)$
shows *P u*
⟨*proof*⟩

lemma *simple-function-If-set*:
assumes *sf*: *simple-function M f simple-function M g* **and** *A*: $A \cap space\ M \in sets\ M$
shows *simple-function M* $(\lambda x.\ if\ x \in A\ then\ f\ x\ else\ g\ x)$ (**is** *simple-function M ?IF*)
⟨*proof*⟩

lemma *simple-function-If*:
assumes *sf*: *simple-function M f simple-function M g* **and** *P*: $\{x{\in}space\ M.\ P\ x\} \in sets\ M$
shows *simple-function M* $(\lambda x.\ if\ P\ x\ then\ f\ x\ else\ g\ x)$
⟨*proof*⟩

lemma *simple-function-subalgebra*:
assumes *simple-function N f*
and *N-subalgebra*: $sets\ N \subseteq sets\ M\ space\ N = space\ M$
shows *simple-function M f*
⟨*proof*⟩

lemma *simple-function-comp*:
assumes *T*: $T \in measurable\ M\ M'$
and *f*: *simple-function M' f*
shows *simple-function M* $(\lambda x.\ f\ (T\ x))$
⟨*proof*⟩

5.2 Simple integral

definition *simple-integral* :: *'a measure* ⇒ (*'a* ⇒ *ennreal*) ⇒ *ennreal* ($integral^S$)
where
$integral^S$ *M f* = (∑ *x* ∈ *f* ' *space M*. *x* ∗ *emeasure M* (*f* −' {*x*} ∩ *space M*))

syntax
-simple-integral :: *pttrn* ⇒ *ennreal* ⇒ *'a measure* ⇒ *ennreal* (∫ ^{S} -. - ∂- [*60*,*61*] *110*)

translations
∫ ^{S} *x*. *f* ∂*M* == *CONST simple-integral M* (%*x*. *f*)

lemma *simple-integral-cong*:
assumes ⋀*t*. *t* ∈ *space M* ⟹ *f t* = *g t*
shows $integral^S$ *M f* = $integral^S$ *M g*
⟨*proof*⟩

lemma *simple-integral-const*[*simp*]:
(∫ ^{S}*x*. *c* ∂*M*) = *c* ∗ (*emeasure M*) (*space M*)
⟨*proof*⟩

lemma *simple-function-partition*:
assumes *f*: *simple-function M f* **and** *g*: *simple-function M g*
assumes *sub*: ⋀*x y*. *x* ∈ *space M* ⟹ *y* ∈ *space M* ⟹ *g x* = *g y* ⟹ *f x* = *f y*
assumes *v*: ⋀*x*. *x* ∈ *space M* ⟹ *f x* = *v* (*g x*)
shows $integral^S$ *M f* = (∑ *y*∈*g* ' *space M*. *v y* ∗ *emeasure M* {*x*∈*space M*. *g x* = *y*})
(**is** - = *?r*)
⟨*proof*⟩

lemma *simple-integral-add*[*simp*]:
assumes *f*: *simple-function M f* **and** ⋀*x*. *0* ≤ *f x* **and** *g*: *simple-function M g* **and** ⋀*x*. *0* ≤ *g x*
shows (∫ ^{S}*x*. *f x* + *g x* ∂*M*) = $integral^S$ *M f* + $integral^S$ *M g*
⟨*proof*⟩

lemma *simple-integral-setsum*[*simp*]:
assumes ⋀*i x*. *i* ∈ *P* ⟹ *0* ≤ *f i x*
assumes ⋀*i*. *i* ∈ *P* ⟹ *simple-function M* (*f i*)
shows (∫ ^{S}*x*. (∑ *i*∈*P*. *f i x*) ∂*M*) = (∑ *i*∈*P*. $integral^S$ *M* (*f i*))
⟨*proof*⟩

lemma *simple-integral-mult*[*simp*]:
assumes *f*: *simple-function M f*
shows (∫ ^{S}*x*. *c* ∗ *f x* ∂*M*) = *c* ∗ $integral^S$ *M f*
⟨*proof*⟩

lemma *simple-integral-mono-AE*:
assumes *f*[*measurable*]: *simple-function M f* **and** *g*[*measurable*]: *simple-function*

M g
  and *mono*: *AE x in M*. *f x* ≤ *g x*
  shows $integral^S$ *M f* ≤ $integral^S$ *M g*
⟨*proof*⟩

lemma *simple-integral-mono*:
  assumes *simple-function M f* **and** *simple-function M g*
  and *mono*: $\bigwedge$ *x*. *x* ∈ *space M* ⟹ *f x* ≤ *g x*
  shows $integral^S$ *M f* ≤ $integral^S$ *M g*
  ⟨*proof*⟩

lemma *simple-integral-cong-AE*:
  assumes *simple-function M f* **and** *simple-function M g*
  and *AE x in M*. *f x* = *g x*
  shows $integral^S$ *M f* = $integral^S$ *M g*
  ⟨*proof*⟩

lemma *simple-integral-cong'*:
  assumes *sf*: *simple-function M f simple-function M g*
  and *mea*: (*emeasure M*) {*x*∈*space M*. *f x* ≠ *g x*} = *0*
  shows $integral^S$ *M f* = $integral^S$ *M g*
⟨*proof*⟩

lemma *simple-integral-indicator*:
  assumes *A*: *A* ∈ *sets M*
  assumes *f*: *simple-function M f*
  shows ($\int^S$*x*. *f x* ∗ *indicator A x* ∂*M*) =
    ($\sum$ *x* ∈ *f* ' *space M*. *x* ∗ *emeasure M* (*f* −' {*x*} ∩ *space M* ∩ *A*))
⟨*proof*⟩

lemma *simple-integral-indicator-only*[*simp*]:
  assumes *A* ∈ *sets M*
  shows $integral^S$ *M* (*indicator A*) = *emeasure M A*
  ⟨*proof*⟩

lemma *simple-integral-null-set*:
  assumes *simple-function M u* $\bigwedge$*x*. *0* ≤ *u x* **and** *N* ∈ *null-sets M*
  shows ($\int^S$*x*. *u x* ∗ *indicator N x* ∂*M*) = *0*
⟨*proof*⟩

lemma *simple-integral-cong-AE-mult-indicator*:
  assumes *sf*: *simple-function M f* **and** *eq*: *AE x in M*. *x* ∈ *S* **and** *S* ∈ *sets M*
  shows $integral^S$ *M f* = ($\int^S$*x*. *f x* ∗ *indicator S x* ∂*M*)
  ⟨*proof*⟩

lemma *simple-integral-cmult-indicator*:
  assumes *A*: *A* ∈ *sets M*
  shows ($\int^S$*x*. *c* ∗ *indicator A x* ∂*M*) = *c* ∗ *emeasure M A*
  ⟨*proof*⟩

lemma *simple-integral-nonneg*:
assumes *f*: *simple-function M f* **and** *ae*: *AE x in M. 0 ≤ f x*
shows *0 ≤ integral^S M f*
⟨*proof*⟩

5.3 Integral on nonnegative functions

definition *nn-integral* :: *'a measure ⇒ ('a ⇒ ennreal) ⇒ ennreal* (*integral^N*)
where
integral^N M f = (SUP g : {g. simple-function M g ∧ g ≤ f}. integral^S M g)

syntax
-nn-integral :: *pttrn ⇒ ennreal ⇒ 'a measure ⇒ ennreal* (*∫ $^+$((2 -./ -)/ ∂-)* [*60,61*] *110*)

translations
∫ $^+$x. f ∂M == CONST nn-integral M (λx. f)

lemma *nn-integral-def-finite*:
integral^N M f = (SUP g : {g. simple-function M g ∧ g ≤ f ∧ (∀ x. g x < top)}. integral^S M g)
(**is** - = *SUPREMUM ?A ?f*)
⟨*proof*⟩

lemma *nn-integral-mono-AE*:
assumes *ae*: *AE x in M. u x ≤ v x* **shows** *integral^N M u ≤ integral^N M v*
⟨*proof*⟩

lemma *nn-integral-mono*:
(⋀x. x ∈ space M ⟹ u x ≤ v x) ⟹ integral^N M u ≤ integral^N M v
⟨*proof*⟩

lemma *mono-nn-integral*: *mono F ⟹ mono (λx. integral^N M (F x))*
⟨*proof*⟩

lemma *nn-integral-cong-AE*:
AE x in M. u x = v x ⟹ integral^N M u = integral^N M v
⟨*proof*⟩

lemma *nn-integral-cong*:
(⋀x. x ∈ space M ⟹ u x = v x) ⟹ integral^N M u = integral^N M v
⟨*proof*⟩

lemma *nn-integral-cong-simp*:
(⋀x. x ∈ space M =simp=> u x = v x) ⟹ integral^N M u = integral^N M v
⟨*proof*⟩

lemma *nn-integral-cong-strong*:

$M = N \Longrightarrow (\bigwedge x.\ x \in space\ M \Longrightarrow u\ x = v\ x) \Longrightarrow integral^N\ M\ u = integral^N\ N\ v$
⟨proof⟩

lemma *incseq-nn-integral*:
assumes *incseq f* **shows** $incseq\ (\lambda i.\ integral^N\ M\ (f\ i))$
⟨proof⟩

lemma *nn-integral-eq-simple-integral*:
assumes *f*: *simple-function M f* **shows** $integral^N\ M\ f = integral^S\ M\ f$
⟨proof⟩

Beppo-Levi monotone convergence theorem

lemma *nn-integral-monotone-convergence-SUP*:
assumes *f*: *incseq f* **and** [*measurable*]: $\bigwedge i.\ f\ i \in borel\text{-}measurable\ M$
shows $(\int^+ x.\ (SUP\ i.\ f\ i\ x)\ \partial M) = (SUP\ i.\ integral^N\ M\ (f\ i))$
⟨proof⟩

lemma *sup-continuous-nn-integral*[*order-continuous-intros*]:
assumes *f*: $\bigwedge y.\ sup\text{-}continuous\ (f\ y)$
assumes [*measurable*]: $\bigwedge x.\ (\lambda y.\ f\ y\ x) \in borel\text{-}measurable\ M$
shows $sup\text{-}continuous\ (\lambda x.\ (\int^+ y.\ f\ y\ x\ \partial M))$
⟨proof⟩

lemma *nn-integral-monotone-convergence-SUP-AE*:
assumes *f*: $\bigwedge i.\ AE\ x\ in\ M.\ f\ i\ x \le f\ (Suc\ i)\ x$ $\bigwedge i.\ f\ i \in borel\text{-}measurable\ M$
shows $(\int^+ x.\ (SUP\ i.\ f\ i\ x)\ \partial M) = (SUP\ i.\ integral^N\ M\ (f\ i))$
⟨proof⟩

lemma *nn-integral-monotone-convergence-simple*:
$incseq\ f \Longrightarrow (\bigwedge i.\ simple\text{-}function\ M\ (f\ i)) \Longrightarrow (SUP\ i.\ \int^S x.\ f\ i\ x\ \partial M) = (\int^+ x.\ (SUP\ i.\ f\ i\ x)\ \partial M)$
⟨proof⟩

lemma *SUP-simple-integral-sequences*:
assumes *f*: *incseq f* $\bigwedge i.\ simple\text{-}function\ M\ (f\ i)$
and *g*: *incseq g* $\bigwedge i.\ simple\text{-}function\ M\ (g\ i)$
and *eq*: $AE\ x\ in\ M.\ (SUP\ i.\ f\ i\ x) = (SUP\ i.\ g\ i\ x)$
shows $(SUP\ i.\ integral^S\ M\ (f\ i)) = (SUP\ i.\ integral^S\ M\ (g\ i))$
(**is** *SUPREMUM - ?F = SUPREMUM - ?G*)
⟨proof⟩

lemma *nn-integral-const*[*simp*]: $(\int^+ x.\ c\ \partial M) = c * emeasure\ M\ (space\ M)$
⟨proof⟩

lemma *nn-integral-linear*:
assumes *f*: $f \in borel\text{-}measurable\ M$ **and** *g*: $g \in borel\text{-}measurable\ M$
shows $(\int^+ x.\ a * f\ x + g\ x\ \partial M) = a * integral^N\ M\ f + integral^N\ M\ g$
(**is** $integral^N\ M\ ?L = -$)

⟨*proof*⟩

lemma *nn-integral-cmult*: $f \in$ *borel-measurable* $M \Longrightarrow (\int^+ x.\ c * f\,x\ \partial M) = c *$ *integral*^{N} $M\,f$
⟨*proof*⟩

lemma *nn-integral-multc*: $f \in$ *borel-measurable* $M \Longrightarrow (\int^+ x.\ f\,x * c\ \partial M) =$ *integral*^{N} $M\,f * c$
⟨*proof*⟩

lemma *nn-integral-divide*: $f \in$ *borel-measurable* $M \Longrightarrow (\int^+ x.\ f\,x\ /\ c\ \partial M) = (\int^+ x.\ f\,x\ \partial M)\ /\ c$
⟨*proof*⟩

lemma *nn-integral-indicator*[*simp*]: $A \in$ *sets* $M \Longrightarrow (\int^+ x.$ *indicator* $A\ x \partial M) =$ (*emeasure* M) A
⟨*proof*⟩

lemma *nn-integral-cmult-indicator*: $A \in$ *sets* $M \Longrightarrow (\int^+ x.\ c *$ *indicator* $A\ x\ \partial M) = c *$ *emeasure* $M\ A$
⟨*proof*⟩

lemma *nn-integral-indicator'*:
assumes [*measurable*]: $A \cap$ *space* $M \in$ *sets* M
shows $(\int^+ x.$ *indicator* $A\ x\ \partial M) =$ *emeasure* M ($A \cap$ *space* M)
⟨*proof*⟩

lemma *nn-integral-indicator-singleton*[*simp*]:
assumes [*measurable*]: $\{y\} \in$ *sets* M **shows** $(\int^+ x.\ f\,x *$ *indicator* $\{y\}\ x\ \partial M) = f\,y *$ *emeasure* $M\ \{y\}$
⟨*proof*⟩

lemma *nn-integral-set-ennreal*:
$(\int^+ x.$ *ennreal* $(f\,x) *$ *indicator* $A\ x\ \partial M) = (\int^+ x.$ *ennreal* $(f\,x *$ *indicator* $A\ x)\ \partial M)$
⟨*proof*⟩

lemma *nn-integral-indicator-singleton'*[*simp*]:
assumes [*measurable*]: $\{y\} \in$ *sets* M
shows $(\int^+ x.$ *ennreal* $(f\,x *$ *indicator* $\{y\}\ x)\ \partial M) = f\,y *$ *emeasure* $M\ \{y\}$
⟨*proof*⟩

lemma *nn-integral-add*:
$f \in$ *borel-measurable* $M \Longrightarrow g \in$ *borel-measurable* $M \Longrightarrow (\int^+ x.\ f\,x + g\,x\ \partial M) =$ *integral*^{N} $M\,f$ + *integral*^{N} $M\,g$
⟨*proof*⟩

lemma *nn-integral-setsum*:
$(\bigwedge i.\ i \in P \Longrightarrow f\ i \in$ *borel-measurable* $M) \Longrightarrow (\int^+ x.\ (\sum i{\in}P.\ f\ i\ x)\ \partial M) =$

($\sum i{\in}P.$ *integral*^{N} *M* (*f i*))
⟨*proof*⟩

lemma *nn-integral-suminf*:
assumes *f*: $\bigwedge i.\ f\ i \in$ *borel-measurable M*
shows ($\int^+ x.\ (\sum i.\ f\ i\ x)\ \partial M$) = ($\sum i.$ *integral*^{N} *M* (*f i*))
⟨*proof*⟩

lemma *nn-integral-bound-simple-function*:
assumes *bnd*: $\bigwedge x.\ x \in$ *space M* $\Longrightarrow f\ x < \infty$
assumes *f*[*measurable*]: *simple-function M f*
assumes *supp*: *emeasure M* {*x*∈*space M*. *f x* ≠ *0*} < ∞
shows *nn-integral M f* < ∞
⟨*proof*⟩

lemma *nn-integral-Markov-inequality*:
assumes *u*: *u* ∈ *borel-measurable M* **and** *A* ∈ *sets M*
shows (*emeasure M*) ({*x*∈*space M*. *1* ≤ *c* ∗ *u x*} ∩ *A*) ≤ *c* ∗ ($\int^+ x.\ u\ x\ *$ *indicator A x* ∂*M*)
(**is** (*emeasure M*) *?A* ≤ - ∗ *?PI*)
⟨*proof*⟩

lemma *nn-integral-noteq-infinite*:
assumes *g*: *g* ∈ *borel-measurable M* **and** *integral*^{N} *M g* ≠ ∞
shows *AE x in M*. *g x* ≠ ∞
⟨*proof*⟩

lemma *nn-integral-PInf*:
assumes *f*: *f* ∈ *borel-measurable M* **and** *not-Inf*: *integral*^{N} *M f* ≠ ∞
shows *emeasure M* (*f* −‘ {∞} ∩ *space M*) = *0*
⟨*proof*⟩

lemma *simple-integral-PInf*:
simple-function M f ⟹ *integral*^{S} *M f* ≠ ∞ ⟹ *emeasure M* (*f* −‘ {∞} ∩ *space M*) = *0*
⟨*proof*⟩

lemma *nn-integral-PInf-AE*:
assumes *f* ∈ *borel-measurable M integral*^{N} *M f* ≠ ∞ **shows** *AE x in M*. *f x* ≠ ∞
⟨*proof*⟩

lemma *nn-integral-diff*:
assumes *f*: *f* ∈ *borel-measurable M*
and *g*: *g* ∈ *borel-measurable M*
and *fin*: *integral*^{N} *M g* ≠ ∞
and *mono*: *AE x in M*. *g x* ≤ *f x*
shows ($\int^+ x.\ f\ x - g\ x\ \partial M$) = *integral*^{N} *M f* − *integral*^{N} *M g*
⟨*proof*⟩

lemma *nn-integral-mult-bounded-inf*:
assumes f: $f \in$ *borel-measurable* M $(\int^{+} x.\ f\ x\ \partial M) < \infty$ **and** c: $c \neq \infty$ **and** *ae*: $AE\ x\ in\ M.\ g\ x \leq c * f\ x$
shows $(\int^{+} x.\ g\ x\ \partial M) < \infty$
⟨*proof*⟩

Fatou's lemma: convergence theorem on limes inferior

lemma *nn-integral-monotone-convergence-INF-AE'*:
assumes f: $\bigwedge i.\ AE\ x\ in\ M.\ f\ (Suc\ i)\ x \leq f\ i\ x$ **and** [*measurable*]: $\bigwedge i.\ f\ i \in$ *borel-measurable* M
and $*$: $(\int^{+} x.\ f\ 0\ x\ \partial M) < \infty$
shows $(\int^{+} x.\ (INF\ i.\ f\ i\ x)\ \partial M) = (INF\ i.\ integral^{N}\ M\ (f\ i))$
⟨*proof*⟩

lemma *nn-integral-monotone-convergence-INF-AE*:
fixes f :: *nat* ⇒ *'a* ⇒ *ennreal*
assumes f: $\bigwedge i.\ AE\ x\ in\ M.\ f\ (Suc\ i)\ x \leq f\ i\ x$
and [*measurable*]: $\bigwedge i.\ f\ i \in$ *borel-measurable* M
and *fin*: $(\int^{+} x.\ f\ i\ x\ \partial M) < \infty$
shows $(\int^{+} x.\ (INF\ i.\ f\ i\ x)\ \partial M) = (INF\ i.\ integral^{N}\ M\ (f\ i))$
⟨*proof*⟩

lemma *nn-integral-monotone-convergence-INF-decseq*:
assumes f: *decseq* f **and** $*$: $\bigwedge i.\ f\ i \in$ *borel-measurable* M $(\int^{+} x.\ f\ i\ x\ \partial M) < \infty$
shows $(\int^{+} x.\ (INF\ i.\ f\ i\ x)\ \partial M) = (INF\ i.\ integral^{N}\ M\ (f\ i))$
⟨*proof*⟩

lemma *nn-integral-liminf*:
fixes u :: *nat* ⇒ *'a* ⇒ *ennreal*
assumes u: $\bigwedge i.\ u\ i \in$ *borel-measurable* M
shows $(\int^{+} x.\ liminf\ (\lambda n.\ u\ n\ x)\ \partial M) \leq liminf\ (\lambda n.\ integral^{N}\ M\ (u\ n))$
⟨*proof*⟩

lemma *nn-integral-limsup*:
fixes u :: *nat* ⇒ *'a* ⇒ *ennreal*
assumes [*measurable*]: $\bigwedge i.\ u\ i \in$ *borel-measurable* M $w \in$ *borel-measurable* M
assumes *bounds*: $\bigwedge i.\ AE\ x\ in\ M.\ u\ i\ x \leq w\ x$ **and** w: $(\int^{+} x.\ w\ x\ \partial M) < \infty$
shows $limsup\ (\lambda n.\ integral^{N}\ M\ (u\ n)) \leq (\int^{+} x.\ limsup\ (\lambda n.\ u\ n\ x)\ \partial M)$
⟨*proof*⟩

lemma *nn-integral-LIMSEQ*:
assumes f: *incseq* f $\bigwedge i.\ f\ i \in$ *borel-measurable* M
and u: $\bigwedge x.\ (\lambda i.\ f\ i\ x) \longrightarrow u\ x$
shows $(\lambda n.\ integral^{N}\ M\ (f\ n)) \longrightarrow integral^{N}\ M\ u$
⟨*proof*⟩

lemma *nn-integral-dominated-convergence*:

assumes [*measurable*]:
$\bigwedge i.\ u\ i \in$ *borel-measurable M* $u' \in$ *borel-measurable M* $w \in$ *borel-measurable M*
and *bound*: $\bigwedge j.\ AE\ x\ in\ M.\ u\ j\ x \le w\ x$
and *w*: $(\int^+ x.\ w\ x\ \partial M) < \infty$
and *u'*: $AE\ x\ in\ M.\ (\lambda i.\ u\ i\ x) \longrightarrow u'\ x$
shows $(\lambda i.\ (\int^+ x.\ u\ i\ x\ \partial M)) \longrightarrow (\int^+ x.\ u'\ x\ \partial M)$
⟨*proof*⟩

lemma *inf-continuous-nn-integral*[*order-continuous-intros*]:
assumes *f*: $\bigwedge y.$ *inf-continuous* $(f\ y)$
assumes [*measurable*]: $\bigwedge x.\ (\lambda y.\ f\ y\ x) \in$ *borel-measurable M*
assumes *bnd*: $\bigwedge x.\ (\int^+ y.\ f\ y\ x\ \partial M) \neq \infty$
shows *inf-continuous* $(\lambda x.\ (\int^+ y.\ f\ y\ x\ \partial M))$
⟨*proof*⟩

lemma *nn-integral-null-set*:
assumes $N \in$ *null-sets M* **shows** $(\int^+ x.\ u\ x * \text{indicator}\ N\ x\ \partial M) = 0$
⟨*proof*⟩

lemma *nn-integral-0-iff*:
assumes *u*: $u \in$ *borel-measurable M*
shows $\text{integral}^N\ M\ u = 0 \longleftrightarrow \text{emeasure}\ M\ \{x{\in}\text{space}\ M.\ u\ x \neq 0\} = 0$
(**is** - $\longleftrightarrow$ (*emeasure M*) *?A* = *0*)
⟨*proof*⟩

lemma *nn-integral-0-iff-AE*:
assumes *u*: $u \in$ *borel-measurable M*
shows $\text{integral}^N\ M\ u = 0 \longleftrightarrow (AE\ x\ in\ M.\ u\ x = 0)$
⟨*proof*⟩

lemma *AE-iff-nn-integral*:
$\{x{\in}\text{space}\ M.\ P\ x\} \in \text{sets}\ M \Longrightarrow (AE\ x\ in\ M.\ P\ x) \longleftrightarrow \text{integral}^N\ M\ (\text{indicator}\ \{x.\ \neg\ P\ x\}) = 0$
⟨*proof*⟩

lemma *nn-integral-less*:
assumes [*measurable*]: $f \in$ *borel-measurable M* $g \in$ *borel-measurable M*
assumes *f*: $(\int^+ x.\ f\ x\ \partial M) \neq \infty$
assumes *ord*: $AE\ x\ in\ M.\ f\ x \le g\ x$ $\neg\ (AE\ x\ in\ M.\ g\ x \le f\ x)$
shows $(\int^+ x.\ f\ x\ \partial M) < (\int^+ x.\ g\ x\ \partial M)$
⟨*proof*⟩

lemma *nn-integral-subalgebra*:
assumes *f*: $f \in$ *borel-measurable N*
and *N*: $\text{sets}\ N \subseteq \text{sets}\ M$ $\text{space}\ N = \text{space}\ M$ $\bigwedge A.\ A \in \text{sets}\ N \Longrightarrow \text{emeasure}\ N\ A = \text{emeasure}\ M\ A$
shows $\text{integral}^N\ N\ f = \text{integral}^N\ M\ f$
⟨*proof*⟩

lemma *nn-integral-nat-function*:
fixes $f :: 'a \Rightarrow nat$
assumes $f \in measurable\ M\ (count\text{-}space\ UNIV)$
shows $(\int^+ x.\ of\text{-}nat\ (f\ x)\ \partial M) = (\sum t.\ emeasure\ M\ \{x \in space\ M.\ t < f\ x\})$
⟨*proof*⟩

lemma *nn-integral-lfp*:
assumes $sets[simp]$: $\bigwedge s.\ sets\ (M\ s) = sets\ N$
assumes f: $sup\text{-}continuous\ f$
assumes g: $sup\text{-}continuous\ g$
assumes $meas$: $\bigwedge F.\ F \in borel\text{-}measurable\ N \Longrightarrow f\ F \in borel\text{-}measurable\ N$
assumes $step$: $\bigwedge F\ s.\ F \in borel\text{-}measurable\ N \Longrightarrow integral^N\ (M\ s)\ (f\ F) = g\ (\lambda s.\ integral^N\ (M\ s)\ F)\ s$
shows $(\int^+ \omega.\ lfp\ f\ \omega\ \partial M\ s) = lfp\ g\ s$
⟨*proof*⟩

lemma *nn-integral-gfp*:
assumes $sets[simp]$: $\bigwedge s.\ sets\ (M\ s) = sets\ N$
assumes f: $inf\text{-}continuous\ f$ **and** g: $inf\text{-}continuous\ g$
assumes $meas$: $\bigwedge F.\ F \in borel\text{-}measurable\ N \Longrightarrow f\ F \in borel\text{-}measurable\ N$
assumes $bound$: $\bigwedge F\ s.\ F \in borel\text{-}measurable\ N \Longrightarrow (\int^+ x.\ f\ F\ x\ \partial M\ s) < \infty$
assumes $non\text{-}zero$: $\bigwedge s.\ emeasure\ (M\ s)\ (space\ (M\ s)) \neq 0$
assumes $step$: $\bigwedge F\ s.\ F \in borel\text{-}measurable\ N \Longrightarrow integral^N\ (M\ s)\ (f\ F) = g\ (\lambda s.\ integral^N\ (M\ s)\ F)\ s$
shows $(\int^+ \omega.\ gfp\ f\ \omega\ \partial M\ s) = gfp\ g\ s$
⟨*proof*⟩

5.4 Integral under concrete measures

lemma *nn-integral-empty*:
assumes $space\ M = \{\}$
shows $nn\text{-}integral\ M\ f = 0$
⟨*proof*⟩

5.4.1 Distributions

lemma *nn-integral-distr*:
assumes T: $T \in measurable\ M\ M'$ **and** f: $f \in borel\text{-}measurable\ (distr\ M\ M'\ T)$
shows $integral^N\ (distr\ M\ M'\ T)\ f = (\int^+ x.\ f\ (T\ x)\ \partial M)$
⟨*proof*⟩

5.4.2 Counting space

lemma *simple-function-count-space*[*simp*]:
$simple\text{-}function\ (count\text{-}space\ A)\ f \longleftrightarrow finite\ (f\ `\ A)$
⟨*proof*⟩

lemma *nn-integral-count-space*:
assumes A: $finite\ \{a \in A.\ 0 < f\ a\}$

shows *integral*^{N} (*count-space A*) *f* = ($\sum$ *a*|*a*∈*A* ∧ *0* < *f a*. *f a*)
⟨*proof*⟩

lemma *nn-integral-count-space-finite*:
finite A ⟹ ($\int$ $^+$*x*. *f x* ∂*count-space A*) = ($\sum$ *a*∈*A*. *f a*)
⟨*proof*⟩

lemma *nn-integral-count-space'*:
assumes *finite A* ⋀*x*. *x* ∈ *B* ⟹ *x* ∉ *A* ⟹ *f x* = *0 A* ⊆ *B*
shows ($\int$ $^+$*x*. *f x* ∂*count-space B*) = ($\sum$ *x*∈*A*. *f x*)
⟨*proof*⟩

lemma *nn-integral-bij-count-space*:
assumes *g*: *bij-betw g A B*
shows ($\int$ $^+$*x*. *f* (*g x*) ∂*count-space A*) = ($\int$ $^+$*x*. *f x* ∂*count-space B*)
⟨*proof*⟩

lemma *nn-integral-indicator-finite*:
fixes *f* :: *'a* ⇒ *ennreal*
assumes *f*: *finite A* **and** [*measurable*]: ⋀*a*. *a* ∈ *A* ⟹ {*a*} ∈ *sets M*
shows ($\int$ $^+$*x*. *f x* ∗ *indicator A x* ∂*M*) = ($\sum$ *x*∈*A*. *f x* ∗ *emeasure M* {*x*})
⟨*proof*⟩

lemma *nn-integral-count-space-nat*:
fixes *f* :: *nat* ⇒ *ennreal*
shows ($\int$ $^+$*i*. *f i* ∂*count-space UNIV*) = ($\sum$ *i*. *f i*)
⟨*proof*⟩

lemma *nn-integral-enat-function*:
assumes *f*: *f* ∈ *measurable M* (*count-space UNIV*)
shows ($\int$ $^+$ *x*. *ennreal-of-enat* (*f x*) ∂*M*) = ($\sum$ *t*. *emeasure M* {*x* ∈ *space M*. *t* < *f x*})
⟨*proof*⟩

lemma *nn-integral-count-space-nn-integral*:
fixes *f* :: *'i* ⇒ *'a* ⇒ *ennreal*
assumes *countable I* **and** [*measurable*]: ⋀*i*. *i* ∈ *I* ⟹ *f i* ∈ *borel-measurable M*
shows ($\int$ $^+$*x*. $\int$ $^+$*i*. *f i x* ∂*count-space I* ∂*M*) = ($\int$ $^+$*i*. $\int$ $^+$*x*. *f i x* ∂*M* ∂*count-space I*)
⟨*proof*⟩

lemma *emeasure-UN-countable*:
assumes *sets*[*measurable*]: ⋀*i*. *i* ∈ *I* ⟹ *X i* ∈ *sets M* **and** *I*[*simp*]: *countable I*
assumes *disj*: *disjoint-family-on X I*
shows *emeasure M* (*UNION I X*) = ($\int$ $^+$*i*. *emeasure M* (*X i*) ∂*count-space I*)
⟨*proof*⟩

lemma *emeasure-countable-singleton*:
assumes *sets*: ⋀*x*. *x* ∈ *X* ⟹ {*x*} ∈ *sets M* **and** *X*: *countable X*

shows *emeasure M X* = ($\int^+ x$. *emeasure M* {*x*} ∂*count-space X*)
⟨*proof*⟩

lemma *measure-eqI-countable*:
assumes [*simp*]: *sets M* = *Pow A sets N* = *Pow A* **and** *A*: *countable A*
assumes *eq*: ⋀*a*. *a* ∈ *A* ⟹ *emeasure M* {*a*} = *emeasure N* {*a*}
shows *M* = *N*
⟨*proof*⟩

lemma *measure-eqI-countable-AE*:
assumes [*simp*]: *sets M* = *UNIV sets N* = *UNIV*
assumes *ae*: *AE x in M*. *x* ∈ Ω *AE x in N*. *x* ∈ Ω **and** [*simp*]: *countable* Ω
assumes *eq*: ⋀*x*. *x* ∈ Ω ⟹ *emeasure M* {*x*} = *emeasure N* {*x*}
shows *M* = *N*
⟨*proof*⟩

lemma *nn-integral-monotone-convergence-SUP-nat*:
fixes *f* :: *'a* ⇒ *nat* ⇒ *ennreal*
assumes *chain*: *Complete-Partial-Order.chain op* ≤ (*f* ‘ *Y*)
and *nonempty*: *Y* ≠ {}
shows ($\int^+ x$. (*SUP i*:*Y*. *f i x*) ∂*count-space UNIV*) = (*SUP i*:*Y*. ($\int^+ x$. *f i x* ∂*count-space UNIV*))
(**is** *?lhs* = *?rhs* **is** *integral*^{N} *?M* - = -)
⟨*proof*⟩

lemma *power-series-tendsto-at-left*:
assumes *nonneg*: ⋀*i*. *0* ≤ *f i* **and** *summable*: ⋀*z*. *0* ≤ *z* ⟹ *z* < *1* ⟹ *summable* (λ*n*. *f n* ∗ *z*ˆ*n*)
shows ((λ*z*. *ennreal* (∑*n*. *f n* ∗ *z*ˆ*n*)) ⟶ (∑*n*. *ennreal* (*f n*))) (*at-left* (*1*::*real*))
⟨*proof*⟩

5.4.3 Measures with Restricted Space

lemma *simple-function-restrict-space-ennreal*:
fixes *f* :: *'a* ⇒ *ennreal*
assumes Ω ∩ *space M* ∈ *sets M*
shows *simple-function* (*restrict-space M* Ω) *f* ⟷ *simple-function M* (λ*x*. *f x* ∗ *indicator* Ω *x*)
⟨*proof*⟩

lemma *simple-function-restrict-space*:
fixes *f* :: *'a* ⇒ *'b*::*real-normed-vector*
assumes Ω ∩ *space M* ∈ *sets M*
shows *simple-function* (*restrict-space M* Ω) *f* ⟷ *simple-function M* (λ*x*. *indicator* Ω *x* $*_R$ *f x*)
⟨*proof*⟩

lemma *simple-integral-restrict-space*:

assumes Ω: $\Omega \cap$ *space M* $\in$ *sets M simple-function* (*restrict-space M* Ω) *f*
shows *simple-integral* (*restrict-space M* Ω) *f* = *simple-integral M* ($\lambda x.$ *f x* $*$ *indicator* Ω *x*)
⟨*proof*⟩

lemma *nn-integral-restrict-space*:
assumes Ω[*simp*]: $\Omega \cap$ *space M* $\in$ *sets M*
shows *nn-integral* (*restrict-space M* Ω) *f* = *nn-integral M* ($\lambda x.$ *f x* $*$ *indicator* Ω *x*)
⟨*proof*⟩

lemma *nn-integral-count-space-indicator*:
assumes *NO-MATCH* (*UNIV*::'*a set*) (*X*::'*a set*)
shows ($\int^+ x.$ *f x* ∂*count-space X*) = ($\int^+ x.$ *f x* $*$ *indicator X x* ∂*count-space UNIV*)
⟨*proof*⟩

lemma *nn-integral-count-space-eq*:
($\bigwedge x.$ $x \in A - B \Longrightarrow f\ x = 0$) $\Longrightarrow$ ($\bigwedge x.$ $x \in B - A \Longrightarrow f\ x = 0$) $\Longrightarrow$
($\int^+ x.$ *f x* ∂*count-space A*) = ($\int^+ x.$ *f x* ∂*count-space B*)
⟨*proof*⟩

lemma *nn-integral-ge-point*:
assumes $x \in A$
shows $p\ x \leq \int^+$ *x. p x* ∂*count-space A*
⟨*proof*⟩

5.4.4 Measure spaces with an associated density

definition *density* :: '*a measure* $\Rightarrow$ ('*a* $\Rightarrow$ *ennreal*) $\Rightarrow$ '*a measure* **where**
density M f = *measure-of* (*space M*) (*sets M*) ($\lambda A.$ $\int^+$ *x. f x* $*$ *indicator A x* ∂M)

lemma
shows *sets-density*[*simp*, *measurable-cong*]: *sets* (*density M f*) = *sets M*
and *space-density*[*simp*]: *space* (*density M f*) = *space M*
⟨*proof*⟩

lemma *space-density-imp*[*measurable-dest*]:
$\bigwedge$*x M f. x* $\in$ *space* (*density M f*) $\Longrightarrow$ *x* $\in$ *space M* ⟨*proof*⟩

lemma
shows *measurable-density-eq1*[*simp*]: *g* $\in$ *measurable* (*density Mg f*) *Mg'* $\longleftrightarrow$ *g* $\in$ *measurable Mg Mg'*
and *measurable-density-eq2*[*simp*]: *h* $\in$ *measurable Mh* (*density Mh' f*) $\longleftrightarrow$ *h* $\in$ *measurable Mh Mh'*
and *simple-function-density-eq*[*simp*]: *simple-function* (*density Mu f*) *u* $\longleftrightarrow$ *simple-function Mu u*

⟨*proof*⟩

lemma *density-cong*: $f \in$ *borel-measurable M* $\Longrightarrow f' \in$ *borel-measurable M* $\Longrightarrow$
(*AE x in M*. $f\ x = f'\ x$) $\Longrightarrow$ *density M f* $=$ *density M f'*
⟨*proof*⟩

lemma *emeasure-density*:
assumes *f*[*measurable*]: $f \in$ *borel-measurable M* **and** *A*[*measurable*]: $A \in$ *sets M*
shows *emeasure* (*density M f*) $A = (\int^{+} x.\ f\ x * \textit{indicator}\ A\ x\ \partial M)$
(**is** - = *?μ A*)
⟨*proof*⟩

lemma *null-sets-density-iff*:
assumes *f*: $f \in$ *borel-measurable M*
shows $A \in$ *null-sets* (*density M f*) $\longleftrightarrow A \in$ *sets M* $\wedge$ (*AE x in M*. $x \in A \longrightarrow$ $f\ x = 0$)
⟨*proof*⟩

lemma *AE-density*:
assumes *f*: $f \in$ *borel-measurable M*
shows (*AE x in density M f*. *P x*) $\longleftrightarrow$ (*AE x in M*. $0 < f\ x \longrightarrow P\ x$)
⟨*proof*⟩

lemma *nn-integral-density*:
assumes *f*: $f \in$ *borel-measurable M*
assumes *g*: $g \in$ *borel-measurable M*
shows $\textit{integral}^N$ (*density M f*) $g = (\int^{+} x.\ f\ x * g\ x\ \partial M)$
⟨*proof*⟩

lemma *density-distr*:
assumes [*measurable*]: $f \in$ *borel-measurable N X* $\in$ *measurable M N*
shows *density* (*distr M N X*) f = *distr* (*density M* ($\lambda x.\ f\ (X\ x)$)) *N X*
⟨*proof*⟩

lemma *emeasure-restricted*:
assumes *S*: $S \in$ *sets M* **and** *X*: $X \in$ *sets M*
shows *emeasure* (*density M* (*indicator S*)) X = *emeasure M* ($S \cap X$)
⟨*proof*⟩

lemma *measure-restricted*:
$S \in$ *sets M* $\Longrightarrow X \in$ *sets M* $\Longrightarrow$ *measure* (*density M* (*indicator S*)) X = *measure M* ($S \cap X$)
⟨*proof*⟩

lemma (**in** *finite-measure*) *finite-measure-restricted*:
$S \in$ *sets M* $\Longrightarrow$ *finite-measure* (*density M* (*indicator S*))
⟨*proof*⟩

lemma *emeasure-density-const*:

A ∈ *sets M* ⟹ *emeasure* (*density M* (λ-. *c*)) *A* = *c* ∗ *emeasure M A*
⟨*proof*⟩

lemma *measure-density-const*:
A ∈ *sets M* ⟹ *c* ≠ ∞ ⟹ *measure* (*density M* (λ-. *c*)) *A* = *enn2real c* ∗ *measure M A*
⟨*proof*⟩

lemma *density-density-eq*:
f ∈ *borel-measurable M* ⟹ *g* ∈ *borel-measurable M* ⟹
density (*density M f*) *g* = *density M* (λ*x*. *f x* ∗ *g x*)
⟨*proof*⟩

lemma *distr-density-distr*:
assumes *T*: *T* ∈ *measurable M M'* **and** *T'*: *T'* ∈ *measurable M' M*
and *inv*: ∀ *x*∈*space M*. *T'* (*T x*) = *x*
assumes *f*: *f* ∈ *borel-measurable M'*
shows *distr* (*density* (*distr M M' T*) *f*) *M T'* = *density M* (*f* ∘ *T*) (**is** *?R* = *?L*)
⟨*proof*⟩

lemma *density-density-divide*:
fixes *f g* :: *'a* ⇒ *real*
assumes *f*: *f* ∈ *borel-measurable M AE x in M*. *0* ≤ *f x*
assumes *g*: *g* ∈ *borel-measurable M AE x in M*. *0* ≤ *g x*
assumes *ac*: *AE x in M*. *f x* = *0* ⟶ *g x* = *0*
shows *density* (*density M f*) (λ*x*. *g x* / *f x*) = *density M g*
⟨*proof*⟩

lemma *density-1*: *density M* (λ-. *1*) = *M*
⟨*proof*⟩

lemma *emeasure-density-add*:
assumes *X*: *X* ∈ *sets M*
assumes *Mf*[*measurable*]: *f* ∈ *borel-measurable M*
assumes *Mg*[*measurable*]: *g* ∈ *borel-measurable M*
shows *emeasure* (*density M f*) *X* + *emeasure* (*density M g*) *X* =
emeasure (*density M* (λ*x*. *f x* + *g x*)) *X*
⟨*proof*⟩

5.4.5 Point measure

definition *point-measure* :: *'a set* ⇒ (*'a* ⇒ *ennreal*) ⇒ *'a measure* **where**
point-measure A f = *density* (*count-space A*) *f*

lemma
shows *space-point-measure*: *space* (*point-measure A f*) = *A*
and *sets-point-measure*: *sets* (*point-measure A f*) = *Pow A*
⟨*proof*⟩

lemma *sets-point-measure-count-space*[*measurable-cong*]: *sets* (*point-measure A f*) = *sets* (*count-space A*)
⟨*proof*⟩

lemma *measurable-point-measure-eq1*[*simp*]:
g ∈ *measurable* (*point-measure A f*) *M* ⟷ *g* ∈ *A* → *space M*
⟨*proof*⟩

lemma *measurable-point-measure-eq2-finite*[*simp*]:
finite A ⟹
g ∈ *measurable M* (*point-measure A f*) ⟷
(*g* ∈ *space M* → *A* ∧ (∀ *a*∈*A*. *g* −‘ {*a*} ∩ *space M* ∈ *sets M*))
⟨*proof*⟩

lemma *simple-function-point-measure*[*simp*]:
simple-function (*point-measure A f*) *g* ⟷ *finite* (*g* ‘ *A*)
⟨*proof*⟩

lemma *emeasure-point-measure*:
assumes *A*: *finite* {*a*∈*X*. *0* < *f a*} *X* ⊆ *A*
shows *emeasure* (*point-measure A f*) *X* = ($\sum$ *a*|*a*∈*X* ∧ *0* < *f a*. *f a*)
⟨*proof*⟩

lemma *emeasure-point-measure-finite*:
finite A ⟹ *X* ⊆ *A* ⟹ *emeasure* (*point-measure A f*) *X* = ($\sum$ *a*∈*X*. *f a*)
⟨*proof*⟩

lemma *emeasure-point-measure-finite2*:
X ⊆ *A* ⟹ *finite X* ⟹ *emeasure* (*point-measure A f*) *X* = ($\sum$ *a*∈*X*. *f a*)
⟨*proof*⟩

lemma *null-sets-point-measure-iff*:
X ∈ *null-sets* (*point-measure A f*) ⟷ *X* ⊆ *A* ∧ (∀ *x*∈*X*. *f x* = *0*)
⟨*proof*⟩

lemma *AE-point-measure*:
(*AE x in point-measure A f*. *P x*) ⟷ (∀ *x*∈*A*. *0* < *f x* ⟶ *P x*)
⟨*proof*⟩

lemma *nn-integral-point-measure*:
finite {*a*∈*A*. *0* < *f a* ∧ *0* < *g a*} ⟹
$integral^N$ (*point-measure A f*) *g* = ($\sum$ *a*|*a*∈*A* ∧ *0* < *f a* ∧ *0* < *g a*. *f a* ∗ *g a*)
⟨*proof*⟩

lemma *nn-integral-point-measure-finite*:
finite A ⟹ $integral^N$ (*point-measure A f*) *g* = ($\sum$ *a*∈*A*. *f a* ∗ *g a*)
⟨*proof*⟩

5.4.6 Uniform measure

definition *uniform-measure M A = density M (λx. indicator A x / emeasure M A)*

lemma
shows *sets-uniform-measure[simp, measurable-cong]: sets (uniform-measure M A) = sets M*
and *space-uniform-measure[simp]: space (uniform-measure M A) = space M*
⟨*proof*⟩

lemma *emeasure-uniform-measure[simp]:*
assumes *A: A ∈ sets M* **and** *B: B ∈ sets M*
shows *emeasure (uniform-measure M A) B = emeasure M (A ∩ B) / emeasure M A*
⟨*proof*⟩

lemma *measure-uniform-measure[simp]:*
assumes *A: emeasure M A ≠ 0 emeasure M A ≠ ∞* **and** *B: B ∈ sets M*
shows *measure (uniform-measure M A) B = measure M (A ∩ B) / measure M A*
⟨*proof*⟩

lemma *AE-uniform-measureI:*
A ∈ sets M ⟹ (AE x in M. x ∈ A ⟶ P x) ⟹ (AE x in uniform-measure M A. P x)
⟨*proof*⟩

lemma *emeasure-uniform-measure-1:*
emeasure M S ≠ 0 ⟹ emeasure M S ≠ ∞ ⟹ emeasure (uniform-measure M S) S = 1
⟨*proof*⟩

lemma *nn-integral-uniform-measure:*
assumes *f[measurable]: f ∈ borel-measurable M* **and** *S[measurable]: S ∈ sets M*
shows *(∫⁺x. f x ∂uniform-measure M S) = (∫⁺x. f x ∗ indicator S x ∂M) / emeasure M S*
⟨*proof*⟩

lemma *AE-uniform-measure:*
assumes *emeasure M A ≠ 0 emeasure M A < ∞*
shows *(AE x in uniform-measure M A. P x) ⟷ (AE x in M. x ∈ A ⟶ P x)*
⟨*proof*⟩

5.4.7 Null measure

lemma *null-measure-eq-density: null-measure M = density M (λ-. 0)*
⟨*proof*⟩

lemma *nn-integral-null-measure[simp]: (∫⁺x. f x ∂null-measure M) = 0*

⟨proof⟩

lemma *density-null-measure*[*simp*]: *density* (*null-measure M*) *f* = *null-measure M*
⟨proof⟩

5.4.8 Uniform count measure

definition *uniform-count-measure A* = *point-measure A* (λ*x*. *1* / *card A*)

lemma
shows *space-uniform-count-measure*: *space* (*uniform-count-measure A*) = *A*
and *sets-uniform-count-measure*: *sets* (*uniform-count-measure A*) = *Pow A*
⟨proof⟩

lemma *sets-uniform-count-measure-count-space*[*measurable-cong*]:
sets (*uniform-count-measure A*) = *sets* (*count-space A*)
⟨proof⟩

lemma *emeasure-uniform-count-measure*:
finite A ⟹ *X* ⊆ *A* ⟹ *emeasure* (*uniform-count-measure A*) *X* = *card X* / *card A*
⟨proof⟩

lemma *measure-uniform-count-measure*:
finite A ⟹ *X* ⊆ *A* ⟹ *measure* (*uniform-count-measure A*) *X* = *card X* / *card A*
⟨proof⟩

lemma *space-uniform-count-measure-empty-iff* [*simp*]:
space (*uniform-count-measure X*) = {} ⟷ *X* = {}
⟨proof⟩

lemma *sets-uniform-count-measure-eq-UNIV* [*simp*]:
sets (*uniform-count-measure UNIV*) = *UNIV* ⟷ *True*
UNIV = *sets* (*uniform-count-measure UNIV*) ⟷ *True*
⟨proof⟩

5.4.9 Scaled measure

lemma *nn-integral-scale-measure*:
assumes *f*: *f* ∈ *borel-measurable M*
shows *nn-integral* (*scale-measure r M*) *f* = *r* ∗ *nn-integral M f*
⟨proof⟩

end

6 Binary product measures

theory *Binary-Product-Measure*

imports *Nonnegative-Lebesgue-Integration*
begin

lemma *Pair-vimage-times*[*simp*]: $Pair\ x\ -`\ (A \times B) = (if\ x \in A\ then\ B\ else\ \{\})$
⟨*proof*⟩

lemma *rev-Pair-vimage-times*[*simp*]: $(\lambda x.\ (x,\ y))\ -`\ (A \times B) = (if\ y \in B\ then\ A\ else\ \{\})$
⟨*proof*⟩

6.1 Binary products

definition *pair-measure* (**infixr** $\bigotimes_M$ *80*) **where**
$A \bigotimes_M B = measure\text{-}of\ (space\ A \times space\ B)$
$\{a \times b \mid a\ b.\ a \in sets\ A \wedge b \in sets\ B\}$
$(\lambda X.\ \int^+ x.\ (\int^+ y.\ indicator\ X\ (x,y)\ \partial B)\ \partial A)$

lemma *pair-measure-closed*: $\{a \times b \mid a\ b.\ a \in sets\ A \wedge b \in sets\ B\} \subseteq Pow\ (space\ A \times space\ B)$
⟨*proof*⟩

lemma *space-pair-measure*:
$space\ (A \bigotimes_M B) = space\ A \times space\ B$
⟨*proof*⟩

lemma *SIGMA-Collect-eq*: $(SIGMA\ x{:}space\ M.\ \{y{\in}space\ N.\ P\ x\ y\}) = \{x{\in}space\ (M \bigotimes_M N).\ P\ (fst\ x)\ (snd\ x)\}$
⟨*proof*⟩

lemma *sets-pair-measure*:
$sets\ (A \bigotimes_M B) = sigma\text{-}sets\ (space\ A \times space\ B)\ \{a \times b \mid a\ b.\ a \in sets\ A \wedge b \in sets\ B\}$
⟨*proof*⟩

lemma *sets-pair-in-sets*:
assumes N: $space\ A \times space\ B = space\ N$
assumes $\bigwedge a\ b.\ a \in sets\ A \Longrightarrow b \in sets\ B \Longrightarrow a \times b \in sets\ N$
shows $sets\ (A \bigotimes_M B) \subseteq sets\ N$
⟨*proof*⟩

lemma *sets-pair-measure-cong*[*measurable-cong*, *cong*]:
$sets\ M1 = sets\ M1' \Longrightarrow sets\ M2 = sets\ M2' \Longrightarrow sets\ (M1 \bigotimes_M M2) = sets\ (M1' \bigotimes_M M2')$
⟨*proof*⟩

lemma *pair-measureI*[*intro*, *simp*, *measurable*]:
$x \in sets\ A \Longrightarrow y \in sets\ B \Longrightarrow x \times y \in sets\ (A \bigotimes_M B)$
⟨*proof*⟩

lemma *sets-Pair*: $\{x\} \in$ *sets M1* $\Longrightarrow \{y\} \in$ *sets M2* $\Longrightarrow \{(x, y)\} \in$ *sets* (*M1* $\bigotimes_M$ *M2*)
⟨*proof*⟩

lemma *measurable-pair-measureI*:
assumes *1*: $f \in$ *space M* $\rightarrow$ *space M1* $\times$ *space M2*
assumes *2*: $\bigwedge A\ B.\ A \in$ *sets M1* $\Longrightarrow B \in$ *sets M2* $\Longrightarrow f$ −' $(A \times B) \cap$ *space M* $\in$ *sets M*
shows $f \in$ *measurable M* (*M1* $\bigotimes_M$ *M2*)
⟨*proof*⟩

lemma *measurable-split-replace*[*measurable* (*raw*)]:
($\lambda x.$ *f x* (*fst* (*g x*)) (*snd* (*g x*))) $\in$ *measurable M N* $\Longrightarrow$ ($\lambda x.$ *case-prod* (*f x*) (*g x*)) $\in$ *measurable M N*
⟨*proof*⟩

lemma *measurable-Pair*[*measurable* (*raw*)]:
assumes *f*: $f \in$ *measurable M M1* **and** *g*: $g \in$ *measurable M M2*
shows ($\lambda x.$ (*f x*, *g x*)) $\in$ *measurable M* (*M1* $\bigotimes_M$ *M2*)
⟨*proof*⟩

lemma *measurable-fst*[*intro*!, *simp*, *measurable*]: *fst* $\in$ *measurable* (*M1* $\bigotimes_M$ *M2*) *M1*
⟨*proof*⟩

lemma *measurable-snd*[*intro*!, *simp*, *measurable*]: *snd* $\in$ *measurable* (*M1* $\bigotimes_M$ *M2*) *M2*
⟨*proof*⟩

lemma *measurable-Pair-compose-split*[*measurable-dest*]:
assumes *f*: *case-prod f* $\in$ *measurable* (*M1* $\bigotimes_M$ *M2*) *N*
assumes *g*: $g \in$ *measurable M M1* **and** *h*: $h \in$ *measurable M M2*
shows ($\lambda x.$ *f* (*g x*) (*h x*)) $\in$ *measurable M N*
⟨*proof*⟩

lemma *measurable-Pair1-compose*[*measurable-dest*]:
assumes *f*: ($\lambda x.$ (*f x*, *g x*)) $\in$ *measurable M* (*M1* $\bigotimes_M$ *M2*)
assumes [*measurable*]: $h \in$ *measurable N M*
shows ($\lambda x.$ *f* (*h x*)) $\in$ *measurable N M1*
⟨*proof*⟩

lemma *measurable-Pair2-compose*[*measurable-dest*]:
assumes *f*: ($\lambda x.$ (*f x*, *g x*)) $\in$ *measurable M* (*M1* $\bigotimes_M$ *M2*)
assumes [*measurable*]: $h \in$ *measurable N M*
shows ($\lambda x.$ *g* (*h x*)) $\in$ *measurable N M2*
⟨*proof*⟩

lemma *measurable-pair*:
assumes (*fst* $\circ$ *f*) $\in$ *measurable M M1* (*snd* $\circ$ *f*) $\in$ *measurable M M2*

shows $f \in$ *measurable* M $(M1 \bigotimes_M M2)$
⟨*proof*⟩

lemma
assumes f[*measurable*]: $f \in$ *measurable* M $(N \bigotimes_M P)$
shows *measurable-fst'*: $(\lambda x.\ fst\ (f\ x)) \in$ *measurable* M N
and *measurable-snd'*: $(\lambda x.\ snd\ (f\ x)) \in$ *measurable* M P
⟨*proof*⟩

lemma
assumes f[*measurable*]: $f \in$ *measurable* M N
shows *measurable-fst''*: $(\lambda x.\ f\ (fst\ x)) \in$ *measurable* $(M \bigotimes_M P)$ N
and *measurable-snd''*: $(\lambda x.\ f\ (snd\ x)) \in$ *measurable* $(P \bigotimes_M M)$ N
⟨*proof*⟩

lemma *sets-pair-eq-sets-fst-snd*:
sets $(A \bigotimes_M B)$ = *sets* (*Sup-sigma* {*vimage-algebra* (*space* A × *space* B) *fst* A, *vimage-algebra* (*space* A × *space* B) *snd* B})
(**is** *?P* = *sets* (*Sup-sigma* {*?fst*, *?snd*}))
⟨*proof*⟩

lemma *measurable-pair-iff*:
$f \in$ *measurable* M $(M1 \bigotimes_M M2) \longleftrightarrow (fst \circ f) \in$ *measurable* M $M1 \wedge (snd \circ f) \in$ *measurable* M $M2$
⟨*proof*⟩

lemma *measurable-split-conv*:
$(\lambda(x,\ y).\ f\ x\ y) \in$ *measurable* A $B \longleftrightarrow (\lambda x.\ f\ (fst\ x)\ (snd\ x)) \in$ *measurable* A B
⟨*proof*⟩

lemma *measurable-pair-swap'*: $(\lambda(x,y).\ (y,\ x)) \in$ *measurable* $(M1 \bigotimes_M M2)$ $(M2 \bigotimes_M M1)$
⟨*proof*⟩

lemma *measurable-pair-swap*:
assumes f: $f \in$ *measurable* $(M1 \bigotimes_M M2)$ M **shows** $(\lambda(x,y).\ f\ (y,\ x)) \in$ *measurable* $(M2 \bigotimes_M M1)$ M
⟨*proof*⟩

lemma *measurable-pair-swap-iff*:
$f \in$ *measurable* $(M2 \bigotimes_M M1)$ $M \longleftrightarrow (\lambda(x,y).\ f\ (y,x)) \in$ *measurable* $(M1 \bigotimes_M M2)$ M
⟨*proof*⟩

lemma *measurable-Pair1'*: $x \in$ *space* $M1 \Longrightarrow$ *Pair* $x \in$ *measurable* $M2$ $(M1 \bigotimes_M M2)$
⟨*proof*⟩

lemma *sets-Pair1*[*measurable* (*raw*)]:

assumes *A*: $A \in$ *sets* (*M1* $\bigotimes_M$ *M2*) **shows** *Pair x* $-$‘ $A \in$ *sets M2*
⟨*proof*⟩

lemma *measurable-Pair2'*: $y \in$ *space M2* $\Longrightarrow$ ($\lambda x.$ (x, y)) $\in$ *measurable M1* (*M1* $\bigotimes_M$ *M2*)
⟨*proof*⟩

lemma *sets-Pair2*: **assumes** *A*: $A \in$ *sets* (*M1* $\bigotimes_M$ *M2*) **shows** ($\lambda x.$ (x, y)) $-$‘ $A \in$ *sets M1*
⟨*proof*⟩

lemma *measurable-Pair2*:
assumes *f*: $f \in$ *measurable* (*M1* $\bigotimes_M$ *M2*) *M* **and** *x*: $x \in$ *space M1*
shows ($\lambda y.$ f (x, y)) $\in$ *measurable M2 M*
⟨*proof*⟩

lemma *measurable-Pair1*:
assumes *f*: $f \in$ *measurable* (*M1* $\bigotimes_M$ *M2*) *M* **and** *y*: $y \in$ *space M2*
shows ($\lambda x.$ f (x, y)) $\in$ *measurable M1 M*
⟨*proof*⟩

lemma *Int-stable-pair-measure-generator*: *Int-stable* $\{a \times b \mid a\ b.\ a \in$ *sets* $A \wedge b \in$ *sets* $B\}$
⟨*proof*⟩

lemma (**in** *finite-measure*) *finite-measure-cut-measurable*:
assumes [*measurable*]: $Q \in$ *sets* (*N* $\bigotimes_M$ *M*)
shows ($\lambda x.$ *emeasure M* (*Pair x* $-$‘ Q)) $\in$ *borel-measurable N*
(**is** *?s* $Q \in$ -)
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *measurable-emeasure-Pair*:
assumes *Q*: $Q \in$ *sets* (*N* $\bigotimes_M$ *M*) **shows** ($\lambda x.$ *emeasure M* (*Pair x* $-$‘ Q)) $\in$ *borel-measurable N* (**is** *?s* $Q \in$ -)
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *measurable-emeasure*[*measurable* (*raw*)]:
assumes *space*: $\bigwedge x.\ x \in$ *space N* $\Longrightarrow A\ x \subseteq$ *space M*
assumes *A*: $\{x \in$ *space* (*N* $\bigotimes_M$ *M*). *snd* $x \in A$ (*fst* x)$\} \in$ *sets* (*N* $\bigotimes_M$ *M*)
shows ($\lambda x.$ *emeasure M* ($A\ x$)) $\in$ *borel-measurable N*
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *emeasure-pair-measure*:
assumes $X \in$ *sets* (*N* $\bigotimes_M$ *M*)
shows *emeasure* (*N* $\bigotimes_M$ *M*) X = ($\int^+ x.\ \int^+ y.$ *indicator* X (x, y) $\partial M\ \partial N$) (**is** - = *?μ* X)
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *emeasure-pair-measure-alt*:

assumes X: $X \in sets\ (N \bigotimes_M M)$
shows $emeasure\ (N \bigotimes_M M)\ X = (\int^+ x.\ emeasure\ M\ (Pair\ x -` X)\ \partial N)$
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *emeasure-pair-measure-Times*:
assumes A: $A \in sets\ N$ **and** B: $B \in sets\ M$
shows $emeasure\ (N \bigotimes_M M)\ (A \times B) = emeasure\ N\ A * emeasure\ M\ B$
⟨*proof*⟩

6.2 Binary products of σ-finite emeasure spaces

locale *pair-sigma-finite* = *M1?*: *sigma-finite-measure M1* + *M2?*: *sigma-finite-measure M2*
for *M1* :: *'a measure* **and** *M2* :: *'b measure*

lemma (**in** *pair-sigma-finite*) *measurable-emeasure-Pair1*:
$Q \in sets\ (M1 \bigotimes_M M2) \Longrightarrow (\lambda x.\ emeasure\ M2\ (Pair\ x -` Q)) \in borel\text{-}measurable\ M1$
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *measurable-emeasure-Pair2*:
assumes Q: $Q \in sets\ (M1 \bigotimes_M M2)$ **shows** $(\lambda y.\ emeasure\ M1\ ((\lambda x.\ (x, y)) -` Q)) \in borel\text{-}measurable\ M2$
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *sigma-finite-up-in-pair-measure-generator*:
defines $E \equiv \{A \times B \mid A\ B.\ A \in sets\ M1 \wedge B \in sets\ M2\}$
shows $\exists F{::}nat \Rightarrow ('a \times 'b)\ set.\ range\ F \subseteq E \wedge incseq\ F \wedge (\bigcup i.\ F\ i) = space\ M1 \times space\ M2 \wedge$
$(\forall i.\ emeasure\ (M1 \bigotimes_M M2)\ (F\ i) \neq \infty)$
⟨*proof*⟩

sublocale *pair-sigma-finite* ⊆ *P?*: *sigma-finite-measure* $M1 \bigotimes_M M2$
⟨*proof*⟩

lemma *sigma-finite-pair-measure*:
assumes A: *sigma-finite-measure A* **and** B: *sigma-finite-measure B*
shows *sigma-finite-measure* $(A \bigotimes_M B)$
⟨*proof*⟩

lemma *sets-pair-swap*:
assumes $A \in sets\ (M1 \bigotimes_M M2)$
shows $(\lambda(x, y).\ (y, x)) -` A \cap space\ (M2 \bigotimes_M M1) \in sets\ (M2 \bigotimes_M M1)$
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *distr-pair-swap*:
$M1 \bigotimes_M M2 = distr\ (M2 \bigotimes_M M1)\ (M1 \bigotimes_M M2)\ (\lambda(x, y).\ (y, x))$ (**is** *?P* = *?D*)
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *emeasure-pair-measure-alt2*:
assumes *A*: $A \in sets\ (M1 \bigotimes_M M2)$
shows $emeasure\ (M1 \bigotimes_M M2)\ A = (\int^+ y.\ emeasure\ M1\ ((\lambda x.\ (x,\ y)) -` A)\ \partial M2)$
(**is** - = $?\nu\ A$)
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *AE-pair*:
assumes $AE\ x\ in\ (M1 \bigotimes_M M2).\ Q\ x$
shows $AE\ x\ in\ M1.\ (AE\ y\ in\ M2.\ Q\ (x,\ y))$
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *AE-pair-measure*:
assumes $\{x \in space\ (M1 \bigotimes_M M2).\ P\ x\} \in sets\ (M1 \bigotimes_M M2)$
assumes *ae*: $AE\ x\ in\ M1.\ AE\ y\ in\ M2.\ P\ (x,\ y)$
shows $AE\ x\ in\ M1 \bigotimes_M M2.\ P\ x$
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *AE-pair-iff*:
$\{x \in space\ (M1 \bigotimes_M M2).\ P\ (fst\ x)\ (snd\ x)\} \in sets\ (M1 \bigotimes_M M2) \Longrightarrow$
$(AE\ x\ in\ M1.\ AE\ y\ in\ M2.\ P\ x\ y) \longleftrightarrow (AE\ x\ in\ (M1 \bigotimes_M M2).\ P\ (fst\ x)\ (snd\ x))$
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *AE-commute*:
assumes *P*: $\{x \in space\ (M1 \bigotimes_M M2).\ P\ (fst\ x)\ (snd\ x)\} \in sets\ (M1 \bigotimes_M M2)$
shows $(AE\ x\ in\ M1.\ AE\ y\ in\ M2.\ P\ x\ y) \longleftrightarrow (AE\ y\ in\ M2.\ AE\ x\ in\ M1.\ P\ x\ y)$
⟨*proof*⟩

6.3 Fubinis theorem

lemma *measurable-compose-Pair1*:
$x \in space\ M1 \Longrightarrow g \in measurable\ (M1 \bigotimes_M M2)\ L \Longrightarrow (\lambda y.\ g\ (x,\ y)) \in measurable\ M2\ L$
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *borel-measurable-nn-integral-fst*:
assumes *f*: $f \in borel\text{-}measurable\ (M1 \bigotimes_M M)$
shows $(\lambda x.\ \int^+ y.\ f\ (x,\ y)\ \partial M) \in borel\text{-}measurable\ M1$
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *nn-integral-fst*:
assumes *f*: $f \in borel\text{-}measurable\ (M1 \bigotimes_M M)$
shows $(\int^+ x.\ \int^+ y.\ f\ (x,\ y)\ \partial M\ \partial M1) = integral^N\ (M1 \bigotimes_M M)\ f$ (**is** $?I\ f$ = -)
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *borel-measurable-nn-integral*[*measurable* (*raw*)]:
case-prod $f \in$ *borel-measurable* $(N \bigotimes_M M) \Longrightarrow (\lambda x. \int^+ y.\ f\ x\ y\ \partial M) \in$ *borel-measurable* N
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *nn-integral-snd*:
assumes f[*measurable*]: $f \in$ *borel-measurable* $(M1 \bigotimes_M M2)$
shows $(\int^+ y.\ (\int^+ x.\ f\ (x,\ y)\ \partial M1)\ \partial M2) = integral^N\ (M1 \bigotimes_M M2)\ f$
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *Fubini*:
assumes f: $f \in$ *borel-measurable* $(M1 \bigotimes_M M2)$
shows $(\int^+ y.\ (\int^+ x.\ f\ (x,\ y)\ \partial M1)\ \partial M2) = (\int^+ x.\ (\int^+ y.\ f\ (x,\ y)\ \partial M2)\ \partial M1)$
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *Fubini'*:
assumes f: *case-prod* $f \in$ *borel-measurable* $(M1 \bigotimes_M M2)$
shows $(\int^+ y.\ (\int^+ x.\ f\ x\ y\ \partial M1)\ \partial M2) = (\int^+ x.\ (\int^+ y.\ f\ x\ y\ \partial M2)\ \partial M1)$
⟨*proof*⟩

6.4 Products on counting spaces, densities and distributions

lemma *sigma-prod*:
assumes *X-cover*: $\exists E \subseteq A.$ *countable* $E \wedge X = \bigcup E$ **and** A: $A \subseteq Pow\ X$
assumes *Y-cover*: $\exists E \subseteq B.$ *countable* $E \wedge Y = \bigcup E$ **and** B: $B \subseteq Pow\ Y$
shows *sigma* $X\ A \bigotimes_M$ *sigma* $Y\ B$ = *sigma* $(X \times Y)\ \{a \times b \mid a\ b.\ a \in A \wedge b \in B\}$
(**is** $?P = ?S$)
⟨*proof*⟩

lemma *sigma-sets-pair-measure-generator-finite*:
assumes *finite* A **and** *finite* B
shows *sigma-sets* $(A \times B)\ \{ a \times b \mid a\ b.\ a \subseteq A \wedge b \subseteq B\} = Pow\ (A \times B)$
(**is** *sigma-sets ?prod ?sets* = -)
⟨*proof*⟩

lemma *borel-prod*:
(*borel* $\bigotimes_M$ *borel*) = (*borel* :: (*'a*::*second-countable-topology* × *'b*::*second-countable-topology*) *measure*)
(**is** $?P = ?B$)
⟨*proof*⟩

lemma *pair-measure-count-space*:
assumes A: *finite* A **and** B: *finite* B
shows *count-space* $A \bigotimes_M$ *count-space* B = *count-space* $(A \times B)$ (**is** $?P = ?C$)
⟨*proof*⟩

lemma *emeasure-prod-count-space*:
assumes *A*: $A \in$ *sets* (*count-space UNIV* $\bigotimes_M$ *M*) (**is** $A \in$ *sets* (*?A* $\bigotimes_M$ *?B*))
shows *emeasure* (*?A* $\bigotimes_M$ *?B*) *A* = ($\int^+$ *x*. $\int^+$ *y*. *indicator A* (*x*, *y*) ∂*?B* ∂*?A*)
⟨*proof*⟩

lemma *emeasure-prod-count-space-single*[*simp*]: *emeasure* (*count-space UNIV* $\bigotimes_M$ *count-space UNIV*) {*x*} = *1*
⟨*proof*⟩

lemma *emeasure-count-space-prod-eq*:
fixes *A* :: (*'a* × *'b*) *set*
assumes *A*: $A \in$ *sets* (*count-space UNIV* $\bigotimes_M$ *count-space UNIV*) (**is** $A \in$ *sets* (*?A* $\bigotimes_M$ *?B*))
shows *emeasure* (*?A* $\bigotimes_M$ *?B*) *A* = *emeasure* (*count-space UNIV*) *A*
⟨*proof*⟩

lemma *nn-integral-count-space-prod-eq*:
nn-integral (*count-space UNIV* $\bigotimes_M$ *count-space UNIV*) *f* = *nn-integral* (*count-space UNIV*) *f*
(**is** *nn-integral ?P f* = -)
⟨*proof*⟩

lemma *pair-measure-density*:
assumes *f*: $f \in$ *borel-measurable M1*
assumes *g*: $g \in$ *borel-measurable M2*
assumes *sigma-finite-measure M2 sigma-finite-measure* (*density M2 g*)
shows *density M1 f* $\bigotimes_M$ *density M2 g* = *density* (*M1* $\bigotimes_M$ *M2*) (λ(*x*,*y*). *f x* ∗ *g y*) (**is** *?L* = *?R*)
⟨*proof*⟩

lemma *sigma-finite-measure-distr*:
assumes *sigma-finite-measure* (*distr M N f*) **and** *f*: $f \in$ *measurable M N*
shows *sigma-finite-measure M*
⟨*proof*⟩

lemma *pair-measure-distr*:
assumes *f*: $f \in$ *measurable M S* **and** *g*: $g \in$ *measurable N T*
assumes *sigma-finite-measure* (*distr N T g*)
shows *distr M S f* $\bigotimes_M$ *distr N T g* = *distr* (*M* $\bigotimes_M$ *N*) (*S* $\bigotimes_M$ *T*) (λ(*x*, *y*). (*f x*, *g y*)) (**is** *?P* = *?D*)
⟨*proof*⟩

lemma *pair-measure-eqI*:
assumes *sigma-finite-measure M1 sigma-finite-measure M2*
assumes *sets*: *sets* (*M1* $\bigotimes_M$ *M2*) = *sets M*
assumes *emeasure*: $\bigwedge$*A B*. $A \in$ *sets M1* $\Longrightarrow$ $B \in$ *sets M2* $\Longrightarrow$ *emeasure M1 A* ∗ *emeasure M2 B* = *emeasure M* (*A* × *B*)
shows *M1* $\bigotimes_M$ *M2* = *M*
⟨*proof*⟩

lemma *sets-pair-countable*:
assumes *countable S1 countable S2*
assumes M: *sets* M = *Pow S1* **and** N: *sets* N = *Pow S2*
shows *sets* $(M \bigotimes_M N)$ = *Pow* (*S1* × *S2*)
⟨*proof*⟩

lemma *pair-measure-countable*:
assumes *countable S1 countable S2*
shows *count-space S1* $\bigotimes_M$ *count-space S2* = *count-space* (*S1* × *S2*)
⟨*proof*⟩

lemma *nn-integral-fst-count-space*:
($\int^+ x. \int^+ y.\ f\ (x,\ y)$ ∂*count-space UNIV* ∂*count-space UNIV*) = *integral*^{N} (*count-space UNIV*) *f*
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *nn-integral-snd-count-space*:
($\int^+ y. \int^+ x.\ f\ (x,\ y)$ ∂*count-space UNIV* ∂*count-space UNIV*) = *integral*^{N} (*count-space UNIV*) *f*
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *measurable-pair-measure-countable1*:
assumes *countable A*
and [*measurable*]: $\bigwedge x.\ x \in A \Longrightarrow (\lambda y.\ f\ (x,\ y)) \in$ *measurable N K*
shows $f \in$ *measurable* (*count-space A* $\bigotimes_M$ N) K
⟨*proof*⟩

6.5 Product of Borel spaces

lemma *borel-Times*:
fixes A :: *'a::topological-space set* **and** B :: *'b::topological-space set*
assumes A: $A \in$ *sets borel* **and** B: $B \in$ *sets borel*
shows $A \times B \in$ *sets borel*
⟨*proof*⟩

lemma *finite-measure-pair-measure*:
assumes *finite-measure M finite-measure N*
shows *finite-measure* $(N \bigotimes_M M)$
⟨*proof*⟩

end

7 Finite product measures

theory *Finite-Product-Measure*
imports *Binary-Product-Measure*

begin

lemma *PiE-choice*: $(\exists f \in PiE\ I\ F.\ \forall i \in I.\ P\ i\ (f\ i)) \longleftrightarrow (\forall i \in I.\ \exists x \in F\ i.\ P\ i\ x)$
⟨*proof*⟩

lemma *case-prod-const*: $(\lambda(i, j).\ c) = (\lambda\text{-}.\ c)$
⟨*proof*⟩

7.0.1 More about Function restricted by *extensional*

definition
$merge\ I\ J = (\lambda(x, y)\ i.\ if\ i \in I\ then\ x\ i\ else\ if\ i \in J\ then\ y\ i\ else\ undefined)$

lemma *merge-apply*[*simp*]:
$I \cap J = \{\} \Longrightarrow i \in I \Longrightarrow merge\ I\ J\ (x, y)\ i = x\ i$
$I \cap J = \{\} \Longrightarrow i \in J \Longrightarrow merge\ I\ J\ (x, y)\ i = y\ i$
$J \cap I = \{\} \Longrightarrow i \in I \Longrightarrow merge\ I\ J\ (x, y)\ i = x\ i$
$J \cap I = \{\} \Longrightarrow i \in J \Longrightarrow merge\ I\ J\ (x, y)\ i = y\ i$
$i \notin I \Longrightarrow i \notin J \Longrightarrow merge\ I\ J\ (x, y)\ i = undefined$
⟨*proof*⟩

lemma *merge-commute*:
$I \cap J = \{\} \Longrightarrow merge\ I\ J\ (x, y) = merge\ J\ I\ (y, x)$
⟨*proof*⟩

lemma *Pi-cancel-merge-range*[*simp*]:
$I \cap J = \{\} \Longrightarrow x \in Pi\ I\ (merge\ I\ J\ (A, B)) \longleftrightarrow x \in Pi\ I\ A$
$I \cap J = \{\} \Longrightarrow x \in Pi\ I\ (merge\ J\ I\ (B, A)) \longleftrightarrow x \in Pi\ I\ A$
$J \cap I = \{\} \Longrightarrow x \in Pi\ I\ (merge\ I\ J\ (A, B)) \longleftrightarrow x \in Pi\ I\ A$
$J \cap I = \{\} \Longrightarrow x \in Pi\ I\ (merge\ J\ I\ (B, A)) \longleftrightarrow x \in Pi\ I\ A$
⟨*proof*⟩

lemma *Pi-cancel-merge*[*simp*]:
$I \cap J = \{\} \Longrightarrow merge\ I\ J\ (x, y) \in Pi\ I\ B \longleftrightarrow x \in Pi\ I\ B$
$J \cap I = \{\} \Longrightarrow merge\ I\ J\ (x, y) \in Pi\ I\ B \longleftrightarrow x \in Pi\ I\ B$
$I \cap J = \{\} \Longrightarrow merge\ I\ J\ (x, y) \in Pi\ J\ B \longleftrightarrow y \in Pi\ J\ B$
$J \cap I = \{\} \Longrightarrow merge\ I\ J\ (x, y) \in Pi\ J\ B \longleftrightarrow y \in Pi\ J\ B$
⟨*proof*⟩

lemma *extensional-merge*[*simp*]: $merge\ I\ J\ (x, y) \in extensional\ (I \cup J)$
⟨*proof*⟩

lemma *restrict-merge*[*simp*]:
$I \cap J = \{\} \Longrightarrow restrict\ (merge\ I\ J\ (x, y))\ I = restrict\ x\ I$
$I \cap J = \{\} \Longrightarrow restrict\ (merge\ I\ J\ (x, y))\ J = restrict\ y\ J$
$J \cap I = \{\} \Longrightarrow restrict\ (merge\ I\ J\ (x, y))\ I = restrict\ x\ I$
$J \cap I = \{\} \Longrightarrow restrict\ (merge\ I\ J\ (x, y))\ J = restrict\ y\ J$
⟨*proof*⟩

lemma *split-merge*: P (*merge* I J (x,y) i) $\longleftrightarrow$ ($i \in I \longrightarrow P$ $(x\ i)$) $\wedge$ ($i \in J - I$ $\longrightarrow P$ $(y\ i)$) $\wedge$ ($i \notin I \cup J \longrightarrow P$ *undefined*)
⟨*proof*⟩

lemma *PiE-cancel-merge*[*simp*]:
$I \cap J = \{\} \Longrightarrow$
merge I J $(x,\ y) \in$ *PiE* $(I \cup J)$ $B \longleftrightarrow x \in$ *Pi* I $B \wedge y \in$ *Pi* J B
⟨*proof*⟩

lemma *merge-singleton*[*simp*]: $i \notin I \Longrightarrow$ *merge* I $\{i\}$ (x,y) = *restrict* $(x(i := y\ i))$ (*insert* i I)
⟨*proof*⟩

lemma *extensional-merge-sub*: $I \cup J \subseteq K \Longrightarrow$ *merge* I J $(x,\ y) \in$ *extensional* K
⟨*proof*⟩

lemma *merge-restrict*[*simp*]:
merge I J (*restrict* x I, y) = *merge* I J $(x,\ y)$
merge I J (x, *restrict* y J) = *merge* I J $(x,\ y)$
⟨*proof*⟩

lemma *merge-x-x-eq-restrict*[*simp*]:
merge I J $(x,\ x)$ = *restrict* x $(I \cup J)$
⟨*proof*⟩

lemma *injective-vimage-restrict*:
assumes J: $J \subseteq I$
and *sets*: $A \subseteq (\Pi_E\ i{\in}J.\ S\ i)$ $B \subseteq (\Pi_E\ i{\in}J.\ S\ i)$ **and** *ne*: $(\Pi_E\ i{\in}I.\ S\ i) \neq \{\}$
and *eq*: ($\lambda x.$ *restrict* x J) −' $A \cap (\Pi_E\ i{\in}I.\ S\ i)$ = ($\lambda x.$ *restrict* x J) −' $B \cap (\Pi_E\ i{\in}I.\ S\ i)$
shows $A = B$
⟨*proof*⟩

lemma *restrict-vimage*:
$I \cap J = \{\} \Longrightarrow$
($\lambda x.$ (*restrict* x I, *restrict* x J)) −' (Pi_E I E × Pi_E J F) = *Pi* $(I \cup J)$ (*merge* I J $(E,\ F)$)
⟨*proof*⟩

lemma *merge-vimage*:
$I \cap J = \{\} \Longrightarrow$ *merge* I J −' Pi_E $(I \cup J)$ E = *Pi* I E × *Pi* J E
⟨*proof*⟩

7.1 Finite product spaces

7.1.1 Products

definition *prod-emb* **where**
prod-emb I M K X = ($\lambda x.$ *restrict* x K) −' $X \cap$ (*PIE* $i{:}I.$ *space* $(M\ i)$)

lemma *prod-emb-iff*:
$f \in \textit{prod-emb}\ I\ M\ K\ X \longleftrightarrow f \in \textit{extensional}\ I \land (\textit{restrict}\ f\ K \in X) \land (\forall\, i{\in}I.\ f\ i \in \textit{space}\ (M\ i))$
⟨*proof*⟩

lemma
shows *prod-emb-empty*[*simp*]: $\textit{prod-emb}\ M\ L\ K\ \{\} = \{\}$
and *prod-emb-Un*[*simp*]: $\textit{prod-emb}\ M\ L\ K\ (A \cup B) = \textit{prod-emb}\ M\ L\ K\ A \cup \textit{prod-emb}\ M\ L\ K\ B$
and *prod-emb-Int*: $\textit{prod-emb}\ M\ L\ K\ (A \cap B) = \textit{prod-emb}\ M\ L\ K\ A \cap \textit{prod-emb}\ M\ L\ K\ B$
and *prod-emb-UN*[*simp*]: $\textit{prod-emb}\ M\ L\ K\ (\bigcup i{\in}I.\ F\ i) = (\bigcup i{\in}I.\ \textit{prod-emb}\ M\ L\ K\ (F\ i))$
and *prod-emb-INT*[*simp*]: $I \neq \{\} \Longrightarrow \textit{prod-emb}\ M\ L\ K\ (\bigcap i{\in}I.\ F\ i) = (\bigcap i{\in}I.\ \textit{prod-emb}\ M\ L\ K\ (F\ i))$
and *prod-emb-Diff*[*simp*]: $\textit{prod-emb}\ M\ L\ K\ (A - B) = \textit{prod-emb}\ M\ L\ K\ A - \textit{prod-emb}\ M\ L\ K\ B$
⟨*proof*⟩

lemma *prod-emb-PiE*: $J \subseteq I \Longrightarrow (\bigwedge i.\ i \in J \Longrightarrow E\ i \subseteq \textit{space}\ (M\ i)) \Longrightarrow$
$\textit{prod-emb}\ I\ M\ J\ (\Pi_E\ i{\in}J.\ E\ i) = (\Pi_E\ i{\in}I.\ \textit{if}\ i \in J\ \textit{then}\ E\ i\ \textit{else}\ \textit{space}\ (M\ i))$
⟨*proof*⟩

lemma *prod-emb-PiE-same-index*[*simp*]:
$(\bigwedge i.\ i \in I \Longrightarrow E\ i \subseteq \textit{space}\ (M\ i)) \Longrightarrow \textit{prod-emb}\ I\ M\ I\ (\textit{Pi}_E\ I\ E) = \textit{Pi}_E\ I\ E$
⟨*proof*⟩

lemma *prod-emb-trans*[*simp*]:
$J \subseteq K \Longrightarrow K \subseteq L \Longrightarrow \textit{prod-emb}\ L\ M\ K\ (\textit{prod-emb}\ K\ M\ J\ X) = \textit{prod-emb}\ L\ M\ J\ X$
⟨*proof*⟩

lemma *prod-emb-Pi*:
assumes $X \in (\Pi\ j{\in}J.\ \textit{sets}\ (M\ j))\ J \subseteq K$
shows $\textit{prod-emb}\ K\ M\ J\ (\textit{Pi}_E\ J\ X) = (\Pi_E\ i{\in}K.\ \textit{if}\ i \in J\ \textit{then}\ X\ i\ \textit{else}\ \textit{space}\ (M\ i))$
⟨*proof*⟩

lemma *prod-emb-id*:
$B \subseteq (\Pi_E\ i{\in}L.\ \textit{space}\ (M\ i)) \Longrightarrow \textit{prod-emb}\ L\ M\ L\ B = B$
⟨*proof*⟩

lemma *prod-emb-mono*:
$F \subseteq G \Longrightarrow \textit{prod-emb}\ A\ M\ B\ F \subseteq \textit{prod-emb}\ A\ M\ B\ G$
⟨*proof*⟩

definition *PiM* :: $'i\ \textit{set} \Rightarrow ('i \Rightarrow 'a\ \textit{measure}) \Rightarrow ('i \Rightarrow 'a)\ \textit{measure}$ **where**
$\textit{PiM}\ I\ M = \textit{extend-measure}\ (\Pi_E\ i{\in}I.\ \textit{space}\ (M\ i))$
$\{(J,\ X).\ (J \neq \{\} \lor I = \{\}) \land \textit{finite}\ J \land J \subseteq I \land X \in (\Pi\ j{\in}J.\ \textit{sets}\ (M\ j))\}$

$(\lambda(J, X).\ \textit{prod-emb}\ I\ M\ J\ (\Pi_E\ j{\in}J.\ X\ j))$
$(\lambda(J, X).\ \prod j{\in}J \cup \{i{\in}I.\ \textit{emeasure}\ (M\ i)\ (\textit{space}\ (M\ i)) \neq 1\}.\ \textit{if}\ j \in J\ \textit{then}\ \textit{emeasure}\ (M\ j)\ (X\ j)\ \textit{else}\ \textit{emeasure}\ (M\ j)\ (\textit{space}\ (M\ j)))$

definition *prod-algebra* :: $'i\ set \Rightarrow ('i \Rightarrow 'a\ measure) \Rightarrow ('i \Rightarrow 'a)\ set\ set$ **where**
$\textit{prod-algebra}\ I\ M = (\lambda(J, X).\ \textit{prod-emb}\ I\ M\ J\ (\Pi_E\ j{\in}J.\ X\ j))$ ‘
$\{(J, X).\ (J \neq \{\} \vee I = \{\}) \wedge \textit{finite}\ J \wedge J \subseteq I \wedge X \in (\Pi\ j{\in}J.\ \textit{sets}\ (M\ j))\}$

abbreviation
$Pi_M\ I\ M \equiv PiM\ I\ M$

syntax
-PiM :: $pttrn \Rightarrow 'i\ set \Rightarrow 'a\ measure \Rightarrow ('i => 'a)\ measure$ $((3\Pi_M$ -∈-./ -$)$ $10)$
translations
$\Pi_M\ x{\in}I.\ M == CONST\ PiM\ I\ (\%x.\ M)$

lemma *extend-measure-cong*:
assumes $\Omega = \Omega'\ I = I'\ G = G'\ \bigwedge i.\ i \in I' \Longrightarrow \mu\ i = \mu'\ i$
shows $\textit{extend-measure}\ \Omega\ I\ G\ \mu = \textit{extend-measure}\ \Omega'\ I'\ G'\ \mu'$
⟨*proof*⟩

lemma *Pi-cong-sets*:
$[\![I = J;\ \bigwedge x.\ x \in I \Longrightarrow M\ x = N\ x]\!] \Longrightarrow Pi\ I\ M = Pi\ J\ N$
⟨*proof*⟩

lemma *PiM-cong*:
assumes $I = J\ \bigwedge x.\ x \in I \Longrightarrow M\ x = N\ x$
shows $PiM\ I\ M = PiM\ J\ N$
⟨*proof*⟩

lemma *prod-algebra-sets-into-space*:
$\textit{prod-algebra}\ I\ M \subseteq Pow\ (\Pi_E\ i{\in}I.\ \textit{space}\ (M\ i))$
⟨*proof*⟩

lemma *prod-algebra-eq-finite*:
assumes I: *finite* I
shows $\textit{prod-algebra}\ I\ M = \{(\Pi_E\ i{\in}I.\ X\ i)\ |X.\ X \in (\Pi\ j{\in}I.\ \textit{sets}\ (M\ j))\}$ (**is** *?L* = *?R*)
⟨*proof*⟩

lemma *prod-algebraI*:
$\textit{finite}\ J \Longrightarrow (J \neq \{\} \vee I = \{\}) \Longrightarrow J \subseteq I \Longrightarrow (\bigwedge i.\ i \in J \Longrightarrow E\ i \in \textit{sets}\ (M\ i))$
$\Longrightarrow \textit{prod-emb}\ I\ M\ J\ (PIE\ j{:}J.\ E\ j) \in \textit{prod-algebra}\ I\ M$
⟨*proof*⟩

lemma *prod-algebraI-finite*:
$\textit{finite}\ I \Longrightarrow (\forall\, i{\in}I.\ E\ i \in \textit{sets}\ (M\ i)) \Longrightarrow (Pi_E\ I\ E) \in \textit{prod-algebra}\ I\ M$
⟨*proof*⟩

lemma *Int-stable-PiE*: *Int-stable* $\{Pi_E\ J\ E \mid E.\ \forall\, i{\in}I.\ E\ i \in sets\ (M\ i)\}$
⟨*proof*⟩

lemma *prod-algebraE*:
assumes *A*: $A \in prod\text{-}algebra\ I\ M$
obtains *J E* **where** $A = prod\text{-}emb\ I\ M\ J\ (PIE\ j{:}J.\ E\ j)$
$finite\ J\ J \neq \{\} \vee I = \{\}\ J \subseteq I\ \bigwedge i.\ i \in J \Longrightarrow E\ i \in sets\ (M\ i)$
⟨*proof*⟩

lemma *prod-algebraE-all*:
assumes *A*: $A \in prod\text{-}algebra\ I\ M$
obtains *E* **where** $A = Pi_E\ I\ E\ E \in (\Pi\ i{\in}I.\ sets\ (M\ i))$
⟨*proof*⟩

lemma *Int-stable-prod-algebra*: *Int-stable* (*prod-algebra I M*)
⟨*proof*⟩

lemma *prod-algebra-mono*:
assumes *space*: $\bigwedge i.\ i \in I \Longrightarrow space\ (E\ i) = space\ (F\ i)$
assumes *sets*: $\bigwedge i.\ i \in I \Longrightarrow sets\ (E\ i) \subseteq sets\ (F\ i)$
shows $prod\text{-}algebra\ I\ E \subseteq prod\text{-}algebra\ I\ F$
⟨*proof*⟩

lemma *prod-algebra-cong*:
assumes $I = J$ **and** *sets*: $(\bigwedge i.\ i \in I \Longrightarrow sets\ (M\ i) = sets\ (N\ i))$
shows $prod\text{-}algebra\ I\ M = prod\text{-}algebra\ J\ N$
⟨*proof*⟩

lemma *space-in-prod-algebra*:
$(\Pi_E\ i{\in}I.\ space\ (M\ i)) \in prod\text{-}algebra\ I\ M$
⟨*proof*⟩

lemma *space-PiM*: $space\ (\Pi_M\ i{\in}I.\ M\ i) = (\Pi_E\ i{\in}I.\ space\ (M\ i))$
⟨*proof*⟩

lemma *prod-emb-subset-PiM*[*simp*]: $prod\text{-}emb\ I\ M\ K\ X \subseteq space\ (PiM\ I\ M)$
⟨*proof*⟩

lemma *space-PiM-empty-iff*[*simp*]: $space\ (PiM\ I\ M) = \{\} \longleftrightarrow (\exists\, i{\in}I.\ space\ (M\ i) = \{\})$
⟨*proof*⟩

lemma *undefined-in-PiM-empty*[*simp*]: $(\lambda x.\ undefined) \in space\ (PiM\ \{\}\ M)$
⟨*proof*⟩

lemma *sets-PiM*: $sets\ (\Pi_M\ i{\in}I.\ M\ i) = sigma\text{-}sets\ (\Pi_E\ i{\in}I.\ space\ (M\ i))\ (prod\text{-}algebra\ I\ M)$
⟨*proof*⟩

lemma *sets-PiM-single*: *sets* (*PiM I M*) =
sigma-sets (Π_E *i*∈*I*. *space* (*M i*)) {{*f*∈Π_E *i*∈*I*. *space* (*M i*). *f i* ∈ *A*} | *i A*. *i* ∈ *I* ∧ *A* ∈ *sets* (*M i*)}
(**is** - = *sigma-sets* ?Ω ?*R*)
⟨*proof*⟩

lemma *sets-PiM-eq-proj*:
I ≠ {} ⟹ *sets* (*PiM I M*) = *sets* ($\bigsqcup_\sigma$ *i*∈*I*. *vimage-algebra* (Π_E *i*∈*I*. *space* (*M i*)) (λ*x*. *x i*) (*M i*))
⟨*proof*⟩

lemma
shows *space-PiM-empty*: *space* (Pi_M {} *M*) = {λ*k*. *undefined*}
and *sets-PiM-empty*: *sets* (Pi_M {} *M*) = { {}, {λ*k*. *undefined*} }
⟨*proof*⟩

lemma *sets-PiM-sigma*:
assumes Ω-*cover*: ⋀*i*. *i* ∈ *I* ⟹ ∃*S*⊆*E i*. *countable S* ∧ Ω *i* = ⋃*S*
assumes *E*: ⋀*i*. *i* ∈ *I* ⟹ *E i* ⊆ *Pow* (Ω *i*)
assumes *J*: ⋀*j*. *j* ∈ *J* ⟹ *finite j* ⋃*J* = *I*
defines *P* ≡ {{*f*∈(Π_E *i*∈*I*. Ω *i*). ∀*i*∈*j*. *f i* ∈ *A i*} | *A j*. *j* ∈ *J* ∧ *A* ∈ *Pi j E*}
shows *sets* (Π_M *i*∈*I*. *sigma* (Ω *i*) (*E i*)) = *sets* (*sigma* (Π_E *i*∈*I*. Ω *i*) *P*)
⟨*proof*⟩

lemma *sets-PiM-in-sets*:
assumes *space*: *space N* = (Π_E *i*∈*I*. *space* (*M i*))
assumes *sets*: ⋀*i A*. *i* ∈ *I* ⟹ *A* ∈ *sets* (*M i*) ⟹ {*x*∈*space N*. *x i* ∈ *A*} ∈ *sets N*
shows *sets* (Π_M *i* ∈ *I*. *M i*) ⊆ *sets N*
⟨*proof*⟩

lemma *sets-PiM-cong*[*measurable-cong*]:
assumes *I* = *J* ⋀*i*. *i* ∈ *J* ⟹ *sets* (*M i*) = *sets* (*N i*) **shows** *sets* (*PiM I M*) = *sets* (*PiM J N*)
⟨*proof*⟩

lemma *sets-PiM-I*:
assumes *finite J J* ⊆ *I* ∀*i*∈*J*. *E i* ∈ *sets* (*M i*)
shows *prod-emb I M J* (*PIE j*:*J*. *E j*) ∈ *sets* (Π_M *i*∈*I*. *M i*)
⟨*proof*⟩

lemma *measurable-PiM*:
assumes *space*: *f* ∈ *space N* → (Π_E *i*∈*I*. *space* (*M i*))
assumes *sets*: ⋀*X J*. *J* ≠ {} ∨ *I* = {} ⟹ *finite J* ⟹ *J* ⊆ *I* ⟹ (⋀*i*. *i* ∈ *J* ⟹ *X i* ∈ *sets* (*M i*)) ⟹
f −' *prod-emb I M J* (Pi_E *J X*) ∩ *space N* ∈ *sets N*
shows *f* ∈ *measurable N* (*PiM I M*)
⟨*proof*⟩

lemma *measurable-PiM-Collect*:
assumes *space*: $f \in space\ N \to (\Pi_E\ i{\in}I.\ space\ (M\ i))$
assumes *sets*: $\bigwedge X\ J.\ J \neq \{\} \lor I = \{\} \Longrightarrow finite\ J \Longrightarrow J \subseteq I \Longrightarrow (\bigwedge i.\ i \in J \Longrightarrow X\ i \in sets\ (M\ i)) \Longrightarrow$
$\{\omega{\in}space\ N.\ \forall\, i{\in}J.\ f\ \omega\ i \in X\ i\} \in sets\ N$
shows $f \in measurable\ N\ (PiM\ I\ M)$
⟨*proof*⟩

lemma *measurable-PiM-single*:
assumes *space*: $f \in space\ N \to (\Pi_E\ i{\in}I.\ space\ (M\ i))$
assumes *sets*: $\bigwedge A\ i.\ i \in I \Longrightarrow A \in sets\ (M\ i) \Longrightarrow \{\omega \in space\ N.\ f\ \omega\ i \in A\} \in sets\ N$
shows $f \in measurable\ N\ (PiM\ I\ M)$
⟨*proof*⟩

lemma *measurable-PiM-single'*:
assumes *f*: $\bigwedge i.\ i \in I \Longrightarrow f\ i \in measurable\ N\ (M\ i)$
and $(\lambda\omega\ i.\ f\ i\ \omega) \in space\ N \to (\Pi_E\ i{\in}I.\ space\ (M\ i))$
shows $(\lambda\omega\ i.\ f\ i\ \omega) \in measurable\ N\ (Pi_M\ I\ M)$
⟨*proof*⟩

lemma *sets-PiM-I-finite*[*measurable*]:
assumes *finite I* **and** *sets*: $(\bigwedge i.\ i \in I \Longrightarrow E\ i \in sets\ (M\ i))$
shows $(PIE\ j{:}I.\ E\ j) \in sets\ (\Pi_M\ i{\in}I.\ M\ i)$
⟨*proof*⟩

lemma *measurable-component-singleton*[*measurable* (*raw*)]:
assumes $i \in I$ **shows** $(\lambda x.\ x\ i) \in measurable\ (Pi_M\ I\ M)\ (M\ i)$
⟨*proof*⟩

lemma *measurable-component-singleton'*[*measurable-dest*]:
assumes *f*: $f \in measurable\ N\ (Pi_M\ I\ M)$
assumes *g*: $g \in measurable\ L\ N$
assumes *i*: $i \in I$
shows $(\lambda x.\ (f\ (g\ x))\ i) \in measurable\ L\ (M\ i)$
⟨*proof*⟩

lemma *measurable-PiM-component-rev*:
$i \in I \Longrightarrow f \in measurable\ (M\ i)\ N \Longrightarrow (\lambda x.\ f\ (x\ i)) \in measurable\ (PiM\ I\ M)\ N$
⟨*proof*⟩

lemma *measurable-case-nat*[*measurable* (*raw*)]:
assumes [*measurable* (*raw*)]: $i = 0 \Longrightarrow f \in measurable\ M\ N$
$\bigwedge j.\ i = Suc\ j \Longrightarrow (\lambda x.\ g\ x\ j) \in measurable\ M\ N$
shows $(\lambda x.\ case\text{-}nat\ (f\ x)\ (g\ x)\ i) \in measurable\ M\ N$
⟨*proof*⟩

lemma *measurable-case-nat'*[*measurable* (*raw*)]:

assumes *fg*[*measurable*]: $f \in$ *measurable* N M $g \in$ *measurable* N $(\Pi_M\ i{\in}UNIV.\ M)$
shows $(\lambda x.$ *case-nat* $(f\ x)\ (g\ x)) \in$ *measurable* N $(\Pi_M\ i{\in}UNIV.\ M)$
⟨*proof*⟩

lemma *measurable-add-dim*[*measurable*]:
$(\lambda(f,\ y).\ f(i := y)) \in$ *measurable* $(Pi_M\ I\ M \bigotimes_M M\ i)\ (Pi_M\ (insert\ i\ I)\ M)$
(**is** *?f* $\in$ *measurable ?P ?I*)
⟨*proof*⟩

lemma *measurable-fun-upd*:
assumes I: $I = J \cup \{i\}$
assumes f[*measurable*]: $f \in$ *measurable* N $(PiM\ J\ M)$
assumes h[*measurable*]: $h \in$ *measurable* N $(M\ i)$
shows $(\lambda x.\ (f\ x)\ (i := h\ x)) \in$ *measurable* N $(PiM\ I\ M)$
⟨*proof*⟩

lemma *measurable-component-update*:
$x \in$ *space* $(Pi_M\ I\ M) \Longrightarrow i \notin I \Longrightarrow (\lambda v.\ x(i := v)) \in$ *measurable* $(M\ i)\ (Pi_M\ (insert\ i\ I)\ M)$
⟨*proof*⟩

lemma *measurable-merge*[*measurable*]:
merge I $J \in$ *measurable* $(Pi_M\ I\ M \bigotimes_M Pi_M\ J\ M)\ (Pi_M\ (I \cup J)\ M)$
(**is** *?f* $\in$ *measurable ?P ?U*)
⟨*proof*⟩

lemma *measurable-restrict*[*measurable* (*raw*)]:
assumes X: $\bigwedge i.\ i \in I \Longrightarrow X\ i \in$ *measurable* N $(M\ i)$
shows $(\lambda x.\ \lambda i{\in}I.\ X\ i\ x) \in$ *measurable* N $(Pi_M\ I\ M)$
⟨*proof*⟩

lemma *measurable-abs-UNIV*:
$(\bigwedge n.\ (\lambda\omega.\ f\ n\ \omega) \in$ *measurable* M $(N\ n)) \Longrightarrow (\lambda\omega\ n.\ f\ n\ \omega) \in$ *measurable* M $(PiM\ UNIV\ N)$
⟨*proof*⟩

lemma *measurable-restrict-subset*: $J \subseteq L \Longrightarrow (\lambda f.$ *restrict* f $J) \in$ *measurable* $(Pi_M\ L\ M)\ (Pi_M\ J\ M)$
⟨*proof*⟩

lemma *measurable-restrict-subset'*:
assumes $J \subseteq L$ $\bigwedge x.\ x \in J \Longrightarrow$ *sets* $(M\ x) =$ *sets* $(N\ x)$
shows $(\lambda f.$ *restrict* f $J) \in$ *measurable* $(Pi_M\ L\ M)\ (Pi_M\ J\ N)$
⟨*proof*⟩

lemma *measurable-prod-emb*[*intro*, *simp*]:
$J \subseteq L \Longrightarrow X \in$ *sets* $(Pi_M\ J\ M) \Longrightarrow$ *prod-emb* L M J $X \in$ *sets* $(Pi_M\ L\ M)$
⟨*proof*⟩

lemma *merge-in-prod-emb*:
assumes $y \in space\ (PiM\ I\ M)\ x \in X$ **and** X: $X \in sets\ (Pi_M\ J\ M)$ **and** $J \subseteq I$
shows $merge\ J\ I\ (x,\ y) \in prod\text{-}emb\ I\ M\ J\ X$
⟨*proof*⟩

lemma *prod-emb-eq-emptyD*:
assumes J: $J \subseteq I$ **and** *ne*: $space\ (PiM\ I\ M) \neq \{\}$ **and** X: $X \in sets\ (Pi_M\ J\ M)$
and $*$: $prod\text{-}emb\ I\ M\ J\ X = \{\}$
shows $X = \{\}$
⟨*proof*⟩

lemma *sets-in-Pi-aux*:
$finite\ I \Longrightarrow (\bigwedge j.\ j \in I \Longrightarrow \{x{\in}space\ (M\ j).\ x \in F\ j\} \in sets\ (M\ j)) \Longrightarrow$
$\{x{\in}space\ (PiM\ I\ M).\ x \in Pi\ I\ F\} \in sets\ (PiM\ I\ M)$
⟨*proof*⟩

lemma *sets-in-Pi*[*measurable* (*raw*)]:
$finite\ I \Longrightarrow f \in measurable\ N\ (PiM\ I\ M) \Longrightarrow$
$(\bigwedge j.\ j \in I \Longrightarrow \{x{\in}space\ (M\ j).\ x \in F\ j\} \in sets\ (M\ j)) \Longrightarrow$
$Measurable.pred\ N\ (\lambda x.\ f\ x \in Pi\ I\ F)$
⟨*proof*⟩

lemma *sets-in-extensional-aux*:
$\{x{\in}space\ (PiM\ I\ M).\ x \in extensional\ I\} \in sets\ (PiM\ I\ M)$
⟨*proof*⟩

lemma *sets-in-extensional*[*measurable* (*raw*)]:
$f \in measurable\ N\ (PiM\ I\ M) \Longrightarrow Measurable.pred\ N\ (\lambda x.\ f\ x \in extensional\ I)$
⟨*proof*⟩

lemma *sets-PiM-I-countable*:
assumes I: $countable\ I$ **and** E: $\bigwedge i.\ i \in I \Longrightarrow E\ i \in sets\ (M\ i)$ **shows** $Pi_E\ I\ E \in sets\ (Pi_M\ I\ M)$
⟨*proof*⟩

lemma *sets-PiM-D-countable*:
assumes A: $A \in PiM\ I\ M$
shows $\exists J{\subseteq}I.\ \exists X{\in}PiM\ J\ M.\ countable\ J \wedge A = prod\text{-}emb\ I\ M\ J\ X$
⟨*proof*⟩

lemma *measure-eqI-PiM-finite*:
assumes [*simp*]: $finite\ I\ sets\ P = PiM\ I\ M\ sets\ Q = PiM\ I\ M$
assumes *eq*: $\bigwedge A.\ (\bigwedge i.\ i \in I \Longrightarrow A\ i \in sets\ (M\ i)) \Longrightarrow P\ (Pi_E\ I\ A) = Q\ (Pi_E\ I\ A)$
assumes A: $range\ A \subseteq prod\text{-}algebra\ I\ M\ (\bigcup i.\ A\ i) = space\ (PiM\ I\ M)\ \bigwedge i{::}nat.\ P\ (A\ i) \neq \infty$
shows $P = Q$

⟨*proof*⟩

lemma *measure-eqI-PiM-infinite*:
assumes [*simp*]: *sets P = PiM I M sets Q = PiM I M*
assumes *eq*: $\bigwedge A\ J.\ finite\ J \Longrightarrow J \subseteq I \Longrightarrow (\bigwedge i.\ i \in J \Longrightarrow A\ i \in sets\ (M\ i)) \Longrightarrow$
$P\ (prod\text{-}emb\ I\ M\ J\ (Pi_E\ J\ A)) = Q\ (prod\text{-}emb\ I\ M\ J\ (Pi_E\ J\ A))$
assumes *A*: *finite-measure P*
shows *P = Q*
⟨*proof*⟩

locale *product-sigma-finite* =
fixes $M :: 'i \Rightarrow 'a\ measure$
assumes *sigma-finite-measures*: $\bigwedge i.\ sigma\text{-}finite\text{-}measure\ (M\ i)$

sublocale *product-sigma-finite* ⊆ *M?*: *sigma-finite-measure M i* **for** *i*
⟨*proof*⟩

locale *finite-product-sigma-finite* = *product-sigma-finite M* **for** $M :: 'i \Rightarrow 'a$ *measure* +
fixes $I :: 'i\ set$
assumes *finite-index*: *finite I*

lemma (**in** *finite-product-sigma-finite*) *sigma-finite-pairs*:
$\exists F::'i \Rightarrow nat \Rightarrow 'a\ set.$
$(\forall i \in I.\ range\ (F\ i) \subseteq sets\ (M\ i)) \wedge$
$(\forall k.\ \forall i \in I.\ emeasure\ (M\ i)\ (F\ i\ k) \neq \infty) \wedge incseq\ (\lambda k.\ \Pi_E\ i \in I.\ F\ i\ k) \wedge$
$(\bigcup k.\ \Pi_E\ i \in I.\ F\ i\ k) = space\ (PiM\ I\ M)$
⟨*proof*⟩

lemma *emeasure-PiM-empty*[*simp*]: $emeasure\ (PiM\ \{\}\ M)\ \{\lambda\text{-}.\ undefined\} = 1$
⟨*proof*⟩

lemma *PiM-empty*: $PiM\ \{\}\ M = count\text{-}space\ \{\lambda\text{-}.\ undefined\}$
⟨*proof*⟩

lemma (**in** *product-sigma-finite*) *emeasure-PiM*:
$finite\ I \Longrightarrow (\bigwedge i.\ i \in I \Longrightarrow A\ i \in sets\ (M\ i)) \Longrightarrow emeasure\ (PiM\ I\ M)\ (Pi_E\ I\ A) = (\prod i \in I.\ emeasure\ (M\ i)\ (A\ i))$
⟨*proof*⟩

lemma (**in** *product-sigma-finite*) *PiM-eqI*:
assumes *I*[*simp*]: *finite I* **and** *P*: *sets P = PiM I M*
assumes *eq*: $\bigwedge A.\ (\bigwedge i.\ i \in I \Longrightarrow A\ i \in sets\ (M\ i)) \Longrightarrow P\ (Pi_E\ I\ A) = (\prod i \in I.\ emeasure\ (M\ i)\ (A\ i))$
shows *P = PiM I M*
⟨*proof*⟩

lemma (**in** *product-sigma-finite*) *sigma-finite*:

assumes *finite I*
shows *sigma-finite-measure* (*PiM I M*)
⟨*proof*⟩

sublocale *finite-product-sigma-finite* ⊆ *sigma-finite-measure* Pi_M *I M*
⟨*proof*⟩

lemma (**in** *finite-product-sigma-finite*) *measure-times*:
$(\bigwedge i.\ i \in I \Longrightarrow A\ i \in sets\ (M\ i)) \Longrightarrow emeasure\ (Pi_M\ I\ M)\ (Pi_E\ I\ A) = (\prod i{\in}I.\ emeasure\ (M\ i)\ (A\ i))$
⟨*proof*⟩

lemma (**in** *product-sigma-finite*) *nn-integral-empty*:
$0 \le f\ (\lambda k.\ undefined) \Longrightarrow integral^N\ (Pi_M\ \{\}\ M)\ f = f\ (\lambda k.\ undefined)$
⟨*proof*⟩

lemma (**in** *product-sigma-finite*) *distr-merge*:
assumes *IJ*[*simp*]: $I \cap J = \{\}$ **and** *fin*: *finite I finite J*
shows $distr\ (Pi_M\ I\ M \bigotimes_M Pi_M\ J\ M)\ (Pi_M\ (I \cup J)\ M)\ (merge\ I\ J) = Pi_M\ (I \cup J)\ M$
(**is** *?D* = *?P*)
⟨*proof*⟩

lemma (**in** *product-sigma-finite*) *product-nn-integral-fold*:
assumes *IJ*: $I \cap J = \{\}$ *finite I finite J*
and *f*[*measurable*]: $f \in borel\text{-}measurable\ (Pi_M\ (I \cup J)\ M)$
shows $integral^N\ (Pi_M\ (I \cup J)\ M)\ f =$
$(\int^+ x.\ (\int^+ y.\ f\ (merge\ I\ J\ (x,\ y))\ \partial(Pi_M\ J\ M))\ \partial(Pi_M\ I\ M))$
⟨*proof*⟩

lemma (**in** *product-sigma-finite*) *distr-singleton*:
$distr\ (Pi_M\ \{i\}\ M)\ (M\ i)\ (\lambda x.\ x\ i) = M\ i$ (**is** *?D* = -)
⟨*proof*⟩

lemma (**in** *product-sigma-finite*) *product-nn-integral-singleton*:
assumes *f*: $f \in borel\text{-}measurable\ (M\ i)$
shows $integral^N\ (Pi_M\ \{i\}\ M)\ (\lambda x.\ f\ (x\ i)) = integral^N\ (M\ i)\ f$
⟨*proof*⟩

lemma (**in** *product-sigma-finite*) *product-nn-integral-insert*:
assumes *I*[*simp*]: *finite I* $i \notin I$
and *f*: $f \in borel\text{-}measurable\ (Pi_M\ (insert\ i\ I)\ M)$
shows $integral^N\ (Pi_M\ (insert\ i\ I)\ M)\ f = (\int^+ x.\ (\int^+ y.\ f\ (x(i := y))\ \partial(M\ i))\ \partial(Pi_M\ I\ M))$
⟨*proof*⟩

lemma (**in** *product-sigma-finite*) *product-nn-integral-insert-rev*:
assumes *I*[*simp*]: *finite I* $i \notin I$
and [*measurable*]: $f \in borel\text{-}measurable\ (Pi_M\ (insert\ i\ I)\ M)$

shows $integral^N$ $(Pi_M$ $(insert\ i\ I)$ $M)$ $f = (\int^+ y.\ (\int^+ x.\ f\ (x(i := y))\ \partial(Pi_M\ I\ M))\ \partial(M\ i))$
⟨proof⟩

lemma (**in** *product-sigma-finite*) *product-nn-integral-setprod*:
assumes *finite I* $\bigwedge i.\ i \in I \Longrightarrow f\ i \in$ *borel-measurable* $(M\ i)$
shows $(\int^+ x.\ (\prod i{\in}I.\ f\ i\ (x\ i))\ \partial Pi_M\ I\ M) = (\prod i{\in}I.\ integral^N\ (M\ i)\ (f\ i))$
⟨proof⟩

lemma (**in** *product-sigma-finite*) *product-nn-integral-pair*:
assumes [*measurable*]: *case-prod f* $\in$ *borel-measurable* $(M\ x \bigotimes_M M\ y)$
assumes *xy*: $x \neq y$
shows $(\int^+\sigma.\ f\ (\sigma\ x)\ (\sigma\ y)\ \partial PiM\ \{x, y\}\ M) = (\int^+z.\ f\ (fst\ z)\ (snd\ z)\ \partial(M\ x \bigotimes_M M\ y))$
⟨proof⟩

lemma (**in** *product-sigma-finite*) *distr-component*:
distr $(M\ i)$ $(Pi_M\ \{i\}\ M)$ $(\lambda x.\ \lambda i{\in}\{i\}.\ x) = Pi_M\ \{i\}\ M$ (**is** *?D* = *?P*)
⟨proof⟩

lemma (**in** *product-sigma-finite*)
assumes *IJ*: $I \cap J = \{\}$ *finite I finite J* **and** *A*: $A \in sets\ (Pi_M\ (I \cup J)\ M)$
shows *emeasure-fold-integral*:
emeasure $(Pi_M\ (I \cup J)\ M)\ A = (\int^+x.\ emeasure\ (Pi_M\ J\ M)\ ((\lambda y.\ merge\ I\ J\ (x, y)) -` A \cap space\ (Pi_M\ J\ M))\ \partial Pi_M\ I\ M)$ (**is** *?I*)
and *emeasure-fold-measurable*:
$(\lambda x.\ emeasure\ (Pi_M\ J\ M)\ ((\lambda y.\ merge\ I\ J\ (x, y)) -` A \cap space\ (Pi_M\ J\ M))) \in$ *borel-measurable* $(Pi_M\ I\ M)$ (**is** *?B*)
⟨proof⟩

lemma *sets-Collect-single*:
$i \in I \Longrightarrow A \in sets\ (M\ i) \Longrightarrow \{\ x \in space\ (Pi_M\ I\ M).\ x\ i \in A\ \} \in sets\ (Pi_M\ I\ M)$
⟨proof⟩

lemma *pair-measure-eq-distr-PiM*:
fixes *M1* :: *'a measure* **and** *M2* :: *'a measure*
assumes *sigma-finite-measure M1 sigma-finite-measure M2*
shows $(M1 \bigotimes_M M2) = distr\ (Pi_M\ UNIV\ (case\text{-}bool\ M1\ M2))\ (M1 \bigotimes_M M2)\ (\lambda x.\ (x\ True,\ x\ False))$
(**is** *?P* = *?D*)
⟨proof⟩

end

8 Bochner Integration for Vector-Valued Functions

theory *Bochner-Integration*
imports *Finite-Product-Measure*

begin

In the following development of the Bochner integral we use second countable topologies instead of separable spaces. A second countable topology is also separable.

lemma *borel-measurable-implies-sequence-metric*:
fixes $f :: 'a \Rightarrow 'b :: \{metric\text{-}space,\ second\text{-}countable\text{-}topology\}$
assumes [*measurable*]: $f \in borel\text{-}measurable\ M$
shows $\exists F.\ (\forall i.\ simple\text{-}function\ M\ (F\ i)) \wedge (\forall x{\in}space\ M.\ (\lambda i.\ F\ i\ x) \longrightarrow f\ x) \wedge$
$(\forall i.\ \forall x{\in}space\ M.\ dist\ (F\ i\ x)\ z \leq 2 * dist\ (f\ x)\ z)$
⟨*proof*⟩

lemma
fixes $f :: 'a \Rightarrow 'b{::}semiring\text{-}1$ **assumes** *finite A*
shows *setsum-mult-indicator*[*simp*]: $(\sum x \in A.\ f\ x * indicator\ (B\ x)\ (g\ x)) = (\sum x{\in}\{x{\in}A.\ g\ x \in B\ x\}.\ f\ x)$
and *setsum-indicator-mult*[*simp*]: $(\sum x \in A.\ indicator\ (B\ x)\ (g\ x) * f\ x) = (\sum x{\in}\{x{\in}A.\ g\ x \in B\ x\}.\ f\ x)$
⟨*proof*⟩

lemma *borel-measurable-induct-real*[*consumes 2*, *case-names set mult add seq*]:
fixes $P :: ('a \Rightarrow real) \Rightarrow bool$
assumes *u*: $u \in borel\text{-}measurable\ M\ \bigwedge x.\ 0 \leq u\ x$
assumes *set*: $\bigwedge A.\ A \in sets\ M \Longrightarrow P\ (indicator\ A)$
assumes *mult*: $\bigwedge u\ c.\ 0 \leq c \Longrightarrow u \in borel\text{-}measurable\ M \Longrightarrow (\bigwedge x.\ 0 \leq u\ x) \Longrightarrow P\ u \Longrightarrow P\ (\lambda x.\ c * u\ x)$
assumes *add*: $\bigwedge u\ v.\ u \in borel\text{-}measurable\ M \Longrightarrow (\bigwedge x.\ 0 \leq u\ x) \Longrightarrow P\ u \Longrightarrow v \in borel\text{-}measurable\ M \Longrightarrow (\bigwedge x.\ 0 \leq v\ x) \Longrightarrow (\bigwedge x.\ x \in space\ M \Longrightarrow u\ x = 0 \vee v\ x = 0) \Longrightarrow P\ v \Longrightarrow P\ (\lambda x.\ v\ x + u\ x)$
assumes *seq*: $\bigwedge U.\ (\bigwedge i.\ U\ i \in borel\text{-}measurable\ M) \Longrightarrow (\bigwedge i\ x.\ 0 \leq U\ i\ x) \Longrightarrow (\bigwedge i.\ P\ (U\ i)) \Longrightarrow incseq\ U \Longrightarrow (\bigwedge x.\ x \in space\ M \Longrightarrow (\lambda i.\ U\ i\ x) \longrightarrow u\ x) \Longrightarrow P\ u$
shows $P\ u$
⟨*proof*⟩

lemma *scaleR-cong-right*:
fixes $x :: 'a :: real\text{-}vector$
shows $(x \neq 0 \Longrightarrow r = p) \Longrightarrow r *_R x = p *_R x$
⟨*proof*⟩

inductive *simple-bochner-integrable* $:: 'a\ measure \Rightarrow ('a \Rightarrow 'b{::}real\text{-}vector) \Rightarrow bool$
for *M f* **where**
$simple\text{-}function\ M\ f \Longrightarrow emeasure\ M\ \{y{\in}space\ M.\ f\ y \neq 0\} \neq \infty \Longrightarrow$
$simple\text{-}bochner\text{-}integrable\ M\ f$

lemma *simple-bochner-integrable-compose2*:
assumes *p-0*: $p\ 0\ 0 = 0$
shows $simple\text{-}bochner\text{-}integrable\ M\ f \Longrightarrow simple\text{-}bochner\text{-}integrable\ M\ g \Longrightarrow$

simple-bochner-integrable M $(\lambda x.\ p\ (f\ x)\ (g\ x))$
⟨*proof*⟩

lemma *simple-function-finite-support*:
assumes *f*: *simple-function M f* **and** *fin*: $(\int^+ x.\ f\ x\ \partial M) < \infty$ **and** *nn*: $\bigwedge x.\ 0 \leq f\ x$
shows *emeasure M* $\{x \in space\ M.\ f\ x \neq 0\} \neq \infty$
⟨*proof*⟩

lemma *simple-bochner-integrableI-bounded*:
assumes *f*: *simple-function M f* **and** *fin*: $(\int^+ x.\ norm\ (f\ x)\ \partial M) < \infty$
shows *simple-bochner-integrable M f*
⟨*proof*⟩

definition *simple-bochner-integral* :: $'a$ *measure* $\Rightarrow$ $('a \Rightarrow 'b::real\text{-}vector) \Rightarrow 'b$
where
simple-bochner-integral M f $= (\sum y \in f\text{`}space\ M.\ measure\ M\ \{x \in space\ M.\ f\ x = y\} *_R y)$

lemma *simple-bochner-integral-partition*:
assumes *f*: *simple-bochner-integrable M f* **and** *g*: *simple-function M g*
assumes *sub*: $\bigwedge x\ y.\ x \in space\ M \Longrightarrow y \in space\ M \Longrightarrow g\ x = g\ y \Longrightarrow f\ x = f\ y$
assumes *v*: $\bigwedge x.\ x \in space\ M \Longrightarrow f\ x = v\ (g\ x)$
shows *simple-bochner-integral M f* $= (\sum y \in g\ \text{`}\ space\ M.\ measure\ M\ \{x \in space\ M.\ g\ x = y\} *_R v\ y)$
(**is** - = *?r*)
⟨*proof*⟩

lemma *simple-bochner-integral-add*:
assumes *f*: *simple-bochner-integrable M f* **and** *g*: *simple-bochner-integrable M g*
shows *simple-bochner-integral M* $(\lambda x.\ f\ x + g\ x) =$
simple-bochner-integral M f + *simple-bochner-integral M g*
⟨*proof*⟩

lemma (**in** *linear*) *simple-bochner-integral-linear*:
assumes *g*: *simple-bochner-integrable M g*
shows *simple-bochner-integral M* $(\lambda x.\ f\ (g\ x)) = f$ (*simple-bochner-integral M g*)
⟨*proof*⟩

lemma *simple-bochner-integral-minus*:
assumes *f*: *simple-bochner-integrable M f*
shows *simple-bochner-integral M* $(\lambda x.\ -\ f\ x) = -$ *simple-bochner-integral M f*
⟨*proof*⟩

lemma *simple-bochner-integral-diff*:
assumes *f*: *simple-bochner-integrable M f* **and** *g*: *simple-bochner-integrable M g*
shows *simple-bochner-integral M* $(\lambda x.\ f\ x - g\ x) =$
simple-bochner-integral M f − *simple-bochner-integral M g*

⟨proof⟩

lemma *simple-bochner-integral-norm-bound*:
assumes *f*: *simple-bochner-integrable M f*
shows *norm* (*simple-bochner-integral M f*) ≤ *simple-bochner-integral M* (λ*x*. *norm* (*f x*))
⟨proof⟩

lemma *simple-bochner-integral-nonneg*[*simp*]:
fixes *f* :: ′*a* ⇒ *real*
shows (⋀*x*. *0* ≤ *f x*) ⟹ *0* ≤ *simple-bochner-integral M f*
⟨proof⟩

lemma *simple-bochner-integral-eq-nn-integral*:
assumes *f*: *simple-bochner-integrable M f* ⋀*x*. *0* ≤ *f x*
shows *simple-bochner-integral M f* = (∫⁺*x*. *f x* ∂*M*)
⟨proof⟩

lemma *simple-bochner-integral-bounded*:
fixes *f* :: ′*a* ⇒ ′*b*::{*real-normed-vector*, *second-countable-topology*}
assumes *f*[*measurable*]: *f* ∈ *borel-measurable M*
assumes *s*: *simple-bochner-integrable M s* **and** *t*: *simple-bochner-integrable M t*
shows *ennreal* (*norm* (*simple-bochner-integral M s* − *simple-bochner-integral M t*)) ≤
(∫⁺ *x*. *norm* (*f x* − *s x*) ∂*M*) + (∫⁺ *x*. *norm* (*f x* − *t x*) ∂*M*)
(**is** *ennreal* (*norm* (*?s* − *?t*)) ≤ *?S* + *?T*)
⟨proof⟩

inductive *has-bochner-integral* :: ′*a measure* ⇒ (′*a* ⇒ ′*b*) ⇒ ′*b*::{*real-normed-vector*, *second-countable-topology*} ⇒ *bool*
for *M f x* **where**
f ∈ *borel-measurable M* ⟹
(⋀*i*. *simple-bochner-integrable M* (*s i*)) ⟹
(λ*i*. ∫⁺*x*. *norm* (*f x* − *s i x*) ∂*M*) ⟶ *0* ⟹
(λ*i*. *simple-bochner-integral M* (*s i*)) ⟶ *x* ⟹
has-bochner-integral M f x

lemma *has-bochner-integral-cong*:
assumes *M* = *N* ⋀*x*. *x* ∈ *space N* ⟹ *f x* = *g x x* = *y*
shows *has-bochner-integral M f x* ⟷ *has-bochner-integral N g y*
⟨proof⟩

lemma *has-bochner-integral-cong-AE*:
f ∈ *borel-measurable M* ⟹ *g* ∈ *borel-measurable M* ⟹ (*AE x in M*. *f x* = *g x*) ⟹
has-bochner-integral M f x ⟷ *has-bochner-integral M g x*
⟨proof⟩

lemma *borel-measurable-has-bochner-integral*:

has-bochner-integral M f x $\Longrightarrow$ *f* $\in$ *borel-measurable M*
⟨*proof*⟩

lemma *borel-measurable-has-bochner-integral'*[*measurable-dest*]:
has-bochner-integral M f x $\Longrightarrow$ *g* $\in$ *measurable N M* $\Longrightarrow$ (λx. *f* (*g x*)) $\in$ *borel-measurable N*
⟨*proof*⟩

lemma *has-bochner-integral-simple-bochner-integrable*:
simple-bochner-integrable M f $\Longrightarrow$ *has-bochner-integral M f* (*simple-bochner-integral M f*)
⟨*proof*⟩

lemma *has-bochner-integral-real-indicator*:
assumes [*measurable*]: *A* $\in$ *sets M* **and** *A*: *emeasure M A* $< \infty$
shows *has-bochner-integral M* (*indicator A*) (*measure M A*)
⟨*proof*⟩

lemma *has-bochner-integral-add*[*intro*]:
has-bochner-integral M f x $\Longrightarrow$ *has-bochner-integral M g y* $\Longrightarrow$
has-bochner-integral M (λx. *f x* + *g x*) (*x* + *y*)
⟨*proof*⟩

lemma *has-bochner-integral-bounded-linear*:
assumes *bounded-linear T*
shows *has-bochner-integral M f x* $\Longrightarrow$ *has-bochner-integral M* (λx. *T* (*f x*)) (*T x*)
⟨*proof*⟩

lemma *has-bochner-integral-zero*[*intro*]: *has-bochner-integral M* (λx. *0*) *0*
⟨*proof*⟩

lemma *has-bochner-integral-scaleR-left*[*intro*]:
(*c* $\neq$ *0* $\Longrightarrow$ *has-bochner-integral M f x*) $\Longrightarrow$ *has-bochner-integral M* (λx. *f x* $*_R$ *c*) (*x* $*_R$ *c*)
⟨*proof*⟩

lemma *has-bochner-integral-scaleR-right*[*intro*]:
(*c* $\neq$ *0* $\Longrightarrow$ *has-bochner-integral M f x*) $\Longrightarrow$ *has-bochner-integral M* (λx. *c* $*_R$ *f x*) (*c* $*_R$ *x*)
⟨*proof*⟩

lemma *has-bochner-integral-mult-left*[*intro*]:
fixes *c* :: -::{*real-normed-algebra*,*second-countable-topology*}
shows (*c* $\neq$ *0* $\Longrightarrow$ *has-bochner-integral M f x*) $\Longrightarrow$ *has-bochner-integral M* (λx. *f x* $*$ *c*) (*x* $*$ *c*)
⟨*proof*⟩

lemma *has-bochner-integral-mult-right*[*intro*]:

fixes *c* :: -::{*real-normed-algebra,second-countable-topology*}
shows $(c \neq 0 \Longrightarrow$ *has-bochner-integral M f x*$) \Longrightarrow$ *has-bochner-integral M* $(\lambda x.\ c * f\ x)\ (c * x)$
⟨*proof*⟩

lemmas *has-bochner-integral-divide* =
has-bochner-integral-bounded-linear[*OF bounded-linear-divide*]

lemma *has-bochner-integral-divide-zero*[*intro*]:
fixes *c* :: -::{*real-normed-field*, *field*, *second-countable-topology*}
shows $(c \neq 0 \Longrightarrow$ *has-bochner-integral M f x*$) \Longrightarrow$ *has-bochner-integral M* $(\lambda x.\ f\ x\ /\ c)\ (x\ /\ c)$
⟨*proof*⟩

lemma *has-bochner-integral-inner-left*[*intro*]:
$(c \neq 0 \Longrightarrow$ *has-bochner-integral M f x*$) \Longrightarrow$ *has-bochner-integral M* $(\lambda x.\ f\ x \cdot c)\ (x \cdot c)$
⟨*proof*⟩

lemma *has-bochner-integral-inner-right*[*intro*]:
$(c \neq 0 \Longrightarrow$ *has-bochner-integral M f x*$) \Longrightarrow$ *has-bochner-integral M* $(\lambda x.\ c \cdot f\ x)\ (c \cdot x)$
⟨*proof*⟩

lemmas *has-bochner-integral-minus* =
has-bochner-integral-bounded-linear[*OF bounded-linear-minus*[*OF bounded-linear-ident*]]
lemmas *has-bochner-integral-Re* =
has-bochner-integral-bounded-linear[*OF bounded-linear-Re*]
lemmas *has-bochner-integral-Im* =
has-bochner-integral-bounded-linear[*OF bounded-linear-Im*]
lemmas *has-bochner-integral-cnj* =
has-bochner-integral-bounded-linear[*OF bounded-linear-cnj*]
lemmas *has-bochner-integral-of-real* =
has-bochner-integral-bounded-linear[*OF bounded-linear-of-real*]
lemmas *has-bochner-integral-fst* =
has-bochner-integral-bounded-linear[*OF bounded-linear-fst*]
lemmas *has-bochner-integral-snd* =
has-bochner-integral-bounded-linear[*OF bounded-linear-snd*]

lemma *has-bochner-integral-indicator*:
$A \in$ *sets M* $\Longrightarrow$ *emeasure M A* $< \infty \Longrightarrow$
has-bochner-integral M $(\lambda x.$ *indicator A x* $*_R\ c)$ (*measure M A* $*_R\ c)$
⟨*proof*⟩

lemma *has-bochner-integral-diff*:
has-bochner-integral M f x $\Longrightarrow$ *has-bochner-integral M g y* $\Longrightarrow$
has-bochner-integral M $(\lambda x.\ f\ x - g\ x)\ (x - y)$
⟨*proof*⟩

lemma *has-bochner-integral-setsum*:
$(\bigwedge i.\ i \in I \Longrightarrow \textit{has-bochner-integral } M\ (f\ i)\ (x\ i)) \Longrightarrow$
$\textit{has-bochner-integral } M\ (\lambda x.\ \sum i{\in}I.\ f\ i\ x)\ (\sum i{\in}I.\ x\ i)$
⟨*proof*⟩

lemma *has-bochner-integral-implies-finite-norm*:
$\textit{has-bochner-integral } M\ f\ x \Longrightarrow (\int^{+} x.\ \textit{norm}\ (f\ x)\ \partial M) < \infty$
⟨*proof*⟩

lemma *has-bochner-integral-norm-bound*:
assumes *i*: *has-bochner-integral M f x*
shows $\textit{norm } x \le (\int^{+} x.\ \textit{norm}\ (f\ x)\ \partial M)$
⟨*proof*⟩

lemma *has-bochner-integral-eq*:
$\textit{has-bochner-integral } M\ f\ x \Longrightarrow \textit{has-bochner-integral } M\ f\ y \Longrightarrow x = y$
⟨*proof*⟩

lemma *has-bochner-integralI-AE*:
assumes *f*: *has-bochner-integral M f x*
and *g*: $g \in \textit{borel-measurable } M$
and *ae*: $\textit{AE } x \textit{ in } M.\ f\ x = g\ x$
shows *has-bochner-integral M g x*
⟨*proof*⟩

lemma *has-bochner-integral-eq-AE*:
assumes *f*: *has-bochner-integral M f x*
and *g*: *has-bochner-integral M g y*
and *ae*: $\textit{AE } x \textit{ in } M.\ f\ x = g\ x$
shows $x = y$
⟨*proof*⟩

lemma *simple-bochner-integrable-restrict-space*:
fixes $f :: - \Rightarrow {'b}{::}\textit{real-normed-vector}$
assumes Ω: $\Omega \cap \textit{space } M \in \textit{sets } M$
shows $\textit{simple-bochner-integrable } (\textit{restrict-space } M\ \Omega)\ f \longleftrightarrow$
$\textit{simple-bochner-integrable } M\ (\lambda x.\ \textit{indicator } \Omega\ x *_R f\ x)$
⟨*proof*⟩

lemma *simple-bochner-integral-restrict-space*:
fixes $f :: - \Rightarrow {'b}{::}\textit{real-normed-vector}$
assumes Ω: $\Omega \cap \textit{space } M \in \textit{sets } M$
assumes *f*: $\textit{simple-bochner-integrable } (\textit{restrict-space } M\ \Omega)\ f$
shows $\textit{simple-bochner-integral } (\textit{restrict-space } M\ \Omega)\ f =$
$\textit{simple-bochner-integral } M\ (\lambda x.\ \textit{indicator } \Omega\ x *_R f\ x)$
⟨*proof*⟩

context
notes [[*inductive-internals*]]

begin

inductive *integrable* **for** *M f* **where**
has-bochner-integral M f x $\Longrightarrow$ *integrable M f*

end

definition *lebesgue-integral* (*integral*^{L}) **where**
integral^L *M f* = (*if* $\exists$ *x*. *has-bochner-integral M f x then THE x*. *has-bochner-integral M f x else 0*)

syntax
-lebesgue-integral :: *pttrn* $\Rightarrow$ *real* $\Rightarrow$ *'a measure* $\Rightarrow$ *real* ($\int$ ((*2* -./ -)/ ∂-) [*60*,*61*] *110*)

translations
$\int$ *x*. *f* ∂*M* == *CONST lebesgue-integral M* (λ*x*. *f*)

syntax
-ascii-lebesgue-integral :: *pttrn* $\Rightarrow$ *'a measure* $\Rightarrow$ *real* $\Rightarrow$ *real* ((*3LINT* (*1*-)/|(-)./ -) [*0*,*110*,*60*] *60*)

translations
LINT x|*M*. *f* == *CONST lebesgue-integral M* (λ*x*. *f*)

lemma *has-bochner-integral-integral-eq*: *has-bochner-integral M f x* $\Longrightarrow$ *integral*^{L} *M f* = *x*
⟨*proof*⟩

lemma *has-bochner-integral-integrable*:
integrable M f $\Longrightarrow$ *has-bochner-integral M f* (*integral*^{L} *M f*)
⟨*proof*⟩

lemma *has-bochner-integral-iff*:
has-bochner-integral M f x $\longleftrightarrow$ *integrable M f* $\wedge$ *integral*^{L} *M f* = *x*
⟨*proof*⟩

lemma *simple-bochner-integrable-eq-integral*:
simple-bochner-integrable M f $\Longrightarrow$ *simple-bochner-integral M f* = *integral*^{L} *M f*
⟨*proof*⟩

lemma *not-integrable-integral-eq*: $\neg$ *integrable M f* $\Longrightarrow$ *integral*^{L} *M f* = *0*
⟨*proof*⟩

lemma *integral-eq-cases*:
integrable M f $\longleftrightarrow$ *integrable N g* $\Longrightarrow$
(*integrable M f* $\Longrightarrow$ *integrable N g* $\Longrightarrow$ *integral*^{L} *M f* = *integral*^{L} *N g*) $\Longrightarrow$
integral^L *M f* = *integral*^{L} *N g*
⟨*proof*⟩

lemma *borel-measurable-integrable*[*measurable-dest*]: *integrable* $M f \Longrightarrow f \in$ *borel-measurable* M
⟨*proof*⟩

lemma *borel-measurable-integrable'*[*measurable-dest*]:
integrable $M f \Longrightarrow g \in$ *measurable* $N M \Longrightarrow (\lambda x.\ f\ (g\ x)) \in$ *borel-measurable* N
⟨*proof*⟩

lemma *integrable-cong*:
$M = N \Longrightarrow (\bigwedge x.\ x \in$ *space* $N \Longrightarrow f\,x = g\,x) \Longrightarrow$ *integrable* $M f \longleftrightarrow$ *integrable* $N\ g$
⟨*proof*⟩

lemma *integrable-cong-AE*:
$f \in$ *borel-measurable* $M \Longrightarrow g \in$ *borel-measurable* $M \Longrightarrow AE\ x\ in\ M.\ f\,x = g\,x \Longrightarrow$
integrable $M f \longleftrightarrow$ *integrable* $M\ g$
⟨*proof*⟩

lemma *integral-cong*:
$M = N \Longrightarrow (\bigwedge x.\ x \in$ *space* $N \Longrightarrow f\,x = g\,x) \Longrightarrow integral^L\ M f = integral^L\ N\ g$
⟨*proof*⟩

lemma *integral-cong-AE*:
$f \in$ *borel-measurable* $M \Longrightarrow g \in$ *borel-measurable* $M \Longrightarrow AE\ x\ in\ M.\ f\,x = g\,x \Longrightarrow$
$integral^L\ M f = integral^L\ M\ g$
⟨*proof*⟩

lemma *integrable-add*[*simp*, *intro*]: *integrable* $M\ f \Longrightarrow$ *integrable* $M\ g \Longrightarrow$ *integrable* $M\ (\lambda x.\ f\,x + g\ x)$
⟨*proof*⟩

lemma *integrable-zero*[*simp*, *intro*]: *integrable* $M\ (\lambda x.\ 0)$
⟨*proof*⟩

lemma *integrable-setsum*[*simp*, *intro*]: $(\bigwedge i.\ i \in I \Longrightarrow$ *integrable* $M\ (f\ i)) \Longrightarrow$ *integrable* $M\ (\lambda x.\ \sum i{\in}I.\ f\ i\ x)$
⟨*proof*⟩

lemma *integrable-indicator*[*simp*, *intro*]: $A \in$ *sets* $M \Longrightarrow$ *emeasure* $M\ A < \infty \Longrightarrow$
integrable $M\ (\lambda x.$ *indicator* $A\ x *_R c)$
⟨*proof*⟩

lemma *integrable-real-indicator*[*simp*, *intro*]: $A \in$ *sets* $M \Longrightarrow$ *emeasure* $M\ A < \infty \Longrightarrow$
integrable M (*indicator* A :: $'a \Rightarrow$ *real*)

⟨*proof*⟩

lemma *integrable-diff*[*simp*, *intro*]: *integrable M f* $\Longrightarrow$ *integrable M g* $\Longrightarrow$ *integrable M* ($\lambda x.\ f\ x - g\ x$)
⟨*proof*⟩

lemma *integrable-bounded-linear*: *bounded-linear T* $\Longrightarrow$ *integrable M f* $\Longrightarrow$ *integrable M* ($\lambda x.\ T\ (f\ x)$)
⟨*proof*⟩

lemma *integrable-scaleR-left*[*simp*, *intro*]: ($c \neq 0 \Longrightarrow$ *integrable M f*) $\Longrightarrow$ *integrable M* ($\lambda x.\ f\ x *_R c$)
⟨*proof*⟩

lemma *integrable-scaleR-right*[*simp*, *intro*]: ($c \neq 0 \Longrightarrow$ *integrable M f*) $\Longrightarrow$ *integrable M* ($\lambda x.\ c *_R f\ x$)
⟨*proof*⟩

lemma *integrable-mult-left*[*simp*, *intro*]:
fixes *c* :: -::{*real-normed-algebra*,*second-countable-topology*}
shows ($c \neq 0 \Longrightarrow$ *integrable M f*) $\Longrightarrow$ *integrable M* ($\lambda x.\ f\ x * c$)
⟨*proof*⟩

lemma *integrable-mult-right*[*simp*, *intro*]:
fixes *c* :: -::{*real-normed-algebra*,*second-countable-topology*}
shows ($c \neq 0 \Longrightarrow$ *integrable M f*) $\Longrightarrow$ *integrable M* ($\lambda x.\ c * f\ x$)
⟨*proof*⟩

lemma *integrable-divide-zero*[*simp*, *intro*]:
fixes *c* :: -::{*real-normed-field*, *field*, *second-countable-topology*}
shows ($c \neq 0 \Longrightarrow$ *integrable M f*) $\Longrightarrow$ *integrable M* ($\lambda x.\ f\ x\ /\ c$)
⟨*proof*⟩

lemma *integrable-inner-left*[*simp*, *intro*]:
($c \neq 0 \Longrightarrow$ *integrable M f*) $\Longrightarrow$ *integrable M* ($\lambda x.\ f\ x \cdot c$)
⟨*proof*⟩

lemma *integrable-inner-right*[*simp*, *intro*]:
($c \neq 0 \Longrightarrow$ *integrable M f*) $\Longrightarrow$ *integrable M* ($\lambda x.\ c \cdot f\ x$)
⟨*proof*⟩

lemmas *integrable-minus*[*simp*, *intro*] =
integrable-bounded-linear[*OF bounded-linear-minus*[*OF bounded-linear-ident*]]
lemmas *integrable-divide*[*simp*, *intro*] =
integrable-bounded-linear[*OF bounded-linear-divide*]
lemmas *integrable-Re*[*simp*, *intro*] =
integrable-bounded-linear[*OF bounded-linear-Re*]
lemmas *integrable-Im*[*simp*, *intro*] =
integrable-bounded-linear[*OF bounded-linear-Im*]

lemmas *integrable-cnj*[*simp*, *intro*] =
integrable-bounded-linear[*OF bounded-linear-cnj*]
lemmas *integrable-of-real*[*simp*, *intro*] =
integrable-bounded-linear[*OF bounded-linear-of-real*]
lemmas *integrable-fst*[*simp*, *intro*] =
integrable-bounded-linear[*OF bounded-linear-fst*]
lemmas *integrable-snd*[*simp*, *intro*] =
integrable-bounded-linear[*OF bounded-linear-snd*]

lemma *integral-zero*[*simp*]: $integral^L\ M\ (\lambda x.\ 0) = 0$
⟨*proof*⟩

lemma *integral-add*[*simp*]: $integrable\ M\ f \Longrightarrow integrable\ M\ g \Longrightarrow$
$integral^L\ M\ (\lambda x.\ f\ x + g\ x) = integral^L\ M\ f + integral^L\ M\ g$
⟨*proof*⟩

lemma *integral-diff*[*simp*]: $integrable\ M\ f \Longrightarrow integrable\ M\ g \Longrightarrow$
$integral^L\ M\ (\lambda x.\ f\ x - g\ x) = integral^L\ M\ f - integral^L\ M\ g$
⟨*proof*⟩

lemma *integral-setsum*: $(\bigwedge i.\ i \in I \Longrightarrow integrable\ M\ (f\ i)) \Longrightarrow$
$integral^L\ M\ (\lambda x.\ \sum i{\in}I.\ f\ i\ x) = (\sum i{\in}I.\ integral^L\ M\ (f\ i))$
⟨*proof*⟩

lemma *integral-setsum'*[*simp*]: $(\bigwedge i.\ i \in I$ =*simp*=> $integrable\ M\ (f\ i)) \Longrightarrow$
$integral^L\ M\ (\lambda x.\ \sum i{\in}I.\ f\ i\ x) = (\sum i{\in}I.\ integral^L\ M\ (f\ i))$
⟨*proof*⟩

lemma *integral-bounded-linear*: $bounded\text{-}linear\ T \Longrightarrow integrable\ M\ f \Longrightarrow$
$integral^L\ M\ (\lambda x.\ T\ (f\ x)) = T\ (integral^L\ M\ f)$
⟨*proof*⟩

lemma *integral-bounded-linear'*:
assumes T: *bounded-linear* T **and** T': *bounded-linear* T'
assumes $*$: $\neg\ (\forall x.\ T\ x = 0) \Longrightarrow (\forall x.\ T'\ (T\ x) = x)$
shows $integral^L\ M\ (\lambda x.\ T\ (f\ x)) = T\ (integral^L\ M\ f)$
⟨*proof*⟩

lemma *integral-scaleR-left*[*simp*]: $(c \neq 0 \Longrightarrow integrable\ M\ f) \Longrightarrow (\int x.\ f\ x *_R c$
$\partial M) = integral^L\ M\ f *_R c$
⟨*proof*⟩

lemma *integral-scaleR-right*[*simp*]: $(\int x.\ c *_R f\ x\ \partial M) = c *_R integral^L\ M\ f$
⟨*proof*⟩

lemma *integral-mult-left*[*simp*]:
fixes c :: -::{*real-normed-algebra*,*second-countable-topology*}
shows $(c \neq 0 \Longrightarrow integrable\ M\ f) \Longrightarrow (\int x.\ f\ x * c\ \partial M) = integral^L\ M\ f * c$
⟨*proof*⟩

lemma *integral-mult-right*[*simp*]:
fixes c :: -::{*real-normed-algebra*,*second-countable-topology*}
shows $(c \neq 0 \Longrightarrow integrable\ M\ f) \Longrightarrow (\int x.\ c * f\ x\ \partial M) = c * integral^L\ M\ f$
⟨*proof*⟩

lemma *integral-mult-left-zero*[*simp*]:
fixes c :: -::{*real-normed-field*,*second-countable-topology*}
shows $(\int x.\ f\ x * c\ \partial M) = integral^L\ M\ f * c$
⟨*proof*⟩

lemma *integral-mult-right-zero*[*simp*]:
fixes c :: -::{*real-normed-field*,*second-countable-topology*}
shows $(\int x.\ c * f\ x\ \partial M) = c * integral^L\ M\ f$
⟨*proof*⟩

lemma *integral-inner-left*[*simp*]: $(c \neq 0 \Longrightarrow integrable\ M\ f) \Longrightarrow (\int x.\ f\ x \cdot c\ \partial M) = integral^L\ M\ f \cdot c$
⟨*proof*⟩

lemma *integral-inner-right*[*simp*]: $(c \neq 0 \Longrightarrow integrable\ M\ f) \Longrightarrow (\int x.\ c \cdot f\ x\ \partial M) = c \cdot integral^L\ M\ f$
⟨*proof*⟩

lemma *integral-divide-zero*[*simp*]:
fixes c :: -::{*real-normed-field*, *field*, *second-countable-topology*}
shows $integral^L\ M\ (\lambda x.\ f\ x\ /\ c) = integral^L\ M\ f\ /\ c$
⟨*proof*⟩

lemma *integral-minus*[*simp*]: $integral^L\ M\ (\lambda x.\ -\ f\ x) = -\ integral^L\ M\ f$
⟨*proof*⟩

lemma *integral-complex-of-real*[*simp*]: $integral^L\ M\ (\lambda x.\ complex\text{-}of\text{-}real\ (f\ x)) = of\text{-}real\ (integral^L\ M\ f)$
⟨*proof*⟩

lemma *integral-cnj*[*simp*]: $integral^L\ M\ (\lambda x.\ cnj\ (f\ x)) = cnj\ (integral^L\ M\ f)$
⟨*proof*⟩

lemmas *integral-divide*[*simp*] =
integral-bounded-linear[*OF bounded-linear-divide*]
lemmas *integral-Re*[*simp*] =
integral-bounded-linear[*OF bounded-linear-Re*]
lemmas *integral-Im*[*simp*] =
integral-bounded-linear[*OF bounded-linear-Im*]
lemmas *integral-of-real*[*simp*] =
integral-bounded-linear[*OF bounded-linear-of-real*]
lemmas *integral-fst*[*simp*] =
integral-bounded-linear[*OF bounded-linear-fst*]

lemmas *integral-snd*[*simp*] =
integral-bounded-linear[*OF bounded-linear-snd*]

lemma *integral-norm-bound-ennreal*:
integrable M f $\Longrightarrow$ *norm* (*integral*^{L} *M f*) $\leq$ ($\int^+ x$. *norm* (*f x*) ∂M)
⟨*proof*⟩

lemma *integrableI-sequence*:
fixes *f* :: *'a* $\Rightarrow$ *'b*::{*banach*, *second-countable-topology*}
assumes *f*[*measurable*]: *f* $\in$ *borel-measurable M*
assumes *s*: $\bigwedge i$. *simple-bochner-integrable M* (*s i*)
assumes *lim*: (λi. $\int^+ x$. *norm* (*f x* $-$ *s i x*) ∂M) $\longrightarrow$ *0* (**is** *?S* $\longrightarrow$ *0*)
shows *integrable M f*
⟨*proof*⟩

lemma *nn-integral-dominated-convergence-norm*:
fixes *u'* :: - $\Rightarrow$ -::{*real-normed-vector*, *second-countable-topology*}
assumes [*measurable*]:
$\bigwedge i$. *u i* $\in$ *borel-measurable M u'* $\in$ *borel-measurable M w* $\in$ *borel-measurable M*
and *bound*: $\bigwedge j$. *AE x in M*. *norm* (*u j x*) $\leq$ *w x*
and *w*: ($\int^+ x$. *w x* ∂M) $< \infty$
and *u'*: *AE x in M*. (λi. *u i x*) $\longrightarrow$ *u' x*
shows (λi. ($\int^+ x$. *norm* (*u' x* $-$ *u i x*) ∂M)) $\longrightarrow$ *0*
⟨*proof*⟩

lemma *integrableI-bounded*:
fixes *f* :: *'a* $\Rightarrow$ *'b*::{*banach*, *second-countable-topology*}
assumes *f*[*measurable*]: *f* $\in$ *borel-measurable M* **and** *fin*: ($\int^+ x$. *norm* (*f x*) ∂M) $< \infty$
shows *integrable M f*
⟨*proof*⟩

lemma *integrableI-bounded-set*:
fixes *f* :: *'a* $\Rightarrow$ *'b*::{*banach*, *second-countable-topology*}
assumes [*measurable*]: *A* $\in$ *sets M f* $\in$ *borel-measurable M*
assumes *finite*: *emeasure M A* $< \infty$
and *bnd*: *AE x in M*. *x* $\in$ *A* $\longrightarrow$ *norm* (*f x*) $\leq$ *B*
and *null*: *AE x in M*. *x* $\notin$ *A* $\longrightarrow$ *f x* = *0*
shows *integrable M f*
⟨*proof*⟩

lemma *integrableI-bounded-set-indicator*:
fixes *f* :: *'a* $\Rightarrow$ *'b*::{*banach*, *second-countable-topology*}
shows *A* $\in$ *sets M* $\Longrightarrow$ *f* $\in$ *borel-measurable M* $\Longrightarrow$
emeasure M A $< \infty$ $\Longrightarrow$ (*AE x in M*. *x* $\in$ *A* $\longrightarrow$ *norm* (*f x*) $\leq$ *B*) $\Longrightarrow$
integrable M (λx. *indicator A x* $*_R$ *f x*)
⟨*proof*⟩

lemma *integrableI-nonneg*:
fixes f :: $'a \Rightarrow real$
assumes $f \in$ *borel-measurable M AE x in M*. $0 \leq f\,x$ $(\int^+ x.\ f\,x\ \partial M) < \infty$
shows *integrable M f*
⟨*proof*⟩

lemma *integrable-iff-bounded*:
fixes f :: $'a \Rightarrow 'b$::{*banach*, *second-countable-topology*}
shows *integrable M f* $\longleftrightarrow f \in$ *borel-measurable M* $\wedge$ $(\int^+ x.\ norm\ (f\,x)\ \partial M) < \infty$
⟨*proof*⟩

lemma *integrable-bound*:
fixes f :: $'a \Rightarrow 'b$::{*banach*, *second-countable-topology*}
and g :: $'a \Rightarrow 'c$::{*banach*, *second-countable-topology*}
shows *integrable M f* $\Longrightarrow g \in$ *borel-measurable M* $\Longrightarrow$ (*AE x in M*. *norm* $(g\ x) \leq$ *norm* $(f\,x)) \Longrightarrow$
integrable M g
⟨*proof*⟩

lemma *integrable-mult-indicator*:
fixes f :: $'a \Rightarrow 'b$::{*banach*, *second-countable-topology*}
shows $A \in$ *sets M* $\Longrightarrow$ *integrable M f* $\Longrightarrow$ *integrable M* $(\lambda x.\ indicator\ A\ x *_R f\ x)$
⟨*proof*⟩

lemma *integrable-real-mult-indicator*:
fixes f :: $'a \Rightarrow real$
shows $A \in$ *sets M* $\Longrightarrow$ *integrable M f* $\Longrightarrow$ *integrable M* $(\lambda x.\ f\,x * indicator\ A\ x)$
⟨*proof*⟩

lemma *integrable-abs*[*simp*, *intro*]:
fixes f :: $'a \Rightarrow real$
assumes [*measurable*]: *integrable M f* **shows** *integrable M* $(\lambda x.\ |f\,x|)$
⟨*proof*⟩

lemma *integrable-norm*[*simp*, *intro*]:
fixes f :: $'a \Rightarrow 'b$::{*banach*, *second-countable-topology*}
assumes [*measurable*]: *integrable M f* **shows** *integrable M* $(\lambda x.\ norm\ (f\,x))$
⟨*proof*⟩

lemma *integrable-norm-cancel*:
fixes f :: $'a \Rightarrow 'b$::{*banach*, *second-countable-topology*}
assumes [*measurable*]: *integrable M* $(\lambda x.\ norm\ (f\ x))$ $f \in$ *borel-measurable M*
shows *integrable M f*
⟨*proof*⟩

lemma *integrable-norm-iff*:

fixes f :: $'a \Rightarrow 'b$::{*banach*, *second-countable-topology*}
shows $f \in$ *borel-measurable* $M \Longrightarrow$ *integrable* M (λx. *norm* ($f\,x$)) $\longleftrightarrow$ *integrable* $M\,f$
⟨*proof*⟩

lemma *integrable-abs-cancel*:
fixes f :: $'a \Rightarrow$ *real*
assumes [*measurable*]: *integrable* M (λx. $|f\,x|$) $f \in$ *borel-measurable* M **shows** *integrable* $M\,f$
⟨*proof*⟩

lemma *integrable-abs-iff*:
fixes f :: $'a \Rightarrow$ *real*
shows $f \in$ *borel-measurable* $M \Longrightarrow$ *integrable* M (λx. $|f\,x|$) $\longleftrightarrow$ *integrable* $M\,f$
⟨*proof*⟩

lemma *integrable-max*[*simp*, *intro*]:
fixes f :: $'a \Rightarrow$ *real*
assumes *fg*[*measurable*]: *integrable* $M\,f$ *integrable* $M\,g$
shows *integrable* M (λx. *max* ($f\,x$) ($g\,x$))
⟨*proof*⟩

lemma *integrable-min*[*simp*, *intro*]:
fixes f :: $'a \Rightarrow$ *real*
assumes *fg*[*measurable*]: *integrable* $M\,f$ *integrable* $M\,g$
shows *integrable* M (λx. *min* ($f\,x$) ($g\,x$))
⟨*proof*⟩

lemma *integral-minus-iff*[*simp*]:
integrable M (λx. $-\,f\,x$ ::$'a$::{*banach*, *second-countable-topology*}) $\longleftrightarrow$ *integrable* $M\,f$
⟨*proof*⟩

lemma *integrable-indicator-iff*:
integrable M (*indicator* A::- $\Rightarrow$ *real*) $\longleftrightarrow$ $A \cap$ *space* $M \in$ *sets* $M \wedge$ *emeasure* M ($A \cap$ *space* M) $< \infty$
⟨*proof*⟩

lemma *integral-indicator*[*simp*]: *integral*^{L} M (*indicator* A) = *measure* M ($A \cap$ *space* M)
⟨*proof*⟩

lemma *integrable-discrete-difference*:
fixes f :: $'a \Rightarrow 'b$::{*banach*, *second-countable-topology*}
assumes X: *countable* X
assumes *null*: $\bigwedge x$. $x \in X \Longrightarrow$ *emeasure* M $\{x\}$ = *0*
assumes *sets*: $\bigwedge x$. $x \in X \Longrightarrow \{x\} \in$ *sets* M
assumes *eq*: $\bigwedge x$. $x \in$ *space* $M \Longrightarrow x \notin X \Longrightarrow f\,x = g\,x$
shows *integrable* $M\,f \longleftrightarrow$ *integrable* $M\,g$

⟨*proof*⟩

lemma *integral-discrete-difference*:
fixes $f :: 'a \Rightarrow 'b::\{banach, second\text{-}countable\text{-}topology\}$
assumes *X*: *countable X*
assumes *null*: $\bigwedge x.\ x \in X \Longrightarrow emeasure\ M\ \{x\} = 0$
assumes *sets*: $\bigwedge x.\ x \in X \Longrightarrow \{x\} \in sets\ M$
assumes *eq*: $\bigwedge x.\ x \in space\ M \Longrightarrow x \notin X \Longrightarrow f\ x = g\ x$
shows $integral^L\ M\ f = integral^L\ M\ g$
⟨*proof*⟩

lemma *has-bochner-integral-discrete-difference*:
fixes $f :: 'a \Rightarrow 'b::\{banach, second\text{-}countable\text{-}topology\}$
assumes *X*: *countable X*
assumes *null*: $\bigwedge x.\ x \in X \Longrightarrow emeasure\ M\ \{x\} = 0$
assumes *sets*: $\bigwedge x.\ x \in X \Longrightarrow \{x\} \in sets\ M$
assumes *eq*: $\bigwedge x.\ x \in space\ M \Longrightarrow x \notin X \Longrightarrow f\ x = g\ x$
shows *has-bochner-integral M f x* $\longleftrightarrow$ *has-bochner-integral M g x*
⟨*proof*⟩

lemma
fixes $f :: 'a \Rightarrow 'b::\{banach, second\text{-}countable\text{-}topology\}$ **and** $w :: 'a \Rightarrow real$
assumes $f \in borel\text{-}measurable\ M\ \bigwedge i.\ s\ i \in borel\text{-}measurable\ M\ integrable\ M\ w$
assumes *lim*: $AE\ x\ in\ M.\ (\lambda i.\ s\ i\ x) \longlonglongrightarrow f\ x$
assumes *bound*: $\bigwedge i.\ AE\ x\ in\ M.\ norm\ (s\ i\ x) \leq w\ x$
shows *integrable-dominated-convergence*: *integrable M f*
and *integrable-dominated-convergence2*: $\bigwedge i.\ integrable\ M\ (s\ i)$
and *integral-dominated-convergence*: $(\lambda i.\ integral^L\ M\ (s\ i)) \longlonglongrightarrow integral^L\ M\ f$
⟨*proof*⟩

context
fixes $s :: real \Rightarrow 'a \Rightarrow 'b::\{banach, second\text{-}countable\text{-}topology\}$ **and** $w :: 'a \Rightarrow real$
and $f :: 'a \Rightarrow 'b$ **and** *M*
assumes $f \in borel\text{-}measurable\ M\ \bigwedge t.\ s\ t \in borel\text{-}measurable\ M\ integrable\ M\ w$
assumes *lim*: $AE\ x\ in\ M.\ ((\lambda i.\ s\ i\ x) \longlongrightarrow f\ x)\ at\text{-}top$
assumes *bound*: $\forall_F\ i\ in\ at\text{-}top.\ AE\ x\ in\ M.\ norm\ (s\ i\ x) \leq w\ x$
begin

lemma *integral-dominated-convergence-at-top*: $((\lambda t.\ integral^L\ M\ (s\ t)) \longlongrightarrow integral^L\ M\ f)\ at\text{-}top$
⟨*proof*⟩

lemma *integrable-dominated-convergence-at-top*: *integrable M f*
⟨*proof*⟩

end

lemma *integrable-mult-left-iff*:
fixes $f :: 'a \Rightarrow real$
shows $integrable\ M\ (\lambda x.\ c * f\ x) \longleftrightarrow c = 0 \lor integrable\ M\ f$
⟨*proof*⟩

lemma *integrableI-nn-integral-finite*:
assumes [*measurable*]: $f \in borel\text{-}measurable\ M$
and *nonneg*: $AE\ x\ in\ M.\ 0 \leq f\ x$
and *finite*: $(\int^+ x.\ f\ x\ \partial M) = ennreal\ x$
shows $integrable\ M\ f$
⟨*proof*⟩

lemma *integral-nonneg-AE*:
fixes $f :: 'a \Rightarrow real$
assumes *nonneg*: $AE\ x\ in\ M.\ 0 \leq f\ x$
shows $0 \leq integral^L\ M\ f$
⟨*proof*⟩

lemma *integral-nonneg*[*simp*]:
fixes $f :: 'a \Rightarrow real$
shows $(\bigwedge x.\ x \in space\ M \Longrightarrow 0 \leq f\ x) \Longrightarrow 0 \leq integral^L\ M\ f$
⟨*proof*⟩

lemma *nn-integral-eq-integral*:
assumes f: $integrable\ M\ f$
assumes *nonneg*: $AE\ x\ in\ M.\ 0 \leq f\ x$
shows $(\int^+ x.\ f\ x\ \partial M) = integral^L\ M\ f$
⟨*proof*⟩

lemma
fixes $f :: \text{-} \Rightarrow \text{-} \Rightarrow 'a :: \{banach,\ second\text{-}countable\text{-}topology\}$
assumes *integrable*[*measurable*]: $\bigwedge i.\ integrable\ M\ (f\ i)$
and *summable*: $AE\ x\ in\ M.\ summable\ (\lambda i.\ norm\ (f\ i\ x))$
and *sums*: $summable\ (\lambda i.\ (\int x.\ norm\ (f\ i\ x)\ \partial M))$
shows *integrable-suminf*: $integrable\ M\ (\lambda x.\ (\sum i.\ f\ i\ x))$ (**is** *integrable M ?S*)
and *sums-integral*: $(\lambda i.\ integral^L\ M\ (f\ i))\ sums\ (\int x.\ (\sum i.\ f\ i\ x)\ \partial M)$ (**is** *?f sums ?x*)
and *integral-suminf*: $(\int x.\ (\sum i.\ f\ i\ x)\ \partial M) = (\sum i.\ integral^L\ M\ (f\ i))$
and *summable-integral*: $summable\ (\lambda i.\ integral^L\ M\ (f\ i))$
⟨*proof*⟩

lemma *integral-norm-bound*:
fixes $f :: \text{-} \Rightarrow 'a :: \{banach,\ second\text{-}countable\text{-}topology\}$
shows $integrable\ M\ f \Longrightarrow norm\ (integral^L\ M\ f) \leq (\int x.\ norm\ (f\ x)\ \partial M)$
⟨*proof*⟩

lemma *integral-eq-nn-integral*:
assumes [*measurable*]: $f \in borel\text{-}measurable\ M$
assumes *nonneg*: $AE\ x\ in\ M.\ 0 \leq f\ x$

shows $integral^L\ M\,f = enn2real\ (\int^+ x.\ ennreal\ (f\,x)\ \partial M)$
⟨*proof*⟩

lemma *enn2real-nn-integral-eq-integral*:
assumes *eq*: $AE\ x\ in\ M.\ f\,x = ennreal\ (g\ x)$ **and** *nn*: $AE\ x\ in\ M.\ 0 \le g\ x$
and *fin*: $(\int^+ x.\ f\,x\ \partial M) < top$
and [*measurable*]: $g \in M \rightarrow_M borel$
shows $enn2real\ (\int^+ x.\ f\,x\ \partial M) = (\int x.\ g\ x\ \partial M)$
⟨*proof*⟩

lemma *has-bochner-integral-nn-integral*:
assumes $f \in borel\text{-}measurable\ M\ AE\ x\ in\ M.\ 0 \le f\,x\ 0 \le x$
assumes $(\int^+ x.\ f\,x\ \partial M) = ennreal\ x$
shows *has-bochner-integral M f x*
⟨*proof*⟩

lemma *integrableI-simple-bochner-integrable*:
fixes $f :: 'a \Rightarrow 'b::\{banach,\ second\text{-}countable\text{-}topology\}$
shows *simple-bochner-integrable M f* $\Longrightarrow$ *integrable M f*
⟨*proof*⟩

lemma *integrable-induct*[*consumes 1*, *case-names base add lim*, *induct pred*: *integrable*]:
fixes $f :: 'a \Rightarrow 'b::\{banach,\ second\text{-}countable\text{-}topology\}$
assumes *integrable M f*
assumes *base*: $\bigwedge A\ c.\ A \in sets\ M \Longrightarrow emeasure\ M\ A < \infty \Longrightarrow P\ (\lambda x.\ indicator\ A\ x *_R c)$
assumes *add*: $\bigwedge f\ g.\ integrable\ M\ f \Longrightarrow P\ f \Longrightarrow integrable\ M\ g \Longrightarrow P\ g \Longrightarrow P\ (\lambda x.\ f\,x + g\ x)$
assumes *lim*: $\bigwedge f\ s.\ (\bigwedge i.\ integrable\ M\ (s\ i)) \Longrightarrow (\bigwedge i.\ P\ (s\ i)) \Longrightarrow$
$(\bigwedge x.\ x \in space\ M \Longrightarrow (\lambda i.\ s\ i\ x) \longlonglongrightarrow f\,x) \Longrightarrow$
$(\bigwedge i\ x.\ x \in space\ M \Longrightarrow norm\ (s\ i\ x) \le 2 * norm\ (f\,x)) \Longrightarrow integrable\ M\ f \Longrightarrow P\ f$
shows $P\ f$
⟨*proof*⟩

lemma *integral-eq-zero-AE*:
$(AE\ x\ in\ M.\ f\,x = 0) \Longrightarrow integral^L\ M\ f = 0$
⟨*proof*⟩

lemma *integral-nonneg-eq-0-iff-AE*:
fixes $f :: \text{-} \Rightarrow real$
assumes *f*[*measurable*]: *integrable M f* **and** *nonneg*: $AE\ x\ in\ M.\ 0 \le f\,x$
shows $integral^L\ M\ f = 0 \longleftrightarrow (AE\ x\ in\ M.\ f\,x = 0)$
⟨*proof*⟩

lemma *integral-mono-AE*:
fixes $f :: 'a \Rightarrow real$
assumes $integrable\ M\ f\ integrable\ M\ g\ AE\ x\ in\ M.\ f\,x \le g\ x$

shows *integral*^{L} *M f* ≤ *integral*^{L} *M g*
⟨*proof*⟩

lemma *integral-mono*:
fixes *f* :: *'a* ⇒ *real*
shows *integrable M f* ⟹ *integrable M g* ⟹ (⋀*x*. *x* ∈ *space M* ⟹ *f x* ≤ *g x*) ⟹
integral^L *M f* ≤ *integral*^{L} *M g*
⟨*proof*⟩

lemma (**in** *finite-measure*) *integrable-measure*:
assumes *I*: *disjoint-family-on X I countable I*
shows *integrable* (*count-space I*) (λ*i*. *measure M* (*X i*))
⟨*proof*⟩

lemma *integrableI-real-bounded*:
assumes *f*: *f* ∈ *borel-measurable M* **and** *ae*: *AE x in M*. *0* ≤ *f x* **and** *fin*: *integral*^{N} *M f* < ∞
shows *integrable M f*
⟨*proof*⟩

lemma *integral-real-bounded*:
assumes *0* ≤ *r integral*^{N} *M f* ≤ *ennreal r*
shows *integral*^{L} *M f* ≤ *r*
⟨*proof*⟩

8.1 Restricted measure spaces

lemma *integrable-restrict-space*:
fixes *f* :: *'a* ⇒ *'b*::{*banach*, *second-countable-topology*}
assumes Ω[*simp*]: Ω ∩ *space M* ∈ *sets M*
shows *integrable* (*restrict-space M* Ω) *f* ⟷ *integrable M* (λ*x*. *indicator* Ω *x* $*_R$ *f x*)
⟨*proof*⟩

lemma *integral-restrict-space*:
fixes *f* :: *'a* ⇒ *'b*::{*banach*, *second-countable-topology*}
assumes Ω[*simp*]: Ω ∩ *space M* ∈ *sets M*
shows *integral*^{L} (*restrict-space M* Ω) *f* = *integral*^{L} *M* (λ*x*. *indicator* Ω *x* $*_R$ *f x*)
⟨*proof*⟩

lemma *integral-empty*:
assumes *space M* = {}
shows *integral*^{L} *M f* = *0*
⟨*proof*⟩

8.2 Measure spaces with an associated density

lemma *integrable-density*:

fixes f :: $'a \Rightarrow 'b$::{*banach*, *second-countable-topology*} **and** g :: $'a \Rightarrow$ *real*
assumes [*measurable*]: $f \in$ *borel-measurable* M $g \in$ *borel-measurable* M
and *nn*: *AE* x *in* M. $0 \leq g\ x$
shows *integrable* (*density* M g) $f \longleftrightarrow$ *integrable* M ($\lambda x.\ g\ x *_R f\ x$)
⟨*proof*⟩

lemma *integral-density*:
fixes f :: $'a \Rightarrow 'b$::{*banach*, *second-countable-topology*} **and** g :: $'a \Rightarrow$ *real*
assumes f: $f \in$ *borel-measurable* M
and g[*measurable*]: $g \in$ *borel-measurable* M *AE* x *in* M. $0 \leq g\ x$
shows $integral^L$ (*density* M g) f = $integral^L$ M ($\lambda x.\ g\ x *_R f\ x$)
⟨*proof*⟩

lemma
fixes g :: $'a \Rightarrow$ *real*
assumes $f \in$ *borel-measurable* M *AE* x *in* M. $0 \leq f\ x$ $g \in$ *borel-measurable* M
shows *integral-real-density*: $integral^L$ (*density* M f) g = ($\int$ x. $f\ x * g\ x$ ∂M)
and *integrable-real-density*: *integrable* (*density* M f) $g \longleftrightarrow$ *integrable* M ($\lambda x.\ f\ x * g\ x$)
⟨*proof*⟩

lemma *has-bochner-integral-density*:
fixes f :: $'a \Rightarrow 'b$::{*banach*, *second-countable-topology*} **and** g :: $'a \Rightarrow$ *real*
shows $f \in$ *borel-measurable* M $\Longrightarrow$ $g \in$ *borel-measurable* M $\Longrightarrow$ (*AE* x *in* M. $0 \leq g\ x$) $\Longrightarrow$
has-bochner-integral M ($\lambda x.\ g\ x *_R f\ x$) x $\Longrightarrow$ *has-bochner-integral* (*density* M g) f x
⟨*proof*⟩

8.3 Distributions

lemma *integrable-distr-eq*:
fixes f :: $'a \Rightarrow 'b$::{*banach*, *second-countable-topology*}
assumes [*measurable*]: $g \in$ *measurable* M N $f \in$ *borel-measurable* N
shows *integrable* (*distr* M N g) $f \longleftrightarrow$ *integrable* M ($\lambda x.\ f\ (g\ x)$)
⟨*proof*⟩

lemma *integrable-distr*:
fixes f :: $'a \Rightarrow 'b$::{*banach*, *second-countable-topology*}
shows $T \in$ *measurable* M M' $\Longrightarrow$ *integrable* (*distr* M M' T) f $\Longrightarrow$ *integrable* M ($\lambda x.\ f\ (T\ x)$)
⟨*proof*⟩

lemma *integral-distr*:
fixes f :: $'a \Rightarrow 'b$::{*banach*, *second-countable-topology*}
assumes g[*measurable*]: $g \in$ *measurable* M N **and** f: $f \in$ *borel-measurable* N
shows $integral^L$ (*distr* M N g) f = $integral^L$ M ($\lambda x.\ f\ (g\ x)$)
⟨*proof*⟩

lemma *has-bochner-integral-distr*:
fixes f :: $'a \Rightarrow$ $'b$::{*banach*, *second-countable-topology*}
shows $f \in$ *borel-measurable* $N \Longrightarrow g \in$ *measurable* M $N \Longrightarrow$
has-bochner-integral M $(\lambda x.\ f\ (g\ x))\ x \Longrightarrow$ *has-bochner-integral* (*distr* M N g) $f\ x$
⟨*proof*⟩

8.4 Lebesgue integration on *count-space*

lemma *integrable-count-space*:
fixes f :: $'a \Rightarrow$ $'b$::{*banach*,*second-countable-topology*}
shows *finite* $X \Longrightarrow$ *integrable* (*count-space* X) f
⟨*proof*⟩

lemma *measure-count-space*[*simp*]:
$B \subseteq A \Longrightarrow$ *finite* $B \Longrightarrow$ *measure* (*count-space* A) B = *card* B
⟨*proof*⟩

lemma *lebesgue-integral-count-space-finite-support*:
assumes f: *finite* $\{a{\in}A.\ f\ a \neq 0\}$
shows $(\int x.\ f\ x\ \partial$*count-space* $A) = (\sum a \mid a \in A \wedge f\ a \neq 0.\ f\ a)$
⟨*proof*⟩

lemma *lebesgue-integral-count-space-finite*: *finite* $A \Longrightarrow (\int x.\ f\ x\ \partial$*count-space* $A)$ $= (\sum a{\in}A.\ f\ a)$
⟨*proof*⟩

lemma *integrable-count-space-nat-iff*:
fixes f :: *nat* $\Rightarrow$ -::{*banach*,*second-countable-topology*}
shows *integrable* (*count-space UNIV*) $f \longleftrightarrow$ *summable* $(\lambda x.\ norm\ (f\ x))$
⟨*proof*⟩

lemma *sums-integral-count-space-nat*:
fixes f :: *nat* $\Rightarrow$ -::{*banach*,*second-countable-topology*}
assumes $*$: *integrable* (*count-space UNIV*) f
shows f *sums* (*integral*^{L} (*count-space UNIV*) f)
⟨*proof*⟩

lemma *integral-count-space-nat*:
fixes f :: *nat* $\Rightarrow$ -::{*banach*,*second-countable-topology*}
shows *integrable* (*count-space UNIV*) $f \Longrightarrow$ *integral*^{L} (*count-space UNIV*) f = $(\sum x.\ f\ x)$
⟨*proof*⟩

8.5 Point measure

lemma *lebesgue-integral-point-measure-finite*:
fixes g :: $'a \Rightarrow$ $'b$::{*banach*, *second-countable-topology*}
shows *finite* $A \Longrightarrow (\bigwedge a.\ a \in A \Longrightarrow 0 \leq f\ a) \Longrightarrow$
integral^L (*point-measure* A f) $g = (\sum a{\in}A.\ f\ a *_R g\ a)$

⟨*proof*⟩

lemma *integrable-point-measure-finite*:
fixes $g :: 'a \Rightarrow 'b::\{banach, second\text{-}countable\text{-}topology\}$ **and** $f :: 'a \Rightarrow real$
shows $finite\ A \Longrightarrow integrable\ (point\text{-}measure\ A\ f)\ g$
⟨*proof*⟩

8.6 Lebesgue integration on *null-measure*

lemma *has-bochner-integral-null-measure-iff* [*iff*]:
$has\text{-}bochner\text{-}integral\ (null\text{-}measure\ M)\ f\ 0 \longleftrightarrow f \in borel\text{-}measurable\ M$
⟨*proof*⟩

lemma *integrable-null-measure-iff* [*iff*]: $integrable\ (null\text{-}measure\ M)\ f \longleftrightarrow f \in borel\text{-}measurable\ M$
⟨*proof*⟩

lemma *integral-null-measure*[*simp*]: $integral^L\ (null\text{-}measure\ M)\ f = 0$
⟨*proof*⟩

8.7 Legacy lemmas for the real-valued Lebesgue integral

lemma *real-lebesgue-integral-def*:
assumes $f[measurable]$: $integrable\ M\ f$
shows $integral^L\ M\ f = enn2real\ (\int^+ x.\ f\ x\ \partial M) - enn2real\ (\int^+ x.\ ennreal\ (-\ f\ x)\ \partial M)$
⟨*proof*⟩

lemma *real-integrable-def*:
$integrable\ M\ f \longleftrightarrow f \in borel\text{-}measurable\ M\ \wedge$
$(\int^+ x.\ ennreal\ (f\ x)\ \partial M) \neq \infty \wedge (\int^+ x.\ ennreal\ (-\ f\ x)\ \partial M) \neq \infty$
⟨*proof*⟩

lemma *integrableD*[*dest*]:
assumes $integrable\ M\ f$
shows $f \in borel\text{-}measurable\ M\ (\int^+ x.\ ennreal\ (f\ x)\ \partial M) \neq \infty\ (\int^+ x.\ ennreal\ (-\ f\ x)\ \partial M) \neq \infty$
⟨*proof*⟩

lemma *integrableE*:
assumes $integrable\ M\ f$
obtains $r\ q$ **where**
$(\int^+ x.\ ennreal\ (f\ x)\partial M) = ennreal\ r$
$(\int^+ x.\ ennreal\ (-f\ x)\partial M) = ennreal\ q$
$f \in borel\text{-}measurable\ M\ integral^L\ M\ f = r - q$
⟨*proof*⟩

lemma *integral-monotone-convergence-nonneg*:
fixes $f :: nat \Rightarrow 'a \Rightarrow real$
assumes i: $\bigwedge i.\ integrable\ M\ (f\ i)$ **and** $mono$: $AE\ x\ in\ M.\ mono\ (\lambda n.\ f\ n\ x)$

and *pos*: $\bigwedge i.\ AE\ x\ in\ M.\ 0 \leq f\ i\ x$
and *lim*: $AE\ x\ in\ M.\ (\lambda i.\ f\ i\ x) \longrightarrow u\ x$
and *ilim*: $(\lambda i.\ integral^L\ M\ (f\ i)) \longrightarrow x$
and *u*: $u \in borel\text{-}measurable\ M$
shows *integrable M u*
and $integral^L\ M\ u = x$
⟨*proof*⟩

lemma
fixes $f :: nat \Rightarrow 'a \Rightarrow real$
assumes *f*: $\bigwedge i.\ integrable\ M\ (f\ i)$ **and** *mono*: $AE\ x\ in\ M.\ mono\ (\lambda n.\ f\ n\ x)$
and *lim*: $AE\ x\ in\ M.\ (\lambda i.\ f\ i\ x) \longrightarrow u\ x$
and *ilim*: $(\lambda i.\ integral^L\ M\ (f\ i)) \longrightarrow x$
and *u*: $u \in borel\text{-}measurable\ M$
shows *integrable-monotone-convergence*: *integrable M u*
and *integral-monotone-convergence*: $integral^L\ M\ u = x$
and *has-bochner-integral-monotone-convergence*: *has-bochner-integral M u x*
⟨*proof*⟩

lemma *integral-norm-eq-0-iff*:
fixes $f :: 'a \Rightarrow 'b::\{banach,\ second\text{-}countable\text{-}topology\}$
assumes *f*[*measurable*]: *integrable M f*
shows $(\int x.\ norm\ (f\ x)\ \partial M) = 0 \longleftrightarrow emeasure\ M\ \{x \in space\ M.\ f\ x \neq 0\} = 0$
⟨*proof*⟩

lemma *integral-0-iff*:
fixes $f :: 'a \Rightarrow real$
shows $integrable\ M\ f \Longrightarrow (\int x.\ |f\ x|\ \partial M) = 0 \longleftrightarrow emeasure\ M\ \{x \in space\ M.\ f\ x \neq 0\} = 0$
⟨*proof*⟩

lemma (**in** *finite-measure*) *integrable-const*[*intro*!, *simp*]: $integrable\ M\ (\lambda x.\ a)$
⟨*proof*⟩

lemma *lebesgue-integral-const*[*simp*]:
fixes $a :: 'a :: \{banach,\ second\text{-}countable\text{-}topology\}$
shows $(\int x.\ a\ \partial M) = measure\ M\ (space\ M) *_R a$
⟨*proof*⟩

lemma (**in** *finite-measure*) *integrable-const-bound*:
fixes $f :: 'a \Rightarrow 'b::\{banach,second\text{-}countable\text{-}topology\}$
shows $AE\ x\ in\ M.\ norm\ (f\ x) \leq B \Longrightarrow f \in borel\text{-}measurable\ M \Longrightarrow integrable\ M\ f$
⟨*proof*⟩

lemma *integral-indicator-finite-real*:
fixes $f :: 'a \Rightarrow real$
assumes [*simp*]: *finite A*
assumes [*measurable*]: $\bigwedge a.\ a \in A \Longrightarrow \{a\} \in sets\ M$

assumes *finite*: $\bigwedge a.\ a \in A \Longrightarrow emeasure\ M\ \{a\} < \infty$
shows $(\int x.\ f\,x * indicator\ A\ x\ \partial M) = (\sum a{\in}A.\ f\ a * measure\ M\ \{a\})$
⟨*proof*⟩

lemma (**in** *finite-measure*) *ennreal-integral-real*:
assumes [*measurable*]: $f \in borel\text{-}measurable\ M$
assumes *ae*: $AE\ x\ in\ M.\ f\,x \leq ennreal\ B\ 0 \leq B$
shows $ennreal\ (\int x.\ enn2real\ (f\,x)\ \partial M) = (\int^+ x.\ f\,x\ \partial M)$
⟨*proof*⟩

lemma (**in** *finite-measure*) *integral-less-AE*:
fixes $X\ Y :: 'a \Rightarrow real$
assumes *int*: $integrable\ M\ X\ integrable\ M\ Y$
assumes *A*: $(emeasure\ M)\ A \neq 0\ A \in sets\ M\ AE\ x\ in\ M.\ x \in A \longrightarrow X\,x \neq Y\,x$
assumes *gt*: $AE\ x\ in\ M.\ X\,x \leq Y\,x$
shows $integral^L\ M\ X < integral^L\ M\ Y$
⟨*proof*⟩

lemma (**in** *finite-measure*) *integral-less-AE-space*:
fixes $X\ Y :: 'a \Rightarrow real$
assumes *int*: $integrable\ M\ X\ integrable\ M\ Y$
assumes *gt*: $AE\ x\ in\ M.\ X\,x < Y\,x\ emeasure\ M\ (space\ M) \neq 0$
shows $integral^L\ M\ X < integral^L\ M\ Y$
⟨*proof*⟩

lemma *tendsto-integral-at-top*:
fixes $f :: real \Rightarrow 'a::\{banach,\ second\text{-}countable\text{-}topology\}$
assumes [*measurable-cong*]: $sets\ M = sets\ borel$ **and** f[*measurable*]: $integrable\ M\ f$
shows $((\lambda y.\ \int\ x.\ indicator\ \{..\ y\}\ x *_R f\,x\ \partial M) \longrightarrow \int\ x.\ f\,x\ \partial M)\ at\text{-}top$
⟨*proof*⟩

lemma
fixes $f :: real \Rightarrow real$
assumes *M*: $sets\ M = sets\ borel$
assumes *nonneg*: $AE\ x\ in\ M.\ 0 \leq f\,x$
assumes *borel*: $f \in borel\text{-}measurable\ borel$
assumes *int*: $\bigwedge y.\ integrable\ M\ (\lambda x.\ f\,x * indicator\ \{..\ y\}\ x)$
assumes *conv*: $((\lambda y.\ \int\ x.\ f\,x * indicator\ \{..\ y\}\ x\ \partial M) \longrightarrow x)\ at\text{-}top$
shows *has-bochner-integral-monotone-convergence-at-top*: $has\text{-}bochner\text{-}integral\ M\ f\ x$
and *integrable-monotone-convergence-at-top*: $integrable\ M\ f$
and *integral-monotone-convergence-at-top*:$integral^L\ M\ f = x$
⟨*proof*⟩

8.8 Product measure

lemma (**in** *sigma-finite-measure*) *borel-measurable-lebesgue-integrable*[*measurable (raw)*]:

fixes f :: - ⇒ - ⇒ -::{*banach*, *second-countable-topology*}
assumes [*measurable*]: *case-prod f* ∈ *borel-measurable* ($N \bigotimes_M M$)
shows *Measurable.pred N* (λ*x*. *integrable M* (*f x*))
⟨*proof*⟩

lemma *Collect-subset* [*simp*]: {*x*∈*A*. *P x*} ⊆ *A* ⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *measurable-measure*[*measurable* (*raw*)]:
(⋀*x*. *x* ∈ *space N* ⟹ *A x* ⊆ *space M*) ⟹
{*x* ∈ *space* ($N \bigotimes_M M$). *snd x* ∈ *A* (*fst x*)} ∈ *sets* ($N \bigotimes_M M$) ⟹
(λ*x*. *measure M* (*A x*)) ∈ *borel-measurable N*
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *borel-measurable-lebesgue-integral*[*measurable* (*raw*)]:
fixes f :: - ⇒ - ⇒ -::{*banach*, *second-countable-topology*}
assumes *f*[*measurable*]: *case-prod f* ∈ *borel-measurable* ($N \bigotimes_M M$)
shows (λ*x*. ∫ *y*. *f x y* ∂*M*) ∈ *borel-measurable N*
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *integrable-product-swap*:
fixes f :: - ⇒ -::{*banach*, *second-countable-topology*}
assumes *integrable* ($M1 \bigotimes_M M2$) *f*
shows *integrable* ($M2 \bigotimes_M M1$) (λ(*x*,*y*). *f* (*y*,*x*))
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *integrable-product-swap-iff*:
fixes f :: - ⇒ -::{*banach*, *second-countable-topology*}
shows *integrable* ($M2 \bigotimes_M M1$) (λ(*x*,*y*). *f* (*y*,*x*)) ⟷ *integrable* ($M1 \bigotimes_M M2$)
f
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *integral-product-swap*:
fixes f :: - ⇒ -::{*banach*, *second-countable-topology*}
assumes *f*: *f* ∈ *borel-measurable* ($M1 \bigotimes_M M2$)
shows (∫ (*x*,*y*). *f* (*y*,*x*) ∂($M2 \bigotimes_M M1$)) = $integral^L$ ($M1 \bigotimes_M M2$) *f*
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *Fubini-integrable*:
fixes f :: - ⇒ -::{*banach*, *second-countable-topology*}
assumes *f*[*measurable*]: *f* ∈ *borel-measurable* ($M1 \bigotimes_M M2$)
and *integ1*: *integrable M1* (λ*x*. ∫ *y*. *norm* (*f* (*x*, *y*)) ∂*M2*)
and *integ2*: *AE x in M1*. *integrable M2* (λ*y*. *f* (*x*, *y*))
shows *integrable* ($M1 \bigotimes_M M2$) *f*
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *emeasure-pair-measure-finite*:
assumes *A*: *A* ∈ *sets* ($M1 \bigotimes_M M2$) **and** *finite*: *emeasure* ($M1 \bigotimes_M M2$) *A* <
∞
shows *AE x in M1*. *emeasure M2* {*y*∈*space M2*. (*x*, *y*) ∈ *A*} < ∞

⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *AE-integrable-fst'*:
  fixes f :: - ⇒ -::{*banach*, *second-countable-topology*}
  assumes f[*measurable*]: *integrable* (*M1* $\bigotimes_M$ *M2*) f
  shows *AE x in M1*. *integrable M2* (λy. f (x, y))
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *integrable-fst'*:
  fixes f :: - ⇒ -::{*banach*, *second-countable-topology*}
  assumes f[*measurable*]: *integrable* (*M1* $\bigotimes_M$ *M2*) f
  shows *integrable M1* (λx. $\int y$. f (x, y) ∂*M2*)
  ⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *integral-fst'*:
  fixes f :: - ⇒ -::{*banach*, *second-countable-topology*}
  assumes f: *integrable* (*M1* $\bigotimes_M$ *M2*) f
  shows ($\int x$. ($\int y$. f (x, y) ∂*M2*) ∂*M1*) = *integral*^{L} (*M1* $\bigotimes_M$ *M2*) f
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*)
  fixes f :: - ⇒ - ⇒ -::{*banach*, *second-countable-topology*}
  assumes f: *integrable* (*M1* $\bigotimes_M$ *M2*) (*case-prod* f)
  shows *AE-integrable-fst*: *AE x in M1*. *integrable M2* (λy. f x y) (**is** *?AE*)
    and *integrable-fst*: *integrable M1* (λx. $\int y$. f x y ∂*M2*) (**is** *?INT*)
    and *integral-fst*: ($\int x$. ($\int y$. f x y ∂*M2*) ∂*M1*) = *integral*^{L} (*M1* $\bigotimes_M$ *M2*) (λ(x, y). f x y) (**is** *?EQ*)
  ⟨*proof*⟩

lemma (**in** *pair-sigma-finite*)
  fixes f :: - ⇒ - ⇒ -::{*banach*, *second-countable-topology*}
  assumes f[*measurable*]: *integrable* (*M1* $\bigotimes_M$ *M2*) (*case-prod* f)
  shows *AE-integrable-snd*: *AE y in M2*. *integrable M1* (λx. f x y) (**is** *?AE*)
    and *integrable-snd*: *integrable M2* (λy. $\int x$. f x y ∂*M1*) (**is** *?INT*)
    and *integral-snd*: ($\int y$. ($\int x$. f x y ∂*M1*) ∂*M2*) = *integral*^{L} (*M1* $\bigotimes_M$ *M2*) (*case-prod* f) (**is** *?EQ*)
⟨*proof*⟩

lemma (**in** *pair-sigma-finite*) *Fubini-integral*:
  fixes f :: - ⇒ - ⇒ - :: {*banach*, *second-countable-topology*}
  assumes f: *integrable* (*M1* $\bigotimes_M$ *M2*) (*case-prod* f)
  shows ($\int y$. ($\int x$. f x y ∂*M1*) ∂*M2*) = ($\int x$. ($\int y$. f x y ∂*M2*) ∂*M1*)
  ⟨*proof*⟩

lemma (**in** *product-sigma-finite*) *product-integral-singleton*:
  fixes f :: - ⇒ -::{*banach*, *second-countable-topology*}
  shows f ∈ *borel-measurable* (*M i*) ⟹ ($\int x$. f (x i) ∂Pi_M {i} *M*) = *integral*^{L} (*M i*) f
  ⟨*proof*⟩

lemma (**in** *product-sigma-finite*) *product-integral-fold*:
  fixes f :: - $\Rightarrow$ -::{*banach*, *second-countable-topology*}
  assumes *IJ*[*simp*]: $I \cap J = \{\}$ **and** *fin*: *finite I finite J*
  and *f*: *integrable* (Pi_M ($I \cup J$) M) f
  shows $integral^L$ (Pi_M ($I \cup J$) M) f = ($\int x.$ ($\int y.$ f (*merge I J* (x, y)) ∂Pi_M J M) ∂Pi_M I M)
⟨*proof*⟩

lemma (**in** *product-sigma-finite*) *product-integral-insert*:
  fixes f :: - $\Rightarrow$ -::{*banach*, *second-countable-topology*}
  assumes *I*: *finite I* $i \notin I$
    and *f*: *integrable* (Pi_M (*insert i I*) M) f
  shows $integral^L$ (Pi_M (*insert i I*) M) f = ($\int x.$ ($\int y.$ f ($x(i:=y)$) ∂M i) ∂Pi_M I M)
⟨*proof*⟩

lemma (**in** *product-sigma-finite*) *product-integrable-setprod*:
  fixes f :: $'i \Rightarrow 'a \Rightarrow$ -::{*real-normed-field*,*banach*,*second-countable-topology*}
  assumes [*simp*]: *finite I* **and** *integrable*: $\bigwedge i.\ i \in I \Longrightarrow$ *integrable* (M i) (f i)
  shows *integrable* (Pi_M I M) ($\lambda x.$ ($\prod i{\in}I.$ f i (x i))) (**is** *integrable* - *?f*)
⟨*proof*⟩

lemma (**in** *product-sigma-finite*) *product-integral-setprod*:
  fixes f :: $'i \Rightarrow 'a \Rightarrow$ -::{*real-normed-field*,*banach*,*second-countable-topology*}
  assumes *finite I* **and** *integrable*: $\bigwedge i.\ i \in I \Longrightarrow$ *integrable* (M i) (f i)
  shows ($\int x.$ ($\prod i{\in}I.$ f i (x i)) ∂Pi_M I M) = ($\prod i{\in}I.$ $integral^L$ (M i) (f i))
⟨*proof*⟩

lemma *integrable-subalgebra*:
  fixes f :: $'a \Rightarrow 'b$::{*banach*, *second-countable-topology*}
  assumes *borel*: $f \in$ *borel-measurable N*
  and *N*: *sets N* $\subseteq$ *sets M space N* = *space M* $\bigwedge A.\ A \in$ *sets N* $\Longrightarrow$ *emeasure N A* = *emeasure M A*
  shows *integrable N f* $\longleftrightarrow$ *integrable M f* (**is** *?P*)
⟨*proof*⟩

lemma *integral-subalgebra*:
  fixes f :: $'a \Rightarrow 'b$::{*banach*, *second-countable-topology*}
  assumes *borel*: $f \in$ *borel-measurable N*
  and *N*: *sets N* $\subseteq$ *sets M space N* = *space M* $\bigwedge A.\ A \in$ *sets N* $\Longrightarrow$ *emeasure N A* = *emeasure M A*
  shows $integral^L$ N f = $integral^L$ M f
⟨*proof*⟩

hide-const (**open**) *simple-bochner-integral*
hide-const (**open**) *simple-bochner-integrable*

end

9 Caratheodory Extension Theorem

theory *Caratheodory*
imports *Measure-Space*
begin

Originally from the Hurd/Coble measure theory development, translated by Lawrence Paulson.

lemma *suminf-ennreal-2dimen*:
fixes f:: *nat* × *nat* ⇒ *ennreal*
assumes $\bigwedge m.\ g\ m = (\sum n.\ f\ (m,n))$
shows $(\sum i.\ f\ (\textit{prod-decode}\ i)) = \textit{suminf}\ g$
⟨*proof*⟩

9.1 Characterizations of Measures

definition *outer-measure-space* **where**
outer-measure-space M f ⟷ *positive M f* ∧ *increasing M f* ∧ *countably-subadditive M f*

9.1.1 Lambda Systems

definition *lambda-system* :: *'a set* ⇒ *'a set set* ⇒ (*'a set* ⇒ *ennreal*) ⇒ *'a set set* **where**
lambda-system Ω *M f* = $\{l \in M.\ \forall x \in M.\ f\ (l \cap x) + f\ ((\Omega - l) \cap x) = f\ x\}$

lemma (**in** *algebra*) *lambda-system-eq*:
lambda-system Ω *M f* = $\{l \in M.\ \forall x \in M.\ f\ (x \cap l) + f\ (x - l) = f\ x\}$
⟨*proof*⟩

lemma (**in** *algebra*) *lambda-system-empty*: *positive M f* ⟹ {} ∈ *lambda-system* Ω *M f*
⟨*proof*⟩

lemma *lambda-system-sets*: $x \in$ *lambda-system* Ω *M f* ⟹ $x \in M$
⟨*proof*⟩

lemma (**in** *algebra*) *lambda-system-Compl*:
fixes f:: *'a set* ⇒ *ennreal*
assumes x: $x \in$ *lambda-system* Ω *M f*
shows $\Omega - x \in$ *lambda-system* Ω *M f*
⟨*proof*⟩

lemma (**in** *algebra*) *lambda-system-Int*:
fixes f:: *'a set* ⇒ *ennreal*
assumes *xl*: $x \in$ *lambda-system* Ω *M f* **and** *yl*: $y \in$ *lambda-system* Ω *M f*
shows $x \cap y \in$ *lambda-system* Ω *M f*
⟨*proof*⟩

lemma (**in** *algebra*) *lambda-system-Un*:

fixes f:: $'a\ set \Rightarrow ennreal$
assumes *xl*: $x \in$ *lambda-system* Ω $M f$ **and** *yl*: $y \in$ *lambda-system* Ω $M f$
shows $x \cup y \in$ *lambda-system* Ω $M f$
⟨*proof*⟩

lemma (**in** *algebra*) *lambda-system-algebra*:
positive $M f \Longrightarrow$ *algebra* Ω (*lambda-system* Ω $M f$)
⟨*proof*⟩

lemma (**in** *algebra*) *lambda-system-strong-additive*:
assumes *z*: $z \in M$ **and** *disj*: $x \cap y = \{\}$
and *xl*: $x \in$ *lambda-system* Ω $M f$ **and** *yl*: $y \in$ *lambda-system* Ω $M f$
shows $f\ (z \cap (x \cup y)) = f\ (z \cap x) + f\ (z \cap y)$
⟨*proof*⟩

lemma (**in** *algebra*) *lambda-system-additive*: *additive* (*lambda-system* Ω $M f$) f
⟨*proof*⟩

lemma *lambda-system-increasing*: *increasing* $M f \Longrightarrow$ *increasing* (*lambda-system* Ω $M f$) f
⟨*proof*⟩

lemma *lambda-system-positive*: *positive* $M f \Longrightarrow$ *positive* (*lambda-system* Ω $M f$) f
⟨*proof*⟩

lemma (**in** *algebra*) *lambda-system-strong-sum*:
fixes A:: $nat \Rightarrow 'a\ set$ **and** f :: $'a\ set \Rightarrow ennreal$
assumes f: *positive* $M f$ **and** *a*: $a \in M$
and *A*: *range* $A \subseteq$ *lambda-system* Ω $M f$
and *disj*: *disjoint-family* A
shows $(\sum i = 0..<n.\ f\ (a \cap A\ i)) = f\ (a \cap (\bigcup i\in\{0..<n\}.\ A\ i))$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *lambda-system-caratheodory*:
assumes *oms*: *outer-measure-space* $M f$
and *A*: *range* $A \subseteq$ *lambda-system* Ω $M f$
and *disj*: *disjoint-family* A
shows $(\bigcup i.\ A\ i) \in$ *lambda-system* Ω $M f \wedge (\sum i.\ f\ (A\ i)) = f\ (\bigcup i.\ A\ i)$
⟨*proof*⟩

lemma (**in** *sigma-algebra*) *caratheodory-lemma*:
assumes *oms*: *outer-measure-space* $M f$
defines $L \equiv$ *lambda-system* Ω $M f$
shows *measure-space* Ω L f
⟨*proof*⟩

definition *outer-measure* :: $'a\ set\ set \Rightarrow ('a\ set \Rightarrow ennreal) \Rightarrow 'a\ set \Rightarrow ennreal$
where

outer-measure M f X =
(*INF A*:{*A. range A* ⊆ *M* ∧ *disjoint-family A* ∧ *X* ⊆ (⋃*i. A i*)}. ∑*i. f* (*A i*))

lemma (**in** *ring-of-sets*) *outer-measure-agrees*:
assumes *posf*: *positive M f* **and** *ca*: *countably-additive M f* **and** *s*: *s* ∈ *M*
shows *outer-measure M f s* = *f s*
⟨*proof*⟩

lemma *outer-measure-empty*:
positive M f ⟹ {} ∈ *M* ⟹ *outer-measure M f* {} = *0*
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *positive-outer-measure*:
assumes *positive M f* **shows** *positive* (*Pow* Ω) (*outer-measure M f*)
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *increasing-outer-measure*: *increasing* (*Pow* Ω) (*outer-measure M f*)
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *outer-measure-le*:
assumes *pos*: *positive M f* **and** *inc*: *increasing M f* **and** *A*: *range A* ⊆ *M* **and** *X*: *X* ⊆ (⋃*i. A i*)
shows *outer-measure M f X* ≤ (∑*i. f* (*A i*))
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *outer-measure-close*:
outer-measure M f X < *e* ⟹ ∃*A. range A* ⊆ *M* ∧ *disjoint-family A* ∧ *X* ⊆ (⋃*i. A i*) ∧ (∑*i. f* (*A i*)) < *e*
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *countably-subadditive-outer-measure*:
assumes *posf*: *positive M f* **and** *inc*: *increasing M f*
shows *countably-subadditive* (*Pow* Ω) (*outer-measure M f*)
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *outer-measure-space-outer-measure*:
positive M f ⟹ *increasing M f* ⟹ *outer-measure-space* (*Pow* Ω) (*outer-measure M f*)
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *algebra-subset-lambda-system*:
assumes *posf*: *positive M f* **and** *inc*: *increasing M f*
and *add*: *additive M f*
shows *M* ⊆ *lambda-system* Ω (*Pow* Ω) (*outer-measure M f*)
⟨*proof*⟩

lemma *measure-down*: *measure-space* Ω *N* μ ⟹ *sigma-algebra* Ω *M* ⟹ *M* ⊆ *N*

$\Longrightarrow$ *measure-space* Ω *M* μ
⟨*proof*⟩

9.2 Caratheodory's theorem

theorem (**in** *ring-of-sets*) *caratheodory'*:
assumes *posf*: *positive M f* **and** *ca*: *countably-additive M f*
shows $\exists\, \mu$ :: *'a set* $\Rightarrow$ *ennreal*. ($\forall$ *s* $\in$ *M*. μ *s* = *f s*) $\wedge$ *measure-space* Ω (*sigma-sets* Ω *M*) μ
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *caratheodory-empty-continuous*:
assumes *f*: *positive M f additive M f* **and** *fin*: $\bigwedge$*A*. *A* $\in$ *M* $\Longrightarrow$ *f A* $\neq \infty$
assumes *cont*: $\bigwedge$*A*. *range A* $\subseteq$ *M* $\Longrightarrow$ *decseq A* $\Longrightarrow$ ($\bigcap$ *i*. *A i*) = {} $\Longrightarrow$ (λ*i*. *f* (*A i*)) $\longrightarrow$ *0*
shows $\exists\, \mu$ :: *'a set* $\Rightarrow$ *ennreal*. ($\forall$ *s* $\in$ *M*. μ *s* = *f s*) $\wedge$ *measure-space* Ω (*sigma-sets* Ω *M*) μ
⟨*proof*⟩

9.3 Volumes

definition *volume* :: *'a set set* $\Rightarrow$ (*'a set* $\Rightarrow$ *ennreal*) $\Rightarrow$ *bool* **where**
volume M f $\longleftrightarrow$
(*f* {} = *0*) $\wedge$ ($\forall$ *a*$\in$*M*. *0* $\leq$ *f a*) $\wedge$
($\forall$ *C*$\subseteq$*M*. *disjoint C* $\longrightarrow$ *finite C* $\longrightarrow$ $\bigcup$ *C* $\in$ *M* $\longrightarrow$ *f* ($\bigcup$ *C*) = ($\sum$ *c*$\in$*C*. *f c*))

lemma *volumeI*:
assumes *f* {} = *0*
assumes $\bigwedge$*a*. *a* $\in$ *M* $\Longrightarrow$ *0* $\leq$ *f a*
assumes $\bigwedge$*C*. *C* $\subseteq$ *M* $\Longrightarrow$ *disjoint C* $\Longrightarrow$ *finite C* $\Longrightarrow$ $\bigcup$ *C* $\in$ *M* $\Longrightarrow$ *f* ($\bigcup$ *C*) = ($\sum$ *c*$\in$*C*. *f c*)
shows *volume M f*
⟨*proof*⟩

lemma *volume-positive*:
volume M f $\Longrightarrow$ *a* $\in$ *M* $\Longrightarrow$ *0* $\leq$ *f a*
⟨*proof*⟩

lemma *volume-empty*:
volume M f $\Longrightarrow$ *f* {} = *0*
⟨*proof*⟩

lemma *volume-finite-additive*:
assumes *volume M f*
assumes *A*: $\bigwedge$*i*. *i* $\in$ *I* $\Longrightarrow$ *A i* $\in$ *M disjoint-family-on A I finite I UNION I A* $\in$ *M*
shows *f* (*UNION I A*) = ($\sum$ *i*$\in$*I*. *f* (*A i*))
⟨*proof*⟩

lemma (**in** *ring-of-sets*) *volume-additiveI*:

assumes *pos*: $\bigwedge a.\ a \in M \Longrightarrow 0 \leq \mu\ a$
assumes [*simp*]: $\mu\ \{\} = 0$
assumes *add*: $\bigwedge a\ b.\ a \in M \Longrightarrow b \in M \Longrightarrow a \cap b = \{\} \Longrightarrow \mu\ (a \cup b) = \mu\ a + \mu\ b$
shows *volume* $M\ \mu$
⟨*proof*⟩

lemma (**in** *semiring-of-sets*) *extend-volume*:
assumes *volume* $M\ \mu$
shows $\exists \mu'.$ *volume generated-ring* $\mu' \wedge (\forall a \in M.\ \mu'\ a = \mu\ a)$
⟨*proof*⟩

9.3.1 Caratheodory on semirings

theorem (**in** *semiring-of-sets*) *caratheodory*:
assumes *pos*: *positive* $M\ \mu$ **and** *ca*: *countably-additive* $M\ \mu$
shows $\exists \mu' :: 'a\ set \Rightarrow ennreal.\ (\forall s \in M.\ \mu'\ s = \mu\ s) \wedge$ *measure-space* Ω (*sigma-sets* $\Omega\ M$) μ'
⟨*proof*⟩

lemma *extend-measure-caratheodory*:
fixes $G :: 'i \Rightarrow 'a\ set$
assumes M: $M =$ *extend-measure* $\Omega\ I\ G\ \mu$
assumes $i \in I$
assumes *semiring-of-sets* $\Omega\ (G\ `\ I)$
assumes *empty*: $\bigwedge i.\ i \in I \Longrightarrow G\ i = \{\} \Longrightarrow \mu\ i = 0$
assumes *inj*: $\bigwedge i\ j.\ i \in I \Longrightarrow j \in I \Longrightarrow G\ i = G\ j \Longrightarrow \mu\ i = \mu\ j$
assumes *nonneg*: $\bigwedge i.\ i \in I \Longrightarrow 0 \leq \mu\ i$
assumes *add*: $\bigwedge A{::}nat \Rightarrow 'i.\ \bigwedge j.\ A \in UNIV \rightarrow I \Longrightarrow j \in I \Longrightarrow$ *disjoint-family* $(G \circ A) \Longrightarrow$
$(\bigcup i.\ G\ (A\ i)) = G\ j \Longrightarrow (\sum n.\ \mu\ (A\ n)) = \mu\ j$
shows *emeasure* $M\ (G\ i) = \mu\ i$
⟨*proof*⟩

lemma *extend-measure-caratheodory-pair*:
fixes $G :: 'i \Rightarrow 'j \Rightarrow 'a\ set$
assumes M: $M =$ *extend-measure* $\Omega\ \{(a, b).\ P\ a\ b\}\ (\lambda(a, b).\ G\ a\ b)\ (\lambda(a, b).\ \mu\ a\ b)$
assumes $P\ i\ j$
assumes *semiring*: *semiring-of-sets* $\Omega\ \{G\ a\ b \mid a\ b.\ P\ a\ b\}$
assumes *empty*: $\bigwedge i\ j.\ P\ i\ j \Longrightarrow G\ i\ j = \{\} \Longrightarrow \mu\ i\ j = 0$
assumes *inj*: $\bigwedge i\ j\ k\ l.\ P\ i\ j \Longrightarrow P\ k\ l \Longrightarrow G\ i\ j = G\ k\ l \Longrightarrow \mu\ i\ j = \mu\ k\ l$
assumes *nonneg*: $\bigwedge i\ j.\ P\ i\ j \Longrightarrow 0 \leq \mu\ i\ j$
assumes *add*: $\bigwedge A{::}nat \Rightarrow 'i.\ \bigwedge B{::}nat \Rightarrow 'j.\ \bigwedge j\ k.$
$(\bigwedge n.\ P\ (A\ n)\ (B\ n)) \Longrightarrow P\ j\ k \Longrightarrow$ *disjoint-family* $(\lambda n.\ G\ (A\ n)\ (B\ n)) \Longrightarrow$
$(\bigcup i.\ G\ (A\ i)\ (B\ i)) = G\ j\ k \Longrightarrow (\sum n.\ \mu\ (A\ n)\ (B\ n)) = \mu\ j\ k$
shows *emeasure* $M\ (G\ i\ j) = \mu\ i\ j$
⟨*proof*⟩

end

10 Lebesgue measure

theory *Lebesgue-Measure*
imports *Finite-Product-Measure Bochner-Integration Caratheodory*
begin

10.1 Every right continuous and nondecreasing function gives rise to a measure

definition *interval-measure* :: (*real* ⇒ *real*) ⇒ *real measure* **where**
interval-measure F = *extend-measure UNIV* {(*a*, *b*). *a* ≤ *b*} (λ(*a*, *b*). {*a* <.. *b*}) (λ(*a*, *b*). *ennreal* (*F b* − *F a*))

lemma *emeasure-interval-measure-Ioc*:
assumes *a* ≤ *b*
assumes *mono-F*: ⋀*x y*. *x* ≤ *y* ⟹ *F x* ≤ *F y*
assumes *right-cont-F* : ⋀*a*. *continuous* (*at-right a*) *F*
shows *emeasure* (*interval-measure F*) {*a* <.. *b*} = *F b* − *F a*
⟨*proof*⟩

lemma *measure-interval-measure-Ioc*:
assumes *a* ≤ *b*
assumes *mono-F*: ⋀*x y*. *x* ≤ *y* ⟹ *F x* ≤ *F y*
assumes *right-cont-F* : ⋀*a*. *continuous* (*at-right a*) *F*
shows *measure* (*interval-measure F*) {*a* <.. *b*} = *F b* − *F a*
⟨*proof*⟩

lemma *emeasure-interval-measure-Ioc-eq*:
(⋀*x y*. *x* ≤ *y* ⟹ *F x* ≤ *F y*) ⟹ (⋀*a*. *continuous* (*at-right a*) *F*) ⟹
emeasure (*interval-measure F*) {*a* <.. *b*} = (*if a* ≤ *b then F b* − *F a else 0*)
⟨*proof*⟩

lemma *sets-interval-measure* [*simp*, *measurable-cong*]: *sets* (*interval-measure F*) = *sets borel*
⟨*proof*⟩

lemma *space-interval-measure* [*simp*]: *space* (*interval-measure F*) = *UNIV*
⟨*proof*⟩

lemma *emeasure-interval-measure-Icc*:
assumes *a* ≤ *b*
assumes *mono-F*: ⋀*x y*. *x* ≤ *y* ⟹ *F x* ≤ *F y*
assumes *cont-F* : *continuous-on UNIV F*
shows *emeasure* (*interval-measure F*) {*a* .. *b*} = *F b* − *F a*
⟨*proof*⟩

lemma *sigma-finite-interval-measure*:

assumes *mono-F*: $\bigwedge x\ y.\ x \le y \Longrightarrow F\ x \le F\ y$
assumes *right-cont-F* : $\bigwedge a.$ *continuous* (*at-right* a) F
shows *sigma-finite-measure* (*interval-measure* F)
⟨*proof*⟩

10.2 Lebesgue-Borel measure

definition *lborel* :: (*'a* :: *euclidean-space*) *measure* **where**
lborel = *distr* (Π_M $b \in Basis$. *interval-measure* ($\lambda x.\ x$)) *borel* ($\lambda f.\ \sum b \in Basis.\ f\ b *_R b$)

lemma
shows *sets-lborel*[*simp*, *measurable-cong*]: *sets lborel* = *sets borel*
and *space-lborel*[*simp*]: *space lborel* = *space borel*
and *measurable-lborel1*[*simp*]: *measurable M lborel* = *measurable M borel*
and *measurable-lborel2*[*simp*]: *measurable lborel M* = *measurable borel M*
⟨*proof*⟩

context
begin

interpretation *sigma-finite-measure interval-measure* ($\lambda x.\ x$)
⟨*proof*⟩
interpretation *finite-product-sigma-finite* λ-. *interval-measure* ($\lambda x.\ x$) *Basis*
⟨*proof*⟩

lemma *lborel-eq-real*: *lborel* = *interval-measure* ($\lambda x.\ x$)
⟨*proof*⟩

lemma *lborel-eq*: *lborel* = *distr* (Π_M $b \in Basis$. *lborel*) *borel* ($\lambda f.\ \sum b \in Basis.\ f\ b *_R b$)
⟨*proof*⟩

lemma *nn-integral-lborel-setprod*:
assumes [*measurable*]: $\bigwedge b.\ b \in Basis \Longrightarrow f\ b \in$ *borel-measurable borel*
assumes *nn*[*simp*]: $\bigwedge b\ x.\ b \in Basis \Longrightarrow 0 \le f\ b\ x$
shows ($\int^+ x.$ ($\prod b \in Basis.\ f\ b\ (x \cdot b)$) ∂*lborel*) = ($\prod b \in Basis.$ ($\int^+ x.\ f\ b\ x$ ∂*lborel*))
⟨*proof*⟩

lemma *emeasure-lborel-Icc*[*simp*]:
fixes *l u* :: *real*
assumes [*simp*]: $l \le u$
shows *emeasure lborel* $\{l\ ..\ u\} = u - l$
⟨*proof*⟩

lemma *emeasure-lborel-Icc-eq*: *emeasure lborel* $\{l\ ..\ u\}$ = *ennreal* (*if* $l \le u$ *then* $u - l$ *else* 0)
⟨*proof*⟩

lemma *emeasure-lborel-cbox*[*simp*]:
  assumes [*simp*]: $\bigwedge b.\ b \in Basis \Longrightarrow l \cdot b \leq u \cdot b$
  shows $emeasure\ lborel\ (cbox\ l\ u) = (\prod b{\in}Basis.\ (u - l) \cdot b)$
⟨*proof*⟩

lemma *AE-lborel-singleton*: $AE\ x\ in\ lborel{::}'a{::}euclidean\text{-}space\ measure.\ x \neq c$
  ⟨*proof*⟩

lemma *emeasure-lborel-Ioo*[*simp*]:
  assumes [*simp*]: $l \leq u$
  shows $emeasure\ lborel\ \{l\ {<}..{<}\ u\} = ennreal\ (u - l)$
⟨*proof*⟩

lemma *emeasure-lborel-Ioc*[*simp*]:
  assumes [*simp*]: $l \leq u$
  shows $emeasure\ lborel\ \{l\ {<}..\ u\} = ennreal\ (u - l)$
⟨*proof*⟩

lemma *emeasure-lborel-Ico*[*simp*]:
  assumes [*simp*]: $l \leq u$
  shows $emeasure\ lborel\ \{l\ ..{<}\ u\} = ennreal\ (u - l)$
⟨*proof*⟩

lemma *emeasure-lborel-box*[*simp*]:
  assumes [*simp*]: $\bigwedge b.\ b \in Basis \Longrightarrow l \cdot b \leq u \cdot b$
  shows $emeasure\ lborel\ (box\ l\ u) = (\prod b{\in}Basis.\ (u - l) \cdot b)$
⟨*proof*⟩

lemma *emeasure-lborel-cbox-eq*:
  $emeasure\ lborel\ (cbox\ l\ u) = (if\ \forall\, b{\in}Basis.\ l \cdot b \leq u \cdot b\ then\ \prod b{\in}Basis.\ (u - l) \cdot b\ else\ 0)$
  ⟨*proof*⟩

lemma *emeasure-lborel-box-eq*:
  $emeasure\ lborel\ (box\ l\ u) = (if\ \forall\, b{\in}Basis.\ l \cdot b \leq u \cdot b\ then\ \prod b{\in}Basis.\ (u - l) \cdot b\ else\ 0)$
  ⟨*proof*⟩

lemma
  fixes $l\ u :: real$
  assumes [*simp*]: $l \leq u$
  shows *measure-lborel-Icc*[*simp*]: $measure\ lborel\ \{l\ ..\ u\} = u - l$
    and *measure-lborel-Ico*[*simp*]: $measure\ lborel\ \{l\ ..{<}\ u\} = u - l$
    and *measure-lborel-Ioc*[*simp*]: $measure\ lborel\ \{l\ {<}..\ u\} = u - l$
    and *measure-lborel-Ioo*[*simp*]: $measure\ lborel\ \{l\ {<}..{<}\ u\} = u - l$
  ⟨*proof*⟩

lemma

assumes [*simp*]: $\bigwedge b.\ b \in Basis \Longrightarrow l \cdot b \leq u \cdot b$
shows *measure-lborel-box*[*simp*]: $measure\ lborel\ (box\ l\ u) = (\prod b{\in}Basis.\ (u - l) \cdot b)$
and *measure-lborel-cbox*[*simp*]: $measure\ lborel\ (cbox\ l\ u) = (\prod b{\in}Basis.\ (u - l) \cdot b)$
⟨*proof*⟩

lemma *sigma-finite-lborel*: *sigma-finite-measure lborel*
⟨*proof*⟩

end

lemma *emeasure-lborel-UNIV*: $emeasure\ lborel\ (UNIV::'a::euclidean\text{-}space\ set) = \infty$
⟨*proof*⟩

lemma *emeasure-lborel-singleton*[*simp*]: $emeasure\ lborel\ \{x\} = 0$
⟨*proof*⟩

lemma *emeasure-lborel-countable*:
fixes $A :: 'a::euclidean\text{-}space\ set$
assumes *countable A*
shows $emeasure\ lborel\ A = 0$
⟨*proof*⟩

lemma *countable-imp-null-set-lborel*: $countable\ A \Longrightarrow A \in null\text{-}sets\ lborel$
⟨*proof*⟩

lemma *finite-imp-null-set-lborel*: $finite\ A \Longrightarrow A \in null\text{-}sets\ lborel$
⟨*proof*⟩

lemma *lborel-neq-count-space*[*simp*]: $lborel \neq count\text{-}space\ (A::('a::ordered\text{-}euclidean\text{-}space)\ set)$
⟨*proof*⟩

10.3 Affine transformation on the Lebesgue-Borel

lemma *lborel-eqI*:
fixes $M :: 'a::euclidean\text{-}space\ measure$
assumes *emeasure-eq*: $\bigwedge l\ u.\ (\bigwedge b.\ b \in Basis \Longrightarrow l \cdot b \leq u \cdot b) \Longrightarrow emeasure\ M\ (box\ l\ u) = (\prod b{\in}Basis.\ (u - l) \cdot b)$
assumes *sets-eq*: $sets\ M = sets\ borel$
shows $lborel = M$
⟨*proof*⟩

lemma *lborel-affine*:
fixes $t :: 'a::euclidean\text{-}space$ **assumes** $c \neq 0$
shows $lborel = density\ (distr\ lborel\ borel\ (\lambda x.\ t + c *_R x))\ (\lambda\text{-}.\ |c|\ \hat{}\ DIM('a))$
(**is** - = *?D*)

⟨proof⟩

lemma *lborel-real-affine*:
$c \neq 0 \Longrightarrow lborel = density\ (distr\ lborel\ borel\ (\lambda x.\ t + c * x))\ (\lambda\text{-}.\ ennreal\ (abs\ c))$
⟨proof⟩

lemma *AE-borel-affine*:
fixes $P :: real \Rightarrow bool$
shows $c \neq 0 \Longrightarrow Measurable.pred\ borel\ P \Longrightarrow AE\ x\ in\ lborel.\ P\ x \Longrightarrow AE\ x\ in\ lborel.\ P\ (t + c * x)$
⟨proof⟩

lemma *nn-integral-real-affine*:
fixes $c :: real$ **assumes** [*measurable*]: $f \in borel\text{-}measurable\ borel$ **and** c: $c \neq 0$
shows $(\int^+ x.\ f\ x\ \partial lborel) = |c| * (\int^+ x.\ f\ (t + c * x)\ \partial lborel)$
⟨proof⟩

lemma *lborel-integrable-real-affine*:
fixes $f :: real \Rightarrow 'a :: \{banach,\ second\text{-}countable\text{-}topology\}$
assumes f: *integrable lborel f*
shows $c \neq 0 \Longrightarrow integrable\ lborel\ (\lambda x.\ f\ (t + c * x))$
⟨proof⟩

lemma *lborel-integrable-real-affine-iff*:
fixes $f :: real \Rightarrow 'a :: \{banach,\ second\text{-}countable\text{-}topology\}$
shows $c \neq 0 \Longrightarrow integrable\ lborel\ (\lambda x.\ f\ (t + c * x)) \longleftrightarrow integrable\ lborel\ f$
⟨proof⟩

lemma *lborel-integral-real-affine*:
fixes $f :: real \Rightarrow 'a :: \{banach,\ second\text{-}countable\text{-}topology\}$ **and** $c :: real$
assumes c: $c \neq 0$ **shows** $(\int x.\ f\ x\ \partial\ lborel) = |c| *_R (\int x.\ f\ (t + c * x)\ \partial lborel)$
⟨proof⟩

lemma *divideR-right*:
fixes $x\ y :: 'a::real\text{-}normed\text{-}vector$
shows $r \neq 0 \Longrightarrow y = x /_R r \longleftrightarrow r *_R y = x$
⟨proof⟩

lemma *lborel-has-bochner-integral-real-affine-iff*:
fixes $x :: 'a :: \{banach,\ second\text{-}countable\text{-}topology\}$
shows $c \neq 0 \Longrightarrow$
has-bochner-integral lborel f x $\longleftrightarrow$
has-bochner-integral lborel $(\lambda x.\ f\ (t + c * x))\ (x /_R |c|)$
⟨proof⟩

lemma *lborel-distr-uminus*: $distr\ lborel\ borel\ uminus = (lborel :: real\ measure)$
⟨proof⟩

lemma *lborel-distr-mult*:
assumes $(c::real) \neq 0$
shows $distr\ lborel\ borel\ (op * c) = density\ lborel\ (\lambda\text{-}.\ inverse\ |c|)$
⟨*proof*⟩

lemma *lborel-distr-mult'*:
assumes $(c::real) \neq 0$
shows $lborel = density\ (distr\ lborel\ borel\ (op * c))\ (\lambda\text{-}.\ |c|)$
⟨*proof*⟩

lemma *lborel-distr-plus*: $distr\ lborel\ borel\ (op + c) = (lborel :: real\ measure)$
⟨*proof*⟩

interpretation *lborel*: *sigma-finite-measure lborel*
⟨*proof*⟩

interpretation *lborel-pair*: *pair-sigma-finite lborel lborel* ⟨*proof*⟩

lemma *lborel-prod*:
$lborel \bigotimes_M lborel = (lborel :: ('a::euclidean\text{-}space \times 'b::euclidean\text{-}space)\ measure)$
⟨*proof*⟩

lemma *lborelD-Collect*[*measurable* (*raw*)]: $\{x \in space\ borel.\ P\ x\} \in sets\ borel \Longrightarrow \{x \in space\ lborel.\ P\ x\} \in sets\ lborel$ ⟨*proof*⟩
lemma *lborelD*[*measurable* (*raw*)]: $A \in sets\ borel \Longrightarrow A \in sets\ lborel$ ⟨*proof*⟩

10.4 Equivalence Lebesgue integral on *lborel* and HK-integral

lemma *has-integral-measure-lborel*:
fixes $A :: 'a::euclidean\text{-}space\ set$
assumes *A*[*measurable*]: $A \in sets\ borel$ **and** *finite*: $emeasure\ lborel\ A < \infty$
shows $((\lambda x.\ 1)\ has\text{-}integral\ measure\ lborel\ A)\ A$
⟨*proof*⟩

lemma *nn-integral-has-integral*:
fixes $f::'a::euclidean\text{-}space \Rightarrow real$
assumes f: $f \in borel\text{-}measurable\ borel\ \bigwedge x.\ 0 \leq f\ x\ (\int^+ x.\ f\ x\ \partial lborel) = ennreal\ r\ 0 \leq r$
shows $(f\ has\text{-}integral\ r)\ UNIV$
⟨*proof*⟩

lemma *nn-integral-lborel-eq-integral*:
fixes $f::'a::euclidean\text{-}space \Rightarrow real$
assumes f: $f \in borel\text{-}measurable\ borel\ \bigwedge x.\ 0 \leq f\ x\ (\int^+ x.\ f\ x\ \partial lborel) < \infty$
shows $(\int^+ x.\ f\ x\ \partial lborel) = integral\ UNIV\ f$
⟨*proof*⟩

lemma *nn-integral-integrable-on*:

fixes *f*::*'a*::*euclidean-space* $\Rightarrow$ *real*
assumes *f*: *f* $\in$ *borel-measurable borel* $\bigwedge x.\ 0 \le f\ x$ ($\int^+ x.\ f\ x\ \partial lborel$) $< \infty$
shows *f integrable-on UNIV*
⟨*proof*⟩

lemma *nn-integral-has-integral-lborel*:
fixes *f* :: *'a*::*euclidean-space* $\Rightarrow$ *real*
assumes *f-borel*: *f* $\in$ *borel-measurable borel* **and** *nonneg*: $\bigwedge x.\ 0 \le f\ x$
assumes *I*: (*f has-integral I*) *UNIV*
shows $integral^N$ *lborel f* = *I*
⟨*proof*⟩

lemma *has-integral-iff-emeasure-lborel*:
fixes *A* :: *'a*::*euclidean-space set*
assumes *A*[*measurable*]: *A* $\in$ *sets borel* **and** [*simp*]: $0 \le r$
shows (($\lambda x.\ 1$) *has-integral r*) *A* $\longleftrightarrow$ *emeasure lborel A* = *ennreal r*
⟨*proof*⟩

lemma *has-integral-integral-real*:
fixes *f*::*'a*::*euclidean-space* $\Rightarrow$ *real*
assumes *f*: *integrable lborel f*
shows (*f has-integral* ($integral^L$ *lborel f*)) *UNIV*
⟨*proof*⟩

context
fixes *f*::*'a*::*euclidean-space* $\Rightarrow$ *'b*::*euclidean-space*
begin

lemma *has-integral-integral-lborel*:
assumes *f*: *integrable lborel f*
shows (*f has-integral* ($integral^L$ *lborel f*)) *UNIV*
⟨*proof*⟩

lemma *integrable-on-lborel*: *integrable lborel f* $\Longrightarrow$ *f integrable-on UNIV*
⟨*proof*⟩

lemma *integral-lborel*: *integrable lborel f* $\Longrightarrow$ *integral UNIV f* = ($\int x.\ f\ x\ \partial lborel$)
⟨*proof*⟩

end

10.5 Fundamental Theorem of Calculus for the Lebesgue integral

lemma *emeasure-bounded-finite*:
assumes *bounded A* **shows** *emeasure lborel A* $< \infty$
⟨*proof*⟩

lemma *emeasure-compact-finite*: *compact A* $\Longrightarrow$ *emeasure lborel A* $< \infty$

⟨proof⟩

lemma *borel-integrable-compact*:
fixes f :: $'a$::*euclidean-space* ⇒ $'b$::{*banach*, *second-countable-topology*}
assumes *compact S continuous-on S f*
shows *integrable lborel* (λx. *indicator S x* $*_R$ *f x*)
⟨proof⟩

lemma *borel-integrable-atLeastAtMost*:
fixes f :: *real* ⇒ *real*
assumes f: $\bigwedge x.\ a \leq x \Longrightarrow x \leq b \Longrightarrow$ *isCont f x*
shows *integrable lborel* (λx. *f x* $*$ *indicator* {*a* .. *b*} *x*) (**is** *integrable - ?f*)
⟨proof⟩

For the positive integral we replace continuity with Borel-measurability.

lemma
fixes f :: *real* ⇒ *real*
assumes [*measurable*]: $f \in$ *borel-measurable borel*
assumes f: $\bigwedge x.\ x \in \{a..b\} \Longrightarrow$ *DERIV F x* :> *f x* $\bigwedge x.\ x \in \{a..b\} \Longrightarrow 0 \leq f\ x$
and $a \leq b$
shows *nn-integral-FTC-Icc*: ($\int^+ x$. *ennreal* (*f x*) $*$ *indicator* {*a* .. *b*} *x* ∂*lborel*) = *F b* − *F a* (**is** *?nn*)
and *has-bochner-integral-FTC-Icc-nonneg*:
has-bochner-integral lborel (λx. *f x* $*$ *indicator* {*a* .. *b*} *x*) (*F b* − *F a*) (**is** *?has*)
and *integral-FTC-Icc-nonneg*: ($\int x$. *f x* $*$ *indicator* {*a* .. *b*} *x* ∂*lborel*) = *F b* − *F a* (**is** *?eq*)
and *integrable-FTC-Icc-nonneg*: *integrable lborel* (λx. *f x* $*$ *indicator* {*a* .. *b*} *x*) (**is** *?int*)
⟨proof⟩

lemma
fixes f :: *real* ⇒ $'a$ :: *euclidean-space*
assumes $a \leq b$
assumes $\bigwedge x.\ a \leq x \Longrightarrow x \leq b \Longrightarrow$ (*F has-vector-derivative f x*) (*at x within* {*a* .. *b*})
assumes *cont*: *continuous-on* {*a* .. *b*} *f*
shows *has-bochner-integral-FTC-Icc*:
has-bochner-integral lborel (λx. *indicator* {*a* .. *b*} *x* $*_R$ *f x*) (*F b* − *F a*) (**is** *?has*)
and *integral-FTC-Icc*: ($\int x$. *indicator* {*a* .. *b*} *x* $*_R$ *f x* ∂*lborel*) = *F b* − *F a* (**is** *?eq*)
⟨proof⟩

lemma
fixes f :: *real* ⇒ *real*
assumes $a \leq b$
assumes *deriv*: $\bigwedge x.\ a \leq x \Longrightarrow x \leq b \Longrightarrow$ *DERIV F x* :> *f x*
assumes *cont*: $\bigwedge x.\ a \leq x \Longrightarrow x \leq b \Longrightarrow$ *isCont f x*

shows *has-bochner-integral-FTC-Icc-real*:
has-bochner-integral lborel (λx. *f x* $*$ *indicator* $\{a\ ..\ b\}$ *x*) (*F b* $-$ *F a*) (**is** *?has*)
and *integral-FTC-Icc-real*: ($\int x$. *f x* $*$ *indicator* $\{a\ ..\ b\}$ *x* ∂*lborel*) = *F b* $-$ *F a* (**is** *?eq*)
⟨*proof*⟩

lemma *nn-integral-FTC-atLeast*:
fixes *f* :: *real* $\Rightarrow$ *real*
assumes *f-borel*: *f* $\in$ *borel-measurable borel*
assumes *f*: $\bigwedge x$. *a* $\leq$ *x* $\Longrightarrow$ *DERIV F x* :> *f x*
assumes *nonneg*: $\bigwedge x$. *a* $\leq$ *x* $\Longrightarrow$ *0* $\leq$ *f x*
assumes *lim*: (*F* $\longrightarrow$ *T*) *at-top*
shows ($\int^+ x$. *ennreal* (*f x*) $*$ *indicator* $\{a\ ..\}$ *x* ∂*lborel*) = *T* $-$ *F a*
⟨*proof*⟩

lemma *integral-power*:
a $\leq$ *b* $\Longrightarrow$ ($\int x$. *x^k* $*$ *indicator* $\{a..b\}$ *x* ∂*lborel*) = (*b^Suc k* $-$ *a^Suc k*) / *Suc k*
⟨*proof*⟩

10.6 Integration by parts

lemma *integral-by-parts-integrable*:
fixes *f g F G*::*real* $\Rightarrow$ *real*
assumes *a* $\leq$ *b*
assumes *cont-f*[*intro*]: !!*x*. *a* $\leq$*x* $\Longrightarrow$ *x*$\leq$*b* $\Longrightarrow$ *isCont f x*
assumes *cont-g*[*intro*]: !!*x*. *a* $\leq$*x* $\Longrightarrow$ *x*$\leq$*b* $\Longrightarrow$ *isCont g x*
assumes [*intro*]: !!*x*. *DERIV F x* :> *f x*
assumes [*intro*]: !!*x*. *DERIV G x* :> *g x*
shows *integrable lborel* (λx.((*F x*) $*$ (*g x*) + (*f x*) $*$ (*G x*)) $*$ *indicator* $\{a\ ..\ b\}$ *x*)
⟨*proof*⟩

lemma *integral-by-parts*:
fixes *f g F G*::*real* $\Rightarrow$ *real*
assumes [*arith*]: *a* $\leq$ *b*
assumes *cont-f*[*intro*]: !!*x*. *a* $\leq$*x* $\Longrightarrow$ *x*$\leq$*b* $\Longrightarrow$ *isCont f x*
assumes *cont-g*[*intro*]: !!*x*. *a* $\leq$*x* $\Longrightarrow$ *x*$\leq$*b* $\Longrightarrow$ *isCont g x*
assumes [*intro*]: !!*x*. *DERIV F x* :> *f x*
assumes [*intro*]: !!*x*. *DERIV G x* :> *g x*
shows ($\int x$. (*F x* $*$ *g x*) $*$ *indicator* $\{a\ ..\ b\}$ *x* ∂*lborel*)
= *F b* $*$ *G b* $-$ *F a* $*$ *G a* $-$ $\int x$. (*f x* $*$ *G x*) $*$ *indicator* $\{a\ ..\ b\}$ *x* ∂*lborel*
⟨*proof*⟩

lemma *integral-by-parts'*:
fixes *f g F G*::*real* $\Rightarrow$ *real*
assumes *a* $\leq$ *b*

assumes $!!x.\ a \leq x \Longrightarrow x \leq b \Longrightarrow isCont\ f\ x$
assumes $!!x.\ a \leq x \Longrightarrow x \leq b \Longrightarrow isCont\ g\ x$
assumes $!!x.\ DERIV\ F\ x :> f\ x$
assumes $!!x.\ DERIV\ G\ x :> g\ x$
shows $(\int x.\ indicator\ \{a\ ..\ b\}\ x *_R (F\ x * g\ x)\ \partial lborel)$
$= F\ b * G\ b - F\ a * G\ a - \int x.\ indicator\ \{a\ ..\ b\}\ x *_R (f\ x * G\ x)$
$\partial lborel$
⟨*proof*⟩

lemma *has-bochner-integral-even-function*:
fixes $f :: real \Rightarrow 'a :: \{banach, second\text{-}countable\text{-}topology\}$
assumes f: *has-bochner-integral lborel* $(\lambda x.\ indicator\ \{0..\}\ x *_R f\ x)\ x$
assumes *even*: $\bigwedge x.\ f\ (-\ x) = f\ x$
shows *has-bochner-integral lborel f* $(2 *_R x)$
⟨*proof*⟩

lemma *has-bochner-integral-odd-function*:
fixes $f :: real \Rightarrow 'a :: \{banach, second\text{-}countable\text{-}topology\}$
assumes f: *has-bochner-integral lborel* $(\lambda x.\ indicator\ \{0..\}\ x *_R f\ x)\ x$
assumes *odd*: $\bigwedge x.\ f\ (-\ x) = -\ f\ x$
shows *has-bochner-integral lborel f 0*
⟨*proof*⟩

end

11 Radon-Nikodým derivative

theory *Radon-Nikodym*
imports *Bochner-Integration*
begin

definition *diff-measure M N* =
measure-of (*space M*) (*sets M*) ($\lambda A.$ *emeasure M A* − *emeasure N A*)

lemma
shows *space-diff-measure*[*simp*]: *space* (*diff-measure M N*) = *space M*
and *sets-diff-measure*[*simp*]: *sets* (*diff-measure M N*) = *sets M*
⟨*proof*⟩

lemma *emeasure-diff-measure*:
assumes *fin*: *finite-measure M finite-measure N* **and** *sets-eq*: *sets M* = *sets N*
assumes *pos*: $\bigwedge A.\ A \in sets\ M \Longrightarrow emeasure\ N\ A \leq emeasure\ M\ A$ **and** *A*: A
$\in sets\ M$
shows *emeasure* (*diff-measure M N*) *A* = *emeasure M A* − *emeasure N A* (**is** -
= *?μ A*)
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *Ex-finite-integrable-function*:
$\exists h \in borel\text{-}measurable\ M.\ integral^N\ M\ h \neq \infty \wedge (\forall x \in space\ M.\ 0 < h\ x \wedge h\ x <$

∞)
⟨*proof*⟩

11.1 Absolutely continuous

definition *absolutely-continuous* :: *'a measure* ⇒ *'a measure* ⇒ *bool* **where**
absolutely-continuous M N ⟷ *null-sets M* ⊆ *null-sets N*

lemma *absolutely-continuousI-count-space*: *absolutely-continuous* (*count-space A*) *M*
⟨*proof*⟩

lemma *absolutely-continuousI-density*:
f ∈ *borel-measurable M* ⟹ *absolutely-continuous M* (*density M f*)
⟨*proof*⟩

lemma *absolutely-continuousI-point-measure-finite*:
(⋀*x*. ⟦ *x* ∈ *A* ; *f x* ≤ *0* ⟧ ⟹ *g x* ≤ *0*) ⟹ *absolutely-continuous* (*point-measure A f*) (*point-measure A g*)
⟨*proof*⟩

lemma *absolutely-continuous-AE*:
assumes *sets-eq*: *sets M'* = *sets M*
and *absolutely-continuous M M' AE x in M*. *P x*
shows *AE x in M'*. *P x*
⟨*proof*⟩

11.2 Existence of the Radon-Nikodym derivative

lemma (**in** *finite-measure*) *Radon-Nikodym-aux-epsilon*:
fixes *e* :: *real* **assumes** *0* < *e*
assumes *finite-measure N* **and** *sets-eq*: *sets N* = *sets M*
shows ∃ *A*∈*sets M*. *measure M* (*space M*) − *measure N* (*space M*) ≤ *measure M A* − *measure N A* ∧
(∀ *B*∈*sets M*. *B* ⊆ *A* ⟶ − *e* < *measure M B* − *measure N B*)
⟨*proof*⟩

lemma (**in** *finite-measure*) *Radon-Nikodym-aux*:
assumes *finite-measure N* **and** *sets-eq*: *sets N* = *sets M*
shows ∃ *A*∈*sets M*. *measure M* (*space M*) − *measure N* (*space M*) ≤
measure M A − *measure N A* ∧
(∀ *B*∈*sets M*. *B* ⊆ *A* ⟶ *0* ≤ *measure M B* − *measure N B*)
⟨*proof*⟩

lemma (**in** *finite-measure*) *Radon-Nikodym-finite-measure*:
assumes *finite-measure N* **and** *sets-eq*: *sets N* = *sets M*
assumes *absolutely-continuous M N*
shows ∃*f* ∈ *borel-measurable M*. (∀ *x*. *0* ≤ *f x*) ∧ *density M f* = *N*
⟨*proof*⟩

lemma (**in** *finite-measure*) *split-space-into-finite-sets-and-rest*:
assumes *ac*: *absolutely-continuous M N* **and** *sets-eq*: *sets N = sets M*
shows $\exists A0 \in sets\ M.\ \exists B::nat \Rightarrow 'a\ set.\ disjoint\text{-}family\ B \wedge range\ B \subseteq sets\ M \wedge A0 = space\ M - (\bigcup i.\ B\ i) \wedge$
$(\forall A \in sets\ M.\ A \subseteq A0 \longrightarrow (emeasure\ M\ A = 0 \wedge N\ A = 0) \vee (emeasure\ M\ A > 0 \wedge N\ A = \infty)) \wedge$
$(\forall i.\ N\ (B\ i) \neq \infty)$
⟨*proof*⟩

lemma (**in** *finite-measure*) *Radon-Nikodym-finite-measure-infinite*:
assumes *absolutely-continuous M N* **and** *sets-eq*: *sets N = sets M*
shows $\exists f \in borel\text{-}measurable\ M.\ (\forall x.\ 0 \leq f\ x) \wedge density\ M\ f = N$
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *Radon-Nikodym*:
assumes *ac*: *absolutely-continuous M N* **assumes** *sets-eq*: *sets N = sets M*
shows $\exists f \in borel\text{-}measurable\ M.\ (\forall x.\ 0 \leq f\ x) \wedge density\ M\ f = N$
⟨*proof*⟩

11.3 Uniqueness of densities

lemma *finite-density-unique*:
assumes *borel*: $f \in borel\text{-}measurable\ M\ g \in borel\text{-}measurable\ M$
assumes *pos*: $AE\ x\ in\ M.\ 0 \leq f\ x\ AE\ x\ in\ M.\ 0 \leq g\ x$
and *fin*: $integral^N\ M\ f \neq \infty$
shows $density\ M\ f = density\ M\ g \longleftrightarrow (AE\ x\ in\ M.\ f\ x = g\ x)$
⟨*proof*⟩

lemma (**in** *finite-measure*) *density-unique-finite-measure*:
assumes *borel*: $f \in borel\text{-}measurable\ M\ f' \in borel\text{-}measurable\ M$
assumes *pos*: $AE\ x\ in\ M.\ 0 \leq f\ x\ AE\ x\ in\ M.\ 0 \leq f'\ x$
assumes *f*: $\bigwedge A.\ A \in sets\ M \Longrightarrow (\int^+ x.\ f\ x * indicator\ A\ x\ \partial M) = (\int^+ x.\ f'\ x * indicator\ A\ x\ \partial M)$
(**is** $\bigwedge A.\ A \in sets\ M \Longrightarrow ?P\ f\ A = ?P\ f'\ A$)
shows $AE\ x\ in\ M.\ f\ x = f'\ x$
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *density-unique*:
assumes *f*: $f \in borel\text{-}measurable\ M$
assumes *f'*: $f' \in borel\text{-}measurable\ M$
assumes *density-eq*: $density\ M\ f = density\ M\ f'$
shows $AE\ x\ in\ M.\ f\ x = f'\ x$
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *density-unique-iff*:
assumes *f*: $f \in borel\text{-}measurable\ M$ **and** *f'*: $f' \in borel\text{-}measurable\ M$
shows $density\ M\ f = density\ M\ f' \longleftrightarrow (AE\ x\ in\ M.\ f\ x = f'\ x)$
⟨*proof*⟩

lemma *sigma-finite-density-unique*:
assumes *borel*: $f \in$ *borel-measurable M g* $\in$ *borel-measurable M*
and *fin*: *sigma-finite-measure* (*density M f*)
shows *density M f* = *density M g* $\longleftrightarrow$ (*AE x in M*. *f x* = *g x*)
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *sigma-finite-iff-density-finite'*:
assumes *f*: $f \in$ *borel-measurable M*
shows *sigma-finite-measure* (*density M f*) $\longleftrightarrow$ (*AE x in M*. *f x* $\neq \infty$)
(**is** *sigma-finite-measure ?N* $\longleftrightarrow$ -)
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *sigma-finite-iff-density-finite*:
$f \in$ *borel-measurable M* $\Longrightarrow$ *sigma-finite-measure* (*density M f*) $\longleftrightarrow$ (*AE x in M*. *f x* $\neq \infty$)
⟨*proof*⟩

11.4 Radon-Nikodym derivative

definition *RN-deriv* :: *'a measure* $\Rightarrow$ *'a measure* $\Rightarrow$ *'a* $\Rightarrow$ *ennreal* **where**
RN-deriv M N =
(*if* $\exists f$. $f \in$ *borel-measurable M* $\wedge$ *density M f* = *N*
then SOME f. $f \in$ *borel-measurable M* $\wedge$ *density M f* = *N*
else (λ-. *0*))

lemma *RN-derivI*:
assumes $f \in$ *borel-measurable M density M f* = *N*
shows *density M* (*RN-deriv M N*) = *N*
⟨*proof*⟩

lemma *borel-measurable-RN-deriv*[*measurable*]: *RN-deriv M N* $\in$ *borel-measurable M*
⟨*proof*⟩

lemma *density-RN-deriv-density*:
assumes *f*: $f \in$ *borel-measurable M*
shows *density M* (*RN-deriv M* (*density M f*)) = *density M f*
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *density-RN-deriv*:
absolutely-continuous M N $\Longrightarrow$ *sets N* = *sets M* $\Longrightarrow$ *density M* (*RN-deriv M N*) = *N*
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *RN-deriv-nn-integral*:
assumes *N*: *absolutely-continuous M N sets N* = *sets M*
and *f*: $f \in$ *borel-measurable M*
shows $integral^N$ *N f* = ($\int^+ x$. *RN-deriv M N x* $*$ *f x* ∂M)
⟨*proof*⟩

lemma *null-setsD-AE*: $N \in$ *null-sets* $M \Longrightarrow$ *AE x in M*. $x \notin N$
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *RN-deriv-unique*:
assumes *f*: $f \in$ *borel-measurable M*
and *eq*: *density M f* = *N*
shows *AE x in M*. *f x* = *RN-deriv M N x*
⟨*proof*⟩

lemma *RN-deriv-unique-sigma-finite*:
assumes *f*: $f \in$ *borel-measurable M*
and *eq*: *density M f* = *N* **and** *fin*: *sigma-finite-measure N*
shows *AE x in M*. *f x* = *RN-deriv M N x*
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *RN-deriv-distr*:
fixes $T :: 'a \Rightarrow 'b$
assumes *T*: $T \in$ *measurable M M'* **and** *T'*: $T' \in$ *measurable M' M*
and *inv*: $\forall x \in$ *space M*. T' $(T\ x) = x$
and *ac*[*simp*]: *absolutely-continuous* (*distr M M' T*) (*distr N M' T*)
and *N*: *sets N* = *sets M*
shows *AE x in M*. *RN-deriv* (*distr M M' T*) (*distr N M' T*) (*T x*) = *RN-deriv M N x*
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *RN-deriv-finite*:
assumes *N*: *sigma-finite-measure N* **and** *ac*: *absolutely-continuous M N sets N* = *sets M*
shows *AE x in M*. *RN-deriv M N x* $\neq \infty$
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*)
assumes *N*: *sigma-finite-measure N* **and** *ac*: *absolutely-continuous M N sets N* = *sets M*
and *f*: $f \in$ *borel-measurable M*
shows *RN-deriv-integrable*: *integrable N f* $\longleftrightarrow$
integrable M (λx. *enn2real* (*RN-deriv M N x*) $*$ *f x*) (**is** *?integrable*)
and *RN-deriv-integral*: $integral^L$ *N f* = ($\int x$. *enn2real* (*RN-deriv M N x*) $*$ *f x* ∂M) (**is** *?integral*)
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *real-RN-deriv*:
assumes *finite-measure N*
assumes *ac*: *absolutely-continuous M N sets N* = *sets M*
obtains *D* **where** $D \in$ *borel-measurable M*
and *AE x in M*. *RN-deriv M N x* = *ennreal* (*D x*)
and *AE x in N*. $0 < D\ x$
and $\bigwedge x.\ 0 \leq D\ x$

⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *RN-deriv-singleton*:
assumes *ac*: *absolutely-continuous M N sets N = sets M*
and *x*: $\{x\} \in$ *sets M*
shows *N* $\{x\}$ = *RN-deriv M N x* $*$ *emeasure M* $\{x\}$
⟨*proof*⟩

end

12 Probability measure

theory *Probability-Measure*
imports *Lebesgue-Measure Radon-Nikodym*
begin

lemma (**in** *finite-measure*) *countable-support*:
countable $\{x.$ *measure M* $\{x\} \neq 0\}$
⟨*proof*⟩

locale *prob-space* = *finite-measure* +
assumes *emeasure-space-1*: *emeasure M* (*space M*) = *1*

lemma *prob-spaceI*[*Pure.intro*!]:
assumes $*$: *emeasure M* (*space M*) = *1*
shows *prob-space M*
⟨*proof*⟩

lemma *prob-space-imp-sigma-finite*: *prob-space M* $\Longrightarrow$ *sigma-finite-measure M*
⟨*proof*⟩

abbreviation (**in** *prob-space*) *events* $\equiv$ *sets M*
abbreviation (**in** *prob-space*) *prob* $\equiv$ *measure M*
abbreviation (**in** *prob-space*) *random-variable* M' $X \equiv X \in$ *measurable M* M'
abbreviation (**in** *prob-space*) *expectation* $\equiv$ *integral*^{L} *M*
abbreviation (**in** *prob-space*) *variance X* $\equiv$ *integral*^{L} *M* ($\lambda x.$ (*X x* $-$ *expectation X*)^2)

lemma (**in** *prob-space*) *finite-measure* [*simp*]: *finite-measure M*
⟨*proof*⟩

lemma (**in** *prob-space*) *prob-space-distr*:
assumes *f*: $f \in$ *measurable M* M' **shows** *prob-space* (*distr M* M' *f*)
⟨*proof*⟩

lemma *prob-space-distrD*:
assumes *f*: $f \in$ *measurable M N* **and** *M*: *prob-space* (*distr M N f*) **shows** *prob-space M*
⟨*proof*⟩

lemma (**in** *prob-space*) *prob-space*: *prob* (*space M*) = *1*
⟨*proof*⟩

lemma (**in** *prob-space*) *prob-le-1*[*simp*, *intro*]: *prob A* ≤ *1*
⟨*proof*⟩

lemma (**in** *prob-space*) *not-empty*: *space M* ≠ {}
⟨*proof*⟩

lemma (**in** *prob-space*) *emeasure-eq-1-AE*:
S ∈ *sets M* ⟹ *AE x in M*. *x* ∈ *S* ⟹ *emeasure M S* = *1*
⟨*proof*⟩

lemma (**in** *prob-space*) *emeasure-le-1*: *emeasure M S* ≤ *1*
⟨*proof*⟩

lemma (**in** *prob-space*) *AE-iff-emeasure-eq-1*:
assumes [*measurable*]: *Measurable.pred M P*
shows (*AE x in M*. *P x*) ⟷ *emeasure M* {*x*∈*space M*. *P x*} = *1*
⟨*proof*⟩

lemma (**in** *prob-space*) *measure-le-1*: *emeasure M X* ≤ *1*
⟨*proof*⟩

lemma (**in** *prob-space*) *AE-I-eq-1*:
assumes *emeasure M* {*x*∈*space M*. *P x*} = *1* {*x*∈*space M*. *P x*} ∈ *sets M*
shows *AE x in M*. *P x*
⟨*proof*⟩

lemma *prob-space-restrict-space*:
S ∈ *sets M* ⟹ *emeasure M S* = *1* ⟹ *prob-space* (*restrict-space M S*)
⟨*proof*⟩

lemma (**in** *prob-space*) *prob-compl*:
assumes *A*: *A* ∈ *events*
shows *prob* (*space M* − *A*) = *1* − *prob A*
⟨*proof*⟩

lemma (**in** *prob-space*) *AE-in-set-eq-1*:
assumes *A*[*measurable*]: *A* ∈ *events* **shows** (*AE x in M*. *x* ∈ *A*) ⟷ *prob A* = *1*
⟨*proof*⟩

lemma (**in** *prob-space*) *AE-False*: (*AE x in M*. *False*) ⟷ *False*
⟨*proof*⟩

lemma (**in** *prob-space*) *AE-prob-1*:
assumes *prob A* = *1* **shows** *AE x in M*. *x* ∈ *A*

⟨*proof*⟩

lemma (**in** *prob-space*) *AE-const*[*simp*]: $(AE\ x\ in\ M.\ P) \longleftrightarrow P$
⟨*proof*⟩

lemma (**in** *prob-space*) *ae-filter-bot*: $ae\text{-}filter\ M \neq bot$
⟨*proof*⟩

lemma (**in** *prob-space*) *AE-contr*:
assumes *ae*: $AE\ \omega\ in\ M.\ P\ \omega$ $AE\ \omega\ in\ M.\ \neg\ P\ \omega$
shows *False*
⟨*proof*⟩

lemma (**in** *prob-space*) *integral-ge-const*:
fixes c :: *real*
shows $integrable\ M\ f \Longrightarrow (AE\ x\ in\ M.\ c \leq f\ x) \Longrightarrow c \leq (\int x.\ f\ x\ \partial M)$
⟨*proof*⟩

lemma (**in** *prob-space*) *integral-le-const*:
fixes c :: *real*
shows $integrable\ M\ f \Longrightarrow (AE\ x\ in\ M.\ f\ x \leq c) \Longrightarrow (\int x.\ f\ x\ \partial M) \leq c$
⟨*proof*⟩

lemma (**in** *prob-space*) *nn-integral-ge-const*:
$(AE\ x\ in\ M.\ c \leq f\ x) \Longrightarrow c \leq (\int^{+} x.\ f\ x\ \partial M)$
⟨*proof*⟩

lemma (**in** *prob-space*) *expectation-less*:
fixes X :: - $\Rightarrow$ *real*
assumes [*simp*]: *integrable M X*
assumes *gt*: $AE\ x\ in\ M.\ X\ x < b$
shows $expectation\ X < b$
⟨*proof*⟩

lemma (**in** *prob-space*) *expectation-greater*:
fixes X :: - $\Rightarrow$ *real*
assumes [*simp*]: *integrable M X*
assumes *gt*: $AE\ x\ in\ M.\ a < X\ x$
shows $a < expectation\ X$
⟨*proof*⟩

lemma (**in** *prob-space*) *jensens-inequality*:
fixes q :: *real* $\Rightarrow$ *real*
assumes X: $integrable\ M\ X$ $AE\ x\ in\ M.\ X\ x \in I$
assumes I: $I = \{a <..< b\} \vee I = \{a <..\} \vee I = \{..< b\} \vee I = UNIV$
assumes q: $integrable\ M\ (\lambda x.\ q\ (X\ x))$ $convex\text{-}on\ I\ q$
shows $q\ (expectation\ X) \leq expectation\ (\lambda x.\ q\ (X\ x))$
⟨*proof*⟩

12.1 Introduce binder for probability

syntax
-prob :: *pttrn* ⇒ *logic* ⇒ *logic* ⇒ *logic* ((′$\mathcal{P}$′((/- *in* -./ -)′)))

translations
$\mathcal{P}$(*x in M*. *P*) => *CONST measure M* {*x* ∈ *CONST space M*. *P*}

⟨*ML*⟩

definition
cond-prob M P Q = $\mathcal{P}$(ω *in M*. *P* ω ∧ *Q* ω) / $\mathcal{P}$(ω *in M*. *Q* ω)

syntax
-conditional-prob :: *pttrn* ⇒ *logic* ⇒ *logic* ⇒ *logic* ⇒ *logic* ((′$\mathcal{P}$′(- *in* -. - |/ -′)))

translations
$\mathcal{P}$(*x in M*. *P* | *Q*) => *CONST cond-prob M* (λ*x*. *P*) (λ*x*. *Q*)

lemma (**in** *prob-space*) *AE-E-prob*:
assumes *ae*: *AE x in M*. *P x*
obtains *S* **where** *S* ⊆ {*x* ∈ *space M*. *P x*} *S* ∈ *events prob S* = *1*
⟨*proof*⟩

lemma (**in** *prob-space*) *prob-neg*: {*x*∈*space M*. *P x*} ∈ *events* ⟹ $\mathcal{P}$(*x in M*. ¬ *P x*) = *1* − $\mathcal{P}$(*x in M*. *P x*)
⟨*proof*⟩

lemma (**in** *prob-space*) *prob-eq-AE*:
(*AE x in M*. *P x* ⟷ *Q x*) ⟹ {*x*∈*space M*. *P x*} ∈ *events* ⟹ {*x*∈*space M*. *Q x*} ∈ *events* ⟹ $\mathcal{P}$(*x in M*. *P x*) = $\mathcal{P}$(*x in M*. *Q x*)
⟨*proof*⟩

lemma (**in** *prob-space*) *prob-eq-0-AE*:
assumes *not*: *AE x in M*. ¬ *P x* **shows** $\mathcal{P}$(*x in M*. *P x*) = *0*
⟨*proof*⟩

lemma (**in** *prob-space*) *prob-Collect-eq-0*:
{*x* ∈ *space M*. *P x*} ∈ *sets M* ⟹ $\mathcal{P}$(*x in M*. *P x*) = *0* ⟷ (*AE x in M*. ¬ *P x*)
⟨*proof*⟩

lemma (**in** *prob-space*) *prob-Collect-eq-1*:
{*x* ∈ *space M*. *P x*} ∈ *sets M* ⟹ $\mathcal{P}$(*x in M*. *P x*) = *1* ⟷ (*AE x in M*. *P x*)
⟨*proof*⟩

lemma (**in** *prob-space*) *prob-eq-0*:
A ∈ *sets M* ⟹ *prob A* = *0* ⟷ (*AE x in M*. *x* ∉ *A*)
⟨*proof*⟩

lemma (**in** *prob-space*) *prob-eq-1*:

$A \in sets\ M \Longrightarrow prob\ A = 1 \longleftrightarrow (AE\ x\ in\ M.\ x \in A)$
⟨proof⟩

lemma (**in** *prob-space*) *prob-sums*:
assumes P: $\bigwedge n.\ \{x{\in}space\ M.\ P\ n\ x\} \in events$
assumes Q: $\{x{\in}space\ M.\ Q\ x\} \in events$
assumes *ae*: $AE\ x\ in\ M.\ (\forall n.\ P\ n\ x \longrightarrow Q\ x) \wedge (Q\ x \longrightarrow (\exists! n.\ P\ n\ x))$
shows $(\lambda n.\ \mathcal{P}(x\ in\ M.\ P\ n\ x))\ sums\ \mathcal{P}(x\ in\ M.\ Q\ x)$
⟨proof⟩

lemma (**in** *prob-space*) *prob-setsum*:
assumes [*simp*, *intro*]: *finite I*
assumes P: $\bigwedge n.\ n \in I \Longrightarrow \{x{\in}space\ M.\ P\ n\ x\} \in events$
assumes Q: $\{x{\in}space\ M.\ Q\ x\} \in events$
assumes *ae*: $AE\ x\ in\ M.\ (\forall n{\in}I.\ P\ n\ x \longrightarrow Q\ x) \wedge (Q\ x \longrightarrow (\exists! n{\in}I.\ P\ n\ x))$
shows $\mathcal{P}(x\ in\ M.\ Q\ x) = (\sum n{\in}I.\ \mathcal{P}(x\ in\ M.\ P\ n\ x))$
⟨proof⟩

lemma (**in** *prob-space*) *prob-EX-countable*:
assumes *sets*: $\bigwedge i.\ i \in I \Longrightarrow \{x{\in}space\ M.\ P\ i\ x\} \in sets\ M$ **and** I: *countable I*
assumes *disj*: $AE\ x\ in\ M.\ \forall i{\in}I.\ \forall j{\in}I.\ P\ i\ x \longrightarrow P\ j\ x \longrightarrow i = j$
shows $\mathcal{P}(x\ in\ M.\ \exists i{\in}I.\ P\ i\ x) = (\int^+ i.\ \mathcal{P}(x\ in\ M.\ P\ i\ x)\ \partial count\text{-}space\ I)$
⟨proof⟩

lemma (**in** *prob-space*) *cond-prob-eq-AE*:
assumes P: $AE\ x\ in\ M.\ Q\ x \longrightarrow P\ x \longleftrightarrow P'\ x\ \{x{\in}space\ M.\ P\ x\} \in events$ $\{x{\in}space\ M.\ P'\ x\} \in events$
assumes Q: $AE\ x\ in\ M.\ Q\ x \longleftrightarrow Q'\ x\ \{x{\in}space\ M.\ Q\ x\} \in events\ \{x{\in}space\ M.\ Q'\ x\} \in events$
shows $cond\text{-}prob\ M\ P\ Q = cond\text{-}prob\ M\ P'\ Q'$
⟨proof⟩

lemma (**in** *prob-space*) *joint-distribution-Times-le-fst*:
$random\text{-}variable\ MX\ X \Longrightarrow random\text{-}variable\ MY\ Y \Longrightarrow A \in sets\ MX \Longrightarrow B \in sets\ MY$
$\Longrightarrow emeasure\ (distr\ M\ (MX \bigotimes_M MY)\ (\lambda x.\ (X\ x,\ Y\ x)))\ (A \times B) \le emeasure\ (distr\ M\ MX\ X)\ A$
⟨proof⟩

lemma (**in** *prob-space*) *joint-distribution-Times-le-snd*:
$random\text{-}variable\ MX\ X \Longrightarrow random\text{-}variable\ MY\ Y \Longrightarrow A \in sets\ MX \Longrightarrow B \in sets\ MY$
$\Longrightarrow emeasure\ (distr\ M\ (MX \bigotimes_M MY)\ (\lambda x.\ (X\ x,\ Y\ x)))\ (A \times B) \le emeasure\ (distr\ M\ MY\ Y)\ B$
⟨proof⟩

lemma (**in** *prob-space*) *variance-eq*:
fixes $X :: 'a \Rightarrow real$

assumes [*simp*]: *integrable M X*
assumes [*simp*]: *integrable M* $(\lambda x.\ (X\ x)^2)$
shows *variance X* = *expectation* $(\lambda x.\ (X\ x)^2) - (\textit{expectation}\ X)^2$
⟨*proof*⟩

lemma (**in** *prob-space*) *variance-positive*: $0 \leq$ *variance* $(X::'a \Rightarrow real)$
⟨*proof*⟩

lemma (**in** *prob-space*) *variance-mean-zero*:
expectation X = *0* $\Longrightarrow$ *variance X* = *expectation* (λ*x*. (*X x*) ˆ*2*)
⟨*proof*⟩

locale *pair-prob-space* = *pair-sigma-finite M1 M2* + *M1*: *prob-space M1* + *M2*: *prob-space M2* **for** *M1 M2*

sublocale *pair-prob-space* ⊆ *P?*: *prob-space M1* $\bigotimes_M$ *M2*
⟨*proof*⟩

locale *product-prob-space* = *product-sigma-finite M* **for** *M* :: *'i* ⇒ *'a measure* +
fixes *I* :: *'i set*
assumes *prob-space*: ⋀*i*. *prob-space* (*M i*)

sublocale *product-prob-space* ⊆ *M?*: *prob-space M i* **for** *i*
⟨*proof*⟩

locale *finite-product-prob-space* = *finite-product-sigma-finite M I* + *product-prob-space M I* **for** *M I*

sublocale *finite-product-prob-space* ⊆ *prob-space* Π_M *i*∈*I*. *M i*
⟨*proof*⟩

lemma (**in** *finite-product-prob-space*) *prob-times*:
assumes *X*: ⋀*i*. *i* ∈ *I* ⟹ *X i* ∈ *sets* (*M i*)
shows *prob* (Π_E *i*∈*I*. *X i*) = (∏ *i*∈*I*. *M.prob i* (*X i*))
⟨*proof*⟩

12.2 Distributions

definition *distributed* :: *'a measure* ⇒ *'b measure* ⇒ (*'a* ⇒ *'b*) ⇒ (*'b* ⇒ *ennreal*) ⇒ *bool*
where
distributed M N X f ⟷
distr M N X = *density N f* ∧ *f* ∈ *borel-measurable N* ∧ *X* ∈ *measurable M N*

term *distributed*

lemma
assumes *distributed M N X f*
shows *distributed-distr-eq-density*: *distr M N X* = *density N f*

and *distributed-measurable*: $X \in$ *measurable M N*
and *distributed-borel-measurable*: $f \in$ *borel-measurable N*
⟨*proof*⟩

lemma
assumes *D*: *distributed M N X f*
shows *distributed-measurable'*[*measurable-dest*]:
$g \in$ *measurable L M* $\Longrightarrow (\lambda x.\ X\ (g\ x)) \in$ *measurable L N*
and *distributed-borel-measurable'*[*measurable-dest*]:
$h \in$ *measurable L N* $\Longrightarrow (\lambda x.\ f\ (h\ x)) \in$ *borel-measurable L*
⟨*proof*⟩

lemma *distributed-real-measurable*:
$(\bigwedge x.\ x \in$ *space N* $\Longrightarrow 0 \leq f\ x) \Longrightarrow$ *distributed M N X* $(\lambda x.$ *ennreal* $(f\ x)) \Longrightarrow$
$f \in$ *borel-measurable N*
⟨*proof*⟩

lemma *distributed-real-measurable'*:
$(\bigwedge x.\ x \in$ *space N* $\Longrightarrow 0 \leq f\ x) \Longrightarrow$ *distributed M N X* $(\lambda x.$ *ennreal* $(f\ x)) \Longrightarrow$
$h \in$ *measurable L N* $\Longrightarrow (\lambda x.\ f\ (h\ x)) \in$ *borel-measurable L*
⟨*proof*⟩

lemma *joint-distributed-measurable1*:
distributed M $(S \bigotimes_M T)\ (\lambda x.\ (X\ x,\ Y\ x))\ f \Longrightarrow h1 \in$ *measurable N M* $\Longrightarrow$
$(\lambda x.\ X\ (h1\ x)) \in$ *measurable N S*
⟨*proof*⟩

lemma *joint-distributed-measurable2*:
distributed M $(S \bigotimes_M T)\ (\lambda x.\ (X\ x,\ Y\ x))\ f \Longrightarrow h2 \in$ *measurable N M* $\Longrightarrow$
$(\lambda x.\ Y\ (h2\ x)) \in$ *measurable N T*
⟨*proof*⟩

lemma *distributed-count-space*:
assumes *X*: *distributed M* (*count-space A*) *X P* **and** *a*: $a \in A$ **and** *A*: *finite A*
shows $P\ a =$ *emeasure M* $(X\ -\text{`}\ \{a\} \cap$ *space M*$)$
⟨*proof*⟩

lemma *distributed-cong-density*:
$(AE\ x\ in\ N.\ f\ x = g\ x) \Longrightarrow g \in$ *borel-measurable N* $\Longrightarrow f \in$ *borel-measurable N*
$\Longrightarrow$
distributed M N X f $\longleftrightarrow$ *distributed M N X g*
⟨*proof*⟩

lemma (**in** *prob-space*) *distributed-imp-emeasure-nonzero*:
assumes *X*: *distributed M MX X Px*
shows *emeasure MX* $\{x \in$ *space MX*$.\ Px\ x \neq 0\} \neq 0$
⟨*proof*⟩

lemma *subdensity*:

assumes T: $T \in$ *measurable* P Q
assumes f: *distributed* M P X f
assumes g: *distributed* M Q Y g
assumes Y: $Y = T \circ X$
shows *AE* x *in* P. g $(T\ x) = 0 \longrightarrow f\ x = 0$
⟨*proof*⟩

lemma *subdensity-real*:
fixes g :: $'a \Rightarrow$ *real* **and** f :: $'b \Rightarrow$ *real*
assumes T: $T \in$ *measurable* P Q
assumes f: *distributed* M P X f
assumes g: *distributed* M Q Y g
assumes Y: $Y = T \circ X$
shows (*AE* x *in* P. $0 \leq g\ (T\ x)) \Longrightarrow$ (*AE* x *in* P. $0 \leq f\ x) \Longrightarrow$ *AE* x *in* P. g $(T\ x) = 0 \longrightarrow f\ x = 0$
⟨*proof*⟩

lemma *distributed-emeasure*:
distributed M N X $f \Longrightarrow A \in$ *sets* $N \Longrightarrow$ *emeasure* M $(X$ $-`$ $A \cap$ *space* $M) =$ $(\int^+ x.\ f\ x *$ *indicator* $A\ x\ \partial N)$
⟨*proof*⟩

lemma *distributed-nn-integral*:
distributed M N X $f \Longrightarrow g \in$ *borel-measurable* $N \Longrightarrow (\int^+ x.\ f\ x * g\ x\ \partial N) =$ $(\int^+ x.\ g\ (X\ x)\ \partial M)$
⟨*proof*⟩

lemma *distributed-integral*:
distributed M N X $f \Longrightarrow g \in$ *borel-measurable* $N \Longrightarrow (\bigwedge x.\ x \in$ *space* $N \Longrightarrow 0$ $\leq f\ x) \Longrightarrow$
$(\int x.\ f\ x * g\ x\ \partial N) = (\int x.\ g\ (X\ x)\ \partial M)$
⟨*proof*⟩

lemma *distributed-transform-integral*:
assumes Px: *distributed* M N X Px $\bigwedge x.\ x \in$ *space* $N \Longrightarrow 0 \leq Px\ x$
assumes *distributed* M P Y Py $\bigwedge x.\ x \in$ *space* $P \Longrightarrow 0 \leq Py\ x$
assumes Y: $Y = T \circ X$ **and** T: $T \in$ *measurable* N P **and** f: $f \in$ *borel-measurable* P
shows $(\int x.\ Py\ x * f\ x\ \partial P) = (\int x.\ Px\ x * f\ (T\ x)\ \partial N)$
⟨*proof*⟩

lemma (**in** *prob-space*) *distributed-unique*:
assumes Px: *distributed* M S X Px
assumes Py: *distributed* M S X Py
shows *AE* x *in* S. $Px\ x = Py\ x$
⟨*proof*⟩

lemma (**in** *prob-space*) *distributed-jointI*:
assumes *sigma-finite-measure* S *sigma-finite-measure* T

assumes $X[measurable]$: $X \in measurable\ M\ S$ **and** $Y[measurable]$: $Y \in measurable\ M\ T$
assumes $[measurable]$: $f \in borel\text{-}measurable\ (S \bigotimes_M T)$ **and** f: $AE\ x\ in\ S \bigotimes_M T.\ 0 \leq f\ x$
assumes eq: $\bigwedge A\ B.\ A \in sets\ S \Longrightarrow B \in sets\ T \Longrightarrow$
$emeasure\ M\ \{x \in space\ M.\ X\ x \in A \wedge Y\ x \in B\} = (\int^+ x.\ (\int^+ y.\ f\ (x, y) * indicator\ B\ y\ \partial T) * indicator\ A\ x\ \partial S)$
shows $distributed\ M\ (S \bigotimes_M T)\ (\lambda x.\ (X\ x,\ Y\ x))\ f$
⟨*proof*⟩

lemma (**in** *prob-space*) *distributed-swap*:
assumes $sigma\text{-}finite\text{-}measure\ S\ sigma\text{-}finite\text{-}measure\ T$
assumes Pxy: $distributed\ M\ (S \bigotimes_M T)\ (\lambda x.\ (X\ x,\ Y\ x))\ Pxy$
shows $distributed\ M\ (T \bigotimes_M S)\ (\lambda x.\ (Y\ x,\ X\ x))\ (\lambda(x, y).\ Pxy\ (y, x))$
⟨*proof*⟩

lemma (**in** *prob-space*) *distr-marginal1*:
assumes $sigma\text{-}finite\text{-}measure\ S\ sigma\text{-}finite\text{-}measure\ T$
assumes Pxy: $distributed\ M\ (S \bigotimes_M T)\ (\lambda x.\ (X\ x,\ Y\ x))\ Pxy$
defines $Px \equiv \lambda x.\ (\int^+ z.\ Pxy\ (x, z)\ \partial T)$
shows $distributed\ M\ S\ X\ Px$
⟨*proof*⟩

lemma (**in** *prob-space*) *distr-marginal2*:
assumes S: $sigma\text{-}finite\text{-}measure\ S$ **and** T: $sigma\text{-}finite\text{-}measure\ T$
assumes Pxy: $distributed\ M\ (S \bigotimes_M T)\ (\lambda x.\ (X\ x,\ Y\ x))\ Pxy$
shows $distributed\ M\ T\ Y\ (\lambda y.\ (\int^+ x.\ Pxy\ (x, y)\ \partial S))$
⟨*proof*⟩

lemma (**in** *prob-space*) *distributed-marginal-eq-joint1*:
assumes T: $sigma\text{-}finite\text{-}measure\ T$
assumes S: $sigma\text{-}finite\text{-}measure\ S$
assumes Px: $distributed\ M\ S\ X\ Px$
assumes Pxy: $distributed\ M\ (S \bigotimes_M T)\ (\lambda x.\ (X\ x,\ Y\ x))\ Pxy$
shows $AE\ x\ in\ S.\ Px\ x = (\int^+ y.\ Pxy\ (x, y)\ \partial T)$
⟨*proof*⟩

lemma (**in** *prob-space*) *distributed-marginal-eq-joint2*:
assumes T: $sigma\text{-}finite\text{-}measure\ T$
assumes S: $sigma\text{-}finite\text{-}measure\ S$
assumes Py: $distributed\ M\ T\ Y\ Py$
assumes Pxy: $distributed\ M\ (S \bigotimes_M T)\ (\lambda x.\ (X\ x,\ Y\ x))\ Pxy$
shows $AE\ y\ in\ T.\ Py\ y = (\int^+ x.\ Pxy\ (x, y)\ \partial S)$
⟨*proof*⟩

lemma (**in** *prob-space*) *distributed-joint-indep'*:
assumes S: $sigma\text{-}finite\text{-}measure\ S$ **and** T: $sigma\text{-}finite\text{-}measure\ T$
assumes $X[measurable]$: $distributed\ M\ S\ X\ Px$ **and** $Y[measurable]$: $distributed\ M\ T\ Y\ Py$

assumes *indep*: $distr\ M\ S\ X \bigotimes_M distr\ M\ T\ Y = distr\ M\ (S \bigotimes_M T)\ (\lambda x.\ (X\ x,\ Y\ x))$
shows $distributed\ M\ (S \bigotimes_M T)\ (\lambda x.\ (X\ x,\ Y\ x))\ (\lambda(x,\ y).\ Px\ x * Py\ y)$
⟨*proof*⟩

lemma *distributed-integrable*:
$distributed\ M\ N\ X\ f \Longrightarrow g \in borel\text{-}measurable\ N \Longrightarrow (\bigwedge x.\ x \in space\ N \Longrightarrow 0 \le f\ x) \Longrightarrow$
$integrable\ N\ (\lambda x.\ f\ x * g\ x) \longleftrightarrow integrable\ M\ (\lambda x.\ g\ (X\ x))$
⟨*proof*⟩

lemma *distributed-transform-integrable*:
assumes *Px*: $distributed\ M\ N\ X\ Px\ \bigwedge x.\ x \in space\ N \Longrightarrow 0 \le Px\ x$
assumes $distributed\ M\ P\ Y\ Py\ \bigwedge x.\ x \in space\ P \Longrightarrow 0 \le Py\ x$
assumes *Y*: $Y = (\lambda x.\ T\ (X\ x))$ **and** *T*: $T \in measurable\ N\ P$ **and** *f*: $f \in borel\text{-}measurable\ P$
shows $integrable\ P\ (\lambda x.\ Py\ x * f\ x) \longleftrightarrow integrable\ N\ (\lambda x.\ Px\ x * f\ (T\ x))$
⟨*proof*⟩

lemma *distributed-integrable-var*:
fixes $X :: 'a \Rightarrow real$
shows $distributed\ M\ lborel\ X\ (\lambda x.\ ennreal\ (f\ x)) \Longrightarrow (\bigwedge x.\ 0 \le f\ x) \Longrightarrow$
$integrable\ lborel\ (\lambda x.\ f\ x * x) \Longrightarrow integrable\ M\ X$
⟨*proof*⟩

lemma (**in** *prob-space*) *distributed-variance*:
fixes $f::real \Rightarrow real$
assumes *D*: $distributed\ M\ lborel\ X\ f$ **and** [*simp*]: $\bigwedge x.\ 0 \le f\ x$
shows $variance\ X = (\int x.\ x^2 * f\ (x + expectation\ X)\ \partial lborel)$
⟨*proof*⟩

lemma (**in** *prob-space*) *variance-affine*:
fixes $f::real \Rightarrow real$
assumes [*arith*]: $b \ne 0$
assumes *D*[*intro*]: $distributed\ M\ lborel\ X\ f$
assumes [*simp*]: $prob\text{-}space\ (density\ lborel\ f)$
assumes *I*[*simp*]: $integrable\ M\ X$
assumes *I2*[*simp*]: $integrable\ M\ (\lambda x.\ (X\ x)^2)$
shows $variance\ (\lambda x.\ a + b * X\ x) = b^2 * variance\ X$
⟨*proof*⟩

definition
$simple\text{-}distributed\ M\ X\ f \longleftrightarrow$
$(\forall x.\ 0 \le f\ x) \wedge$
$distributed\ M\ (count\text{-}space\ (X`space\ M))\ X\ (\lambda x.\ ennreal\ (f\ x)) \wedge$
$finite\ (X`space\ M)$

lemma *simple-distributed-nonneg*[*dest*]: $simple\text{-}distributed\ M\ X\ f \Longrightarrow 0 \le f\ x$
⟨*proof*⟩

lemma *simple-distributed*:
simple-distributed M X Px $\Longrightarrow$ *distributed M* (*count-space* (*X*'*space M*)) *X Px*
⟨*proof*⟩

lemma *simple-distributed-finite*[*dest*]: *simple-distributed M X P* $\Longrightarrow$ *finite* (*X*'*space M*)
⟨*proof*⟩

lemma (**in** *prob-space*) *distributed-simple-function-superset*:
assumes *X*: *simple-function M X* $\bigwedge x.\ x \in X$ ' *space M* $\Longrightarrow$ *P x* = *measure M* (*X* −' {*x*} ∩ *space M*)
assumes *A*: *X*'*space M* ⊆ *A finite A*
defines *S* ≡ *count-space A* **and** *P'* ≡ (λ*x*. *if x* ∈ *X*'*space M then P x else 0*)
shows *distributed M S X P'*
⟨*proof*⟩

lemma (**in** *prob-space*) *simple-distributedI*:
assumes *X*: *simple-function M X*
$\bigwedge x.\ 0 \leq P\ x$
$\bigwedge x.\ x \in X$ ' *space M* $\Longrightarrow$ *P x* = *measure M* (*X* −' {*x*} ∩ *space M*)
shows *simple-distributed M X P*
⟨*proof*⟩

lemma *simple-distributed-joint-finite*:
assumes *X*: *simple-distributed M* (λ*x*. (*X x*, *Y x*)) *Px*
shows *finite* (*X* ' *space M*) *finite* (*Y* ' *space M*)
⟨*proof*⟩

lemma *simple-distributed-joint2-finite*:
assumes *X*: *simple-distributed M* (λ*x*. (*X x*, *Y x*, *Z x*)) *Px*
shows *finite* (*X* ' *space M*) *finite* (*Y* ' *space M*) *finite* (*Z* ' *space M*)
⟨*proof*⟩

lemma *simple-distributed-simple-function*:
simple-distributed M X Px $\Longrightarrow$ *simple-function M X*
⟨*proof*⟩

lemma *simple-distributed-measure*:
simple-distributed M X P $\Longrightarrow$ *a* ∈ *X*'*space M* $\Longrightarrow$ *P a* = *measure M* (*X* −' {*a*} ∩ *space M*)
⟨*proof*⟩

lemma (**in** *prob-space*) *simple-distributed-joint*:
assumes *X*: *simple-distributed M* (λ*x*. (*X x*, *Y x*)) *Px*
defines *S* ≡ *count-space* (*X*'*space M*) $\bigotimes_M$ *count-space* (*Y*'*space M*)
defines *P* ≡ (λ*x*. *if x* ∈ (λ*x*. (*X x*, *Y x*))'*space M then Px x else 0*)
shows *distributed M S* (λ*x*. (*X x*, *Y x*)) *P*
⟨*proof*⟩

lemma (**in** *prob-space*) *simple-distributed-joint2*:
assumes *X*: *simple-distributed M* (λx. (*X x*, *Y x*, *Z x*)) *Px*
defines $S \equiv$ *count-space* (*X'space M*) $\bigotimes_M$ *count-space* (*Y'space M*) $\bigotimes_M$ *count-space* (*Z'space M*)
defines $P \equiv$ (λx. *if* $x \in$ (λx. (*X x*, *Y x*, *Z x*))*'space M then Px x else 0*)
shows *distributed M S* (λx. (*X x*, *Y x*, *Z x*)) *P*
⟨*proof*⟩

lemma (**in** *prob-space*) *simple-distributed-setsum-space*:
assumes *X*: *simple-distributed M X f*
shows *setsum f* (*X'space M*) = *1*
⟨*proof*⟩

lemma (**in** *prob-space*) *distributed-marginal-eq-joint-simple*:
assumes *Px*: *simple-function M X*
assumes *Py*: *simple-distributed M Y Py*
assumes *Pxy*: *simple-distributed M* (λx. (*X x*, *Y x*)) *Pxy*
assumes *y*: $y \in$ *Y'space M*
shows *Py y* = ($\sum x \in$*X'space M*. *if* (*x*, *y*) $\in$ (λx. (*X x*, *Y x*)) ' *space M then Pxy* (*x*, *y*) *else 0*)
⟨*proof*⟩

lemma *distributedI-real*:
fixes *f* :: *'a* $\Rightarrow$ *real*
assumes *gen*: *sets M1* = *sigma-sets* (*space M1*) *E* **and** *Int-stable E*
and *A*: *range A* $\subseteq$ *E* ($\bigcup$*i*::*nat*. *A i*) = *space M1* $\bigwedge$*i*. *emeasure* (*distr M M1 X*) (*A i*) $\neq \infty$
and *X*: $X \in$ *measurable M M1*
and *f*: $f \in$ *borel-measurable M1 AE x in M1*. *0* $\leq$ *f x*
and *eq*: $\bigwedge$*A*. $A \in E \Longrightarrow$ *emeasure M* (*X* −' *A* $\cap$ *space M*) = ($\int^+$ *x*. *f x* $*$ *indicator A x* ∂*M1*)
shows *distributed M M1 X f*
⟨*proof*⟩

lemma *distributedI-borel-atMost*:
fixes *f* :: *real* $\Rightarrow$ *real*
assumes [*measurable*]: $X \in$ *borel-measurable M*
and [*measurable*]: $f \in$ *borel-measurable borel* **and** *f*[*simp*]: *AE x in lborel*. *0* $\leq$ *f x*
and *g-eq*: $\bigwedge$*a*. ($\int^+$*x*. *f x* $*$ *indicator* {..*a*} *x* ∂*lborel*) = *ennreal* (*g a*)
and *M-eq*: $\bigwedge$*a*. *emeasure M* {*x*$\in$*space M*. *X x* $\leq$ *a*} = *ennreal* (*g a*)
shows *distributed M lborel X f*
⟨*proof*⟩

lemma (**in** *prob-space*) *uniform-distributed-params*:
assumes *X*: *distributed M MX X* (λx. *indicator A x* / *measure MX A*)
shows $A \in$ *sets MX measure MX A* $\neq$ *0*
⟨*proof*⟩

lemma *prob-space-uniform-measure*:
assumes *A*: *emeasure M A* $\neq$ *0 emeasure M A* $\neq \infty$
shows *prob-space* (*uniform-measure M A*)
⟨*proof*⟩

lemma *prob-space-uniform-count-measure*: *finite A* $\Longrightarrow$ *A* $\neq$ {} $\Longrightarrow$ *prob-space* (*uniform-count-measure A*)
⟨*proof*⟩

lemma (**in** *prob-space*) *measure-uniform-measure-eq-cond-prob*:
assumes [*measurable*]: *Measurable.pred M P Measurable.pred M Q*
shows $\mathcal{P}$(*x in uniform-measure M* {*x*∈*space M*. *Q x*}. *P x*) = $\mathcal{P}$(*x in M*. *P x* | *Q x*)
⟨*proof*⟩

lemma *prob-space-point-measure*:
finite S $\Longrightarrow$ ($\bigwedge$*s*. *s* ∈ *S* $\Longrightarrow$ *0* ≤ *p s*) $\Longrightarrow$ ($\sum$*s*∈*S*. *p s*) = *1* $\Longrightarrow$ *prob-space* (*point-measure S p*)
⟨*proof*⟩

lemma (**in** *prob-space*) *distr-pair-fst*: *distr* (*N* $\bigotimes_M$ *M*) *N fst* = *N*
⟨*proof*⟩

lemma (**in** *product-prob-space*) *distr-reorder*:
assumes *inj-on t J t* ∈ *J* → *K finite K*
shows *distr* (*PiM K M*) (Pi_M *J* (λ*x*. *M* (*t x*))) (λω. λ*n*∈*J*. ω (*t n*)) = *PiM J* (λ*x*. *M* (*t x*))
⟨*proof*⟩

lemma (**in** *product-prob-space*) *distr-restrict*:
J ⊆ *K* $\Longrightarrow$ *finite K* $\Longrightarrow$ (Π_M *i*∈*J*. *M i*) = *distr* (Π_M *i*∈*K*. *M i*) (Π_M *i*∈*J*. *M i*) (λ*f*. *restrict f J*)
⟨*proof*⟩

lemma (**in** *product-prob-space*) *emeasure-prod-emb*[*simp*]:
assumes *L*: *J* ⊆ *L finite L* **and** *X*: *X* ∈ *sets* (Pi_M *J M*)
shows *emeasure* (Pi_M *L M*) (*prod-emb L M J X*) = *emeasure* (Pi_M *J M*) *X*
⟨*proof*⟩

lemma *emeasure-distr-restrict*:
assumes *I* ⊆ *K* **and** *Q*[*measurable-cong*]: *sets Q* = *sets* (*PiM K M*) **and** *A*[*measurable*]: *A* ∈ *sets* (*PiM I M*)
shows *emeasure* (*distr Q* (*PiM I M*) (λω. *restrict* ω *I*)) *A* = *emeasure Q* (*prod-emb K M I A*)
⟨*proof*⟩

end

theory *Complete-Measure*
imports *Bochner-Integration Probability-Measure*
begin

definition
split-completion $M\ A\ p = (\mathit{if}\ A \in \mathit{sets}\ M\ \mathit{then}\ p = (A, \{\})\ \mathit{else}$
$\exists N'.\ A = \mathit{fst}\ p \cup \mathit{snd}\ p \land \mathit{fst}\ p \cap \mathit{snd}\ p = \{\} \land \mathit{fst}\ p \in \mathit{sets}\ M \land \mathit{snd}\ p \subseteq N'$
$\land N' \in \mathit{null\text{-}sets}\ M)$

definition
main-part $M\ A = \mathit{fst}\ (\mathit{Eps}\ (\mathit{split\text{-}completion}\ M\ A))$

definition
null-part $M\ A = \mathit{snd}\ (\mathit{Eps}\ (\mathit{split\text{-}completion}\ M\ A))$

definition *completion* :: *'a measure* ⇒ *'a measure* **where**
completion $M = \mathit{measure\text{-}of}\ (\mathit{space}\ M)\ \{\ S \cup N\ |S\ N\ N'.\ S \in \mathit{sets}\ M \land N' \in$
$\mathit{null\text{-}sets}\ M \land N \subseteq N'\ \}$
$(\mathit{emeasure}\ M \circ \mathit{main\text{-}part}\ M)$

lemma *completion-into-space*:
$\{\ S \cup N\ |S\ N\ N'.\ S \in \mathit{sets}\ M \land N' \in \mathit{null\text{-}sets}\ M \land N \subseteq N'\ \} \subseteq \mathit{Pow}\ (\mathit{space}$
$M)$
⟨*proof*⟩

lemma *space-completion*[*simp*]: $\mathit{space}\ (\mathit{completion}\ M) = \mathit{space}\ M$
⟨*proof*⟩

lemma *completionI*:
assumes $A = S \cup N\ N \subseteq N'\ N' \in \mathit{null\text{-}sets}\ M\ S \in \mathit{sets}\ M$
shows $A \in \{\ S \cup N\ |S\ N\ N'.\ S \in \mathit{sets}\ M \land N' \in \mathit{null\text{-}sets}\ M \land N \subseteq N'\ \}$
⟨*proof*⟩

lemma *completionE*:
assumes $A \in \{\ S \cup N\ |S\ N\ N'.\ S \in \mathit{sets}\ M \land N' \in \mathit{null\text{-}sets}\ M \land N \subseteq N'\ \}$
obtains $S\ N\ N'$ **where** $A = S \cup N\ N \subseteq N'\ N' \in \mathit{null\text{-}sets}\ M\ S \in \mathit{sets}\ M$
⟨*proof*⟩

lemma *sigma-algebra-completion*:
sigma-algebra $(\mathit{space}\ M)\ \{\ S \cup N\ |S\ N\ N'.\ S \in \mathit{sets}\ M \land N' \in \mathit{null\text{-}sets}\ M \land N$
$\subseteq N'\ \}$
(**is** *sigma-algebra* - *?A*)
⟨*proof*⟩

lemma *sets-completion*:
$\mathit{sets}\ (\mathit{completion}\ M) = \{\ S \cup N\ |S\ N\ N'.\ S \in \mathit{sets}\ M \land N' \in \mathit{null\text{-}sets}\ M \land N$
$\subseteq N'\ \}$
⟨*proof*⟩

lemma *sets-completionE*:
assumes $A \in sets\ (completion\ M)$
obtains $S\ N\ N'$ **where** $A = S \cup N\ N \subseteq N'\ N' \in null\text{-}sets\ M\ S \in sets\ M$
⟨*proof*⟩

lemma *sets-completionI*:
assumes $A = S \cup N\ N \subseteq N'\ N' \in null\text{-}sets\ M\ S \in sets\ M$
shows $A \in sets\ (completion\ M)$
⟨*proof*⟩

lemma *sets-completionI-sets*[*intro*, *simp*]:
$A \in sets\ M \Longrightarrow A \in sets\ (completion\ M)$
⟨*proof*⟩

lemma *null-sets-completion*:
assumes $N' \in null\text{-}sets\ M\ N \subseteq N'$ **shows** $N \in sets\ (completion\ M)$
⟨*proof*⟩

lemma *split-completion*:
assumes $A \in sets\ (completion\ M)$
shows *split-completion M A* (*main-part M A*, *null-part M A*)
⟨*proof*⟩

lemma
assumes $S \in sets\ (completion\ M)$
shows *main-part-sets*[*intro*, *simp*]: $main\text{-}part\ M\ S \in sets\ M$
and *main-part-null-part-Un*[*simp*]: $main\text{-}part\ M\ S \cup null\text{-}part\ M\ S = S$
and *main-part-null-part-Int*[*simp*]: $main\text{-}part\ M\ S \cap null\text{-}part\ M\ S = \{\}$
⟨*proof*⟩

lemma *main-part*[*simp*]: $S \in sets\ M \Longrightarrow main\text{-}part\ M\ S = S$
⟨*proof*⟩

lemma *null-part*:
assumes $S \in sets\ (completion\ M)$ **shows** $\exists N.\ N \in null\text{-}sets\ M \land null\text{-}part\ M\ S \subseteq N$
⟨*proof*⟩

lemma *null-part-sets*[*intro*, *simp*]:
assumes $S \in sets\ M$ **shows** $null\text{-}part\ M\ S \in sets\ M$ $emeasure\ M\ (null\text{-}part\ M\ S) = 0$
⟨*proof*⟩

lemma *emeasure-main-part-UN*:
fixes $S :: nat \Rightarrow 'a\ set$
assumes $range\ S \subseteq sets\ (completion\ M)$
shows $emeasure\ M\ (main\text{-}part\ M\ (\bigcup i.\ (S\ i))) = emeasure\ M\ (\bigcup i.\ main\text{-}part\ M\ (S\ i))$

⟨*proof*⟩

lemma *emeasure-completion*[*simp*]:
assumes *S*: $S \in$ *sets* (*completion M*) **shows** *emeasure* (*completion M*) $S =$ *emeasure M* (*main-part M S*)
⟨*proof*⟩

lemma *emeasure-completion-UN*:
range S $\subseteq$ *sets* (*completion M*) $\Longrightarrow$
emeasure (*completion M*) ($\bigcup i{::}nat.\ (S\ i)$) = *emeasure M* ($\bigcup i.$ *main-part M* ($S\ i$))
⟨*proof*⟩

lemma *emeasure-completion-Un*:
assumes *S*: $S \in$ *sets* (*completion M*) **and** *T*: $T \in$ *sets* (*completion M*)
shows *emeasure* (*completion M*) ($S \cup T$) = *emeasure M* (*main-part M S* $\cup$ *main-part M T*)
⟨*proof*⟩

lemma *sets-completionI-sub*:
assumes *N*: $N' \in$ *null-sets M* $N \subseteq N'$
shows $N \in$ *sets* (*completion M*)
⟨*proof*⟩

lemma *completion-ex-simple-function*:
assumes *f*: *simple-function* (*completion M*) *f*
shows $\exists f'$. *simple-function M* $f' \wedge$ (*AE x in M*. $f\ x = f'\ x$)
⟨*proof*⟩

lemma *completion-ex-borel-measurable*:
fixes $g :: 'a \Rightarrow$ *ennreal*
assumes *g*: $g \in$ *borel-measurable* (*completion M*)
shows $\exists g' \in$ *borel-measurable M*. (*AE x in M*. $g\ x = g'\ x$)
⟨*proof*⟩

lemma (**in** *prob-space*) *prob-space-completion*: *prob-space* (*completion M*)
⟨*proof*⟩

lemma *null-sets-completionI*: $N \in$ *null-sets M* $\Longrightarrow N \in$ *null-sets* (*completion M*)
⟨*proof*⟩

lemma *AE-completion*: (*AE x in M*. *P x*) $\Longrightarrow$ (*AE x in completion M*. *P x*)
⟨*proof*⟩

lemma *null-sets-completion-iff*: $N \in$ *sets M* $\Longrightarrow N \in$ *null-sets* (*completion M*) $\longleftrightarrow N \in$ *null-sets M*
⟨*proof*⟩

lemma *AE-completion-iff*: $\{x \in$ *space M*. *P x*$\} \in$ *sets M* $\Longrightarrow$ (*AE x in M*. *P x*)

$\longleftrightarrow$ (*AE x in completion M. P x*)
⟨*proof*⟩

end

13 Finite Maps

theory *Fin-Map*
imports *Finite-Product-Measure*
begin

Auxiliary type that is instantiated to *polish-space*, needed for the proof of projective limit. *extensional* functions are used for the representation in order to stay close to the developments of (finite) products Pi_E and their sigma-algebra Pi_M.

typedef (*'i*, *'a*) *finmap* ((- $\Rightarrow_F$ /-) [*22*, *21*] *21*) =
{(*I*::*'i set*, *f*::*'i* ⇒ *'a*). *finite I* ∧ *f* ∈ *extensional I*} ⟨*proof*⟩

13.1 Domain and Application

definition *domain* **where** *domain P* = *fst* (*Rep-finmap P*)

lemma *finite-domain*[*simp*, *intro*]: *finite* (*domain P*)
⟨*proof*⟩

definition *proj* (*'*((-)*'*)$_F$ [*0*] *1000*) **where** *proj P i* = *snd* (*Rep-finmap P*) *i*

declare [[*coercion proj*]]

lemma *extensional-proj*[*simp*, *intro*]: $(P)_F$ ∈ *extensional* (*domain P*)
⟨*proof*⟩

lemma *proj-undefined*[*simp*, *intro*]: *i* ∉ *domain P* ⟹ *P i* = *undefined*
⟨*proof*⟩

lemma *finmap-eq-iff*: *P* = *Q* ⟷ (*domain P* = *domain Q* ∧ (∀ *i*∈*domain P*. *P i* = *Q i*))
⟨*proof*⟩

13.2 Countable Finite Maps

instance *finmap* :: (*countable*, *countable*) *countable*
⟨*proof*⟩

13.3 Constructor of Finite Maps

definition *finmap-of inds f* = *Abs-finmap* (*inds*, *restrict f inds*)

lemma *proj-finmap-of*[*simp*]:

assumes *finite inds*
shows $(\textit{finmap-of inds } f)_F = \textit{restrict } f \textit{ inds}$
⟨*proof*⟩

lemma *domain-finmap-of* [*simp*]:
assumes *finite inds*
shows *domain* (*finmap-of inds f*) = *inds*
⟨*proof*⟩

lemma *finmap-of-eq-iff* [*simp*]:
assumes *finite i finite j*
shows $\textit{finmap-of } i\ m = \textit{finmap-of } j\ n \longleftrightarrow i = j \wedge (\forall k{\in}i.\ m\ k{=}\ n\ k)$
⟨*proof*⟩

lemma *finmap-of-inj-on-extensional-finite*:
assumes *finite K*
assumes $S \subseteq \textit{extensional } K$
shows *inj-on* (*finmap-of K*) *S*
⟨*proof*⟩

13.4 Product set of Finite Maps

This is *Pi* for Finite Maps, most of this is copied

definition $Pi' :: 'i\ set \Rightarrow ('i \Rightarrow 'a\ set) \Rightarrow ('i \Rightarrow_F 'a)\ set$ **where**
$Pi'\ I\ A = \{\ P.\ \textit{domain}\ P = I \wedge (\forall i.\ i \in I \longrightarrow (P)_F\ i \in A\ i)\ \}$

syntax
$\textit{-Pi}' :: [\textit{pttrn}, 'a\ set, 'b\ set] => ('a => 'b)\ set\ \ ((3\Pi'\ \text{-}{\in}\text{-./ -})\ \ 10)$
translations
$\Pi'\ x{\in}A.\ B == \textit{CONST}\ Pi'\ A\ (\lambda x.\ B)$

13.4.1 Basic Properties of Pi'

lemma $Pi'\text{-}I[\textit{intro}!]$: $\textit{domain}\ f = A \Longrightarrow (\bigwedge x.\ x \in A \Longrightarrow f\ x \in B\ x) \Longrightarrow f \in Pi'\ A\ B$
⟨*proof*⟩

lemma $Pi'\text{-}I'[\textit{simp}]$: $\textit{domain}\ f = A \Longrightarrow (\bigwedge x.\ x \in A \longrightarrow f\ x \in B\ x) \Longrightarrow f \in Pi'\ A\ B$
⟨*proof*⟩

lemma $Pi'\text{-}\textit{mem}$: $f{\in}\ Pi'\ A\ B \Longrightarrow x \in A \Longrightarrow f\ x \in B\ x$
⟨*proof*⟩

lemma $Pi'\text{-}\textit{iff}$: $f \in Pi'\ I\ X \longleftrightarrow \textit{domain}\ f = I \wedge (\forall i{\in}I.\ f\ i \in X\ i)$
⟨*proof*⟩

lemma $Pi'E$ [*elim*]:

$f \in Pi'\ A\ B \Longrightarrow (f\ x \in B\ x \Longrightarrow domain\ f = A \Longrightarrow Q) \Longrightarrow (x \notin A \Longrightarrow Q) \Longrightarrow Q$
⟨*proof*⟩

lemma *in-Pi'-cong*:
$domain\ f = domain\ g \Longrightarrow (\bigwedge w.\ w \in A \Longrightarrow f\ w = g\ w) \Longrightarrow f \in Pi'\ A\ B \longleftrightarrow g \in Pi'\ A\ B$
⟨*proof*⟩

lemma *Pi'-eq-empty*[*simp*]:
assumes *finite A* **shows** $(Pi'\ A\ B) = \{\} \longleftrightarrow (\exists x{\in}A.\ B\ x = \{\})$
⟨*proof*⟩

lemma *Pi'-mono*: $(\bigwedge x.\ x \in A \Longrightarrow B\ x \subseteq C\ x) \Longrightarrow Pi'\ A\ B \subseteq Pi'\ A\ C$
⟨*proof*⟩

lemma *Pi-Pi'*: $finite\ A \Longrightarrow (Pi_E\ A\ B) = proj\ `\ Pi'\ A\ B$
⟨*proof*⟩

13.5 Topological Space of Finite Maps

instantiation *finmap* :: (*type*, *topological-space*) *topological-space*
begin

definition *open-finmap* :: $('a \Rightarrow_F 'b)\ set \Rightarrow bool$ **where**
[*code del*]: $open\text{-}finmap = generate\text{-}topology\ \{Pi'\ a\ b | a\ b.\ \forall i{\in}a.\ open\ (b\ i)\}$

lemma *open-Pi'I*: $(\bigwedge i.\ i \in I \Longrightarrow open\ (A\ i)) \Longrightarrow open\ (Pi'\ I\ A)$
⟨*proof*⟩

instance ⟨*proof*⟩

end

lemma *open-restricted-space*:
shows $open\ \{m.\ P\ (domain\ m)\}$
⟨*proof*⟩

lemma *closed-restricted-space*:
shows $closed\ \{m.\ P\ (domain\ m)\}$
⟨*proof*⟩

lemma *tendsto-proj*: $((\lambda x.\ x) \longrightarrow a)\ F \Longrightarrow ((\lambda x.\ (x)_F\ i) \longrightarrow (a)_F\ i)\ F$
⟨*proof*⟩

lemma *continuous-proj*:
shows $continuous\text{-}on\ s\ (\lambda x.\ (x)_F\ i)$
⟨*proof*⟩

instance *finmap* :: (*type*, *first-countable-topology*) *first-countable-topology*
⟨*proof*⟩

13.6 Metric Space of Finite Maps

instantiation *finmap* :: (*type*, *metric-space*) *dist*
begin

definition *dist-finmap* **where**
dist P Q = *Max* (*range* (λi. *dist* ($(P)_F$ *i*) ($(Q)_F$ *i*))) + (*if domain P* = *domain Q then 0 else 1*)

instance ⟨*proof*⟩
end

instantiation *finmap* :: (*type*, *metric-space*) *uniformity-dist*
begin

definition [*code del*]:
(*uniformity* :: (('*a*, '*b*) *finmap* × ('*a*, '*b*) *finmap*) *filter*) =
(*INF e*:{*0* <..}. *principal* {(*x*, *y*). *dist x y* < *e*})

instance
⟨*proof*⟩
end

declare *uniformity-Abort*[**where** '*a*=('*a*, '*b*::*metric-space*) *finmap*, *code*]

instantiation *finmap* :: (*type*, *metric-space*) *metric-space*
begin

lemma *finite-proj-image'*: $x \notin$ *domain P* $\Longrightarrow$ *finite* ($(P)_F$ ‘ *S*)
⟨*proof*⟩

lemma *finite-proj-image*: *finite* ($(P)_F$ ‘ *S*)
⟨*proof*⟩

lemma *finite-proj-diag*: *finite* ((λi. *d* ($(P)_F$ *i*) ($(Q)_F$ *i*)) ‘ *S*)
⟨*proof*⟩

lemma *dist-le-1-imp-domain-eq*:
shows *dist P Q* < *1* $\Longrightarrow$ *domain P* = *domain Q*
⟨*proof*⟩

lemma *dist-proj*:
shows *dist* ($(x)_F$ *i*) ($(y)_F$ *i*) ≤ *dist x y*
⟨*proof*⟩

lemma *dist-finmap-lessI*:

assumes *domain P = domain Q*
assumes *0 < e*
assumes $\bigwedge i.\ i \in domain\ P \Longrightarrow dist\ (P\ i)\ (Q\ i) < e$
shows *dist P Q < e*
⟨*proof*⟩

instance
⟨*proof*⟩

end

13.7 Complete Space of Finite Maps

lemma *tendsto-finmap*:
fixes $f::nat \Rightarrow ('i \Rightarrow_F ('a::metric\text{-}space))$
assumes *ind-f*: $\bigwedge n.\ domain\ (f\ n) = domain\ g$
assumes *proj-g*: $\bigwedge i.\ i \in domain\ g \Longrightarrow (\lambda n.\ (f\ n)\ i) \longrightarrow g\ i$
shows $f \longrightarrow g$
⟨*proof*⟩

instance *finmap* :: (*type*, *complete-space*) *complete-space*
⟨*proof*⟩

13.8 Second Countable Space of Finite Maps

instantiation *finmap* :: (*countable*, *second-countable-topology*) *second-countable-topology*
begin

definition *basis-proj*::*'b set set*
where *basis-proj* = (*SOME B. countable B* ∧ *topological-basis B*)

lemma *countable-basis-proj*: *countable basis-proj* **and** *basis-proj*: *topological-basis basis-proj*
⟨*proof*⟩

definition *basis-finmap*::$('a \Rightarrow_F 'b)$ *set set*
where *basis-finmap* = {*Pi' I S*|*I S. finite I* ∧ (∀ *i* ∈ *I. S i* ∈ *basis-proj*)}

lemma *in-basis-finmapI*:
assumes *finite I* **assumes** $\bigwedge i.\ i \in I \Longrightarrow S\ i \in basis\text{-}proj$
shows *Pi' I S* ∈ *basis-finmap*
⟨*proof*⟩

lemma *basis-finmap-eq*:
assumes *basis-proj* ≠ {}
shows *basis-finmap* = (λ*f. Pi'* (*domain f*) (λ*i. from-nat-into basis-proj* $((f)_F$ *i*))) ‘
(*UNIV*::$('a \Rightarrow_F nat)$ *set*) (**is** - = *?f* ‘ -)
⟨*proof*⟩

lemma *basis-finmap-eq-empty*: *basis-proj* = {} $\Longrightarrow$ *basis-finmap* = {*Pi'* {} *undefined*}
⟨*proof*⟩

lemma *countable-basis-finmap*: *countable basis-finmap*
⟨*proof*⟩

lemma *finmap-topological-basis*:
topological-basis basis-finmap
⟨*proof*⟩

lemma *range-enum-basis-finmap-imp-open*:
assumes $x \in$ *basis-finmap*
shows *open x*
⟨*proof*⟩

instance ⟨*proof*⟩

end

13.9 Polish Space of Finite Maps

instance *finmap* :: (*countable*, *polish-space*) *polish-space* ⟨*proof*⟩

13.10 Product Measurable Space of Finite Maps

definition *PiF I M* ≡
sigma (⋃ *J* ∈ *I*. (Π′ *j*∈*J*. *space* (*M j*))) {(Π′ *j*∈*J*. *X j*) |*X J*. *J* ∈ *I* ∧ *X* ∈ (Π *j*∈*J*. *sets* (*M j*))}

abbreviation
Pi_F *I M* ≡ *PiF I M*

syntax
-PiF :: *pttrn* ⇒ *'i set* ⇒ *'a measure* ⇒ (*'i* => *'a*) *measure* ((*3*Π_F -∈-./ -) *10*)
translations
Π_F *x*∈*I*. *M* == *CONST PiF I* (%*x*. *M*)

lemma *PiF-gen-subset*: {(Π′ *j*∈*J*. *X j*) |*X J*. *J* ∈ *I* ∧ *X* ∈ (Π *j*∈*J*. *sets* (*M j*))} ⊆
Pow (⋃ *J* ∈ *I*. (Π′ *j*∈*J*. *space* (*M j*)))
⟨*proof*⟩

lemma *space-PiF*: *space* (*PiF I M*) = (⋃ *J* ∈ *I*. (Π′ *j*∈*J*. *space* (*M j*)))
⟨*proof*⟩

lemma *sets-PiF*:
sets (*PiF I M*) = *sigma-sets* (⋃ *J* ∈ *I*. (Π′ *j*∈*J*. *space* (*M j*)))
{(Π′ *j*∈*J*. *X j*) |*X J*. *J* ∈ *I* ∧ *X* ∈ (Π *j*∈*J*. *sets* (*M j*))}
⟨*proof*⟩

lemma *sets-PiF-singleton*:
sets (*PiF* {*I*} *M*) = *sigma-sets* (Π' *j*∈*I*. *space* (*M j*))
{(Π' *j*∈*I*. *X j*) |*X*. *X* ∈ (Π *j*∈*I*. *sets* (*M j*))}
⟨*proof*⟩

lemma *in-sets-PiFI*:
assumes *X* = (*Pi' J S*) *J* ∈ *I* ⋀*i*. *i*∈*J* ⟹ *S i* ∈ *sets* (*M i*)
shows *X* ∈ *sets* (*PiF I M*)
⟨*proof*⟩

lemma *product-in-sets-PiFI*:
assumes *J* ∈ *I* ⋀*i*. *i*∈*J* ⟹ *S i* ∈ *sets* (*M i*)
shows (*Pi' J S*) ∈ *sets* (*PiF I M*)
⟨*proof*⟩

lemma *singleton-space-subset-in-sets*:
fixes *J*
assumes *J* ∈ *I*
assumes *finite J*
shows *space* (*PiF* {*J*} *M*) ∈ *sets* (*PiF I M*)
⟨*proof*⟩

lemma *singleton-subspace-set-in-sets*:
assumes *A*: *A* ∈ *sets* (*PiF* {*J*} *M*)
assumes *finite J*
assumes *J* ∈ *I*
shows *A* ∈ *sets* (*PiF I M*)
⟨*proof*⟩

lemma *finite-measurable-singletonI*:
assumes *finite I*
assumes ⋀*J*. *J* ∈ *I* ⟹ *finite J*
assumes *MN*: ⋀*J*. *J* ∈ *I* ⟹ *A* ∈ *measurable* (*PiF* {*J*} *M*) *N*
shows *A* ∈ *measurable* (*PiF I M*) *N*
⟨*proof*⟩

lemma *countable-finite-comprehension*:
fixes *f* :: *'a*::*countable set* ⇒ -
assumes ⋀*s*. *P s* ⟹ *finite s*
assumes ⋀*s*. *P s* ⟹ *f s* ∈ *sets M*
shows ⋃{*f s*|*s*. *P s*} ∈ *sets M*
⟨*proof*⟩

lemma *space-subset-in-sets*:
fixes *J*::*'a*::*countable set set*
assumes *J* ⊆ *I*
assumes ⋀*j*. *j* ∈ *J* ⟹ *finite j*
shows *space* (*PiF J M*) ∈ *sets* (*PiF I M*)

⟨*proof*⟩

lemma *subspace-set-in-sets*:
fixes $J::'a::countable\ set\ set$
assumes A: $A \in sets\ (PiF\ J\ M)$
assumes $J \subseteq I$
assumes $\bigwedge j.\ j \in J \Longrightarrow finite\ j$
shows $A \in sets\ (PiF\ I\ M)$
⟨*proof*⟩

lemma *countable-measurable-PiFI*:
fixes $I::'a::countable\ set\ set$
assumes MN: $\bigwedge J.\ J \in I \Longrightarrow finite\ J \Longrightarrow A \in measurable\ (PiF\ \{J\}\ M)\ N$
shows $A \in measurable\ (PiF\ I\ M)\ N$
⟨*proof*⟩

lemma *measurable-PiF*:
assumes f: $\bigwedge x.\ x \in space\ N \Longrightarrow domain\ (f\ x) \in I \wedge (\forall\, i{\in}domain\ (f\ x).\ (f\ x)\ i \in space\ (M\ i))$
assumes S: $\bigwedge J\ S.\ J \in I \Longrightarrow (\bigwedge i.\ i \in J \Longrightarrow S\ i \in sets\ (M\ i)) \Longrightarrow$
$f\ -`\ (Pi'\ J\ S) \cap space\ N \in sets\ N$
shows $f \in measurable\ N\ (PiF\ I\ M)$
⟨*proof*⟩

lemma *restrict-sets-measurable*:
assumes A: $A \in sets\ (PiF\ I\ M)$ **and** $J \subseteq I$
shows $A \cap \{m.\ domain\ m \in J\} \in sets\ (PiF\ J\ M)$
⟨*proof*⟩

lemma *measurable-finmap-of*:
assumes f: $\bigwedge i.\ (\exists\, x \in space\ N.\ i \in J\ x) \Longrightarrow (\lambda x.\ f\ x\ i) \in measurable\ N\ (M\ i)$
assumes J: $\bigwedge x.\ x \in space\ N \Longrightarrow J\ x \in I\ \bigwedge x.\ x \in space\ N \Longrightarrow finite\ (J\ x)$
assumes JN: $\bigwedge S.\ \{x.\ J\ x = S\} \cap space\ N \in sets\ N$
shows $(\lambda x.\ finmap\text{-}of\ (J\ x)\ (f\ x)) \in measurable\ N\ (PiF\ I\ M)$
⟨*proof*⟩

lemma *measurable-PiM-finmap-of*:
assumes *finite J*
shows $finmap\text{-}of\ J \in measurable\ (Pi_M\ J\ M)\ (PiF\ \{J\}\ M)$
⟨*proof*⟩

lemma *proj-measurable-singleton*:
assumes $A \in sets\ (M\ i)$
shows $(\lambda x.\ (x)_F\ i)\ -`\ A \cap space\ (PiF\ \{I\}\ M) \in sets\ (PiF\ \{I\}\ M)$
⟨*proof*⟩

lemma *measurable-proj-singleton*:
assumes $i \in I$
shows $(\lambda x.\ (x)_F\ i) \in measurable\ (PiF\ \{I\}\ M)\ (M\ i)$

⟨proof⟩

lemma *measurable-proj-countable*:
fixes $I::'a::countable\ set\ set$
assumes $y \in space\ (M\ i)$
shows $(\lambda x.\ if\ i \in domain\ x\ then\ (x)_F\ i\ else\ y) \in measurable\ (PiF\ I\ M)\ (M\ i)$
⟨proof⟩

lemma *measurable-restrict-proj*:
assumes $J \in II$ *finite* J
shows *finmap-of* $J \in measurable\ (PiM\ J\ M)\ (PiF\ II\ M)$
⟨proof⟩

lemma *measurable-proj-PiM*:
fixes $J\ K\ ::'a::countable\ set$ **and** $I::'a\ set\ set$
assumes *finite* $J\ J \in I$
assumes $x \in space\ (PiM\ J\ M)$
shows $proj \in measurable\ (PiF\ \{J\}\ M)\ (PiM\ J\ M)$
⟨proof⟩

lemma *space-PiF-singleton-eq-product*:
assumes *finite* I
shows $space\ (PiF\ \{I\}\ M) = (\Pi'\ i{\in}I.\ space\ (M\ i))$
⟨proof⟩

adapted from $sets\ (Pi_M\ ?I\ ?M) = sigma\text{-}sets\ (\Pi_E\ i{\in}?I.\ space\ (?M\ i))\ \{\{f \in \Pi_E\ i{\in}?I.\ space\ (?M\ i).\ f\ i \in A\}\ |i\ A.\ i \in ?I \wedge A \in sets\ (?M\ i)\}$

lemma *sets-PiF-single*:
assumes *finite* $I\ I \neq \{\}$
shows $sets\ (PiF\ \{I\}\ M) =$
$sigma\text{-}sets\ (\Pi'\ i{\in}I.\ space\ (M\ i))$
$\{\{f{\in}\Pi'\ i{\in}I.\ space\ (M\ i).\ f\ i \in A\}\ |\ i\ A.\ i \in I \wedge A \in sets\ (M\ i)\}$
(**is** $- = sigma\text{-}sets\ ?\Omega\ ?R$)
⟨proof⟩

adapted from $(\bigwedge i.\ i \in ?I \Longrightarrow ?A\ i = ?B\ i) \Longrightarrow Pi_E\ ?I\ ?A = Pi_E\ ?I\ ?B$

lemma *Pi'-cong*:
assumes *finite* I
assumes $\bigwedge i.\ i \in I \Longrightarrow f\ i = g\ i$
shows $Pi'\ I\ f = Pi'\ I\ g$
⟨proof⟩

adapted from $[\![finite\ ?I;\ \bigwedge i\ n\ m.\ [\![i \in ?I;\ n \le m]\!] \Longrightarrow ?A\ n\ i \subseteq ?A\ m\ i]\!] \Longrightarrow (\bigcup_n\ Pi\ ?I\ (?A\ n)) = (\Pi\ i{\in}?I.\ \bigcup_n\ ?A\ n\ i)$

lemma *Pi'-UN*:
fixes $A :: nat \Rightarrow 'i \Rightarrow 'a\ set$
assumes *finite* I
assumes *mono*: $\bigwedge i\ n\ m.\ i \in I \Longrightarrow n \le m \Longrightarrow A\ n\ i \subseteq A\ m\ i$

shows $(\bigcup n.\ Pi'\ I\ (A\ n)) = Pi'\ I\ (\lambda i.\ \bigcup n.\ A\ n\ i)$
$\langle proof \rangle$

adapted from $[\![\bigwedge i.\ i \in ?I \Longrightarrow \exists S \subseteq ?E\ i.\ countable\ S \wedge ?\Omega\ i = \bigcup S;\ \bigwedge i.\ i \in ?I \Longrightarrow ?E\ i \subseteq Pow\ (?\Omega\ i);\ \bigwedge j.\ j \in ?J \Longrightarrow finite\ j;\ \bigcup ?J = ?I]\!] \Longrightarrow sets\ (Pi_M\ ?I\ (\lambda i.\ sigma\ (?\Omega\ i)\ (?E\ i))) = sets\ (sigma\ (Pi_E\ ?I\ ?\Omega)\ \{\{f \in Pi_E\ ?I\ ?\Omega.\ \forall i{\in}j.\ f\ i \in A\ i\}\ |A\ j.\ j \in ?J \wedge A \in Pi\ j\ ?E\})$

lemma *sigma-fprod-algebra-sigma-eq*:
fixes $E :: 'i \Rightarrow 'a\ set\ set$ **and** $S :: 'i \Rightarrow nat \Rightarrow 'a\ set$
assumes [*simp*]: $finite\ I\ I \neq \{\}$
and *S-union*: $\bigwedge i.\ i \in I \Longrightarrow (\bigcup j.\ S\ i\ j) = space\ (M\ i)$
and *S-in-E*: $\bigwedge i.\ i \in I \Longrightarrow range\ (S\ i) \subseteq E\ i$
assumes *E-closed*: $\bigwedge i.\ i \in I \Longrightarrow E\ i \subseteq Pow\ (space\ (M\ i))$
and *E-generates*: $\bigwedge i.\ i \in I \Longrightarrow sets\ (M\ i) = sigma\text{-}sets\ (space\ (M\ i))\ (E\ i)$
defines $P == \{\ Pi'\ I\ F\ |\ F.\ \forall i{\in}I.\ F\ i \in E\ i\ \}$
shows $sets\ (PiF\ \{I\}\ M) = sigma\text{-}sets\ (space\ (PiF\ \{I\}\ M))\ P$
$\langle proof \rangle$

lemma *product-open-generates-sets-PiF-single*:
assumes $I \neq \{\}$
assumes [*simp*]: *finite I*
shows $sets\ (PiF\ \{I\}\ (\lambda\text{-}.\ borel::'b::second\text{-}countable\text{-}topology\ measure)) =$
$sigma\text{-}sets\ (space\ (PiF\ \{I\}\ (\lambda\text{-}.\ borel)))\ \{Pi'\ I\ F\ |F.\ (\forall i{\in}I.\ F\ i \in Collect\ open)\}$
$\langle proof \rangle$

lemma *finmap-UNIV* [*simp*]: $(\bigcup J{\in}Collect\ finite.\ \Pi' j{\in}J.\ UNIV) = UNIV$ $\langle proof \rangle$

lemma *borel-eq-PiF-borel*:
shows $(borel :: ('i::countable \Rightarrow_F 'a::polish\text{-}space)\ measure) =$
$PiF\ (Collect\ finite)\ (\lambda\text{-}.\ borel :: 'a\ measure)$
$\langle proof \rangle$

13.11 Isomorphism between Functions and Finite Maps

lemma *measurable-finmap-compose*:
shows $(\lambda m.\ compose\ J\ m\ f) \in measurable\ (PiM\ (f\ `\ J)\ (\lambda\text{-}.\ M))\ (PiM\ J\ (\lambda\text{-}.\ M))$
$\langle proof \rangle$

lemma *measurable-compose-inv*:
assumes *inj*: $\bigwedge j.\ j \in J \Longrightarrow f'\ (f\ j) = j$
shows $(\lambda m.\ compose\ (f\ `\ J)\ m\ f') \in measurable\ (PiM\ J\ (\lambda\text{-}.\ M))\ (PiM\ (f\ `\ J)\ (\lambda\text{-}.\ M))$
$\langle proof \rangle$

locale *function-to-finmap* =
fixes $J::'a\ set$ **and** $f :: 'a \Rightarrow 'b::countable$ **and** f'
assumes [*simp*]: *finite J*

assumes *inv*: $i \in J \Longrightarrow f'\ (f\ i) = i$
begin

to measure finmaps

definition $fm = (\mathit{finmap\text{-}of}\ (f\ `\ J))\ o\ (\lambda g.\ \mathit{compose}\ (f\ `\ J)\ g\ f')$

lemma *domain-fm*[*simp*]: $\mathit{domain}\ (fm\ x) = f\ `\ J$
⟨*proof*⟩

lemma *fm-restrict*[*simp*]: $fm\ (\mathit{restrict}\ y\ J) = fm\ y$
⟨*proof*⟩

lemma *fm-product*:
assumes $\bigwedge i.\ \mathit{space}\ (M\ i) = \mathit{UNIV}$
shows $fm\ -`\ Pi'\ (f\ `\ J)\ S \cap \mathit{space}\ (Pi_M\ J\ M) = (\Pi_E\ j \in J.\ S\ (f\ j))$
⟨*proof*⟩

lemma *fm-measurable*:
assumes $f\ `\ J \in N$
shows $fm \in \mathit{measurable}\ (Pi_M\ J\ (\lambda\text{-}.\ M))\ (Pi_F\ N\ (\lambda\text{-}.\ M))$
⟨*proof*⟩

lemma *proj-fm*:
assumes $x \in J$
shows $fm\ m\ (f\ x) = m\ x$
⟨*proof*⟩

lemma *inj-on-compose-f'*: $\mathit{inj\text{-}on}\ (\lambda g.\ \mathit{compose}\ (f\ `\ J)\ g\ f')\ (\mathit{extensional}\ J)$
⟨*proof*⟩

lemma *inj-on-fm*:
assumes $\bigwedge i.\ \mathit{space}\ (M\ i) = \mathit{UNIV}$
shows $\mathit{inj\text{-}on}\ fm\ (\mathit{space}\ (Pi_M\ J\ M))$
⟨*proof*⟩

to measure functions

definition $mf = (\lambda g.\ \mathit{compose}\ J\ g\ f)\ o\ \mathit{proj}$

lemma *mf-fm*:
assumes $x \in \mathit{space}\ (Pi_M\ J\ (\lambda\text{-}.\ M))$
shows $mf\ (fm\ x) = x$
⟨*proof*⟩

lemma *mf-measurable*:
assumes $\mathit{space}\ M = \mathit{UNIV}$
shows $mf \in \mathit{measurable}\ (PiF\ \{f\ `\ J\}\ (\lambda\text{-}.\ M))\ (PiM\ J\ (\lambda\text{-}.\ M))$
⟨*proof*⟩

lemma *fm-image-measurable*:

assumes *space M = UNIV*
assumes *X* ∈ *sets* (*Pi*$_M$ *J* (λ-. *M*))
shows *fm* ‘ *X* ∈ *sets* (*PiF* {*f* ‘ *J*} (λ-. *M*))
⟨*proof*⟩

lemma *fm-image-measurable-finite*:
assumes *space M = UNIV*
assumes *X* ∈ *sets* (*Pi*$_M$ *J* (λ-. *M*::′*c measure*))
shows *fm* ‘ *X* ∈ *sets* (*PiF* (*Collect finite*) (λ-. *M*::′*c measure*))
⟨*proof*⟩

measure on finmaps

definition *mapmeasure M N = distr M* (*PiF* (*Collect finite*) *N*) (*fm*)

lemma *sets-mapmeasure*[*simp*]: *sets* (*mapmeasure M N*) = *sets* (*PiF* (*Collect finite*) *N*)
⟨*proof*⟩

lemma *space-mapmeasure*[*simp*]: *space* (*mapmeasure M N*) = *space* (*PiF* (*Collect finite*) *N*)
⟨*proof*⟩

lemma *mapmeasure-PiF*:
assumes *s1*: *space M = space* (*Pi*$_M$ *J* (λ-. *N*))
assumes *s2*: *sets M = sets* (*Pi*$_M$ *J* (λ-. *N*))
assumes *space N = UNIV*
assumes *X* ∈ *sets* (*PiF* (*Collect finite*) (λ-. *N*))
shows *emeasure* (*mapmeasure M* (λ-. *N*)) *X = emeasure M* ((*fm* −‘ *X* ∩ *extensional J*))
⟨*proof*⟩

lemma *mapmeasure-PiM*:
fixes *N*::′*c measure*
assumes *s1*: *space M = space* (*Pi*$_M$ *J* (λ-. *N*))
assumes *s2*: *sets M* = (*Pi*$_M$ *J* (λ-. *N*))
assumes *N*: *space N = UNIV*
assumes *X*: *X* ∈ *sets M*
shows *emeasure M X = emeasure* (*mapmeasure M* (λ-. *N*)) (*fm* ‘ *X*)
⟨*proof*⟩

end

end

14 Regularity of Measures

theory *Regularity*
imports *Measure-Space Borel-Space*
begin

lemma
fixes $M::'a::\{second\text{-}countable\text{-}topology,\ complete\text{-}space\}\ measure$
assumes sb: $sets\ M = sets\ borel$
assumes $emeasure\ M\ (space\ M) \neq \infty$
assumes $B \in sets\ borel$
shows $inner\text{-}regular$: $emeasure\ M\ B =$
$(SUP\ K : \{K.\ K \subseteq B \wedge compact\ K\}.\ emeasure\ M\ K)$ (**is** $?inner\ B$)
and $outer\text{-}regular$: $emeasure\ M\ B =$
$(INF\ U : \{U.\ B \subseteq U \wedge open\ U\}.\ emeasure\ M\ U)$ (**is** $?outer\ B$)
⟨*proof*⟩

end

theory *Set-Integral*
imports *Bochner-Integration Lebesgue-Measure*
begin

abbreviation $set\text{-}borel\text{-}measurable\ M\ A\ f \equiv (\lambda x.\ indicator\ A\ x *_R f\ x) \in borel\text{-}measurable\ M$

abbreviation $set\text{-}integrable\ M\ A\ f \equiv integrable\ M\ (\lambda x.\ indicator\ A\ x *_R f\ x)$

abbreviation $set\text{-}lebesgue\text{-}integral\ M\ A\ f \equiv lebesgue\text{-}integral\ M\ (\lambda x.\ indicator\ A\ x *_R f\ x)$

syntax
$\text{-}ascii\text{-}set\text{-}lebesgue\text{-}integral :: pttrn \Rightarrow 'a\ set \Rightarrow 'a\ measure \Rightarrow real \Rightarrow real$
((4LINT (-):(-)/|(-)./ -) [0,60,110,61] 60)

translations
LINT x:A|M. f == CONST set-lebesgue-integral M A (λx. f)

abbreviation
$set\text{-}almost\text{-}everywhere\ A\ M\ P \equiv AE\ x\ in\ M.\ x \in A \longrightarrow P\ x$

syntax
$\text{-}set\text{-}almost\text{-}everywhere :: pttrn \Rightarrow 'a\ set \Rightarrow 'a \Rightarrow bool \Rightarrow bool$
(AE -∈- in -./ - [0,0,0,10] 10)

translations
AE $x \in A$ in M. P == CONST set-almost-everywhere A M (λx. P)

syntax

-lebesgue-borel-integral :: *pttrn* ⇒ *real* ⇒ *real*
((*2LBINT* -./ -) [*0*,*60*] *60*)

translations
LBINT x. f == *CONST lebesgue-integral CONST lborel* (λ*x. f*)

syntax
-set-lebesgue-borel-integral :: *pttrn* ⇒ *real set* ⇒ *real* ⇒ *real*
((*3LBINT* -:-./ -) [*0*,*60*,*61*] *60*)

translations
LBINT x:*A. f* == *CONST set-lebesgue-integral CONST lborel A* (λ*x. f*)

lemma *set-borel-measurable-sets*:
  fixes *f* :: - ⇒ -::*real-normed-vector*
  assumes *set-borel-measurable M X f B* ∈ *sets borel X* ∈ *sets M*
  shows *f* −' *B* ∩ *X* ∈ *sets M*
⟨*proof*⟩

lemma *set-lebesgue-integral-cong*:
  assumes *A* ∈ *sets M* **and** ∀ *x. x* ∈ *A* ⟶ *f x* = *g x*
  shows (*LINT x*:*A*|*M. f x*) = (*LINT x*:*A*|*M. g x*)
  ⟨*proof*⟩

lemma *set-lebesgue-integral-cong-AE*:
  assumes [*measurable*]: *A* ∈ *sets M f* ∈ *borel-measurable M g* ∈ *borel-measurable M*
  assumes *AE x* ∈ *A in M. f x* = *g x*
  shows *LINT x*:*A*|*M. f x* = *LINT x*:*A*|*M. g x*
⟨*proof*⟩

lemma *set-integrable-cong-AE*:
    f ∈ *borel-measurable M* ⟹ *g* ∈ *borel-measurable M* ⟹
    AE x ∈ *A in M. f x* = *g x* ⟹ *A* ∈ *sets M* ⟹
    set-integrable M A f = *set-integrable M A g*
  ⟨*proof*⟩

lemma *set-integrable-subset*:
  fixes *M A B* **and** *f* :: - ⇒ - :: {*banach*, *second-countable-topology*}
  assumes *set-integrable M A f B* ∈ *sets M B* ⊆ *A*
  shows *set-integrable M B f*
⟨*proof*⟩

lemma *set-integral-scaleR-right* [*simp*]: *LINT t*:*A*|*M*. *a* $*_R$ *f t* = *a* $*_R$ (*LINT t*:*A*|*M*. *f t*)
⟨*proof*⟩

lemma *set-integral-mult-right* [*simp*]:
fixes *a* :: ′*a*::{*real-normed-field*, *second-countable-topology*}
shows *LINT t*:*A*|*M*. *a* ∗ *f t* = *a* ∗ (*LINT t*:*A*|*M*. *f t*)
⟨*proof*⟩

lemma *set-integral-mult-left* [*simp*]:
fixes *a* :: ′*a*::{*real-normed-field*, *second-countable-topology*}
shows *LINT t*:*A*|*M*. *f t* ∗ *a* = (*LINT t*:*A*|*M*. *f t*) ∗ *a*
⟨*proof*⟩

lemma *set-integral-divide-zero* [*simp*]:
fixes *a* :: ′*a*::{*real-normed-field*, *field*, *second-countable-topology*}
shows *LINT t*:*A*|*M*. *f t* / *a* = (*LINT t*:*A*|*M*. *f t*) / *a*
⟨*proof*⟩

lemma *set-integrable-scaleR-right* [*simp*, *intro*]:
shows (*a* ≠ *0* ⟹ *set-integrable M A f*) ⟹ *set-integrable M A* (λ*t*. *a* $*_R$ *f t*)
⟨*proof*⟩

lemma *set-integrable-scaleR-left* [*simp*, *intro*]:
fixes *a* :: - :: {*banach*, *second-countable-topology*}
shows (*a* ≠ *0* ⟹ *set-integrable M A f*) ⟹ *set-integrable M A* (λ*t*. *f t* $*_R$ *a*)
⟨*proof*⟩

lemma *set-integrable-mult-right* [*simp*, *intro*]:
fixes *a* :: ′*a*::{*real-normed-field*, *second-countable-topology*}
shows (*a* ≠ *0* ⟹ *set-integrable M A f*) ⟹ *set-integrable M A* (λ*t*. *a* ∗ *f t*)
⟨*proof*⟩

lemma *set-integrable-mult-left* [*simp*, *intro*]:
fixes *a* :: ′*a*::{*real-normed-field*, *second-countable-topology*}
shows (*a* ≠ *0* ⟹ *set-integrable M A f*) ⟹ *set-integrable M A* (λ*t*. *f t* ∗ *a*)
⟨*proof*⟩

lemma *set-integrable-divide* [*simp*, *intro*]:
fixes *a* :: ′*a*::{*real-normed-field*, *field*, *second-countable-topology*}
assumes *a* ≠ *0* ⟹ *set-integrable M A f*
shows *set-integrable M A* (λ*t*. *f t* / *a*)
⟨*proof*⟩

lemma *set-integral-add* [*simp*, *intro*]:
fixes *f g* :: - ⇒ - :: {*banach*, *second-countable-topology*}
assumes *set-integrable M A f set-integrable M A g*
shows *set-integrable M A* (λ*x*. *f x* + *g x*)

and $LINT\ x{:}A|M.\ f\ x + g\ x = (LINT\ x{:}A|M.\ f\ x) + (LINT\ x{:}A|M.\ g\ x)$
⟨*proof*⟩

lemma *set-integral-diff* [*simp*, *intro*]:
assumes *set-integrable M A f set-integrable M A g*
shows *set-integrable M A* $(\lambda x.\ f\ x - g\ x)$ **and** $LINT\ x{:}A|M.\ f\ x - g\ x = (LINT\ x{:}A|M.\ f\ x) - (LINT\ x{:}A|M.\ g\ x)$
⟨*proof*⟩

lemma *set-integral-reflect*:
fixes S **and** $f :: real \Rightarrow 'a :: \{banach,\ second\text{-}countable\text{-}topology\}$
shows $(LBINT\ x : S.\ f\ x) = (LBINT\ x : \{x.\ - x \in S\}.\ f\ (- x))$
⟨*proof*⟩

lemma *set-integral-uminus*: *set-integrable M A f* $\Longrightarrow LINT\ x{:}A|M.\ - f\ x = - (LINT\ x{:}A|M.\ f\ x)$
⟨*proof*⟩

lemma *set-integral-complex-of-real*:
$LINT\ x{:}A|M.\ complex\text{-}of\text{-}real\ (f\ x) = of\text{-}real\ (LINT\ x{:}A|M.\ f\ x)$
⟨*proof*⟩

lemma *set-integral-mono*:
fixes $f\ g :: \text{-} \Rightarrow real$
assumes *set-integrable M A f set-integrable M A g*
$\bigwedge x.\ x \in A \Longrightarrow f\ x \le g\ x$
shows $(LINT\ x{:}A|M.\ f\ x) \le (LINT\ x{:}A|M.\ g\ x)$
⟨*proof*⟩

lemma *set-integral-mono-AE*:
fixes $f\ g :: \text{-} \Rightarrow real$
assumes *set-integrable M A f set-integrable M A g*
$AE\ x \in A\ in\ M.\ f\ x \le g\ x$
shows $(LINT\ x{:}A|M.\ f\ x) \le (LINT\ x{:}A|M.\ g\ x)$
⟨*proof*⟩

lemma *set-integrable-abs*: *set-integrable M A f* $\Longrightarrow$ *set-integrable M A* $(\lambda x.\ |f\ x| :: real)$
⟨*proof*⟩

lemma *set-integrable-abs-iff*:
fixes $f :: \text{-} \Rightarrow real$
shows *set-borel-measurable M A f* $\Longrightarrow$ *set-integrable M A* $(\lambda x.\ |f\ x|)$ = *set-integrable M A f*
⟨*proof*⟩

lemma *set-integrable-abs-iff'*:
fixes $f :: \text{-} \Rightarrow real$

shows *f* ∈ *borel-measurable M* ⟹ *A* ∈ *sets M* ⟹
set-integrable M A (λ*x*. |*f x*|) = *set-integrable M A f*
⟨*proof*⟩

lemma *set-integrable-discrete-difference*:
fixes *f* :: ′*a* ⇒ ′*b*::{*banach*, *second-countable-topology*}
assumes *countable X*
assumes *diff*: (*A* − *B*) ∪ (*B* − *A*) ⊆ *X*
assumes ⋀*x*. *x* ∈ *X* ⟹ *emeasure M* {*x*} = *0* ⋀*x*. *x* ∈ *X* ⟹ {*x*} ∈ *sets M*
shows *set-integrable M A f* ⟷ *set-integrable M B f*
⟨*proof*⟩

lemma *set-integral-discrete-difference*:
fixes *f* :: ′*a* ⇒ ′*b*::{*banach*, *second-countable-topology*}
assumes *countable X*
assumes *diff*: (*A* − *B*) ∪ (*B* − *A*) ⊆ *X*
assumes ⋀*x*. *x* ∈ *X* ⟹ *emeasure M* {*x*} = *0* ⋀*x*. *x* ∈ *X* ⟹ {*x*} ∈ *sets M*
shows *set-lebesgue-integral M A f* = *set-lebesgue-integral M B f*
⟨*proof*⟩

lemma *set-integrable-Un*:
fixes *f g* :: - ⇒ - :: {*banach*, *second-countable-topology*}
assumes *f-A*: *set-integrable M A f* **and** *f-B*: *set-integrable M B f*
and [*measurable*]: *A* ∈ *sets M B* ∈ *sets M*
shows *set-integrable M* (*A* ∪ *B*) *f*
⟨*proof*⟩

lemma *set-integrable-UN*:
fixes *f* :: - ⇒ - :: {*banach*, *second-countable-topology*}
assumes *finite I* ⋀*i*. *i*∈*I* ⟹ *set-integrable M* (*A i*) *f*
⋀*i*. *i*∈*I* ⟹ *A i* ∈ *sets M*
shows *set-integrable M* (⋃*i*∈*I*. *A i*) *f*
⟨*proof*⟩

lemma *set-integral-Un*:
fixes *f* :: - ⇒ - :: {*banach*, *second-countable-topology*}
assumes *A* ∩ *B* = {}
and *set-integrable M A f*
and *set-integrable M B f*
shows *LINT x*:*A*∪*B*|*M*. *f x* = (*LINT x*:*A*|*M*. *f x*) + (*LINT x*:*B*|*M*. *f x*)
⟨*proof*⟩

lemma *set-integral-cong-set*:
fixes *f* :: - ⇒ - :: {*banach*, *second-countable-topology*}
assumes [*measurable*]: *set-borel-measurable M A f set-borel-measurable M B f*
and *ae*: *AE x in M*. *x* ∈ *A* ⟷ *x* ∈ *B*
shows *LINT x*:*B*|*M*. *f x* = *LINT x*:*A*|*M*. *f x*
⟨*proof*⟩

lemma *set-borel-measurable-subset*:
fixes f :: - ⇒ - :: {*banach*, *second-countable-topology*}
assumes [*measurable*]: *set-borel-measurable M A f B* ∈ *sets M* **and** $B \subseteq A$
shows *set-borel-measurable M B f*
⟨*proof*⟩

lemma *set-integral-Un-AE*:
fixes f :: - ⇒ - :: {*banach*, *second-countable-topology*}
assumes *ae*: *AE x in M*. ¬ ($x \in A \wedge x \in B$) **and** [*measurable*]: $A \in$ *sets M B* ∈ *sets M*
and *set-integrable M A f*
and *set-integrable M B f*
shows *LINT x*:$A \cup B$|*M*. *f x* = (*LINT x*:*A*|*M*. *f x*) + (*LINT x*:*B*|*M*. *f x*)
⟨*proof*⟩

lemma *set-integral-finite-Union*:
fixes f :: - ⇒ - :: {*banach*, *second-countable-topology*}
assumes *finite I disjoint-family-on A I*
and ⋀*i*. $i \in I \Longrightarrow$ *set-integrable M* (*A i*) *f* ⋀*i*. $i \in I \Longrightarrow A\ i \in$ *sets M*
shows (*LINT x*:(⋃*i*∈*I*. *A i*)|*M*. *f x*) = (∑*i*∈*I*. *LINT x*:*A i*|*M*. *f x*)
⟨*proof*⟩

lemma *pos-integrable-to-top*:
fixes *l*::*real*
assumes ⋀*i*. *A i* ∈ *sets M mono A*
assumes *nneg*: ⋀*x i*. $x \in A\ i \Longrightarrow 0 \leq f\ x$
and *intgbl*: ⋀*i*::*nat*. *set-integrable M* (*A i*) *f*
and *lim*: (λ*i*::*nat*. *LINT x*:*A i*|*M*. *f x*) ⟶ *l*
shows *set-integrable M* (⋃*i*. *A i*) *f*
⟨*proof*⟩

lemma *lebesgue-integral-countable-add*:
fixes f :: - ⇒ ′*a* :: {*banach*, *second-countable-topology*}
assumes *meas*[*intro*]: ⋀*i*::*nat*. *A i* ∈ *sets M*
and *disj*: ⋀*i j*. $i \neq j \Longrightarrow A\ i \cap A\ j = \{\}$
and *intgbl*: *set-integrable M* (⋃*i*. *A i*) *f*
shows *LINT x*:(⋃*i*. *A i*)|*M*. *f x* = (∑*i*. (*LINT x*:(*A i*)|*M*. *f x*))
⟨*proof*⟩

lemma *set-integral-cont-up*:
fixes f :: - ⇒ ′*a* :: {*banach*, *second-countable-topology*}
assumes [*measurable*]: ⋀*i*. *A i* ∈ *sets M* **and** *A*: *incseq A*
and *intgbl*: *set-integrable M* (⋃*i*. *A i*) *f*
shows (λ*i*. *LINT x*:(*A i*)|*M*. *f x*) ⟶ *LINT x*:(⋃*i*. *A i*)|*M*. *f x*
⟨*proof*⟩

lemma *set-integral-cont-down*:
fixes f :: - ⇒ $'a$ :: {*banach*, *second-countable-topology*}
assumes [*measurable*]: $\bigwedge i.\ A\ i \in sets\ M$ **and** A: *decseq* A
and *int0*: *set-integrable* M $(A\ 0)$ f
shows $(\lambda i{::}nat.\ LINT\ x{:}(A\ i)|M.\ f\ x) \longrightarrow LINT\ x{:}(\bigcap i.\ A\ i)|M.\ f\ x$
⟨*proof*⟩

lemma *set-integral-at-point*:
fixes a :: *real*
assumes *set-integrable* M $\{a\}$ f
and [*simp*]: $\{a\} \in sets\ M$ **and** $(emeasure\ M)\ \{a\} \neq \infty$
shows $(LINT\ x{:}\{a\}\ |\ M.\ f\ x) = f\ a * measure\ M\ \{a\}$
⟨*proof*⟩

abbreviation *complex-integrable* :: $'a$ *measure* ⇒ ($'a$ ⇒ *complex*) ⇒ *bool* **where**
complex-integrable $M\ f \equiv$ *integrable* $M\ f$

abbreviation *complex-lebesgue-integral* :: $'a$ *measure* ⇒ ($'a$ ⇒ *complex*) ⇒ *complex* ($integral^C$) **where**
$integral^C\ M\ f == integral^L\ M\ f$

syntax
-complex-lebesgue-integral :: *pttrn* ⇒ *complex* ⇒ $'a$ *measure* ⇒ *complex*
$(\int^C$ -. - ∂- [*60*,*61*] *110*)

translations
$\int^C x.\ f\ \partial M$ == *CONST complex-lebesgue-integral* M $(\lambda x.\ f)$

syntax
-ascii-complex-lebesgue-integral :: *pttrn* ⇒ $'a$ *measure* ⇒ *real* ⇒ *real*
((*3CLINT* -|-. -) [*0*,*110*,*60*] *60*)

translations
CLINT $x|M.\ f$ == *CONST complex-lebesgue-integral* M $(\lambda x.\ f)$

lemma *complex-integrable-cnj* [*simp*]:
complex-integrable M $(\lambda x.\ cnj\ (f\ x)) \longleftrightarrow$ *complex-integrable* $M\ f$
⟨*proof*⟩

lemma *complex-of-real-integrable-eq*:
complex-integrable M $(\lambda x.\ complex\text{-}of\text{-}real\ (f\ x)) \longleftrightarrow$ *integrable* $M\ f$
⟨*proof*⟩

abbreviation *complex-set-integrable* :: $'a$ *measure* ⇒ $'a$ *set* ⇒ ($'a$ ⇒ *complex*) ⇒ *bool* **where**
complex-set-integrable $M\ A\ f \equiv$ *set-integrable* $M\ A\ f$

abbreviation *complex-set-lebesgue-integral* :: *'a measure* ⇒ *'a set* ⇒ (*'a* ⇒ *complex*) ⇒ *complex* **where**
complex-set-lebesgue-integral M A f ≡ *set-lebesgue-integral M A f*

syntax
-ascii-complex-set-lebesgue-integral :: *pttrn* ⇒ *'a set* ⇒ *'a measure* ⇒ *real* ⇒ *real*
((*4CLINT* -:-|-. -) [*0*,*60*,*110*,*61*] *60*)

translations
CLINT x:*A*|*M*. *f* == *CONST complex-set-lebesgue-integral M A* (λ*x*. *f*)

lemma *borel-integrable-atLeastAtMost'*:
fixes *f* :: *real* ⇒ *'a*::{*banach*, *second-countable-topology*}
assumes *f*: *continuous-on* {*a*..*b*} *f*
shows *set-integrable lborel* {*a*..*b*} *f* (**is** *integrable - ?f*)
⟨*proof*⟩

lemma *integral-FTC-atLeastAtMost*:
fixes *f* :: *real* ⇒ *'a* :: *euclidean-space*
assumes $a \leq b$
and *F*: ⋀*x*. $a \leq x \Longrightarrow x \leq b \Longrightarrow$ (*F has-vector-derivative f x*) (*at x within* {*a* .. *b*})
and *f*: *continuous-on* {*a* .. *b*} *f*
shows *integral*^{L} *lborel* (λ*x*. *indicator* {*a* .. *b*} *x* $*_R$ *f x*) = *F b* − *F a*
⟨*proof*⟩

lemma *set-borel-integral-eq-integral*:
fixes *f* :: *real* ⇒ *'a*::*euclidean-space*
assumes *set-integrable lborel S f*
shows *f integrable-on S LINT x* : *S* | *lborel*. *f x* = *integral S f*
⟨*proof*⟩

lemma *set-borel-measurable-continuous*:
fixes *f* :: - ⇒ -::*real-normed-vector*
assumes *S* ∈ *sets borel continuous-on S f*
shows *set-borel-measurable borel S f*
⟨*proof*⟩

lemma *set-measurable-continuous-on-ivl*:
assumes *continuous-on* {*a*..*b*} (*f* :: *real* ⇒ *real*)
shows *set-borel-measurable borel* {*a*..*b*} *f*
⟨*proof*⟩

end

theory *Interval-Integral*

imports *Set-Integral*
begin

lemma *continuous-on-vector-derivative*:
$(\bigwedge x.\ x \in S \Longrightarrow (f\ \textit{has-vector-derivative}\ f'\ x)\ (\textit{at}\ x\ \textit{within}\ S)) \Longrightarrow \textit{continuous-on}\ S\ f$
⟨*proof*⟩

lemma *has-vector-derivative-weaken*:
fixes $x\ D$ **and** $f\ g\ s\ t$
assumes f: $(f\ \textit{has-vector-derivative}\ D)\ (\textit{at}\ x\ \textit{within}\ t)$
and $x \in s\ s \subseteq t$
and $\bigwedge x.\ x \in s \Longrightarrow f\ x = g\ x$
shows $(g\ \textit{has-vector-derivative}\ D)\ (\textit{at}\ x\ \textit{within}\ s)$
⟨*proof*⟩

definition $\textit{einterval}\ a\ b = \{x.\ a < \textit{ereal}\ x \wedge \textit{ereal}\ x < b\}$

lemma *einterval-eq*[*simp*]:
shows *einterval-eq-Icc*: $\textit{einterval}\ (\textit{ereal}\ a)\ (\textit{ereal}\ b) = \{a <..< b\}$
and *einterval-eq-Ici*: $\textit{einterval}\ (\textit{ereal}\ a)\ \infty = \{a <..\}$
and *einterval-eq-Iic*: $\textit{einterval}\ (-\ \infty)\ (\textit{ereal}\ b) = \{..< b\}$
and *einterval-eq-UNIV*: $\textit{einterval}\ (-\ \infty)\ \infty = \textit{UNIV}$
⟨*proof*⟩

lemma *einterval-same*: $\textit{einterval}\ a\ a = \{\}$
⟨*proof*⟩

lemma *einterval-iff*: $x \in \textit{einterval}\ a\ b \longleftrightarrow a < \textit{ereal}\ x \wedge \textit{ereal}\ x < b$
⟨*proof*⟩

lemma *einterval-nonempty*: $a < b \Longrightarrow \exists\, c.\ c \in \textit{einterval}\ a\ b$
⟨*proof*⟩

lemma *open-einterval*[*simp*]: $\textit{open}\ (\textit{einterval}\ a\ b)$
⟨*proof*⟩

lemma *borel-einterval*[*measurable*]: $\textit{einterval}\ a\ b \in \textit{sets borel}$
⟨*proof*⟩

lemma *filterlim-sup1*: $(\textit{LIM}\ x\ F.\ f\ x :> G1) \Longrightarrow (\textit{LIM}\ x\ F.\ f\ x :> (\textit{sup}\ G1\ G2))$
⟨*proof*⟩

lemma *ereal-incseq-approx*:
fixes $a\ b :: \textit{ereal}$
assumes $a < b$
obtains $X :: \textit{nat} \Rightarrow \textit{real}$ **where**

incseq X $\bigwedge i.\ a < X\ i$ $\bigwedge i.\ X\ i < b$ $X \longrightarrow b$
⟨*proof*⟩

lemma *ereal-decseq-approx*:
fixes *a b* :: *ereal*
assumes $a < b$
obtains *X* :: *nat* ⇒ *real* **where**
decseq X $\bigwedge i.\ a < X\ i$ $\bigwedge i.\ X\ i < b$ $X \longrightarrow a$
⟨*proof*⟩

lemma *einterval-Icc-approximation*:
fixes *a b* :: *ereal*
assumes $a < b$
obtains *u l* :: *nat* ⇒ *real* **where**
einterval a b = ($\bigcup$ *i*. {*l i* .. *u i*})
incseq u decseq l $\bigwedge i.\ l\ i < u\ i$ $\bigwedge i.\ a < l\ i$ $\bigwedge i.\ u\ i < b$
$l \longrightarrow a$ $u \longrightarrow b$
⟨*proof*⟩

definition *interval-lebesgue-integral* :: *real measure* ⇒ *ereal* ⇒ *ereal* ⇒ (*real* ⇒ *'a*) ⇒ *'a*::{*banach*, *second-countable-topology*} **where**
interval-lebesgue-integral M a b f =
(*if* $a \le b$ *then* (*LINT x*:*einterval a b*|*M*. *f x*) *else* − (*LINT x*:*einterval b a*|*M*. *f x*))

syntax
-ascii-interval-lebesgue-integral :: *pttrn* ⇒ *real* ⇒ *real* ⇒ *real measure* ⇒ *real* ⇒ *real*
((*5LINT -=-..-|-. -*) [*0*,*60*,*60*,*61*,*100*] *60*)

translations
LINT x=a..b|M. f == *CONST interval-lebesgue-integral M a b* (λ*x*. *f*)

definition *interval-lebesgue-integrable* :: *real measure* ⇒ *ereal* ⇒ *ereal* ⇒ (*real* ⇒ *'a*::{*banach*, *second-countable-topology*}) ⇒ *bool* **where**
interval-lebesgue-integrable M a b f =
(*if* $a \le b$ *then set-integrable M* (*einterval a b*) *f else set-integrable M* (*einterval b a*) *f*)

syntax
-ascii-interval-lebesgue-borel-integral :: *pttrn* ⇒ *real* ⇒ *real* ⇒ *real* ⇒ *real*
((*4LBINT -=-..-. -*) [*0*,*60*,*60*,*61*] *60*)

translations
LBINT x=a..b. f == *CONST interval-lebesgue-integral CONST lborel a b* (λ*x*. *f*)

lemma *interval-lebesgue-integral-cong*:
$a \le b \Longrightarrow (\bigwedge x.\ x \in einterval\ a\ b \Longrightarrow f\,x = g\ x) \Longrightarrow einterval\ a\ b \in sets\ M \Longrightarrow$
$interval\text{-}lebesgue\text{-}integral\ M\ a\ b\ f = interval\text{-}lebesgue\text{-}integral\ M\ a\ b\ g$
⟨*proof*⟩

lemma *interval-lebesgue-integral-cong-AE*:
$f \in borel\text{-}measurable\ M \Longrightarrow g \in borel\text{-}measurable\ M \Longrightarrow$
$a \le b \Longrightarrow AE\ x \in einterval\ a\ b\ in\ M.\ f\ x = g\ x \Longrightarrow einterval\ a\ b \in sets\ M$
$\Longrightarrow$
$interval\text{-}lebesgue\text{-}integral\ M\ a\ b\ f = interval\text{-}lebesgue\text{-}integral\ M\ a\ b\ g$
⟨*proof*⟩

lemma *interval-integrable-mirror*:
shows $interval\text{-}lebesgue\text{-}integrable\ lborel\ a\ b\ (\lambda x.\ f\ (-x)) \longleftrightarrow$
$interval\text{-}lebesgue\text{-}integrable\ lborel\ (-b)\ (-a)\ f$
⟨*proof*⟩

lemma *interval-lebesgue-integral-add* [*intro*, *simp*]:
fixes $M\ a\ b\ f$
assumes $interval\text{-}lebesgue\text{-}integrable\ M\ a\ b\ f\ interval\text{-}lebesgue\text{-}integrable\ M\ a\ b\ g$
shows $interval\text{-}lebesgue\text{-}integrable\ M\ a\ b\ (\lambda x.\ f\,x + g\ x)$ **and**
$interval\text{-}lebesgue\text{-}integral\ M\ a\ b\ (\lambda x.\ f\,x + g\ x) =$
$interval\text{-}lebesgue\text{-}integral\ M\ a\ b\ f + interval\text{-}lebesgue\text{-}integral\ M\ a\ b\ g$
⟨*proof*⟩

lemma *interval-lebesgue-integral-diff* [*intro*, *simp*]:
fixes $M\ a\ b\ f$
assumes $interval\text{-}lebesgue\text{-}integrable\ M\ a\ b\ f$
$interval\text{-}lebesgue\text{-}integrable\ M\ a\ b\ g$
shows $interval\text{-}lebesgue\text{-}integrable\ M\ a\ b\ (\lambda x.\ f\,x - g\ x)$ **and**
$interval\text{-}lebesgue\text{-}integral\ M\ a\ b\ (\lambda x.\ f\,x - g\ x) =$
$interval\text{-}lebesgue\text{-}integral\ M\ a\ b\ f - interval\text{-}lebesgue\text{-}integral\ M\ a\ b\ g$
⟨*proof*⟩

lemma *interval-lebesgue-integrable-mult-right* [*intro*, *simp*]:
fixes $M\ a\ b\ c$ **and** $f :: real \Rightarrow 'a::\{banach, real\text{-}normed\text{-}field, second\text{-}countable\text{-}topology\}$
shows $(c \neq 0 \Longrightarrow interval\text{-}lebesgue\text{-}integrable\ M\ a\ b\ f) \Longrightarrow$
$interval\text{-}lebesgue\text{-}integrable\ M\ a\ b\ (\lambda x.\ c * f\ x)$
⟨*proof*⟩

lemma *interval-lebesgue-integrable-mult-left* [*intro*, *simp*]:
fixes $M\ a\ b\ c$ **and** $f :: real \Rightarrow 'a::\{banach, real\text{-}normed\text{-}field, second\text{-}countable\text{-}topology\}$
shows $(c \neq 0 \Longrightarrow interval\text{-}lebesgue\text{-}integrable\ M\ a\ b\ f) \Longrightarrow$
$interval\text{-}lebesgue\text{-}integrable\ M\ a\ b\ (\lambda x.\ f\,x * c)$
⟨*proof*⟩

lemma *interval-lebesgue-integrable-divide* [*intro*, *simp*]:
fixes $M\ a\ b\ c$ **and** $f :: real \Rightarrow 'a::\{banach, real\text{-}normed\text{-}field, field, second\text{-}countable\text{-}topology\}$

shows ($c \neq 0 \Longrightarrow$ *interval-lebesgue-integrable M a b f*) $\Longrightarrow$
interval-lebesgue-integrable M a b ($\lambda x.\ f\ x\ /\ c$)
⟨*proof*⟩

lemma *interval-lebesgue-integral-mult-right* [*simp*]:
fixes *M a b c* **and** *f* :: *real* ⇒ *'a*::{*banach*, *real-normed-field*, *second-countable-topology*}
shows *interval-lebesgue-integral M a b* ($\lambda x.\ c * f\ x$) =
c * *interval-lebesgue-integral M a b f*
⟨*proof*⟩

lemma *interval-lebesgue-integral-mult-left* [*simp*]:
fixes *M a b c* **and** *f* :: *real* ⇒ *'a*::{*banach*, *real-normed-field*, *second-countable-topology*}
shows *interval-lebesgue-integral M a b* ($\lambda x.\ f\ x * c$) =
interval-lebesgue-integral M a b f * *c*
⟨*proof*⟩

lemma *interval-lebesgue-integral-divide* [*simp*]:
fixes *M a b c* **and** *f* :: *real* ⇒ *'a*::{*banach*, *real-normed-field*, *field*, *second-countable-topology*}
shows *interval-lebesgue-integral M a b* ($\lambda x.\ f\ x\ /\ c$) =
interval-lebesgue-integral M a b f / *c*
⟨*proof*⟩

lemma *interval-lebesgue-integral-uminus*:
interval-lebesgue-integral M a b ($\lambda x.\ -\ f\ x$) = − *interval-lebesgue-integral M a b*
f
⟨*proof*⟩

lemma *interval-lebesgue-integral-of-real*:
interval-lebesgue-integral M a b ($\lambda x.$ *complex-of-real* ($f\ x$)) =
of-real (*interval-lebesgue-integral M a b f*)
⟨*proof*⟩

lemma *interval-lebesgue-integral-le-eq*:
fixes *a b f*
assumes $a \leq b$
shows *interval-lebesgue-integral M a b f* = (*LINT x* : *einterval a b* | *M*. *f x*)
⟨*proof*⟩

lemma *interval-lebesgue-integral-gt-eq*:
fixes *a b f*
assumes $a > b$
shows *interval-lebesgue-integral M a b f* = −(*LINT x* : *einterval b a* | *M*. *f x*)
⟨*proof*⟩

lemma *interval-lebesgue-integral-gt-eq'*:
fixes *a b f*
assumes $a > b$
shows *interval-lebesgue-integral M a b f* = − *interval-lebesgue-integral M b a f*
⟨*proof*⟩

lemma *interval-integral-endpoints-same* [*simp*]: (*LBINT x=a..a. f x*) = *0*
⟨*proof*⟩

lemma *interval-integral-endpoints-reverse*: (*LBINT x=a..b. f x*) = −(*LBINT x=b..a. f x*)
⟨*proof*⟩

lemma *interval-integrable-endpoints-reverse*:
interval-lebesgue-integrable lborel a b f ⟷
interval-lebesgue-integrable lborel b a f
⟨*proof*⟩

lemma *interval-integral-reflect*:
(*LBINT x=a..b. f x*) = (*LBINT x=−b..−a. f* (−*x*))
⟨*proof*⟩

lemma *interval-lebesgue-integral-0-infty*:
interval-lebesgue-integrable M 0 ∞ *f* ⟷ *set-integrable M* {*0*<..} *f*
interval-lebesgue-integral M 0 ∞ *f* = (*LINT x*:{*0*<..}|*M. f x*)
⟨*proof*⟩

lemma *interval-integral-to-infinity-eq*: (*LINT x=ereal a..*∞ | *M. f x*) = (*LINT x* : {*a*<..} | *M. f x*)
⟨*proof*⟩

lemma *interval-integrable-to-infinity-eq*: (*interval-lebesgue-integrable M a* ∞ *f*) =

(*set-integrable M* {*a*<..} *f*)
⟨*proof*⟩

lemma *interval-integral-zero* [*simp*]:
fixes *a b* :: *ereal*
shows*LBINT x=a..b. 0* = *0*
⟨*proof*⟩

lemma *interval-integral-const* [*intro*, *simp*]:
fixes *a b c* :: *real*
shows *interval-lebesgue-integrable lborel a b* (λ*x. c*) **and** *LBINT x=a..b. c* = *c* ∗ (*b* − *a*)
⟨*proof*⟩

lemma *interval-integral-cong-AE*:
assumes [*measurable*]: *f* ∈ *borel-measurable borel g* ∈ *borel-measurable borel*
assumes *AE x* ∈ *einterval* (*min a b*) (*max a b*) *in lborel. f x* = *g x*
shows *interval-lebesgue-integral lborel a b f* = *interval-lebesgue-integral lborel a b g*

⟨*proof*⟩

lemma *interval-integral-cong*:
assumes $\bigwedge x.\ x \in einterval\ (min\ a\ b)\ (max\ a\ b) \Longrightarrow f\ x = g\ x$
shows $interval\text{-}lebesgue\text{-}integral\ lborel\ a\ b\ f = interval\text{-}lebesgue\text{-}integral\ lborel\ a\ b\ g$
⟨*proof*⟩

lemma *interval-lebesgue-integrable-cong-AE*:
$f \in borel\text{-}measurable\ lborel \Longrightarrow g \in borel\text{-}measurable\ lborel \Longrightarrow$
$AE\ x \in einterval\ (min\ a\ b)\ (max\ a\ b)\ in\ lborel.\ f\ x = g\ x \Longrightarrow$
$interval\text{-}lebesgue\text{-}integrable\ lborel\ a\ b\ f = interval\text{-}lebesgue\text{-}integrable\ lborel\ a\ b\ g$
⟨*proof*⟩

lemma *interval-integrable-abs-iff*:
fixes $f :: real \Rightarrow real$
shows $f \in borel\text{-}measurable\ lborel \Longrightarrow$
$interval\text{-}lebesgue\text{-}integrable\ lborel\ a\ b\ (\lambda x.\ |f\ x|) = interval\text{-}lebesgue\text{-}integrable\ lborel\ a\ b\ f$
⟨*proof*⟩

lemma *interval-integral-Icc*:
fixes $a\ b :: real$
shows $a \le b \Longrightarrow (LBINT\ x{=}a..b.\ f\ x) = (LBINT\ x : \{a..b\}.\ f\ x)$
⟨*proof*⟩

lemma *interval-integral-Icc'*:
$a \le b \Longrightarrow (LBINT\ x{=}a..b.\ f\ x) = (LBINT\ x : \{x.\ a \le ereal\ x \wedge ereal\ x \le b\}.\ f\ x)$
⟨*proof*⟩

lemma *interval-integral-Ioc*:
$a \le b \Longrightarrow (LBINT\ x{=}a..b.\ f\ x) = (LBINT\ x : \{a{<}..b\}.\ f\ x)$
⟨*proof*⟩

lemma *interval-integral-Ioc'*:
$a \le b \Longrightarrow (LBINT\ x{=}a..b.\ f\ x) = (LBINT\ x : \{x.\ a < ereal\ x \wedge ereal\ x \le b\}.\ f\ x)$
⟨*proof*⟩

lemma *interval-integral-Ico*:
$a \le b \Longrightarrow (LBINT\ x{=}a..b.\ f\ x) = (LBINT\ x : \{a..{<}b\}.\ f\ x)$
⟨*proof*⟩

lemma *interval-integral-Ioi*:
$|a| < \infty \Longrightarrow (LBINT\ x{=}a..\infty.\ f\ x) = (LBINT\ x : \{real\text{-}of\text{-}ereal\ a\ {<}..\}.\ f\ x)$
⟨*proof*⟩

lemma *interval-integral-Ioo*:
$a \leq b \Longrightarrow |a| < \infty$ ==> $|b| < \infty \Longrightarrow$ (*LBINT* x=$a..b$. $f\ x$) = (*LBINT* x : {*real-of-ereal* a <..< *real-of-ereal* b}. $f\ x$)
⟨*proof*⟩

lemma *interval-integral-discrete-difference*:
fixes f :: *real* ⇒ ′*b*::{*banach*, *second-countable-topology*} **and** $a\ b$ :: *ereal*
assumes *countable* X
and *eq*: $\bigwedge x.\ a \leq b \Longrightarrow a < x \Longrightarrow x < b \Longrightarrow x \notin X \Longrightarrow f\ x = g\ x$
and *anti-eq*: $\bigwedge x.\ b \leq a \Longrightarrow b < x \Longrightarrow x < a \Longrightarrow x \notin X \Longrightarrow f\ x = g\ x$
assumes $\bigwedge x.\ x \in X \Longrightarrow$ *emeasure* M $\{x\} = 0$ $\bigwedge x.\ x \in X \Longrightarrow \{x\} \in$ *sets* M
shows *interval-lebesgue-integral* $M\ a\ b\ f$ = *interval-lebesgue-integral* $M\ a\ b\ g$
⟨*proof*⟩

lemma *interval-integral-sum*:
fixes $a\ b\ c$ :: *ereal*
assumes *integrable*: *interval-lebesgue-integrable lborel* (*min* a (*min* b c)) (*max* a (*max* b c)) f
shows (*LBINT* x=$a..b$. $f\ x$) + (*LBINT* x=$b..c$. $f\ x$) = (*LBINT* x=$a..c$. $f\ x$)
⟨*proof*⟩

lemma *interval-integrable-isCont*:
fixes $a\ b$ **and** f :: *real* ⇒ ′*a*::{*banach*, *second-countable-topology*}
shows ($\bigwedge x.$ *min* $a\ b \leq x \Longrightarrow x \leq$ *max* $a\ b \Longrightarrow$ *isCont* $f\ x$) $\Longrightarrow$
interval-lebesgue-integrable lborel $a\ b\ f$
⟨*proof*⟩

lemma *interval-integrable-continuous-on*:
fixes $a\ b$ :: *real* **and** f
assumes $a \leq b$ **and** *continuous-on* $\{a..b\}$ f
shows *interval-lebesgue-integrable lborel* $a\ b\ f$
⟨*proof*⟩

lemma *interval-integral-eq-integral*:
fixes f :: *real* ⇒ ′*a*::*euclidean-space*
shows $a \leq b \Longrightarrow$ *set-integrable lborel* $\{a..b\}$ $f \Longrightarrow$ *LBINT* x=$a..b$. $f\ x$ = *integral* $\{a..b\}$ f
⟨*proof*⟩

lemma *interval-integral-eq-integral′*:
fixes f :: *real* ⇒ ′*a*::*euclidean-space*
shows $a \leq b \Longrightarrow$ *set-integrable lborel* (*einterval* a b) $f \Longrightarrow$ *LBINT* x=$a..b$. $f\ x$ = *integral* (*einterval* a b) f
⟨*proof*⟩

lemma *interval-integral-Icc-approx-nonneg*:
fixes $a\ b$ :: *ereal*

assumes $a < b$
fixes $u\ l :: nat \Rightarrow real$
assumes *approx*: $einterval\ a\ b = (\bigcup i.\ \{l\ i\ ..\ u\ i\})$
$incseq\ u\ decseq\ l\ \bigwedge i.\ l\ i < u\ i\ \bigwedge i.\ a < l\ i\ \bigwedge i.\ u\ i < b$
$l \longrightarrow a\ u \longrightarrow b$
fixes $f :: real \Rightarrow real$
assumes *f-integrable*: $\bigwedge i.$ *set-integrable lborel* $\{l\ i..u\ i\}\ f$
assumes *f-nonneg*: *AE x in lborel*. $a <$ *ereal* $x \longrightarrow$ *ereal* $x < b \longrightarrow 0 \le f\ x$
assumes *f-measurable*: *set-borel-measurable lborel* (*einterval a b*) *f*
assumes *lbint-lim*: (λi. *LBINT* x=$l\ i$.. $u\ i$. $f\ x$) $\longrightarrow C$
shows
set-integrable lborel (*einterval a b*) *f*
(*LBINT* x=$a..b$. $f\ x$) $= C$
⟨*proof*⟩

lemma *interval-integral-Icc-approx-integrable*:
fixes $u\ l :: nat \Rightarrow real$ **and** $a\ b :: ereal$
fixes $f :: real \Rightarrow 'a$::{*banach*, *second-countable-topology*}
assumes $a < b$
assumes *approx*: $einterval\ a\ b = (\bigcup i.\ \{l\ i\ ..\ u\ i\})$
$incseq\ u\ decseq\ l\ \bigwedge i.\ l\ i < u\ i\ \bigwedge i.\ a < l\ i\ \bigwedge i.\ u\ i < b$
$l \longrightarrow a\ u \longrightarrow b$
assumes *f-integrable*: *set-integrable lborel* (*einterval a b*) *f*
shows (λi. *LBINT* x=$l\ i$.. $u\ i$. $f\ x$) $\longrightarrow$ (*LBINT* x=$a..b$. $f\ x$)
⟨*proof*⟩

lemma *interval-integral-FTC-finite*:
fixes $f\ F :: real \Rightarrow 'a$::*euclidean-space* **and** $a\ b :: real$
assumes f: *continuous-on* $\{min\ a\ b..max\ a\ b\}\ f$
assumes F: $\bigwedge x.\ min\ a\ b \le x \Longrightarrow x \le max\ a\ b \Longrightarrow$ (F *has-vector-derivative* (f x)) (*at* x *within*
$\{min\ a\ b..max\ a\ b\}$)
shows (*LBINT* x=$a..b$. $f\ x$) $= F\ b - F\ a$
⟨*proof*⟩

lemma *interval-integral-FTC-nonneg*:
fixes $f\ F :: real \Rightarrow real$ **and** $a\ b :: ereal$
assumes $a < b$
assumes F: $\bigwedge x.\ a <$ *ereal* $x \Longrightarrow$ *ereal* $x < b \Longrightarrow$ *DERIV* $F\ x$:> $f\ x$
assumes f: $\bigwedge x.\ a <$ *ereal* $x \Longrightarrow$ *ereal* $x < b \Longrightarrow$ *isCont* $f\ x$
assumes *f-nonneg*: *AE x in lborel*. $a <$ *ereal* $x \longrightarrow$ *ereal* $x < b \longrightarrow 0 \le f\ x$
assumes A: (($F \circ$ *real-of-ereal*) $\longrightarrow A$) (*at-right* a)
assumes B: (($F \circ$ *real-of-ereal*) $\longrightarrow B$) (*at-left* b)

shows
set-integrable lborel (*einterval a b*) *f*
(*LBINT x=a..b. f x*) = *B* − *A*
⟨*proof*⟩

lemma *interval-integral-FTC-integrable*:
fixes *f F* :: *real* ⇒ *'a::euclidean-space* **and** *a b* :: *ereal*
assumes *a* < *b*
assumes *F*: ⋀*x*. *a* < *ereal x* ⟹ *ereal x* < *b* ⟹ (*F has-vector-derivative f x*) (*at x*)
assumes *f*: ⋀*x*. *a* < *ereal x* ⟹ *ereal x* < *b* ⟹ *isCont f x*
assumes *f-integrable*: *set-integrable lborel* (*einterval a b*) *f*
assumes *A*: ((*F* ∘ *real-of-ereal*) ⟶ *A*) (*at-right a*)
assumes *B*: ((*F* ∘ *real-of-ereal*) ⟶ *B*) (*at-left b*)
shows (*LBINT x=a..b. f x*) = *B* − *A*
⟨*proof*⟩

lemma *interval-integral-FTC2*:
fixes *a b c* :: *real* **and** *f* :: *real* ⇒ *'a::euclidean-space*
assumes *a* ≤ *c c* ≤ *b*
and *contf*: *continuous-on* {*a..b*} *f*
fixes *x* :: *real*
assumes *a* ≤ *x* **and** *x* ≤ *b*
shows ((λ*u*. *LBINT y=c..u. f y*) *has-vector-derivative* (*f x*)) (*at x within* {*a..b*})
⟨*proof*⟩

lemma *einterval-antiderivative*:
fixes *a b* :: *ereal* **and** *f* :: *real* ⇒ *'a::euclidean-space*
assumes *a* < *b* **and** *contf*: ⋀*x* :: *real*. *a* < *x* ⟹ *x* < *b* ⟹ *isCont f x*
shows ∃ *F*. ∀ *x* :: *real*. *a* < *x* ⟶ *x* < *b* ⟶ (*F has-vector-derivative f x*) (*at x*)
⟨*proof*⟩

lemma *interval-integral-substitution-finite*:
fixes *a b* :: *real* **and** *f* :: *real* ⇒ *'a::euclidean-space*
assumes *a* ≤ *b*
and *derivg*: ⋀*x*. *a* ≤ *x* ⟹ *x* ≤ *b* ⟹ (*g has-real-derivative* (*g' x*)) (*at x within* {*a..b*})
and *contf* : *continuous-on* (*g* ‘ {*a..b*}) *f*
and *contg'*: *continuous-on* {*a..b*} *g'*
shows *LBINT x=a..b. g' x* $*_R$ *f* (*g x*) = *LBINT y=g a..g b. f y*
⟨*proof*⟩

lemma *interval-integral-substitution-integrable*:

fixes f :: *real* ⇒ *'a::euclidean-space* **and** a b u v :: *ereal*
assumes $a < b$
and *deriv-g*: $\bigwedge x.\ a < ereal\ x \Longrightarrow ereal\ x < b \Longrightarrow DERIV\ g\ x :> g'\ x$
and *contf*: $\bigwedge x.\ a < ereal\ x \Longrightarrow ereal\ x < b \Longrightarrow isCont\ f\ (g\ x)$
and *contg'*: $\bigwedge x.\ a < ereal\ x \Longrightarrow ereal\ x < b \Longrightarrow isCont\ g'\ x$
and *g'-nonneg*: $\bigwedge x.\ a \leq ereal\ x \Longrightarrow ereal\ x \leq b \Longrightarrow 0 \leq g'\ x$
and *A*: $((ereal \circ g \circ real\text{-}of\text{-}ereal) \longrightarrow A)\ (at\text{-}right\ a)$
and *B*: $((ereal \circ g \circ real\text{-}of\text{-}ereal) \longrightarrow B)\ (at\text{-}left\ b)$
and *integrable*: *set-integrable lborel* $(einterval\ a\ b)\ (\lambda x.\ g'\ x *_R f\ (g\ x))$
and *integrable2*: *set-integrable lborel* $(einterval\ A\ B)\ (\lambda x.\ f\ x)$
shows $(LBINT\ x{=}A..B.\ f\ x) = (LBINT\ x{=}a..b.\ g'\ x *_R f\ (g\ x))$
⟨*proof*⟩

lemma *interval-integral-substitution-nonneg*:
fixes $f\ g\ g'$:: *real* ⇒ *real* **and** a b u v :: *ereal*
assumes $a < b$
and *deriv-g*: $\bigwedge x.\ a < ereal\ x \Longrightarrow ereal\ x < b \Longrightarrow DERIV\ g\ x :> g'\ x$
and *contf*: $\bigwedge x.\ a < ereal\ x \Longrightarrow ereal\ x < b \Longrightarrow isCont\ f\ (g\ x)$
and *contg'*: $\bigwedge x.\ a < ereal\ x \Longrightarrow ereal\ x < b \Longrightarrow isCont\ g'\ x$
and *f-nonneg*: $\bigwedge x.\ a < ereal\ x \Longrightarrow ereal\ x < b \Longrightarrow 0 \leq f\ (g\ x)$
and *g'-nonneg*: $\bigwedge x.\ a \leq ereal\ x \Longrightarrow ereal\ x \leq b \Longrightarrow 0 \leq g'\ x$
and *A*: $((ereal \circ g \circ real\text{-}of\text{-}ereal) \longrightarrow A)\ (at\text{-}right\ a)$
and *B*: $((ereal \circ g \circ real\text{-}of\text{-}ereal) \longrightarrow B)\ (at\text{-}left\ b)$
and *integrable-fg*: *set-integrable lborel* $(einterval\ a\ b)\ (\lambda x.\ f\ (g\ x) * g'\ x)$
shows
set-integrable lborel $(einterval\ A\ B)\ f$
$(LBINT\ x{=}A..B.\ f\ x) = (LBINT\ x{=}a..b.\ (f\ (g\ x) * g'\ x))$
⟨*proof*⟩

syntax
-complex-lebesgue-borel-integral :: *pttrn* ⇒ *real* ⇒ *complex*
((*2CLBINT* -. -) [*0*,*60*] *60*)

translations
CLBINT x. f == *CONST complex-lebesgue-integral CONST lborel* (λ*x. f*)

syntax
-complex-set-lebesgue-borel-integral :: *pttrn* ⇒ *real set* ⇒ *real* ⇒ *complex*
((*3CLBINT* -:-. -) [*0*,*60*,*61*] *60*)

translations
CLBINT x:A. f == *CONST complex-set-lebesgue-integral CONST lborel A* (λ*x. f*)

abbreviation *complex-interval-lebesgue-integral* ::
real measure ⇒ *ereal* ⇒ *ereal* ⇒ (*real* ⇒ *complex*) ⇒ *complex* **where**
complex-interval-lebesgue-integral M a b f ≡ *interval-lebesgue-integral M a b f*

abbreviation *complex-interval-lebesgue-integrable* ::
real measure ⇒ *ereal* ⇒ *ereal* ⇒ (*real* ⇒ *complex*) ⇒ *bool* **where**
complex-interval-lebesgue-integrable M a b f ≡ *interval-lebesgue-integrable M a b f*

syntax
-ascii-complex-interval-lebesgue-borel-integral :: *pttrn* ⇒ *ereal* ⇒ *ereal* ⇒ *real* ⇒ *complex*
((*4CLBINT* -=-..-. -) [*0*,*60*,*60*,*61*] *60*)

translations
CLBINT x=a..b. f == *CONST complex-interval-lebesgue-integral CONST lborel a b* (λ*x*. *f*)

lemma *interval-integral-norm*:
fixes *f* :: *real* ⇒ *'a* :: {*banach*, *second-countable-topology*}
shows *interval-lebesgue-integrable lborel a b f* ⟹ *a* ≤ *b* ⟹
norm (*LBINT t=a..b. f t*) ≤ *LBINT t=a..b. norm* (*f t*)
⟨*proof*⟩

lemma *interval-integral-norm2*:
interval-lebesgue-integrable lborel a b f ⟹
norm (*LBINT t=a..b. f t*) ≤ |*LBINT t=a..b. norm* (*f t*)|
⟨*proof*⟩

lemma *integral-cos*: *t* ≠ *0* ⟹ *LBINT x=a..b. cos* (*t* ∗ *x*) = *sin* (*t* ∗ *b*) / *t* − *sin* (*t* ∗ *a*) / *t*
⟨*proof*⟩

end

15 Integration by Substition

theory *Lebesgue-Integral-Substitution*
imports *Interval-Integral*
begin

lemma *nn-integral-substitution-aux*:
fixes *f* :: *real* ⇒ *ennreal*
assumes *Mf*: *f* ∈ *borel-measurable borel*
assumes *nonnegf*: ⋀*x*. *f x* ≥ *0*
assumes *derivg*: ⋀*x*. *x* ∈ {*a..b*} ⟹ (*g has-real-derivative g′ x*) (*at x*)
assumes *contg′*: *continuous-on* {*a..b*} *g′*
assumes *derivg-nonneg*: ⋀*x*. *x* ∈ {*a..b*} ⟹ *g′ x* ≥ *0*
assumes *a* < *b*
shows (∫⁺*x*. *f x* ∗ *indicator* {*g a..g b*} *x* ∂*lborel*) =

$(\int^+ x.\ f\ (g\ x) * g'\ x * \mathit{indicator}\ \{a..b\}\ x\ \partial \mathit{lborel})$
⟨*proof*⟩

lemma *nn-integral-substitution*:
fixes $f :: \mathit{real} \Rightarrow \mathit{real}$
assumes *Mf*[*measurable*]: *set-borel-measurable borel* $\{g\ a..g\ b\}\ f$
assumes *derivg*: $\bigwedge x.\ x \in \{a..b\} \Longrightarrow (g\ \mathit{has\text{-}real\text{-}derivative}\ g'\ x)\ (\mathit{at}\ x)$
assumes *contg'*: *continuous-on* $\{a..b\}\ g'$
assumes *derivg-nonneg*: $\bigwedge x.\ x \in \{a..b\} \Longrightarrow g'\ x \geq 0$
assumes $a \leq b$
shows $(\int^+ x.\ f\ x * \mathit{indicator}\ \{g\ a..g\ b\}\ x\ \partial \mathit{lborel}) =$
$(\int^+ x.\ f\ (g\ x) * g'\ x * \mathit{indicator}\ \{a..b\}\ x\ \partial \mathit{lborel})$
⟨*proof*⟩

lemma *integral-substitution*:
assumes *integrable*: *set-integrable lborel* $\{g\ a..g\ b\}\ f$
assumes *derivg*: $\bigwedge x.\ x \in \{a..b\} \Longrightarrow (g\ \mathit{has\text{-}real\text{-}derivative}\ g'\ x)\ (\mathit{at}\ x)$
assumes *contg'*: *continuous-on* $\{a..b\}\ g'$
assumes *derivg-nonneg*: $\bigwedge x.\ x \in \{a..b\} \Longrightarrow g'\ x \geq 0$
assumes $a \leq b$
shows *set-integrable lborel* $\{a..b\}\ (\lambda x.\ f\ (g\ x) * g'\ x)$
and $(\mathit{LBINT}\ x.\ f\ x * \mathit{indicator}\ \{g\ a..g\ b\}\ x) = (\mathit{LBINT}\ x.\ f\ (g\ x) * g'\ x * \mathit{indicator}\ \{a..b\}\ x)$
⟨*proof*⟩

lemma *interval-integral-substitution*:
assumes *integrable*: *set-integrable lborel* $\{g\ a..g\ b\}\ f$
assumes *derivg*: $\bigwedge x.\ x \in \{a..b\} \Longrightarrow (g\ \mathit{has\text{-}real\text{-}derivative}\ g'\ x)\ (\mathit{at}\ x)$
assumes *contg'*: *continuous-on* $\{a..b\}\ g'$
assumes *derivg-nonneg*: $\bigwedge x.\ x \in \{a..b\} \Longrightarrow g'\ x \geq 0$
assumes $a \leq b$
shows *set-integrable lborel* $\{a..b\}\ (\lambda x.\ f\ (g\ x) * g'\ x)$
and $(\mathit{LBINT}\ x{=}g\ a..g\ b.\ f\ x) = (\mathit{LBINT}\ x{=}a..b.\ f\ (g\ x) * g'\ x)$
⟨*proof*⟩

lemma *set-borel-integrable-singleton*[*simp*]:
set-integrable lborel $\{x\}\ (f :: \mathit{real} \Rightarrow \mathit{real})$
⟨*proof*⟩

end

16 Adhoc overloading of constants based on their types

theory *Adhoc-Overloading*
imports *Pure*
keywords *adhoc-overloading* :: *thy-decl* **and** *no-adhoc-overloading* :: *thy-decl*
begin

⟨*ML*⟩

end

17 Monad notation for arbitrary types

theory *Monad-Syntax*
imports *Main* ~~/*src*/*Tools*/*Adhoc-Overloading*
begin

We provide a convenient do-notation for monadic expressions well-known from Haskell. *Let* is printed specially in do-expressions.

consts
bind :: ['*a*, '*b* ⇒ '*c*] ⇒ '*d* (**infixr** ≫= *54*)

notation (*ASCII*)
bind (**infixr** >>= *54*)

abbreviation (*do-notation*)
bind-do :: ['*a*, '*b* ⇒ '*c*] ⇒ '*d*
where *bind-do* ≡ *bind*

notation (**output**)
bind-do (**infixr** ≫= *54*)

notation (*ASCII* **output**)
bind-do (**infixr** >>= *54*)

nonterminal *do-binds* **and** *do-bind*
syntax
-do-block :: *do-binds* ⇒ '*a* (*do* {//(*2* -)//} [*12*] *62*)
-do-bind :: [*pttrn*, '*a*] ⇒ *do-bind* ((*2*- ←/ -) *13*)
-do-let :: [*pttrn*, '*a*] ⇒ *do-bind* ((*2let* - =/ -) [*1000*, *13*] *13*)
-do-then :: '*a* ⇒ *do-bind* (- [*14*] *13*)
-do-final :: '*a* ⇒ *do-binds* (-)
-do-cons :: [*do-bind*, *do-binds*] ⇒ *do-binds* (-;//- [*13*, *12*] *12*)
-thenM :: ['*a*, '*b*] ⇒ '*c* (**infixr** ≫ *54*)

syntax (*ASCII*)
-do-bind :: [*pttrn*, '*a*] ⇒ *do-bind* ((*2*- <−/ -) *13*)
-thenM :: ['*a*, '*b*] ⇒ '*c* (**infixr** >> *54*)

translations
-do-block (*-do-cons* (*-do-then* *t*) (*-do-final* *e*))
⇌ *CONST* *bind-do* *t* (λ-. *e*)
-do-block (*-do-cons* (*-do-bind* *p* *t*) (*-do-final* *e*))

$\rightleftharpoons$ *CONST bind-do t* ($\lambda p.\ e$)
-do-block (*-do-cons* (*-do-let p t*) *bs*)
$\rightleftharpoons$ *let p* = *t in -do-block bs*
-do-block (*-do-cons b* (*-do-cons c cs*))
$\rightleftharpoons$ *-do-block* (*-do-cons b* (*-do-final* (*-do-block* (*-do-cons c cs*))))
-do-cons (*-do-let p t*) (*-do-final s*)
$\rightleftharpoons$ *-do-final* (*let p* = *t in s*)
-do-block (*-do-final e*) $\rightharpoonup$ *e*
($m \gg n$) $\rightharpoonup$ ($m \ggg$ (λ-. n))

adhoc-overloading
bind Set.bind Predicate.bind Option.bind List.bind

end

theory *Giry-Monad*
imports *Probability-Measure Lebesgue-Integral-Substitution ~~/src/HOL/Library/Monad-Syntax*
begin

18 Sub-probability spaces

locale *subprob-space* = *finite-measure* +
assumes *emeasure-space-le-1*: *emeasure M* (*space M*) $\leq$ *1*
assumes *subprob-not-empty*: *space M* $\neq$ {}

lemma *subprob-spaceI*[*Pure.intro*!]:
assumes ∗: *emeasure M* (*space M*) $\leq$ *1*
assumes *space M* $\neq$ {}
shows *subprob-space M*
⟨*proof*⟩

lemma *prob-space-imp-subprob-space*:
prob-space M $\Longrightarrow$ *subprob-space M*
⟨*proof*⟩

lemma *subprob-space-imp-sigma-finite*: *subprob-space M* $\Longrightarrow$ *sigma-finite-measure M*
⟨*proof*⟩

sublocale *prob-space* $\subseteq$ *subprob-space*
⟨*proof*⟩

lemma *subprob-space-sigma* [*simp*]: $\Omega \neq$ {} $\Longrightarrow$ *subprob-space* (*sigma* Ω *X*)
⟨*proof*⟩

lemma *subprob-space-null-measure*: *space M* $\neq$ {} $\Longrightarrow$ *subprob-space* (*null-measure M*)
⟨*proof*⟩

lemma (**in** *subprob-space*) *subprob-space-distr*:
assumes *f*: $f \in$ *measurable M M′* **and** *space M′* $\neq \{\}$ **shows** *subprob-space* (*distr M M′ f*)
⟨*proof*⟩

lemma (**in** *subprob-space*) *subprob-emeasure-le-1*: *emeasure M X* $\leq$ *1*
⟨*proof*⟩

lemma (**in** *subprob-space*) *subprob-measure-le-1*: *measure M X* $\leq$ *1*
⟨*proof*⟩

lemma (**in** *subprob-space*) *nn-integral-le-const*:
assumes *0* $\leq$ *c AE x in M*. *f x* $\leq$ *c*
shows ($\int^+ x.\ f\ x\ \partial M$) $\leq$ *c*
⟨*proof*⟩

lemma *emeasure-density-distr-interval*:
fixes *h* :: *real* ⇒ *real* **and** *g* :: *real* ⇒ *real* **and** *g′* :: *real* ⇒ *real*
assumes [*simp*]: *a* $\leq$ *b*
assumes *Mf*[*measurable*]: *f* $\in$ *borel-measurable borel*
assumes *Mg*[*measurable*]: *g* $\in$ *borel-measurable borel*
assumes *Mg′*[*measurable*]: *g′* $\in$ *borel-measurable borel*
assumes *Mh*[*measurable*]: *h* $\in$ *borel-measurable borel*
assumes *prob*: *subprob-space* (*density lborel f*)
assumes *nonnegf*: $\bigwedge x.\ f\ x \geq 0$
assumes *derivg*: $\bigwedge x.\ x \in \{a..b\} \Longrightarrow$ (*g has-real-derivative g′ x*) (*at x*)
assumes *contg′*: *continuous-on* $\{a..b\}$ *g′*
assumes *mono*: *strict-mono-on g* $\{a..b\}$ **and** *inv*: $\bigwedge x.\ h\ x \in \{a..b\} \Longrightarrow g\ (h\ x) = x$
assumes *range*: $\{a..b\} \subseteq$ *range h*
shows *emeasure* (*distr* (*density lborel f*) *lborel h*) $\{a..b\}$ =
emeasure (*density lborel* ($\lambda x.\ f\ (g\ x) * g'\ x$)) $\{a..b\}$
⟨*proof*⟩

locale *pair-subprob-space* =
pair-sigma-finite M1 M2 + *M1*: *subprob-space M1* + *M2*: *subprob-space M2* **for** *M1 M2*

sublocale *pair-subprob-space* ⊆ *P?*: *subprob-space M1* $\bigotimes_M$ *M2*
⟨*proof*⟩

lemma *subprob-space-null-measure-iff*:
subprob-space (*null-measure M*) ⟷ *space M* $\neq \{\}$
⟨*proof*⟩

lemma *subprob-space-restrict-space*:
assumes *M*: *subprob-space M*
and *A*: *A* ∩ *space M* ∈ *sets M A* ∩ *space M* $\neq \{\}$

shows *subprob-space* (*restrict-space* M A)
⟨*proof*⟩

definition *subprob-algebra* :: *'a measure* ⇒ *'a measure measure* **where**
subprob-algebra K =
($\bigsqcup_\sigma$ A∈*sets* K. *vimage-algebra* {M. *subprob-space* M ∧ *sets* M = *sets* K} (λM. *emeasure* M A) *borel*)

lemma *space-subprob-algebra*: *space* (*subprob-algebra* A) = {M. *subprob-space* M ∧ *sets* M = *sets* A}
⟨*proof*⟩

lemma *subprob-algebra-cong*: *sets* M = *sets* N ⟹ *subprob-algebra* M = *subprob-algebra* N
⟨*proof*⟩

lemma *measurable-emeasure-subprob-algebra*[*measurable*]:
a ∈ *sets* A ⟹ (λM. *emeasure* M a) ∈ *borel-measurable* (*subprob-algebra* A)
⟨*proof*⟩

lemma *measurable-measure-subprob-algebra*[*measurable*]:
a ∈ *sets* A ⟹ (λM. *measure* M a) ∈ *borel-measurable* (*subprob-algebra* A)
⟨*proof*⟩

lemma *subprob-measurableD*:
assumes N: N ∈ *measurable* M (*subprob-algebra* S) **and** x: x ∈ *space* M
shows *space* (N x) = *space* S
and *sets* (N x) = *sets* S
and *measurable* (N x) K = *measurable* S K
and *measurable* K (N x) = *measurable* K S
⟨*proof*⟩

⟨*ML*⟩

context
fixes K M N **assumes** K: K ∈ *measurable* M (*subprob-algebra* N)
begin

lemma *subprob-space-kernel*: a ∈ *space* M ⟹ *subprob-space* (K a)
⟨*proof*⟩

lemma *sets-kernel*: a ∈ *space* M ⟹ *sets* (K a) = *sets* N
⟨*proof*⟩

lemma *measurable-emeasure-kernel*[*measurable*]:
A ∈ *sets* N ⟹ (λa. *emeasure* (K a) A) ∈ *borel-measurable* M
⟨*proof*⟩

end

lemma *measurable-subprob-algebra*:
$(\bigwedge a.\ a \in \mathit{space}\ M \Longrightarrow \mathit{subprob\text{-}space}\ (K\ a)) \Longrightarrow$
$(\bigwedge a.\ a \in \mathit{space}\ M \Longrightarrow \mathit{sets}\ (K\ a) = \mathit{sets}\ N) \Longrightarrow$
$(\bigwedge A.\ A \in \mathit{sets}\ N \Longrightarrow (\lambda a.\ \mathit{emeasure}\ (K\ a)\ A) \in \mathit{borel\text{-}measurable}\ M) \Longrightarrow$
$K \in \mathit{measurable}\ M\ (\mathit{subprob\text{-}algebra}\ N)$
⟨*proof*⟩

lemma *measurable-submarkov*:
$K \in \mathit{measurable}\ M\ (\mathit{subprob\text{-}algebra}\ M) \longleftrightarrow$
$(\forall x{\in}\mathit{space}\ M.\ \mathit{subprob\text{-}space}\ (K\ x) \wedge \mathit{sets}\ (K\ x) = \mathit{sets}\ M) \wedge$
$(\forall A{\in}\mathit{sets}\ M.\ (\lambda x.\ \mathit{emeasure}\ (K\ x)\ A) \in \mathit{measurable}\ M\ \mathit{borel})$
⟨*proof*⟩

lemma *space-subprob-algebra-empty-iff*:
$\mathit{space}\ (\mathit{subprob\text{-}algebra}\ N) = \{\} \longleftrightarrow \mathit{space}\ N = \{\}$
⟨*proof*⟩

lemma *nn-integral-measurable-subprob-algebra*[*measurable*]:
assumes *f*: $f \in \mathit{borel\text{-}measurable}\ N$
shows $(\lambda M.\ \mathit{integral}^N\ M\ f) \in \mathit{borel\text{-}measurable}\ (\mathit{subprob\text{-}algebra}\ N)$ (**is** - ∈ *?B*)
⟨*proof*⟩

lemma *measurable-distr*:
assumes [*measurable*]: $f \in \mathit{measurable}\ M\ N$
shows $(\lambda M'.\ \mathit{distr}\ M'\ N\ f) \in \mathit{measurable}\ (\mathit{subprob\text{-}algebra}\ M)\ (\mathit{subprob\text{-}algebra}\ N)$
⟨*proof*⟩

lemma *emeasure-space-subprob-algebra*[*measurable*]:
$(\lambda a.\ \mathit{emeasure}\ a\ (\mathit{space}\ a)) \in \mathit{borel\text{-}measurable}\ (\mathit{subprob\text{-}algebra}\ N)$
⟨*proof*⟩

lemma *integrable-measurable-subprob-algebra*[*measurable*]:
fixes $f :: 'a \Rightarrow 'b{::}\{\mathit{banach},\ \mathit{second\text{-}countable\text{-}topology}\}$
assumes [*measurable*]: $f \in \mathit{borel\text{-}measurable}\ N$
shows *Measurable.pred* $(\mathit{subprob\text{-}algebra}\ N)\ (\lambda M.\ \mathit{integrable}\ M\ f)$
⟨*proof*⟩

lemma *integral-measurable-subprob-algebra*[*measurable*]:
fixes $f :: 'a \Rightarrow 'b{::}\{\mathit{banach},\ \mathit{second\text{-}countable\text{-}topology}\}$
assumes *f* [*measurable*]: $f \in \mathit{borel\text{-}measurable}\ N$
shows $(\lambda M.\ \mathit{integral}^L\ M\ f) \in \mathit{subprob\text{-}algebra}\ N \rightarrow_M \mathit{borel}$
⟨*proof*⟩

lemma *measurable-pair-measure*:
assumes *f*: $f \in \mathit{measurable}\ M\ (\mathit{subprob\text{-}algebra}\ N)$
assumes *g*: $g \in \mathit{measurable}\ M\ (\mathit{subprob\text{-}algebra}\ L)$

shows $(\lambda x.\ f\ x \bigotimes_M\ g\ x) \in$ *measurable M* (*subprob-algebra* $(N \bigotimes_M L)$)
⟨*proof*⟩

lemma *restrict-space-measurable*:
assumes X: $X \neq \{\}$ $X \in$ *sets K*
assumes N: $N \in$ *measurable M* (*subprob-algebra K*)
shows $(\lambda x.$ *restrict-space* $(N\,x)\ X) \in$ *measurable M* (*subprob-algebra* (*restrict-space K X*))
⟨*proof*⟩

19 Properties of return

definition *return* :: *'a measure* ⇒ *'a* ⇒ *'a measure* **where**
return R x = *measure-of* (*space R*) (*sets R*) (λA. *indicator A x*)

lemma *space-return*[*simp*]: *space* (*return M x*) = *space M*
⟨*proof*⟩

lemma *sets-return*[*simp*]: *sets* (*return M x*) = *sets M*
⟨*proof*⟩

lemma *measurable-return1*[*simp*]: *measurable* (*return N x*) *L* = *measurable N L*
⟨*proof*⟩

lemma *measurable-return2*[*simp*]: *measurable L* (*return N x*) = *measurable L N*
⟨*proof*⟩

lemma *return-sets-cong*: *sets M* = *sets N* ⟹ *return M* = *return N*
⟨*proof*⟩

lemma *return-cong*: *sets A* = *sets B* ⟹ *return A x* = *return B x*
⟨*proof*⟩

lemma *emeasure-return*[*simp*]:
assumes $A \in$ *sets M*
shows *emeasure* (*return M x*) *A* = *indicator A x*
⟨*proof*⟩

lemma *prob-space-return*: $x \in$ *space M* ⟹ *prob-space* (*return M x*)
⟨*proof*⟩

lemma *subprob-space-return*: $x \in$ *space M* ⟹ *subprob-space* (*return M x*)
⟨*proof*⟩

lemma *subprob-space-return-ne*:
assumes *space M* $\neq \{\}$ **shows** *subprob-space* (*return M x*)
⟨*proof*⟩

lemma *measure-return*: **assumes** X: $X \in$ *sets M* **shows** *measure* (*return M x*)

X = *indicator X x*
⟨*proof*⟩

lemma *AE-return*:
assumes [*simp*]: $x \in$ *space M* **and** [*measurable*]: *Measurable.pred M P*
shows (*AE y in return M x. P y*) $\longleftrightarrow$ *P x*
⟨*proof*⟩

lemma *nn-integral-return*:
assumes $x \in$ *space M* $g \in$ *borel-measurable M*
shows ($\int^+$ *a. g a* ∂*return M x*) = *g x*
⟨*proof*⟩

lemma *integral-return*:
fixes *g* :: - ⇒ *'a* :: {*banach*, *second-countable-topology*}
assumes $x \in$ *space M* $g \in$ *borel-measurable M*
shows ($\int$ *a. g a* ∂*return M x*) = *g x*
⟨*proof*⟩

lemma *return-measurable*[*measurable*]: *return N* $\in$ *measurable N* (*subprob-algebra N*)
⟨*proof*⟩

lemma *distr-return*:
assumes $f \in$ *measurable M N* **and** $x \in$ *space M*
shows *distr* (*return M x*) *N f* = *return N* (*f x*)
⟨*proof*⟩

lemma *return-restrict-space*:
$\Omega \in$ *sets M* $\Longrightarrow$ *return* (*restrict-space M* Ω) *x* = *restrict-space* (*return M x*) Ω
⟨*proof*⟩

lemma *measurable-distr2*:
assumes *f*[*measurable*]: *case-prod f* $\in$ *measurable* ($L \bigotimes_M M$) *N*
assumes *g*[*measurable*]: $g \in$ *measurable L* (*subprob-algebra M*)
shows (λx. *distr* (*g x*) *N* (*f x*)) $\in$ *measurable L* (*subprob-algebra N*)
⟨*proof*⟩

lemma *nn-integral-measurable-subprob-algebra2*:
assumes *f*[*measurable*]: (λ(*x*, *y*). *f x y*) $\in$ *borel-measurable* ($M \bigotimes_M N$)
assumes *N*[*measurable*]: $L \in$ *measurable M* (*subprob-algebra N*)
shows (λx. *integral*^{N} (*L x*) (*f x*)) $\in$ *borel-measurable M*
⟨*proof*⟩

lemma *emeasure-measurable-subprob-algebra2*:
assumes *A*[*measurable*]: (*SIGMA x*:*space M*. *A x*) $\in$ *sets* ($M \bigotimes_M N$)
assumes *L*[*measurable*]: $L \in$ *measurable M* (*subprob-algebra N*)
shows (λx. *emeasure* (*L x*) (*A x*)) $\in$ *borel-measurable M*
⟨*proof*⟩

lemma *measure-measurable-subprob-algebra2*:
assumes *A*[*measurable*]: (*SIGMA x*:*space M*. *A x*) ∈ *sets* ($M \bigotimes_M N$)
assumes *L*[*measurable*]: *L* ∈ *measurable M* (*subprob-algebra N*)
shows (*λx*. *measure* (*L x*) (*A x*)) ∈ *borel-measurable M*
⟨*proof*⟩

definition *select-sets M* = (*SOME N*. *sets M* = *sets* (*subprob-algebra N*))

lemma *select-sets1*:
sets M = *sets* (*subprob-algebra N*) ⟹ *sets M* = *sets* (*subprob-algebra* (*select-sets M*))
⟨*proof*⟩

lemma *sets-select-sets*[*simp*]:
assumes *sets*: *sets M* = *sets* (*subprob-algebra N*)
shows *sets* (*select-sets M*) = *sets N*
⟨*proof*⟩

lemma *space-select-sets*[*simp*]:
sets M = *sets* (*subprob-algebra N*) ⟹ *space* (*select-sets M*) = *space N*
⟨*proof*⟩

20 Join

definition *join* :: *'a measure measure* ⇒ *'a measure* **where**
join M = *measure-of* (*space* (*select-sets M*)) (*sets* (*select-sets M*)) (*λB*. $\int^+$ *M'*. *emeasure M' B* ∂*M*)

lemma
shows *space-join*[*simp*]: *space* (*join M*) = *space* (*select-sets M*)
and *sets-join*[*simp*]: *sets* (*join M*) = *sets* (*select-sets M*)
⟨*proof*⟩

lemma *emeasure-join*:
assumes *M*[*simp*, *measurable-cong*]: *sets M* = *sets* (*subprob-algebra N*) **and** *A*: *A* ∈ *sets N*
shows *emeasure* (*join M*) *A* = ($\int^+$ *M'*. *emeasure M' A* ∂*M*)
⟨*proof*⟩

lemma *measurable-join*:
join ∈ *measurable* (*subprob-algebra* (*subprob-algebra N*)) (*subprob-algebra N*)
⟨*proof*⟩

lemma *nn-integral-join*:
assumes *f*: *f* ∈ *borel-measurable N*
and *M*[*measurable-cong*]: *sets M* = *sets* (*subprob-algebra N*)
shows ($\int^+$*x*. *f x* ∂*join M*) = ($\int^+$*M'*. $\int^+$*x*. *f x* ∂*M'* ∂*M*)
⟨*proof*⟩

lemma *measurable-join1*:
  $[\![f \in measurable\ N\ K;\ sets\ M = sets\ (subprob\text{-}algebra\ N)]\!]$
  $\Longrightarrow f \in measurable\ (join\ M)\ K$
⟨*proof*⟩

lemma
  fixes $f :: - \Rightarrow real$
  assumes *f-measurable* [*measurable*]: $f \in borel\text{-}measurable\ N$
  and *f-bounded*: $\bigwedge x.\ x \in space\ N \Longrightarrow |f\ x| \le B$
  and *M* [*measurable-cong*]: $sets\ M = sets\ (subprob\text{-}algebra\ N)$
  and *fin*: *finite-measure M*
  and *M-bounded*: $AE\ M'\ in\ M.\ emeasure\ M'\ (space\ M') \le ennreal\ B'$
  shows *integrable-join*: *integrable* (*join M*) *f* (**is** *?integrable*)
  and *integral-join*: $integral^L\ (join\ M)\ f = \int M'.\ integral^L\ M'\ f\ \partial M$ (**is** *?integral*)
⟨*proof*⟩

lemma *join-assoc*:
  assumes *M*[*measurable-cong*]: $sets\ M = sets\ (subprob\text{-}algebra\ (subprob\text{-}algebra\ N))$
  shows $join\ (distr\ M\ (subprob\text{-}algebra\ N)\ join) = join\ (join\ M)$
⟨*proof*⟩

lemma *join-return*:
  assumes $sets\ M = sets\ N$ **and** *subprob-space M*
  shows $join\ (return\ (subprob\text{-}algebra\ N)\ M) = M$
  ⟨*proof*⟩

lemma *join-return'*:
  assumes $sets\ N = sets\ M$
  shows $join\ (distr\ M\ (subprob\text{-}algebra\ N)\ (return\ N)) = M$
⟨*proof*⟩

lemma *join-distr-distr*:
  fixes $f :: 'a \Rightarrow 'b$ **and** $M :: 'a\ measure\ measure$ **and** $N :: 'b\ measure$
  assumes $sets\ M = sets\ (subprob\text{-}algebra\ R)$ **and** $f \in measurable\ R\ N$
  shows $join\ (distr\ M\ (subprob\text{-}algebra\ N)\ (\lambda M.\ distr\ M\ N\ f)) = distr\ (join\ M)\ N\ f$ (**is** *?r = ?l*)
⟨*proof*⟩

definition $bind :: 'a\ measure \Rightarrow ('a \Rightarrow 'b\ measure) \Rightarrow 'b\ measure$ **where**
  $bind\ M\ f = (if\ space\ M = \{\}\ then\ count\text{-}space\ \{\}\ else$
    $join\ (distr\ M\ (subprob\text{-}algebra\ (f\ (SOME\ x.\ x \in space\ M)))\ f))$

adhoc-overloading *Monad-Syntax.bind bind*

lemma *bind-empty*:
  $space\ M = \{\} \Longrightarrow bind\ M\ f = count\text{-}space\ \{\}$
  ⟨*proof*⟩

lemma *bind-nonempty*:
space M $\neq$ {} $\Longrightarrow$ *bind M f* = *join* (*distr M* (*subprob-algebra* (*f* (*SOME x*. *x* $\in$ *space M*))) *f*)
⟨*proof*⟩

lemma *sets-bind-empty*: *sets M* = {} $\Longrightarrow$ *sets* (*bind M f*) = {{}}
⟨*proof*⟩

lemma *space-bind-empty*: *space M* = {} $\Longrightarrow$ *space* (*bind M f*) = {}
⟨*proof*⟩

lemma *sets-bind*[*simp*, *measurable-cong*]:
assumes *f*: $\bigwedge$*x*. *x* $\in$ *space M* $\Longrightarrow$ *sets* (*f x*) = *sets N* **and** *M*: *space M* $\neq$ {}
shows *sets* (*bind M f*) = *sets N*
⟨*proof*⟩

lemma *space-bind*[*simp*]:
assumes $\bigwedge$*x*. *x* $\in$ *space M* $\Longrightarrow$ *sets* (*f x*) = *sets N* **and** *space M* $\neq$ {}
shows *space* (*bind M f*) = *space N*
⟨*proof*⟩

lemma *bind-cong*:
assumes $\forall$ *x* $\in$ *space M*. *f x* = *g x*
shows *bind M f* = *bind M g*
⟨*proof*⟩

lemma *bind-nonempty′*:
assumes *f* $\in$ *measurable M* (*subprob-algebra N*) *x* $\in$ *space M*
shows *bind M f* = *join* (*distr M* (*subprob-algebra N*) *f*)
⟨*proof*⟩

lemma *bind-nonempty″*:
assumes *f* $\in$ *measurable M* (*subprob-algebra N*) *space M* $\neq$ {}
shows *bind M f* = *join* (*distr M* (*subprob-algebra N*) *f*)
⟨*proof*⟩

lemma *emeasure-bind*:
⟦*space M* $\neq$ {}; *f* $\in$ *measurable M* (*subprob-algebra N*);*X* $\in$ *sets N*⟧
$\Longrightarrow$ *emeasure* (*M* ⋙ *f*) *X* = $\int^+$*x*. *emeasure* (*f x*) *X* ∂*M*
⟨*proof*⟩

lemma *nn-integral-bind*:
assumes *f*: *f* $\in$ *borel-measurable B*
assumes *N*: *N* $\in$ *measurable M* (*subprob-algebra B*)
shows ($\int^+$*x*. *f x* ∂(*M* ⋙ *N*)) = ($\int^+$*x*. $\int^+$*y*. *f y* ∂*N x* ∂*M*)
⟨*proof*⟩

lemma *AE-bind*:

assumes $P[measurable]$: $Measurable.pred\ B\ P$
assumes $N[measurable]$: $N \in measurable\ M\ (subprob\text{-}algebra\ B)$
shows $(AE\ x\ in\ M \gg= N.\ P\ x) \longleftrightarrow (AE\ x\ in\ M.\ AE\ y\ in\ N\ x.\ P\ y)$
⟨*proof*⟩

lemma *measurable-bind'*:
assumes $M1$: $f \in measurable\ M\ (subprob\text{-}algebra\ N)$ **and**
$M2$: $case\text{-}prod\ g \in measurable\ (M \bigotimes_M N)\ (subprob\text{-}algebra\ R)$
shows $(\lambda x.\ bind\ (f\ x)\ (g\ x)) \in measurable\ M\ (subprob\text{-}algebra\ R)$
⟨*proof*⟩

lemma *measurable-bind*[*measurable* (*raw*)]:
assumes $M1$: $f \in measurable\ M\ (subprob\text{-}algebra\ N)$ **and**
$M2$: $(\lambda x.\ g\ (fst\ x)\ (snd\ x)) \in measurable\ (M \bigotimes_M N)\ (subprob\text{-}algebra\ R)$
shows $(\lambda x.\ bind\ (f\ x)\ (g\ x)) \in measurable\ M\ (subprob\text{-}algebra\ R)$
⟨*proof*⟩

lemma *measurable-bind2*:
assumes $f \in measurable\ M\ (subprob\text{-}algebra\ N)$ **and** $g \in measurable\ N\ (subprob\text{-}algebra\ R)$
shows $(\lambda x.\ bind\ (f\ x)\ g) \in measurable\ M\ (subprob\text{-}algebra\ R)$
⟨*proof*⟩

lemma *subprob-space-bind*:
assumes $subprob\text{-}space\ M\ f \in measurable\ M\ (subprob\text{-}algebra\ N)$
shows $subprob\text{-}space\ (M \gg= f)$
⟨*proof*⟩

lemma
fixes f :: - $\Rightarrow$ *real*
assumes *f-measurable* [*measurable*]: $f \in borel\text{-}measurable\ K$
and *f-bounded*: $\bigwedge x.\ x \in space\ K \Longrightarrow |f\ x| \leq B$
and N [*measurable*]: $N \in measurable\ M\ (subprob\text{-}algebra\ K)$
and *fin*: *finite-measure M*
and *M-bounded*: $AE\ x\ in\ M.\ emeasure\ (N\ x)\ (space\ (N\ x)) \leq ennreal\ B'$
shows *integrable-bind*: *integrable* (*bind M N*) *f* (**is** *?integrable*)
and *integral-bind*: $integral^L\ (bind\ M\ N)\ f = \int\ x.\ integral^L\ (N\ x)\ f\ \partial M$ (**is** *?integral*)
⟨*proof*⟩

lemma (**in** *prob-space*) *prob-space-bind*:
assumes *ae*: $AE\ x\ in\ M.\ prob\text{-}space\ (N\ x)$
and $N[measurable]$: $N \in measurable\ M\ (subprob\text{-}algebra\ S)$
shows $prob\text{-}space\ (M \gg= N)$
⟨*proof*⟩

lemma (**in** *subprob-space*) *bind-in-space*:
$A \in measurable\ M\ (subprob\text{-}algebra\ N) \Longrightarrow (M \gg= A) \in space\ (subprob\text{-}algebra\ N)$

⟨*proof*⟩

lemma (**in** *subprob-space*) *measure-bind*:
assumes f: $f \in$ *measurable* M (*subprob-algebra* N) **and** X: $X \in$ *sets* N
shows *measure* ($M \gg= f$) $X = \int x.$ *measure* ($f\ x$) $X\ \partial M$
⟨*proof*⟩

lemma *emeasure-bind-const*:
space $M \neq \{\} \Longrightarrow X \in$ *sets* $N \Longrightarrow$ *subprob-space* $N \Longrightarrow$
emeasure ($M \gg= (\lambda x.\ N)$) $X =$ *emeasure* $N\ X *$ *emeasure* M (*space* M)
⟨*proof*⟩

lemma *emeasure-bind-const′*:
assumes *subprob-space* M *subprob-space* N
shows *emeasure* ($M \gg= (\lambda x.\ N)$) $X =$ *emeasure* $N\ X *$ *emeasure* M (*space* M)
⟨*proof*⟩

lemma *emeasure-bind-const-prob-space*:
assumes *prob-space* M *subprob-space* N
shows *emeasure* ($M \gg= (\lambda x.\ N)$) $X =$ *emeasure* $N\ X$
⟨*proof*⟩

lemma *bind-return*:
assumes $f \in$ *measurable* M (*subprob-algebra* N) **and** $x \in$ *space* M
shows *bind* (*return* $M\ x$) $f = f\ x$
⟨*proof*⟩

lemma *bind-return′*:
shows *bind* M (*return* M) $= M$
⟨*proof*⟩

lemma *distr-bind*:
assumes N: $N \in$ *measurable* M (*subprob-algebra* K) *space* $M \neq \{\}$
assumes f: $f \in$ *measurable* $K\ R$
shows *distr* ($M \gg= N$) $R\ f = (M \gg= (\lambda x.$ *distr* ($N\ x$) $R\ f$))
⟨*proof*⟩

lemma *bind-distr*:
assumes f[*measurable*]: $f \in$ *measurable* $M\ X$
assumes N[*measurable*]: $N \in$ *measurable* X (*subprob-algebra* K) **and** *space* M
$\neq \{\}$
shows (*distr* $M\ X\ f \gg= N$) $= (M \gg= (\lambda x.\ N\ (f\ x)))$
⟨*proof*⟩

lemma *bind-count-space-singleton*:
assumes *subprob-space* ($f\ x$)
shows *count-space* $\{x\} \gg= f = f\ x$
⟨*proof*⟩

lemma *restrict-space-bind*:
assumes *N*: $N \in$ *measurable M* (*subprob-algebra K*)
assumes *space M* $\neq$ {}
assumes *X*[*simp*]: $X \in$ *sets K X* $\neq$ {}
shows *restrict-space* (*bind M N*) *X* = *bind M* (λx. *restrict-space* (*N x*) *X*)
⟨*proof*⟩

lemma *bind-restrict-space*:
assumes *A*: $A \cap$ *space M* $\neq$ {} $A \cap$ *space M* $\in$ *sets M*
and *f*: $f \in$ *measurable* (*restrict-space M A*) (*subprob-algebra N*)
shows *restrict-space M A* ≫= *f* = *M* ≫= (λx. *if* $x \in A$ *then f x else null-measure* (*f* (*SOME x*. $x \in A \wedge x \in$ *space M*)))
(**is** *?lhs* = *?rhs* **is** - = *M* ≫= *?f*)
⟨*proof*⟩

lemma *bind-const'*: ⟦*prob-space M*; *subprob-space N*⟧ ⟹ *M* ≫= (λx. *N*) = *N*
⟨*proof*⟩

lemma *bind-return-distr*:
space M $\neq$ {} ⟹ $f \in$ *measurable M N* ⟹ *bind M* (*return N* ∘ *f*) = *distr M N f*
⟨*proof*⟩

lemma *bind-return-distr'*:
space M $\neq$ {} ⟹ $f \in$ *measurable M N* ⟹ *bind M* (λx. *return N* (*f x*)) = *distr M N f*
⟨*proof*⟩

lemma *bind-assoc*:
fixes *f* :: *'a* ⇒ *'b measure* **and** *g* :: *'b* ⇒ *'c measure*
assumes *M1*: $f \in$ *measurable M* (*subprob-algebra N*) **and** *M2*: $g \in$ *measurable N* (*subprob-algebra R*)
shows *bind* (*bind M f*) *g* = *bind M* (λx. *bind* (*f x*) *g*)
⟨*proof*⟩

lemma *double-bind-assoc*:
assumes *Mg*: $g \in$ *measurable N* (*subprob-algebra N'*)
assumes *Mf*: $f \in$ *measurable M* (*subprob-algebra M'*)
assumes *Mh*: *case-prod h* $\in$ *measurable* (*M* $\bigotimes_M$ *M'*) *N*
shows *do* {*x* ← *M*; *y* ← *f x*; *g* (*h x y*)} = *do* {*x* ← *M*; *y* ← *f x*; *return N* (*h x y*)} ≫= *g*
⟨*proof*⟩

lemma (**in** *prob-space*) *M-in-subprob*[*measurable* (*raw*)]: $M \in$ *space* (*subprob-algebra M*)
⟨*proof*⟩

lemma (**in** *pair-prob-space*) *pair-measure-eq-bind*:
(*M1* $\bigotimes_M$ *M2*) = (*M1* ≫= (λx. *M2* ≫= (λy. *return* (*M1* $\bigotimes_M$ *M2*) (*x*, *y*))))

⟨*proof*⟩

lemma (**in** *pair-prob-space*) *bind-rotate*:
assumes *C*[*measurable*]: $(\lambda(x, y).\ C\ x\ y) \in$ *measurable* $(M1 \bigotimes_M M2)$ (*subprob-algebra* *N*)
shows $(M1 \gg= (\lambda x.\ M2 \gg= (\lambda y.\ C\ x\ y))) = (M2 \gg= (\lambda y.\ M1 \gg= (\lambda x.\ C\ x\ y)))$
⟨*proof*⟩

21 Measures form a ω-chain complete partial order

definition *SUP-measure* :: $(nat \Rightarrow 'a\ measure) \Rightarrow 'a\ measure$ **where**
SUP-measure M = *measure-of* $(\bigcup i.\ space\ (M\ i))\ (\bigcup i.\ sets\ (M\ i))\ (\lambda A.\ SUP\ i.\ emeasure\ (M\ i)\ A)$

lemma
assumes *const*: $\bigwedge i\ j.\ sets\ (M\ i) = sets\ (M\ j)$
shows *space-SUP-measure*: *space* (*SUP-measure* M) = *space* $(M\ i)$ (**is** *?sp*)
and *sets-SUP-measure*: *sets* (*SUP-measure* M) = *sets* $(M\ i)$ (**is** *?st*)
⟨*proof*⟩

lemma *emeasure-SUP-measure*:
assumes *const*: $\bigwedge i\ j.\ sets\ (M\ i) = sets\ (M\ j)$
and *mono*: *mono* $(\lambda i.\ emeasure\ (M\ i))$
shows *emeasure* (*SUP-measure* M) $A = (SUP\ i.\ emeasure\ (M\ i)\ A)$
⟨*proof*⟩

lemma *bind-return''*: $sets\ M = sets\ N \Longrightarrow M \gg= return\ N = M$
⟨*proof*⟩

lemma (**in** *prob-space*) *distr-const*[*simp*]:
$c \in space\ N \Longrightarrow distr\ M\ N\ (\lambda x.\ c) = return\ N\ c$
⟨*proof*⟩

lemma *return-count-space-eq-density*:
return (*count-space* M) x = *density* (*count-space* M) (*indicator* $\{x\}$)
⟨*proof*⟩

lemma *null-measure-in-space-subprob-algebra* [*simp*]:
null-measure $M \in$ *space* (*subprob-algebra* M) $\longleftrightarrow$ *space* $M \neq \{\}$
⟨*proof*⟩

end

22 Projective Family

theory *Projective-Family*

imports *Finite-Product-Measure Giry-Monad*
begin

lemma *vimage-restrict-preseve-mono*:
assumes *J*: $J \subseteq I$
and *sets*: $A \subseteq (\Pi_E\ i{\in}J.\ S\ i)$ $B \subseteq (\Pi_E\ i{\in}J.\ S\ i)$ **and** *ne*: $(\Pi_E\ i{\in}I.\ S\ i) \neq \{\}$
and *eq*: $(\lambda x.\ restrict\ x\ J)$ −‘ $A \cap (\Pi_E\ i{\in}I.\ S\ i) \subseteq (\lambda x.\ restrict\ x\ J)$ −‘ $B \cap (\Pi_E\ i{\in}I.\ S\ i)$
shows $A \subseteq B$
⟨*proof*⟩

locale *projective-family* =
fixes I :: *'i set* **and** P :: *'i set* ⇒ (*'i* ⇒ *'a*) *measure* **and** M :: *'i* ⇒ *'a measure*
assumes *P*: $\bigwedge J\ H.\ J \subseteq H \Longrightarrow$ *finite* $H \Longrightarrow H \subseteq I \Longrightarrow P\ J =$ *distr* $(P\ H)$ $(PiM\ J\ M)$ $(\lambda f.\ restrict\ f\ J)$
assumes *prob-space-P*: $\bigwedge J.$ *finite* $J \Longrightarrow J \subseteq I \Longrightarrow$ *prob-space* $(P\ J)$
begin

lemma *sets-P*: *finite* $J \Longrightarrow J \subseteq I \Longrightarrow$ *sets* $(P\ J) =$ *sets* $(PiM\ J\ M)$
⟨*proof*⟩

lemma *space-P*: *finite* $J \Longrightarrow J \subseteq I \Longrightarrow$ *space* $(P\ J) =$ *space* $(PiM\ J\ M)$
⟨*proof*⟩

lemma *not-empty-M*: $i \in I \Longrightarrow$ *space* $(M\ i) \neq \{\}$
⟨*proof*⟩

lemma *not-empty*: *space* $(PiM\ I\ M) \neq \{\}$
⟨*proof*⟩

abbreviation
emb L K ≡ *prod-emb L M K*

lemma *emb-preserve-mono*:
assumes $J \subseteq L$ $L \subseteq I$ **and** *sets*: $X \in$ *sets* $(Pi_M\ J\ M)$ $Y \in$ *sets* $(Pi_M\ J\ M)$
assumes *emb L J X* ⊆ *emb L J Y*
shows $X \subseteq Y$
⟨*proof*⟩

lemma *emb-injective*:
assumes *L*: $J \subseteq L$ $L \subseteq I$ **and** *X*: $X \in$ *sets* $(Pi_M\ J\ M)$ **and** *Y*: $Y \in$ *sets* $(Pi_M\ J\ M)$
shows *emb L J X* = *emb L J Y* ⟹ $X = Y$
⟨*proof*⟩

lemma *emeasure-P*: $J \subseteq K \Longrightarrow$ *finite* $K \Longrightarrow K \subseteq I \Longrightarrow X \in$ *sets* $(PiM\ J\ M)$ $\Longrightarrow P\ K$ (*emb K J X*) $= P\ J\ X$
⟨*proof*⟩

inductive-set *generator* :: $('i \Rightarrow 'a)$ *set set* **where**
finite J $\Longrightarrow J \subseteq I \Longrightarrow X \in$ *sets* $(Pi_M\ J\ M) \Longrightarrow$ *emb I J X* $\in$ *generator*

lemma *algebra-generator*: *algebra* (*space* (*PiM I M*)) *generator*
⟨*proof*⟩

interpretation *generator*: *algebra space* (*PiM I M*) *generator*
⟨*proof*⟩

lemma *sets-PiM-generator*: *sets* (*PiM I M*) = *sigma-sets* (*space* (*PiM I M*)) *generator*
⟨*proof*⟩

definition *mu-G* (μG) **where**
$\mu G\ A = (THE\ x.\ \forall J \subseteq I.\ finite\ J \longrightarrow (\forall X \in sets\ (Pi_M\ J\ M).\ A = emb\ I\ J\ X \longrightarrow x = emeasure\ (P\ J)\ X))$

definition *lim* :: $('i \Rightarrow 'a)$ *measure* **where**
$lim = extend\text{-}measure\ (space\ (PiM\ I\ M))\ generator\ (\lambda x.\ x)\ \mu G$

lemma *space-lim*[*simp*]: *space lim* = *space* (*PiM I M*)
⟨*proof*⟩

lemma *sets-lim*[*simp*, *measurable*]: *sets lim* = *sets* (*PiM I M*)
⟨*proof*⟩

lemma *mu-G-spec*:
assumes *J*: *finite J* $J \subseteq I$ $X \in sets\ (Pi_M\ J\ M)$
shows $\mu G\ (emb\ I\ J\ X) = emeasure\ (P\ J)\ X$
⟨*proof*⟩

lemma *positive-mu-G*: *positive generator* μG
⟨*proof*⟩

lemma *additive-mu-G*: *additive generator* μG
⟨*proof*⟩

lemma *emeasure-lim*:
assumes *JX*: *finite J* $J \subseteq I$ $X \in sets\ (PiM\ J\ M)$
assumes *cont*: $\bigwedge J\ X.\ (\bigwedge i.\ J\ i \subseteq I) \Longrightarrow incseq\ J \Longrightarrow (\bigwedge i.\ finite\ (J\ i)) \Longrightarrow (\bigwedge i.\ X\ i \in sets\ (PiM\ (J\ i)\ M)) \Longrightarrow$
$decseq\ (\lambda i.\ emb\ I\ (J\ i)\ (X\ i)) \Longrightarrow 0 < (INF\ i.\ P\ (J\ i)\ (X\ i)) \Longrightarrow (\bigcap i.\ emb\ I\ (J\ i)\ (X\ i)) \neq \{\}$
shows *emeasure lim* (*emb I J X*) = *P J X*
⟨*proof*⟩

end

sublocale *product-prob-space* $\subseteq$ *projective-family I* $\lambda J.$ *PiM J M M*

⟨*proof*⟩

Proof due to Ionescu Tulcea.

locale *Ionescu-Tulcea* =
fixes P :: *nat* ⇒ (*nat* ⇒ ′*a*) ⇒ ′*a measure* **and** M :: *nat* ⇒ ′*a measure*
assumes *P*[*measurable*]: ⋀*i*. *P i* ∈ *measurable* (*PiM* {*0*..<*i*} *M*) (*subprob-algebra* (*M i*))
assumes *prob-space-P*: ⋀*i x*. *x* ∈ *space* (*PiM* {*0*..<*i*} *M*) ⟹ *prob-space* (*P i x*)
begin

lemma *non-empty*[*simp*]: *space* (*M i*) ≠ {}
⟨*proof*⟩

lemma *space-PiM-not-empty*[*simp*]: *space* (*PiM UNIV M*) ≠ {}
⟨*proof*⟩

lemma *space-P*: *x* ∈ *space* (*PiM* {*0*..<*n*} *M*) ⟹ *space* (*P n x*) = *space* (*M n*)
⟨*proof*⟩

lemma *sets-P*[*measurable-cong*]: *x* ∈ *space* (*PiM* {*0*..<*n*} *M*) ⟹ *sets* (*P n x*) = *sets* (*M n*)
⟨*proof*⟩

definition *eP* :: *nat* ⇒ (*nat* ⇒ ′*a*) ⇒ (*nat* ⇒ ′*a*) *measure* **where**
eP n ω = *distr* (*P n ω*) (*PiM* {*0*..<*Suc n*} *M*) (*fun-upd ω n*)

lemma *measurable-eP*[*measurable*]:
eP n ∈ *measurable* (*PiM* {*0*..< *n*} *M*) (*subprob-algebra* (*PiM* {*0*..<*Suc n*} *M*))
⟨*proof*⟩

lemma *space-eP*:
x ∈ *space* (*PiM* {*0*..<*n*} *M*) ⟹ *space* (*eP n x*) = *space* (*PiM* {*0*..<*Suc n*} *M*)
⟨*proof*⟩

lemma *sets-eP*[*measurable*]:
x ∈ *space* (*PiM* {*0*..<*n*} *M*) ⟹ *sets* (*eP n x*) = *sets* (*PiM* {*0*..<*Suc n*} *M*)
⟨*proof*⟩

lemma *prob-space-eP*: *x* ∈ *space* (*PiM* {*0*..<*n*} *M*) ⟹ *prob-space* (*eP n x*)
⟨*proof*⟩

lemma *nn-integral-eP*:
ω ∈ *space* (*PiM* {*0*..<*n*} *M*) ⟹ *f* ∈ *borel-measurable* (*PiM* {*0*..<*Suc n*} *M*) ⟹
(∫⁺*x*. *f x* ∂*eP n ω*) = (∫⁺*x*. *f* (*ω*(*n* := *x*)) ∂*P n ω*)
⟨*proof*⟩

lemma *emeasure-eP*:

assumes $\omega[simp]$: $\omega \in space\ (PiM\ \{0..{<}n\}\ M)$ **and** $A[measurable]$: $A \in sets\ (PiM\ \{0..{<}Suc\ n\}\ M)$
shows $eP\ n\ \omega\ A = P\ n\ \omega\ ((\lambda x.\ \omega(n := x))\ -`\ A \cap space\ (M\ n))$
$\langle proof \rangle$

primrec $C :: nat \Rightarrow nat \Rightarrow (nat \Rightarrow 'a) \Rightarrow (nat \Rightarrow 'a)\ measure$ **where**
$C\ n\ 0\ \omega = return\ (PiM\ \{0..{<}n\}\ M)\ \omega$
$|\ C\ n\ (Suc\ m)\ \omega = C\ n\ m\ \omega \gg= eP\ (n + m)$

lemma *measurable-C*[*measurable*]:
$C\ n\ m \in measurable\ (PiM\ \{0..{<}n\}\ M)\ (subprob\text{-}algebra\ (PiM\ \{0..{<}n + m\}\ M))$
$\langle proof \rangle$

lemma *space-C*:
$x \in space\ (PiM\ \{0..{<}n\}\ M) \Longrightarrow space\ (C\ n\ m\ x) = space\ (PiM\ \{0..{<}n + m\}\ M)$
$\langle proof \rangle$

lemma *sets-C*[*measurable-cong*]:
$x \in space\ (PiM\ \{0..{<}n\}\ M) \Longrightarrow sets\ (C\ n\ m\ x) = sets\ (PiM\ \{0..{<}n + m\}\ M)$
$\langle proof \rangle$

lemma *prob-space-C*: $x \in space\ (PiM\ \{0..{<}n\}\ M) \Longrightarrow prob\text{-}space\ (C\ n\ m\ x)$
$\langle proof \rangle$

lemma *split-C*:
assumes ω: $\omega \in space\ (PiM\ \{0..{<}n\}\ M)$ **shows** $(C\ n\ m\ \omega \gg= C\ (n + m)\ l) = C\ n\ (m + l)\ \omega$
$\langle proof \rangle$

lemma *nn-integral-C*:
assumes $m \le m'$ **and** $f[measurable]$: $f \in borel\text{-}measurable\ (PiM\ \{0..{<}n{+}m\}\ M)$
and *nonneg*: $\bigwedge x.\ x \in space\ (PiM\ \{0..{<}n{+}m\}\ M) \Longrightarrow 0 \le f\ x$
and x: $x \in space\ (PiM\ \{0..{<}n\}\ M)$
shows $(\int^{+} x.\ f\ x\ \partial C\ n\ m\ x) = (\int^{+} x.\ f\ (restrict\ x\ \{0..{<}n{+}m\})\ \partial C\ n\ m'\ x)$
$\langle proof \rangle$

lemma *emeasure-C*:
assumes $m \le m'$ **and** $A[measurable]$: $A \in sets\ (PiM\ \{0..{<}n{+}m\}\ M)$ **and** $[simp]$: $x \in space\ (PiM\ \{0..{<}n\}\ M)$
shows $emeasure\ (C\ n\ m'\ x)\ (prod\text{-}emb\ \{0..{<}n + m'\}\ M\ \{0..{<}n{+}m\}\ A) = emeasure\ (C\ n\ m\ x)\ A$
$\langle proof \rangle$

lemma *distr-C*:
assumes $m \le m'$ **and** $[simp]$: $x \in space\ (PiM\ \{0..{<}n\}\ M)$

shows $C\ n\ m\ x = distr\ (C\ n\ m'\ x)\ (PiM\ \{0..<n+m\}\ M)\ (\lambda x.\ restrict\ x\ \{0..<n+m\})$
⟨*proof*⟩

definition *up-to* :: *nat set* ⇒ *nat* **where**
$up\text{-}to\ J = (LEAST\ n.\ \forall\, i \geq n.\ i \notin J)$

lemma *up-to-less*: $finite\ J \Longrightarrow i \in J \Longrightarrow i < up\text{-}to\ J$
⟨*proof*⟩

lemma *up-to-iff*: $finite\ J \Longrightarrow up\text{-}to\ J \leq n \longleftrightarrow (\forall\, i{\in}J.\ i < n)$
⟨*proof*⟩

lemma *up-to-iff-Ico*: $finite\ J \Longrightarrow up\text{-}to\ J \leq n \longleftrightarrow J \subseteq \{0..<n\}$
⟨*proof*⟩

lemma *up-to*: $finite\ J \Longrightarrow J \subseteq \{0..< up\text{-}to\ J\}$
⟨*proof*⟩

lemma *up-to-mono*: $J \subseteq H \Longrightarrow finite\ H \Longrightarrow up\text{-}to\ J \leq up\text{-}to\ H$
⟨*proof*⟩

definition *CI* :: *nat set* ⇒ (*nat* ⇒ *'a*) *measure* **where**
$CI\ J = distr\ (C\ 0\ (up\text{-}to\ J)\ (\lambda x.\ undefined))\ (PiM\ J\ M)\ (\lambda f.\ restrict\ f\ J)$

sublocale *PF*: *projective-family UNIV CI*
⟨*proof*⟩

lemma *emeasure-CI'*:
$finite\ J \Longrightarrow X \in sets\ (PiM\ J\ M) \Longrightarrow CI\ J\ X = C\ 0\ (up\text{-}to\ J)\ (\lambda\text{-}.\ undefined)\ (PF.emb\ \{0..<up\text{-}to\ J\}\ J\ X)$
⟨*proof*⟩

lemma *emeasure-CI*:
$J \subseteq \{0..<n\} \Longrightarrow X \in sets\ (PiM\ J\ M) \Longrightarrow CI\ J\ X = C\ 0\ n\ (\lambda\text{-}.\ undefined)\ (PF.emb\ \{0..<n\}\ J\ X)$
⟨*proof*⟩

lemma *lim*:
assumes *J*: *finite J* **and** *X*: $X \in sets\ (PiM\ J\ M)$
shows $emeasure\ PF.lim\ (PF.emb\ UNIV\ J\ X) = emeasure\ (CI\ J)\ X$
⟨*proof*⟩

lemma *distr-lim*: **assumes** *J*[*simp*]: *finite J* **shows** $distr\ PF.lim\ (PiM\ J\ M)\ (\lambda x.\ restrict\ x\ J) = CI\ J$
⟨*proof*⟩

end

lemma (**in** *product-prob-space*) *emeasure-lim-emb*:
assumes $*$: *finite J J* $\subseteq$ *I X* $\in$ *sets* (*PiM J M*)
shows *emeasure lim* (*emb I J X*) = *emeasure* (Pi_M *J M*) *X*
⟨*proof*⟩

end

23 Infinite Product Measure

theory *Infinite-Product-Measure*
imports *Probability-Measure Caratheodory Projective-Family*
begin

lemma (**in** *product-prob-space*) *distr-PiM-restrict-finite*:
assumes *finite J J* $\subseteq$ *I*
shows *distr* (*PiM I M*) (*PiM J M*) (λx. *restrict x J*) = *PiM J M*
⟨*proof*⟩

lemma (**in** *product-prob-space*) *emeasure-PiM-emb'*:
J $\subseteq$ *I* $\Longrightarrow$ *finite J* $\Longrightarrow$ *X* $\in$ *sets* (*PiM J M*) $\Longrightarrow$ *emeasure* (Pi_M *I M*) (*emb I J X*) = *PiM J M X*
⟨*proof*⟩

lemma (**in** *product-prob-space*) *emeasure-PiM-emb*:
J $\subseteq$ *I* $\Longrightarrow$ *finite J* $\Longrightarrow$ ($\bigwedge$*i*. *i* $\in$ *J* $\Longrightarrow$ *X i* $\in$ *sets* (*M i*)) $\Longrightarrow$
emeasure (Pi_M *I M*) (*emb I J* (Pi_E *J X*)) = ($\prod$ *i*$\in$*J*. *emeasure* (*M i*) (*X i*))
⟨*proof*⟩

sublocale *product-prob-space* $\subseteq$ *P?*: *prob-space* Pi_M *I M*
⟨*proof*⟩

lemma (**in** *product-prob-space*) *emeasure-PiM-Collect*:
assumes *X*: *J* $\subseteq$ *I finite J* $\bigwedge$*i*. *i* $\in$ *J* $\Longrightarrow$ *X i* $\in$ *sets* (*M i*)
shows *emeasure* (Pi_M *I M*) {*x*$\in$*space* (Pi_M *I M*). $\forall$ *i*$\in$*J*. *x i* $\in$ *X i*} = ($\prod$ *i*$\in$*J*. *emeasure* (*M i*) (*X i*))
⟨*proof*⟩

lemma (**in** *product-prob-space*) *emeasure-PiM-Collect-single*:
assumes *X*: *i* $\in$ *I A* $\in$ *sets* (*M i*)
shows *emeasure* (Pi_M *I M*) {*x*$\in$*space* (Pi_M *I M*). *x i* $\in$ *A*} = *emeasure* (*M i*) *A*
⟨*proof*⟩

lemma (**in** *product-prob-space*) *measure-PiM-emb*:
assumes *J* $\subseteq$ *I finite J* $\bigwedge$*i*. *i* $\in$ *J* $\Longrightarrow$ *X i* $\in$ *sets* (*M i*)
shows *measure* (*PiM I M*) (*emb I J* (Pi_E *J X*)) = ($\prod$ *i*$\in$*J*. *measure* (*M i*) (*X i*))
⟨*proof*⟩

lemma *sets-Collect-single'*:
$i \in I \Longrightarrow \{x{\in}space\ (M\ i).\ P\ x\} \in sets\ (M\ i) \Longrightarrow \{x{\in}space\ (PiM\ I\ M).\ P\ (x\ i)\} \in sets\ (PiM\ I\ M)$
⟨*proof*⟩

lemma (**in** *finite-product-prob-space*) *finite-measure-PiM-emb*:
$(\bigwedge i.\ i \in I \Longrightarrow A\ i \in sets\ (M\ i)) \Longrightarrow measure\ (PiM\ I\ M)\ (Pi_E\ I\ A) = (\prod i{\in}I.\ measure\ (M\ i)\ (A\ i))$
⟨*proof*⟩

lemma (**in** *product-prob-space*) *PiM-component*:
assumes $i \in I$
shows $distr\ (PiM\ I\ M)\ (M\ i)\ (\lambda\omega.\ \omega\ i) = M\ i$
⟨*proof*⟩

lemma (**in** *product-prob-space*) *PiM-eq*:
assumes M': $sets\ M' = sets\ (PiM\ I\ M)$
assumes *eq*: $\bigwedge J\ F.\ finite\ J \Longrightarrow J \subseteq I \Longrightarrow (\bigwedge j.\ j \in J \Longrightarrow F\ j \in sets\ (M\ j)) \Longrightarrow$
$emeasure\ M'\ (prod\text{-}emb\ I\ M\ J\ (\Pi_E\ j{\in}J.\ F\ j)) = (\prod j{\in}J.\ emeasure\ (M\ j)\ (F\ j))$
shows $M' = (PiM\ I\ M)$
⟨*proof*⟩

lemma (**in** *product-prob-space*) *AE-component*: $i \in I \Longrightarrow AE\ x\ in\ M\ i.\ P\ x \Longrightarrow AE\ x\ in\ PiM\ I\ M.\ P\ (x\ i)$
⟨*proof*⟩

23.1 Sequence space

definition *comb-seq* :: $nat \Rightarrow (nat \Rightarrow 'a) \Rightarrow (nat \Rightarrow 'a) \Rightarrow (nat \Rightarrow 'a)$ **where**
$comb\text{-}seq\ i\ \omega\ \omega'\ j = (if\ j < i\ then\ \omega\ j\ else\ \omega'\ (j - i))$

lemma *split-comb-seq*: $P\ (comb\text{-}seq\ i\ \omega\ \omega'\ j) \longleftrightarrow (j < i \longrightarrow P\ (\omega\ j)) \wedge (\forall k.\ j = i + k \longrightarrow P\ (\omega'\ k))$
⟨*proof*⟩

lemma *split-comb-seq-asm*: $P\ (comb\text{-}seq\ i\ \omega\ \omega'\ j) \longleftrightarrow \neg\ ((j < i \wedge \neg\ P\ (\omega\ j)) \vee (\exists k.\ j = i + k \wedge \neg\ P\ (\omega'\ k)))$
⟨*proof*⟩

lemma *measurable-comb-seq*:
$(\lambda(\omega, \omega').\ comb\text{-}seq\ i\ \omega\ \omega') \in measurable\ ((\Pi_M\ i{\in}UNIV.\ M) \bigotimes_M (\Pi_M\ i{\in}UNIV.\ M))\ (\Pi_M\ i{\in}UNIV.\ M)$
⟨*proof*⟩

lemma *measurable-comb-seq'*[*measurable (raw)*]:
assumes *f*: $f \in measurable\ N\ (\Pi_M\ i{\in}UNIV.\ M)$ **and** *g*: $g \in measurable\ N\ (\Pi_M\ i{\in}UNIV.\ M)$

shows $(\lambda x.\ comb\text{-}seq\ i\ (f\ x)\ (g\ x)) \in measurable\ N\ (\Pi_M\ i{\in}UNIV.\ M)$
⟨*proof*⟩

lemma *comb-seq-0*: $comb\text{-}seq\ 0\ \omega\ \omega' = \omega'$
⟨*proof*⟩

lemma *comb-seq-Suc*: $comb\text{-}seq\ (Suc\ n)\ \omega\ \omega' = comb\text{-}seq\ n\ \omega\ (case\text{-}nat\ (\omega\ n)\ \omega')$
⟨*proof*⟩

lemma *comb-seq-Suc-0*[*simp*]: $comb\text{-}seq\ (Suc\ 0)\ \omega = case\text{-}nat\ (\omega\ 0)$
⟨*proof*⟩

lemma *comb-seq-less*: $i < n \Longrightarrow comb\text{-}seq\ n\ \omega\ \omega'\ i = \omega\ i$
⟨*proof*⟩

lemma *comb-seq-add*: $comb\text{-}seq\ n\ \omega\ \omega'\ (i + n) = \omega'\ i$
⟨*proof*⟩

lemma *case-nat-comb-seq*: $case\text{-}nat\ s'\ (comb\text{-}seq\ n\ \omega\ \omega')\ (i + n) = case\text{-}nat\ (case\text{-}nat\ s'\ \omega\ n)\ \omega'\ i$
⟨*proof*⟩

lemma *case-nat-comb-seq'*:
$case\text{-}nat\ s\ (comb\text{-}seq\ i\ \omega\ \omega') = comb\text{-}seq\ (Suc\ i)\ (case\text{-}nat\ s\ \omega)\ \omega'$
⟨*proof*⟩

locale *sequence-space* = *product-prob-space* $\lambda i.\ M\ UNIV :: nat\ set$ **for** M
begin

abbreviation $S \equiv \Pi_M\ i{\in}UNIV{::}nat\ set.\ M$

lemma *infprod-in-sets*[*intro*]:
fixes $E :: nat \Rightarrow 'a\ set$ **assumes** E: $\bigwedge i.\ E\ i \in sets\ M$
shows $Pi\ UNIV\ E \in sets\ S$
⟨*proof*⟩

lemma *measure-PiM-countable*:
fixes $E :: nat \Rightarrow 'a\ set$ **assumes** E: $\bigwedge i.\ E\ i \in sets\ M$
shows $(\lambda n.\ \prod i{\le}n.\ measure\ M\ (E\ i)) \longrightarrow measure\ S\ (Pi\ UNIV\ E)$
⟨*proof*⟩

lemma *nat-eq-diff-eq*:
fixes $a\ b\ c :: nat$
shows $c \le b \Longrightarrow a = b - c \longleftrightarrow a + c = b$
⟨*proof*⟩

lemma *PiM-comb-seq*:
$distr\ (S \bigotimes_M S)\ S\ (\lambda(\omega, \omega').\ comb\text{-}seq\ i\ \omega\ \omega') = S$ (**is** *?D* = -)
⟨*proof*⟩

lemma *PiM-iter*:
  distr ($M \bigotimes_M S$) *S* ($\lambda(s, \omega)$. *case-nat s* ω) = *S* (**is** *?D* = -)
⟨*proof*⟩

end

end

24 Projective Limit

theory *Projective-Limit*
  imports
    Caratheodory
    Fin-Map
    Regularity
    Projective-Family
    Infinite-Product-Measure
    ~~/src/HOL/Library/Diagonal-Subsequence
begin

24.1 Sequences of Finite Maps in Compact Sets

locale *finmap-seqs-into-compact* =
  fixes K::*nat* ⇒ (*nat* $\Rightarrow_F$ *'a*::*metric-space*) *set* **and** *f*::*nat* ⇒ (*nat* $\Rightarrow_F$ *'a*) **and** *M*
  assumes *compact*: $\bigwedge n$. *compact* (K n)
  assumes *f-in-K*: $\bigwedge n$. $K\ n \neq \{\}$
  assumes *domain-K*: $\bigwedge n$. $k \in K\ n \Longrightarrow$ *domain* k = *domain* ($f\ n$)
  assumes *proj-in-K*:
    $\bigwedge t\ n\ m.\ m \geq n \Longrightarrow t \in$ *domain* ($f\ n$) $\Longrightarrow (f\ m)_F\ t \in (\lambda k.\ (k)_F\ t)$ ‘ $K\ n$
begin

lemma *proj-in-K'*: ($\exists n.\ \forall m \geq n.\ (f\ m)_F\ t \in (\lambda k.\ (k)_F\ t)$ ‘ $K\ n$)
  ⟨*proof*⟩

lemma *proj-in-KE*:
  obtains n **where** $\bigwedge m.\ m \geq n \Longrightarrow (f\ m)_F\ t \in (\lambda k.\ (k)_F\ t)$ ‘ $K\ n$
  ⟨*proof*⟩

lemma *compact-projset*:
  shows *compact* (($\lambda k.\ (k)_F\ i$) ‘ $K\ n$)
  ⟨*proof*⟩

end

lemma *compactE'*:
  fixes S :: *'a* :: *metric-space set*
  assumes *compact* S $\forall n \geq m.\ f\ n \in S$

obtains *l r* **where** $l \in S$ *subseq r* $((f \circ r) \longrightarrow l)$ *sequentially*
⟨*proof*⟩

sublocale *finmap-seqs-into-compact* ⊆ *subseqs* $\lambda n\ s.\ (\exists\, l.\ (\lambda i.\ ((f\ o\ s)\ i)_F\ n) \longrightarrow l)$
⟨*proof*⟩

lemma (**in** *finmap-seqs-into-compact*) *diagonal-tendsto*: $\exists\, l.\ (\lambda i.\ (f\ (diagseq\ i))_F\ n) \longrightarrow l$
⟨*proof*⟩

24.2 Daniell-Kolmogorov Theorem

Existence of Projective Limit

locale *polish-projective* = *projective-family I P* λ-. *borel*::′*a*::*polish-space measure*
for *I*::′*i set* **and** *P*
begin

lemma *emeasure-lim-emb*:
assumes *X*: $J \subseteq I$ *finite J* $X \in$ *sets* (Π_M *i*∈*J*. *borel*)
shows *lim* (*emb I J X*) = *P J X*
⟨*proof*⟩

lemma *measure-lim-emb*:
$J \subseteq I \Longrightarrow$ *finite J* $\Longrightarrow X \in$ *sets* (Π_M *i*∈*J*. *borel*) $\Longrightarrow$ *measure lim* (*emb I J X*) = *measure* (*P J*) *X*
⟨*proof*⟩

end

hide-const (**open**) *PiF*
hide-const (**open**) Pi_F
hide-const (**open**) *Pi*′
hide-const (**open**) *Abs-finmap*
hide-const (**open**) *Rep-finmap*
hide-const (**open**) *finmap-of*
hide-const (**open**) *proj*
hide-const (**open**) *domain*
hide-const (**open**) *basis-finmap*

sublocale *polish-projective* ⊆ *P*: *prob-space lim*
⟨*proof*⟩

locale *polish-product-prob-space* =
product-prob-space λ-. *borel*::(′*a*::*polish-space*) *measure I* **for** *I*::′*i set*

sublocale *polish-product-prob-space* ⊆ *P*: *polish-projective I* λ*J*. *PiM J* (λ-. *borel*::(′*a*) *measure*)
⟨*proof*⟩

lemma (**in** *polish-product-prob-space*) *limP-eq-PiM*: *lim* = *PiM I* (λ-. *borel*)
⟨*proof*⟩

end

25 Probability mass function

theory *Probability-Mass-Function*
imports
Giry-Monad
~~/src/HOL/Library/Multiset
begin

lemma *AE-emeasure-singleton*:
assumes *x*: *emeasure M* {*x*} ≠ *0* **and** *ae*: *AE x in M*. *P x* **shows** *P x*
⟨*proof*⟩

lemma *AE-measure-singleton*: *measure M* {*x*} ≠ *0* ⟹ *AE x in M*. *P x* ⟹ *P x*
⟨*proof*⟩

lemma (**in** *finite-measure*) *AE-support-countable*:
assumes [*simp*]: *sets M* = *UNIV*
shows (*AE x in M*. *measure M* {*x*} ≠ *0*) ⟷ (∃ *S*. *countable S* ∧ (*AE x in M*. *x* ∈ *S*))
⟨*proof*⟩

25.1 PMF as measure

typedef *'a pmf* = {*M* :: *'a measure*. *prob-space M* ∧ *sets M* = *UNIV* ∧ (*AE x in M*. *measure M* {*x*} ≠ *0*)}
morphisms *measure-pmf Abs-pmf*
⟨*proof*⟩

declare [[*coercion measure-pmf*]]

lemma *prob-space-measure-pmf*: *prob-space* (*measure-pmf p*)
⟨*proof*⟩

interpretation *measure-pmf*: *prob-space measure-pmf M* **for** *M*
⟨*proof*⟩

interpretation *measure-pmf*: *subprob-space measure-pmf M* **for** *M*
⟨*proof*⟩

lemma *subprob-space-measure-pmf*: *subprob-space* (*measure-pmf x*)
⟨*proof*⟩

locale *pmf-as-measure*

begin

setup-lifting *type-definition-pmf*

end

context
begin

interpretation *pmf-as-measure* ⟨*proof*⟩

lemma *sets-measure-pmf*[*simp*]: *sets* (*measure-pmf p*) = *UNIV*
⟨*proof*⟩

lemma *sets-measure-pmf-count-space*[*measurable-cong*]:
sets (*measure-pmf M*) = *sets* (*count-space UNIV*)
⟨*proof*⟩

lemma *space-measure-pmf*[*simp*]: *space* (*measure-pmf p*) = *UNIV*
⟨*proof*⟩

lemma *measure-pmf-UNIV* [*simp*]: *measure* (*measure-pmf p*) *UNIV* = *1*
⟨*proof*⟩

lemma *measure-pmf-in-subprob-algebra*[*measurable* (*raw*)]: *measure-pmf x* ∈ *space* (*subprob-algebra* (*count-space UNIV*))
⟨*proof*⟩

lemma *measurable-pmf-measure1*[*simp*]: *measurable* (*M* :: *'a pmf*) *N* = *UNIV* → *space N*
⟨*proof*⟩

lemma *measurable-pmf-measure2*[*simp*]: *measurable N* (*M* :: *'a pmf*) = *measurable N* (*count-space UNIV*)
⟨*proof*⟩

lemma *measurable-pair-restrict-pmf2*:
assumes *countable A*
assumes [*measurable*]: $\bigwedge y.\ y \in A \Longrightarrow (\lambda x.\ f\ (x,\ y)) \in$ *measurable M L*
shows *f* ∈ *measurable* (*M* $\bigotimes_M$ *restrict-space* (*measure-pmf N*) *A*) *L* (**is** *f* ∈ *measurable ?M* -)
⟨*proof*⟩

lemma *measurable-pair-restrict-pmf1*:
assumes *countable A*
assumes [*measurable*]: $\bigwedge x.\ x \in A \Longrightarrow (\lambda y.\ f\ (x,\ y)) \in$ *measurable N L*
shows *f* ∈ *measurable* (*restrict-space* (*measure-pmf M*) *A* $\bigotimes_M$ *N*) *L*
⟨*proof*⟩

lift-definition *pmf* :: *'a pmf* ⇒ *'a* ⇒ *real* **is** $\lambda M\ x.$ *measure M* $\{x\}$ ⟨*proof*⟩

lift-definition *set-pmf* :: *'a pmf* ⇒ *'a set* **is** $\lambda M.\ \{x.$ *measure M* $\{x\} \neq 0\}$ ⟨*proof*⟩
declare [[*coercion set-pmf*]]

lemma *AE-measure-pmf*: *AE x in* (*M*::*'a pmf*). $x \in M$
⟨*proof*⟩

lemma *emeasure-pmf-single-eq-zero-iff*:
fixes *M* :: *'a pmf*
shows *emeasure M* $\{y\} = 0 \longleftrightarrow y \notin M$
⟨*proof*⟩

lemma *AE-measure-pmf-iff*: (*AE x in measure-pmf M. P x*) ⟷ (∀ *y*∈*M*. *P y*)
⟨*proof*⟩

lemma *AE-pmfI*: ($\bigwedge y.\ y \in$ *set-pmf M* ⟹ *P y*) ⟹ *almost-everywhere* (*measure-pmf M*) *P*
⟨*proof*⟩

lemma *countable-set-pmf* [*simp*]: *countable* (*set-pmf p*)
⟨*proof*⟩

lemma *pmf-positive*: $x \in$ *set-pmf p* ⟹ $0 <$ *pmf p x*
⟨*proof*⟩

lemma *pmf-nonneg*[*simp*]: $0 \leq$ *pmf p x*
⟨*proof*⟩

lemma *pmf-le-1*: *pmf p x* ≤ 1
⟨*proof*⟩

lemma *set-pmf-not-empty*: *set-pmf M* $\neq \{\}$
⟨*proof*⟩

lemma *set-pmf-iff*: $x \in$ *set-pmf M* ⟷ *pmf M x* $\neq 0$
⟨*proof*⟩

lemma *pmf-positive-iff*: $0 <$ *pmf p x* ⟷ $x \in$ *set-pmf p*
⟨*proof*⟩

lemma *set-pmf-eq*: *set-pmf M* = $\{x.$ *pmf M x* $\neq 0\}$
⟨*proof*⟩

lemma *emeasure-pmf-single*:
fixes *M* :: *'a pmf*
shows *emeasure M* $\{x\}$ = *pmf M x*
⟨*proof*⟩

lemma *measure-pmf-single*: *measure* (*measure-pmf M*) {*x*} = *pmf M x*
⟨*proof*⟩

lemma *emeasure-measure-pmf-finite*: *finite S* ⟹ *emeasure* (*measure-pmf M*) *S* = ($\sum$ *s*∈*S*. *pmf M s*)
⟨*proof*⟩

lemma *measure-measure-pmf-finite*: *finite S* ⟹ *measure* (*measure-pmf M*) *S* = *setsum* (*pmf M*) *S*
⟨*proof*⟩

lemma *nn-integral-measure-pmf-support*:
fixes *f* :: ′*a* ⇒ *ennreal*
assumes *f*: *finite A* **and** *nn*: ⋀*x*. *x* ∈ *A* ⟹ *0* ≤ *f x* ⋀*x*. *x* ∈ *set-pmf M* ⟹ *x* ∉ *A* ⟹ *f x* = *0*
shows ($\int^+$*x*. *f x* ∂*measure-pmf M*) = ($\sum$ *x*∈*A*. *f x* ∗ *pmf M x*)
⟨*proof*⟩

lemma *nn-integral-measure-pmf-finite*:
fixes *f* :: ′*a* ⇒ *ennreal*
assumes *f*: *finite* (*set-pmf M*) **and** *nn*: ⋀*x*. *x* ∈ *set-pmf M* ⟹ *0* ≤ *f x*
shows ($\int^+$*x*. *f x* ∂*measure-pmf M*) = ($\sum$ *x*∈*set-pmf M*. *f x* ∗ *pmf M x*)
⟨*proof*⟩

lemma *integrable-measure-pmf-finite*:
fixes *f* :: ′*a* ⇒ ′*b*::{*banach*, *second-countable-topology*}
shows *finite* (*set-pmf M*) ⟹ *integrable M f*
⟨*proof*⟩

lemma *integral-measure-pmf*:
assumes [*simp*]: *finite A* **and** ⋀*a*. *a* ∈ *set-pmf M* ⟹ *f a* ≠ *0* ⟹ *a* ∈ *A*
shows ($\int$ *x*. *f x* ∂*measure-pmf M*) = ($\sum$ *a*∈*A*. *f a* ∗ *pmf M a*)
⟨*proof*⟩

lemma *integrable-pmf*: *integrable* (*count-space X*) (*pmf M*)
⟨*proof*⟩

lemma *integral-pmf*: ($\int$ *x*. *pmf M x* ∂*count-space X*) = *measure M X*
⟨*proof*⟩

lemma *integral-pmf-restrict*:
(*f*::′*a* ⇒ ′*b*::{*banach*, *second-countable-topology*}) ∈ *borel-measurable* (*count-space UNIV*) ⟹
($\int$ *x*. *f x* ∂*measure-pmf M*) = ($\int$ *x*. *f x* ∂*restrict-space M M*)
⟨*proof*⟩

lemma *emeasure-pmf*: *emeasure* (*M*::′*a pmf*) *M* = *1*
⟨*proof*⟩

lemma *emeasure-pmf-UNIV* [*simp*]: *emeasure* (*measure-pmf M*) *UNIV* = *1*
⟨*proof*⟩

lemma *in-null-sets-measure-pmfI*:
$A \cap$ *set-pmf p* = {} $\Longrightarrow A \in$ *null-sets* (*measure-pmf p*)
⟨*proof*⟩

lemma *measure-subprob*: *measure-pmf M* $\in$ *space* (*subprob-algebra* (*count-space UNIV*))
⟨*proof*⟩

25.2 Monad Interpretation

lemma *measurable-measure-pmf*[*measurable*]:
(λx. *measure-pmf* (*M x*)) $\in$ *measurable* (*count-space UNIV*) (*subprob-algebra* (*count-space UNIV*))
⟨*proof*⟩

lemma *bind-measure-pmf-cong*:
assumes $\bigwedge x$. *A x* $\in$ *space* (*subprob-algebra N*) $\bigwedge x$. *B x* $\in$ *space* (*subprob-algebra N*)
assumes $\bigwedge i$. *i* $\in$ *set-pmf x* $\Longrightarrow$ *A i* = *B i*
shows *bind* (*measure-pmf x*) *A* = *bind* (*measure-pmf x*) *B*
⟨*proof*⟩

lift-definition *bind-pmf* :: *'a pmf* $\Rightarrow$ (*'a* $\Rightarrow$ *'b pmf*) $\Rightarrow$ *'b pmf* **is** *bind*
⟨*proof*⟩

lemma *ennreal-pmf-bind*: *pmf* (*bind-pmf N f*) *i* = ($\int^+ x$. *pmf* (*f x*) *i* ∂*measure-pmf N*)
⟨*proof*⟩

lemma *pmf-bind*: *pmf* (*bind-pmf N f*) *i* = ($\int x$. *pmf* (*f x*) *i* ∂*measure-pmf N*)
⟨*proof*⟩

lemma *bind-pmf-const*[*simp*]: *bind-pmf M* (λx. *c*) = *c*
⟨*proof*⟩

lemma *set-bind-pmf*[*simp*]: *set-pmf* (*bind-pmf M N*) = ($\bigcup M \in$*set-pmf M*. *set-pmf* (*N M*))
⟨*proof*⟩

lemma *bind-pmf-cong*:
assumes *p* = *q*
shows ($\bigwedge x$. *x* $\in$ *set-pmf q* $\Longrightarrow$ *f x* = *g x*) $\Longrightarrow$ *bind-pmf p f* = *bind-pmf q g*
⟨*proof*⟩

lemma *bind-pmf-cong-simp*:
p = *q* $\Longrightarrow$ ($\bigwedge x$. *x* $\in$ *set-pmf q* =*simp*=> *f x* = *g x*) $\Longrightarrow$ *bind-pmf p f* = *bind-pmf*

q g
⟨*proof*⟩

lemma *measure-pmf-bind*: *measure-pmf* (*bind-pmf M f*) = (*measure-pmf M* $\gg=$ (λx. *measure-pmf* (*f x*)))
⟨*proof*⟩

lemma *nn-integral-bind-pmf*[*simp*]: ($\int^+ x$. *f x* ∂*bind-pmf M N*) = ($\int^+ x$. $\int^+ y$. *f y* ∂*N x* ∂*M*)
⟨*proof*⟩

lemma *emeasure-bind-pmf*[*simp*]: *emeasure* (*bind-pmf M N*) *X* = ($\int^+ x$. *emeasure* (*N x*) *X* ∂*M*)
⟨*proof*⟩

lift-definition *return-pmf* :: *'a* ⇒ *'a pmf* **is** *return* (*count-space UNIV*)
⟨*proof*⟩

lemma *bind-return-pmf*: *bind-pmf* (*return-pmf x*) *f* = *f x*
⟨*proof*⟩

lemma *set-return-pmf*[*simp*]: *set-pmf* (*return-pmf x*) = {*x*}
⟨*proof*⟩

lemma *bind-return-pmf'*: *bind-pmf N return-pmf* = *N*
⟨*proof*⟩

lemma *bind-assoc-pmf*: *bind-pmf* (*bind-pmf A B*) *C* = *bind-pmf A* (λx. *bind-pmf* (*B x*) *C*)
⟨*proof*⟩

definition *map-pmf f M* = *bind-pmf M* (λx. *return-pmf* (*f x*))

lemma *map-bind-pmf*: *map-pmf f* (*bind-pmf M g*) = *bind-pmf M* (λx. *map-pmf f* (*g x*))
⟨*proof*⟩

lemma *bind-map-pmf*: *bind-pmf* (*map-pmf f M*) *g* = *bind-pmf M* (λx. *g* (*f x*))
⟨*proof*⟩

lemma *map-pmf-transfer*[*transfer-rule*]:
rel-fun op = (*rel-fun cr-pmf cr-pmf*) (λf *M*. *distr M* (*count-space UNIV*) *f*) *map-pmf*
⟨*proof*⟩

lemma *map-pmf-rep-eq*:
measure-pmf (*map-pmf f M*) = *distr* (*measure-pmf M*) (*count-space UNIV*) *f*
⟨*proof*⟩

lemma *map-pmf-id*[*simp*]: *map-pmf id* = *id*
⟨*proof*⟩

lemma *map-pmf-ident*[*simp*]: *map-pmf* ($\lambda x.\ x$) = ($\lambda x.\ x$)
⟨*proof*⟩

lemma *map-pmf-compose*: *map-pmf* ($f \circ g$) = *map-pmf f* ∘ *map-pmf g*
⟨*proof*⟩

lemma *map-pmf-comp*: *map-pmf f* (*map-pmf g M*) = *map-pmf* ($\lambda x.\ f\ (g\ x)$) *M*
⟨*proof*⟩

lemma *map-pmf-cong*: $p = q \Longrightarrow (\bigwedge x.\ x \in$ *set-pmf* $q \Longrightarrow f\ x = g\ x) \Longrightarrow$ *map-pmf* *f p* = *map-pmf g q*
⟨*proof*⟩

lemma *pmf-set-map*: *set-pmf* ∘ *map-pmf f* = *op* ‘ *f* ∘ *set-pmf*
⟨*proof*⟩

lemma *set-map-pmf*[*simp*]: *set-pmf* (*map-pmf f M*) = *f*‘*set-pmf M*
⟨*proof*⟩

lemma *emeasure-map-pmf*[*simp*]: *emeasure* (*map-pmf f M*) *X* = *emeasure M* (*f* −‘ *X*)
⟨*proof*⟩

lemma *measure-map-pmf*[*simp*]: *measure* (*map-pmf f M*) *X* = *measure M* (*f* −‘ *X*)
⟨*proof*⟩

lemma *nn-integral-map-pmf*[*simp*]: ($\int^+ x.\ f\ x\ \partial$*map-pmf g M*) = ($\int^+ x.\ f\ (g\ x)\ \partial M$)
⟨*proof*⟩

lemma *ennreal-pmf-map*: *pmf* (*map-pmf f p*) *x* = ($\int^+ y.$ *indicator* (*f* −‘ $\{x\}$) *y* ∂*measure-pmf p*)
⟨*proof*⟩

lemma *pmf-map*: *pmf* (*map-pmf f p*) *x* = *measure p* (*f* −‘ $\{x\}$)
⟨*proof*⟩

lemma *nn-integral-pmf*: ($\int^+ x.$ *pmf p x* ∂*count-space A*) = *emeasure* (*measure-pmf p*) *A*
⟨*proof*⟩

lemma *integral-map-pmf*[*simp*]:
fixes *f* :: *'a* ⇒ *'b*::{*banach*, *second-countable-topology*}
shows *integral*^{L} (*map-pmf g p*) *f* = *integral*^{L} *p* ($\lambda x.\ f\ (g\ x)$)
⟨*proof*⟩

lemma *map-return-pmf* [*simp*]: *map-pmf f* (*return-pmf x*) = *return-pmf* (*f x*)
⟨*proof*⟩

lemma *map-pmf-const*[*simp*]: *map-pmf* (λ-. *c*) *M* = *return-pmf c*
⟨*proof*⟩

lemma *pmf-return* [*simp*]: *pmf* (*return-pmf x*) *y* = *indicator* {*y*} *x*
⟨*proof*⟩

lemma *nn-integral-return-pmf*[*simp*]: *0* ≤ *f x* ⟹ (∫ ⁺*x*. *f x* ∂*return-pmf x*) = *f x*
⟨*proof*⟩

lemma *emeasure-return-pmf*[*simp*]: *emeasure* (*return-pmf x*) *X* = *indicator X x*
⟨*proof*⟩

lemma *return-pmf-inj*[*simp*]: *return-pmf x* = *return-pmf y* ⟷ *x* = *y*
⟨*proof*⟩

lemma *map-pmf-eq-return-pmf-iff*:
map-pmf f p = *return-pmf x* ⟷ (∀ *y* ∈ *set-pmf p*. *f y* = *x*)
⟨*proof*⟩

definition *pair-pmf A B* = *bind-pmf A* (λ*x*. *bind-pmf B* (λ*y*. *return-pmf* (*x*, *y*)))

lemma *pmf-pair*: *pmf* (*pair-pmf M N*) (*a*, *b*) = *pmf M a* ∗ *pmf N b*
⟨*proof*⟩

lemma *set-pair-pmf*[*simp*]: *set-pmf* (*pair-pmf A B*) = *set-pmf A* × *set-pmf B*
⟨*proof*⟩

lemma *measure-pmf-in-subprob-space*[*measurable* (*raw*)]:
measure-pmf M ∈ *space* (*subprob-algebra* (*count-space UNIV*))
⟨*proof*⟩

lemma *nn-integral-pair-pmf′*: (∫ ⁺*x*. *f x* ∂*pair-pmf A B*) = (∫ ⁺*a*. ∫ ⁺*b*. *f* (*a*, *b*) ∂*B* ∂*A*)
⟨*proof*⟩

lemma *bind-pair-pmf*:
assumes *M*[*measurable*]: *M* ∈ *measurable* (*count-space UNIV* $\bigotimes_M$ *count-space UNIV*) (*subprob-algebra N*)
shows *measure-pmf* (*pair-pmf A B*) ⋙ *M* = (*measure-pmf A* ⋙ (λ*x*. *measure-pmf B* ⋙ (λ*y*. *M* (*x*, *y*))))
(**is** *?L* = *?R*)
⟨*proof*⟩

lemma *map-fst-pair-pmf*: *map-pmf fst* (*pair-pmf A B*) = *A*

⟨proof⟩

lemma *map-snd-pair-pmf*: *map-pmf snd* (*pair-pmf A B*) = *B*
⟨proof⟩

lemma *nn-integral-pmf'*:
inj-on f A $\Longrightarrow$ ($\int^{+}x$. *pmf p* (*f x*) ∂*count-space A*) = *emeasure p* (*f ' A*)
⟨proof⟩

lemma *pmf-le-0-iff* [*simp*]: *pmf M p* $\leq$ *0* $\longleftrightarrow$ *pmf M p* = *0*
⟨proof⟩

lemma *min-pmf-0*[*simp*]: *min* (*pmf M p*) *0* = *0 min 0* (*pmf M p*) = *0*
⟨proof⟩

lemma *pmf-eq-0-set-pmf*: *pmf M p* = *0* $\longleftrightarrow$ *p* $\notin$ *set-pmf M*
⟨proof⟩

lemma *pmf-map-inj*: *inj-on f* (*set-pmf M*) $\Longrightarrow$ *x* $\in$ *set-pmf M* $\Longrightarrow$ *pmf* (*map-pmf f M*) (*f x*) = *pmf M x*
⟨proof⟩

lemma *pmf-map-inj'*: *inj f* $\Longrightarrow$ *pmf* (*map-pmf f M*) (*f x*) = *pmf M x*
⟨proof⟩

lemma *pmf-map-outside*: *x* $\notin$ *f ' set-pmf M* $\Longrightarrow$ *pmf* (*map-pmf f M*) *x* = *0*
⟨proof⟩

25.3 PMFs as function

context
fixes *f* :: *'a* $\Rightarrow$ *real*
assumes *nonneg*: $\bigwedge x$. *0* $\leq$ *f x*
assumes *prob*: ($\int^{+}x$. *f x* ∂*count-space UNIV*) = *1*
begin

lift-definition *embed-pmf* :: *'a pmf* **is** *density* (*count-space UNIV*) (*ennreal* $\circ$ *f*)
⟨proof⟩

lemma *pmf-embed-pmf*: *pmf embed-pmf x* = *f x*
⟨proof⟩

lemma *set-embed-pmf*: *set-pmf embed-pmf* = {*x*. *f x* $\neq$ *0*}
⟨proof⟩

end

lemma *embed-pmf-transfer*:
rel-fun (*eq-onp* (λf. ($\forall x$. *0* $\leq$ *f x*) $\wedge$ ($\int^{+}x$. *ennreal* (*f x*) ∂*count-space UNIV*)

$= 1)$) *pmf-as-measure.cr-pmf* ($\lambda f.$ *density* (*count-space UNIV*) (*ennreal* $\circ$ f)) *embed-pmf*
⟨*proof*⟩

lemma *measure-pmf-eq-density*: *measure-pmf p* = *density* (*count-space UNIV*) (*pmf p*)
⟨*proof*⟩

lemma *td-pmf-embed-pmf*:
type-definition pmf embed-pmf $\{f::'a \Rightarrow real.\ (\forall x.\ 0 \leq f\ x) \wedge (\int^+ x.\ ennreal\ (f\ x)\ \partial count\text{-}space\ UNIV) = 1\}$
⟨*proof*⟩

end

lemma *nn-integral-measure-pmf*: $(\int^+ x.\ f\ x\ \partial measure\text{-}pmf\ p) = \int^+ x.\ ennreal\ (pmf\ p\ x) * f\ x\ \partial count\text{-}space\ UNIV$
⟨*proof*⟩

locale *pmf-as-function*
begin

setup-lifting *td-pmf-embed-pmf*

lemma *set-pmf-transfer*[*transfer-rule*]:
assumes *bi-total A*
shows *rel-fun* (*pcr-pmf A*) (*rel-set A*) $(\lambda f.\ \{x.\ f\ x \neq 0\})$ *set-pmf*
⟨*proof*⟩

end

context
begin

interpretation *pmf-as-function* ⟨*proof*⟩

lemma *pmf-eqI*: $(\bigwedge i.\ pmf\ M\ i = pmf\ N\ i) \Longrightarrow M = N$
⟨*proof*⟩

lemma *pmf-eq-iff*: $M = N \longleftrightarrow (\forall i.\ pmf\ M\ i = pmf\ N\ i)$
⟨*proof*⟩

lemma *bind-commute-pmf*: *bind-pmf A* ($\lambda x.$ *bind-pmf B* ($C\ x$)) = *bind-pmf B* ($\lambda y.$ *bind-pmf A* ($\lambda x.\ C\ x\ y$))
⟨*proof*⟩

lemma *pair-map-pmf1*: *pair-pmf* (*map-pmf f A*) *B* = *map-pmf* (*apfst f*) (*pair-pmf A B*)
⟨*proof*⟩

lemma *pair-map-pmf2*: *pair-pmf A* (*map-pmf f B*) = *map-pmf* (*apsnd f*) (*pair-pmf A B*)
⟨*proof*⟩

lemma *map-pair*: *map-pmf* (λ(*a*, *b*). (*f a*, *g b*)) (*pair-pmf A B*) = *pair-pmf* (*map-pmf f A*) (*map-pmf g B*)
⟨*proof*⟩

end

lemma *pair-return-pmf1*: *pair-pmf* (*return-pmf x*) *y* = *map-pmf* (*Pair x*) *y*
⟨*proof*⟩

lemma *pair-return-pmf2*: *pair-pmf x* (*return-pmf y*) = *map-pmf* (λ*x*. (*x*, *y*)) *x*
⟨*proof*⟩

lemma *pair-pair-pmf*: *pair-pmf* (*pair-pmf u v*) *w* = *map-pmf* (λ(*x*, (*y*, *z*)). ((*x*, *y*), *z*)) (*pair-pmf u* (*pair-pmf v w*))
⟨*proof*⟩

lemma *pair-commute-pmf*: *pair-pmf x y* = *map-pmf* (λ(*x*, *y*). (*y*, *x*)) (*pair-pmf y x*)
⟨*proof*⟩

lemma *set-pmf-subset-singleton*: *set-pmf p* ⊆ {*x*} ⟷ *p* = *return-pmf x*
⟨*proof*⟩

lemma *bind-eq-return-pmf*:
bind-pmf p f = *return-pmf x* ⟷ (∀ *y*∈*set-pmf p*. *f y* = *return-pmf x*)
(**is** *?lhs* ⟷ *?rhs*)
⟨*proof*⟩

lemma *pmf-False-conv-True*: *pmf p False* = *1* − *pmf p True*
⟨*proof*⟩

lemma *pmf-True-conv-False*: *pmf p True* = *1* − *pmf p False*
⟨*proof*⟩

25.4 Conditional Probabilities

lemma *measure-pmf-zero-iff*: *measure* (*measure-pmf p*) *s* = *0* ⟷ *set-pmf p* ∩ *s* = {}
⟨*proof*⟩

context
fixes *p* :: *'a pmf* **and** *s* :: *'a set*
assumes *not-empty*: *set-pmf p* ∩ *s* ≠ {}
begin

interpretation *pmf-as-measure* ⟨*proof*⟩

lemma *emeasure-measure-pmf-not-zero*: *emeasure* (*measure-pmf p*) *s* ≠ *0*
⟨*proof*⟩

lemma *measure-measure-pmf-not-zero*: *measure* (*measure-pmf p*) *s* ≠ *0*
⟨*proof*⟩

lift-definition *cond-pmf* :: *'a pmf* **is**
uniform-measure (*measure-pmf p*) *s*
⟨*proof*⟩

lemma *pmf-cond*: *pmf cond-pmf x* = (*if x* ∈ *s then pmf p x* / *measure p s else 0*)
⟨*proof*⟩

lemma *set-cond-pmf*[*simp*]: *set-pmf cond-pmf* = *set-pmf p* ∩ *s*
⟨*proof*⟩

end

lemma *cond-map-pmf*:
assumes *set-pmf p* ∩ *f* −' *s* ≠ {}
shows *cond-pmf* (*map-pmf f p*) *s* = *map-pmf f* (*cond-pmf p* (*f* −' *s*))
⟨*proof*⟩

lemma *bind-cond-pmf-cancel*:
assumes [*simp*]: ⋀*x*. *x* ∈ *set-pmf p* ⟹ *set-pmf q* ∩ {*y*. *R x y*} ≠ {}
assumes [*simp*]: ⋀*y*. *y* ∈ *set-pmf q* ⟹ *set-pmf p* ∩ {*x*. *R x y*} ≠ {}
assumes [*simp*]: ⋀*x y*. *x* ∈ *set-pmf p* ⟹ *y* ∈ *set-pmf q* ⟹ *R x y* ⟹ *measure q* {*y*. *R x y*} = *measure p* {*x*. *R x y*}
shows *bind-pmf p* (λ*x*. *cond-pmf q* {*y*. *R x y*}) = *q*
⟨*proof*⟩

25.5 Relator

inductive *rel-pmf* :: (*'a* ⇒ *'b* ⇒ *bool*) ⇒ *'a pmf* ⇒ *'b pmf* ⇒ *bool*
for *R p q*
where
⟦ ⋀*x y*. (*x*, *y*) ∈ *set-pmf pq* ⟹ *R x y*;
map-pmf fst pq = *p*; *map-pmf snd pq* = *q* ⟧
⟹ *rel-pmf R p q*

lemma *rel-pmfI*:
assumes *R*: *rel-set R* (*set-pmf p*) (*set-pmf q*)
assumes *eq*: ⋀*x y*. *x* ∈ *set-pmf p* ⟹ *y* ∈ *set-pmf q* ⟹ *R x y* ⟹
measure p {*x*. *R x y*} = *measure q* {*y*. *R x y*}
shows *rel-pmf R p q*
⟨*proof*⟩

lemma *rel-pmf-imp-rel-set*: *rel-pmf R p q* $\Longrightarrow$ *rel-set R* (*set-pmf p*) (*set-pmf q*)
⟨*proof*⟩

lemma *rel-pmfD-measure*:
assumes *rel-R*: *rel-pmf R p q* **and** *R*: $\bigwedge a\ b.\ R\ a\ b \Longrightarrow R\ a\ y \longleftrightarrow R\ x\ b$
assumes $x \in$ *set-pmf p* $y \in$ *set-pmf q*
shows *measure p* $\{x.\ R\ x\ y\}$ = *measure q* $\{y.\ R\ x\ y\}$
⟨*proof*⟩

lemma *rel-pmf-measureD*:
assumes *rel-pmf R p q*
shows *measure* (*measure-pmf p*) $A \leq$ *measure* (*measure-pmf q*) $\{y.\ \exists x{\in}A.\ R\ x\ y\}$ (**is** *?lhs* $\leq$ *?rhs*)
⟨*proof*⟩

lemma *rel-pmf-iff-measure*:
assumes *symp R transp R*
shows *rel-pmf R p q* $\longleftrightarrow$
rel-set R (*set-pmf p*) (*set-pmf q*) $\wedge$
($\forall x{\in}$*set-pmf p*. $\forall y{\in}$*set-pmf q*. $R\ x\ y \longrightarrow$ *measure p* $\{x.\ R\ x\ y\}$ = *measure q* $\{y.\ R\ x\ y\}$)
⟨*proof*⟩

lemma *quotient-rel-set-disjoint*:
equivp R $\Longrightarrow C \in$ *UNIV* // $\{(x, y).\ R\ x\ y\} \Longrightarrow$ *rel-set R A B* $\Longrightarrow A \cap C = \{\} \longleftrightarrow B \cap C = \{\}$
⟨*proof*⟩

lemma *quotientD*: *equiv X R* $\Longrightarrow A \in X$ // $R \Longrightarrow x \in A \Longrightarrow A = R$ `` $\{x\}$
⟨*proof*⟩

lemma *rel-pmf-iff-equivp*:
assumes *equivp R*
shows *rel-pmf R p q* $\longleftrightarrow$ ($\forall C{\in}$*UNIV* // $\{(x, y).\ R\ x\ y\}$. *measure p C* = *measure q C*)
(**is** - $\longleftrightarrow$ ($\forall C{\in}$-//*?R*. -))
⟨*proof*⟩

bnf *pmf*: *'a pmf map*: *map-pmf sets*: *set-pmf bd* : *natLeq rel*: *rel-pmf*
⟨*proof*⟩

lemma *map-pmf-idI*: ($\bigwedge x.\ x \in$ *set-pmf p* $\Longrightarrow f\ x = x$) $\Longrightarrow$ *map-pmf f p* = *p*
⟨*proof*⟩

lemma *rel-pmf-conj*[*simp*]:
rel-pmf ($\lambda x\ y.\ P \wedge Q\ x\ y$) $x\ y \longleftrightarrow P \wedge$ *rel-pmf Q x y*
rel-pmf ($\lambda x\ y.\ Q\ x\ y \wedge P$) $x\ y \longleftrightarrow P \wedge$ *rel-pmf Q x y*
⟨*proof*⟩

lemma *rel-pmf-top*[*simp*]: *rel-pmf top* = *top*
⟨*proof*⟩

lemma *rel-pmf-return-pmf1*: *rel-pmf R* (*return-pmf x*) *M* ⟷ (∀ *a*∈*M*. *R x a*)
⟨*proof*⟩

lemma *rel-pmf-return-pmf2*: *rel-pmf R M* (*return-pmf x*) ⟷ (∀ *a*∈*M*. *R a x*)
⟨*proof*⟩

lemma *rel-return-pmf*[*simp*]: *rel-pmf R* (*return-pmf x1*) (*return-pmf x2*) = *R x1 x2*
⟨*proof*⟩

lemma *rel-pmf-False*[*simp*]: *rel-pmf* (λ*x y*. *False*) *x y* = *False*
⟨*proof*⟩

lemma *rel-pmf-rel-prod*:
rel-pmf (*rel-prod R S*) (*pair-pmf A A′*) (*pair-pmf B B′*) ⟷ *rel-pmf R A B* ∧ *rel-pmf S A′ B′*
⟨*proof*⟩

lemma *rel-pmf-reflI*:
assumes ⋀*x*. *x* ∈ *set-pmf p* ⟹ *P x x*
shows *rel-pmf P p p*
⟨*proof*⟩

lemma *rel-pmf-bij-betw*:
assumes *f*: *bij-betw f* (*set-pmf p*) (*set-pmf q*)
and *eq*: ⋀*x*. *x* ∈ *set-pmf p* ⟹ *pmf p x* = *pmf q* (*f x*)
shows *rel-pmf* (λ*x y*. *f x* = *y*) *p q*
⟨*proof*⟩

context
begin

interpretation *pmf-as-measure* ⟨*proof*⟩

definition *join-pmf M* = *bind-pmf M* (λ*x*. *x*)

lemma *bind-eq-join-pmf*: *bind-pmf M f* = *join-pmf* (*map-pmf f M*)
⟨*proof*⟩

lemma *join-eq-bind-pmf*: *join-pmf M* = *bind-pmf M id*
⟨*proof*⟩

lemma *pmf-join*: *pmf* (*join-pmf N*) *i* = (∫ *M*. *pmf M i* ∂*measure-pmf N*)
⟨*proof*⟩

lemma *ennreal-pmf-join*: *ennreal* (*pmf* (*join-pmf* *N*) *i*) = ($\int^+$ *M*. *pmf* *M* *i* ∂*measure-pmf* *N*)
⟨*proof*⟩

lemma *set-pmf-join-pmf*[*simp*]: *set-pmf* (*join-pmf* *f*) = (⋃ *p*∈*set-pmf* *f*. *set-pmf* *p*)
⟨*proof*⟩

lemma *join-return-pmf*: *join-pmf* (*return-pmf* *M*) = *M*
⟨*proof*⟩

lemma *map-join-pmf*: *map-pmf* *f* (*join-pmf* *AA*) = *join-pmf* (*map-pmf* (*map-pmf* *f*) *AA*)
⟨*proof*⟩

lemma *join-map-return-pmf*: *join-pmf* (*map-pmf* *return-pmf* *A*) = *A*
⟨*proof*⟩

end

lemma *rel-pmf-joinI*:
assumes *rel-pmf* (*rel-pmf* *P*) *p* *q*
shows *rel-pmf* *P* (*join-pmf* *p*) (*join-pmf* *q*)
⟨*proof*⟩

lemma *rel-pmf-bindI*:
assumes *pq*: *rel-pmf* *R* *p* *q*
and *fg*: ⋀*x* *y*. *R* *x* *y* ⟹ *rel-pmf* *P* (*f* *x*) (*g* *y*)
shows *rel-pmf* *P* (*bind-pmf* *p* *f*) (*bind-pmf* *q* *g*)
⟨*proof*⟩

Proof that *rel-pmf* preserves orders. Antisymmetry proof follows Thm. 1 in N. Saheb-Djahromi, Cpo's of measures for nondeterminism, Theoretical Computer Science 12(1):19–37, 1980, http://dx.doi.org/10.1016/0304-3975(80)90003-1

lemma
assumes ∗: *rel-pmf* *R* *p* *q*
and *refl*: *reflp* *R* **and** *trans*: *transp* *R*
shows *measure-Ici*: *measure* *p* {*y*. *R* *x* *y*} ≤ *measure* *q* {*y*. *R* *x* *y*} (**is** *?thesis1*)
and *measure-Ioi*: *measure* *p* {*y*. *R* *x* *y* ∧ ¬ *R* *y* *x*} ≤ *measure* *q* {*y*. *R* *x* *y* ∧ ¬ *R* *y* *x*} (**is** *?thesis2*)
⟨*proof*⟩

lemma *rel-pmf-inf*:
fixes *p* *q* :: *'a* *pmf*
assumes *1*: *rel-pmf* *R* *p* *q*
assumes *2*: *rel-pmf* *R* *q* *p*
and *refl*: *reflp* *R* **and** *trans*: *transp* *R*
shows *rel-pmf* (*inf* *R* R^{-1-1}) *p* *q*

⟨*proof*⟩

lemma *rel-pmf-antisym*:
fixes *p q* :: *'a pmf*
assumes *1*: *rel-pmf R p q*
assumes *2*: *rel-pmf R q p*
and *refl*: *reflp R* **and** *trans*: *transp R* **and** *antisym*: *antisymP R*
shows *p = q*
⟨*proof*⟩

lemma *reflp-rel-pmf*: *reflp R ⟹ reflp (rel-pmf R)*
⟨*proof*⟩

lemma *antisymP-rel-pmf*:
⟦ *reflp R*; *transp R*; *antisymP R* ⟧
⟹ *antisymP (rel-pmf R)*
⟨*proof*⟩

lemma *transp-rel-pmf*:
assumes *transp R*
shows *transp (rel-pmf R)*
⟨*proof*⟩

25.6 Distributions

context
begin

interpretation *pmf-as-function* ⟨*proof*⟩

25.6.1 Bernoulli Distribution

lift-definition *bernoulli-pmf* :: *real ⇒ bool pmf* **is**
λp b. ((λp. if b then p else 1 − p) ∘ min 1 ∘ max 0) p
⟨*proof*⟩

lemma *pmf-bernoulli-True*[*simp*]: *0 ≤ p ⟹ p ≤ 1 ⟹ pmf (bernoulli-pmf p) True = p*
⟨*proof*⟩

lemma *pmf-bernoulli-False*[*simp*]: *0 ≤ p ⟹ p ≤ 1 ⟹ pmf (bernoulli-pmf p) False = 1 − p*
⟨*proof*⟩

lemma *set-pmf-bernoulli*[*simp*]: *0 < p ⟹ p < 1 ⟹ set-pmf (bernoulli-pmf p) = UNIV*
⟨*proof*⟩

lemma *nn-integral-bernoulli-pmf*[*simp*]:
assumes [*simp*]: *0 ≤ p p ≤ 1 ⋀x. 0 ≤ f x*

shows ($\int^+ x.\ f\ x\ \partial$*bernoulli-pmf p*) = *f True* $*$ *p* + *f False* $*$ (*1* − *p*)
⟨*proof*⟩

lemma *integral-bernoulli-pmf*[*simp*]:
assumes [*simp*]: *0* ≤ *p p* ≤ *1*
shows ($\int x.\ f\ x\ \partial$*bernoulli-pmf p*) = *f True* $*$ *p* + *f False* $*$ (*1* − *p*)
⟨*proof*⟩

lemma *pmf-bernoulli-half* [*simp*]: *pmf* (*bernoulli-pmf* (*1* / *2*)) *x* = *1* / *2*
⟨*proof*⟩

lemma *measure-pmf-bernoulli-half*: *measure-pmf* (*bernoulli-pmf* (*1* / *2*)) = *uniform-count-measure UNIV*
⟨*proof*⟩

25.6.2 Geometric Distribution

context
fixes *p* :: *real* **assumes** *p*[*arith*]: *0* < *p p* ≤ *1*
begin

lift-definition *geometric-pmf* :: *nat pmf* **is** λ*n*. (*1* − *p*) ^*n* $*$ *p*
⟨*proof*⟩

lemma *pmf-geometric*[*simp*]: *pmf geometric-pmf n* = (*1* − *p*) ^*n* $*$ *p*
⟨*proof*⟩

end

lemma *set-pmf-geometric*: *0* < *p* ⟹ *p* < *1* ⟹ *set-pmf* (*geometric-pmf p*) = *UNIV*
⟨*proof*⟩

25.6.3 Uniform Multiset Distribution

context
fixes *M* :: *'a multiset* **assumes** *M-not-empty*: *M* ≠ {#}
begin

lift-definition *pmf-of-multiset* :: *'a pmf* **is** λ*x*. *count M x* / *size M*
⟨*proof*⟩

lemma *pmf-of-multiset*[*simp*]: *pmf pmf-of-multiset x* = *count M x* / *size M*
⟨*proof*⟩

lemma *set-pmf-of-multiset*[*simp*]: *set-pmf pmf-of-multiset* = *set-mset M*
⟨*proof*⟩

end

25.6.4 Uniform Distribution

context
fixes *S* :: *'a set* **assumes** *S-not-empty*: $S \neq \{\}$ **and** *S-finite*: *finite S*
begin

lift-definition *pmf-of-set* :: *'a pmf* **is** $\lambda x.$ *indicator S x / card S*
⟨*proof*⟩

lemma *pmf-of-set*[*simp*]: *pmf pmf-of-set x* = *indicator S x / card S*
⟨*proof*⟩

lemma *set-pmf-of-set*[*simp*]: *set-pmf pmf-of-set* = *S*
⟨*proof*⟩

lemma *emeasure-pmf-of-set-space*[*simp*]: *emeasure pmf-of-set S* = *1*
⟨*proof*⟩

lemma *nn-integral-pmf-of-set*: *nn-integral* (*measure-pmf pmf-of-set*) *f* = *setsum f S / card S*
⟨*proof*⟩

lemma *integral-pmf-of-set*: $integral^L$ (*measure-pmf pmf-of-set*) *f* = *setsum f S / card S*
⟨*proof*⟩

lemma *emeasure-pmf-of-set*: *emeasure* (*measure-pmf pmf-of-set*) *A* = *card* ($S \cap A$) / *card S*
⟨*proof*⟩

lemma *measure-pmf-of-set*: *measure* (*measure-pmf pmf-of-set*) *A* = *card* ($S \cap A$) / *card S*
⟨*proof*⟩

end

lemma *pmf-of-set-singleton*: *pmf-of-set* $\{x\}$ = *return-pmf x*
⟨*proof*⟩

lemma *map-pmf-of-set-inj*:
assumes *f*: *inj-on f A*
and [*simp*]: $A \neq \{\}$ *finite A*
shows *map-pmf f* (*pmf-of-set A*) = *pmf-of-set* (*f* ‘ *A*) (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *bernoulli-pmf-half-conv-pmf-of-set*: *bernoulli-pmf* (*1* / *2*) = *pmf-of-set UNIV*
⟨*proof*⟩

25.6.5 Poisson Distribution

context
fixes *rate* :: *real* **assumes** *rate-pos*: $0 < rate$
begin

lift-definition *poisson-pmf* :: *nat pmf* **is** $\lambda k.\ rate\ \hat{}\ k\ /\ fact\ k * exp\ (-rate)$
⟨*proof*⟩

lemma *pmf-poisson*[*simp*]: *pmf poisson-pmf k* = $rate\ \hat{}\ k\ /\ fact\ k * exp\ (-rate)$
⟨*proof*⟩

lemma *set-pmf-poisson*[*simp*]: *set-pmf poisson-pmf* = *UNIV*
⟨*proof*⟩

end

25.6.6 Binomial Distribution

context
fixes *n* :: *nat* **and** *p* :: *real* **assumes** *p-nonneg*: $0 \leq p$ **and** *p-le-1*: $p \leq 1$
begin

lift-definition *binomial-pmf* :: *nat pmf* **is** $\lambda k.\ (n\ choose\ k) * p\hat{}k * (1 - p)\hat{}(n - k)$
⟨*proof*⟩

lemma *pmf-binomial*[*simp*]: *pmf binomial-pmf k* = $(n\ choose\ k) * p\hat{}k * (1 - p)\hat{}(n - k)$
⟨*proof*⟩

lemma *set-pmf-binomial-eq*: *set-pmf binomial-pmf* = (*if* $p = 0$ *then* $\{0\}$ *else if* $p = 1$ *then* $\{n\}$ *else* $\{..\ n\}$)
⟨*proof*⟩

end

end

lemma *set-pmf-binomial-0*[*simp*]: *set-pmf* (*binomial-pmf n 0*) = $\{0\}$
⟨*proof*⟩

lemma *set-pmf-binomial-1*[*simp*]: *set-pmf* (*binomial-pmf n 1*) = $\{n\}$
⟨*proof*⟩

lemma *set-pmf-binomial*[*simp*]: $0 < p \Longrightarrow p < 1 \Longrightarrow$ *set-pmf* (*binomial-pmf n p*) = $\{..n\}$
⟨*proof*⟩

context begin interpretation *lifting-syntax* ⟨*proof*⟩

lemma *bind-pmf-parametric* [*transfer-rule*]:
(*rel-pmf A* ===> (*A* ===> *rel-pmf B*) ===> *rel-pmf B*) *bind-pmf bind-pmf*
⟨*proof*⟩

lemma *return-pmf-parametric* [*transfer-rule*]: (*A* ===> *rel-pmf A*) *return-pmf return-pmf*
⟨*proof*⟩

end

end

26 Infinite Streams

theory *Stream*
imports ~~/*src*/*HOL*/*Library*/*Nat-Bijection*
begin

codatatype (*sset*: ′*a*) *stream* =
SCons (*shd*: ′*a*) (*stl*: ′*a stream*) (**infixr** ## *65*)
for
map: *smap*
rel: *stream-all2*

context
begin

qualified definition *smember* :: ′*a* ⇒ ′*a stream* ⇒ *bool* **where**
[*code-abbrev*]: *smember x s* ⟷ *x* ∈ *sset s*

lemma *smember-code*[*code*, *simp*]: *smember x* (*y* ## *s*) = (*if x* = *y then True else smember x s*)
⟨*proof*⟩

end

lemmas *smap-simps*[*simp*] = *stream.map-sel*
lemmas *shd-sset* = *stream.set-sel*(*1*)
lemmas *stl-sset* = *stream.set-sel*(*2*)

theorem *sset-induct*[*consumes 1*, *case-names shd stl*, *induct set*: *sset*]:
assumes *y* ∈ *sset s* **and** ⋀*s*. *P* (*shd s*) *s* **and** ⋀*s y*. ⟦*y* ∈ *sset* (*stl s*); *P y* (*stl s*)⟧ ⟹ *P y s*
shows *P y s*
⟨*proof*⟩

lemma *smap-ctr*: *smap f s* = *x* ## *s*′ ⟷ *f* (*shd s*) = *x* ∧ *smap f* (*stl s*) = *s*′

⟨*proof*⟩

26.1 prepend list to stream

primrec *shift* :: *'a list* ⇒ *'a stream* ⇒ *'a stream* (**infixr** @− *65*) **where**
shift [] *s* = *s*
| *shift* (*x* # *xs*) *s* = *x* ## *shift xs s*

lemma *smap-shift*[*simp*]: *smap f* (*xs* @− *s*) = *map f xs* @− *smap f s*
⟨*proof*⟩

lemma *shift-append*[*simp*]: (*xs* @ *ys*) @− *s* = *xs* @− *ys* @− *s*
⟨*proof*⟩

lemma *shift-simps*[*simp*]:
shd (*xs* @− *s*) = (*if xs* = [] *then shd s else hd xs*)
stl (*xs* @− *s*) = (*if xs* = [] *then stl s else tl xs* @− *s*)
⟨*proof*⟩

lemma *sset-shift*[*simp*]: *sset* (*xs* @− *s*) = *set xs* ∪ *sset s*
⟨*proof*⟩

lemma *shift-left-inj*[*simp*]: *xs* @− *s1* = *xs* @− *s2* ⟷ *s1* = *s2*
⟨*proof*⟩

26.2 set of streams with elements in some fixed set

context
notes [[*inductive-internals*]]
begin

coinductive-set
streams :: *'a set* ⇒ *'a stream set*
for *A* :: *'a set*
where
Stream[*intro*!, *simp*, *no-atp*]: ⟦*a* ∈ *A*; *s* ∈ *streams A*⟧ ⟹ *a* ## *s* ∈ *streams A*

end

lemma *in-streams*: *stl s* ∈ *streams S* ⟹ *shd s* ∈ *S* ⟹ *s* ∈ *streams S*
⟨*proof*⟩

lemma *streamsE*: *s* ∈ *streams A* ⟹ (*shd s* ∈ *A* ⟹ *stl s* ∈ *streams A* ⟹ *P*) ⟹ *P*
⟨*proof*⟩

lemma *Stream-image*: *x* ## *y* ∈ (*op* ## *x'*) ‘ *Y* ⟷ *x* = *x'* ∧ *y* ∈ *Y*
⟨*proof*⟩

lemma *shift-streams*: ⟦*w* ∈ *lists A*; *s* ∈ *streams A*⟧ ⟹ *w* @− *s* ∈ *streams A*

⟨*proof*⟩

lemma *streams-Stream*: $x\ \#\#\ s \in streams\ A \longleftrightarrow x \in A \wedge s \in streams\ A$
⟨*proof*⟩

lemma *streams-stl*: $s \in streams\ A \Longrightarrow stl\ s \in streams\ A$
⟨*proof*⟩

lemma *streams-shd*: $s \in streams\ A \Longrightarrow shd\ s \in A$
⟨*proof*⟩

lemma *sset-streams*:
assumes $sset\ s \subseteq A$
shows $s \in streams\ A$
⟨*proof*⟩

lemma *streams-sset*:
assumes $s \in streams\ A$
shows $sset\ s \subseteq A$
⟨*proof*⟩

lemma *streams-iff-sset*: $s \in streams\ A \longleftrightarrow sset\ s \subseteq A$
⟨*proof*⟩

lemma *streams-mono*: $s \in streams\ A \Longrightarrow A \subseteq B \Longrightarrow s \in streams\ B$
⟨*proof*⟩

lemma *streams-mono2*: $S \subseteq T \Longrightarrow streams\ S \subseteq streams\ T$
⟨*proof*⟩

lemma *smap-streams*: $s \in streams\ A \Longrightarrow (\bigwedge x.\ x \in A \Longrightarrow f\ x \in B) \Longrightarrow smap\ f\ s \in streams\ B$
⟨*proof*⟩

lemma *streams-empty*: $streams\ \{\} = \{\}$
⟨*proof*⟩

lemma *streams-UNIV*[*simp*]: $streams\ UNIV = UNIV$
⟨*proof*⟩

26.3 nth, take, drop for streams

primrec *snth* :: $'a\ stream \Rightarrow nat \Rightarrow 'a$ (**infixl** !! *100*) **where**
$s\ !!\ 0 = shd\ s$
$|\ s\ !!\ Suc\ n = stl\ s\ !!\ n$

lemma *snth-Stream*: $(x\ \#\#\ s)\ !!\ Suc\ i = s\ !!\ i$
⟨*proof*⟩

lemma *snth-smap*[*simp*]: *smap f s* !! *n* = *f* (*s* !! *n*)
⟨*proof*⟩

lemma *shift-snth-less*[*simp*]: *p* < *length xs* ⟹ (*xs* @− *s*) !! *p* = *xs* ! *p*
⟨*proof*⟩

lemma *shift-snth-ge*[*simp*]: *p* ≥ *length xs* ⟹ (*xs* @− *s*) !! *p* = *s* !! (*p* − *length xs*)
⟨*proof*⟩

lemma *shift-snth*: (*xs* @− *s*) !! *n* = (*if n* < *length xs then xs* ! *n else s* !! (*n* − *length xs*))
⟨*proof*⟩

lemma *snth-sset*[*simp*]: *s* !! *n* ∈ *sset s*
⟨*proof*⟩

lemma *sset-range*: *sset s* = *range* (*snth s*)
⟨*proof*⟩

lemma *streams-iff-snth*: *s* ∈ *streams X* ⟷ (∀ *n*. *s* !! *n* ∈ *X*)
⟨*proof*⟩

lemma *snth-in*: *s* ∈ *streams X* ⟹ *s* !! *n* ∈ *X*
⟨*proof*⟩

primrec *stake* :: *nat* ⇒ *'a stream* ⇒ *'a list* **where**
stake 0 s = []
| *stake* (*Suc n*) *s* = *shd s* # *stake n* (*stl s*)

lemma *length-stake*[*simp*]: *length* (*stake n s*) = *n*
⟨*proof*⟩

lemma *stake-smap*[*simp*]: *stake n* (*smap f s*) = *map f* (*stake n s*)
⟨*proof*⟩

lemma *take-stake*: *take n* (*stake m s*) = *stake* (*min n m*) *s*
⟨*proof*⟩

primrec *sdrop* :: *nat* ⇒ *'a stream* ⇒ *'a stream* **where**
sdrop 0 s = *s*
| *sdrop* (*Suc n*) *s* = *sdrop n* (*stl s*)

lemma *sdrop-simps*[*simp*]:
shd (*sdrop n s*) = *s* !! *n stl* (*sdrop n s*) = *sdrop* (*Suc n*) *s*
⟨*proof*⟩

lemma *sdrop-smap*[*simp*]: *sdrop n* (*smap f s*) = *smap f* (*sdrop n s*)
⟨*proof*⟩

lemma *sdrop-stl*: *sdrop n* (*stl s*) = *stl* (*sdrop n s*)
⟨*proof*⟩

lemma *drop-stake*: *drop n* (*stake m s*) = *stake* (*m* − *n*) (*sdrop n s*)
⟨*proof*⟩

lemma *stake-sdrop*: *stake n s* @− *sdrop n s* = *s*
⟨*proof*⟩

lemma *id-stake-snth-sdrop*:
s = *stake i s* @− *s* !! *i* ## *sdrop* (*Suc i*) *s*
⟨*proof*⟩

lemma *smap-alt*: *smap f s* = *s′* ⟷ (∀ *n*. *f* (*s* !! *n*) = *s′* !! *n*) (**is** *?L* = *?R*)
⟨*proof*⟩

lemma *stake-invert-Nil*[*iff*]: *stake n s* = [] ⟷ *n* = *0*
⟨*proof*⟩

lemma *sdrop-shift*: *sdrop i* (*w* @− *s*) = *drop i w* @− *sdrop* (*i* − *length w*) *s*
⟨*proof*⟩

lemma *stake-shift*: *stake i* (*w* @− *s*) = *take i w* @ *stake* (*i* − *length w*) *s*
⟨*proof*⟩

lemma *stake-add*[*simp*]: *stake m s* @ *stake n* (*sdrop m s*) = *stake* (*m* + *n*) *s*
⟨*proof*⟩

lemma *sdrop-add*[*simp*]: *sdrop n* (*sdrop m s*) = *sdrop* (*m* + *n*) *s*
⟨*proof*⟩

lemma *sdrop-snth*: *sdrop n s* !! *m* = *s* !! (*n* + *m*)
⟨*proof*⟩

partial-function (*tailrec*) *sdrop-while* :: (*′a* ⇒ *bool*) ⇒ *′a stream* ⇒ *′a stream*
where
sdrop-while P s = (*if P* (*shd s*) *then sdrop-while P* (*stl s*) *else s*)

lemma *sdrop-while-SCons*[*code*]:
sdrop-while P (*a* ## *s*) = (*if P a then sdrop-while P s else a* ## *s*)
⟨*proof*⟩

lemma *sdrop-while-sdrop-LEAST*:
assumes ∃ *n*. *P* (*s* !! *n*)
shows *sdrop-while* (*Not o P*) *s* = *sdrop* (*LEAST n*. *P* (*s* !! *n*)) *s*
⟨*proof*⟩

primcorec *sfilter* **where**

shd (*sfilter P s*) = *shd* (*sdrop-while* (*Not o P*) *s*)
| *stl* (*sfilter P s*) = *sfilter P* (*stl* (*sdrop-while* (*Not o P*) *s*))

lemma *sfilter-Stream*: *sfilter P* (*x* ## *s*) = (*if P x then x* ## *sfilter P s else sfilter P s*)
⟨*proof*⟩

26.4 unary predicates lifted to streams

definition *stream-all P s* = (∀ *p*. *P* (*s* !! *p*))

lemma *stream-all-iff* [*iff*]: *stream-all P s* ⟷ *Ball* (*sset s*) *P*
⟨*proof*⟩

lemma *stream-all-shift*[*simp*]: *stream-all P* (*xs* @− *s*) = (*list-all P xs* ∧ *stream-all P s*)
⟨*proof*⟩

lemma *stream-all-Stream*: *stream-all P* (*x* ## *X*) ⟷ *P x* ∧ *stream-all P X*
⟨*proof*⟩

26.5 recurring stream out of a list

primcorec *cycle* :: *'a list* ⇒ *'a stream* **where**
shd (*cycle xs*) = *hd xs*
| *stl* (*cycle xs*) = *cycle* (*tl xs* @ [*hd xs*])

lemma *cycle-decomp*: *u* ≠ [] ⟹ *cycle u* = *u* @− *cycle u*
⟨*proof*⟩

lemma *cycle-Cons*[*code*]: *cycle* (*x* # *xs*) = *x* ## *cycle* (*xs* @ [*x*])
⟨*proof*⟩

lemma *cycle-rotated*: ⟦*v* ≠ []; *cycle u* = *v* @− *s*⟧ ⟹ *cycle* (*tl u* @ [*hd u*]) = *tl v* @− *s*
⟨*proof*⟩

lemma *stake-append*: *stake n* (*u* @− *s*) = *take* (*min* (*length u*) *n*) *u* @ *stake* (*n* − *length u*) *s*
⟨*proof*⟩

lemma *stake-cycle-le*[*simp*]:
assumes *u* ≠ [] *n* < *length u*
shows *stake n* (*cycle u*) = *take n u*
⟨*proof*⟩

lemma *stake-cycle-eq*[*simp*]: *u* ≠ [] ⟹ *stake* (*length u*) (*cycle u*) = *u*
⟨*proof*⟩

lemma *sdrop-cycle-eq*[*simp*]: *u* ≠ [] ⟹ *sdrop* (*length u*) (*cycle u*) = *cycle u*

⟨proof⟩

lemma *stake-cycle-eq-mod-0*[*simp*]: $[\![u \neq []; n \text{ mod length } u = 0]\!] \Longrightarrow$
stake n (cycle u) = concat (replicate (n div length u) u)
⟨proof⟩

lemma *sdrop-cycle-eq-mod-0*[*simp*]: $[\![u \neq []; n \text{ mod length } u = 0]\!] \Longrightarrow$
sdrop n (cycle u) = cycle u
⟨proof⟩

lemma *stake-cycle*: $u \neq [] \Longrightarrow$
stake n (cycle u) = concat (replicate (n div length u) u) @ take (n mod length u) u
⟨proof⟩

lemma *sdrop-cycle*: $u \neq [] \Longrightarrow$ *sdrop n (cycle u) = cycle (rotate (n mod length u) u)*
⟨proof⟩

26.6 iterated application of a function

primcorec *siterate* **where**
shd (siterate f x) = x
| *stl (siterate f x) = siterate f (f x)*

lemma *stake-Suc*: *stake (Suc n) s = stake n s @ [s !! n]*
⟨proof⟩

lemma *snth-siterate*[*simp*]: *siterate f x !! n = (f^^n) x*
⟨proof⟩

lemma *sdrop-siterate*[*simp*]: *sdrop n (siterate f x) = siterate f ((f^^n) x)*
⟨proof⟩

lemma *stake-siterate*[*simp*]: *stake n (siterate f x) = map (λn. (f^^n) x) [0 ..< n]*
⟨proof⟩

lemma *sset-siterate*: *sset (siterate f x) = {(f^^n) x | n. True}*
⟨proof⟩

lemma *smap-siterate*: *smap f (siterate f x) = siterate f (f x)*
⟨proof⟩

26.7 stream repeating a single element

abbreviation *sconst* ≡ *siterate id*

lemma *shift-replicate-sconst*[*simp*]: *replicate n x @− sconst x = sconst x*
⟨proof⟩

lemma *sset-sconst*[*simp*]: *sset* (*sconst x*) = {*x*}
⟨*proof*⟩

lemma *sconst-alt*: *s* = *sconst x* ⟷ *sset s* = {*x*}
⟨*proof*⟩

lemma *sconst-cycle*: *sconst x* = *cycle* [*x*]
⟨*proof*⟩

lemma *smap-sconst*: *smap f* (*sconst x*) = *sconst* (*f x*)
⟨*proof*⟩

lemma *sconst-streams*: *x* ∈ *A* ⟹ *sconst x* ∈ *streams A*
⟨*proof*⟩

26.8 stream of natural numbers

abbreviation *fromN* ≡ *siterate Suc*

abbreviation *nats* ≡ *fromN 0*

lemma *sset-fromN*[*simp*]: *sset* (*fromN n*) = {*n* ..}
⟨*proof*⟩

lemma *stream-smap-fromN*: *s* = *smap* (λ*j*. *let i* = *j* − *n in s* !! *i*) (*fromN n*)
⟨*proof*⟩

lemma *stream-smap-nats*: *s* = *smap* (*snth s*) *nats*
⟨*proof*⟩

26.9 flatten a stream of lists

primcorec *flat* **where**
shd (*flat ws*) = *hd* (*shd ws*)
| *stl* (*flat ws*) = *flat* (*if tl* (*shd ws*) = [] *then stl ws else tl* (*shd ws*) ## *stl ws*)

lemma *flat-Cons*[*simp*, *code*]: *flat* ((*x* # *xs*) ## *ws*) = *x* ## *flat* (*if xs* = [] *then ws else xs* ## *ws*)
⟨*proof*⟩

lemma *flat-Stream*[*simp*]: *xs* ≠ [] ⟹ *flat* (*xs* ## *ws*) = *xs* @− *flat ws*
⟨*proof*⟩

lemma *flat-unfold*: *shd ws* ≠ [] ⟹ *flat ws* = *shd ws* @− *flat* (*stl ws*)
⟨*proof*⟩

lemma *flat-snth*: ∀ *xs* ∈ *sset s*. *xs* ≠ [] ⟹ *flat s* !! *n* = (*if n* < *length* (*shd s*) *then*
shd s ! *n else flat* (*stl s*) !! (*n* − *length* (*shd s*)))
⟨*proof*⟩

lemma *sset-flat*[*simp*]: $\forall xs \in sset\ s.\ xs \neq [] \Longrightarrow$
$sset\ (flat\ s) = (\bigcup xs \in sset\ s.\ set\ xs)$ (**is** $?P \Longrightarrow ?L = ?R$)
⟨*proof*⟩

26.10 merge a stream of streams

definition *smerge* :: $'a\ stream\ stream \Rightarrow 'a\ stream$ **where**
$smerge\ ss = flat\ (smap\ (\lambda n.\ map\ (\lambda s.\ s\ !!\ n)\ (stake\ (Suc\ n)\ ss)\ @\ stake\ n\ (ss\ !!\ n))\ nats)$

lemma *stake-nth*[*simp*]: $m < n \Longrightarrow stake\ n\ s\ !\ m = s\ !!\ m$
⟨*proof*⟩

lemma *snth-sset-smerge*: $ss\ !!\ n\ !!\ m \in sset\ (smerge\ ss)$
⟨*proof*⟩

lemma *sset-smerge*: $sset\ (smerge\ ss) = UNION\ (sset\ ss)\ sset$
⟨*proof*⟩

26.11 product of two streams

definition *sproduct* :: $'a\ stream \Rightarrow 'b\ stream \Rightarrow ('a \times 'b)\ stream$ **where**
$sproduct\ s1\ s2 = smerge\ (smap\ (\lambda x.\ smap\ (Pair\ x)\ s2)\ s1)$

lemma *sset-sproduct*: $sset\ (sproduct\ s1\ s2) = sset\ s1 \times sset\ s2$
⟨*proof*⟩

26.12 interleave two streams

primcorec *sinterleave* **where**
$shd\ (sinterleave\ s1\ s2) = shd\ s1$
$|\ stl\ (sinterleave\ s1\ s2) = sinterleave\ s2\ (stl\ s1)$

lemma *sinterleave-code*[*code*]:
$sinterleave\ (x\ \#\#\ s1)\ s2 = x\ \#\#\ sinterleave\ s2\ s1$
⟨*proof*⟩

lemma *sinterleave-snth*[*simp*]:
$even\ n \Longrightarrow sinterleave\ s1\ s2\ !!\ n = s1\ !!\ (n\ div\ 2)$
$odd\ n \Longrightarrow sinterleave\ s1\ s2\ !!\ n = s2\ !!\ (n\ div\ 2)$
⟨*proof*⟩

lemma *sset-sinterleave*: $sset\ (sinterleave\ s1\ s2) = sset\ s1 \cup sset\ s2$
⟨*proof*⟩

26.13 zip

primcorec *szip* **where**
$shd\ (szip\ s1\ s2) = (shd\ s1,\ shd\ s2)$

| *stl* (*szip s1 s2*) = *szip* (*stl s1*) (*stl s2*)

lemma *szip-unfold*[*code*]: *szip* (*a* ## *s1*) (*b* ## *s2*) = (*a*, *b*) ## (*szip s1 s2*)
⟨*proof*⟩

lemma *snth-szip*[*simp*]: *szip s1 s2* !! *n* = (*s1* !! *n*, *s2* !! *n*)
⟨*proof*⟩

lemma *stake-szip*[*simp*]:
stake n (*szip s1 s2*) = *zip* (*stake n s1*) (*stake n s2*)
⟨*proof*⟩

lemma *sdrop-szip*[*simp*]: *sdrop n* (*szip s1 s2*) = *szip* (*sdrop n s1*) (*sdrop n s2*)
⟨*proof*⟩

lemma *smap-szip-fst*:
smap (λ*x*. *f* (*fst x*)) (*szip s1 s2*) = *smap f s1*
⟨*proof*⟩

lemma *smap-szip-snd*:
smap (λ*x*. *g* (*snd x*)) (*szip s1 s2*) = *smap g s2*
⟨*proof*⟩

26.14 zip via function

primcorec *smap2* **where**
shd (*smap2 f s1 s2*) = *f* (*shd s1*) (*shd s2*)
| *stl* (*smap2 f s1 s2*) = *smap2 f* (*stl s1*) (*stl s2*)

lemma *smap2-unfold*[*code*]:
smap2 f (*a* ## *s1*) (*b* ## *s2*) = *f a b* ## (*smap2 f s1 s2*)
⟨*proof*⟩

lemma *smap2-szip*:
smap2 f s1 s2 = *smap* (*case-prod f*) (*szip s1 s2*)
⟨*proof*⟩

lemma *smap-smap2*[*simp*]:
smap f (*smap2 g s1 s2*) = *smap2* (λ*x y*. *f* (*g x y*)) *s1 s2*
⟨*proof*⟩

lemma *smap2-alt*:
(*smap2 f s1 s2* = *s*) = (∀ *n*. *f* (*s1* !! *n*) (*s2* !! *n*) = *s* !! *n*)
⟨*proof*⟩

lemma *snth-smap2*[*simp*]:
smap2 f s1 s2 !! *n* = *f* (*s1* !! *n*) (*s2* !! *n*)
⟨*proof*⟩

lemma *stake-smap2*[*simp*]:
stake n (*smap2 f s1 s2*) = *map* (*case-prod f*) (*zip* (*stake n s1*) (*stake n s2*))
⟨*proof*⟩

lemma *sdrop-smap2*[*simp*]:
sdrop n (*smap2 f s1 s2*) = *smap2 f* (*sdrop n s1*) (*sdrop n s2*)
⟨*proof*⟩

end

27 List prefixes, suffixes, and homeomorphic embedding

theory *Sublist*
imports *Main*
begin

27.1 Prefix order on lists

definition *prefixeq* :: *'a list* ⇒ *'a list* ⇒ *bool*
where *prefixeq xs ys* ⟷ (∃ *zs*. *ys* = *xs* @ *zs*)

definition *prefix* :: *'a list* ⇒ *'a list* ⇒ *bool*
where *prefix xs ys* ⟷ *prefixeq xs ys* ∧ *xs* ≠ *ys*

interpretation *prefix-order*: *order prefixeq prefix*
⟨*proof*⟩

interpretation *prefix-bot*: *order-bot Nil prefixeq prefix*
⟨*proof*⟩

lemma *prefixeqI* [*intro?*]: *ys* = *xs* @ *zs* ⟹ *prefixeq xs ys*
⟨*proof*⟩

lemma *prefixeqE* [*elim?*]:
assumes *prefixeq xs ys*
obtains *zs* **where** *ys* = *xs* @ *zs*
⟨*proof*⟩

lemma *prefixI'* [*intro?*]: *ys* = *xs* @ *z* # *zs* ⟹ *prefix xs ys*
⟨*proof*⟩

lemma *prefixE'* [*elim?*]:
assumes *prefix xs ys*
obtains *z zs* **where** *ys* = *xs* @ *z* # *zs*
⟨*proof*⟩

lemma *prefixI* [*intro?*]: *prefixeq xs ys* ⟹ *xs* ≠ *ys* ⟹ *prefix xs ys*

⟨*proof*⟩

lemma *prefixE* [*elim?*]:
fixes *xs ys* :: *'a list*
assumes *prefix xs ys*
obtains *prefixeq xs ys* **and** $xs \neq ys$
⟨*proof*⟩

27.2 Basic properties of prefixes

theorem *Nil-prefixeq* [*iff*]: *prefixeq* [] *xs*
⟨*proof*⟩

theorem *prefixeq-Nil* [*simp*]: (*prefixeq xs* []) = (*xs* = [])
⟨*proof*⟩

lemma *prefixeq-snoc* [*simp*]: *prefixeq xs* (*ys* @ [*y*]) ⟷ *xs* = *ys* @ [*y*] ∨ *prefixeq xs ys*
⟨*proof*⟩

lemma *Cons-prefixeq-Cons* [*simp*]: *prefixeq* (*x* # *xs*) (*y* # *ys*) = (*x* = *y* ∧ *prefixeq xs ys*)
⟨*proof*⟩

lemma *prefixeq-code* [*code*]:
prefixeq [] *xs* ⟷ *True*
prefixeq (*x* # *xs*) [] ⟷ *False*
prefixeq (*x* # *xs*) (*y* # *ys*) ⟷ *x* = *y* ∧ *prefixeq xs ys*
⟨*proof*⟩

lemma *same-prefixeq-prefixeq* [*simp*]: *prefixeq* (*xs* @ *ys*) (*xs* @ *zs*) = *prefixeq ys zs*
⟨*proof*⟩

lemma *same-prefixeq-nil* [*iff*]: *prefixeq* (*xs* @ *ys*) *xs* = (*ys* = [])
⟨*proof*⟩

lemma *prefixeq-prefixeq* [*simp*]: *prefixeq xs ys* ⟹ *prefixeq xs* (*ys* @ *zs*)
⟨*proof*⟩

lemma *append-prefixeqD*: *prefixeq* (*xs* @ *ys*) *zs* ⟹ *prefixeq xs zs*
⟨*proof*⟩

theorem *prefixeq-Cons*: *prefixeq xs* (*y* # *ys*) = (*xs* = [] ∨ (∃ *zs*. *xs* = *y* # *zs* ∧ *prefixeq zs ys*))
⟨*proof*⟩

theorem *prefixeq-append*:
prefixeq xs (*ys* @ *zs*) = (*prefixeq xs ys* ∨ (∃ *us*. *xs* = *ys* @ *us* ∧ *prefixeq us zs*))
⟨*proof*⟩

lemma *append-one-prefixeq*:
prefixeq xs ys ⟹ *length xs* < *length ys* ⟹ *prefixeq* (*xs* @ [*ys* ! *length xs*]) *ys*
⟨*proof*⟩

theorem *prefixeq-length-le*: *prefixeq xs ys* ⟹ *length xs* ≤ *length ys*
⟨*proof*⟩

lemma *prefixeq-same-cases*:
prefixeq (xs_1::'*a list*) *ys* ⟹ *prefixeq* xs_2 *ys* ⟹ *prefixeq* xs_1 xs_2 ∨ *prefixeq* xs_2 xs_1
⟨*proof*⟩

lemma *set-mono-prefixeq*: *prefixeq xs ys* ⟹ *set xs* ⊆ *set ys*
⟨*proof*⟩

lemma *take-is-prefixeq*: *prefixeq* (*take n xs*) *xs*
⟨*proof*⟩

lemma *map-prefixeqI*: *prefixeq xs ys* ⟹ *prefixeq* (*map f xs*) (*map f ys*)
⟨*proof*⟩

lemma *prefixeq-length-less*: *prefix xs ys* ⟹ *length xs* < *length ys*
⟨*proof*⟩

lemma *prefix-simps* [*simp*, *code*]:
prefix xs [] ⟷ *False*
prefix [] (*x* # *xs*) ⟷ *True*
prefix (*x* # *xs*) (*y* # *ys*) ⟷ *x* = *y* ∧ *prefix xs ys*
⟨*proof*⟩

lemma *take-prefix*: *prefix xs ys* ⟹ *prefix* (*take n xs*) *ys*
⟨*proof*⟩

lemma *not-prefixeq-cases*:
assumes *pfx*: ¬ *prefixeq ps ls*
obtains
(*c1*) *ps* ≠ [] **and** *ls* = []
| (*c2*) *a as x xs* **where** *ps* = *a#as* **and** *ls* = *x#xs* **and** *x* = *a* **and** ¬ *prefixeq as xs*
| (*c3*) *a as x xs* **where** *ps* = *a#as* **and** *ls* = *x#xs* **and** *x* ≠ *a*
⟨*proof*⟩

lemma *not-prefixeq-induct* [*consumes 1*, *case-names Nil Neq Eq*]:
assumes *np*: ¬ *prefixeq ps ls*
and *base*: ⋀*x xs*. *P* (*x#xs*) []
and *r1*: ⋀*x xs y ys*. *x* ≠ *y* ⟹ *P* (*x#xs*) (*y#ys*)
and *r2*: ⋀*x xs y ys*. ⟦ *x* = *y*; ¬ *prefixeq xs ys*; *P xs ys* ⟧ ⟹ *P* (*x#xs*) (*y#ys*)
shows *P ps ls* ⟨*proof*⟩

27.3 Parallel lists

definition *parallel* :: $'a$ *list* ⇒ $'a$ *list* ⇒ *bool* (**infixl** ‖ *50*)
  where (*xs* ‖ *ys*) = (¬ *prefixeq xs ys* ∧ ¬ *prefixeq ys xs*)

lemma *parallelI* [*intro*]: ¬ *prefixeq xs ys* ⟹ ¬ *prefixeq ys xs* ⟹ *xs* ‖ *ys*
  ⟨*proof*⟩

lemma *parallelE* [*elim*]:
  assumes *xs* ‖ *ys*
  obtains ¬ *prefixeq xs ys* ∧ ¬ *prefixeq ys xs*
  ⟨*proof*⟩

theorem *prefixeq-cases*:
  obtains *prefixeq xs ys* | *prefix ys xs* | *xs* ‖ *ys*
  ⟨*proof*⟩

theorem *parallel-decomp*:
  xs ‖ *ys* ⟹ ∃ *as b bs c cs*. *b* ≠ *c* ∧ *xs* = *as* @ *b* # *bs* ∧ *ys* = *as* @ *c* # *cs*
⟨*proof*⟩

lemma *parallel-append*: *a* ‖ *b* ⟹ *a* @ *c* ‖ *b* @ *d*
  ⟨*proof*⟩

lemma *parallel-appendI*: *xs* ‖ *ys* ⟹ *x* = *xs* @ *xs'* ⟹ *y* = *ys* @ *ys'* ⟹ *x* ‖ *y*
  ⟨*proof*⟩

lemma *parallel-commute*: *a* ‖ *b* ⟷ *b* ‖ *a*
  ⟨*proof*⟩

27.4 Suffix order on lists

definition *suffixeq* :: $'a$ *list* ⇒ $'a$ *list* ⇒ *bool*
  where *suffixeq xs ys* = (∃ *zs*. *ys* = *zs* @ *xs*)

definition *suffix* :: $'a$ *list* ⇒ $'a$ *list* ⇒ *bool*
  where *suffix xs ys* ⟷ (∃ *us*. *ys* = *us* @ *xs* ∧ *us* ≠ [])

lemma *suffix-imp-suffixeq*:
  suffix xs ys ⟹ *suffixeq xs ys*
  ⟨*proof*⟩

lemma *suffixeqI* [*intro?*]: *ys* = *zs* @ *xs* ⟹ *suffixeq xs ys*
  ⟨*proof*⟩

lemma *suffixeqE* [*elim?*]:
  assumes *suffixeq xs ys*
  obtains *zs* **where** *ys* = *zs* @ *xs*
  ⟨*proof*⟩

lemma *suffixeq-refl* [*iff*]: *suffixeq xs xs*
⟨*proof*⟩
lemma *suffix-trans*:
suffix xs ys ⟹ *suffix ys zs* ⟹ *suffix xs zs*
⟨*proof*⟩
lemma *suffixeq-trans*: ⟦*suffixeq xs ys*; *suffixeq ys zs*⟧ ⟹ *suffixeq xs zs*
⟨*proof*⟩
lemma *suffixeq-antisym*: ⟦*suffixeq xs ys*; *suffixeq ys xs*⟧ ⟹ *xs* = *ys*
⟨*proof*⟩

lemma *suffixeq-tl* [*simp*]: *suffixeq* (*tl xs*) *xs*
⟨*proof*⟩

lemma *suffix-tl* [*simp*]: *xs* ≠ [] ⟹ *suffix* (*tl xs*) *xs*
⟨*proof*⟩

lemma *Nil-suffixeq* [*iff*]: *suffixeq* [] *xs*
⟨*proof*⟩
lemma *suffixeq-Nil* [*simp*]: (*suffixeq xs* []) = (*xs* = [])
⟨*proof*⟩

lemma *suffixeq-ConsI*: *suffixeq xs ys* ⟹ *suffixeq xs* (*y* # *ys*)
⟨*proof*⟩
lemma *suffixeq-ConsD*: *suffixeq* (*x* # *xs*) *ys* ⟹ *suffixeq xs ys*
⟨*proof*⟩

lemma *suffixeq-appendI*: *suffixeq xs ys* ⟹ *suffixeq xs* (*zs* @ *ys*)
⟨*proof*⟩
lemma *suffixeq-appendD*: *suffixeq* (*zs* @ *xs*) *ys* ⟹ *suffixeq xs ys*
⟨*proof*⟩

lemma *suffix-set-subset*:
suffix xs ys ⟹ *set xs* ⊆ *set ys* ⟨*proof*⟩

lemma *suffixeq-set-subset*:
suffixeq xs ys ⟹ *set xs* ⊆ *set ys* ⟨*proof*⟩

lemma *suffixeq-ConsD2*: *suffixeq* (*x* # *xs*) (*y* # *ys*) ⟹ *suffixeq xs ys*
⟨*proof*⟩

lemma *suffixeq-to-prefixeq* [*code*]: *suffixeq xs ys* ⟷ *prefixeq* (*rev xs*) (*rev ys*)
⟨*proof*⟩

lemma *distinct-suffixeq*: *distinct ys* ⟹ *suffixeq xs ys* ⟹ *distinct xs*
⟨*proof*⟩

lemma *suffixeq-map*: *suffixeq xs ys* ⟹ *suffixeq* (*map f xs*) (*map f ys*)
⟨*proof*⟩

lemma *suffixeq-drop*: *suffixeq* (*drop n as*) *as*
⟨*proof*⟩

lemma *suffixeq-take*: *suffixeq xs ys* ⟹ *ys* = *take* (*length ys* − *length xs*) *ys* @ *xs*
⟨*proof*⟩

lemma *suffixeq-suffix-reflclp-conv*: *suffixeq* = $suffix^{==}$
⟨*proof*⟩

lemma *parallelD1*: *x* ∥ *y* ⟹ ¬ *prefixeq x y*
⟨*proof*⟩

lemma *parallelD2*: *x* ∥ *y* ⟹ ¬ *prefixeq y x*
⟨*proof*⟩

lemma *parallel-Nil1* [*simp*]: ¬ *x* ∥ []
⟨*proof*⟩

lemma *parallel-Nil2* [*simp*]: ¬ [] ∥ *x*
⟨*proof*⟩

lemma *Cons-parallelI1*: *a* ≠ *b* ⟹ *a* # *as* ∥ *b* # *bs*
⟨*proof*⟩

lemma *Cons-parallelI2*: ⟦ *a* = *b*; *as* ∥ *bs* ⟧ ⟹ *a* # *as* ∥ *b* # *bs*
⟨*proof*⟩

lemma *not-equal-is-parallel*:
assumes *neq*: *xs* ≠ *ys*
and *len*: *length xs* = *length ys*
shows *xs* ∥ *ys*
⟨*proof*⟩

lemma *suffix-reflclp-conv*: $suffix^{==}$ = *suffixeq*
⟨*proof*⟩

lemma *suffix-lists*: *suffix xs ys* ⟹ *ys* ∈ *lists A* ⟹ *xs* ∈ *lists A*
⟨*proof*⟩

27.5 Homeomorphic embedding on lists

inductive *list-emb* :: ('*a* ⇒ '*a* ⇒ *bool*) ⇒ '*a list* ⇒ '*a list* ⇒ *bool*
for *P* :: ('*a* ⇒ '*a* ⇒ *bool*)
where
list-emb-Nil [*intro*, *simp*]: *list-emb P* [] *ys*
| *list-emb-Cons* [*intro*] : *list-emb P xs ys* ⟹ *list-emb P xs* (*y*#*ys*)
| *list-emb-Cons2* [*intro*]: *P x y* ⟹ *list-emb P xs ys* ⟹ *list-emb P* (*x*#*xs*) (*y*#*ys*)

lemma *list-emb-mono*:

assumes $\bigwedge x\ y.\ P\ x\ y \longrightarrow Q\ x\ y$
shows *list-emb P xs ys* $\longrightarrow$ *list-emb Q xs ys*
⟨*proof*⟩

lemma *list-emb-Nil2* [*simp*]:
assumes *list-emb P xs* [] **shows** *xs* = []
⟨*proof*⟩

lemma *list-emb-refl*:
assumes $\bigwedge x.\ x \in set\ xs \Longrightarrow P\ x\ x$
shows *list-emb P xs xs*
⟨*proof*⟩

lemma *list-emb-Cons-Nil* [*simp*]: *list-emb P* (*x#xs*) [] = *False*
⟨*proof*⟩

lemma *list-emb-append2* [*intro*]: *list-emb P xs ys* $\Longrightarrow$ *list-emb P xs* (*zs* @ *ys*)
⟨*proof*⟩

lemma *list-emb-prefix* [*intro*]:
assumes *list-emb P xs ys* **shows** *list-emb P xs* (*ys* @ *zs*)
⟨*proof*⟩

lemma *list-emb-ConsD*:
assumes *list-emb P* (*x#xs*) *ys*
shows $\exists$ *us v vs. ys* = *us* @ *v* # *vs* $\wedge$ *P x v* $\wedge$ *list-emb P xs vs*
⟨*proof*⟩

lemma *list-emb-appendD*:
assumes *list-emb P* (*xs* @ *ys*) *zs*
shows $\exists$ *us vs. zs* = *us* @ *vs* $\wedge$ *list-emb P xs us* $\wedge$ *list-emb P ys vs*
⟨*proof*⟩

lemma *list-emb-suffix*:
assumes *list-emb P xs ys* **and** *suffix ys zs*
shows *list-emb P xs zs*
⟨*proof*⟩

lemma *list-emb-suffixeq*:
assumes *list-emb P xs ys* **and** *suffixeq ys zs*
shows *list-emb P xs zs*
⟨*proof*⟩

lemma *list-emb-length*: *list-emb P xs ys* $\Longrightarrow$ *length xs* $\leq$ *length ys*
⟨*proof*⟩

lemma *list-emb-trans*:
assumes $\bigwedge x\ y\ z.\ [\![x \in set\ xs;\ y \in set\ ys;\ z \in set\ zs;\ P\ x\ y;\ P\ y\ z]\!] \Longrightarrow P\ x\ z$
shows ⟦*list-emb P xs ys*; *list-emb P ys zs*⟧ $\Longrightarrow$ *list-emb P xs zs*

⟨*proof*⟩

lemma *list-emb-set*:
assumes *list-emb P xs ys* **and** $x \in set\ xs$
obtains *y* **where** $y \in set\ ys$ **and** *P x y*
⟨*proof*⟩

27.6 Sublists (special case of homeomorphic embedding)

abbreviation *sublisteq* :: *'a list* ⇒ *'a list* ⇒ *bool*
where *sublisteq xs ys* ≡ *list-emb* (*op* =) *xs ys*

lemma *sublisteq-Cons2*: *sublisteq xs ys* ⟹ *sublisteq* (*x#xs*) (*x#ys*) ⟨*proof*⟩

lemma *sublisteq-same-length*:
assumes *sublisteq xs ys* **and** *length xs* = *length ys* **shows** *xs* = *ys*
⟨*proof*⟩

lemma *not-sublisteq-length* [*simp*]: *length ys* < *length xs* ⟹ ¬ *sublisteq xs ys*
⟨*proof*⟩

lemma [*code*]:
list-emb P [] *ys* ⟷ *True*
list-emb P (*x#xs*) [] ⟷ *False*
⟨*proof*⟩

lemma *sublisteq-Cons'*: *sublisteq* (*x#xs*) *ys* ⟹ *sublisteq xs ys*
⟨*proof*⟩

lemma *sublisteq-Cons2'*:
assumes *sublisteq* (*x#xs*) (*x#ys*) **shows** *sublisteq xs ys*
⟨*proof*⟩

lemma *sublisteq-Cons2-neq*:
assumes *sublisteq* (*x#xs*) (*y#ys*)
shows $x \neq y$ ⟹ *sublisteq* (*x#xs*) *ys*
⟨*proof*⟩

lemma *sublisteq-Cons2-iff* [*simp*, *code*]:
sublisteq (*x#xs*) (*y#ys*) = (*if x* = *y then sublisteq xs ys else sublisteq* (*x#xs*) *ys*)
⟨*proof*⟩

lemma *sublisteq-append'*: *sublisteq* (*zs* @ *xs*) (*zs* @ *ys*) ⟷ *sublisteq xs ys*
⟨*proof*⟩

lemma *sublisteq-refl* [*simp*, *intro*!]: *sublisteq xs xs* ⟨*proof*⟩

lemma *sublisteq-antisym*:
assumes *sublisteq xs ys* **and** *sublisteq ys xs*

shows *xs* = *ys*
⟨*proof*⟩

lemma *sublisteq-trans*: *sublisteq xs ys* $\Longrightarrow$ *sublisteq ys zs* $\Longrightarrow$ *sublisteq xs zs*
⟨*proof*⟩

lemma *sublisteq-append-le-same-iff*: *sublisteq* (*xs* @ *ys*) *ys* $\longleftrightarrow$ *xs* = []
⟨*proof*⟩

lemma *list-emb-append-mono*:
⟦ *list-emb P xs xs'*; *list-emb P ys ys'* ⟧ $\Longrightarrow$ *list-emb P* (*xs*@*ys*) (*xs'*@*ys'*)
⟨*proof*⟩

27.7 Appending elements

lemma *sublisteq-append* [*simp*]:
sublisteq (*xs* @ *zs*) (*ys* @ *zs*) $\longleftrightarrow$ *sublisteq xs ys* (**is** *?l* = *?r*)
⟨*proof*⟩

lemma *sublisteq-drop-many*: *sublisteq xs ys* $\Longrightarrow$ *sublisteq xs* (*zs* @ *ys*)
⟨*proof*⟩

lemma *sublisteq-rev-drop-many*: *sublisteq xs ys* $\Longrightarrow$ *sublisteq xs* (*ys* @ *zs*)
⟨*proof*⟩

27.8 Relation to standard list operations

lemma *sublisteq-map*:
assumes *sublisteq xs ys* **shows** *sublisteq* (*map f xs*) (*map f ys*)
⟨*proof*⟩

lemma *sublisteq-filter-left* [*simp*]: *sublisteq* (*filter P xs*) *xs*
⟨*proof*⟩

lemma *sublisteq-filter* [*simp*]:
assumes *sublisteq xs ys* **shows** *sublisteq* (*filter P xs*) (*filter P ys*)
⟨*proof*⟩

lemma *sublisteq xs ys* $\longleftrightarrow$ ($\exists N$. *xs* = *sublist ys N*) (**is** *?L* = *?R*)
⟨*proof*⟩

end

28 Linear Temporal Logic on Streams

theory *Linear-Temporal-Logic-on-Streams*
imports *Stream Sublist Extended-Nat Infinite-Set*
begin

29 Preliminaries

lemma *shift-prefix*:
assumes *xl* @− *xs* = *yl* @− *ys* **and** *length xl* ≤ *length yl*
shows *prefixeq xl yl*
⟨*proof*⟩

lemma *shift-prefix-cases*:
assumes *xl* @− *xs* = *yl* @− *ys*
shows *prefixeq xl yl* ∨ *prefixeq yl xl*
⟨*proof*⟩

30 Linear temporal logic

abbreviation (*input*) *IMPL* (**infix** *impl 60*)
where φ *impl* ψ ≡ λ *xs*. φ *xs* ⟶ ψ *xs*

abbreviation (*input*) *OR* (**infix** *or 60*)
where φ *or* ψ ≡ λ *xs*. φ *xs* ∨ ψ *xs*

abbreviation (*input*) *AND* (**infix** *aand 60*)
where φ *aand* ψ ≡ λ *xs*. φ *xs* ∧ ψ *xs*

abbreviation (*input*) *not* φ ≡ λ *xs*. ¬ φ *xs*

abbreviation (*input*) *true* ≡ λ *xs*. *True*

abbreviation (*input*) *false* ≡ λ *xs*. *False*

lemma *impl-not-or*: φ *impl* ψ = (*not* φ) *or* ψ
⟨*proof*⟩

lemma *not-or*: *not* (φ *or* ψ) = (*not* φ) *aand* (*not* ψ)
⟨*proof*⟩

lemma *not-aand*: *not* (φ *aand* ψ) = (*not* φ) *or* (*not* ψ)
⟨*proof*⟩

lemma *non-not*[*simp*]: *not* (*not* φ) = φ ⟨*proof*⟩

fun *holds* **where** *holds P xs* ⟷ *P* (*shd xs*)
fun *nxt* **where** *nxt* φ *xs* = φ (*stl xs*)

definition *HLD s* = *holds* (λ*x*. *x* ∈ *s*)

abbreviation *HLD-nxt* (**infixr** · *65*) **where**
s · *P* ≡ *HLD s aand nxt P*

context
  notes [[*inductive-internals*]]
begin

inductive *ev* **for** φ **where**
base: $\varphi\ xs \Longrightarrow ev\ \varphi\ xs$
|
step: $ev\ \varphi\ (stl\ xs) \Longrightarrow ev\ \varphi\ xs$

coinductive *alw* **for** φ **where**
alw: $[\![\varphi\ xs;\ alw\ \varphi\ (stl\ xs)]\!] \Longrightarrow alw\ \varphi\ xs$

coinductive *UNTIL* (**infix** *until 60*) **for** $\varphi\ \psi$ **where**
base: $\psi\ xs \Longrightarrow (\varphi\ until\ \psi)\ xs$
|
step: $[\![\varphi\ xs;\ (\varphi\ until\ \psi)\ (stl\ xs)]\!] \Longrightarrow (\varphi\ until\ \psi)\ xs$

end

lemma *holds-mono*:
assumes *holds*: *holds P xs* **and** *0*: $\bigwedge x.\ P\ x \Longrightarrow Q\ x$
shows *holds Q xs*
⟨*proof*⟩

lemma *holds-aand*:
$(holds\ P\ aand\ holds\ Q)\ steps \longleftrightarrow holds\ (\lambda\ step.\ P\ step \wedge Q\ step)\ steps$ ⟨*proof*⟩

lemma *HLD-iff*: $HLD\ s\ \omega \longleftrightarrow shd\ \omega \in s$
  ⟨*proof*⟩

lemma *HLD-Stream*[*simp*]: $HLD\ X\ (x\ \#\#\ \omega) \longleftrightarrow x \in X$
  ⟨*proof*⟩

lemma *nxt-mono*:
assumes *nxt*: *nxt* φ *xs* **and** *0*: $\bigwedge xs.\ \varphi\ xs \Longrightarrow \psi\ xs$
shows *nxt* ψ *xs*
⟨*proof*⟩

declare *ev.intros*[*intro*]
declare *alw.cases*[*elim*]

lemma *ev-induct-strong*[*consumes 1*, *case-names base step*]:
  $ev\ \varphi\ x \Longrightarrow (\bigwedge xs.\ \varphi\ xs \Longrightarrow P\ xs) \Longrightarrow (\bigwedge xs.\ ev\ \varphi\ (stl\ xs) \Longrightarrow \neg\ \varphi\ xs \Longrightarrow P\ (stl\ xs) \Longrightarrow P\ xs) \Longrightarrow P\ x$
  ⟨*proof*⟩

lemma *alw-coinduct*[*consumes 1*, *case-names alw stl*]:
  $X\ x \Longrightarrow (\bigwedge x.\ X\ x \Longrightarrow \varphi\ x) \Longrightarrow (\bigwedge x.\ X\ x \Longrightarrow \neg\ alw\ \varphi\ (stl\ x) \Longrightarrow X\ (stl\ x))$

$\Longrightarrow$ *alw* φ *x*
⟨*proof*⟩

lemma *ev-mono*:
assumes *ev*: *ev* φ *xs* **and** *0*: $\bigwedge$ *xs*. φ *xs* $\Longrightarrow$ ψ *xs*
shows *ev* ψ *xs*
⟨*proof*⟩

lemma *alw-mono*:
assumes *alw*: *alw* φ *xs* **and** *0*: $\bigwedge$ *xs*. φ *xs* $\Longrightarrow$ ψ *xs*
shows *alw* ψ *xs*
⟨*proof*⟩

lemma *until-monoL*:
assumes *until*: (φ*1* *until* ψ) *xs* **and** *0*: $\bigwedge$ *xs*. φ*1* *xs* $\Longrightarrow$ φ*2* *xs*
shows (φ*2* *until* ψ) *xs*
⟨*proof*⟩

lemma *until-monoR*:
assumes *until*: (φ *until* ψ*1*) *xs* **and** *0*: $\bigwedge$ *xs*. ψ*1* *xs* $\Longrightarrow$ ψ*2* *xs*
shows (φ *until* ψ*2*) *xs*
⟨*proof*⟩

lemma *until-mono*:
assumes *until*: (φ*1* *until* ψ*1*) *xs* **and**
0: $\bigwedge$ *xs*. φ*1* *xs* $\Longrightarrow$ φ*2* *xs* $\bigwedge$ *xs*. ψ*1* *xs* $\Longrightarrow$ ψ*2* *xs*
shows (φ*2* *until* ψ*2*) *xs*
⟨*proof*⟩

lemma *until-false*: φ *until false* = *alw* φ
⟨*proof*⟩

lemma *ev-nxt*: *ev* φ = (φ *or nxt* (*ev* φ))
⟨*proof*⟩

lemma *alw-nxt*: *alw* φ = (φ *aand nxt* (*alw* φ))
⟨*proof*⟩

lemma *ev-ev*[*simp*]: *ev* (*ev* φ) = *ev* φ
⟨*proof*⟩

lemma *alw-alw*[*simp*]: *alw* (*alw* φ) = *alw* φ
⟨*proof*⟩

lemma *ev-shift*:
assumes *ev* φ *xs*
shows *ev* φ (*xl* @− *xs*)
⟨*proof*⟩

lemma *ev-imp-shift*:
assumes $ev\ \varphi\ xs$ **shows** $\exists\ xl\ xs2.\ xs = xl\ @-\ xs2 \wedge \varphi\ xs2$
⟨*proof*⟩

lemma *alw-ev-shift*: $alw\ \varphi\ xs1 \Longrightarrow ev\ (alw\ \varphi)\ (xl\ @-\ xs1)$
⟨*proof*⟩

lemma *alw-shift*:
assumes $alw\ \varphi\ (xl\ @-\ xs)$
shows $alw\ \varphi\ xs$
⟨*proof*⟩

lemma *ev-ex-nxt*:
assumes $ev\ \varphi\ xs$
shows $\exists\ n.\ (nxt\ \hat{}\hat{}\ n)\ \varphi\ xs$
⟨*proof*⟩

lemma *alw-sdrop*:
assumes $alw\ \varphi\ xs$ **shows** $alw\ \varphi\ (sdrop\ n\ xs)$
⟨*proof*⟩

lemma *nxt-sdrop*: $(nxt\ \hat{}\hat{}\ n)\ \varphi\ xs \longleftrightarrow \varphi\ (sdrop\ n\ xs)$
⟨*proof*⟩

definition $wait\ \varphi\ xs \equiv LEAST\ n.\ (nxt\ \hat{}\hat{}\ n)\ \varphi\ xs$

lemma *nxt-wait*:
assumes $ev\ \varphi\ xs$ **shows** $(nxt\ \hat{}\hat{}\ (wait\ \varphi\ xs))\ \varphi\ xs$
⟨*proof*⟩

lemma *nxt-wait-least*:
assumes *ev*: $ev\ \varphi\ xs$ **and** *nxt*: $(nxt\ \hat{}\hat{}\ n)\ \varphi\ xs$ **shows** $wait\ \varphi\ xs \le n$
⟨*proof*⟩

lemma *sdrop-wait*:
assumes $ev\ \varphi\ xs$ **shows** $\varphi\ (sdrop\ (wait\ \varphi\ xs)\ xs)$
⟨*proof*⟩

lemma *sdrop-wait-least*:
assumes *ev*: $ev\ \varphi\ xs$ **and** *nxt*: $\varphi\ (sdrop\ n\ xs)$ **shows** $wait\ \varphi\ xs \le n$
⟨*proof*⟩

lemma *nxt-ev*: $(nxt\ \hat{}\hat{}\ n)\ \varphi\ xs \Longrightarrow ev\ \varphi\ xs$
⟨*proof*⟩

lemma *not-ev*: $not\ (ev\ \varphi) = alw\ (not\ \varphi)$
⟨*proof*⟩

lemma *not-alw*: $not\ (alw\ \varphi) = ev\ (not\ \varphi)$

⟨*proof*⟩

lemma *not-ev-not*[*simp*]: *not* (*ev* (*not* φ)) = *alw* φ
⟨*proof*⟩

lemma *not-alw-not*[*simp*]: *not* (*alw* (*not* φ)) = *ev* φ
⟨*proof*⟩

lemma *alw-ev-sdrop*:
assumes *alw* (*ev* φ) (*sdrop m xs*)
shows *alw* (*ev* φ) *xs*
⟨*proof*⟩

lemma *ev-alw-imp-alw-ev*:
assumes *ev* (*alw* φ) *xs* **shows** *alw* (*ev* φ) *xs*
⟨*proof*⟩

lemma *alw-aand*: *alw* (φ *aand* ψ) = *alw* φ *aand alw* ψ
⟨*proof*⟩

lemma *ev-or*: *ev* (φ *or* ψ) = *ev* φ *or ev* ψ
⟨*proof*⟩

lemma *ev-alw-aand*:
assumes φ: *ev* (*alw* φ) *xs* **and** ψ: *ev* (*alw* ψ) *xs*
shows *ev* (*alw* (φ *aand* ψ)) *xs*
⟨*proof*⟩

lemma *ev-alw-alw-impl*:
assumes *ev* (*alw* φ) *xs* **and** *alw* (*alw* φ *impl ev* ψ) *xs*
shows *ev* ψ *xs*
⟨*proof*⟩

lemma *ev-alw-stl*[*simp*]: *ev* (*alw* φ) (*stl x*) ⟷ *ev* (*alw* φ) *x*
⟨*proof*⟩

lemma *alw-alw-impl-ev*:
alw (*alw* φ *impl ev* ψ) = (*ev* (*alw* φ) *impl alw* (*ev* ψ)) (**is** *?A* = *?B*)
⟨*proof*⟩

lemma *ev-alw-impl*:
assumes *ev* φ *xs* **and** *alw* (φ *impl* ψ) *xs* **shows** *ev* ψ *xs*
⟨*proof*⟩

lemma *ev-alw-impl-ev*:
assumes *ev* φ *xs* **and** *alw* (φ *impl ev* ψ) *xs* **shows** *ev* ψ *xs*
⟨*proof*⟩

lemma *alw-mp*:

assumes *alw* φ *xs* **and** *alw* (φ *impl* ψ) *xs*
shows *alw* ψ *xs*
⟨*proof*⟩

lemma *all-imp-alw*:
assumes $\bigwedge$ *xs*. φ *xs* **shows** *alw* φ *xs*
⟨*proof*⟩

lemma *alw-impl-ev-alw*:
assumes *alw* (φ *impl ev* ψ) *xs*
shows *alw* (*ev* φ *impl ev* ψ) *xs*
⟨*proof*⟩

lemma *ev-holds-sset*:
ev (*holds P*) *xs* $\longleftrightarrow$ ($\exists$ *x* $\in$ *sset xs*. *P x*) (**is** *?L* $\longleftrightarrow$ *?R*)
⟨*proof*⟩

lemma *alw-invar*:
assumes φ *xs* **and** *alw* (φ *impl nxt* φ) *xs*
shows *alw* φ *xs*
⟨*proof*⟩

lemma *variance*:
assumes *1*: φ *xs* **and** *2*: *alw* (φ *impl* (ψ *or nxt* φ)) *xs*
shows (*alw* φ *or ev* ψ) *xs*
⟨*proof*⟩

lemma *ev-alw-imp-nxt*:
assumes *e*: *ev* φ *xs* **and** *a*: *alw* (φ *impl* (*nxt* φ)) *xs*
shows *ev* (*alw* φ) *xs*
⟨*proof*⟩

inductive *ev-at* :: ('*a stream* $\Rightarrow$ *bool*) $\Rightarrow$ *nat* $\Rightarrow$ '*a stream* $\Rightarrow$ *bool* **for** *P* :: '*a stream* $\Rightarrow$ *bool* **where**
base: *P* ω $\Longrightarrow$ *ev-at P 0* ω
| *step*:$\neg$ *P* ω $\Longrightarrow$ *ev-at P n* (*stl* ω) $\Longrightarrow$ *ev-at P* (*Suc n*) ω

inductive-simps *ev-at-0*[*simp*]: *ev-at P 0* ω
inductive-simps *ev-at-Suc*[*simp*]: *ev-at P* (*Suc n*) ω

lemma *ev-at-imp-snth*: *ev-at P n* ω $\Longrightarrow$ *P* (*sdrop n* ω)
⟨*proof*⟩

lemma *ev-at-HLD-imp-snth*: *ev-at* (*HLD X*) *n* ω $\Longrightarrow$ ω !! *n* $\in$ *X*
⟨*proof*⟩

lemma *ev-at-HLD-single-imp-snth*: *ev-at* (*HLD* {*x*}) *n* ω $\Longrightarrow$ ω !! *n* = *x*

⟨*proof*⟩

lemma *ev-at-unique*: $ev\text{-}at\ P\ n\ \omega \Longrightarrow ev\text{-}at\ P\ m\ \omega \Longrightarrow n = m$
⟨*proof*⟩

lemma *ev-iff-ev-at*: $ev\ P\ \omega \longleftrightarrow (\exists\, n.\ ev\text{-}at\ P\ n\ \omega)$
⟨*proof*⟩

lemma *ev-at-shift*: $ev\text{-}at\ (HLD\ X)\ i\ (stake\ (Suc\ i)\ \omega\ @-\ \omega' :: 's\ stream) \longleftrightarrow ev\text{-}at\ (HLD\ X)\ i\ \omega$
⟨*proof*⟩

lemma *ev-iff-ev-at-unqiue*: $ev\ P\ \omega \longleftrightarrow (\exists !n.\ ev\text{-}at\ P\ n\ \omega)$
⟨*proof*⟩

lemma *alw-HLD-iff-streams*: $alw\ (HLD\ X)\ \omega \longleftrightarrow \omega \in streams\ X$
⟨*proof*⟩

lemma *not-HLD*: $not\ (HLD\ X) = HLD\ (-\ X)$
⟨*proof*⟩

lemma *not-alw-iff*: $\neg\ (alw\ P\ \omega) \longleftrightarrow ev\ (not\ P)\ \omega$
⟨*proof*⟩

lemma *not-ev-iff*: $\neg\ (ev\ P\ \omega) \longleftrightarrow alw\ (not\ P)\ \omega$
⟨*proof*⟩

lemma *ev-Stream*: $ev\ P\ (x\ \#\#\ s) \longleftrightarrow P\ (x\ \#\#\ s) \lor ev\ P\ s$
⟨*proof*⟩

lemma *alw-ev-imp-ev-alw*:
assumes $alw\ (ev\ P)\ \omega$ **shows** $ev\ (P\ aand\ alw\ (ev\ P))\ \omega$
⟨*proof*⟩

lemma *ev-False*: $ev\ (\lambda x.\ False)\ \omega \longleftrightarrow False$
⟨*proof*⟩

lemma *alw-False*: $alw\ (\lambda x.\ False)\ \omega \longleftrightarrow False$
⟨*proof*⟩

lemma *ev-iff-sdrop*: $ev\ P\ \omega \longleftrightarrow (\exists\, m.\ P\ (sdrop\ m\ \omega))$
⟨*proof*⟩

lemma *alw-iff-sdrop*: $alw\ P\ \omega \longleftrightarrow (\forall\, m.\ P\ (sdrop\ m\ \omega))$
⟨*proof*⟩

lemma *infinite-iff-alw-ev*: $infinite\ \{m.\ P\ (sdrop\ m\ \omega)\} \longleftrightarrow alw\ (ev\ P)\ \omega$
⟨*proof*⟩

lemma *alw-inv*:
assumes *stl*: $\bigwedge s.\ f\ (stl\ s) = stl\ (f\ s)$
shows $alw\ P\ (f\ s) \longleftrightarrow alw\ (\lambda x.\ P\ (f\ x))\ s$
⟨*proof*⟩

lemma *ev-inv*:
assumes *stl*: $\bigwedge s.\ f\ (stl\ s) = stl\ (f\ s)$
shows $ev\ P\ (f\ s) \longleftrightarrow ev\ (\lambda x.\ P\ (f\ x))\ s$
⟨*proof*⟩

lemma *alw-smap*: $alw\ P\ (smap\ f\ s) \longleftrightarrow alw\ (\lambda x.\ P\ (smap\ f\ x))\ s$
⟨*proof*⟩

lemma *ev-smap*: $ev\ P\ (smap\ f\ s) \longleftrightarrow ev\ (\lambda x.\ P\ (smap\ f\ x))\ s$
⟨*proof*⟩

lemma *alw-cong*:
assumes *P*: $alw\ P\ \omega$ **and** *eq*: $\bigwedge\omega.\ P\ \omega \Longrightarrow Q1\ \omega \longleftrightarrow Q2\ \omega$
shows $alw\ Q1\ \omega \longleftrightarrow alw\ Q2\ \omega$
⟨*proof*⟩

lemma *ev-cong*:
assumes *P*: $alw\ P\ \omega$ **and** *eq*: $\bigwedge\omega.\ P\ \omega \Longrightarrow Q1\ \omega \longleftrightarrow Q2\ \omega$
shows $ev\ Q1\ \omega \longleftrightarrow ev\ Q2\ \omega$
⟨*proof*⟩

lemma *alwD*: $alw\ P\ x \Longrightarrow P\ x$
⟨*proof*⟩

lemma *alw-alwD*: $alw\ P\ \omega \Longrightarrow alw\ (alw\ P)\ \omega$
⟨*proof*⟩

lemma *alw-ev-stl*: $alw\ (ev\ P)\ (stl\ \omega) \longleftrightarrow alw\ (ev\ P)\ \omega$
⟨*proof*⟩

lemma *holds-Stream*: $holds\ P\ (x\ \#\#\ s) \longleftrightarrow P\ x$
⟨*proof*⟩

lemma *holds-eq1*[*simp*]: $holds\ (op = x) = HLD\ \{x\}$
⟨*proof*⟩

lemma *holds-eq2*[*simp*]: $holds\ (\lambda y.\ y = x) = HLD\ \{x\}$
⟨*proof*⟩

lemma *not-holds-eq*[*simp*]: $holds\ (-\ op = x) = not\ (HLD\ \{x\})$
⟨*proof*⟩

Strong until

context

notes [[*inductive-internals*]]
begin

inductive *suntil* (**infix** *suntil 60*) **for** φ ψ **where**
base: $\psi\ \omega \Longrightarrow (\varphi\ suntil\ \psi)\ \omega$
| *step*: $\varphi\ \omega \Longrightarrow (\varphi\ suntil\ \psi)\ (stl\ \omega) \Longrightarrow (\varphi\ suntil\ \psi)\ \omega$

inductive-simps *suntil-Stream*: $(\varphi\ suntil\ \psi)\ (x\ \#\#\ s)$

end

lemma *suntil-induct-strong*[*consumes 1*, *case-names base step*]:
$(\varphi\ suntil\ \psi)\ x \Longrightarrow$
$(\bigwedge\omega.\ \psi\ \omega \Longrightarrow P\ \omega) \Longrightarrow$
$(\bigwedge\omega.\ \varphi\ \omega \Longrightarrow \neg\ \psi\ \omega \Longrightarrow (\varphi\ suntil\ \psi)\ (stl\ \omega) \Longrightarrow P\ (stl\ \omega) \Longrightarrow P\ \omega) \Longrightarrow P\ x$
⟨*proof*⟩

lemma *ev-suntil*: $(\varphi\ suntil\ \psi)\ \omega \Longrightarrow ev\ \psi\ \omega$
⟨*proof*⟩

lemma *suntil-inv*:
assumes *stl*: $\bigwedge s.\ f\ (stl\ s) = stl\ (f\ s)$
shows $(P\ suntil\ Q)\ (f\ s) \longleftrightarrow ((\lambda x.\ P\ (f\ x))\ suntil\ (\lambda x.\ Q\ (f\ x)))\ s$
⟨*proof*⟩

lemma *suntil-smap*: $(P\ suntil\ Q)\ (smap\ f\ s) \longleftrightarrow ((\lambda x.\ P\ (smap\ f\ x))\ suntil\ (\lambda x.\ Q\ (smap\ f\ x)))\ s$
⟨*proof*⟩

lemma *hld-smap*: $HLD\ x\ (smap\ f\ s) = holds\ (\lambda y.\ f\ y \in x)\ s$
⟨*proof*⟩

lemma *suntil-mono*:
assumes *eq*: $\bigwedge\omega.\ P\ \omega \Longrightarrow Q1\ \omega \Longrightarrow Q2\ \omega$ $\bigwedge\omega.\ P\ \omega \Longrightarrow R1\ \omega \Longrightarrow R2\ \omega$
assumes $*$: $(Q1\ suntil\ R1)\ \omega$ $alw\ P\ \omega$ **shows** $(Q2\ suntil\ R2)\ \omega$
⟨*proof*⟩

lemma *suntil-cong*:
$alw\ P\ \omega \Longrightarrow (\bigwedge\omega.\ P\ \omega \Longrightarrow Q1\ \omega \longleftrightarrow Q2\ \omega) \Longrightarrow (\bigwedge\omega.\ P\ \omega \Longrightarrow R1\ \omega \longleftrightarrow R2\ \omega) \Longrightarrow$
$(Q1\ suntil\ R1)\ \omega \longleftrightarrow (Q2\ suntil\ R2)\ \omega$
⟨*proof*⟩

lemma *ev-suntil-iff*: $ev\ (P\ suntil\ Q)\ \omega \longleftrightarrow ev\ Q\ \omega$
⟨*proof*⟩

lemma *true-suntil*: $((\lambda\text{-}.\ True)\ suntil\ P) = ev\ P$
⟨*proof*⟩

lemma *suntil-lfp*: $(\varphi\ suntil\ \psi) = lfp\ (\lambda P\ s.\ \psi\ s \vee (\varphi\ s \wedge P\ (stl\ s)))$
⟨*proof*⟩

lemma *sfilter-P*[*simp*]: $P\ (shd\ s) \Longrightarrow sfilter\ P\ s = shd\ s\ \#\#\ sfilter\ P\ (stl\ s)$
⟨*proof*⟩

lemma *sfilter-not-P*[*simp*]: $\neg\ P\ (shd\ s) \Longrightarrow sfilter\ P\ s = sfilter\ P\ (stl\ s)$
⟨*proof*⟩

lemma *sfilter-eq*:
assumes $ev\ (holds\ P)\ s$
shows $sfilter\ P\ s = x\ \#\#\ s' \longleftrightarrow$
$P\ x \wedge (not\ (holds\ P)\ suntil\ (HLD\ \{x\}\ aand\ nxt\ (\lambda s.\ sfilter\ P\ s = s')))\ s$
⟨*proof*⟩

lemma *sfilter-streams*:
$alw\ (ev\ (holds\ P))\ \omega \Longrightarrow \omega \in streams\ A \Longrightarrow sfilter\ P\ \omega \in streams\ \{x{\in}A.\ P\ x\}$
⟨*proof*⟩

lemma *alw-sfilter*:
assumes $*$: $alw\ (ev\ (holds\ P))\ s$
shows $alw\ Q\ (sfilter\ P\ s) \longleftrightarrow alw\ (\lambda x.\ Q\ (sfilter\ P\ x))\ s$
⟨*proof*⟩

lemma *ev-sfilter*:
assumes $*$: $alw\ (ev\ (holds\ P))\ s$
shows $ev\ Q\ (sfilter\ P\ s) \longleftrightarrow ev\ (\lambda x.\ Q\ (sfilter\ P\ x))\ s$
⟨*proof*⟩

lemma *holds-sfilter*:
assumes $ev\ (holds\ Q)\ s$ **shows** $holds\ P\ (sfilter\ Q\ s) \longleftrightarrow (not\ (holds\ Q)\ suntil\ (holds\ (Q\ aand\ P)))\ s$
⟨*proof*⟩

lemma *suntil-aand-nxt*:
$(\varphi\ suntil\ (\varphi\ aand\ nxt\ \psi))\ \omega \longleftrightarrow (\varphi\ aand\ nxt\ (\varphi\ suntil\ \psi))\ \omega$
⟨*proof*⟩

lemma *alw-sconst*: $alw\ P\ (sconst\ x) \longleftrightarrow P\ (sconst\ x)$
⟨*proof*⟩

lemma *ev-sconst*: $ev\ P\ (sconst\ x) \longleftrightarrow P\ (sconst\ x)$
⟨*proof*⟩

lemma *suntil-sconst*: $(\varphi\ suntil\ \psi)\ (sconst\ x) \longleftrightarrow \psi\ (sconst\ x)$
⟨*proof*⟩

lemma *hld-smap'*: $HLD\ x\ (smap\ f\ s) = HLD\ (f\ -`\ x)\ s$
⟨*proof*⟩

end

theory *Stream-Space*
imports
Infinite-Product-Measure
~~/src/HOL/Library/Stream
~~/src/HOL/Library/Linear-Temporal-Logic-on-Streams
begin

lemma *stream-eq-Stream-iff*: $s = x\ \#\#\ t \longleftrightarrow (shd\ s = x \wedge stl\ s = t)$
⟨*proof*⟩

lemma *Stream-snth*: $(x\ \#\#\ s)\ !!\ n = (case\ n\ of\ 0 \Rightarrow x \mid Suc\ n \Rightarrow s\ !!\ n)$
⟨*proof*⟩

definition *to-stream* :: $(nat \Rightarrow 'a) \Rightarrow 'a\ stream$ **where**
to-stream X = smap X nats

lemma *to-stream-nat-case*: *to-stream* (*case-nat x X*) = *x* ## *to-stream X*
⟨*proof*⟩

lemma *to-stream-in-streams*: *to-stream X* $\in$ *streams S* $\longleftrightarrow$ ($\forall$ *n*. *X n* $\in$ *S*)
⟨*proof*⟩

definition *stream-space* :: $'a\ measure \Rightarrow 'a\ stream\ measure$ **where**
stream-space M =
distr (Π_M *i*$\in$*UNIV*. *M*) (*vimage-algebra* (*streams* (*space M*)) *snth* (Π_M *i*$\in$*UNIV*. *M*)) *to-stream*

lemma *space-stream-space*: *space* (*stream-space M*) = *streams* (*space M*)
⟨*proof*⟩

lemma *streams-stream-space*[*intro*]: *streams* (*space M*) $\in$ *sets* (*stream-space M*)
⟨*proof*⟩

lemma *stream-space-Stream*:
x ## $\omega \in$ *space* (*stream-space M*) $\longleftrightarrow$ *x* $\in$ *space M* $\wedge$ $\omega \in$ *space* (*stream-space M*)
⟨*proof*⟩

lemma *stream-space-eq-distr*: *stream-space M* = *distr* (Π_M *i*$\in$*UNIV*. *M*) (*stream-space M*) *to-stream*
⟨*proof*⟩

lemma *sets-stream-space-cong*[*measurable-cong*]:
sets M = *sets N* $\Longrightarrow$ *sets* (*stream-space M*) = *sets* (*stream-space N*)
⟨*proof*⟩

lemma *measurable-snth-PiM*: $(\lambda\omega\ n.\ \omega\ !!\ n) \in$ *measurable* (*stream-space M*) (Π_M $i{\in}UNIV.\ M$)
⟨*proof*⟩

lemma *measurable-snth*[*measurable*]: $(\lambda\omega.\ \omega\ !!\ n) \in$ *measurable* (*stream-space M*) *M*
⟨*proof*⟩

lemma *measurable-shd*[*measurable*]: *shd* $\in$ *measurable* (*stream-space M*) *M*
⟨*proof*⟩

lemma *measurable-stream-space2*:
assumes *f-snth*: $\bigwedge n.\ (\lambda x.\ f\ x\ !!\ n) \in$ *measurable N M*
shows $f \in$ *measurable N* (*stream-space M*)
⟨*proof*⟩

lemma *measurable-stream-coinduct*[*consumes 1*, *case-names shd stl*, *coinduct set*: *measurable*]:
assumes *F f*
assumes *h*: $\bigwedge f.\ F\ f \Longrightarrow (\lambda x.\ shd\ (f\ x)) \in$ *measurable N M*
assumes *t*: $\bigwedge f.\ F\ f \Longrightarrow F\ (\lambda x.\ stl\ (f\ x))$
shows $f \in$ *measurable N* (*stream-space M*)
⟨*proof*⟩

lemma *measurable-sdrop*[*measurable*]: *sdrop n* $\in$ *measurable* (*stream-space M*) (*stream-space M*)
⟨*proof*⟩

lemma *measurable-stl*[*measurable*]: $(\lambda\omega.\ stl\ \omega) \in$ *measurable* (*stream-space M*) (*stream-space M*)
⟨*proof*⟩

lemma *measurable-to-stream*[*measurable*]: *to-stream* $\in$ *measurable* (Π_M $i{\in}UNIV.\ M$) (*stream-space M*)
⟨*proof*⟩

lemma *measurable-Stream*[*measurable* (*raw*)]:
assumes *f*[*measurable*]: $f \in$ *measurable N M*
assumes *g*[*measurable*]: $g \in$ *measurable N* (*stream-space M*)
shows $(\lambda x.\ f\ x\ \#\#\ g\ x) \in$ *measurable N* (*stream-space M*)
⟨*proof*⟩

lemma *measurable-smap*[*measurable*]:
assumes *X*[*measurable*]: $X \in$ *measurable N M*
shows *smap X* $\in$ *measurable* (*stream-space N*) (*stream-space M*)
⟨*proof*⟩

lemma *measurable-stake*[*measurable*]:

stake i ∈ *measurable* (*stream-space* (*count-space UNIV*)) (*count-space* (*UNIV* :: ′*a*::*countable list set*))
⟨*proof*⟩

lemma *measurable-shift*[*measurable*]:
assumes *f*: *f* ∈ *measurable N* (*stream-space M*)
assumes [*measurable*]: *g* ∈ *measurable N* (*stream-space M*)
shows (λ*x*. *stake n* (*f x*) @− *g x*) ∈ *measurable N* (*stream-space M*)
⟨*proof*⟩

lemma *measurable-ev-at*[*measurable*]:
assumes [*measurable*]: *Measurable.pred* (*stream-space M*) *P*
shows *Measurable.pred* (*stream-space M*) (*ev-at P n*)
⟨*proof*⟩

lemma *measurable-alw*[*measurable*]:
Measurable.pred (*stream-space M*) *P* ⟹ *Measurable.pred* (*stream-space M*) (*alw P*)
⟨*proof*⟩

lemma *measurable-ev*[*measurable*]:
Measurable.pred (*stream-space M*) *P* ⟹ *Measurable.pred* (*stream-space M*) (*ev P*)
⟨*proof*⟩

lemma *measurable-until*:
assumes [*measurable*]: *Measurable.pred* (*stream-space M*) φ *Measurable.pred* (*stream-space M*) ψ
shows *Measurable.pred* (*stream-space M*) (φ *until* ψ)
⟨*proof*⟩

lemma *measurable-holds* [*measurable*]: *Measurable.pred M P* ⟹ *Measurable.pred* (*stream-space M*) (*holds P*)
⟨*proof*⟩

lemma *measurable-hld*[*measurable*]: **assumes** [*measurable*]: *t* ∈ *sets M* **shows** *Measurable.pred* (*stream-space M*) (*HLD t*)
⟨*proof*⟩

lemma *measurable-nxt*[*measurable* (*raw*)]:
Measurable.pred (*stream-space M*) *P* ⟹ *Measurable.pred* (*stream-space M*) (*nxt P*)
⟨*proof*⟩

lemma *measurable-suntil*[*measurable*]:
assumes [*measurable*]: *Measurable.pred* (*stream-space M*) *Q Measurable.pred* (*stream-space M*) *P*
shows *Measurable.pred* (*stream-space M*) (*Q suntil P*)
⟨*proof*⟩

lemma *measurable-szip*:
 $(\lambda(\omega 1, \omega 2).\ szip\ \omega 1\ \omega 2) \in measurable\ (stream\text{-}space\ M \bigotimes_M stream\text{-}space\ N)\ (stream\text{-}space\ (M \bigotimes_M N))$
⟨*proof*⟩

lemma (**in** *prob-space*) *prob-space-stream-space*: *prob-space* (*stream-space M*)
⟨*proof*⟩

lemma (**in** *prob-space*) *nn-integral-stream-space*:
 assumes [*measurable*]: $f \in borel\text{-}measurable\ (stream\text{-}space\ M)$
 shows $(\int^+ X.\ f\ X\ \partial stream\text{-}space\ M) = (\int^+ x.\ (\int^+ X.\ f\ (x\ \#\#\ X)\ \partial stream\text{-}space\ M)\ \partial M)$
⟨*proof*⟩

lemma (**in** *prob-space*) *emeasure-stream-space*:
 assumes *X*[*measurable*]: $X \in sets\ (stream\text{-}space\ M)$
 shows $emeasure\ (stream\text{-}space\ M)\ X = (\int^+ t.\ emeasure\ (stream\text{-}space\ M)\ \{x \in space\ (stream\text{-}space\ M).\ t\ \#\#\ x \in X\ \}\ \partial M)$
⟨*proof*⟩

lemma (**in** *prob-space*) *prob-stream-space*:
 assumes *P*[*measurable*]: $\{x \in space\ (stream\text{-}space\ M).\ P\ x\} \in sets\ (stream\text{-}space\ M)$
 shows $\mathcal{P}(x\ in\ stream\text{-}space\ M.\ P\ x) = (\int^+ t.\ \mathcal{P}(x\ in\ stream\text{-}space\ M.\ P\ (t\ \#\#\ x))\ \partial M)$
⟨*proof*⟩

lemma (**in** *prob-space*) *AE-stream-space*:
 assumes [*measurable*]: *Measurable.pred* (*stream-space M*) *P*
 shows $(AE\ X\ in\ stream\text{-}space\ M.\ P\ X) = (AE\ x\ in\ M.\ AE\ X\ in\ stream\text{-}space\ M.\ P\ (x\ \#\#\ X))$
⟨*proof*⟩

lemma (**in** *prob-space*) *AE-stream-all*:
 assumes [*measurable*]: *Measurable.pred M P* **and** *P*: *AE x in M. P x*
 shows *AE x in stream-space M. stream-all P x*
⟨*proof*⟩

lemma *streams-sets*:
 assumes *X*[*measurable*]: $X \in sets\ M$ **shows** $streams\ X \in sets\ (stream\text{-}space\ M)$
⟨*proof*⟩

lemma *sets-stream-space-in-sets*:
 assumes *space*: $space\ N = streams\ (space\ M)$
 assumes *sets*: $\bigwedge i.\ (\lambda x.\ x\ !!\ i) \in measurable\ N\ M$
 shows $sets\ (stream\text{-}space\ M) \subseteq sets\ N$
 ⟨*proof*⟩

lemma *sets-stream-space-eq*: *sets* (*stream-space M*) =
sets ($\bigsqcup_\sigma$ *i*∈*UNIV*. *vimage-algebra* (*streams* (*space M*)) (λ*s*. *s* !! *i*) *M*)
⟨*proof*⟩

lemma *sets-restrict-stream-space*:
assumes *S*[*measurable*]: *S* ∈ *sets M*
shows *sets* (*restrict-space* (*stream-space M*) (*streams S*)) = *sets* (*stream-space* (*restrict-space M S*))
⟨*proof*⟩

primrec *sstart* :: ′*a set* ⇒ ′*a list* ⇒ ′*a stream set* **where**
sstart S [] = *streams S*
| [*simp del*]: *sstart S* (*x* # *xs*) = *op* ## *x* ' *sstart S xs*

lemma *in-sstart*[*simp*]: *s* ∈ *sstart S* (*x* # *xs*) ⟷ *shd s* = *x* ∧ *stl s* ∈ *sstart S xs*
⟨*proof*⟩

lemma *sstart-in-streams*: *xs* ∈ *lists S* ⟹ *sstart S xs* ⊆ *streams S*
⟨*proof*⟩

lemma *sstart-eq*: *x* ∈ *streams S* ⟹ *x* ∈ *sstart S xs* = (∀ *i*<*length xs*. *x* !! *i* = *xs* ! *i*)
⟨*proof*⟩

lemma *sstart-sets*: *sstart S xs* ∈ *sets* (*stream-space* (*count-space UNIV*))
⟨*proof*⟩

lemma *sigma-sets-singletons*:
assumes *countable S*
shows *sigma-sets S* ((λ*s*. {*s*})'*S*) = *Pow S*
⟨*proof*⟩

lemma *sets-count-space-eq-sigma*:
countable S ⟹ *sets* (*count-space S*) = *sets* (*sigma S* ((λ*s*. {*s*})'*S*))
⟨*proof*⟩

lemma *sets-stream-space-sstart*:
assumes *S*[*simp*]: *countable S*
shows *sets* (*stream-space* (*count-space S*)) = *sets* (*sigma* (*streams S*) (*sstart S*'*lists S* ∪ {{}}))
⟨*proof*⟩

lemma *Int-stable-sstart*: *Int-stable* (*sstart S*'*lists S* ∪ {{}})
⟨*proof*⟩

lemma *stream-space-eq-sstart*:
assumes *S*[*simp*]: *countable S*
assumes *P*: *prob-space M prob-space N*

assumes *ae*: *AE x in M. x* ∈ *streams S AE x in N. x* ∈ *streams S*
assumes *sets-M*: *sets M* = *sets* (*stream-space* (*count-space UNIV*))
assumes *sets-N*: *sets N* = *sets* (*stream-space* (*count-space UNIV*))
assumes ∗: ⋀*xs. xs* ≠ [] ⟹ *xs* ∈ *lists S* ⟹ *emeasure M* (*sstart S xs*) = *emeasure N* (*sstart S xs*)
shows *M* = *N*
⟨*proof*⟩

end

31 Embed Measure Spaces with a Function

theory *Embed-Measure*
imports *Binary-Product-Measure*
begin

definition *embed-measure* :: *'a measure* ⇒ (*'a* ⇒ *'b*) ⇒ *'b measure* **where**
embed-measure M f = *measure-of* (*f* ' *space M*) {*f* ' *A* |*A. A* ∈ *sets M*}
(λ*A. emeasure M* (*f* −' *A* ∩ *space M*))

lemma *space-embed-measure*: *space* (*embed-measure M f*) = *f* ' *space M*
⟨*proof*⟩

lemma *sets-embed-measure'*:
assumes *inj*: *inj-on f* (*space M*)
shows *sets* (*embed-measure M f*) = {*f* ' *A* |*A. A* ∈ *sets M*}
⟨*proof*⟩

lemma *the-inv-into-vimage*:
inj-on f X ⟹ *A* ⊆ *X* ⟹ *the-inv-into X f* −' *A* ∩ (*f*'*X*) = *f* ' *A*
⟨*proof*⟩

lemma *sets-embed-eq-vimage-algebra*:
assumes *inj-on f* (*space M*)
shows *sets* (*embed-measure M f*) = *sets* (*vimage-algebra* (*f*'*space M*) (*the-inv-into* (*space M*) *f*) *M*)
⟨*proof*⟩

lemma *sets-embed-measure*:
assumes *inj*: *inj f*
shows *sets* (*embed-measure M f*) = {*f* ' *A* |*A. A* ∈ *sets M*}
⟨*proof*⟩

lemma *in-sets-embed-measure*: *A* ∈ *sets M* ⟹ *f* ' *A* ∈ *sets* (*embed-measure M f*)
⟨*proof*⟩

lemma *measurable-embed-measure1*:
assumes *g*: (λ*x. g* (*f x*)) ∈ *measurable M N*
shows *g* ∈ *measurable* (*embed-measure M f*) *N*

⟨*proof*⟩

lemma *measurable-embed-measure2'*:
assumes *inj-on f* (*space M*)
shows $f \in$ *measurable M* (*embed-measure M f*)
⟨*proof*⟩

lemma *measurable-embed-measure2*:
assumes [*simp*]: *inj f* **shows** $f \in$ *measurable M* (*embed-measure M f*)
⟨*proof*⟩

lemma *embed-measure-eq-distr'*:
assumes *inj-on f* (*space M*)
shows *embed-measure M f* = *distr M* (*embed-measure M f*) *f*
⟨*proof*⟩

lemma *embed-measure-eq-distr*:
inj f $\Longrightarrow$ *embed-measure M f* = *distr M* (*embed-measure M f*) *f*
⟨*proof*⟩

lemma *nn-integral-embed-measure'*:
inj-on f (*space M*) $\Longrightarrow$ $g \in$ *borel-measurable* (*embed-measure M f*) $\Longrightarrow$
nn-integral (*embed-measure M f*) *g* = *nn-integral M* ($\lambda x.$ *g* (*f x*))
⟨*proof*⟩

lemma *nn-integral-embed-measure*:
inj f $\Longrightarrow$ $g \in$ *borel-measurable* (*embed-measure M f*) $\Longrightarrow$
nn-integral (*embed-measure M f*) *g* = *nn-integral M* ($\lambda x.$ *g* (*f x*))
⟨*proof*⟩

lemma *emeasure-embed-measure'*:
assumes *inj-on f* (*space M*) $A \in$ *sets* (*embed-measure M f*)
shows *emeasure* (*embed-measure M f*) *A* = *emeasure M* (*f* −' $A \cap$ *space M*)
⟨*proof*⟩

lemma *emeasure-embed-measure*:
assumes *inj f* $A \in$ *sets* (*embed-measure M f*)
shows *emeasure* (*embed-measure M f*) *A* = *emeasure M* (*f* −' $A \cap$ *space M*)
⟨*proof*⟩

lemma *embed-measure-comp*:
assumes [*simp*]: *inj f inj g*
shows *embed-measure* (*embed-measure M f*) *g* = *embed-measure M* ($g \circ f$)
⟨*proof*⟩

lemma *sigma-finite-embed-measure*:
assumes *sigma-finite-measure M* **and** *inj*: *inj f*
shows *sigma-finite-measure* (*embed-measure M f*)
⟨*proof*⟩

lemma *embed-measure-count-space′*:
inj-on f A $\Longrightarrow$ *embed-measure* (*count-space A*) *f* = *count-space* (*f'A*)
⟨*proof*⟩

lemma *embed-measure-count-space*:
inj f $\Longrightarrow$ *embed-measure* (*count-space A*) *f* = *count-space* (*f'A*)
⟨*proof*⟩

lemma *sets-embed-measure-alt*:
inj f $\Longrightarrow$ *sets* (*embed-measure M f*) = (*op'f*) ' *sets M*
⟨*proof*⟩

lemma *emeasure-embed-measure-image′*:
assumes *inj-on f* (*space M*) *X* $\in$ *sets M*
shows *emeasure* (*embed-measure M f*) (*f'X*) = *emeasure M X*
⟨*proof*⟩

lemma *emeasure-embed-measure-image*:
inj f $\Longrightarrow$ *X* $\in$ *sets M* $\Longrightarrow$ *emeasure* (*embed-measure M f*) (*f'X*) = *emeasure M X*
⟨*proof*⟩

lemma *embed-measure-eq-iff*:
assumes *inj f*
shows *embed-measure A f* = *embed-measure B f* $\longleftrightarrow$ *A* = *B* (**is** *?M* = *?N* $\longleftrightarrow$ -)
⟨*proof*⟩

lemma *the-inv-into-in-Pi*: *inj-on f A* $\Longrightarrow$ *the-inv-into A f* $\in$ *f* ' *A* $\to$ *A*
⟨*proof*⟩

lemma *map-prod-image*: *map-prod f g* ' (*A* $\times$ *B*) = (*f'A*) $\times$ (*g'B*)
⟨*proof*⟩

lemma *map-prod-vimage*: *map-prod f g* −' (*A* $\times$ *B*) = (*f*−'*A*) $\times$ (*g*−'*B*)
⟨*proof*⟩

lemma *embed-measure-prod*:
assumes *f*: *inj f* **and** *g*: *inj g* **and** [*simp*]: *sigma-finite-measure M sigma-finite-measure N*
shows *embed-measure M f* $\bigotimes_M$ *embed-measure N g* = *embed-measure* (*M* $\bigotimes_M$ *N*) (λ(*x*, *y*). (*f x*, *g y*))
(**is** *?L* = -)
⟨*proof*⟩

lemma *density-embed-measure*:
assumes *inj*: *inj f* **and** *Mg*[*measurable*]: *g* $\in$ *borel-measurable* (*embed-measure M f*)

shows *density* (*embed-measure* $M f$) g = *embed-measure* (*density* M ($g \circ f$)) f
(**is** *?M1* = *?M2*)
⟨*proof*⟩

lemma *density-embed-measure'*:
assumes *inj*: *inj f* **and** *inv*: $\bigwedge x.\ f'\ (f\ x) = x$ **and** *Mg*[*measurable*]: $g \in$ *borel-measurable* M
shows *density* (*embed-measure* $M f$) ($g \circ f'$) = *embed-measure* (*density* M g) f
⟨*proof*⟩

lemma *inj-on-image-subset-iff*:
assumes *inj-on* f C $A \subseteq C$ $B \subseteq C$
shows f ` $A \subseteq f$ ` $B \longleftrightarrow A \subseteq B$
⟨*proof*⟩

lemma *AE-embed-measure'*:
assumes *inj*: *inj-on* f (*space* M)
shows (*AE* x *in embed-measure* $M f$. $P\ x$) $\longleftrightarrow$ (*AE* x *in* M. P ($f\ x$))
⟨*proof*⟩

lemma *AE-embed-measure*:
assumes *inj*: *inj f*
shows (*AE* x *in embed-measure* $M f$. $P\ x$) $\longleftrightarrow$ (*AE* x *in* M. P ($f\ x$))
⟨*proof*⟩

lemma *nn-integral-monotone-convergence-SUP-countable*:
fixes f :: $'a \Rightarrow\ 'b \Rightarrow$ *ennreal*
assumes *nonempty*: $Y \neq \{\}$
and *chain*: *Complete-Partial-Order.chain op* $\leq$ (f ` Y)
and *countable*: *countable* B
shows ($\int^+ x.$ (*SUP* $i{:}Y$. $f\ i\ x$) ∂*count-space* B) = (*SUP* $i{:}Y$. ($\int^+ x.\ f\ i\ x$ ∂*count-space* B))
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

end

32 Non-denumerability of the Continuum.

theory *ContNotDenum*
imports *Complex-Main Countable-Set*
begin

32.1 Abstract

The following document presents a proof that the Continuum is uncountable. It is formalised in the Isabelle/Isar theorem proving system.

Theorem: The Continuum $\mathbb{R}$ is not denumerable. In other words, there does not exist a function f: $\mathbb{N} \Rightarrow \mathbb{R}$ such that f is surjective.

Outline: An elegant informal proof of this result uses Cantor's Diagonalisation argument. The proof presented here is not this one. First we formalise some properties of closed intervals, then we prove the Nested Interval Property. This property relies on the completeness of the Real numbers and is the foundation for our argument. Informally it states that an intersection of countable closed intervals (where each successive interval is a subset of the last) is non-empty. We then assume a surjective function f: $\mathbb{N} \Rightarrow \mathbb{R}$ exists and find a real x such that x is not in the range of f by generating a sequence of closed intervals then using the NIP.

theorem *real-non-denum*: ¬ (∃ *f* :: *nat* ⇒ *real*. *surj f*)
⟨*proof*⟩

lemma *uncountable-UNIV-real*: *uncountable* (*UNIV*::*real set*)
⟨*proof*⟩

lemma *bij-betw-open-intervals*:
fixes *a b c d* :: *real*
assumes *a* < *b c* < *d*
shows ∃ *f*. *bij-betw f* {*a*<..<*b*} {*c*<..<*d*}
⟨*proof*⟩

lemma *bij-betw-tan*: *bij-betw tan* {−*pi*/*2*<..<*pi*/*2*} *UNIV*
⟨*proof*⟩

lemma *uncountable-open-interval*:
fixes *a b* :: *real*
shows *uncountable* {*a*<..<*b*} ⟷ *a* < *b*
⟨*proof*⟩

lemma *uncountable-half-open-interval-1*:
fixes *a* :: *real* **shows** *uncountable* {*a*..<*b*} ⟷ *a*<*b*
⟨*proof*⟩

lemma *uncountable-half-open-interval-2*:
fixes *a* :: *real* **shows** *uncountable* {*a*<..*b*} ⟷ *a*<*b*
⟨*proof*⟩

lemma *real-interval-avoid-countable-set*:
fixes *a b* :: *real* **and** *A* :: *real set*
assumes *a* < *b* **and** *countable A*
shows ∃ *x*∈{*a*<..<*b*}. *x* ∉ *A*
⟨*proof*⟩

lemma *open-minus-countable*:
fixes *S A* :: *real set* **assumes** *countable A S* ≠ {} *open S*

shows $\exists x \in S.\ x \notin A$
⟨*proof*⟩

end

33 Distribution Functions

Shows that the cumulative distribution function (cdf) of a distribution (a measure on the reals) is nondecreasing and right continuous, which tends to 0 and 1 in either direction.

Conversely, every such function is the cdf of a unique distribution. This direction defines the measure in the obvious way on half-open intervals, and then applies the Caratheodory extension theorem.

theory *Distribution-Functions*
imports *Probability-Measure* ~~/*src*/*HOL*/*Library*/*ContNotDenum*
begin

lemma *UN-Ioc-eq-UNIV*: $(\bigcup n.\ \{\ -real\ n <..\ real\ n\}) = UNIV$
⟨*proof*⟩

33.1 Properties of cdf's

definition
cdf :: *real measure* ⇒ *real* ⇒ *real*
where
cdf M ≡ λ*x*. *measure M* {..*x*}

lemma *cdf-def2*: *cdf M x* = *measure M* {..*x*}
⟨*proof*⟩

locale *finite-borel-measure* = *finite-measure M* **for** *M* :: *real measure* +
assumes *M-super-borel*: *sets borel* ⊆ *sets M*
begin

lemma *sets-M*[*intro*]: *a* ∈ *sets borel* ⟹ *a* ∈ *sets M*
⟨*proof*⟩

lemma *cdf-diff-eq*:
assumes *x* < *y*
shows *cdf M y* − *cdf M x* = *measure M* {*x*<..*y*}
⟨*proof*⟩

lemma *cdf-nondecreasing*: *x* ≤ *y* ⟹ *cdf M x* ≤ *cdf M y*
⟨*proof*⟩

lemma *borel-UNIV*: *space M* = *UNIV*
⟨*proof*⟩

lemma *cdf-nonneg*: *cdf M x* ≥ *0*
⟨*proof*⟩

lemma *cdf-bounded*: *cdf M x* ≤ *measure M* (*space M*)
⟨*proof*⟩

lemma *cdf-lim-infty*:
((λ*i*. *cdf M* (*real i*)) ⟶ *measure M* (*space M*))
⟨*proof*⟩

lemma *cdf-lim-at-top*: (*cdf M* ⟶ *measure M* (*space M*)) *at-top*
⟨*proof*⟩

lemma *cdf-lim-neg-infty*: ((λ*i*. *cdf M* (− *real i*)) ⟶ *0*)
⟨*proof*⟩

lemma *cdf-lim-at-bot*: (*cdf M* ⟶ *0*) *at-bot*
⟨*proof*⟩

lemma *cdf-is-right-cont*: *continuous* (*at-right a*) (*cdf M*)
⟨*proof*⟩

lemma *cdf-at-left*: (*cdf M* ⟶ *measure M* {..<*a*}) (*at-left a*)
⟨*proof*⟩

lemma *isCont-cdf*: *isCont* (*cdf M*) *x* ⟷ *measure M* {*x*} = *0*
⟨*proof*⟩

lemma *countable-atoms*: *countable* {*x*. *measure M* {*x*} > *0*}
⟨*proof*⟩

end

locale *real-distribution* = *prob-space M* **for** *M* :: *real measure* +
assumes *events-eq-borel* [*simp*, *measurable-cong*]: *sets M* = *sets borel* **and** *space-eq-univ* [*simp*]: *space M* = *UNIV*
begin

sublocale *finite-borel-measure M*
⟨*proof*⟩

lemma *cdf-bounded-prob*: ⋀*x*. *cdf M x* ≤ *1*
⟨*proof*⟩

lemma *cdf-lim-infty-prob*: (λ*i*. *cdf M* (*real i*)) ⟶ *1*
⟨*proof*⟩

lemma *cdf-lim-at-top-prob*: (*cdf M* ⟶ *1*) *at-top*

⟨*proof*⟩

lemma *measurable-finite-borel* [*simp*]:
f ∈ *borel-measurable borel* ⟹ *f* ∈ *borel-measurable M*
⟨*proof*⟩

end

lemma (**in** *prob-space*) *real-distribution-distr* [*intro*, *simp*]:
random-variable borel X ⟹ *real-distribution* (*distr M borel X*)
⟨*proof*⟩

33.2 uniqueness

lemma (**in** *real-distribution*) *emeasure-Ioc*:
assumes $a \leq b$ **shows** *emeasure M* {*a* <.. *b*} = *cdf M b* − *cdf M a*
⟨*proof*⟩

lemma *cdf-unique*:
fixes *M1 M2*
assumes *real-distribution M1* **and** *real-distribution M2*
assumes *cdf M1* = *cdf M2*
shows *M1* = *M2*
⟨*proof*⟩

lemma *real-distribution-interval-measure*:
fixes *F* :: *real* ⇒ *real*
assumes *nondecF* : ⋀ *x y*. *x* ≤ *y* ⟹ *F x* ≤ *F y* **and**
right-cont-F : ⋀*a*. *continuous* (*at-right a*) *F* **and**
lim-F-at-bot : (*F* ⟶ *0*) *at-bot* **and**
lim-F-at-top : (*F* ⟶ *1*) *at-top*
shows *real-distribution* (*interval-measure F*)
⟨*proof*⟩

lemma *cdf-interval-measure*:
fixes *F* :: *real* ⇒ *real*
assumes *nondecF* : ⋀ *x y*. *x* ≤ *y* ⟹ *F x* ≤ *F y* **and**
right-cont-F : ⋀*a*. *continuous* (*at-right a*) *F* **and**
lim-F-at-bot : (*F* ⟶ *0*) *at-bot* **and**
lim-F-at-top : (*F* ⟶ *1*) *at-top*
shows *cdf* (*interval-measure F*) = *F*
⟨*proof*⟩

end

34 Weak Convergence of Functions and Distributions

Properties of weak convergence of functions and measures, including the portmanteau theorem.

theory *Weak-Convergence*
  imports *Distribution-Functions*
begin

35 Weak Convergence of Functions

definition
  weak-conv :: (*nat* ⇒ (*real* ⇒ *real*)) ⇒ (*real* ⇒ *real*) ⇒ *bool*
where
  weak-conv F-seq F ≡ ∀ *x*. *isCont F x* ⟶ (λ*n*. *F-seq n x*) ⟶ *F x*

36 Weak Convergence of Distributions

definition
  weak-conv-m :: (*nat* ⇒ *real measure*) ⇒ *real measure* ⇒ *bool*
where
  weak-conv-m M-seq M ≡ *weak-conv* (λ*n*. *cdf* (*M-seq n*)) (*cdf M*)

37 Skorohod's theorem

locale *right-continuous-mono* =
  fixes *f* :: *real* ⇒ *real* **and** *a b* :: *real*
  assumes *cont*: ⋀*x*. *continuous* (*at-right x*) *f*
  assumes *mono*: *mono f*
  assumes *bot*: (*f* ⟶ *a*) *at-bot*
  assumes *top*: (*f* ⟶ *b*) *at-top*
begin

abbreviation *I* :: *real* ⇒ *real* **where**
  I ω ≡ *Inf* {*x*. ω ≤ *f x*}

lemma *pseudoinverse*: **assumes** *a* < ω ω < *b* **shows** ω ≤ *f x* ⟷ *I* ω ≤ *x*
⟨*proof*⟩

lemma *pseudoinverse'*: ∀ ω∈{*a*<..<*b*}. ∀ *x*. ω ≤ *f x* ⟷ *I* ω ≤ *x*
  ⟨*proof*⟩

lemma *mono-I*: *mono-on I* {*a* <..< *b*}
  ⟨*proof*⟩

end

locale *cdf-distribution* = *real-distribution*
begin

abbreviation $C \equiv cdf\ M$

sublocale *right-continuous-mono C 0 1*
⟨*proof*⟩

lemma *measurable-C*[*measurable*]: $C \in$ *borel-measurable borel*
⟨*proof*⟩

lemma *measurable-CI*[*measurable*]: $I \in$ *borel-measurable* (*restrict-space borel* {*0*<..<*1*})
⟨*proof*⟩

lemma *emeasure-distr-I*: *emeasure* (*distr* (*restrict-space lborel* {*0*<..<*1*::*real*}) *borel I*) *UNIV* = *1*
⟨*proof*⟩

lemma *distr-I-eq-M*: *distr* (*restrict-space lborel* {*0*<..<*1*::*real*}) *borel I* = *M* (**is** *?I* = -)
⟨*proof*⟩

end

context
fixes μ :: *nat* ⇒ *real measure*
and *M* :: *real measure*
assumes μ: ⋀*n*. *real-distribution* (μ *n*)
assumes *M*: *real-distribution M*
assumes μ*-to-M*: *weak-conv-m* μ *M*
begin

theorem *Skorohod*:
∃ (Ω :: *real measure*) (*Y-seq* :: *nat* ⇒ *real* ⇒ *real*) (*Y* :: *real* ⇒ *real*).
prob-space Ω ∧
(∀ *n*. *Y-seq n* ∈ *measurable* Ω *borel*) ∧
(∀ *n*. *distr* Ω *borel* (*Y-seq n*) = μ *n*) ∧
Y ∈ *measurable* Ω *lborel* ∧
distr Ω *borel Y* = *M* ∧
(∀ *x* ∈ *space* Ω. (λ*n*. *Y-seq n x*) ⟶ *Y x*)
⟨*proof*⟩

The Portmanteau theorem, that is, the equivalence of various definitions of weak convergence.

theorem *weak-conv-imp-bdd-ae-continuous-conv*:
fixes
f :: *real* ⇒ *'a*::{*banach*, *second-countable-topology*}

assumes
discont-null: M ({x. ¬ *isCont f x*}) = *0* **and**
f-bdd: ⋀x. *norm* (*f x*) ≤ B **and**
[*measurable*]: f ∈ *borel-measurable borel*
shows
(λ n. *integral*^{L} (μ n) f) ⟶ *integral*^{L} *M f*
⟨*proof*⟩

theorem *weak-conv-imp-integral-bdd-continuous-conv*:
fixes f :: *real* ⇒ *'a*::{*banach*, *second-countable-topology*}
assumes
⋀x. *isCont f x* **and**
⋀x. *norm* (*f x*) ≤ B
shows
(λ n. *integral*^{L} (μ n) f) ⟶ *integral*^{L} *M f*
⟨*proof*⟩

theorem *weak-conv-imp-continuity-set-conv*:
fixes f :: *real* ⇒ *real*
assumes [*measurable*]: A ∈ *sets borel* **and** M (*frontier A*) = *0*
shows (λn. *measure* (μ n) A) ⟶ *measure M A*
⟨*proof*⟩

end

definition
cts-step :: *real* ⇒ *real* ⇒ *real* ⇒ *real*
where
cts-step a b x ≡ *if x* ≤ *a then 1 else if x* ≥ *b then 0 else* (b − x) / (b − a)

lemma *cts-step-uniformly-continuous*:
assumes [*arith*]: $a < b$
shows *uniformly-continuous-on UNIV* (*cts-step a b*)
⟨*proof*⟩

lemma (**in** *real-distribution*) *integrable-cts-step*: $a < b$ ⟹ *integrable M* (*cts-step a b*)
⟨*proof*⟩

lemma (**in** *real-distribution*) *cdf-cts-step*:
assumes [*arith*]: $x < y$
shows *cdf M x* ≤ *integral*^{L} *M* (*cts-step x y*) **and** *integral*^{L} *M* (*cts-step x y*) ≤ *cdf M y*
⟨*proof*⟩

context
fixes *M-seq* :: *nat* ⇒ *real measure*
and M :: *real measure*
assumes *distr-M-seq* [*simp*]: ⋀n. *real-distribution* (*M-seq n*)

assumes *distr-M* [*simp*]: *real-distribution M*
begin

theorem *continuity-set-conv-imp-weak-conv*:
fixes f :: *real* ⇒ *real*
assumes ∗: $\bigwedge A.\ A \in$ *sets borel* $\Longrightarrow M$ (*frontier A*) $= 0 \Longrightarrow (\lambda\ n.$ (*measure* (*M-seq n*) *A*)) $\longrightarrow$ *measure M A*
shows *weak-conv-m M-seq M*
⟨*proof*⟩

theorem *integral-cts-step-conv-imp-weak-conv*:
assumes *integral-conv*: $\bigwedge x\ y.\ x < y \Longrightarrow (\lambda n.\ integral^L$ (*M-seq n*) (*cts-step x y*)) $\longrightarrow integral^L$ *M* (*cts-step x y*)
shows *weak-conv-m M-seq M*
⟨*proof*⟩

theorem *integral-bdd-continuous-conv-imp-weak-conv*:
assumes
$\bigwedge f.$ ($\bigwedge x.$ *isCont f x*) $\Longrightarrow$ ($\bigwedge x.$ *abs* (*f x*) ≤ 1) $\Longrightarrow (\lambda n.\ integral^L$ (*M-seq n*) *f*::*real*) $\longrightarrow integral^L$ *M f*
shows
weak-conv-m M-seq M
⟨*proof*⟩

end

end

38 Independent families of events, event sets, and random variables

theory *Independent-Family*
imports *Probability-Measure Infinite-Product-Measure*
begin

definition (**in** *prob-space*)
indep-sets F I ⟷ (∀ *i*∈*I*. *F i* ⊆ *events*) ∧
(∀ *J*⊆*I*. *J* ≠ {} ⟶ *finite J* ⟶ (∀ *A*∈*Pi J F*. *prob* (⋂*j*∈*J*. *A j*) = (∏*j*∈*J*. *prob* (*A j*))))

definition (**in** *prob-space*)
indep-set A B ⟷ *indep-sets* (*case-bool A B*) *UNIV*

definition (**in** *prob-space*)
indep-events-def-alt: *indep-events A I* ⟷ *indep-sets* (λ*i*. {*A i*}) *I*

lemma (**in** *prob-space*) *indep-events-def*:
indep-events A I ⟷ (*A*'*I* ⊆ *events*) ∧

$(\forall J \subseteq I.\ J \neq \{\} \longrightarrow \mathit{finite}\ J \longrightarrow \mathit{prob}\ (\bigcap j \in J.\ A\ j) = (\prod j \in J.\ \mathit{prob}\ (A\ j)))$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-eventsI*:
$(\bigwedge i.\ i \in I \Longrightarrow F\ i \in \mathit{sets}\ M) \Longrightarrow (\bigwedge J.\ J \subseteq I \Longrightarrow \mathit{finite}\ J \Longrightarrow J \neq \{\} \Longrightarrow \mathit{prob}$ $(\bigcap i \in J.\ F\ i) = (\prod i \in J.\ \mathit{prob}\ (F\ i))) \Longrightarrow \mathit{indep\text{-}events}\ F\ I$
⟨*proof*⟩

definition (**in** *prob-space*)
$\mathit{indep\text{-}event}\ A\ B \longleftrightarrow \mathit{indep\text{-}events}\ (\mathit{case\text{-}bool}\ A\ B)\ \mathit{UNIV}$

lemma (**in** *prob-space*) *indep-sets-cong*:
$I = J \Longrightarrow (\bigwedge i.\ i \in I \Longrightarrow F\ i = G\ i) \Longrightarrow \mathit{indep\text{-}sets}\ F\ I \longleftrightarrow \mathit{indep\text{-}sets}\ G\ J$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-events-finite-index-events*:
$\mathit{indep\text{-}events}\ F\ I \longleftrightarrow (\forall J \subseteq I.\ J \neq \{\} \longrightarrow \mathit{finite}\ J \longrightarrow \mathit{indep\text{-}events}\ F\ J)$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-sets-finite-index-sets*:
$\mathit{indep\text{-}sets}\ F\ I \longleftrightarrow (\forall J \subseteq I.\ J \neq \{\} \longrightarrow \mathit{finite}\ J \longrightarrow \mathit{indep\text{-}sets}\ F\ J)$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-sets-mono-index*:
$J \subseteq I \Longrightarrow \mathit{indep\text{-}sets}\ F\ I \Longrightarrow \mathit{indep\text{-}sets}\ F\ J$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-sets-mono-sets*:
assumes *indep*: $\mathit{indep\text{-}sets}\ F\ I$
assumes *mono*: $\bigwedge i.\ i \in I \Longrightarrow G\ i \subseteq F\ i$
shows $\mathit{indep\text{-}sets}\ G\ I$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-sets-mono*:
assumes *indep*: $\mathit{indep\text{-}sets}\ F\ I$
assumes *mono*: $J \subseteq I\ \bigwedge i.\ i \in J \Longrightarrow G\ i \subseteq F\ i$
shows $\mathit{indep\text{-}sets}\ G\ J$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-setsI*:
assumes $\bigwedge i.\ i \in I \Longrightarrow F\ i \subseteq \mathit{events}$
and $\bigwedge A\ J.\ J \neq \{\} \Longrightarrow J \subseteq I \Longrightarrow \mathit{finite}\ J \Longrightarrow (\forall j \in J.\ A\ j \in F\ j) \Longrightarrow \mathit{prob}$ $(\bigcap j \in J.\ A\ j) = (\prod j \in J.\ \mathit{prob}\ (A\ j))$
shows $\mathit{indep\text{-}sets}\ F\ I$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-setsD*:
assumes $\mathit{indep\text{-}sets}\ F\ I$ **and** $J \subseteq I\ J \neq \{\}\ \mathit{finite}\ J\ \forall j \in J.\ A\ j \in F\ j$
shows $\mathit{prob}\ (\bigcap j \in J.\ A\ j) = (\prod j \in J.\ \mathit{prob}\ (A\ j))$

⟨proof⟩

lemma (**in** *prob-space*) *indep-setI*:
assumes *ev*: $A \subseteq events$ $B \subseteq events$
and *indep*: $\bigwedge a\ b.\ a \in A \Longrightarrow b \in B \Longrightarrow prob\ (a \cap b) = prob\ a * prob\ b$
shows *indep-set A B*
⟨proof⟩

lemma (**in** *prob-space*) *indep-setD*:
assumes *indep*: *indep-set A B* **and** *ev*: $a \in A$ $b \in B$
shows $prob\ (a \cap b) = prob\ a * prob\ b$
⟨proof⟩

lemma (**in** *prob-space*)
assumes *indep*: *indep-set A B*
shows *indep-setD-ev1*: $A \subseteq events$
and *indep-setD-ev2*: $B \subseteq events$
⟨proof⟩

lemma (**in** *prob-space*) *indep-sets-dynkin*:
assumes *indep*: *indep-sets F I*
shows *indep-sets* ($\lambda i.$ *dynkin* (*space M*) (*F i*)) *I*
(**is** *indep-sets ?F I*)
⟨proof⟩

lemma (**in** *prob-space*) *indep-sets-sigma*:
assumes *indep*: *indep-sets F I*
assumes *stable*: $\bigwedge i.\ i \in I \Longrightarrow$ *Int-stable* (*F i*)
shows *indep-sets* ($\lambda i.$ *sigma-sets* (*space M*) (*F i*)) *I*
⟨proof⟩

lemma (**in** *prob-space*) *indep-sets-sigma-sets-iff*:
assumes $\bigwedge i.\ i \in I \Longrightarrow$ *Int-stable* (*F i*)
shows *indep-sets* ($\lambda i.$ *sigma-sets* (*space M*) (*F i*)) *I* $\longleftrightarrow$ *indep-sets F I*
⟨proof⟩

definition (**in** *prob-space*)
indep-vars-def2: *indep-vars* M' *X I* $\longleftrightarrow$
($\forall i{\in}I.$ *random-variable* (M' *i*) (*X i*)) $\wedge$
indep-sets ($\lambda i.$ { *X i* −‘ *A* $\cap$ *space M* | *A*. *A* $\in$ *sets* (M' *i*)}) *I*

definition (**in** *prob-space*)
indep-var Ma A Mb B $\longleftrightarrow$ *indep-vars* (*case-bool Ma Mb*) (*case-bool A B*) *UNIV*

lemma (**in** *prob-space*) *indep-vars-def*:
indep-vars M' *X I* $\longleftrightarrow$
($\forall i{\in}I.$ *random-variable* (M' *i*) (*X i*)) $\wedge$
indep-sets ($\lambda i.$ *sigma-sets* (*space M*) { *X i* −‘ *A* $\cap$ *space M* | *A*. *A* $\in$ *sets* (M'
i)}) *I*

⟨*proof*⟩

lemma (**in** *prob-space*) *indep-var-eq*:
indep-var $S\ X\ T\ Y \longleftrightarrow$
($random\text{-}variable\ S\ X \land random\text{-}variable\ T\ Y$) $\land$
indep-set
($sigma\text{-}sets\ (space\ M)\ \{\ X\ -`\ A \cap space\ M\ |\ A.\ A \in sets\ S\}$)
($sigma\text{-}sets\ (space\ M)\ \{\ Y\ -`\ A \cap space\ M\ |\ A.\ A \in sets\ T\}$)
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-sets2-eq*:
$indep\text{-}set\ A\ B \longleftrightarrow A \subseteq events \land B \subseteq events \land (\forall a{\in}A.\ \forall b{\in}B.\ prob\ (a \cap b) = prob\ a * prob\ b)$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-set-sigma-sets*:
assumes *indep-set A B*
assumes *A*: *Int-stable A* **and** *B*: *Int-stable B*
shows $indep\text{-}set\ (sigma\text{-}sets\ (space\ M)\ A)\ (sigma\text{-}sets\ (space\ M)\ B)$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-eventsI-indep-vars*:
assumes *indep*: *indep-vars N X I*
assumes *P*: $\bigwedge i.\ i \in I \Longrightarrow \{x{\in}space\ (N\ i).\ P\ i\ x\} \in sets\ (N\ i)$
shows $indep\text{-}events\ (\lambda i.\ \{x{\in}space\ M.\ P\ i\ (X\ i\ x)\})\ I$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-sets-collect-sigma*:
fixes $I :: 'j \Rightarrow 'i\ set$ **and** $J :: 'j\ set$ **and** $E :: 'i \Rightarrow 'a\ set\ set$
assumes *indep*: $indep\text{-}sets\ E\ (\bigcup j{\in}J.\ I\ j)$
assumes *Int-stable*: $\bigwedge i\ j.\ j \in J \Longrightarrow i \in I\ j \Longrightarrow Int\text{-}stable\ (E\ i)$
assumes *disjoint*: *disjoint-family-on I J*
shows $indep\text{-}sets\ (\lambda j.\ sigma\text{-}sets\ (space\ M)\ (\bigcup i{\in}I\ j.\ E\ i))\ J$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-vars-restrict*:
assumes *ind*: $indep\text{-}vars\ M'\ X\ I$ **and** *K*: $\bigwedge j.\ j \in L \Longrightarrow K\ j \subseteq I$ **and** *J*: *disjoint-family-on K L*
shows $indep\text{-}vars\ (\lambda j.\ PiM\ (K\ j)\ M')\ (\lambda j\ \omega.\ restrict\ (\lambda i.\ X\ i\ \omega)\ (K\ j))\ L$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-var-restrict*:
assumes *ind*: $indep\text{-}vars\ M'\ X\ I$ **and** *AB*: $A \cap B = \{\}\ A \subseteq I\ B \subseteq I$
shows $indep\text{-}var\ (PiM\ A\ M')\ (\lambda\omega.\ restrict\ (\lambda i.\ X\ i\ \omega)\ A)\ (PiM\ B\ M')\ (\lambda\omega.\ restrict\ (\lambda i.\ X\ i\ \omega)\ B)$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-vars-subset*:
assumes $indep\text{-}vars\ M'\ X\ I\ J \subseteq I$

shows *indep-vars* M' X J
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-vars-cong*:
$I = J \Longrightarrow (\bigwedge i.\ i \in I \Longrightarrow X\ i = Y\ i) \Longrightarrow (\bigwedge i.\ i \in I \Longrightarrow M'\ i = N'\ i) \Longrightarrow$ *indep-vars* $M'\ X\ I \longleftrightarrow$ *indep-vars* $N'\ Y\ J$
⟨*proof*⟩

definition (**in** *prob-space*) *tail-events* **where**
tail-events $A = (\bigcap n.$ *sigma-sets* (*space* M) (*UNION* $\{n..\}$ A))

lemma (**in** *prob-space*) *tail-events-sets*:
assumes A: $\bigwedge i$::*nat*. $A\ i \subseteq$ *events*
shows *tail-events* $A \subseteq$ *events*
⟨*proof*⟩

lemma (**in** *prob-space*) *sigma-algebra-tail-events*:
assumes $\bigwedge i$::*nat*. *sigma-algebra* (*space* M) ($A\ i$)
shows *sigma-algebra* (*space* M) (*tail-events* A)
⟨*proof*⟩

lemma (**in** *prob-space*) *kolmogorov-0-1-law*:
fixes A :: *nat* ⇒ *'a set set*
assumes $\bigwedge i$::*nat*. *sigma-algebra* (*space* M) ($A\ i$)
assumes *indep*: *indep-sets* A *UNIV*
and X: $X \in$ *tail-events* A
shows *prob* $X = 0 \lor$ *prob* $X = 1$
⟨*proof*⟩

lemma (**in** *prob-space*) *borel-0-1-law*:
fixes F :: *nat* ⇒ *'a set*
assumes *F2*: *indep-events* F *UNIV*
shows *prob* $(\bigcap n.\ \bigcup m \in \{n..\}.\ F\ m) = 0 \lor$ *prob* $(\bigcap n.\ \bigcup m \in \{n..\}.\ F\ m) = 1$
⟨*proof*⟩

lemma (**in** *prob-space*) *borel-0-1-law-AE*:
fixes P :: *nat* ⇒ *'a* ⇒ *bool*
assumes *indep-events* ($\lambda m.$ $\{x \in$ *space* $M.\ P\ m\ x\}$) *UNIV* (**is** *indep-events ?P -*)
shows (*AE* x *in* M. *infinite* $\{m.\ P\ m\ x\}$) $\lor$ (*AE* x *in* M. *finite* $\{m.\ P\ m\ x\}$)
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-sets-finite*:
assumes I: $I \neq \{\}$ *finite* I
and F: $\bigwedge i.\ i \in I \Longrightarrow F\ i \subseteq$ *events* $\bigwedge i.\ i \in I \Longrightarrow$ *space* $M \in F\ i$
shows *indep-sets* $F\ I \longleftrightarrow (\forall A \in Pi\ I\ F.$ *prob* $(\bigcap j \in I.\ A\ j) = (\prod j \in I.$ *prob* $(A\ j)))$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-vars-finite*:

fixes I :: $'i$ *set*
assumes I: $I \neq \{\}$ *finite* I
and M': $\bigwedge i.\ i \in I \Longrightarrow$ *sets* $(M'\ i) =$ *sigma-sets* (*space* $(M'\ i)$) $(E\ i)$
and *rv*: $\bigwedge i.\ i \in I \Longrightarrow$ *random-variable* $(M'\ i)\ (X\ i)$
and *Int-stable*: $\bigwedge i.\ i \in I \Longrightarrow$ *Int-stable* $(E\ i)$
and *space*: $\bigwedge i.\ i \in I \Longrightarrow$ *space* $(M'\ i) \in E\ i$ **and** *closed*: $\bigwedge i.\ i \in I \Longrightarrow E\ i \subseteq$ *Pow* (*space* $(M'\ i)$)
shows *indep-vars* $M'\ X\ I \longleftrightarrow$
$(\forall A{\in}(\Pi\ i{\in}I.\ E\ i).$ *prob* $(\bigcap j{\in}I.\ X\ j\ -`\ A\ j \cap$ *space* $M) = (\prod j{\in}I.$ *prob* $(X\ j\ -`\ A\ j \cap$ *space* $M)))$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-vars-compose*:
assumes *indep-vars* $M'\ X\ I$
assumes *rv*: $\bigwedge i.\ i \in I \Longrightarrow Y\ i \in$ *measurable* $(M'\ i)\ (N\ i)$
shows *indep-vars* $N\ (\lambda i.\ Y\ i \circ X\ i)\ I$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-vars-compose2*:
assumes *indep-vars* $M'\ X\ I$
assumes *rv*: $\bigwedge i.\ i \in I \Longrightarrow Y\ i \in$ *measurable* $(M'\ i)\ (N\ i)$
shows *indep-vars* $N\ (\lambda i\ x.\ Y\ i\ (X\ i\ x))\ I$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-var-compose*:
assumes *indep-var M1 X1 M2 X2 Y1* $\in$ *measurable M1 N1 Y2* $\in$ *measurable M2 N2*
shows *indep-var N1* (*Y1* $\circ$ *X1*) *N2* (*Y2* $\circ$ *X2*)
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-vars-Min*:
fixes X :: $'i \Rightarrow\ 'a \Rightarrow$ *real*
assumes I: *finite* $I\ i \notin I$ **and** *indep*: *indep-vars* (λ-. *borel*) X (*insert* $i\ I$)
shows *indep-var borel* $(X\ i)$ *borel* ($\lambda\omega$. *Min* $((\lambda i.\ X\ i\ \omega)`I)$)
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-vars-setsum*:
fixes X :: $'i \Rightarrow\ 'a \Rightarrow$ *real*
assumes I: *finite* $I\ i \notin I$ **and** *indep*: *indep-vars* (λ-. *borel*) X (*insert* $i\ I$)
shows *indep-var borel* $(X\ i)$ *borel* $(\lambda\omega.\ \sum i{\in}I.\ X\ i\ \omega)$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-vars-setprod*:
fixes X :: $'i \Rightarrow\ 'a \Rightarrow$ *real*
assumes I: *finite* $I\ i \notin I$ **and** *indep*: *indep-vars* (λ-. *borel*) X (*insert* $i\ I$)
shows *indep-var borel* $(X\ i)$ *borel* $(\lambda\omega.\ \prod i{\in}I.\ X\ i\ \omega)$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-varsD-finite*:

assumes X: *indep-vars* M' X I
assumes I: $I \neq \{\}$ *finite* I $\bigwedge i.\ i \in I \Longrightarrow A\ i \in$ *sets* $(M'\ i)$
shows *prob* $(\bigcap i{\in}I.\ X\ i\ -`\ A\ i \cap$ *space* $M) = (\prod i{\in}I.$ *prob* $(X\ i\ -`\ A\ i \cap$ *space* $M))$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-varsD*:
assumes X: *indep-vars* M' X I
assumes I: $J \neq \{\}$ *finite* J $J \subseteq I$ $\bigwedge i.\ i \in J \Longrightarrow A\ i \in$ *sets* $(M'\ i)$
shows *prob* $(\bigcap i{\in}J.\ X\ i\ -`\ A\ i \cap$ *space* $M) = (\prod i{\in}J.$ *prob* $(X\ i\ -`\ A\ i \cap$ *space* $M))$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-vars-iff-distr-eq-PiM*:
fixes I :: *'i set* **and** X :: $'i \Rightarrow 'a \Rightarrow 'b$
assumes $I \neq \{\}$
assumes *rv*: $\bigwedge i.$ *random-variable* $(M'\ i)\ (X\ i)$
shows *indep-vars* M' X I $\longleftrightarrow$
distr M $(\Pi_M\ i{\in}I.\ M'\ i)$ $(\lambda x.\ \lambda i{\in}I.\ X\ i\ x) = (\Pi_M\ i{\in}I.$ *distr* M $(M'\ i)$ $(X\ i))$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-varD*:
assumes *indep*: *indep-var* *Ma* A *Mb* B
assumes *sets*: $Xa \in$ *sets* *Ma* $Xb \in$ *sets* *Mb*
shows *prob* $((\lambda x.\ (A\ x,\ B\ x))\ -`\ (Xa \times Xb) \cap$ *space* $M) =$
prob $(A\ -`\ Xa \cap$ *space* $M) *$ *prob* $(B\ -`\ Xb \cap$ *space* $M)$
⟨*proof*⟩

lemma (**in** *prob-space*) *prob-indep-random-variable*:
assumes *ind*[*simp*]: *indep-var* N X N Y
assumes [*simp*]: $A \in$ *sets* N $B \in$ *sets* N
shows $\mathcal{P}(x\ in\ M.\ X\ x \in A \wedge Y\ x \in B) = \mathcal{P}(x\ in\ M.\ X\ x \in A) * \mathcal{P}(x\ in\ M.\ Y\ x \in B)$
⟨*proof*⟩

lemma (**in** *prob-space*)
assumes *indep-var* S X T Y
shows *indep-var-rv1*: *random-variable* S X
and *indep-var-rv2*: *random-variable* T Y
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-var-distribution-eq*:
indep-var S X T Y $\longleftrightarrow$ *random-variable* S X $\wedge$ *random-variable* T Y $\wedge$
distr M S X $\bigotimes_M$ *distr* M T Y $=$ *distr* M $(S \bigotimes_M T)$ $(\lambda x.\ (X\ x,\ Y\ x))$ (**is** - $\longleftrightarrow$ - $\wedge$ - $\wedge$ $?S \bigotimes_M ?T = ?J$)
⟨*proof*⟩

lemma (**in** *prob-space*) *distributed-joint-indep*:
assumes S: *sigma-finite-measure* S **and** T: *sigma-finite-measure* T

assumes *X*: *distributed M S X Px* **and** *Y*: *distributed M T Y Py*
assumes *indep*: *indep-var S X T Y*
shows *distributed M* $(S \bigotimes_M T)$ $(\lambda x.\ (X\ x,\ Y\ x))$ $(\lambda(x, y).\ Px\ x * Py\ y)$
⟨*proof*⟩

lemma (**in** *prob-space*) *indep-vars-nn-integral*:
assumes *I*: *finite I indep-vars* (λ-. *borel*) $X\ I\ \bigwedge i\ \omega.\ i \in I \Longrightarrow 0 \leq X\ i\ \omega$
shows $(\int^+\omega.\ (\prod i{\in}I.\ X\ i\ \omega)\ \partial M) = (\prod i{\in}I.\ \int^+\omega.\ X\ i\ \omega\ \partial M)$
⟨*proof*⟩

lemma (**in** *prob-space*)
fixes *X* :: *'i* ⇒ *'a* ⇒ *'b*::{*real-normed-field*, *banach*, *second-countable-topology*}
assumes *I*: *finite I indep-vars* (λ-. *borel*) $X\ I\ \bigwedge i.\ i \in I \Longrightarrow$ *integrable M* (*X i*)
shows *indep-vars-lebesgue-integral*: $(\int \omega.\ (\prod i{\in}I.\ X\ i\ \omega)\ \partial M) = (\prod i{\in}I.\ \int \omega.\ X\ i\ \omega\ \partial M)$ (**is** *?eq*)
and *indep-vars-integrable*: *integrable M* $(\lambda\omega.\ (\prod i{\in}I.\ X\ i\ \omega))$ (**is** *?int*)
⟨*proof*⟩

lemma (**in** *prob-space*)
fixes *X1 X2* :: *'a* ⇒ *'b*::{*real-normed-field*, *banach*, *second-countable-topology*}
assumes *indep-var borel X1 borel X2 integrable M X1 integrable M X2*
shows *indep-var-lebesgue-integral*: $(\int \omega.\ X1\ \omega * X2\ \omega\ \partial M) = (\int \omega.\ X1\ \omega\ \partial M) * (\int \omega.\ X2\ \omega\ \partial M)$ (**is** *?eq*)
and *indep-var-integrable*: *integrable M* $(\lambda\omega.\ X1\ \omega * X2\ \omega)$ (**is** *?int*)
⟨*proof*⟩

end

39 Convolution Measure

theory *Convolution*
imports *Independent-Family*
begin

lemma (**in** *finite-measure*) *sigma-finite-measure*: *sigma-finite-measure M*
⟨*proof*⟩

definition *convolution* :: (*'a* :: *ordered-euclidean-space*) *measure* ⇒ *'a measure* ⇒ *'a measure* (**infix** ⋆ *50*) **where**
convolution M N = *distr* $(M \bigotimes_M N)$ *borel* $(\lambda(x, y).\ x + y)$

lemma
shows *space-convolution*[*simp*]: *space* (*convolution M N*) = *space borel*
and *sets-convolution*[*simp*]: *sets* (*convolution M N*) = *sets borel*
and *measurable-convolution1*[*simp*]: *measurable A* (*convolution M N*) = *measurable A borel*
and *measurable-convolution2*[*simp*]: *measurable* (*convolution M N*) *B* = *measurable borel B*
⟨*proof*⟩

lemma *nn-integral-convolution*:
assumes *finite-measure M finite-measure N*
assumes [*measurable-cong*]: *sets N = sets borel sets M = sets borel*
assumes [*measurable*]: $f \in$ *borel-measurable borel*
shows $(\int^{+} x.\ f\ x\ \partial convolution\ M\ N) = (\int^{+} x.\ \int^{+} y.\ f\ (x + y)\ \partial N\ \partial M)$
⟨*proof*⟩

lemma *convolution-emeasure*:
assumes $A \in$ *sets borel finite-measure M finite-measure N*
assumes [*simp*]: *sets N = sets borel sets M = sets borel*
assumes [*simp*]: *space M = space N space N = space borel*
shows *emeasure* $(M \star N)\ A = \int^{+} x.\ (emeasure\ N\ \{a.\ a + x \in A\})\ \partial M$
⟨*proof*⟩

lemma *convolution-emeasure'*:
assumes [*simp*]:$A \in$ *sets borel*
assumes [*simp*]: *finite-measure M finite-measure N*
assumes [*simp*]: *sets N = sets borel sets M = sets borel*
shows *emeasure* $(M \star N)\ A = \int^{+} x.\ \int^{+} y.\ (indicator\ A\ (x + y))\ \partial N\ \partial M$
⟨*proof*⟩

lemma *convolution-finite*:
assumes [*simp*]: *finite-measure M finite-measure N*
assumes [*measurable-cong*]: *sets N = sets borel sets M = sets borel*
shows *finite-measure* $(M \star N)$
⟨*proof*⟩

lemma *convolution-emeasure-3*:
assumes [*simp*, *measurable*]: $A \in$ *sets borel*
assumes [*simp*]: *finite-measure M finite-measure N finite-measure L*
assumes [*simp*]: *sets N = sets borel sets M = sets borel sets L = sets borel*
shows *emeasure* $(L \star (M \star N))\ A = \int^{+} x.\ \int^{+} y.\ \int^{+} z.\ indicator\ A\ (x + y + z)\ \partial N\ \partial M\ \partial L$
⟨*proof*⟩

lemma *convolution-emeasure-3'*:
assumes [*simp*, *measurable*]:$A \in$ *sets borel*
assumes [*simp*]: *finite-measure M finite-measure N finite-measure L*
assumes [*measurable-cong*, *simp*]: *sets N = sets borel sets M = sets borel sets L = sets borel*
shows *emeasure* $((L \star M) \star N)\ A = \int^{+} x.\ \int^{+} y.\ \int^{+} z.\ indicator\ A\ (x + y + z)\ \partial N\ \partial M\ \partial L$
⟨*proof*⟩

lemma *convolution-commutative*:
assumes [*simp*]: *finite-measure M finite-measure N*
assumes [*measurable-cong*, *simp*]: *sets N = sets borel sets M = sets borel*
shows $(M \star N) = (N \star M)$

⟨*proof*⟩

lemma *convolution-associative*:
assumes [*simp*]: *finite-measure M finite-measure N finite-measure L*
assumes [*simp*]: *sets N* = *sets borel sets M* = *sets borel sets L* = *sets borel*
shows $(L \star (M \star N)) = ((L \star M) \star N)$
⟨*proof*⟩

lemma (**in** *prob-space*) *sum-indep-random-variable*:
assumes *ind*: *indep-var borel X borel Y*
assumes [*simp*, *measurable*]: *random-variable borel X*
assumes [*simp*, *measurable*]: *random-variable borel Y*
shows *distr M borel* $(\lambda x.\ X\ x + Y\ x)$ = *convolution* (*distr M borel X*) (*distr M borel Y*)
⟨*proof*⟩

lemma (**in** *prob-space*) *sum-indep-random-variable-lborel*:
assumes *ind*: *indep-var borel X borel Y*
assumes [*simp*, *measurable*]: *random-variable lborel X*
assumes [*simp*, *measurable*]:*random-variable lborel Y*
shows *distr M lborel* $(\lambda x.\ X\ x + Y\ x)$ = *convolution* (*distr M lborel X*) (*distr M lborel Y*)
⟨*proof*⟩

lemma *convolution-density*:
fixes *f g* :: *real* ⇒ *ennreal*
assumes [*measurable*]: *f* ∈ *borel-measurable borel g* ∈ *borel-measurable borel*
assumes [*simp*]:*finite-measure* (*density lborel f*) *finite-measure* (*density lborel g*)
shows *density lborel f* ⋆ *density lborel g* = *density lborel* $(\lambda x.\ \int^{+} y.\ f\ (x - y) * g\ y\ \partial lborel)$
(**is** *?l* = *?r*)
⟨*proof*⟩

lemma (**in** *prob-space*) *distributed-finite-measure-density*:
distributed M N X f ⟹ *finite-measure* (*density N f*)
⟨*proof*⟩

lemma (**in** *prob-space*) *distributed-convolution*:
fixes *f* :: *real* ⇒ -
fixes *g* :: *real* ⇒ -
assumes *indep*: *indep-var borel X borel Y*
assumes *X*: *distributed M lborel X f*
assumes *Y*: *distributed M lborel Y g*
shows *distributed M lborel* $(\lambda x.\ X\ x + Y\ x)$ $(\lambda x.\ \int^{+} y.\ f\ (x - y) * g\ y\ \partial lborel)$
⟨*proof*⟩

lemma *prob-space-convolution-density*:
fixes *f*:: *real* ⇒ -

fixes *g*:: *real* ⇒ -
assumes [*measurable*]: *f*∈ *borel-measurable borel*
assumes [*measurable*]: *g*∈ *borel-measurable borel*
assumes *gt-0*[*simp*]: $\bigwedge x.\ 0 \leq f\ x$ $\bigwedge x.\ 0 \leq g\ x$
assumes *prob-space* (*density lborel f*) (**is** *prob-space ?F*)
assumes *prob-space* (*density lborel g*) (**is** *prob-space ?G*)
shows *prob-space* (*density lborel* ($\lambda x.\int^{+} y.\ f\ (x - y) * g\ y\ \partial lborel$)) (**is** *prob-space ?D*)
⟨*proof*⟩

end

40 Information theory

theory *Information*
imports
Independent-Family
~~/src/HOL/Library/Convex
begin

lemma *log-le*: $1 < a \Longrightarrow 0 < x \Longrightarrow x \leq y \Longrightarrow log\ a\ x \leq log\ a\ y$
⟨*proof*⟩

lemma *log-less*: $1 < a \Longrightarrow 0 < x \Longrightarrow x < y \Longrightarrow log\ a\ x < log\ a\ y$
⟨*proof*⟩

lemma *setsum-cartesian-product'*:
$(\sum x \in A \times B.\ f\ x) = (\sum x \in A.\ setsum\ (\lambda y.\ f\ (x, y))\ B)$
⟨*proof*⟩

lemma *split-pairs*:
$((A, B) = X) \longleftrightarrow (fst\ X = A \wedge snd\ X = B)$ **and**
$(X = (A, B)) \longleftrightarrow (fst\ X = A \wedge snd\ X = B)$ ⟨*proof*⟩

40.1 Information theory

locale *information-space* = *prob-space* +
fixes *b* :: *real* **assumes** *b-gt-1*: $1 < b$

context *information-space*
begin

Introduce some simplification rules for logarithm of base *b*.

lemma *log-neg-const*:
assumes $x \leq 0$
shows $log\ b\ x = log\ b\ 0$
⟨*proof*⟩

lemma *log-mult-eq*:

$log\ b\ (A * B) = (if\ 0 < A * B\ then\ log\ b\ |A| + log\ b\ |B|\ else\ log\ b\ 0)$
⟨*proof*⟩

lemma *log-inverse-eq*:
$log\ b\ (inverse\ B) = (if\ 0 < B\ then\ - log\ b\ B\ else\ log\ b\ 0)$
⟨*proof*⟩

lemma *log-divide-eq*:
$log\ b\ (A\ /\ B) = (if\ 0 < A * B\ then\ log\ b\ |A| - log\ b\ |B|\ else\ log\ b\ 0)$
⟨*proof*⟩

lemmas *log-simps* = *log-mult-eq log-inverse-eq log-divide-eq*

end

40.2 Kullback–Leibler divergence

The Kullback–Leibler divergence is also known as relative entropy or Kullback–Leibler distance.

definition
entropy-density b M N = *log b* ∘ *enn2real* ∘ *RN-deriv M N*

definition
KL-divergence b M N = $integral^L$ *N* (*entropy-density b M N*)

lemma *measurable-entropy-density*[*measurable*]: *entropy-density b M N* ∈ *borel-measurable M*
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *KL-density*:
fixes *f* :: *'a* ⇒ *real*
assumes *1* < *b*
assumes *f*[*measurable*]: *f* ∈ *borel-measurable M* **and** *nn*: *AE x in M*. *0* ≤ *f x*
shows *KL-divergence b M* (*density M f*) = (∫ *x*. *f x* ∗ *log b* (*f x*) ∂*M*)
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *KL-density-density*:
fixes *f g* :: *'a* ⇒ *real*
assumes *1* < *b*
assumes *f*: *f* ∈ *borel-measurable M AE x in M*. *0* ≤ *f x*
assumes *g*: *g* ∈ *borel-measurable M AE x in M*. *0* ≤ *g x*
assumes *ac*: *AE x in M*. *f x* = *0* ⟶ *g x* = *0*
shows *KL-divergence b* (*density M f*) (*density M g*) = (∫ *x*. *g x* ∗ *log b* (*g x* / *f x*) ∂*M*)
⟨*proof*⟩

lemma (**in** *information-space*) *KL-gt-0*:
fixes *D* :: *'a* ⇒ *real*
assumes *prob-space* (*density M D*)

assumes *D*: *D* ∈ *borel-measurable M AE x in M*. *0* ≤ *D x*
assumes *int*: *integrable M* (*λx*. *D x* ∗ *log b* (*D x*))
assumes *A*: *density M D* ≠ *M*
shows *0* < *KL-divergence b M* (*density M D*)
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *KL-same-eq-0*: *KL-divergence b M M* = *0*
⟨*proof*⟩

lemma (**in** *information-space*) *KL-eq-0-iff-eq*:
fixes *D* :: *'a* ⇒ *real*
assumes *prob-space* (*density M D*)
assumes *D*: *D* ∈ *borel-measurable M AE x in M*. *0* ≤ *D x*
assumes *int*: *integrable M* (*λx*. *D x* ∗ *log b* (*D x*))
shows *KL-divergence b M* (*density M D*) = *0* ⟷ *density M D* = *M*
⟨*proof*⟩

lemma (**in** *information-space*) *KL-eq-0-iff-eq-ac*:
fixes *D* :: *'a* ⇒ *real*
assumes *prob-space N*
assumes *ac*: *absolutely-continuous M N sets N* = *sets M*
assumes *int*: *integrable N* (*entropy-density b M N*)
shows *KL-divergence b M N* = *0* ⟷ *N* = *M*
⟨*proof*⟩

lemma (**in** *information-space*) *KL-nonneg*:
assumes *prob-space* (*density M D*)
assumes *D*: *D* ∈ *borel-measurable M AE x in M*. *0* ≤ *D x*
assumes *int*: *integrable M* (*λx*. *D x* ∗ *log b* (*D x*))
shows *0* ≤ *KL-divergence b M* (*density M D*)
⟨*proof*⟩

lemma (**in** *sigma-finite-measure*) *KL-density-density-nonneg*:
fixes *f g* :: *'a* ⇒ *real*
assumes *1* < *b*
assumes *f*: *f* ∈ *borel-measurable M AE x in M*. *0* ≤ *f x prob-space* (*density M f*)
assumes *g*: *g* ∈ *borel-measurable M AE x in M*. *0* ≤ *g x prob-space* (*density M g*)
assumes *ac*: *AE x in M*. *f x* = *0* ⟶ *g x* = *0*
assumes *int*: *integrable M* (*λx*. *g x* ∗ *log b* (*g x* / *f x*))
shows *0* ≤ *KL-divergence b* (*density M f*) (*density M g*)
⟨*proof*⟩

40.3 Finite Entropy

definition (**in** *information-space*) *finite-entropy* :: *'b measure* ⇒ (*'a* ⇒ *'b*) ⇒ (*'b* ⇒ *real*) ⇒ *bool*
where

finite-entropy S X f $\longleftrightarrow$
  distributed M S X f $\wedge$
  integrable S $(\lambda x.\ f\ x * log\ b\ (f\ x))$ $\wedge$
  $(\forall x{\in}space\ S.\ 0 \leq f\ x)$

lemma (**in** *information-space*) *finite-entropy-simple-function*:
assumes *X*: *simple-function M X*
shows *finite-entropy* (*count-space* (*X'space M*)) *X* ($\lambda a.$ *measure M* $\{x \in space\ M.\ X\ x = a\}$)
⟨*proof*⟩

lemma *ac-fst*:
assumes *sigma-finite-measure T*
shows *absolutely-continuous S* (*distr* $(S \bigotimes_M T)$ *S fst*)
⟨*proof*⟩

lemma *ac-snd*:
assumes *sigma-finite-measure T*
shows *absolutely-continuous T* (*distr* $(S \bigotimes_M T)$ *T snd*)
⟨*proof*⟩

lemma *integrable-cong-AE-imp*:
integrable M g $\Longrightarrow f \in$ *borel-measurable M* $\Longrightarrow$ (*AE x in M*. $g\ x = f\ x$) $\Longrightarrow$ *integrable M f*
⟨*proof*⟩

lemma (**in** *information-space*) *finite-entropy-integrable*:
finite-entropy S X Px $\Longrightarrow$ *integrable S* $(\lambda x.\ Px\ x * log\ b\ (Px\ x))$
⟨*proof*⟩

lemma (**in** *information-space*) *finite-entropy-distributed*:
finite-entropy S X Px $\Longrightarrow$ *distributed M S X Px*
⟨*proof*⟩

lemma (**in** *information-space*) *finite-entropy-nn*:
finite-entropy S X Px $\Longrightarrow x \in space\ S \Longrightarrow 0 \leq Px\ x$
⟨*proof*⟩

lemma (**in** *information-space*) *finite-entropy-measurable*:
finite-entropy S X Px $\Longrightarrow Px \in S \rightarrow_M borel$
⟨*proof*⟩

lemma (**in** *information-space*) *subdensity-finite-entropy*:
fixes $g :: 'b \Rightarrow real$ **and** $f :: 'c \Rightarrow real$
assumes *T*: $T \in measurable\ P\ Q$
assumes *f*: *finite-entropy P X f*
assumes *g*: *finite-entropy Q Y g*
assumes *Y*: $Y = T \circ X$
shows *AE x in P*. $g\ (T\ x) = 0 \longrightarrow f\ x = 0$

⟨*proof*⟩

lemma (**in** *information-space*) *finite-entropy-integrable-transform*:
finite-entropy S X Px ⟹ *distributed M T Y Py* ⟹ (⋀*x*. *x* ∈ *space T* ⟹ *0* ≤ *Py x*) ⟹
X = (λ*x*. *f* (*Y x*)) ⟹ *f* ∈ *measurable T S* ⟹ *integrable T* (λ*x*. *Py x* ∗ *log b* (*Px* (*f x*)))
⟨*proof*⟩

40.4 Mutual Information

definition (**in** *prob-space*)
mutual-information b S T X Y =
KL-divergence b (*distr M S X* ⨂$_M$ *distr M T Y*) (*distr M* (*S* ⨂$_M$ *T*) (λ*x*. (*X x*, *Y x*)))

lemma (**in** *information-space*) *mutual-information-indep-vars*:
fixes *S T X Y*
defines *P* ≡ *distr M S X* ⨂$_M$ *distr M T Y*
defines *Q* ≡ *distr M* (*S* ⨂$_M$ *T*) (λ*x*. (*X x*, *Y x*))
shows *indep-var S X T Y* ⟷
(*random-variable S X* ∧ *random-variable T Y* ∧
absolutely-continuous P Q ∧ *integrable Q* (*entropy-density b P Q*) ∧
mutual-information b S T X Y = *0*)
⟨*proof*⟩

abbreviation (**in** *information-space*)
mutual-information-Pow (ℐ′(- ; -′)) **where**
ℐ(*X* ; *Y*) ≡ *mutual-information b* (*count-space* (*X*‘*space M*)) (*count-space* (*Y*‘*space M*)) *X Y*

lemma (**in** *information-space*)
fixes *Pxy* :: ′*b* × ′*c* ⇒ *real* **and** *Px* :: ′*b* ⇒ *real* **and** *Py* :: ′*c* ⇒ *real*
assumes *S*: *sigma-finite-measure S* **and** *T*: *sigma-finite-measure T*
assumes *Fx*: *finite-entropy S X Px* **and** *Fy*: *finite-entropy T Y Py*
assumes *Fxy*: *finite-entropy* (*S* ⨂$_M$ *T*) (λ*x*. (*X x*, *Y x*)) *Pxy*
defines *f* ≡ λ*x*. *Pxy x* ∗ *log b* (*Pxy x* / (*Px* (*fst x*) ∗ *Py* (*snd x*)))
shows *mutual-information-distr′*: *mutual-information b S T X Y* = *integral*^{L} (*S* ⨂$_M$ *T*) *f* (**is** *?M* = *?R*)
and *mutual-information-nonneg′*: *0* ≤ *mutual-information b S T X Y*
⟨*proof*⟩

lemma (**in** *information-space*)
fixes *Pxy* :: ′*b* × ′*c* ⇒ *real* **and** *Px* :: ′*b* ⇒ *real* **and** *Py* :: ′*c* ⇒ *real*
assumes *sigma-finite-measure S sigma-finite-measure T*
assumes *Px*: *distributed M S X Px* **and** *Px-nn*: ⋀*x*. *x* ∈ *space S* ⟹ *0* ≤ *Px x*
and *Py*: *distributed M T Y Py* **and** *Py-nn*: ⋀*y*. *y* ∈ *space T* ⟹ *0* ≤ *Py y*
and *Pxy*: *distributed M* (*S* ⨂$_M$ *T*) (λ*x*. (*X x*, *Y x*)) *Pxy*
and *Pxy-nn*: ⋀*x y*. *x* ∈ *space S* ⟹ *y* ∈ *space T* ⟹ *0* ≤ *Pxy* (*x*, *y*)

defines $f \equiv \lambda x.\ Pxy\ x * log\ b\ (Pxy\ x\ /\ (Px\ (fst\ x) * Py\ (snd\ x)))$
shows *mutual-information-distr*: $mutual\text{-}information\ b\ S\ T\ X\ Y = integral^L\ (S \bigotimes_M T)\ f$ (**is** $?M = ?R$)
and *mutual-information-nonneg*: $integrable\ (S \bigotimes_M T)\ f \Longrightarrow 0 \leq mutual\text{-}information\ b\ S\ T\ X\ Y$
⟨*proof*⟩

lemma (**in** *information-space*)
fixes $Pxy :: 'b \times 'c \Rightarrow real$ **and** $Px :: 'b \Rightarrow real$ **and** $Py :: 'c \Rightarrow real$
assumes *sigma-finite-measure S sigma-finite-measure T*
assumes *Px*[*measurable*]: *distributed M S X Px* **and** *Px-nn*: $\bigwedge x.\ x \in space\ S \Longrightarrow 0 \leq Px\ x$
and *Py*[*measurable*]: *distributed M T Y Py* **and** *Py-nn*: $\bigwedge x.\ x \in space\ T \Longrightarrow 0 \leq Py\ x$
and *Pxy*[*measurable*]: $distributed\ M\ (S \bigotimes_M T)\ (\lambda x.\ (X\ x,\ Y\ x))\ Pxy$
and *Pxy-nn*: $\bigwedge x.\ x \in space\ (S \bigotimes_M T) \Longrightarrow 0 \leq Pxy\ x$
assumes *ae*: $AE\ x\ in\ S.\ AE\ y\ in\ T.\ Pxy\ (x,\ y) = Px\ x * Py\ y$
shows *mutual-information-eq-0*: $mutual\text{-}information\ b\ S\ T\ X\ Y = 0$
⟨*proof*⟩

lemma (**in** *information-space*) *mutual-information-simple-distributed*:
assumes *X*: *simple-distributed M X Px* **and** *Y*: *simple-distributed M Y Py*
assumes *XY*: $simple\text{-}distributed\ M\ (\lambda x.\ (X\ x,\ Y\ x))\ Pxy$
shows $\mathcal{I}(X\ ;\ Y) = (\sum (x,\ y) \in (\lambda x.\ (X\ x,\ Y\ x))\text{`}space\ M.\ Pxy\ (x,\ y) * log\ b\ (Pxy\ (x,\ y)\ /\ (Px\ x * Py\ y)))$
⟨*proof*⟩

lemma (**in** *information-space*)
fixes $Pxy :: 'b \times 'c \Rightarrow real$ **and** $Px :: 'b \Rightarrow real$ **and** $Py :: 'c \Rightarrow real$
assumes *Px*: *simple-distributed M X Px* **and** *Py*: *simple-distributed M Y Py*
assumes *Pxy*: $simple\text{-}distributed\ M\ (\lambda x.\ (X\ x,\ Y\ x))\ Pxy$
assumes *ae*: $\forall x \in space\ M.\ Pxy\ (X\ x,\ Y\ x) = Px\ (X\ x) * Py\ (Y\ x)$
shows *mutual-information-eq-0-simple*: $\mathcal{I}(X\ ;\ Y) = 0$
⟨*proof*⟩

40.5 Entropy

definition (**in** *prob-space*) $entropy :: real \Rightarrow 'b\ measure \Rightarrow ('a \Rightarrow 'b) \Rightarrow real$ **where**
$entropy\ b\ S\ X = -\ KL\text{-}divergence\ b\ S\ (distr\ M\ S\ X)$

abbreviation (**in** *information-space*)
entropy-Pow ($\mathcal{H}'(\text{-}')$) **where**
$\mathcal{H}(X) \equiv entropy\ b\ (count\text{-}space\ (X\text{`}space\ M))\ X$

lemma (**in** *prob-space*) *distributed-RN-deriv*:
assumes *X*: *distributed M S X Px*
shows $AE\ x\ in\ S.\ RN\text{-}deriv\ S\ (density\ S\ Px)\ x = Px\ x$
⟨*proof*⟩

lemma (**in** *information-space*)
fixes $X :: 'a \Rightarrow 'b$
assumes $X[measurable]$: *distributed M MX X f* **and** *nn*: $\bigwedge x.\ x \in space\ MX \Longrightarrow 0 \le f\ x$
shows *entropy-distr*: $entropy\ b\ MX\ X = -\ (\int x.\ f\ x * log\ b\ (f\ x)\ \partial MX)$ (**is** *?eq*)
⟨*proof*⟩

lemma (**in** *information-space*) *entropy-le*:
fixes $Px :: 'b \Rightarrow real$ **and** $MX :: 'b\ measure$
assumes $X[measurable]$: *distributed M MX X Px* **and** *Px-nn*[*simp*]: $\bigwedge x.\ x \in space\ MX \Longrightarrow 0 \le Px\ x$
and *fin*: $emeasure\ MX\ \{x \in space\ MX.\ Px\ x \neq 0\} \neq top$
and *int*: $integrable\ MX\ (\lambda x.\ -\ Px\ x * log\ b\ (Px\ x))$
shows $entropy\ b\ MX\ X \le log\ b\ (measure\ MX\ \{x \in space\ MX.\ Px\ x \neq 0\})$
⟨*proof*⟩

lemma (**in** *information-space*) *entropy-le-space*:
fixes $Px :: 'b \Rightarrow real$ **and** $MX :: 'b\ measure$
assumes X: *distributed M MX X Px* **and** *Px-nn*[*simp*]: $\bigwedge x.\ x \in space\ MX \Longrightarrow 0 \le Px\ x$
and *fin*: *finite-measure MX*
and *int*: $integrable\ MX\ (\lambda x.\ -\ Px\ x * log\ b\ (Px\ x))$
shows $entropy\ b\ MX\ X \le log\ b\ (measure\ MX\ (space\ MX))$
⟨*proof*⟩

lemma (**in** *information-space*) *entropy-uniform*:
assumes X: $distributed\ M\ MX\ X\ (\lambda x.\ indicator\ A\ x\ /\ measure\ MX\ A)$ (**is** *distributed - - - ?f*)
shows $entropy\ b\ MX\ X = log\ b\ (measure\ MX\ A)$
⟨*proof*⟩

lemma (**in** *information-space*) *entropy-simple-distributed*:
$simple\text{-}distributed\ M\ X\ f \Longrightarrow \mathcal{H}(X) = -\ (\sum x \in X`space\ M.\ f\ x * log\ b\ (f\ x))$
⟨*proof*⟩

lemma (**in** *information-space*) *entropy-le-card-not-0*:
assumes X: *simple-distributed M X f*
shows $\mathcal{H}(X) \le log\ b\ (card\ (X\ `\ space\ M \cap \{x.\ f\ x \neq 0\}))$
⟨*proof*⟩

lemma (**in** *information-space*) *entropy-le-card*:
assumes X: *simple-distributed M X f*
shows $\mathcal{H}(X) \le log\ b\ (real\ (card\ (X\ `\ space\ M)))$
⟨*proof*⟩

40.6 Conditional Mutual Information

definition (**in** *prob-space*)
conditional-mutual-information b MX MY MZ X Y Z ≡

mutual-information b MX (*MY* $\bigotimes_M$ *MZ*) *X* (λx. (*Y x*, *Z x*)) −
mutual-information b MX MZ X Z

abbreviation (**in** *information-space*)
conditional-mutual-information-Pow ($\mathcal{I}'$(- ; - | - ')) **where**
$\mathcal{I}$(*X* ; *Y* | *Z*) ≡ *conditional-mutual-information b*
(*count-space* (*X* ' *space M*)) (*count-space* (*Y* ' *space M*)) (*count-space* (*Z* ' *space M*)) *X Y Z*

lemma (**in** *information-space*)
assumes *S*: *sigma-finite-measure S* **and** *T*: *sigma-finite-measure T* **and** *P*: *sigma-finite-measure P*
assumes *Px*[*measurable*]: *distributed M S X Px*
and *Px-nn*[*simp*]: ⋀*x*. *x* ∈ *space S* ⟹ *0* ≤ *Px x*
assumes *Pz*[*measurable*]: *distributed M P Z Pz*
and *Pz-nn*[*simp*]: ⋀*z*. *z* ∈ *space P* ⟹ *0* ≤ *Pz z*
assumes *Pyz*[*measurable*]: *distributed M* (*T* $\bigotimes_M$ *P*) (λx. (*Y x*, *Z x*)) *Pyz*
and *Pyz-nn*[*simp*]: ⋀*y z*. *y* ∈ *space T* ⟹ *z* ∈ *space P* ⟹ *0* ≤ *Pyz* (*y*, *z*)
assumes *Pxz*[*measurable*]: *distributed M* (*S* $\bigotimes_M$ *P*) (λx. (*X x*, *Z x*)) *Pxz*
and *Pxz-nn*[*simp*]: ⋀*x z*. *x* ∈ *space S* ⟹ *z* ∈ *space P* ⟹ *0* ≤ *Pxz* (*x*, *z*)
assumes *Pxyz*[*measurable*]: *distributed M* (*S* $\bigotimes_M$ *T* $\bigotimes_M$ *P*) (λx. (*X x*, *Y x*, *Z x*)) *Pxyz*
and *Pxyz-nn*[*simp*]: ⋀*x y z*. *x* ∈ *space S* ⟹ *y* ∈ *space T* ⟹ *z* ∈ *space P* ⟹ *0* ≤ *Pxyz* (*x*, *y*, *z*)
assumes *I1*: *integrable* (*S* $\bigotimes_M$ *T* $\bigotimes_M$ *P*) (λ(*x*, *y*, *z*). *Pxyz* (*x*, *y*, *z*) ∗ *log b* (*Pxyz* (*x*, *y*, *z*) / (*Px x* ∗ *Pyz* (*y*, *z*))))
assumes *I2*: *integrable* (*S* $\bigotimes_M$ *T* $\bigotimes_M$ *P*) (λ(*x*, *y*, *z*). *Pxyz* (*x*, *y*, *z*) ∗ *log b* (*Pxz* (*x*, *z*) / (*Px x* ∗ *Pz z*)))
shows *conditional-mutual-information-generic-eq*: *conditional-mutual-information b S T P X Y Z*
= (∫ (*x*, *y*, *z*). *Pxyz* (*x*, *y*, *z*) ∗ *log b* (*Pxyz* (*x*, *y*, *z*) / (*Pxz* (*x*, *z*) ∗ (*Pyz* (*y*,*z*) / *Pz z*))) ∂(*S* $\bigotimes_M$ *T* $\bigotimes_M$ *P*)) (**is** *?eq*)
and *conditional-mutual-information-generic-nonneg*: *0* ≤ *conditional-mutual-information b S T P X Y Z* (**is** *?nonneg*)
⟨*proof*⟩

lemma (**in** *information-space*)
fixes *Px* :: - ⇒ *real*
assumes *S*: *sigma-finite-measure S* **and** *T*: *sigma-finite-measure T* **and** *P*: *sigma-finite-measure P*
assumes *Fx*: *finite-entropy S X Px*
assumes *Fz*: *finite-entropy P Z Pz*
assumes *Fyz*: *finite-entropy* (*T* $\bigotimes_M$ *P*) (λx. (*Y x*, *Z x*)) *Pyz*
assumes *Fxz*: *finite-entropy* (*S* $\bigotimes_M$ *P*) (λx. (*X x*, *Z x*)) *Pxz*
assumes *Fxyz*: *finite-entropy* (*S* $\bigotimes_M$ *T* $\bigotimes_M$ *P*) (λx. (*X x*, *Y x*, *Z x*)) *Pxyz*
shows *conditional-mutual-information-generic-eq'*: *conditional-mutual-information b S T P X Y Z*
= (∫ (*x*, *y*, *z*). *Pxyz* (*x*, *y*, *z*) ∗ *log b* (*Pxyz* (*x*, *y*, *z*) / (*Pxz* (*x*, *z*) ∗ (*Pyz* (*y*,*z*) / *Pz z*))) ∂(*S* $\bigotimes_M$ *T* $\bigotimes_M$ *P*)) (**is** *?eq*)

and *conditional-mutual-information-generic-nonneg'*: $0 \leq$ *conditional-mutual-information* *b S T P X Y Z* (**is** *?nonneg*)
⟨*proof*⟩

lemma (**in** *information-space*) *conditional-mutual-information-eq*:
assumes *Pz*: *simple-distributed M Z Pz*
assumes *Pyz*: *simple-distributed M* $(\lambda x.\ (Y\ x,\ Z\ x))$ *Pyz*
assumes *Pxz*: *simple-distributed M* $(\lambda x.\ (X\ x,\ Z\ x))$ *Pxz*
assumes *Pxyz*: *simple-distributed M* $(\lambda x.\ (X\ x,\ Y\ x,\ Z\ x))$ *Pxyz*
shows $\mathcal{I}(X\ ;\ Y\ |\ Z) =$
$(\sum (x,\ y,\ z) \in (\lambda x.\ (X\ x,\ Y\ x,\ Z\ x))$ '*space M*. *Pxyz* $(x,\ y,\ z) *$ *log b* (*Pxyz* $(x,\ y,\ z)$ / (*Pxz* $(x,\ z) *$ (*Pyz* (y,z) / *Pz z*))))
⟨*proof*⟩

lemma (**in** *information-space*) *conditional-mutual-information-nonneg*:
assumes *X*: *simple-function M X* **and** *Y*: *simple-function M Y* **and** *Z*: *simple-function M Z*
shows $0 \leq \mathcal{I}(X\ ;\ Y\ |\ Z)$
⟨*proof*⟩

40.7 Conditional Entropy

definition (**in** *prob-space*)
conditional-entropy b S T X Y $= - (\int (x,\ y).$ *log b* (*enn2real* (*RN-deriv* $(S \bigotimes_M T)$ (*distr M* $(S \bigotimes_M T)$ $(\lambda x.\ (X\ x,\ Y\ x)))$ $(x,\ y))$ /
enn2real (*RN-deriv T* (*distr M T Y*) *y*)) ∂*distr M* $(S \bigotimes_M T)$ $(\lambda x.\ (X\ x,\ Y\ x)))$

abbreviation (**in** *information-space*)
conditional-entropy-Pow ($\mathcal{H}$'(- | -')) **where**
$\mathcal{H}(X\ |\ Y) \equiv$ *conditional-entropy b* (*count-space* (*X*'*space M*)) (*count-space* (*Y*'*space M*)) *X Y*

lemma (**in** *information-space*) *conditional-entropy-generic-eq*:
fixes *Pxy* :: - ⇒ *real* **and** *Py* :: '*c* ⇒ *real*
assumes *S*: *sigma-finite-measure S* **and** *T*: *sigma-finite-measure T*
assumes *Py*[*measurable*]: *distributed M T Y Py* **and** *Py-nn*[*simp*]: $\bigwedge x.\ x \in$ *space T* $\Longrightarrow 0 \leq$ *Py x*
assumes *Pxy*[*measurable*]: *distributed M* $(S \bigotimes_M T)$ $(\lambda x.\ (X\ x,\ Y\ x))$ *Pxy*
and *Pxy-nn*[*simp*]: $\bigwedge x\ y.\ x \in$ *space S* $\Longrightarrow y \in$ *space T* $\Longrightarrow 0 \leq$ *Pxy* $(x,\ y)$
shows *conditional-entropy b S T X Y* $= - (\int (x,\ y).$ *Pxy* $(x,\ y) *$ *log b* (*Pxy* $(x,\ y)$ / *Py y*) $\partial(S \bigotimes_M T))$
⟨*proof*⟩

lemma (**in** *information-space*) *conditional-entropy-eq-entropy*:
fixes *Px* :: '*b* ⇒ *real* **and** *Py* :: '*c* ⇒ *real*
assumes *S*: *sigma-finite-measure S* **and** *T*: *sigma-finite-measure T*
assumes *Py*[*measurable*]: *distributed M T Y Py*
and *Py-nn*[*simp*]: $\bigwedge x.\ x \in$ *space T* $\Longrightarrow 0 \leq$ *Py x*

assumes *Pxy*[*measurable*]: *distributed M* $(S \bigotimes_M T)$ $(\lambda x.\ (X\ x,\ Y\ x))$ *Pxy*
and *Pxy-nn*[*simp*]: $\bigwedge x\ y.\ x \in$ *space S* $\Longrightarrow y \in$ *space T* $\Longrightarrow 0 \leq$ *Pxy* $(x,\ y)$
assumes *I1*: *integrable* $(S \bigotimes_M T)$ $(\lambda x.$ *Pxy* $x *$ *log b* (*Pxy x*))
assumes *I2*: *integrable* $(S \bigotimes_M T)$ $(\lambda x.$ *Pxy* $x *$ *log b* (*Py* (*snd x*)))
shows *conditional-entropy b S T X Y* = *entropy b* $(S \bigotimes_M T)$ $(\lambda x.\ (X\ x,\ Y\ x))$ − *entropy b T Y*
⟨*proof*⟩

lemma (**in** *information-space*) *conditional-entropy-eq-entropy-simple*:
assumes *X*: *simple-function M X* **and** *Y*: *simple-function M Y*
shows $\mathcal{H}(X \mid Y)$ = *entropy b* (*count-space* (*X'space M*) $\bigotimes_M$ *count-space* (*Y'space M*)) $(\lambda x.\ (X\ x,\ Y\ x)) - \mathcal{H}(Y)$
⟨*proof*⟩

lemma (**in** *information-space*) *conditional-entropy-eq*:
assumes *Y*: *simple-distributed M Y Py*
assumes *XY*: *simple-distributed M* $(\lambda x.\ (X\ x,\ Y\ x))$ *Pxy*
shows $\mathcal{H}(X \mid Y) = -$ ($\sum (x,\ y) \in (\lambda x.\ (X\ x,\ Y\ x))$ ' *space M*. *Pxy* $(x,\ y) *$ *log b* (*Pxy* $(x,\ y)$ / *Py y*))
⟨*proof*⟩

lemma (**in** *information-space*) *conditional-mutual-information-eq-conditional-entropy*:
assumes *X*: *simple-function M X* **and** *Y*: *simple-function M Y*
shows $\mathcal{I}(X\ ;\ X \mid Y) = \mathcal{H}(X \mid Y)$
⟨*proof*⟩

lemma (**in** *information-space*) *conditional-entropy-nonneg*:
assumes *X*: *simple-function M X* **and** *Y*: *simple-function M Y* **shows** $0 \leq \mathcal{H}(X \mid Y)$
⟨*proof*⟩

40.8 Equalities

lemma (**in** *information-space*) *mutual-information-eq-entropy-conditional-entropy-distr*:
fixes *Px* :: *'b* ⇒ *real* **and** *Py* :: *'c* ⇒ *real* **and** *Pxy* :: (*'b* × *'c*) ⇒ *real*
assumes *S*: *sigma-finite-measure S* **and** *T*: *sigma-finite-measure T*
assumes *Px*[*measurable*]: *distributed M S X Px*
and *Px-nn*[*simp*]: $\bigwedge x.\ x \in$ *space S* $\Longrightarrow 0 \leq$ *Px x*
and *Py*[*measurable*]: *distributed M T Y Py*
and *Py-nn*[*simp*]: $\bigwedge x.\ x \in$ *space T* $\Longrightarrow 0 \leq$ *Py x*
and *Pxy*[*measurable*]: *distributed M* $(S \bigotimes_M T)$ $(\lambda x.\ (X\ x,\ Y\ x))$ *Pxy*
and *Pxy-nn*[*simp*]: $\bigwedge x\ y.\ x \in$ *space S* $\Longrightarrow y \in$ *space T* $\Longrightarrow 0 \leq$ *Pxy* $(x,\ y)$
assumes *Ix*: *integrable*$(S \bigotimes_M T)$ $(\lambda x.$ *Pxy* $x *$ *log b* (*Px* (*fst x*)))
assumes *Iy*: *integrable*$(S \bigotimes_M T)$ $(\lambda x.$ *Pxy* $x *$ *log b* (*Py* (*snd x*)))
assumes *Ixy*: *integrable*$(S \bigotimes_M T)$ $(\lambda x.$ *Pxy* $x *$ *log b* (*Pxy x*))
shows *mutual-information b S T X Y* = *entropy b S X* + *entropy b T Y* − *entropy b* $(S \bigotimes_M T)$ $(\lambda x.\ (X\ x,\ Y\ x))$
⟨*proof*⟩

lemma (**in** *information-space*) *mutual-information-eq-entropy-conditional-entropy'*:
fixes *Px* :: *'b* ⇒ *real* **and** *Py* :: *'c* ⇒ *real* **and** *Pxy* :: (*'b* × *'c*) ⇒ *real*
assumes *S*: *sigma-finite-measure S* **and** *T*: *sigma-finite-measure T*
assumes *Px*: *distributed M S X Px* $\bigwedge x.\ x \in space\ S \Longrightarrow 0 \leq Px\ x$
and *Py*: *distributed M T Y Py* $\bigwedge x.\ x \in space\ T \Longrightarrow 0 \leq Py\ x$
assumes *Pxy*: *distributed M* $(S \bigotimes_M T)$ $(\lambda x.\ (X\ x,\ Y\ x))$ *Pxy*
$\bigwedge x.\ x \in space\ (S \bigotimes_M T) \Longrightarrow 0 \leq Pxy\ x$
assumes *Ix*: *integrable*$(S \bigotimes_M T)$ $(\lambda x.\ Pxy\ x * log\ b\ (Px\ (fst\ x)))$
assumes *Iy*: *integrable*$(S \bigotimes_M T)$ $(\lambda x.\ Pxy\ x * log\ b\ (Py\ (snd\ x)))$
assumes *Ixy*: *integrable*$(S \bigotimes_M T)$ $(\lambda x.\ Pxy\ x * log\ b\ (Pxy\ x))$
shows *mutual-information b S T X Y* = *entropy b S X* − *conditional-entropy b S T X Y*
⟨*proof*⟩

lemma (**in** *information-space*) *mutual-information-eq-entropy-conditional-entropy*:
assumes *sf-X*: *simple-function M X* **and** *sf-Y*: *simple-function M Y*
shows $\mathcal{I}(X\ ;\ Y) = \mathcal{H}(X) - \mathcal{H}(X \mid Y)$
⟨*proof*⟩

lemma (**in** *information-space*) *mutual-information-nonneg-simple*:
assumes *sf-X*: *simple-function M X* **and** *sf-Y*: *simple-function M Y*
shows $0 \leq \mathcal{I}(X\ ;\ Y)$
⟨*proof*⟩

lemma (**in** *information-space*) *conditional-entropy-less-eq-entropy*:
assumes *X*: *simple-function M X* **and** *Z*: *simple-function M Z*
shows $\mathcal{H}(X \mid Z) \leq \mathcal{H}(X)$
⟨*proof*⟩

lemma (**in** *information-space*)
fixes *Px* :: *'b* ⇒ *real* **and** *Py* :: *'c* ⇒ *real* **and** *Pxy* :: (*'b* × *'c*) ⇒ *real*
assumes *S*: *sigma-finite-measure S* **and** *T*: *sigma-finite-measure T*
assumes *Px*: *finite-entropy S X Px* **and** *Py*: *finite-entropy T Y Py*
assumes *Pxy*: *finite-entropy* $(S \bigotimes_M T)$ $(\lambda x.\ (X\ x,\ Y\ x))$ *Pxy*
shows *conditional-entropy b S T X Y* ≤ *entropy b S X*
⟨*proof*⟩

lemma (**in** *information-space*) *entropy-chain-rule*:
assumes *X*: *simple-function M X* **and** *Y*: *simple-function M Y*
shows $\mathcal{H}(\lambda x.\ (X\ x,\ Y\ x)) = \mathcal{H}(X) + \mathcal{H}(Y|X)$
⟨*proof*⟩

lemma (**in** *information-space*) *entropy-partition*:
assumes *X*: *simple-function M X*
shows $\mathcal{H}(X) = \mathcal{H}(f \circ X) + \mathcal{H}(X|f \circ X)$
⟨*proof*⟩

corollary (**in** *information-space*) *entropy-data-processing*:
assumes *X*: *simple-function M X* **shows** $\mathcal{H}(f \circ X) \leq \mathcal{H}(X)$

⟨*proof*⟩

corollary (**in** *information-space*) *entropy-of-inj*:
assumes *X*: *simple-function M X* **and** *inj*: *inj-on f* (*X'space M*)
shows $\mathcal{H}(f \circ X) = \mathcal{H}(X)$
⟨*proof*⟩

end

41 Properties of Various Distributions

theory *Distributions*
imports *Convolution Information*
begin

lemma (**in** *prob-space*) *distributed-affine*:
fixes *f* :: *real* ⇒ *ennreal*
assumes *f*: *distributed M lborel X f*
assumes *c*: $c \neq 0$
shows *distributed M lborel* ($\lambda x.\ t + c * X\ x$) ($\lambda x.\ f\ ((x - t)\ /\ c)\ /\ |c|$)
⟨*proof*⟩

lemma (**in** *prob-space*) *distributed-affineI*:
fixes *f* :: *real* ⇒ *ennreal* **and** *c* :: *real*
assumes *f*: *distributed M lborel* ($\lambda x.\ (X\ x - t)\ /\ c$) ($\lambda x.\ |c| * f\ (x * c + t)$)
assumes *c*: $c \neq 0$
shows *distributed M lborel X f*
⟨*proof*⟩

lemma (**in** *prob-space*) *distributed-AE2*:
assumes [*measurable*]: *distributed M N X f Measurable.pred N P*
shows (*AE x in M. P* (*X x*)) ⟷ (*AE x in N.* $0 < f\ x \longrightarrow P\ x$)
⟨*proof*⟩

41.1 Erlang

lemma *nn-intergal-power-times-exp-Icc*:
assumes [*arith*]: $0 \leq a$
shows ($\int^{+} x.$ *ennreal* ($x\hat{}k * exp\ (-x)$) * *indicator* {*0 .. a*} *x* ∂*lborel*) =
(*1* − ($\sum n \leq k.$ ($a\hat{}n * exp\ (-a)$) / *fact n*)) * *fact k* (**is** *?I* = -)
⟨*proof*⟩

lemma *nn-intergal-power-times-exp-Ici*:
shows ($\int^{+} x.$ *ennreal* ($x\hat{}k * exp\ (-x)$) * *indicator* {*0 ..*} *x* ∂*lborel*) = *real-of-nat* (*fact k*)
⟨*proof*⟩

definition *erlang-density* :: *nat* ⇒ *real* ⇒ *real* ⇒ *real* **where**
erlang-density k l x = (*if* $x < 0$ *then 0 else* ($l\hat{}(Suc\ k) * x\hat{}k * exp\ (-\ l * x)$) /

fact k)

definition *erlang-CDF* :: *nat* ⇒ *real* ⇒ *real* ⇒ *real* **where**
erlang-CDF k l x = (*if x* < *0 then 0 else 1* − (∑ *n*≤*k*. ((*l* ∗ *x*) ˆ*n* ∗ *exp* (− *l* ∗ *x*) / *fact n*)))

lemma *erlang-density-nonneg*[*simp*]: *0* ≤ *l* ⟹ *0* ≤ *erlang-density k l x*
⟨*proof*⟩

lemma *borel-measurable-erlang-density*[*measurable*]: *erlang-density k l* ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *erlang-CDF-transform*: *0* < *l* ⟹ *erlang-CDF k l a* = *erlang-CDF k 1* (*l* ∗ *a*)
⟨*proof*⟩

lemma *erlang-CDF-nonneg*[*simp*]: **assumes** *0* < *l* **shows** *0* ≤ *erlang-CDF k l x*
⟨*proof*⟩

lemma *nn-integral-erlang-density*:
assumes [*arith*]: *0* < *l*
shows (∫⁺ *x*. *ennreal* (*erlang-density k l x*) ∗ *indicator* {.. *a*} *x* ∂*lborel*) = *erlang-CDF k l a*
⟨*proof*⟩

lemma *emeasure-erlang-density*:
0 < *l* ⟹ *emeasure* (*density lborel* (*erlang-density k l*)) {.. *a*} = *erlang-CDF k l a*
⟨*proof*⟩

lemma *nn-integral-erlang-ith-moment*:
fixes *k i* :: *nat* **and** *l* :: *real*
assumes [*arith*]: *0* < *l*
shows (∫⁺ *x*. *ennreal* (*erlang-density k l x* ∗ *x* ˆ *i*) ∂*lborel*) = *fact* (*k* + *i*) / (*fact k* ∗ *l* ˆ *i*)
⟨*proof*⟩

lemma *prob-space-erlang-density*:
assumes *l*[*arith*]: *0* < *l*
shows *prob-space* (*density lborel* (*erlang-density k l*)) (**is** *prob-space ?D*)
⟨*proof*⟩

lemma (**in** *prob-space*) *erlang-distributed-le*:
assumes *D*: *distributed M lborel X* (*erlang-density k l*)
assumes [*simp*, *arith*]: *0* < *l 0* ≤ *a*
shows 𝒫(*x in M*. *X x* ≤ *a*) = *erlang-CDF k l a*
⟨*proof*⟩

lemma (**in** *prob-space*) *erlang-distributed-gt*:
assumes *D*[*simp*]: *distributed M lborel X* (*erlang-density k l*)
assumes [*arith*]: $0 < l$ $0 \leq a$
shows $\mathcal{P}$(*x in M*. $a < X\ x$) = *1* − (*erlang-CDF k l a*)
⟨*proof*⟩

lemma *erlang-CDF-at0*: *erlang-CDF k l 0* = *0*
⟨*proof*⟩

lemma *erlang-distributedI*:
assumes *X*[*measurable*]: $X \in$ *borel-measurable M* **and** [*arith*]: $0 < l$
and *X-distr*: $\bigwedge a.\ 0 \leq a \Longrightarrow$ *emeasure M* {*x*∈*space M*. $X\ x \leq a$} = *erlang-CDF k l a*
shows *distributed M lborel X* (*erlang-density k l*)
⟨*proof*⟩

lemma (**in** *prob-space*) *erlang-distributed-iff*:
assumes [*arith*]: $0<l$
shows *distributed M lborel X* (*erlang-density k l*) ⟷
($X \in$ *borel-measurable M* ∧ $0 < l$ ∧ (∀ $a \geq 0$. $\mathcal{P}$(*x in M*. $X\ x \leq a$) = *erlang-CDF k l a*))
⟨*proof*⟩

lemma (**in** *prob-space*) *erlang-distributed-mult-const*:
assumes *erlX*: *distributed M lborel X* (*erlang-density k l*)
assumes *a-pos*[*arith*]: $0 < \alpha$ $0 < l$
shows *distributed M lborel* ($\lambda x.\ \alpha * X\ x$) (*erlang-density k* ($l$ / α))
⟨*proof*⟩

lemma (**in** *prob-space*) *has-bochner-integral-erlang-ith-moment*:
fixes *k i* :: *nat* **and** *l* :: *real*
assumes [*arith*]: $0 < l$ **and** *D*: *distributed M lborel X* (*erlang-density k l*)
shows *has-bochner-integral M* ($\lambda x.\ X\ x$ ^ i) (*fact* ($k + i$) / (*fact k* ∗ l ^ i))
⟨*proof*⟩

lemma (**in** *prob-space*) *erlang-ith-moment-integrable*:
$0 < l \Longrightarrow$ *distributed M lborel X* (*erlang-density k l*) $\Longrightarrow$ *integrable M* ($\lambda x.\ X\ x$ ^ i)
⟨*proof*⟩

lemma (**in** *prob-space*) *erlang-ith-moment*:
$0 < l \Longrightarrow$ *distributed M lborel X* (*erlang-density k l*) $\Longrightarrow$
expectation ($\lambda x.\ X\ x$ ^ i) = *fact* ($k + i$) / (*fact k* ∗ l ^ i)
⟨*proof*⟩

lemma (**in** *prob-space*) *erlang-distributed-variance*:
assumes [*arith*]: $0 < l$ **and** *distributed M lborel X* (*erlang-density k l*)
shows *variance X* = ($k + 1$) / l^2
⟨*proof*⟩

41.2 Exponential distribution

abbreviation *exponential-density* :: *real* ⇒ *real* ⇒ *real* **where**
exponential-density ≡ *erlang-density 0*

lemma *exponential-density-def*:
exponential-density l x = (*if x* < *0 then 0 else l* ∗ *exp* (− *x* ∗ *l*))
⟨*proof*⟩

lemma *erlang-CDF-0*: *erlang-CDF 0 l a* = (*if 0* ≤ *a then 1* − *exp* (− *l* ∗ *a*) *else 0*)
⟨*proof*⟩

lemma *prob-space-exponential-density*: *0* < *l* ⟹ *prob-space* (*density lborel* (*exponential-density l*))
⟨*proof*⟩

lemma (**in** *prob-space*) *exponential-distributedD-le*:
assumes *D*: *distributed M lborel X* (*exponential-density l*) **and** *a*: *0* ≤ *a* **and** *l*: *0* < *l*
shows 𝒫(*x in M*. *X x* ≤ *a*) = *1* − *exp* (− *a* ∗ *l*)
⟨*proof*⟩

lemma (**in** *prob-space*) *exponential-distributedD-gt*:
assumes *D*: *distributed M lborel X* (*exponential-density l*) **and** *a*: *0* ≤ *a* **and** *l*: *0* < *l*
shows 𝒫(*x in M*. *a* < *X x*) = *exp* (− *a* ∗ *l*)
⟨*proof*⟩

lemma (**in** *prob-space*) *exponential-distributed-memoryless*:
assumes *D*: *distributed M lborel X* (*exponential-density l*) **and** *a*: *0* ≤ *a* **and** *l*: *0* < *l* **and** *t*: *0* ≤ *t*
shows 𝒫(*x in M*. *a* + *t* < *X x* | *a* < *X x*) = 𝒫(*x in M*. *t* < *X x*)
⟨*proof*⟩

lemma *exponential-distributedI*:
assumes *X*[*measurable*]: *X* ∈ *borel-measurable M* **and** [*arith*]: *0* < *l*
and *X-distr*: ⋀*a*. *0* ≤ *a* ⟹ *emeasure M* {*x*∈*space M*. *X x* ≤ *a*} = *1* − *exp* (− *a* ∗ *l*)
shows *distributed M lborel X* (*exponential-density l*)
⟨*proof*⟩

lemma (**in** *prob-space*) *exponential-distributed-iff*:
assumes *0* < *l*
shows *distributed M lborel X* (*exponential-density l*) ⟷
(*X* ∈ *borel-measurable M* ∧ (∀ *a*≥*0*. 𝒫(*x in M*. *X x* ≤ *a*) = *1* − *exp* (− *a* ∗ *l*)))
⟨*proof*⟩

lemma (**in** *prob-space*) *exponential-distributed-expectation*:
$0 < l \Longrightarrow$ *distributed M lborel X* (*exponential-density l*) $\Longrightarrow$ *expectation X* $= 1 / l$
⟨*proof*⟩

lemma *exponential-density-nonneg*: $0 < l \Longrightarrow 0 \leq$ *exponential-density l x*
⟨*proof*⟩

lemma (**in** *prob-space*) *exponential-distributed-min*:
assumes $0 < l$ $0 < u$
assumes *expX*: *distributed M lborel X* (*exponential-density l*)
assumes *expY*: *distributed M lborel Y* (*exponential-density u*)
assumes *ind*: *indep-var borel X borel Y*
shows *distributed M lborel* ($\lambda x.$ *min* (X x) (Y x)) (*exponential-density* ($l + u$))
⟨*proof*⟩

lemma (**in** *prob-space*) *exponential-distributed-Min*:
assumes *finI*: *finite I*
assumes *A*: $I \neq \{\}$
assumes *l*: $\bigwedge i.\ i \in I \Longrightarrow 0 < l\ i$
assumes *expX*: $\bigwedge i.\ i \in I \Longrightarrow$ *distributed M lborel* (X i) (*exponential-density* (l i))
assumes *ind*: *indep-vars* ($\lambda i.$ *borel*) X I
shows *distributed M lborel* ($\lambda x.$ *Min* (($\lambda i.$ X i x)'I)) (*exponential-density* ($\sum i{\in}I.$ l i))
⟨*proof*⟩

lemma (**in** *prob-space*) *exponential-distributed-variance*:
$0 < l \Longrightarrow$ *distributed M lborel X* (*exponential-density l*) $\Longrightarrow$ *variance X* $= 1 / l^2$
⟨*proof*⟩

lemma *nn-integral-zero'*: *AE x in M*. $f\ x = 0 \Longrightarrow (\int^+ x.\ f\ x\ \partial M) = 0$
⟨*proof*⟩

lemma *convolution-erlang-density*:
fixes k_1 k_2 :: *nat*
assumes [*simp*, *arith*]: $0 < l$
shows ($\lambda x.$ $\int^+ y.$ *ennreal* (*erlang-density* k_1 l ($x - y$)) $*$ *ennreal* (*erlang-density* k_2 l y) ∂*lborel*) =
(*erlang-density* (*Suc* k_1 + *Suc* k_2 − 1) l)
(**is** *?LHS = ?RHS*)
⟨*proof*⟩

lemma (**in** *prob-space*) *sum-indep-erlang*:
assumes *indep*: *indep-var borel X borel Y*
assumes [*simp*, *arith*]: $0 < l$
assumes *erlX*: *distributed M lborel X* (*erlang-density* k_1 l)
assumes *erlY*: *distributed M lborel Y* (*erlang-density* k_2 l)

shows *distributed M lborel* (λx. *X x* + *Y x*) (*erlang-density* (*Suc* k_1 + *Suc* k_2 − *1*) *l*)
⟨*proof*⟩

lemma (**in** *prob-space*) *erlang-distributed-setsum*:
assumes *finI* : *finite I*
assumes *A*: $I \neq \{\}$
assumes [*simp*, *arith*]: $0 < l$
assumes *expX*: $\bigwedge i.\ i \in I \Longrightarrow$ *distributed M lborel* (*X i*) (*erlang-density* (*k i*) *l*)
assumes *ind*: *indep-vars* (λi. *borel*) *X I*
shows *distributed M lborel* (λx. $\sum i \in I$. *X i x*) (*erlang-density* (($\sum i \in I$. *Suc* (*k i*)) − *1*) *l*)
⟨*proof*⟩

lemma (**in** *prob-space*) *exponential-distributed-setsum*:
assumes *finI*: *finite I*
assumes *A*: $I \neq \{\}$
assumes *l*: $0 < l$
assumes *expX*: $\bigwedge i.\ i \in I \Longrightarrow$ *distributed M lborel* (*X i*) (*exponential-density l*)
assumes *ind*: *indep-vars* (λi. *borel*) *X I*
shows *distributed M lborel* (λx. $\sum i \in I$. *X i x*) (*erlang-density* ((*card I*) − *1*) *l*)
⟨*proof*⟩

lemma (**in** *information-space*) *entropy-exponential*:
assumes *l*[*simp*, *arith*]: $0 < l$
assumes *D*: *distributed M lborel X* (*exponential-density l*)
shows *entropy b lborel X* = *log b* (*exp 1* / *l*)
⟨*proof*⟩

41.3 Uniform distribution

lemma *uniform-distrI*:
assumes *X*: $X \in$ *measurable M M'*
and *A*: $A \in$ *sets M' emeasure M' A* $\neq \infty$ *emeasure M' A* $\neq 0$
assumes *distr*: $\bigwedge B.\ B \in$ *sets M'* $\Longrightarrow$ *emeasure M* (*X* −‘ *B* ∩ *space M*) = *emeasure M'* (*A* ∩ *B*) / *emeasure M' A*
shows *distr M M' X* = *uniform-measure M' A*
⟨*proof*⟩

lemma *uniform-distrI-borel*:
fixes *A* :: *real set*
assumes *X*[*measurable*]: $X \in$ *borel-measurable M* **and** *A*: *emeasure lborel A* = *ennreal r* $0 < r$
and [*measurable*]: $A \in$ *sets borel*
assumes *distr*: $\bigwedge a$. *emeasure M* {*x*∈*space M*. *X x* ≤ *a*} = *emeasure lborel* (*A* ∩ {.. *a*}) / *r*
shows *distributed M lborel X* (λx. *indicator A x* / *measure lborel A*)
⟨*proof*⟩

lemma (**in** *prob-space*) *uniform-distrI-borel-atLeastAtMost*:
fixes *a b* :: *real*
assumes *X*: $X \in$ *borel-measurable M* **and** $a < b$
assumes *distr*: $\bigwedge t.\ a \leq t \Longrightarrow t \leq b \Longrightarrow \mathcal{P}(x\ in\ M.\ X\ x \leq t) = (t - a) / (b - a)$
shows *distributed M lborel X* (λx. *indicator* {*a..b*} *x* / *measure lborel* {*a..b*})
⟨*proof*⟩

lemma (**in** *prob-space*) *uniform-distributed-measure*:
fixes *a b* :: *real*
assumes *D*: *distributed M lborel X* (λx. *indicator* {*a* .. *b*} *x* / *measure lborel* {*a* .. *b*})
assumes *t*: $a \leq t$ $t \leq b$
shows $\mathcal{P}(x\ in\ M.\ X\ x \leq t) = (t - a) / (b - a)$
⟨*proof*⟩

lemma (**in** *prob-space*) *uniform-distributed-bounds*:
fixes *a b* :: *real*
assumes *D*: *distributed M lborel X* (λx. *indicator* {*a* .. *b*} *x* / *measure lborel* {*a* .. *b*})
shows $a < b$
⟨*proof*⟩

lemma (**in** *prob-space*) *uniform-distributed-iff*:
fixes *a b* :: *real*
shows *distributed M lborel X* (λx. *indicator* {*a..b*} *x* / *measure lborel* {*a..b*}) $\longleftrightarrow$
($X \in$ *borel-measurable M* $\wedge\ a < b \wedge (\forall t \in \{a\ ..\ b\}.\ \mathcal{P}(x\ in\ M.\ X\ x \leq t) = (t - a) / (b - a)))$
⟨*proof*⟩

lemma (**in** *prob-space*) *uniform-distributed-expectation*:
fixes *a b* :: *real*
assumes *D*: *distributed M lborel X* (λx. *indicator* {*a* .. *b*} *x* / *measure lborel* {*a* .. *b*})
shows *expectation X* = (*a* + *b*) / *2*
⟨*proof*⟩

lemma (**in** *prob-space*) *uniform-distributed-variance*:
fixes *a b* :: *real*
assumes *D*: *distributed M lborel X* (λx. *indicator* {*a* .. *b*} *x* / *measure lborel* {*a* .. *b*})
shows *variance X* = $(b - a)^2$ / *12*
⟨*proof*⟩

41.4 Normal distribution

definition *normal-density* :: *real* ⇒ *real* ⇒ *real* ⇒ *real* **where**
normal-density $\mu\ \sigma\ x = 1 / sqrt\ (2 * pi * \sigma^2) * exp\ (-(x - \mu)^2 / (2 * \sigma^2))$

abbreviation *std-normal-density* :: *real* ⇒ *real* **where**
std-normal-density ≡ *normal-density 0 1*

lemma *std-normal-density-def*: *std-normal-density x* = (*1* / *sqrt* (*2* ∗ *pi*)) ∗ *exp* (− x^2 / *2*)
⟨*proof*⟩

lemma *normal-density-nonneg*[*simp*]: *0* ≤ *normal-density μ σ x*
⟨*proof*⟩

lemma *normal-density-pos*: *0* < σ ⟹ *0* < *normal-density μ σ x*
⟨*proof*⟩

lemma *borel-measurable-normal-density*[*measurable*]: *normal-density μ σ* ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *gaussian-moment-0*:
has-bochner-integral lborel (λ*x*. *indicator* {*0*..} *x* $*_R$ *exp* (− x^2)) (*sqrt pi* / *2*)
⟨*proof*⟩

lemma *gaussian-moment-1*:
has-bochner-integral lborel (λ*x*::*real*. *indicator* {*0*..} *x* $*_R$ (*exp* (− x^2) ∗ *x*)) (*1* / *2*)
⟨*proof*⟩

lemma
fixes *k* :: *nat*
shows *gaussian-moment-even-pos*:
has-bochner-integral lborel (λ*x*::*real*. *indicator* {*0*..} *x* $*_R$ (*exp* (−x^2)∗*x*ˆ(*2* ∗ *k*)))
((*sqrt pi* / *2*) ∗ (*fact* (*2* ∗ *k*) / (*2* ˆ (*2* ∗ *k*) ∗ *fact k*)))
(**is** *?even*)
and *gaussian-moment-odd-pos*:
has-bochner-integral lborel (λ*x*::*real*. *indicator* {*0*..} *x* $*_R$ (*exp* (−x^2)∗*x*ˆ(*2* ∗ *k* + *1*))) (*fact k* / *2*)
(**is** *?odd*)
⟨*proof*⟩

context
fixes *k* :: *nat* **and** *μ σ* :: *real* **assumes** [*arith*]: *0* < σ
begin

lemma *normal-moment-even*:
has-bochner-integral lborel (λ*x*. *normal-density μ σ x* ∗ (*x* − *μ*) ˆ (*2* ∗ *k*)) (*fact* (*2* ∗ *k*) / ((*2* / $σ^2$)ˆ*k* ∗ *fact k*))
⟨*proof*⟩

lemma *normal-moment-abs-odd*:
has-bochner-integral lborel (λx. *normal-density* μ σ x $*$ $|x - \mu|$ ^($2 * k + 1$))
(2^k $*$ σ ^($2 * k + 1$) $*$ *fact* k $*$ *sqrt* (2 / *pi*))
⟨*proof*⟩

lemma *normal-moment-odd*:
has-bochner-integral lborel (λx. *normal-density* μ σ x $*$ $(x - \mu)$^($2 * k + 1$)) 0
⟨*proof*⟩

lemma *integral-normal-moment-even*:
$integral^L$ *lborel* (λx. *normal-density* μ σ x $*$ $(x - \mu)$^($2 * k$)) = *fact* ($2 * k$) /
((2 / σ^2)^k $*$ *fact* k)
⟨*proof*⟩

lemma *integral-normal-moment-abs-odd*:
$integral^L$ *lborel* (λx. *normal-density* μ σ x $*$ $|x - \mu|$^($2 * k + 1$)) = 2 ^ k $*$ σ
^ ($2 * k + 1$) $*$ *fact* k $*$ *sqrt* (2 / *pi*)
⟨*proof*⟩

lemma *integral-normal-moment-odd*:
$integral^L$ *lborel* (λx. *normal-density* μ σ x $*$ $(x - \mu)$^($2 * k + 1$)) = 0
⟨*proof*⟩

end

context
fixes σ :: *real*
assumes σ-*pos*[*arith*]: $0 < \sigma$
begin

lemma *normal-moment-nz-1*: *has-bochner-integral lborel* (λx. *normal-density* μ σ
$x * x$) μ
⟨*proof*⟩

lemma *integral-normal-moment-nz-1*:
$integral^L$ *lborel* (λx. *normal-density* μ σ $x * x$) = μ
⟨*proof*⟩

lemma *integrable-normal-moment-nz-1*: *integrable lborel* (λx. *normal-density* μ σ
$x * x$)
⟨*proof*⟩

lemma *integrable-normal-moment*: *integrable lborel* (λx. *normal-density* μ σ x $*$
$(x - \mu)$^k)
⟨*proof*⟩

lemma *integrable-normal-moment-abs*: *integrable lborel* (λx. *normal-density* μ σ x
$*$ $|x - \mu|$^k)

⟨*proof*⟩

lemma *integrable-normal-density*[*simp*, *intro*]: *integrable lborel* (*normal-density* μ σ)
⟨*proof*⟩

lemma *integral-normal-density*[*simp*]: ($\int$ *x*. *normal-density* μ σ *x* ∂*lborel*) = *1*
⟨*proof*⟩

lemma *prob-space-normal-density*:
prob-space (*density lborel* (*normal-density* μ σ))
⟨*proof*⟩

end

context
fixes *k* :: *nat*
begin

lemma *std-normal-moment-even*:
has-bochner-integral lborel (λ*x*. *std-normal-density x* ∗ *x* ^ (*2* ∗ *k*)) (*fact* (*2* ∗ *k*) / (*2*^*k* ∗ *fact k*))
⟨*proof*⟩

lemma *std-normal-moment-abs-odd*:
has-bochner-integral lborel (λ*x*. *std-normal-density x* ∗ |*x*|^(*2* ∗ *k* + *1*)) (*sqrt* (*2*/*pi*) ∗ *2*^*k* ∗ *fact k*)
⟨*proof*⟩

lemma *std-normal-moment-odd*:
has-bochner-integral lborel (λ*x*. *std-normal-density x* ∗ *x*^(*2* ∗ *k* + *1*)) *0*
⟨*proof*⟩

lemma *integral-std-normal-moment-even*:
integral^L *lborel* (λ*x*. *std-normal-density x* ∗ *x*^(*2*∗*k*)) = *fact* (*2* ∗ *k*) / (*2*^*k* ∗ *fact k*)
⟨*proof*⟩

lemma *integral-std-normal-moment-abs-odd*:
integral^L *lborel* (λ*x*. *std-normal-density x* ∗ |*x*|^(*2* ∗ *k* + *1*)) = *sqrt* (*2* / *pi*) ∗ *2*^*k* ∗ *fact k*
⟨*proof*⟩

lemma *integral-std-normal-moment-odd*:
integral^L *lborel* (λ*x*. *std-normal-density x* ∗ *x*^(*2* ∗ *k* + *1*)) = *0*
⟨*proof*⟩

lemma *integrable-std-normal-moment-abs*: *integrable lborel* (λx. *std-normal-density* $x * |x|$ ^k)
⟨*proof*⟩

lemma *integrable-std-normal-moment*: *integrable lborel* (λx. *std-normal-density* x $* x$^k)
⟨*proof*⟩

end

lemma (**in** *prob-space*) *normal-density-affine*:
assumes X: *distributed M lborel X* (*normal-density* μ σ)
assumes [*simp*, *arith*]: $0 < \sigma$ $\alpha \neq 0$
shows *distributed M lborel* (λx. $\beta + \alpha * X\ x$) (*normal-density* ($\beta + \alpha * \mu$) ($|\alpha|$ $* \sigma$))
⟨*proof*⟩

lemma (**in** *prob-space*) *normal-standard-normal-convert*:
assumes *pos-var*[*simp*, *arith*]: $0 < \sigma$
shows *distributed M lborel X* (*normal-density* μ σ) = *distributed M lborel* (λx. ($X\ x - \mu$) / σ) *std-normal-density*
⟨*proof*⟩

lemma *conv-normal-density-zero-mean*:
assumes [*simp*, *arith*]: $0 < \sigma$ $0 < \tau$
shows (λx. $\int^{+} y$. *ennreal* (*normal-density 0* σ ($x - y$) $*$ *normal-density 0* τ y) ∂*lborel*) =
normal-density 0 (*sqrt* ($\sigma^2 + \tau^2$)) (**is** *?LHS* = *?RHS*)
⟨*proof*⟩

lemma *conv-std-normal-density*:
(λx. $\int^{+} y$. *ennreal* (*std-normal-density* ($x - y$) $*$ *std-normal-density* y) ∂*lborel*) =
(*normal-density 0* (*sqrt 2*))
⟨*proof*⟩

lemma (**in** *prob-space*) *sum-indep-normal*:
assumes *indep*: *indep-var borel X borel Y*
assumes *pos-var*[*arith*]: $0 < \sigma$ $0 < \tau$
assumes *normalX*[*simp*]: *distributed M lborel X* (*normal-density* μ σ)
assumes *normalY*[*simp*]: *distributed M lborel Y* (*normal-density* ν τ)
shows *distributed M lborel* (λx. $X\ x + Y\ x$) (*normal-density* ($\mu + \nu$) (*sqrt* (σ^2 $+ \tau^2$)))
⟨*proof*⟩

lemma (**in** *prob-space*) *diff-indep-normal*:
assumes *indep*[*simp*]: *indep-var borel X borel Y*
assumes [*simp*, *arith*]: $0 < \sigma$ $0 < \tau$
assumes *normalX*[*simp*]: *distributed M lborel X* (*normal-density* μ σ)

assumes *normalY*[*simp*]: *distributed M lborel Y* (*normal-density* ν τ)
shows *distributed M lborel* ($\lambda x.$ *X x* $-$ *Y x*) (*normal-density* (μ $-$ ν) (*sqrt* (σ^2 + τ^2)))
⟨*proof*⟩

lemma (**in** *prob-space*) *setsum-indep-normal*:
assumes *finite I I* $\neq$ {} *indep-vars* ($\lambda i.$ *borel*) *X I*
assumes $\bigwedge i.\ i \in I \Longrightarrow 0 < \sigma\ i$
assumes *normal*: $\bigwedge i.\ i \in I \Longrightarrow$ *distributed M lborel* (*X i*) (*normal-density* (μ *i*) (σ *i*))
shows *distributed M lborel* ($\lambda x.$ $\sum i{\in}I.$ *X i x*) (*normal-density* ($\sum i{\in}I.$ μ *i*) (*sqrt* ($\sum i{\in}I.$ (σ *i*)^2)))
⟨*proof*⟩

lemma (**in** *prob-space*) *standard-normal-distributed-expectation*:
assumes *D*: *distributed M lborel X std-normal-density*
shows *expectation X* = *0*
⟨*proof*⟩

lemma (**in** *prob-space*) *normal-distributed-expectation*:
assumes σ[*arith*]: $0 < \sigma$
assumes *D*: *distributed M lborel X* (*normal-density* μ σ)
shows *expectation X* = μ
⟨*proof*⟩

lemma (**in** *prob-space*) *normal-distributed-variance*:
fixes *a b* :: *real*
assumes [*simp*, *arith*]: $0 < \sigma$
assumes *D*: *distributed M lborel X* (*normal-density* μ σ)
shows *variance X* = σ^2
⟨*proof*⟩

lemma (**in** *prob-space*) *standard-normal-distributed-variance*:
distributed M lborel X std-normal-density $\Longrightarrow$ *variance X* = *1*
⟨*proof*⟩

lemma (**in** *information-space*) *entropy-normal-density*:
assumes [*arith*]: $0 < \sigma$
assumes *D*: *distributed M lborel X* (*normal-density* μ σ)
shows *entropy b lborel X* = *log b* (*2* $*$ *pi* $*$ *exp 1* $*$ σ^2) / *2*
⟨*proof*⟩

end

42 Characteristic Functions

theory *Characteristic-Functions*
imports *Weak-Convergence Interval-Integral Independent-Family Distributions*
begin

lemma *mult-min-right*: $a \geq 0 \Longrightarrow (a :: real) * min\ b\ c = min\ (a * b)\ (a * c)$
⟨*proof*⟩

lemma *sequentially-even-odd*:
assumes *E*: *eventually* ($\lambda n.\ P\ (2 * n)$) *sequentially* **and** *O*: *eventually* ($\lambda n.\ P\ (2 * n + 1)$) *sequentially*
shows *eventually P sequentially*
⟨*proof*⟩

lemma *limseq-even-odd*:
assumes $(\lambda n.\ f\ (2 * n)) \longrightarrow (l :: 'a :: topological\text{-}space)$
and $(\lambda n.\ f\ (2 * n + 1)) \longrightarrow l$
shows $f \longrightarrow l$
⟨*proof*⟩

42.1 Application of the FTC: integrating $e^i x$

abbreviation *iexp* :: *real* ⇒ *complex* **where**
$iexp \equiv (\lambda x.\ exp\ (\mathrm{i} * complex\text{-}of\text{-}real\ x))$

lemma *isCont-iexp* [*simp*]: *isCont iexp x*
⟨*proof*⟩

lemma *has-vector-derivative-iexp*[*derivative-intros*]:
(*iexp has-vector-derivative* i ∗ *iexp x*) (*at x within s*)
⟨*proof*⟩

lemma *interval-integral-iexp*:
fixes *a b* :: *real*
shows $(CLBINT\ x{=}a..b.\ iexp\ x) = ii * iexp\ a - ii * iexp\ b$
⟨*proof*⟩

42.2 The Characteristic Function of a Real Measure.

definition
char :: *real measure* ⇒ *real* ⇒ *complex*
where
$char\ M\ t = CLINT\ x|M.\ iexp\ (t * x)$

lemma (**in** *real-distribution*) *char-zero*: $char\ M\ 0 = 1$
⟨*proof*⟩

lemma (**in** *prob-space*) *integrable-iexp*:
assumes *f*: $f \in borel\text{-}measurable\ M\ \bigwedge x.\ Im\ (f\ x) = 0$
shows *integrable M* ($\lambda x.\ exp\ (ii * (f\ x))$)
⟨*proof*⟩

lemma (**in** *real-distribution*) *cmod-char-le-1*: $norm\ (char\ M\ t) \leq 1$
⟨*proof*⟩

lemma (**in** *real-distribution*) *isCont-char*: *isCont* (*char M*) *t*
⟨*proof*⟩

lemma (**in** *real-distribution*) *char-measurable* [*measurable*]: *char M* ∈ *borel-measurable borel*
⟨*proof*⟩

42.3 Independence

lemma (**in** *prob-space*) *char-distr-sum*:
fixes *X1 X2* :: *'a* ⇒ *real* **and** *t* :: *real*
assumes *indep-var borel X1 borel X2*
shows *char* (*distr M borel* (λω. *X1* ω + *X2* ω)) *t* =
char (*distr M borel X1*) *t* ∗ *char* (*distr M borel X2*) *t*
⟨*proof*⟩

lemma (**in** *prob-space*) *char-distr-setsum*:
indep-vars (λ*i*. *borel*) *X A* ⟹
char (*distr M borel* (λω. ∑*i*∈*A*. *X i* ω)) *t* = (∏*i*∈*A*. *char* (*distr M borel* (*X i*)) *t*)
⟨*proof*⟩

42.4 Approximations to e^{ix}

Proofs from Billingsley, page 343.

lemma *CLBINT-I0c-power-mirror-iexp*:
fixes *x* :: *real* **and** *n* :: *nat*
defines *f s m* ≡ *complex-of-real* ((*x* − *s*) ˆ *m*)
shows (*CLBINT s=0..x*. *f s n* ∗ *iexp s*) =
*x*ˆ*Suc n* / *Suc n* + (*ii* / *Suc n*) ∗ (*CLBINT s=0..x*. *f s* (*Suc n*) ∗ *iexp s*)
⟨*proof*⟩

lemma *iexp-eq1*:
fixes *x* :: *real*
defines *f s m* ≡ *complex-of-real* ((*x* − *s*) ˆ *m*)
shows *iexp x* =
(∑*k* ≤ *n*. (*ii* ∗ *x*)ˆ*k* / (*fact k*)) + ((*ii* ˆ (*Suc n*)) / (*fact n*)) ∗ (*CLBINT s=0..x*. (*f s n*) ∗ (*iexp s*)) (**is** *?P n*)
⟨*proof*⟩

lemma *iexp-eq2*:
fixes *x* :: *real*
defines *f s m* ≡ *complex-of-real* ((*x* − *s*) ˆ *m*)
shows *iexp x* = (∑*k*≤*Suc n*. (*ii*∗*x*)ˆ*k*/*fact k*) + *ii*ˆ*Suc n*/*fact n* ∗ (*CLBINT s=0..x*. *f s n*∗(*iexp s* − *1*))
⟨*proof*⟩

lemma *abs-LBINT-I0c-abs-power-diff*:

$|LBINT\ s=0..x.\ |(x - s)\hat{\ }n|| = |x\ \hat{\ }\ (Suc\ n)\ /\ (Suc\ n)|$
⟨proof⟩

lemma *iexp-approx1*: $cmod\ (iexp\ x - (\sum k \leq n.\ (ii * x)\hat{\ }k\ /\ fact\ k)) \leq |x|\hat{\ }(Suc\ n)\ /\ fact\ (Suc\ n)$
⟨proof⟩

lemma *iexp-approx2*: $cmod\ (iexp\ x - (\sum k \leq n.\ (ii * x)\hat{\ }k\ /\ fact\ k)) \leq 2 * |x|\hat{\ }n\ /\ fact\ n$
⟨proof⟩

lemma (**in** *real-distribution*) *char-approx1*:
assumes *integrable-moments*: $\bigwedge k.\ k \leq n \Longrightarrow integrable\ M\ (\lambda x.\ x\hat{\ }k)$
shows $cmod\ (char\ M\ t - (\sum k \leq n.\ ((ii * t)\hat{\ }k\ /\ fact\ k) * expectation\ (\lambda x.\ x\hat{\ }k))) \leq$
$(2*|t|\hat{\ }n\ /\ fact\ n) * expectation\ (\lambda x.\ |x|\hat{\ }n)$ (**is** $cmod\ (char\ M\ t - ?t1) \leq$ -)
⟨proof⟩

lemma (**in** *real-distribution*) *char-approx2*:
assumes *integrable-moments*: $\bigwedge k.\ k \leq n \Longrightarrow integrable\ M\ (\lambda x.\ x\ \hat{\ }\ k)$
shows $cmod\ (char\ M\ t - (\sum k \leq n.\ ((ii * t)\hat{\ }k\ /\ fact\ k) * expectation\ (\lambda x.\ x\hat{\ }k))) \leq$
$(|t|\hat{\ }n\ /\ fact\ (Suc\ n)) * expectation\ (\lambda x.\ min\ (2 * |x|\hat{\ }n * Suc\ n)\ (|t| * |x|\hat{\ }Suc\ n))$
(**is** $cmod\ (char\ M\ t - ?t1) \leq$ -)
⟨proof⟩

lemma (**in** *real-distribution*) *char-approx3*:
fixes t
assumes
integrable-1: $integrable\ M\ (\lambda x.\ x)$ **and**
integral-1: $expectation\ (\lambda x.\ x) = 0$ **and**
integrable-2: $integrable\ M\ (\lambda x.\ x\hat{\ }2)$ **and**
integral-2: $variance\ (\lambda x.\ x) = \sigma 2$
shows $cmod\ (char\ M\ t - (1 - t\hat{\ }2 * \sigma 2\ /\ 2)) \leq$
$(t\hat{\ }2\ /\ 6) * expectation\ (\lambda x.\ min\ (6 * x\hat{\ }2)\ (abs\ t * (abs\ x)\hat{\ }3)\ )$
⟨proof⟩

This is a more familiar textbook formulation in terms of random variables, but we will use the previous version for the CLT.

lemma (**in** *prob-space*) *char-approx3'*:
fixes μ :: *real measure* **and** X
assumes *rv-X* [*simp*]: *random-variable borel X*
and [*simp*]: $integrable\ M\ X\ integrable\ M\ (\lambda x.\ (X\ x)\hat{\ }2)\ expectation\ X = 0$
and *var-X*: $variance\ X = \sigma 2$
and *μ-def*: $\mu = distr\ M\ borel\ X$
shows $cmod\ (char\ \mu\ t - (1 - t\hat{\ }2 * \sigma 2\ /\ 2)) \leq$
$(t\hat{\ }2\ /\ 6) * expectation\ (\lambda x.\ min\ (6 * (X\ x)\hat{\ }2)\ (|t| * |X\ x|\hat{\ }3))$
⟨proof⟩

this is the formulation in the book – in terms of a random variable *with* the distribution, rather the distribution itself. I don't know which is more useful, though in principal we can go back and forth between them.

lemma (**in** *prob-space*) *char-approx1'*:
fixes μ :: *real measure* **and** X
assumes *integrable-moments* : $\bigwedge k.\ k \le n \Longrightarrow$ *integrable* M $(\lambda x.\ X\ x\ \hat{}\ k)$
and *rv-X*[*measurable*]: *random-variable borel* X
and *μ-distr* : *distr* M *borel* $X = \mu$
shows *cmod* (*char* μ t $-$ ($\sum k \le n.$ ((*ii* $*$ t) $\hat{}k$ / *fact* k) $*$ *expectation* ($\lambda x.$ (X x) $\hat{}k$))) $\le$
(2 $*$ $|t|\ \hat{}n$ / *fact* n) $*$ *expectation* ($\lambda x.$ $|X\ x|\ \hat{}n$)
⟨*proof*⟩

42.5 Calculation of the Characteristic Function of the Standard Distribution

abbreviation
std-normal-distribution $\equiv$ *density lborel std-normal-density*

lemma *real-dist-normal-dist*: *real-distribution std-normal-distribution*
⟨*proof*⟩

lemma *std-normal-distribution-even-moments*:
fixes k :: *nat*
shows (*LINT* x|*std-normal-distribution*. $x\hat{}(2 * k)$) = *fact* ($2 * k$) / ($2\hat{}k$ $*$ *fact* k)
and *integrable std-normal-distribution* ($\lambda x.$ $x\hat{}(2 * k)$)
⟨*proof*⟩

lemma *integrable-std-normal-distribution-moment*: *integrable std-normal-distribution* ($\lambda x.$ $x\hat{}k$)
⟨*proof*⟩

lemma *integral-std-normal-distribution-moment-odd*:
odd k $\Longrightarrow$ $integral^L$ *std-normal-distribution* ($\lambda x.$ $x\hat{}k$) = 0
⟨*proof*⟩

lemma *std-normal-distribution-even-moments-abs*:
fixes k :: *nat*
shows (*LINT* x|*std-normal-distribution*. $|x|\ \hat{}(2 * k)$) = *fact* ($2 * k$) / ($2\hat{}k$ $*$ *fact* k)
⟨*proof*⟩

lemma *std-normal-distribution-odd-moments-abs*:
fixes k :: *nat*
shows (*LINT* x|*std-normal-distribution*. $|x|\ \hat{}(2 * k + 1)$) = *sqrt* (2 / *pi*) $*$ $2\ \hat{}\ k$ $*$ *fact* k
⟨*proof*⟩

theorem *char-std-normal-distribution*:
char std-normal-distribution = (λt. *complex-of-real* (*exp* (− (t^2) / 2)))
⟨*proof*⟩

end

43 Helly's selection theorem

The set of bounded, monotone, right continuous functions is sequentially compact

theory *Helly-Selection*
imports ~~/*src*/*HOL*/*Library*/*Diagonal-Subsequence Weak-Convergence*
begin

lemma *minus-one-less*: $x - 1 < (x::real)$
⟨*proof*⟩

theorem *Helly-selection*:
fixes f :: *nat* ⇒ *real* ⇒ *real*
assumes *rcont*: $\bigwedge n\ x$. *continuous* (*at-right* x) ($f\ n$)
assumes *mono*: $\bigwedge n$. *mono* ($f\ n$)
assumes *bdd*: $\bigwedge n\ x$. $|f\ n\ x| \leq M$
shows $\exists s$. *subseq* s ∧ ($\exists F$. ($\forall x$. *continuous* (*at-right* x) F) ∧ *mono* F ∧ ($\forall x$. $|F\ x| \leq M$) ∧
($\forall x$. *continuous* (*at* x) F ⟶ (λn. f ($s\ n$) x) ⟶ $F\ x$))
⟨*proof*⟩

definition
tight :: (*nat* ⇒ *real measure*) ⇒ *bool*
where
tight μ ≡ ($\forall n$. *real-distribution* ($\mu\ n$)) ∧ ($\forall$ (ε::*real*)>0. $\exists a\ b$::*real*. $a < b$ ∧ ($\forall n$. *measure* ($\mu\ n$) {a<..b} > 1 − ε))

theorem *tight-imp-convergent-subsubsequence*:
assumes μ: *tight* μ *subseq* s
shows $\exists r\ M$. *subseq* r ∧ *real-distribution* M ∧ *weak-conv-m* ($\mu \circ s \circ r$) M
⟨*proof*⟩

corollary *tight-subseq-weak-converge*:
fixes μ :: *nat* ⇒ *real measure* **and** M :: *real measure*
assumes $\bigwedge n$. *real-distribution* ($\mu\ n$) *real-distribution* M **and** *tight*: *tight* μ **and**
subseq: $\bigwedge s\ \nu$. *subseq* s ⟹ *real-distribution* ν ⟹ *weak-conv-m* ($\mu \circ s$) ν ⟹ *weak-conv-m* ($\mu \circ s$) M
shows *weak-conv-m* μ M

⟨*proof*⟩

end

44 Integral of sinc

theory *Sinc-Integral*
imports *Distributions*
begin

44.1 Various preparatory integrals

Naming convention The theorem name consists of the following parts:

- Kind of integral: *has-bochner-integral* / *integrable* / *LBINT*
- Interval: Interval (0 / infinity / open / closed) (infinity / open / closed)
- Name of the occurring constants: power, exp, m (for minus), scale, sin, ...

lemma *has-bochner-integral-I0i-power-exp-m'*:
has-bochner-integral lborel (λx. x^k ∗ *exp* ($-x$) ∗ *indicator* {*0* ..} *x*::*real*) (*fact* *k*)
⟨*proof*⟩

lemma *has-bochner-integral-I0i-power-exp-m*:
has-bochner-integral lborel (λx. x^k ∗ *exp* ($-x$) ∗ *indicator* {*0* <..} *x*::*real*) (*fact* *k*)
⟨*proof*⟩

lemma *integrable-I0i-exp-mscale*: *0* < (*u*::*real*) ⟹ *set-integrable lborel* {*0* <..} (λx. *exp* (−(*x* ∗ *u*)))
⟨*proof*⟩

lemma *LBINT-I0i-exp-mscale*: *0* < (*u*::*real*) ⟹ *LBINT x*=*0*..∞. *exp* (−(*x* ∗ *u*)) = *1* / *u*
⟨*proof*⟩

lemma *LBINT-I0c-exp-mscale-sin*:
LBINT x=*0*..*t*. *exp* (−(*u* ∗ *x*)) ∗ *sin x* =
(*1* / (*1* + *u*^*2*)) ∗ (*1* − *exp* (−(*u* ∗ *t*)) ∗ (*u* ∗ *sin t* + *cos t*)) (**is** - = *?F t*)
⟨*proof*⟩

lemma *LBINT-I0i-exp-mscale-sin*:
assumes *0* < *x*
shows *LBINT u*=*0*..∞. |*exp* (−*u* ∗ *x*) ∗ *sin x*| = |*sin x*| / *x*
⟨*proof*⟩

lemma
shows *integrable-inverse-1-plus-square*:
set-integrable lborel (*einterval* ($-\infty$) ∞) (λx. *inverse* (*1* + *x*ˆ*2*))
and *LBINT-inverse-1-plus-square*:
LBINT $x = -\infty..\infty$. *inverse* (*1* + *x*ˆ*2*) = *pi*
⟨*proof*⟩

lemma
shows *integrable-I0i-1-div-plus-square*:
interval-lebesgue-integrable lborel 0 ∞ (λx. *1* / (*1* + *x*ˆ*2*))
and *LBINT-I0i-1-div-plus-square*:
LBINT $x = 0..\infty$. *1* / (*1* + *x*ˆ*2*) = *pi* / *2*
⟨*proof*⟩

45 The sinc function, and the sine integral (Si)

abbreviation *sinc* :: *real* ⇒ *real* **where**
sinc ≡ (λx. *if x* = *0 then 1 else sin x* / *x*)

lemma *sinc-at-0*: ((λx. *sin x* / *x*::*real*) ⟶ *1*) (*at 0*)
⟨*proof*⟩

lemma *isCont-sinc*: *isCont sinc x*
⟨*proof*⟩

lemma *continuous-on-sinc*[*continuous-intros*]:
continuous-on S f ⟹ *continuous-on S* (λx. *sinc* (*f x*))
⟨*proof*⟩

lemma *borel-measurable-sinc*[*measurable*]: *sinc* ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *sinc-AE*: *AE x in lborel*. *sin x* / *x* = *sinc x*
⟨*proof*⟩

definition *Si* :: *real* ⇒ *real* **where** *Si t* ≡ *LBINT x*=*0*..*t*. *sin x* / *x*

lemma *sinc-neg* [*simp*]: *sinc* (− *x*) = *sinc x*
⟨*proof*⟩

lemma *Si-alt-def* : *Si t* = *LBINT x*=*0*..*t*. *sinc x*
⟨*proof*⟩

lemma *Si-neg*:
assumes *T* ≥ *0* **shows** *Si* (− *T*) = − *Si T*
⟨*proof*⟩

lemma *integrable-sinc'*:
interval-lebesgue-integrable lborel (*ereal 0*) (*ereal T*) (λt. *sin* ($t * \vartheta$) / t)
⟨*proof*⟩

lemma *DERIV-Si*: (*Si has-real-derivative sinc x*) (*at x*)
⟨*proof*⟩

lemma *isCont-Si*: *isCont Si x*
⟨*proof*⟩

lemma *borel-measurable-Si*[*measurable*]: *Si* ∈ *borel-measurable borel*
⟨*proof*⟩

lemma *Si-at-top-LBINT*:
((λt. (*LBINT x=0..∞. exp* ($-(x * t)$) $*$ ($x *$ *sin t* + *cos t*) / ($1 + x\hat{}2$))) ⟶ 0) *at-top*
⟨*proof*⟩

lemma *Si-at-top-integrable*:
assumes $t \geq 0$
shows *interval-lebesgue-integrable lborel 0 ∞* (λx. *exp* ($-$ ($x * t$)) $*$ ($x *$ *sin t* + *cos t*) / ($1 + x^2$))
⟨*proof*⟩

lemma *Si-at-top*: (*Si* ⟶ *pi* / *2*) *at-top*
⟨*proof*⟩

45.1 The final theorems: boundedness and scalability

lemma *bounded-Si*: $\exists B. \forall T. |Si\ T| \leq B$
⟨*proof*⟩

lemma *LBINT-I0c-sin-scale-divide*:
assumes $T \geq 0$
shows *LBINT t=0..T. sin* ($t * \vartheta$) / t = *sgn* ϑ $*$ *Si* ($T * |\vartheta|$)
⟨*proof*⟩

end

46 The Levy inversion theorem, and the Levy continuity theorem.

theory *Levy*
imports *Characteristic-Functions Helly-Selection Sinc-Integral*
begin

lemma *LIM-zero-cancel*:

fixes f :: - $\Rightarrow$ $'b$::*real-normed-vector*
shows $((\lambda x.\ f\ x - l) \longrightarrow 0)\ F \Longrightarrow (f \longrightarrow l)\ F$
⟨*proof*⟩

46.1 The Levy inversion theorem

lemma *Levy-Inversion-aux1*:
fixes $a\ b$ :: *real*
assumes $a \le b$
shows $((\lambda t.\ (iexp\ (-(t * a)) - iexp\ (-(t * b)))\ /\ (ii * t)) \longrightarrow b - a)\ (at\ 0)$
(**is** $(?F \longrightarrow -)\ (at\ -)$)
⟨*proof*⟩

lemma *Levy-Inversion-aux2*:
fixes $a\ b\ t$ :: *real*
assumes $a \le b$ **and** $t \ne 0$
shows $cmod\ ((iexp\ (t * b) - iexp\ (t * a))\ /\ (ii * t)) \le b - a$ (**is** $?F \le -$)
⟨*proof*⟩

theorem (**in** *real-distribution*) *Levy-Inversion*:
fixes $a\ b$ :: *real*
assumes $a \le b$
defines $\mu \equiv measure\ M$ **and** $\varphi \equiv char\ M$
assumes $\mu\ \{a\} = 0$ **and** $\mu\ \{b\} = 0$
shows $(\lambda T.\ 1\ /\ (2 * pi) * (CLBINT\ t{=}-T..T.\ (iexp\ (-(t * a)) - iexp\ (-(t * b)))\ /\ (ii * t) * \varphi\ t))$
$\longrightarrow \mu\ \{a{<}..b\}$
(**is** $(\lambda T.\ 1\ /\ (2 * pi) * (CLBINT\ t{=}-T..T.\ ?F\ t * \varphi\ t)) \longrightarrow of\text{-}real\ (\mu\ \{a{<}..b\})$)
⟨*proof*⟩

theorem *Levy-uniqueness*:
fixes *M1 M2* :: *real measure*
assumes *real-distribution M1 real-distribution M2* **and**
char M1 = char M2
shows *M1 = M2*
⟨*proof*⟩

46.2 The Levy continuity theorem

theorem *levy-continuity1*:
fixes M :: $nat \Rightarrow$ *real measure* **and** M' :: *real measure*
assumes $\bigwedge n.$ *real-distribution* $(M\ n)$ *real-distribution* M' *weak-conv-m* $M\ M'$
shows $(\lambda n.\ char\ (M\ n)\ t) \longrightarrow char\ M'\ t$
⟨*proof*⟩

theorem *levy-continuity*:
fixes M :: $nat \Rightarrow$ *real measure* **and** M' :: *real measure*
assumes *real-distr-M* : $\bigwedge n.$ *real-distribution* $(M\ n)$

and *real-distr-M′*: *real-distribution M′*
and *char-conv*: $\bigwedge t.\ (\lambda n.\ char\ (M\ n)\ t) \longrightarrow char\ M'\ t$
shows *weak-conv-m M M′*
⟨*proof*⟩

end

47 The Central Limit Theorem

theory *Central-Limit-Theorem*
imports *Levy*
begin

theorem (**in** *prob-space*) *central-limit-theorem*:
fixes $X :: nat \Rightarrow {}'a \Rightarrow real$
and $\mu :: real\ measure$
and $\sigma :: real$
and $S :: nat \Rightarrow {}'a \Rightarrow real$
assumes *X-indep*: *indep-vars* ($\lambda i.\ borel$) *X UNIV*
and *X-integrable*: $\bigwedge n.\ integrable\ M\ (X\ n)$
and *X-mean-0*: $\bigwedge n.\ expectation\ (X\ n) = 0$
and *σ-pos*: $\sigma > 0$
and *X-square-integrable*: $\bigwedge n.\ integrable\ M\ (\lambda x.\ (X\ n\ x)^2)$
and *X-variance*: $\bigwedge n.\ variance\ (X\ n) = \sigma^2$
and *X-distrib*: $\bigwedge n.\ distr\ M\ borel\ (X\ n) = \mu$
defines $S\ n \equiv \lambda x.\ \sum i<n.\ X\ i\ x$
shows *weak-conv-m* ($\lambda n.\ distr\ M\ borel\ (\lambda x.\ S\ n\ x\ /\ sqrt\ (n * \sigma^2))$) *std-normal-distribution*
⟨*proof*⟩

end

theory *Probability*
imports
Discrete-Topology
Complete-Measure
Projective-Limit
Probability-Mass-Function
Stream-Space
Embed-Measure
Central-Limit-Theorem
begin

end