Coproduct Measure

Michikazu Hirata

August 22, 2024

Abstract

This entry formalizes the coproduct measure. Let I be a set and $\{M_i\}_{i\in I}$ measurable spaces. The σ-algebra on $\coprod_{i\in I} M_i = \{(i,x) \mid i \in I \wedge x \in M_i\}$ is defined as the least one making $(\lambda x.\, (i,x))$ measurable for all $i \in I$. Let μ_i be measures on M_i for all $i \in I$ and A a measurable set of $\coprod_{i\in I} M_i$. The coproduct measure $\coprod_{i\in I} \mu_i$ is defined as follows:

$$\left(\coprod_{i\in I} \mu_i\right)(A) = \sum_{i\in I} \mu_i(A_i), \quad \text{where } A_i = \{x \mid (i,x) \in A\}.$$

We also prove the relationship with coproduct quasi-Borel spaces: the functor $R : \mathbf{Meas} \to \mathbf{QBS}$ preserves countable coproducts.

Contents

1 Preliminaries **2**
1.1 Polishness of Extended Reals and Non-Negative Extended Reals 3
1.2 Lemmas for Infinite Sum . . . 7

2 Binary Coproduct Measures **10**
2.1 The Measurable Space and Measurability . . . 10
2.2 Measures . . . 15
2.3 Finiteness . . . 17
2.4 σ-Finiteness . . . 17
2.5 Non-Negative Integral . . . 17
2.6 Integrability . . . 19
2.7 The Lebesgue Integral . . . 19

3 Coproduct Measures **20**
3.1 The Measurable Space and Measurability . . . 20
3.2 Measures . . . 24
3.3 Non-Negative Integral . . . 27
3.4 Integrability . . . 28
3.5 The Lebesgue Integral . . . 28

3.6 Finite Coproduct Measures . . . 30
3.7 Countable Infinite Coproduct Measures . . . 30
3.8 Finiteness . . . 34
3.9 σ-Finiteness . . . 34

4 Additional Properties **35**
4.1 S-Finiteness . . . 35
4.2 Standardness . . . 36
4.3 Relationships with Quasi-Borel Spaces . . . 41

1 Preliminaries

theory *Lemmas-Coproduct-Measure*
 imports *HOL−Probability.Probability*
  Standard-Borel-Spaces.Abstract-Metrizable-Topology
begin

lemma *metrizable-space-metric-space*:
 assumes *d*:*Metric-space UNIV d Metric-space.mtopology UNIV d* = *euclidean*
 shows *class.metric-space d* ($\bigsqcap$ *e*∈{*0*<..}. *principal* {(*x*,*y*). *d x y* < *e*}) *open*
proof −
 interpret *Metric-space UNIV d* **by** *fact*
 show *?thesis*
 proof
  show *open U* ⟷ (∀ *x*∈*U*. ∀$_F$ (*x*′, *y*) *in* $\bigsqcap$ *e*∈{*0*<..}. *principal* {(*F*, *y*). *d F y* < *e*}. *x*′ = *x* ⟶ *y* ∈ *U*) **for** *U*
  proof(*subst eventually-INF-base*)
   show *a* ∈ {*0*<..} ⟹ *b* ∈ {*0*<..} ⟹ ∃ *x*∈{*0*<..}. *principal* {(*F*, *y*). *d F y* < *x*} ≤ *principal* {(*F*, *y*). *d F y* < *a*} ⊓ *principal* {(*F*, *y*). *d F y* < *b*} **for** *a b*
    by(*auto intro*!: *bexI*[**where** *x*=*min a b*])
  next
   show *open U* ⟷ (∀ *x*∈*U*. ∃ *b*∈{*0*<..}. ∀$_F$ (*x*′, *y*) *in principal* {(*F*, *y*). *d F y* < *b*}. *x*′ = *x* ⟶ *y* ∈ *U*)
     by(*fastforce simp*: *openin-mtopology*[*simplified d*(*2*),*simplified*] *eventually-principal*)
  qed *simp*
 qed(*auto simp*: *triangle*′)
qed

corollary *metrizable-space-metric-space-ex*:
 assumes *metrizable-space* (*euclidean* :: ′*a* :: *topological-space topology*)
 shows ∃ (*d* :: ′*a* ⇒ ′*a* ⇒ *real*) *F*. *class.metric-space d F open*
proof −
 from *assms* **obtain** *d* :: ′*a* ⇒ ′*a* ⇒ *real* **where** *Metric-space UNIV d Metric-space.mtopology UNIV d* = *euclidean*
  by (*metis Metric-space.topspace-mtopology metrizable-space-def topspace-euclidean*)
 from *metrizable-space-metric-space*[*OF this*] **show** *?thesis*

by *blast*
qed

lemma *completely-metrizable-space-metric-space*:
assumes *Metric-space* (*UNIV* :: *'a* ::*topological-space set*) *d Metric-space.mtopology UNIV d* = *euclidean Metric-space.mcomplete UNIV d*
shows *class.complete-space d* ($\bigsqcap$ *e*∈{*0*<..}. *principal* {(*x*,*y*). *d x y* < *e*}) *open*
proof −
interpret *Metric-space UNIV d* **by** *fact*
interpret *m*:*metric-space d* $\bigsqcap$ *e*∈{*0*<..}. *principal* {(*x*,*y*). *d x y* < *e*} *open*
by(*auto intro*!: *metrizable-space-metric-space assms*)

have [*simp*]:*topological-space.convergent* (*open* :: *'a set* ⇒ *bool*) = *convergent*
proof
fix *x* :: *nat* ⇒ *'a*
have ∗:*class.topological-space* (*open* :: *'a set* ⇒ *bool*)
by *standard auto*
show *topological-space.convergent open x* = *convergent x*
by(*simp add*: *topological-space.convergent-def*[*OF* ∗] *topological-space.nhds-def*[*OF* ∗] *convergent-def nhds-def*)
qed
show *?thesis*
apply *unfold-locales*
using *assms*(*3*) **by**(*auto simp*: *mcomplete-def assms*(*2*) *MCauchy-def m.Cauchy-def convergent-def*)
qed

lemma *completely-metrizable-space-metric-space-ex*:
assumes *completely-metrizable-space* (*euclidean* :: *'a* :: *topological-space topology*)
shows ∃(*d* :: *'a* ⇒ *'a* ⇒ *real*) *F*. *class.complete-space d F open*
proof −
from *assms* **obtain** *d* :: *'a* ⇒ *'a* ⇒ *real* **where** *Metric-space UNIV d Metric-space.mtopology UNIV d* = *euclidean Metric-space.mcomplete UNIV d*
by (*metis Metric-space.topspace-mtopology completely-metrizable-space-def topspace-euclidean*)
from *completely-metrizable-space-metric-space*[*OF this*] **show** *?thesis*
by *blast*
qed

1.1 Polishness of Extended Reals and Non-Negative Extended Reals

We instantiate *polish-space* for *ereal* and *ennreal* with *non-canonical* metrics in order to change the order of *infsum* using the lemma *infsum-Sigma*.

instantiation *ereal* :: *metric-space*
begin

definition *dist-ereal* :: *ereal* ⇒ *ereal* ⇒ *real*
where *dist-ereal* ≡ *SOME d*. *Metric-space UNIV d* ∧

Metric-space.mtopology UNIV d = euclidean ∧
Metric-space.mcomplete UNIV d

definition *uniformity-ereal* :: (*ereal* × *ereal*) *filter*
where *uniformity-ereal* ≡ $\bigsqcap$ *e*∈{*0*<..}. *principal* {(*x*,*y*). *dist x y* < *e*}

instance
proof −
let *?open* = *open* :: *ereal set* ⇒ *bool*
interpret *c*:*complete-space dist uniformity ?open*
proof −
have ∃ *d*. *Metric-space* (*UNIV* :: *ereal set*) *d* ∧
Metric-space.mtopology UNIV d = *euclidean* ∧
Metric-space.mcomplete UNIV d
by (*metis Polish-space-ereal Metric-space.topspace-mtopology Polish-space-def completely-metrizable-space-def topspace-euclidean*)
hence *Metric-space* (*UNIV* :: *ereal set*) *dist* ∧
Metric-space.mtopology (*UNIV* :: *ereal set*) *dist* = *euclidean* ∧
Metric-space.mcomplete (*UNIV* :: *ereal set*) *dist*
unfolding *dist-ereal-def* **by**(*rule someI-ex*)
with *completely-metrizable-space-metric-space* **show** *class.complete-space dist uniformity ?open*
by(*fastforce simp*: *uniformity-ereal-def*)
qed
have [*simp*]:*topological-space.convergent ?open* = *convergent*
proof
fix *x* :: *nat* ⇒ *ereal*
have ∗:*class.topological-space ?open*
by *standard auto*
show *topological-space.convergent open x* = *convergent x*
by(*simp add*: *topological-space.convergent-def topological-space.nhds-def* ∗ *convergent-def nhds-def*)
qed
show *OFCLASS*(*ereal*, *metric-space-class*)
by *standard* (*use uniformity-ereal-def c.open-uniformity c.dist-triangle2 c.Cauchy-convergent* **in** *auto*)
qed

end

instantiation *ereal* :: *polish-space*
begin

instance
proof
let *?open* = *open* :: *ereal set* ⇒ *bool*
interpret *c*:*complete-space dist uniformity ?open*
proof −
have ∃ *d*. *Metric-space* (*UNIV* :: *ereal set*) *d* ∧

```
          Metric-space.mtopology UNIV d = euclidean ∧
          Metric-space.mcomplete UNIV d
      by (metis Polish-space-ereal Metric-space.topspace-mtopology Polish-space-def
completely-metrizable-space-def topspace-euclidean)
    hence Metric-space (UNIV :: ereal set) dist ∧
          Metric-space.mtopology (UNIV :: ereal set) dist = euclidean ∧
          Metric-space.mcomplete (UNIV :: ereal set) dist
      unfolding dist-ereal-def by(rule someI-ex)
    with completely-metrizable-space-metric-space show class.complete-space dist
uniformity ?open
      by(fastforce simp: uniformity-ereal-def)
  qed
  have [simp]:topological-space.convergent ?open = convergent
  proof
    fix x :: nat ⇒ ereal
    have ∗:class.topological-space ?open
      by standard auto
    show topological-space.convergent open x = convergent x
     by(simp add: topological-space.convergent-def topological-space.nhds-def ∗ con-
vergent-def nhds-def)
  qed
  have [simp]:uniform-space.Cauchy (uniformity :: (ereal × ereal) filter) = Cauchy
   by(auto simp add: metric-space.Cauchy-def[OF metric-space-axioms] Cauchy-def)
  fix x :: nat ⇒ ereal
  show Cauchy x ⟹ convergent x
    using c.Cauchy-convergent by(auto simp: Cauchy-def)
qed

end

instantiation ennreal :: metric-space
begin

definition dist-ennreal :: ennreal ⇒ ennreal ⇒ real
  where dist-ennreal ≡ SOME d. Metric-space UNIV d ∧
                               Metric-space.mtopology UNIV d = euclidean ∧
                               Metric-space.mcomplete UNIV d

definition uniformity-ennreal :: (ennreal × ennreal) filter
  where uniformity-ennreal ≡ ⨅ e∈{0<..}. principal {(x,y). dist x y < e}

instance
proof −
  let ?open = open :: ennreal set ⇒ bool
  interpret c:complete-space dist uniformity ?open
  proof −
    have ∃ d. Metric-space (UNIV :: ennreal set) d ∧
          Metric-space.mtopology UNIV d = euclidean ∧
          Metric-space.mcomplete UNIV d
```

by (*metis Polish-space-ennreal Metric-space.topspace-mtopology Polish-space-def completely-metrizable-space-def topspace-euclidean*)
hence *Metric-space* (*UNIV* :: *ennreal set*) *dist* ∧
Metric-space.mtopology (*UNIV* :: *ennreal set*) *dist* = *euclidean* ∧
Metric-space.mcomplete (*UNIV* :: *ennreal set*) *dist*
unfolding *dist-ennreal-def* **by**(*rule someI-ex*)
with *completely-metrizable-space-metric-space* **show** *class.complete-space dist uniformity ?open*
by(*fastforce simp*: *uniformity-ennreal-def*)
qed
have [*simp*]:*topological-space.convergent ?open* = *convergent*
proof
fix *x* :: *nat* ⇒ *ennreal*
have ∗:*class.topological-space ?open*
by *standard auto*
show *topological-space.convergent open x* = *convergent x*
by(*simp add*: *topological-space.convergent-def*[*OF* ∗] *topological-space.nhds-def*[*OF* ∗] *convergent-def nhds-def*)
qed
show *OFCLASS*(*ennreal*, *metric-space-class*)
by *standard* (*use uniformity-ennreal-def c.open-uniformity c.dist-triangle2 c.Cauchy-convergent* **in** *auto*)
qed

end

instantiation *ennreal* :: *polish-space*
begin

instance
proof
let *?open* = *open* :: *ennreal set* ⇒ *bool*
interpret *c*:*complete-space dist uniformity ?open*
proof −
have ∃ *d*. *Metric-space* (*UNIV* :: *ennreal set*) *d* ∧
Metric-space.mtopology UNIV d = *euclidean* ∧
Metric-space.mcomplete UNIV d
by (*metis Polish-space-ennreal Metric-space.topspace-mtopology Polish-space-def completely-metrizable-space-def topspace-euclidean*)
hence *Metric-space* (*UNIV* :: *ennreal set*) *dist* ∧
Metric-space.mtopology (*UNIV* :: *ennreal set*) *dist* = *euclidean* ∧
Metric-space.mcomplete (*UNIV* :: *ennreal set*) *dist*
unfolding *dist-ennreal-def* **by**(*rule someI-ex*)
with *completely-metrizable-space-metric-space* **show** *class.complete-space dist uniformity ?open*
by(*fastforce simp*: *uniformity-ennreal-def*)
qed
have [*simp*]:*topological-space.convergent ?open* = *convergent*
proof

fix x :: *nat* $\Rightarrow$ *ennreal*
have $*$:*class.topological-space ?open*
by *standard auto*
show *topological-space.convergent open* x = *convergent* x
by(*simp add*: *topological-space.convergent-def topological-space.nhds-def* $*$ *convergent-def nhds-def*)
qed
have [*simp*]:*uniform-space.Cauchy* (*uniformity* :: (*ennreal* $\times$ *ennreal*) *filter*) = *Cauchy*
by(*auto simp add*: *metric-space.Cauchy-def*[*OF metric-space-axioms*] *Cauchy-def*)
fix x :: *nat* $\Rightarrow$ *ennreal*
show *Cauchy* $x \Longrightarrow$ *convergent* x
using *c.Cauchy-convergent* **by**(*auto simp*: *Cauchy-def*)
qed

end

1.2 Lemmas for Infinite Sum

lemma *uniformly-continuous-add-ennreal*: *isUCont* (λ(x::*ennreal*, y). $x + y$)
proof(*safe intro*!: *compact-uniformly-continuous*)
have *compact* (*UNIV* $\times$ *UNIV* :: (*ennreal* $\times$ *ennreal*) *set*)
by(*intro compact-Times compact-UNIV*)
thus *compact* (*UNIV* :: (*ennreal* $\times$ *ennreal*) *set*)
by *simp*
qed(*auto intro*!: *continuous-on-add-ennreal continuous-on-fst continuous-on-snd simp*: *split-beta'*)

lemma *infsum-eq-suminf*:
assumes *f summable-on UNIV*
shows ($\sum_\infty$ $n \in$*UNIV*. f n) = *suminf* f
using *has-sum-imp-sums*[*OF has-sum-infsum*[*OF assms*]]
by (*simp add*: *sums-iff*)

lemma *infsum-Sigma-ennreal*:
fixes f :: - $\Rightarrow$ *ennreal*
shows *infsum* f (*Sigma* A B) = *infsum* (λx. *infsum* (λy. f (x, y)) (B x)) A
by(*auto intro*!: *uniformly-continuous-add-ennreal infsum-Sigma nonneg-summable-on-complete*)

lemma *infsum-swap-ennreal*:
fixes f :: - $\Rightarrow$ - $\Rightarrow$ *ennreal*
shows *infsum* (λx. *infsum* (λy. f x y) B) A = *infsum* (λy. *infsum* (λx. f x y) A) B
by(*auto intro*!: *infsum-swap uniformly-continuous-add-ennreal nonneg-summable-on-complete*)

lemma *has-sum-cmult-right-ennreal*:
fixes f :: - $\Rightarrow$ *ennreal*
assumes $c < \top$ (f *has-sum* a) A
shows ((λx. $c * f$ x) *has-sum* $c * a$) A

using *ennreal-tendsto-cmult*[*OF assms*(*1*)] *assms*(*2*)
by (*force simp add*: *has-sum-def sum-distrib-left*)

lemma *infsum-cmult-right-ennreal*:
fixes f :: - ⇒ *ennreal*
assumes $c < \top$
shows $(\sum_\infty x{\in}A.\ c * f\ x) = c * \mathit{infsum}\ f\ A$
by (*simp add*: *assms has-sum-cmult-right-ennreal infsumI nonneg-summable-on-complete*)

lemma *ennreal-sum-SUP-eq*:
fixes f :: *nat* ⇒ - ⇒ *ennreal*
assumes *finite A* $\bigwedge x.\ x \in A \Longrightarrow \mathit{incseq}\ (\lambda j.\ f\ j\ x)$
shows $(\sum i{\in}A.\ \bigsqcup n.\ f\ n\ i) = (\bigsqcup n.\ \sum i{\in}A.\ f\ n\ i)$
using *assms*
proof *induction*
case *empty*
then show *?case*
by *simp*
next
case *ih*:(*insert x F*)
show *?case* (**is** *?lhs* = *?rhs*)
proof −
have *?lhs* = $(\bigsqcup n.\ f\ n\ x) + (\bigsqcup n.\ \mathit{sum}\ (f\ n)\ F)$
using *ih* **by** *simp*
also have ... = $(\bigsqcup n.\ f\ n\ x + \mathit{sum}\ (f\ n)\ F)$
using *ih* **by**(*auto intro*!: *incseq-sumI2 ennreal-SUP-add*[*symmetric*])
also have ... = *?rhs*
using *ih* **by** *simp*
finally show *?thesis* **.**
qed
qed

lemma *ennreal-infsum-Sup-eq*:
fixes f :: *nat* ⇒ - ⇒ *ennreal*
assumes $\bigwedge x.\ x \in A \Longrightarrow \mathit{incseq}\ (\lambda j.\ f\ j\ x)$
shows $(\sum_\infty x{\in}A.\ (\mathit{SUP}\ j.\ f\ j\ x)) = (\mathit{SUP}\ j.\ (\sum_\infty x{\in}A.\ f\ j\ x))$ (**is** *?lhs* = *?rhs*)
proof −
have *?lhs* = $(\bigsqcup\ (\mathit{sum}\ (\lambda x.\ \bigsqcup j.\ f\ j\ x)\ `\ \{F.\ \mathit{finite}\ F \wedge F \subseteq A\}))$
by(*auto intro*!: *nonneg-infsum-complete simp*: *SUP-upper2 assms*)
also have ... = $(\bigsqcup A{\in}\{F.\ \mathit{finite}\ F \wedge F \subseteq A\}.\ \bigsqcup j.\ \mathit{sum}\ (f\ j)\ A)$
using *assms* **by**(*auto intro*!: *SUP-cong ennreal-sum-SUP-eq*)
also have ... = $(\bigsqcup j.\ \bigsqcup A{\in}\{F.\ \mathit{finite}\ F \wedge F \subseteq A\}.\ \mathit{sum}\ (f\ j)\ A)$
using *SUP-commute* **by** *fast*
also have ... = *?rhs*
by(*subst nonneg-infsum-complete*) (*use assms* **in** *auto*)
finally show *?thesis* **.**
qed

lemma *bounded-infsum-summable*:

assumes $\bigwedge x.\ x \in A \Longrightarrow f\ x \geq 0$ $(\sum_\infty x{\in}A.\ ennreal\ (f\ x)) < top$
shows *f summable-on A*
proof(*rule nonneg-bdd-above-summable-on*)
from *assms*(*2*) **obtain** *K* **where** $K{:}(\sum_\infty x{\in}A.\ ennreal\ (f\ x)) \leq ennreal\ K\ K \geq 0$
using *less-top-ennreal* **by** *force*
show *bdd-above* (*sum f* ' $\{F.\ F \subseteq A \wedge finite\ F\}$)
proof(*safe intro*!: *bdd-aboveI*[**where** *M*=*K*])
fix A'
assume $A'{:}A' \subseteq A$ *finite* A'
have $(\sum_\infty x{\in}A.\ ennreal\ (f\ x)) = (\bigsqcup\ (sum\ (\lambda x.\ ennreal\ (f\ x))\ `\ \{F.\ finite\ F \wedge F \subseteq A\}))$
by (*simp add*: *nonneg-infsum-complete*)
also have ... $= (\bigsqcup\ ((\lambda F.\ ennreal\ (sum\ f\ F))\ `\ \{F.\ finite\ F \wedge F \subseteq A\}))$
by(*auto intro*!: *SUP-cong sum-ennreal assms*)
finally have $(\bigsqcup\ ((\lambda F.\ ennreal\ (sum\ f\ F))\ `\ \{F.\ finite\ F \wedge F \subseteq A\})) \leq ennreal\ K$
using *K* **by** *order*
hence *ennreal* (*sum f* A') $\leq$ *ennreal K*
by (*simp add*: A' *SUP-le-iff*)
thus *sum f* $A' \leq K$
by (*simp add*: *K*(*2*))
qed
qed *fact*

lemma *infsum-less-top-dest*:
fixes *f* :: - ⇒ -::{*ordered-comm-monoid-add*, *topological-comm-monoid-add*, *t2-space*, *complete-linorder*, *linorder-topology*}
assumes $(\sum_\infty x{\in}A.\ f\ x) < top$ $\bigwedge x.\ x \in A \Longrightarrow f\ x \geq 0$ $x \in A$
shows $f\ x < top$
proof(*rule ccontr*)
assume $f{:}\neg\ f\ x < top$
have $(\sum_\infty x{\in}A.\ f\ x) = (\sum_\infty y{\in}A - \{x\} \cup \{x\}.\ f\ y)$
by(*rule arg-cong*[**where** *f*=*infsum* -]) (*use assms* **in** *auto*)
also have ... $= (\sum_\infty y{\in}A - \{x\}.\ f\ y) + (\sum_\infty y{\in}\{x\}.\ f\ y)$
using *assms*(*2*) **by**(*intro infsum-Un-disjoint*) (*auto intro*!: *nonneg-summable-on-complete*)
also have ... $= (\sum_\infty y{\in}A - \{x\}.\ f\ y) + top$
using *f top.not-eq-extremum* **by** *fastforce*
also have ... = *top*
by(*auto intro*!: *add-top infsum-nonneg assms*)
finally show *False*
using *assms*(*1*) **by** *simp*
qed

lemma *infsum-ennreal-eq*:
assumes *f summable-on A* $\bigwedge x.\ x \in A \Longrightarrow f\ x \geq 0$
shows $(\sum_\infty x{\in}A.\ ennreal\ (f\ x)) = ennreal\ (\sum_\infty x{\in}A.\ f\ x)$
proof −
have $(\sum_\infty x{\in}A.\ ennreal\ (f\ x)) = (\bigsqcup\ (sum\ (\lambda x.\ ennreal\ (f\ x))\ `\ \{F.\ finite\ F \wedge$

$F \subseteq A\}))$
  by (*simp add*: *nonneg-infsum-complete*)
 also have ... = ($\bigsqcup$ (($\lambda F.$ *ennreal* (*sum f F*)) ‘ $\{F.$ *finite F* $\wedge$ $F \subseteq A\}))$
  by(*auto intro*!: *SUP-cong sum-ennreal assms*)
 also have ... = *ennreal* ($\sum_\infty x \in A.\ f\ x$)
  using *infsum-nonneg-is-SUPREMUM-ennreal*[*OF assms*] **by** *simp*
 finally show *?thesis* **.**
qed

lemma *abs-summable-on-integrable-iff*:
 fixes *f* :: - ⇒ - :: {*banach*, *second-countable-topology*}
 shows *Infinite-Sum.abs-summable-on f A* ⟷ *integrable* (*count-space A*) *f*
 by (*simp add*: *abs-summable-equivalent abs-summable-on-def*)

lemma *infsum-eq-integral*:
 fixes *f* :: - ⇒ - :: {*banach*, *second-countable-topology*}
 assumes *Infinite-Sum.abs-summable-on f A*
 shows *infsum f A* = $integral^L$ (*count-space A*) *f*
 using *assms infsetsum-infsum*[*of f A*,*symmetric*]
 by(*auto simp*: *abs-summable-on-integrable-iff abs-summable-on-def infsetsum-def*)

end

theory *Coproduct-Measure*
 imports *Lemmas-Coproduct-Measure*
   HOL−Analysis.Analysis
begin

2 Binary Coproduct Measures

definition *copair-measure* :: [′*a measure*, ′*b measure*] ⇒ (′*a* + ′*b*) *measure* (**infixr** $\bigoplus_M$ *65*) **where**
$M \bigoplus_M N$ = *measure-of* (*space M* <+> *space N*)
   ({*Inl* ‘ *A* |*A*. *A* ∈ *sets M*} ∪ {*Inr* ‘ *A*|*A*. *A* ∈ *sets N*})
   (λ*A*. *emeasure M* (*Inl* −‘ *A*) + *emeasure N* (*Inr* −‘ *A*))

2.1 The Measurable Space and Measurability

lemma
 shows *space-copair-measure*: *space* (*copair-measure M N*) = *space M* <+> *space N*
  and *sets-copair-measure-sigma*:
   sets (*copair-measure M N*)
    = *sigma-sets* (*space M* <+> *space N*) ({*Inl* ‘ *A* |*A*. *A* ∈ *sets M*} ∪ {*Inr* ‘ *A*|*A*. *A* ∈ *sets N*})
  and *Inl-measurable*[*measurable*]: *Inl* ∈ $M \to_M M \bigoplus_M N$
  and *Inr-measurable*[*measurable*]: *Inr* ∈ $N \to_M M \bigoplus_M N$
proof −

have *1*:($\{Inl$ ‘ A $|A.$ $A \in$ *sets* $M\} \cup \{Inr$ ‘ $A|A.$ $A \in$ *sets* $N\}) \subseteq$ *Pow* (*space* M <+> *space* N)
using *sets.sets-into-space*[*of* - M] *sets.sets-into-space*[*of* - N] **by** *fastforce*
show *space* (*copair-measure* M N) = *space* M <+> *space* N
and *2*:*sets* (*copair-measure* M N)
= *sigma-sets* (*space* M <+> *space* N) ($\{Inl$ ‘ A $|A.$ $A \in$ *sets* $M\} \cup \{Inr$ ‘ $A|A.$ $A \in$ *sets* $N\}$)
by(*simp-all add*: *copair-measure-def sets-measure-of*[*OF 1*] *space-measure-of*[*OF 1*])
show *Inl* $\in M \to_M M \bigoplus_M N$ *Inr* $\in N \to_M M \bigoplus_M N$
by(*auto intro*!: *measurable-sigma-sets*[*OF 2 1*] *simp*: *vimage-def image-def*)
qed

lemma *sets-copair-measure-cong*:
sets M1 = *sets M2* $\Longrightarrow$ *sets N1* = *sets N2* $\Longrightarrow$ *sets* ($M1 \bigoplus_M N1$) = *sets* ($M2 \bigoplus_M N2$)
by(*simp cong*: *sets-eq-imp-space-eq add*: *sets-copair-measure-sigma*)

lemma *measurable-image-Inl*[*measurable*]: $A \in$ *sets* $M \Longrightarrow$ *Inl* ‘ $A \in$ *sets* ($M \bigoplus_M N$)
using *sets-copair-measure-sigma* **by** *fastforce*

lemma *measurable-image-Inr*[*measurable*]: $A \in$ *sets* $N \Longrightarrow$ *Inr* ‘ $A \in$ *sets* ($M \bigoplus_M N$)
using *sets-copair-measure-sigma* **by** *fastforce*

lemma *measurable-vimage-Inl*:
assumes [*measurable*]:$A \in$ *sets* ($M \bigoplus_M N$)
shows *Inl* $-$‘ $A \in$ *sets* M
proof $-$
have *Inl* $-$‘ A = *Inl* $-$‘ $A \cap$ *space* M
using *sets.sets-into-space*[*OF assms*]
by(*auto simp add*: *space-copair-measure*)
also have ... $\in$ *sets* M
by *simp*
finally show *?thesis* **.**
qed

lemma *measurable-vimage-Inr*:
assumes [*measurable*]:$A \in$ *sets* ($M \bigoplus_M N$)
shows *Inr* $-$‘ $A \in$ *sets* N
proof $-$
have *Inr* $-$‘ A = *Inr* $-$‘ $A \cap$ *space* N
using *sets.sets-into-space*[*OF assms*]
by(*auto simp add*: *space-copair-measure*)
also have ... $\in$ *sets* N
by *simp*
finally show *?thesis* **.**
qed

lemma *in-sets-copair-measure-iff*:
$A \in$ *sets* (*copair-measure M N*) $\longleftrightarrow$ *Inl* $-`$ $A \in$ *sets M* $\wedge$ *Inr* $-`$ $A \in$ *sets N*
proof *safe*
  assume [*measurable*]: *Inl* $-`$ $A \in$ *sets M Inr* $-`$ $A \in$ *sets N*
  have $A = ((Inl\ `\ Inl\ -`\ A) \cup (Inr\ `\ Inr\ -`\ A))$
    by(*simp add*: *vimage-def image-def*) (*safe*, *metis obj-sumE*)
  also have ... $\in$ *sets* (*copair-measure M N*)
    by *measurable*
  finally show $A \in$ *sets* (*copair-measure M N*) **.**
qed(*use measurable-vimage-Inl measurable-vimage-Inr* **in** *auto*)

lemma *measurable-copair-Inl-Inr*:
  assumes [*measurable*]:$(\lambda x.\ f\ (Inl\ x)) \in M \to_M L\ (\lambda x.\ f\ (Inr\ x)) \in N \to_M L$
  shows $f \in M \bigoplus_M N \to_M L$
proof(*rule measurableI*)
  fix A
  assume [*measurable*]:$A \in$ *sets L*
  have $f\ -`\ A = Inl\ `\ ((\lambda x.\ f\ (Inl\ x))\ -`\ A) \cup Inr\ `\ ((\lambda x.\ f\ (Inr\ x))\ -`\ A)$
    by(*simp add*: *image-def vimage-def*) (*safe*, *metis obj-sumE*)
  hence $f\ -`\ A \cap space\ (M \bigoplus_M N)$
        $= Inl\ `\ ((\lambda x.\ f\ (Inl\ x))\ -`\ A \cap space\ M) \cup Inr\ `\ ((\lambda x.\ f\ (Inr\ x))\ -`\ A \cap space\ N)$
    by(*auto simp*: *space-copair-measure*)
  also have ... $\in$ *sets* $(M \bigoplus_M N)$
    by *measurable*
  finally show $f\ -`\ A \cap space\ (M \bigoplus_M N) \in sets\ (M \bigoplus_M N)$ **.**
next
  show $\bigwedge x.\ x \in space\ (M \bigoplus_M N) \Longrightarrow f\,x \in space\ L$
    using *measurable-space*[*OF assms*(*1*)] *measurable-space*[*OF assms*(*2*)]
    by(*auto simp add*: *space-copair-measure*)
qed

corollary *measurable-copair-measure-iff*:
$f \in M \bigoplus_M N \to_M L \longleftrightarrow (\lambda x.\ f\ (Inl\ x)) \in M \to_M L \wedge (\lambda x.\ f\ (Inr\ x)) \in N \to_M L$
  by(*auto simp add*: *measurable-copair-Inl-Inr*)

lemma *measurable-copair-dest1*:
  assumes [*measurable*]:$f \in L \to_M M \bigoplus_M N$ **and** $f\ -`\ (Inl\ `\ space\ M) \cap space\ L = space\ L$
  obtains f' **where** $f' \in L \to_M M\ \bigwedge x.\ x \in space\ L \Longrightarrow f\,x = Inl\ (f'\ x)$
proof $-$
  define f' **where** $f' \equiv (\lambda x.\ SOME\ y.\ f\,x = Inl\ y)$
  have f':$\bigwedge x.\ x \in space\ L \Longrightarrow f\,x = Inl\ (f'\ x)$
    unfolding f'-*def* **by**(*rule someI-ex*) (*use assms*(*2*) **in** *blast*)
  moreover have $f' \in L \to_M M$
  proof(*rule measurableI*)
    show $\bigwedge x.\ x \in space\ L \Longrightarrow f'\ x \in space\ M$

    using f' *measurable-space*[*OF assms*(*1*)]
    by(*auto simp*: *space-copair-measure*)
  next
    fix A
    assume A[*measurable*]:$A \in$ *sets* M
    have [*simp*]:$f' -`\ A \cap$ *space* $L = f -`\ (Inl\ `\ A) \cap$ *space* L
      using f' *sets.sets-into-space*[*OF A*] **by** *auto*
    show $f' -`\ A \cap$ *space* $L \in$ *sets* L
      by *auto*
  qed
  ultimately show *?thesis*
    using *that* **by** *blast*
qed

lemma *measurable-copair-dest2*:
  assumes [*measurable*]:$f \in L \rightarrow_M M \bigoplus_M N$ **and** $f -`\ (Inr\ `$ *space* $N) \cap$ *space* $L =$ *space* L
  obtains f' **where** $f' \in L \rightarrow_M N$ $\bigwedge x.\ x \in$ *space* $L \Longrightarrow f\ x = Inr\ (f'\ x)$
proof $-$
  define f' **where** $f' \equiv (\lambda x.\ SOME\ y.\ f\ x = Inr\ y)$
  have f':$\bigwedge x.\ x \in$ *space* $L \Longrightarrow f\ x = Inr\ (f'\ x)$
    unfolding f'*-def* **by**(*rule someI-ex*) (*use assms*(*2*) **in** *blast*)
  moreover have $f' \in L \rightarrow_M N$
  proof(*rule measurableI*)
    show $\bigwedge x.\ x \in$ *space* $L \Longrightarrow f'\ x \in$ *space* N
      using f' *measurable-space*[*OF assms*(*1*)]
      by(*auto simp*: *space-copair-measure*)
  next
    fix A
    assume A[*measurable*]:$A \in$ *sets* N
    have [*simp*]:$f' -`\ A \cap$ *space* $L = f -`\ (Inr\ `\ A) \cap$ *space* L
      using f' *sets.sets-into-space*[*OF A*] **by** *auto*
    show $f' -`\ A \cap$ *space* $L \in$ *sets* L
      by *auto*
  qed
  ultimately show *?thesis*
    using *that* **by** *blast*
qed

lemma *measurable-copair-dest3*:
  assumes [*measurable*]:$f \in L \rightarrow_M M \bigoplus_M N$
    and $f -`\ (Inl\ `$ *space* $M) \cap$ *space* $L \subset$ *space* L $f -`\ (Inr\ `$ *space* $N) \cap$ *space* $L \subset$ *space* L
  obtains $f'\ f''$ **where** $f' \in L \rightarrow_M M$ $f'' \in L \rightarrow_M N$
    $\bigwedge x.\ x \in$ *space* $L \Longrightarrow x \in f -`\ Inl\ `$ *space* $M \Longrightarrow f\ x = Inl\ (f'\ x)$
    $\bigwedge x.\ x \in$ *space* $L \Longrightarrow x \notin f -`\ Inl\ `$ *space* $M \Longrightarrow f\ x = Inr\ (f''\ x)$
proof $-$
  have *ne*:*space* $M \neq \{\}$ *space* $N \neq \{\}$
    using *assms*(*2*,*3*) *measurable-space*[*OF assms*(*1*)] **by**(*fastforce simp*: *space-copair-measure*)+

define m **where** $m \equiv$ *SOME* y. $y \in$ *space* M
define n **where** $n \equiv$ *SOME* y. $y \in$ *space* N
have m[*measurable*, *simp*]:$m \in$ *space* M **and** n[*measurable*, *simp*]:$n \in$ *space* N
  using *ne* **by**(*auto simp*: *n-def m-def some-in-eq*)
define f' **where** $f' \equiv (\lambda x.$ *if* $x \in f$ −‘ *Inl* ‘ *space* M *then SOME* y. $f\ x =$ *Inl* y *else* $m)$
have $\bigwedge x.$ $x \in$ *space* $L \Longrightarrow x \in f$ −‘ *Inl* ‘ *space* $M \Longrightarrow f\ x =$ *Inl* (*SOME* y. $f\ x =$ *Inl* y)
  unfolding f'*-def* **by**(*rule someI-ex*) (*use assms*(*2*) **in** *blast*)
hence f':$\bigwedge x.$ $x \in$ *space* $L \Longrightarrow$ $x \in f$ −‘ *Inl* ‘ *space* $M \Longrightarrow f\ x =$ *Inl* $(f'\ x)$
  by(*simp add*: f'*-def*)
hence f'*-space*: $x \in$ *space* $L \Longrightarrow f'\ x \in$ *space* M **for** x
  using *measurable-space*[*OF assms*(*1*)]
  by(*cases* $x \in f$ −‘ *Inl* ‘ *space* M) (*auto simp*: *space-copair-measure* f'*-def*)
define f'' **where** $f'' \equiv (\lambda x.$ *if* $x \notin f$ −‘ *Inl* ‘ *space* M *then SOME* y. $f\ x =$ *Inr* y *else* $n)$
have $*$:$\bigwedge x.$ $x \in$ *space* $L \Longrightarrow x \notin f$ −‘ *Inl* ‘ *space* $M \Longrightarrow x \in f$ −‘ *Inr* ‘ *space* N
  using *measurable-space*[*OF assms*(*1*)] **by**(*fastforce simp*: *space-copair-measure*)
have $\bigwedge x.$ $x \in$ *space* $L \Longrightarrow x \notin f$ −‘ *Inl* ‘ *space* $M \Longrightarrow f\ x =$ *Inr* (*SOME* y. $f\ x =$ *Inr* y)
  unfolding f''*-def* **by**(*rule someI-ex*) (*use* $*$ **in** *blast*)
hence f'':$\bigwedge x.$ $x \in$ *space* $L \Longrightarrow x \notin f$ −‘ *Inl* ‘ *space* $M \Longrightarrow f\ x =$ *Inr* $(f''\ x)$
  by(*simp add*: f''*-def*)
hence f''*-space*:$x \in$ *space* $L \Longrightarrow$ $f''\ x \in$ *space* N **for** x
  using *measurable-space*[*OF assms*(*1*),*of* x]
  by(*cases* $x \notin f$ −‘ *Inl* ‘ *space* M) (*auto simp add*: *space-copair-measure* f''*-def*)
have $f' \in L \rightarrow_M M$
proof −
  have $f' = (\lambda x.$ *if* $x \in f$ −‘ *Inl* ‘ *space* M *then* $f'\ x$ *else* $m)$
    by(*auto simp add*: f'*-def*)
  also have ... $\in L \rightarrow_M M$
  proof(*intro measurable-restrict-space-iff*[*THEN iffD1*] *measurableI*)
    fix A
    assume A[*measurable*]:$A \in$ *sets* M
    have [*measurable*]:$f \in$ *restrict-space* L (f −‘ *Inl* ‘ *space* M) $\rightarrow_M M \bigoplus_M N$
      by(*auto intro*!: *measurable-restrict-space1*)
    have [*simp*]:f' −‘ $A \cap$ *space* (*restrict-space* L (f −‘ *Inl* ‘ *space* M))
          $= f$ −‘ (*Inl* ‘ A) $\cap$ *space* (*restrict-space* L (f −‘ *Inl* ‘ *space* M))
      using f' *sets.sets-into-space*[*OF* A] **by**(*fastforce simp*: *space-restrict-space*)
    show f' −‘ $A \cap$ *space* (*restrict-space* L (f −‘ *Inl* ‘ *space* M))
        $\in$ *sets* (*restrict-space* L (f −‘ *Inl* ‘ *space* M))
      by *simp*
  next
    show $\bigwedge x.$ $x \in$ *space* (*restrict-space* L (f −‘ *Inl* ‘ *space* M)) $\Longrightarrow f'\ x \in$ *space* M
      by(*auto simp*: *space-restrict-space* f'*-space*)
  qed *simp-all*
  finally show *?thesis* **.**
qed

moreover have $f'' \in L \to_M N$
proof $-$
have $f'' = (\lambda x.\ if\ x \notin f -` Inl ` space\ M\ then\ f''\ x\ else\ n)$
by(*auto simp add*: f''*-def*)
also have ... $\in L \to_M N$
proof(*rule measurable-If-restrict-space-iff*[*THEN iffD2*,*OF - conjI*[*OF measurableI*]])
fix *A*
assume *A*[*measurable*]:$A \in sets\ N$
have *f*:$f \in restrict\text{-}space\ L\ \{x.\ x \notin f -` Inl ` space\ M\} \to_M M \bigoplus_M N$
by(*auto intro*!: *measurable-restrict-space1*)
have *1*:$f'' -` A \cap space\ (restrict\text{-}space\ L\ \{x.\ x \notin f -` Inl ` space\ M\})$
$= f -` (Inr ` A) \cap space\ (restrict\text{-}space\ L\ \{x.\ x \notin f -` Inl ` space\ M\})$
using f'' *sets.sets-into-space*[*OF A*] **by**(*fastforce simp*: *space-restrict-space*)
show $f'' -` A \cap space\ (restrict\text{-}space\ L\ \{x.\ x \notin f -` Inl ` space\ M\})$
$\in sets\ (restrict\text{-}space\ L\ \{x.\ x \notin f -` Inl ` space\ M\})$
unfolding *1* **using** *f* **by** *simp*
next
show $\bigwedge x.\ x \in space\ (restrict\text{-}space\ L\ \{x.\ x \notin f -` Inl ` space\ M\}) \Longrightarrow f''\ x \in space\ N$
by(*auto simp*: *space-restrict-space* f''*-space*)
qed *simp-all*
finally show *?thesis* **.**
qed
ultimately show *?thesis*
using *that* $f'\ f''$ **by** *blast*
qed

2.2 Measures

lemma *emeasure-copair-measure*:
assumes [*measurable*]: $A \in sets\ (M \bigoplus_M N)$
shows $emeasure\ (M \bigoplus_M N)\ A = emeasure\ M\ (Inl -` A) + emeasure\ N\ (Inr -` A)$
proof(*rule emeasure-measure-of*)
show $\{Inl ` A\ |A.\ A \in sets\ M\} \cup \{Inr ` A|A.\ A \in sets\ N\} \subseteq Pow\ (space\ M <+> space\ N)$
using *sets.sets-into-space*[*of - M*] *sets.sets-into-space*[*of - N*] **by** *fastforce*
show $A \in sets\ (M \bigoplus_M N)$
by *fact*
show $countably\text{-}additive\ (sets\ (M \bigoplus_M N))\ (\lambda a.\ emeasure\ M\ (Inl -` a) + emeasure\ N\ (Inr -` a))$
proof(*safe intro*!: *countably-additiveI*)
note [*measurable*] = *measurable-vimage-Inl*[*of - M N*] *measurable-vimage-Inr*[*of - M N*]
fix $A :: nat \Rightarrow\ \text{-}\ set$
assume *h*:$range\ A \subseteq sets\ (M \bigoplus_M N)$ *disjoint-family A*
then have [*measurable*]: $\bigwedge i.\ A\ i \in sets\ (M \bigoplus_M N)$
by *blast*

have *disj*:*disjoint-family* ($\lambda i.$ *Inl* $-$‘ *A i*) *disjoint-family* ($\lambda i.$ *Inr* $-$‘ *A i*)
using *h* **by**(*auto simp*: *disjoint-family-on-def*)
show ($\sum i.$ *emeasure M* (*Inl* $-$‘ *A i*) + *emeasure N* (*Inr* $-$‘ *A i*))
= *emeasure M* (*Inl* $-$‘ $\bigcup$ (*range A*)) + *emeasure N* (*Inr* $-$‘ $\bigcup$ (*range A*)) (**is** *?lhs* = *?rhs*)
proof −
have *?lhs* = ($\sum i.$ *emeasure M* (*Inl* $-$‘ *A i*)) + ($\sum i.$ *emeasure N* (*Inr* $-$‘ *A i*))
by(*simp add*: *suminf-add*)
also have ... = *emeasure M* ($\bigcup i.$ (*Inl* $-$‘ *A i*)) + *emeasure N* ($\bigcup i.$ (*Inr* $-$‘ *A i*))
proof −
have ($\sum i.$ *emeasure M* (*Inl* $-$‘ *A i*)) = *emeasure M* ($\bigcup i.$ (*Inl* $-$‘ *A i*))
($\sum i.$ *emeasure N* (*Inr* $-$‘ *A i*)) = *emeasure N* ($\bigcup i.$ (*Inr* $-$‘ *A i*))
by(*auto intro*!: *suminf-emeasure disj*)
thus *?thesis*
by *argo*
qed
also have ... = *?rhs*
by(*simp add*: *vimage-UN*)
finally show *?thesis* **.**
qed
qed
qed(*auto simp*: *positive-def copair-measure-def*)

lemma *emeasure-copair-measure-space*:
emeasure ($M \bigoplus_M N$) (*space* ($M \bigoplus_M N$)) = *emeasure M* (*space M*) + *emeasure N* (*space N*)
proof −
have [*simp*]:*Inl* $-$‘ *space* ($M \bigoplus_M N$) = *space M* *Inr* $-$‘ *space* ($M \bigoplus_M N$) = *space N*
by(*auto simp*: *space-copair-measure*)
show *?thesis*
by(*simp add*: *emeasure-copair-measure*)
qed

corollary
shows *emeasure-copair-measure-Inl*: $A \in$ *sets M* $\Longrightarrow$ *emeasure* ($M \bigoplus_M N$) (*Inl* ‘ *A*) = *emeasure M A*
and *emeasure-copair-measure-Inr*: $B \in$ *sets N* $\Longrightarrow$ *emeasure* ($M \bigoplus_M N$) (*Inr* ‘ *B*) = *emeasure N B*
proof −
have [*simp*]:*Inl* $-$‘ *Inl* ‘ *A* = *A* *Inr* $-$‘ *Inl* ‘ *A* = {} *Inl* $-$‘ *Inr* ‘ *B* = {} *Inr* $-$‘ *Inr* ‘ *B* = *B*
by *auto*
show $A \in$ *sets M* $\Longrightarrow$ *emeasure* ($M \bigoplus_M N$) (*Inl* ‘ *A*) = *emeasure M A*
$B \in$ *sets N* $\Longrightarrow$ *emeasure* ($M \bigoplus_M N$) (*Inr* ‘ *B*) = *emeasure N B*
by(*simp-all add*: *emeasure-copair-measure*)
qed

lemma *measure-copair-measure*:
assumes [*measurable*]:$A \in$ *sets* $(M \bigoplus_M N)$ *emeasure* $(M \bigoplus_M N)$ $A < \infty$
shows *measure* $(M \bigoplus_M N)$ A = *measure* M (*Inl* $-$‘ A) + *measure* N (*Inr* $-$‘ A)
using *assms*(*2*) **by**(*auto simp add*: *emeasure-copair-measure measure-def intro*!: *enn2real-plus*)

lemma
shows *measure-copair-measure-Inl*: $A \in$ *sets* $M \Longrightarrow$ *measure* $(M \bigoplus_M N)$ (*Inl* ‘ A) = *measure* M A
and *measure-copair-measure-Inr*: $B \in$ *sets* $N \Longrightarrow$ *measure* $(M \bigoplus_M N)$ (*Inr* ‘ B) = *measure* N B
by(*auto simp*: *emeasure-copair-measure-Inl measure-def emeasure-copair-measure-Inr*)

2.3 Finiteness

lemma *finite-measure-copair-measure*: *finite-measure* $M \Longrightarrow$ *finite-measure* $N \Longrightarrow$ *finite-measure* $(M \bigoplus_M N)$
by(*auto intro*!: *finite-measureI simp*: *emeasure-copair-measure-space finite-measure.finite-emeasure-space*)

2.4 σ-Finiteness

lemma *sigma-finite-measure-copair-measure*:
assumes *sigma-finite-measure* M *sigma-finite-measure* N
shows *sigma-finite-measure* $(M \bigoplus_M N)$
proof $-$
obtain A B **where** AB[*measurable*]: $\bigwedge i.$ A $i \in$ *sets* M ($\bigcup$ (*range* A)) = *space* M $\bigwedge i$::*nat*. *emeasure* M (A i) $\neq \infty$
$\bigwedge i.$ B $i \in$ *sets* N ($\bigcup$ (*range* B)) = *space* N $\bigwedge i$::*nat*. *emeasure* N (B i) $\neq \infty$
by (*metis range-subsetD sigma-finite-measure.sigma-finite assms*)
then have $*$:($\bigcup$ (*range* ($\lambda i.$ *Inl* ‘ (A i) $\cup$ *Inr* ‘ (B i)))) = *space* $(M \bigoplus_M N)$
unfolding *space-copair-measure Plus-def* **by** *fastforce*
have [*simp*]: $\bigwedge i.$ *Inl* $-$‘ *Inl* ‘ A i $\cup$ *Inl* $-$‘ *Inr* ‘ B i = A i $\bigwedge i.$ *Inr* $-$‘ *Inl* ‘ A i $\cup$ *Inr* $-$‘ *Inr* ‘ B i = B i
using *sets.sets-into-space* AB(*1*,*4*) **by** *blast*+
show *?thesis*
apply *standard*
using AB $*$ **by**(*auto intro*!: *exI*[**where** x=*range* ($\lambda i.$ *Inl* ‘ (A i) $\cup$ *Inr* ‘ (B i))]
simp: *space-copair-measure emeasure-copair-measure*)
qed

2.5 Non-Negative Integral

lemma *nn-integral-copair-measure*:
assumes $f \in$ *borel-measurable* $(M \bigoplus_M N)$
shows ($\int^+ x.$ f x $\partial(M \bigoplus_M N)$) = ($\int^+ x.$ f (*Inl* x) ∂M) + ($\int^+ x.$ f (*Inr* x) ∂N)
using *assms*
proof *induction*
case (*cong f g*)

moreover hence $\bigwedge x.\ x \in space\ M \Longrightarrow f\ (Inl\ x) = g\ (Inl\ x)$
$\bigwedge x.\ x \in space\ N \Longrightarrow f\ (Inr\ x) = g\ (Inr\ x)$
by(*auto simp*: *space-copair-measure*)
ultimately show *?case*
by(*simp cong*: *nn-integral-cong*)
next
case [*measurable*]:(*set A*)
note [*measurable*] = *measurable-vimage-Inl*[*of - M N*] *measurable-vimage-Inr*[*of - M N*]
show *?case*
by (*simp add*: *indicator-vimage*[*symmetric*] *emeasure-copair-measure*)
next
case (*mult u c*)
then show *?case*
by(*simp add*: *measurable-copair-measure-iff nn-integral-cmult distrib-left*)
next
case (*add u v*)
then show *?case*
by(*simp add*: *nn-integral-add*)
next
case *h*[*measurable*]:(*seq U*)
have *inc*:$\bigwedge x.\ incseq\ (\lambda i.\ U\ i\ x)$
by (*metis h*(*3*) *incseq-def le-funE*)
have *lim*:$(\lambda i.\ U\ i\ x) \longrightarrow Sup\ (range\ U)\ x$ **for** x
by (*metis SUP-apply LIMSEQ-SUP*[*OF inc*[*of x*]])
have $(\lambda i.\ (\int^+ x.\ U\ i\ x\ \partial(M \bigoplus_M N))) \longrightarrow (\int^+ x.\ (Sup\ (range\ U))\ x\ \partial(M \bigoplus_M N))$
by(*intro nn-integral-LIMSEQ*[*OF - - lim*]) (*auto simp*: *h*)
moreover have $(\lambda i.\ (\int^+ x.\ U\ i\ x\ \partial(M \bigoplus_M N))) \longrightarrow (\int^+ x.\ Sup\ (range\ U)\ (Inl\ x)\ \partial M) + (\int^+ x.\ Sup\ (range\ U)\ (Inr\ x)\ \partial N)$
proof −
have $(\lambda i.\ (\int^+ x.\ U\ i\ x\ \partial(M \bigoplus_M N))) = (\lambda i.\ (\int^+ x.\ U\ i\ (Inl\ x)\ \partial M) + (\int^+ x.\ U\ i\ (Inr\ x)\ \partial N))$
by(*simp add*: *h*)
also have ... $\longrightarrow (\int^+ x.\ Sup\ (range\ U)\ (Inl\ x)\ \partial M) + (\int^+ x.\ Sup\ (range\ U)\ (Inr\ x)\ \partial N)$
proof(*rule tendsto-add*)
have *inc*:$\bigwedge x.\ incseq\ (\lambda i.\ U\ i\ (Inl\ x))$
by (*metis h*(*3*) *incseq-def le-funE*)
have *lim*:$(\lambda i.\ U\ i\ (Inl\ x)) \longrightarrow Sup\ (range\ U)\ (Inl\ x)$ **for** x
by (*metis SUP-apply LIMSEQ-SUP*[*OF inc*[*of x*]])
show $(\lambda i.\ (\int^+ x.\ U\ i\ (Inl\ x)\ \partial M)) \longrightarrow (\int^+ x.\ Sup\ (range\ U)\ (Inl\ x)\ \partial M)$
using *inc* **by**(*intro nn-integral-LIMSEQ*[*OF - - lim*]) (*auto simp*: *incseq-def intro*!: *le-funI*)
next
have *inc*:$\bigwedge x.\ incseq\ (\lambda i.\ U\ i\ (Inr\ x))$
by (*metis h*(*3*) *incseq-def le-funE*)
have *lim*:$(\lambda i.\ U\ i\ (Inr\ x)) \longrightarrow Sup\ (range\ U)\ (Inr\ x)$ **for** x

by (*metis SUP-apply LIMSEQ-SUP*[*OF inc*[*of x*]])
show ($\lambda i.$ ($\int^+ x.$ *U i* (*Inr x*) ∂N)) $\longrightarrow$ ($\int^+ x.$ *Sup* (*range U*) (*Inr x*) ∂N)
using *inc* **by**(*intro nn-integral-LIMSEQ*[*OF - - lim*]) (*auto simp*: *incseq-def intro*!: *le-funI*)
qed
finally show *?thesis* **.**
qed
ultimately show *?case*
using *LIMSEQ-unique* **by** *blast*
qed

2.6 Integrability

lemma *integrable-copair-measure-iff*:
fixes f :: $'a$ + $'b \Rightarrow$ $'c$::{*banach*, *second-countable-topology*}
shows *integrable* ($M \bigoplus_M N$) $f \longleftrightarrow$ *integrable M* ($\lambda x.$ f (*Inl x*)) $\wedge$ *integrable N* ($\lambda x.$ f (*Inr x*))
by(*auto simp add*: *measurable-copair-measure-iff nn-integral-copair-measure integrable-iff-bounded*)

corollary *interable-copair-measureI*:
fixes f :: $'a$ + $'b \Rightarrow$ $'c$::{*banach*, *second-countable-topology*}
shows *integrable M* ($\lambda x.$ f (*Inl x*)) $\Longrightarrow$ *integrable N* ($\lambda x.$ f (*Inr x*)) $\Longrightarrow$ *integrable* ($M \bigoplus_M N$) f
by(*simp add*: *integrable-copair-measure-iff*)

2.7 The Lebesgue Integral

lemma *integral-copair-measure*:
fixes f :: $'a$ + $'b \Rightarrow$ $'c$::{*banach*, *second-countable-topology*}
assumes *integrable* ($M \bigoplus_M N$) f
shows ($\int x.$ $f\,x$ $\partial(M \bigoplus_M N)$) = ($\int x.$ f (*Inl x*) ∂M) + ($\int x.$ f (*Inr x*) ∂N)
using *assms*
proof *induction*
case *h*[*measurable*]:(*base A c*)
note [*measurable*] = *measurable-vimage-Inl*[*of - M N*] *measurable-vimage-Inr*[*of - M N*]
have [*simp*]:*integrable* ($M \bigoplus_M N$) (*indicat-real A*) *integrable M* (*indicat-real* (*Inl* −‘ *A*))
integrable N (*indicat-real* (*Inr* −‘ *A*))
using *h*(*2*) **by**(*auto simp*: *emeasure-copair-measure*)
show *?case*
by(*cases c* = *0*)
(*simp-all add*: *indicator-vimage*[*symmetric*] *measure-copair-measure measure-copair-measure*[*OF - h*(*2*)] *scaleR-left-distrib*)
next
case (*add f g*)
then show *?case*
by(*simp add*: *integrable-copair-measure-iff*)

next
  case *ih*:(*lim f s*)
  have $(\lambda n.\ (\int x.\ s\ n\ x\ \partial(M \bigoplus_M N))) \longrightarrow (\int x.\ f\ x\ \partial(M \bigoplus_M N))$
    using *ih*(*1*−*4*) **by**(*auto intro*!: *integral-dominated-convergence*[**where** *w*=λ*x*. *2* ∗ *norm* (*f x*)])
  moreover have $(\lambda n.\ (\int x.\ s\ n\ x\ \partial(M \bigoplus_M N))) \longrightarrow (\int x.\ f\ (Inl\ x)\ \partial M) + (\int x.\ f\ (Inr\ x)\ \partial N)$
    using *ih*(*1*−*4*)
    by(*auto intro*!: *integral-dominated-convergence*[**where** *w*=λ*x*. *2* ∗ *norm* (*f* (*Inl x*))]
                integral-dominated-convergence[**where** *w*=λ*x*. *2* ∗ *norm* (*f* (*Inr x*))] *tendsto-add*
            simp: *ih*(*5*) *integrable-copair-measure-iff measurable-copair-measure-iff*
                borel-measurable-integrable space-copair-measure InlI InrI)
  ultimately show *?case*
    using *LIMSEQ-unique* **by** *blast*
qed

3 Coproduct Measures

definition *coPiM* :: [′*i set*, ′*i* ⇒ ′*a measure*] ⇒ (′*i* × ′*a*) *measure* **where**
coPiM I Mi ≡ *measure-of*
        (*SIGMA i*:*I*. *space* (*Mi i*))
        {*A*. *A*⊆(*SIGMA i*:*I*. *space* (*Mi i*)) ∧ (∀ *i*∈*I*. *Pair i* −‘ *A* ∈ *sets* (*Mi i*))}
        $(\lambda A.\ (\sum_\infty i{\in}I.\ emeasure\ (Mi\ i)\ (Pair\ i\ -`\ A)))$

syntax
 -coPiM :: *pttrn* ⇒ ′*i set* ⇒ ′*a measure* ⇒ (′*i* × ′*a*) *measure* ((*3*$\amalg_M$ -∈-./ -) *10*)
translations
 $\amalg_M$ *x*∈*I*. *M* ⇌ *CONST coPiM I* (λ*x*. *M*)

3.1 The Measurable Space and Measurability

lemma
  shows *space-coPiM*: *space* (*coPiM I Mi*) = (*SIGMA i*:*I*. *space* (*Mi i*))
    and *sets-coPiM*:
     sets (*coPiM I Mi*) = *sigma-sets* (*SIGMA i*:*I*. *space* (*Mi i*)) {*A*. *A*⊆(*SIGMA i*:*I*. *space* (*Mi i*)) ∧ (∀ *i*∈*I*. *Pair i* −‘ *A* ∈ *sets* (*Mi i*))}
    and *sets-coPiM-eq*:*sets* (*coPiM I Mi*) = {*A*. *A*⊆(*SIGMA i*:*I*. *space* (*Mi i*)) ∧ (∀ *i*∈*I*. *Pair i* −‘ *A* ∈ *sets* (*Mi i*))}
proof −
  have *1*:{*A*. *A*⊆(*SIGMA i*:*I*. *space* (*Mi i*)) ∧ (∀ *i*∈*I*. *Pair i* −‘ *A* ∈ *sets* (*Mi i*))} ⊆ *Pow* (*SIGMA i*:*I*. *space* (*Mi i*))
    using *sets.sets-into-space* **by** *auto*
  show *space* (*coPiM I Mi*) = (*SIGMA i*:*I*. *space* (*Mi i*))
 and *2*:*sets* (*coPiM I Mi*)
        = *sigma-sets* (*SIGMA i*:*I*. *space* (*Mi i*)) {*A*. *A*⊆(*SIGMA i*:*I*. *space* (*Mi i*)) ∧ (∀ *i*∈*I*. *Pair i* −‘ *A* ∈ *sets* (*Mi i*))}

by(*auto simp*: *sets-measure-of*[*OF 1*] *space-measure-of*[*OF 1*] *coPiM-def*)
show *sets* (*coPiM I Mi*) = {*A*. *A*⊆(*SIGMA i*:*I*. *space* (*Mi i*)) ∧ (∀ *i*∈*I*. *Pair i* −' *A* ∈ *sets* (*Mi i*))}
proof −
have *sigma-algebra* (*SIGMA i*:*I*. *space* (*Mi i*)) {*A*. *A* ⊆ (*SIGMA i*:*I*. *space* (*Mi i*)) ∧ (∀ *i*∈*I*. *Pair i* −' *A* ∈ *sets* (*Mi i*))}
proof(*subst Dynkin-system.sigma-algebra-eq-Int-stable*)
show *Dynkin-system* (*SIGMA i*:*I*. *space* (*Mi i*)) {*A*. *A* ⊆ (*SIGMA i*:*I*. *space* (*Mi i*)) ∧ (∀ *i*∈*I*. *Pair i* −' *A* ∈ *sets* (*Mi i*))}
by *unfold-locales* (*auto simp*: *Pair-vimage-Sigma sets.Diff vimage-Diff vimage-Union 1*)
qed (*auto intro*!: *Int-stableI*)
thus *?thesis*
by(*auto simp*: *2 intro*!: *sigma-algebra.sigma-sets-eq*)
qed
qed

lemma *sets-coPiM-cong*:
I = *J* ⟹ (⋀*i*. *i* ∈ *I* ⟹ *sets* (*Mi i*) = *sets* (*Ni i*)) ⟹ *sets* (*coPiM I Mi*) = *sets* (*coPiM J Ni*)
by(*simp cong*: *sets-eq-imp-space-eq Sigma-cong add*: *sets-coPiM*)

lemma *measurable-coPiM2*:
assumes [*measurable*]:⋀*i*. *i* ∈ *I* ⟹ *f i* ∈ *Mi i* $\rightarrow_M$ *N*
shows (λ(*i*,*x*). *f i x*) ∈ *coPiM I Mi* $\rightarrow_M$ *N*
proof(*rule measurableI*)
fix *A*
assume [*measurable*]: *A* ∈ *sets N*
have [*simp*]:
⋀*i*. *i* ∈ *I*
⟹ *Pair i* −' (λ(*x*, *y*). *f x y*) −' *A* ∩ *Pair i* −' (*SIGMA i*:*I*. *space* (*Mi i*)) = *f i* −' *A* ∩ *space* (*Mi i*)
by *auto*
show (λ(*i*, *x*). *f i x*) −' *A* ∩ *space* (*coPiM I Mi*) ∈ *sets* (*coPiM I Mi*)
by(*auto simp*: *sets-coPiM space-coPiM*)
qed(*auto simp*: *space-coPiM measurable-space*[*OF assms*])

lemma *measurable-Pair-coPiM*[*measurable* (*raw*)]:
assumes *i* ∈ *I*
shows *Pair i* ∈ *Mi i* $\rightarrow_M$ *coPiM I Mi*
proof(*rule measurable-sigma-sets*)
show {*A*. *A*⊆(*SIGMA i*:*I*. *space* (*Mi i*)) ∧ (∀ *i*∈*I*. *Pair i* −' *A* ∈ *sets* (*Mi i*))}
⊆ *Pow* (*SIGMA i*:*I*. *space* (*Mi i*))
by *blast*
qed (*auto simp*: *assms sets-coPiM*)

lemma *measurable-Pair-coPiM'*:
assumes *i* ∈ *I* (λ(*i*,*x*). *f i x*) ∈ *coPiM I Mi* $\rightarrow_M$ *N*
shows *f i* ∈ *Mi i* $\rightarrow_M$ *N*

using *measurable-compose*[*OF measurable-Pair-coPiM assms*(*2*)] *assms*(*1*) **by** *fast*

lemma *measurable-copair-iff*: $(\lambda(i,x).\ f\ i\ x) \in coPiM\ I\ Mi \rightarrow_M N \longleftrightarrow (\forall i{\in}I.\ f\ i \in Mi\ i \rightarrow_M N)$
by(*auto intro*!: *measurable-coPiM2 simp*: *measurable-Pair-coPiM'*)

lemma *measurable-copair-iff'*: $f \in coPiM\ I\ Mi \rightarrow_M N \longleftrightarrow (\forall i{\in}I.\ (\lambda x.\ f\ (i,\ x)) \in Mi\ i \rightarrow_M N)$
using *measurable-copair-iff*[*of curry f*] **by**(*simp add*: *split-beta' curry-def*)

lemma *coPair-inverse-space-unit*:
$i \in I \Longrightarrow A \in sets\ (coPiM\ I\ Mi) \Longrightarrow Pair\ i\ -`\ A \cap space\ (Mi\ i) = Pair\ i\ -`\ A$
using *sets.sets-into-space* **by**(*fastforce simp*: *space-coPiM*)

lemma *measurable-Pair-vimage*:
assumes $i \in I\ A \in sets\ (coPiM\ I\ Mi)$
shows $Pair\ i\ -`\ A \in sets\ (Mi\ i)$
using *measurable-sets*[*OF measurable-Pair-coPiM*[*OF assms*(*1*)] *assms*(*2*)]
by (*simp add*: *coPair-inverse-space-unit*[*OF assms*])

lemma *measurable-Sigma-singleton*[*measurable* (*raw*)]:
$\bigwedge i\ A.\ i \in I \Longrightarrow A \in sets\ (Mi\ i) \Longrightarrow \{i\} \times A \in sets\ (coPiM\ I\ Mi)$
using *sets.sets-into-space sets-coPiM* **by** *fastforce*

lemma *sets-coPiM-countable*:
assumes *countable I*
shows $sets\ (coPiM\ I\ Mi) = sigma\text{-}sets\ (SIGMA\ i{:}I.\ space\ (Mi\ i))\ (\bigcup i{\in}I.\ (\times)\ \{i\}\ `\ (sets\ (Mi\ i)))$
unfolding *sets-coPiM*
proof(*safe intro*!: *sigma-sets-eqI*)
fix *a*
assume $h{:}a \subseteq (SIGMA\ i{:}I.\ space\ (Mi\ i))\ \forall i{\in}I.\ Pair\ i\ -`\ a \in sets\ (Mi\ i)$
then have $a = (\bigcup i{\in}I.\ \{i\} \times Pair\ i\ -`\ a)$
by *auto*
moreover have $(\bigcup i{\in}I.\ \{i\} \times Pair\ i\ -`\ a) \in sigma\text{-}sets\ (SIGMA\ i{:}I.\ space\ (Mi\ i))\ (\bigcup i{\in}I.\ (\times)\ \{i\}\ `\ (sets\ (Mi\ i)))$
using *h*(*2*) **by**(*auto intro*!: *sigma-sets-UNION*[*OF countable-image*[*OF assms*]])
ultimately show $a \in sigma\text{-}sets\ (SIGMA\ i{:}I.\ space\ (Mi\ i))\ (\bigcup i{\in}I.\ (\times)\ \{i\}\ `\ (sets\ (Mi\ i)))$
by *argo*
qed(*use sets.sets-into-space* **in** *fastforce*)

lemma *measurable-coPiM1'*:
assumes *countable I*
and [*measurable*]: $a \in N \rightarrow_M count\text{-}space\ I\ \bigwedge i.\ i \in a\ `\ (space\ N) \Longrightarrow g\ i \in N \rightarrow_M Mi\ i$
shows $(\lambda x.\ (a\ x,\ g\ (a\ x)\ x)) \in N \rightarrow_M coPiM\ I\ Mi$
proof(*safe intro*!: *measurable-sigma-sets*[*OF sets-coPiM-countable*[*OF assms*(*1*)]])

fix *i B*
assume *iB*[*measurable*]:*i* ∈ *I B* ∈ *sets* (*Mi i*)
show (λ*x*. (*a x*, *g* (*a x*) *x*)) −‘ ({*i*} × *B*) ∩ *space N* ∈ *sets N*
proof(*cases i* ∈ *a* ‘ (*space N*))
assume [*measurable*]:*i* ∈ *a* ‘ (*space N*)
have (λ*x*. (*a x*, *g* (*a x*) *x*)) −‘ ({*i*} × *B*) ∩ *space N* = (*a* −‘ {*i*} ∩ *space N*) ∩ (*g i* −‘ *B* ∩ *space N*)
by *auto*
also have ... ∈ *sets N*
by *simp*
finally show *?thesis* **.**
next
assume*i* ∉ *a* ‘ (*space N*)
then have (λ*x*. (*a x*, *g* (*a x*) *x*)) −‘ ({*i*} × *B*) ∩ *space N* = {}
using *measurable-space*[*OF assms*(*2*)] **by** *blast*
thus *?thesis*
by *simp*
qed
qed(*use measurable-space*[*OF assms*(*2*)] *measurable-space*[*OF assms*(*3*)] *sets.sets-into-space* **in** *fastforce*)+

lemma *measurable-coPiM1*:
assumes *countable I*
and *a* ∈ *N* $\rightarrow_M$ *count-space I* ⋀*i*. *i* ∈ *I* ⟹ *g i* ∈ *N* $\rightarrow_M$ *Mi i*
shows (λ*x*. (*a x*, *g* (*a x*) *x*)) ∈ *N* $\rightarrow_M$ *coPiM I Mi*
using *measurable-space*[*OF assms*(*2*)] **by**(*auto intro*!: *measurable-coPiM1′ assms*)

lemma *measurable-coPiM1-elements*:
assumes *countable I* **and** [*measurable*]:*f* ∈ *N* $\rightarrow_M$ *coPiM I Mi*
obtains *a g*
where *a* ∈ *N* $\rightarrow_M$ *count-space I*
⋀*i*. *i* ∈ *I* ⟹ *space* (*Mi i*) ≠ {} ⟹ *g i* ∈ *N* $\rightarrow_M$ *Mi i*
f = (λ*x*. (*a x*, *g* (*a x*) *x*))
proof −
define *a* **where** *a* ≡ *fst* ∘ *f*
have *1*[*measurable*]:*a* ∈ *N* $\rightarrow_M$ *count-space I*
proof(*safe intro*!: *measurable-count-space-eq-countable*[*THEN iffD2*] *assms*)
fix *i*
assume *i*:*i* ∈ *I*
have *a* −‘ {*i*} ∩ *space N* = *f* −‘ ({*i*} × *space* (*Mi i*)) ∩ *space N*
using *measurable-space*[*OF assms*(*2*)] **by**(*fastforce simp*: *a-def space-coPiM*)
also have ... ∈ *sets N*
using *i* **by** *auto*
finally show *a* −‘ {*i*} ∩ *space N* ∈ *sets N* **.**
next
show ⋀*x*. *x* ∈ *space N* ⟹ *a x* ∈ *I*
using *measurable-space*[*OF assms*(*2*)] **by**(*fastforce simp*: *space-coPiM a-def*)
qed
define *g* **where** *g* ≡ (λ*i x*. *if a x* = *i then snd* (*f x*) *else* (*SOME y*. *y* ∈ *space*

$(Mi\ i)))$
have $2{:}g\ i \in N \to_M Mi\ i$ **if** $i{:}i \in I$ **and** $ne{:}space\ (Mi\ i) \neq \{\}$ **for** i
unfolding *g-def*
proof(*safe intro*!: *measurable-If-restrict-space-iff*[*THEN iffD2*] *measurable-const some-in-eq*[*THEN iffD2*] *ne*)
show $(\lambda x.\ snd\ (f\ x)) \in restrict\text{-}space\ N\ \{x.\ a\ x = i\} \to_M Mi\ i$
proof(*safe intro*!: *measurableI*)
show $\bigwedge x.\ x \in space\ (restrict\text{-}space\ N\ \{x.\ a\ x = i\}) \Longrightarrow snd\ (f\ x) \in space\ (Mi\ i)$
using *measurable-space*[*OF assms*(*2*)] **by**(*fastforce simp*: *space-restrict-space a-def space-coPiM*)
next
fix A
assume [*measurable*]:$A \in sets\ (Mi\ i)$
have $(\lambda x.\ snd\ (f\ x)) -` A \cap space\ (restrict\text{-}space\ N\ \{x.\ a\ x = i\}) = f -` (\{i\} \times A) \cap space\ N$
using *i measurable-space*[*OF assms*(*2*)] **by**(*fastforce simp*: *space-restrict-space a-def space-coPiM*)
also have ... $\in sets\ N$
using *i* **by** *simp*
finally show $(\lambda x.\ snd\ (f\ x)) -` A \cap space\ (restrict\text{-}space\ N\ \{x.\ a\ x = i\})$
$\in sets\ (restrict\text{-}space\ N\ \{x.\ a\ x = i\})$
by(*auto simp*: *sets-restrict-space space-restrict-space*)
qed
qed(*use i ne* **in** *auto*)
have $3{:}f = (\lambda x.\ (a\ x,\ g\ (a\ x)\ x))$
by(*auto simp*: *a-def g-def*)
show *?thesis*
using *1 2 3 that* **by** *blast*
qed

3.2 Measures

lemma *emeasure-coPiM*:
assumes $A \in sets\ (coPiM\ I\ Mi)$
shows $emeasure\ (coPiM\ I\ Mi)\ A = (\sum_\infty i{\in}I.\ emeasure\ (Mi\ i)\ (Pair\ i -` A))$
proof(*rule emeasure-measure-of*)
show $\{A.\ A \subseteq (SIGMA\ i{:}I.\ space\ (Mi\ i)) \wedge (\forall i{\in}I.\ Pair\ i -` A \in sets\ (Mi\ i))\} \subseteq Pow\ (SIGMA\ i{:}I.\ space\ (Mi\ i))$
by *blast*
next
note *measurable-Pair-vimage*[*of - I - Mi*,*measurable* (*raw*)]
show $countably\text{-}additive\ (sets\ (coPiM\ I\ Mi))\ (\lambda a.\ \sum_\infty i{\in}I.\ emeasure\ (Mi\ i)\ (Pair\ i -` a))$
unfolding *countably-additive-def*
proof *safe*
fix $A :: nat \Rightarrow -$
assume $A{:}range\ A \subseteq sets\ (coPiM\ I\ Mi)\ disjoint\text{-}family\ A$
then have [*measurable*]:$\bigwedge n.\ A\ n \in sets\ (coPiM\ I\ Mi)$

by *blast*
show $(\sum n.\ \sum_\infty i{\in}I.\ emeasure\ (Mi\ i)\ (Pair\ i\ -`\ A\ n))$
$= (\sum_\infty i{\in}I.\ emeasure\ (Mi\ i)\ (Pair\ i\ -`\ \bigcup\ (range\ A)))$ (**is** *?lhs* = *?rhs*)
proof −
have $?lhs = (\sum_\infty n{\in}UNIV.\ \sum_\infty i{\in}I.\ emeasure\ (Mi\ i)\ (Pair\ i\ -`\ A\ n))$
by(*auto intro*!: *infsum-eq-suminf*[*symmetric*] *nonneg-summable-on-complete*)
also have ... $= (\sum_\infty i{\in}I.\ \sum_\infty n{\in}UNIV.\ emeasure\ (Mi\ i)\ (Pair\ i\ -`\ A\ n))$
by(*rule infsum-swap-ennreal*)
also have ... = *?rhs*
proof(*rule infsum-cong*)
fix *i*
assume $i \in I$
then have $(\sum n.\ Mi\ i\ (Pair\ i\ -`\ A\ n)) = Mi\ i\ (\bigcup n.\ Pair\ i\ -`\ A\ n)$
using *A*(*2*) **by**(*intro suminf-emeasure*) (*auto simp*: *disjoint-family-on-def*)
also have ... $= Mi\ i\ (Pair\ i\ -`\ \bigcup\ (range\ A))$
by (*metis vimage-UN*)
finally show $(\sum_\infty n.\ emeasure\ (Mi\ i)\ (Pair\ i\ -`\ A\ n)) = emeasure\ (Mi\ i)$ $(Pair\ i\ -`\ \bigcup\ (range\ A))$
by(*auto simp*: *infsum-eq-suminf*[*OF nonneg-summable-on-complete*])
qed
finally show *?thesis* **.**
qed
qed
next
show $A \in sets\ (coPiM\ I\ Mi)$
by *fact*
qed(*auto simp*: *positive-def coPiM-def*)

corollary *emeasure-coPiM-space*:
$emeasure\ (coPiM\ I\ Mi)\ (space\ (coPiM\ I\ Mi)) = (\sum_\infty i{\in}I.\ emeasure\ (Mi\ i)\ (space\ (Mi\ i)))$
proof −
have [*simp*]: $\bigwedge i.\ i \in I \Longrightarrow Pair\ i\ -`\ space\ (coPiM\ I\ Mi) = space\ (Mi\ i)$
by(*auto simp*: *space-coPiM*)
show *?thesis*
by(*auto simp*: *emeasure-coPiM intro*!: *infsum-cong*)
qed

lemma *emeasure-coPiM-coproj*:
assumes [*measurable*]: $i \in I$ $A \in sets\ (Mi\ i)$
shows $emeasure\ (coPiM\ I\ Mi)\ (\{i\} \times A) = emeasure\ (Mi\ i)\ A$
proof −
have $emeasure\ (coPiM\ I\ Mi)\ (\{i\} \times A) = (\sum_\infty j{\in}I.\ emeasure\ (Mi\ j)\ (if\ j = i\ then\ A\ else\ \{\}))$
by(*simp add*: *emeasure-coPiM*)
also have ... $= (\sum_\infty j{\in}(I - \{i\}) \cup \{i\}.\ emeasure\ (Mi\ j)\ (if\ j = i\ then\ A\ else\ \{\}))$
by(*rule arg-cong*[**where** *f*=*infsum* -]) (*use assms* **in** *auto*)
also have ... $= (\sum_\infty j{\in}I - \{i\}.\ emeasure\ (Mi\ j)\ (if\ j = i\ then\ A\ else\ \{\}))$

```
        + (∑∞j∈{i}. emeasure (Mi j) (if j = i then A else {}))
    by(rule infsum-Un-disjoint) (auto intro!: nonneg-summable-on-complete)
  also have ... = emeasure (Mi i) A
  proof −
    have (∑∞j∈I − {i}. emeasure (Mi j) (if j = i then A else {})) = 0
      by (rule infsum-0) simp
    thus ?thesis by simp
  qed
  finally show ?thesis .
qed

lemma measure-coPiM-coproj: i ∈ I ⟹ A ∈ sets (Mi i) ⟹ measure (coPiM I
Mi) ({i} × A) = measure (Mi i) A
  by(simp add: emeasure-coPiM-coproj measure-def)

lemma emeasure-coPiM-less-top-summable:
  assumes [measurable]:A ∈ sets (coPiM I Mi) emeasure (coPiM I Mi) A < ∞
  shows(λi. measure (Mi i) (Pair i −‘ A)) summable-on I
proof −
  have ∗:(∑∞i∈I. emeasure (Mi i) (Pair i −‘ A)) < top
    using assms(2) by(simp add: emeasure-coPiM)
 from infsum-less-top-dest[OF this] have ifin:⋀i. i ∈ I ⟹ emeasure (Mi i) (Pair
i −‘ A) < top
    by simp
  with ∗ have (∑∞i∈I. ennreal (measure (Mi i) (Pair i −‘ A))) < top
   by (metis (mono-tags, lifting) emeasure-eq-ennreal-measure infsum-cong top.not-eq-extremum)
  thus ?thesis
    by(auto intro!: bounded-infsum-summable)
qed

lemma measure-coPiM:
  assumes [measurable]:A ∈ sets (coPiM I Mi) emeasure (coPiM I Mi) A < ∞
  shows measure (coPiM I Mi) A = (∑∞i∈I. measure (Mi i) (Pair i −‘ A))
proof(subst ennreal-inj[symmetric])
  have ∗:(∑∞i∈I. emeasure (Mi i) (Pair i −‘ A)) < top
    using assms(2) by(simp add: emeasure-coPiM)
 from infsum-less-top-dest[OF this] have ifin:⋀i. i ∈ I ⟹ emeasure (Mi i) (Pair
i −‘ A) < top
    by simp
   show ennreal (measure (coPiM I Mi) A) = ennreal (∑∞i∈I. measure (Mi i)
(Pair i −‘ A)) (is ?lhs = ?rhs)
  proof −
    have ?lhs = emeasure (coPiM I Mi) A
      using assms by(auto intro!: emeasure-eq-ennreal-measure[symmetric])
    also have ... = (∑∞i∈I. emeasure (Mi i) (Pair i −‘ A))
      by(simp add: emeasure-coPiM)
    also have ... = (∑∞i∈I. ennreal (measure (Mi i) (Pair i −‘ A)))
      using ifin by(fastforce intro!: infsum-cong  emeasure-eq-ennreal-measure)
    also have ... = ?rhs
```

by(*simp add*: *infsum-ennreal-eq*[*OF emeasure-coPiM-less-top-summable*[*OF assms*]])
finally show *?thesis* **.**
qed
qed(*auto intro*!: *infsum-nonneg*)

3.3 Non-Negative Integral

lemma *nn-integral-coPiM*:
assumes $f \in$ *borel-measurable* (*coPiM I Mi*)
shows $(\int^+ x.\ f\ x\ \partial coPiM\ I\ Mi) = (\sum_\infty i{\in}I.\ (\int^+ x.\ f\ (i,\ x)\ \partial Mi\ i))$
using *assms*
proof *induction*
case (*cong f g*)
moreover hence $\bigwedge i\ x.\ i \in I \Longrightarrow x \in space\ (Mi\ i) \Longrightarrow f\ (i,\ x) = g\ (i,\ x)$
by(*auto simp*: *space-coPiM*)
ultimately show *?case*
by(*simp cong*: *nn-integral-cong infsum-cong*)
next
case [*measurable*]:(*set A*)
note [*measurable*] = *measurable-Pair-vimage*[*OF - this*]
show *?case*
by(*simp add*: *indicator-vimage*[*symmetric*] *emeasure-coPiM cong*: *infsum-cong*)
next
case (*add u v*)
then show *?case*
by(*simp add*: *nn-integral-add infsum-add nonneg-summable-on-complete cong*: *infsum-cong*)
next
case (*mult u c*)
then show *?case*
by(*simp add*: *nn-integral-cmult infsum-cmult-right-ennreal cong*: *infsum-cong*)
next
case *ih*[*measurable*]:(*seq U*)
show *?case* (**is** *?lhs* = *?rhs*)
proof −
have $?lhs = (\int^+ x.\ (SUP\ j.\ U\ j\ x)\ \partial coPiM\ I\ Mi)$
by(*auto intro*!: *nn-integral-cong simp*: *SUP-apply*[*symmetric*])
also have ... $= (SUP\ j.\ (\int^+ x.\ U\ j\ x\ \partial coPiM\ I\ Mi))$
by(*auto intro*!: *nn-integral-monotone-convergence-SUP ih*(*3*))
also have ... $= (SUP\ j.\ (\sum_\infty i{\in}I.\ (\int^+ x.\ U\ j\ (i,\ x)\ \partial Mi\ i)))$
by(*simp add*: *ih*)
also have ... $= (\sum_\infty i{\in}I.\ (SUP\ j.\ (\int^+ x.\ U\ j\ (i,\ x)\ \partial Mi\ i)))$
using *ih*(*3*) **by**(*auto intro*!: *ennreal-infsum-Sup-eq*[*symmetric*] *incseq-nn-integral simp*: *mono-compose*)
also have ... $= (\sum_\infty i{\in}I.\ (\int^+ x.\ (SUP\ j.\ U\ j\ (i,\ x))\ \partial Mi\ i))$
using *ih*(*3*) **by**(*auto intro*!: *infsum-cong nn-integral-monotone-convergence-SUP*[*symmetric*] *mono-compose*)
also have ... = *?rhs*

by(*simp add*: *SUP-apply*[*symmetric*])
finally show *?thesis* **.**
qed
qed

3.4 Integrability

lemma
fixes f :: - $\Rightarrow$ $'b$::{*banach*, *second-countable-topology*}
assumes *integrable* (*coPiM I Mi*) *f*
shows *integrable-coPiM-dest-sum*:$(\sum_\infty i\in I.\ (\int^+ x.\ norm\ (f\ (i,\ x))\ \partial Mi\ i)) < \infty$
and *integrable-coPiM-dest-integrable*: $\bigwedge i.\ i \in I \Longrightarrow integrable\ (Mi\ i)\ (\lambda x.\ f\ (i,\ x))$
and *integrable-coPiM-summable-norm*: $(\lambda i.\ (\int x.\ norm\ (f\ (i,\ x))\ \partial Mi\ i))$ *summable-on I*
and *integrable-coPiM-abs-summable*: *Infinite-Sum.abs-summable-on* $(\lambda i.\ (\int x.\ f\ (i,\ x)\ \partial Mi\ i))$ *I*
and *integrable-coPiM-summable*: $(\lambda i.\ (\int x.\ f\ (i,\ x)\ \partial Mi\ i))$ *summable-on I*
proof −
show *fin*:$(\sum_\infty i\in I.\ (\int^+ x.\ norm\ (f\ (i,\ x))\ \partial Mi\ i)) < \infty$
using *assms* **by**(*auto simp*: *integrable-iff-bounded nn-integral-coPiM*)
thus *integ*:$i \in I \Longrightarrow integrable\ (Mi\ i)\ (\lambda x.\ f\ (i,\ x))$ **for** *i*
using *assms* **by**(*auto simp*: *integrable-iff-bounded intro*!: *infsum-less-top-dest*[*of* - - *i*])
show *summable*:$(\lambda i.\ (\int x.\ norm\ (f\ (i,\ x))\ \partial Mi\ i))$ *summable-on I*
using *nn-integral-eq-integral*[*OF integrable-norm*[*OF integ*]] *fin*
by(*auto intro*!: *bounded-infsum-summable cong*: *infsum-cong*)
show *Infinite-Sum.abs-summable-on* $(\lambda i.\ (\int x.\ f\ (i,\ x)\ \partial Mi\ i))$ *I*
by(*rule summable-on-comparison-test*[*OF summable*]) *auto*
thus $(\lambda i.\ (\int x.\ f\ (i,\ x)\ \partial Mi\ i))$ *summable-on I*
using *abs-summable-summable* **by** *fastforce*
qed

3.5 The Lebesgue Integral

lemma *integral-coPiM*:
fixes f :: - $\Rightarrow$ $'b$::{*banach*, *second-countable-topology*}
assumes *integrable* (*coPiM I Mi*) *f*
shows $(\int x.\ f\ x\ \partial coPiM\ I\ Mi) = (\sum_\infty i\in I.\ (\int x.\ f\ (i,\ x)\ \partial Mi\ i))$
using *assms*
proof *induction*
case *h*[*measurable*]:(*base A c*)
note [*measurable*] = *measurable-Pair-vimage*[*OF - this*(*1*)]
have [*simp*]: *integrable* (*coPiM I Mi*) (*indicat-real A*)
$\bigwedge i.\ i \in I \Longrightarrow integrable\ (Mi\ i)\ (indicat\text{-}real\ (Pair\ i\ -`\ A))$
using *h*(*2*) **by**(*auto simp*: *emeasure-coPiM dest*: *infsum-less-top-dest*)
show *?case*
using *h*(*2*) *emeasure-coPiM-less-top-summable*[*OF h*]
by(*cases* $c = 0$)

(*auto simp*: *measure-coPiM indicator-vimage*[*symmetric*] *infsum-scaleR-left*[*symmetric*] *cong*: *infsum-cong*)
next
 case *h*:(*add f g*)
 show *?case* (**is** *?lhs* = *?rhs*)
 proof −
  have *?lhs* = ($\sum_\infty i{\in}I.$ ($\int x.\ f\ (i,\ x)\ \partial Mi\ i$)) + ($\sum_\infty i{\in}I.$ ($\int x.\ g\ (i,\ x)\ \partial Mi\ i$))
   using *h* **by** *simp*
  also have ... = ($\sum_\infty i{\in}I.$ ($\int x.\ f\ (i,\ x)\ \partial Mi\ i$) + ($\int x.\ g\ (i,\ x)\ \partial Mi\ i$))
   by(*auto intro*!: *infsum-add*[*symmetric*] *integrable-coPiM-summable h*)
  also have ... = *?rhs*
   using *h*
   by(*auto intro*!: *infsum-cong Bochner-Integration.integral-add*[*symmetric*] *integrable-coPiM-dest-integrable*)
  finally show *?thesis* **.**
 qed
next
 case *ih*:(*lim f fn*)
 note [*measurable*,*simp*] = *ih*(*1*−*4*)
 show *?case* (**is** *?lhs* = *?rhs*)
 proof −
  have *?lhs* = *lim* ($\lambda n.$ ($\int x.\ fn\ n\ x\ \partial(coPiM\ I\ Mi)$))
   by(*auto intro*!: *limI*[*symmetric*] *integral-dominated-convergence*[**where** *w*=$\lambda x.$ *2* ∗ *norm* (*f x*)])
  also have ... = *lim* ($\lambda n.$ ($\sum_\infty i{\in}I.$ ($\int x.\ fn\ n\ (i,\ x)\ \partial Mi\ i$)))
   by(*simp add*: *ih*(*5*))
  also have ... = *lim* ($\lambda n.$ ($\int i.$ ($\int x.\ fn\ n\ (i,\ x)\ \partial Mi\ i$)$\partial$*count-space I*))
   by(*simp add*: *integrable-coPiM-abs-summable infsum-eq-integral*)
  also have ... = ($\int i.$ ($\int x.\ f\ (i,\ x)\ \partial Mi\ i$)$\partial$*count-space I*)
  proof(*intro limI integral-dominated-convergence*[**where** *w*=$\lambda i.$ ($\int x.$ *2* ∗ *norm* (*f* (*i*, *x*)) $\partial Mi\ i$)] *AE-I2*)
   show *integrable* (*count-space I*) ($\lambda i.$ ($\int x.$ *2* ∗ *norm* (*f* (*i*, *x*)) $\partial Mi\ i$))
   by(*auto simp*: *abs-summable-on-integrable-iff*[*symmetric*] *integrable-coPiM-summable-norm*[*OF ih*(*4*)])
  next
   show *i* ∈ *space* (*count-space I*) $\Longrightarrow$ ($\lambda n.$ ($\int x.\ fn\ n\ (i,\ x)\ \partial Mi\ i$)) $\longrightarrow$ ($\int x.\ f\ (i,\ x)\ \partial Mi\ i$) **for** *i*
    by(*auto intro*!: *integral-dominated-convergence*[**where** *w*=$\lambda x.$ *2*∗*norm* (*f* (*i*, *x*))] *integrable-coPiM-dest-integrable*
      simp: *space-coPiM*)
  next
   show *i* ∈ *space* (*count-space I*) $\Longrightarrow$ *norm* (($\int x.\ fn\ n\ (i,\ x)\ \partial Mi\ i$)) ≤ ($\int x.$ *2* ∗ *norm* (*f* (*i*, *x*)) $\partial Mi\ i$) **for** *n i*
    by(*rule order.trans*[**where** *b*=($\int x.$ *norm* (*fn n* (*i*, *x*)) $\partial Mi\ i$)])
     (*auto simp*: *space-coPiM*
     simp del: *Bochner-Integration.integral-mult-right-zero Bochner-Integration.integral-mult-right*
      intro!: *integral-mono integrable-coPiM-dest-integrable*)
  qed *simp-all*
  also have ... = *?rhs*

by(*simp add*: *infsum-eq-integral integrable-coPiM-abs-summable*)
finally show *?thesis* **.**
qed
qed

3.6 Finite Coproduct Measures

lemma *emeasure-coPiM-finite*:
assumes *finite I A* $\in$ *sets* (*coPiM I Mi*)
shows *emeasure* (*coPiM I Mi*) *A* = ($\sum$ *i*$\in$*I*. *emeasure* (*Mi i*) (*Pair i* −' *A*))
using *assms* **by**(*simp add*: *emeasure-coPiM*)

lemma *emeasure-coPiM-finite-space*:
finite I $\Longrightarrow$ *emeasure* (*coPiM I Mi*) (*space* (*coPiM I Mi*)) = ($\sum$ *i*$\in$*I*. *emeasure* (*Mi i*) (*space* (*Mi i*)))
by(*simp add*: *emeasure-coPiM-space*)

lemma *measure-coPiM-finite*:
assumes *finite I A* $\in$ *sets* (*coPiM I Mi*) *emeasure* (*coPiM I Mi*) *A* < ∞
shows *measure* (*coPiM I Mi*) *A* = ($\sum$ *i*$\in$*I*. *measure* (*Mi i*) (*Pair i* −' *A*))
using *assms*(*3*) **by**(*simp add*: *emeasure-coPiM-finite*[*OF assms*(*1*,*2*)] *measure-def enn2real-sum assms*(*1*))

lemma *nn-integral-coPiM-finite*:
assumes *finite I f* $\in$ *borel-measurable* (*coPiM I Mi*)
shows ($\int^+$*x*. *f x* ∂(*coPiM I Mi*)) = ($\sum$ *i*$\in$*I*. ($\int^+$*x*. *f* (*i*, *x*) ∂(*Mi i*)))
by(*simp add*: *nn-integral-coPiM assms*)

lemma *integrable-coPiM-finite-iff*:
fixes *f* :: - $\Rightarrow$ *'c*::{*banach*, *second-countable-topology*}
shows *finite I* $\Longrightarrow$ *integrable* (*coPiM I Mi*) *f* $\longleftrightarrow$ ($\forall$ *i*$\in$*I*. *integrable* (*Mi i*) (λ*x*. *f* (*i*, *x*)))
using *measurable-copair-iff'*[*of f I Mi borel*]
by(*auto simp*: *integrable-iff-bounded nn-integral-coPiM-finite*)

lemma *integral-coPiM-finite*:
fixes *f* :: - $\Rightarrow$ *'c*::{*banach*, *second-countable-topology*}
assumes *finite I integrable* (*coPiM I Mi*) *f*
shows ($\int$ *x*. *f x* ∂(*coPiM I Mi*)) = ($\sum$ *i*$\in$*I*. ($\int$ *x*. *f* (*i*, *x*) ∂(*Mi i*)))
by(*auto simp*: *assms integral-coPiM*)

3.7 Countable Infinite Coproduct Measures

lemma *emeasure-coPiM-countable-infinite*:
assumes [*measurable*]: *bij-betw from-n* (*UNIV* :: *nat set*) *I A* $\in$ *sets* (*coPiM I Mi*)
shows *emeasure* (*coPiM I Mi*) *A* = ($\sum$ *n*. *emeasure* (*Mi* (*from-n n*)) (*Pair* (*from-n n*) −' *A*))
proof −
have *I*:*countable I*

using *assms*(*1*) *countableI-bij* **by** *blast*
have [*measurable*,*simp*]:*Pair* (*from-n n*) −' *A* ∈ *sets* (*Mi* (*from-n n*)) *from-n n* ∈ *I* **for** *n*
using *bij-betwE*[*OF assms*(*1*)] **by**(*auto intro*!: *measurable-Pair-vimage*[**where** *I*=*I*])
have *emeasure* (*coPiM I Mi*) *A* = *emeasure* (*coPiM I Mi*) (⋃*n*. {*from-n n*} × *Pair* (*from-n n*) −' *A*)
using *sets.sets-into-space*[*OF assms*(*2*)] *assms*(*1*)
by(*fastforce intro*!: *arg-cong*[**where** *f*=*emeasure* (*coPiM I Mi*)]
simp: *space-coPiM bij-betw-def*)
also have ... = ($\sum$*n*. *emeasure* (*Mi* (*from-n n*)) (*Pair* (*from-n n*) −' *A*))
using *injD*[*OF bij-betw-imp-inj-on*[*OF assms*(*1*)]]
by(*subst suminf-emeasure*[*symmetric*])
(*auto simp*: *disjoint-family-on-def emeasure-coPiM-coproj intro*!: *suminf-cong*)
finally show *?thesis* **.**
qed

lemmas *emeasure-coPiM-countable-infinite'* = *emeasure-coPiM-countable-infinite*[*OF bij-betw-from-nat-into*]
lemmas *emeasure-coPiM-nat* = *emeasure-coPiM-countable-infinite*[*OF bij-id*,*simplified*]

lemma *measure-coPiM-countable-infinite*:
assumes [*measurable*,*simp*]: *bij-betw from-n* (*UNIV* :: *nat set*) *I A* ∈ *sets* (*coPiM I Mi*)
and *emeasure* (*coPiM I Mi*) *A* < ∞
shows *measure* (*coPiM I Mi*) *A* = ($\sum$*n*. *measure* (*Mi* (*from-n n*)) (*Pair* (*from-n n*) −' *A*)) (**is** *?lhs* = *?rhs*)
and *summable* (λ*n*. *measure* (*Mi* (*from-n n*)) (*Pair* (*from-n n*) −' *A*))
proof −
have *ennreal ?lhs* = *emeasure* (*coPiM I Mi*) *A*
using *assms*(*3*) **by**(*auto intro*!: *emeasure-eq-ennreal-measure*[*symmetric*])
also have ... = ($\sum$*n*. *emeasure* (*Mi* (*from-n n*)) (*Pair* (*from-n n*) −' *A*))
by(*simp add*: *emeasure-coPiM-countable-infinite*)
also have ... = ($\sum$*n*. *ennreal* (*measure* (*Mi* (*from-n n*)) (*Pair* (*from-n n*) −' *A*)))
using *assms*(*3*) *ennreal-suminf-lessD top.not-eq-extremum*
by(*auto intro*!: *suminf-cong emeasure-eq-ennreal-measure*
simp: *emeasure-coPiM-countable-infinite*[*OF assms*(*1*)])
finally have ∗:*ennreal ?lhs* = ($\sum$*n*. *ennreal* (*measure* (*Mi* (*from-n n*)) (*Pair* (*from-n n*) −' *A*))) **.**
thus ∗∗[*simp*]: *summable* (λ*n*. *measure* (*Mi* (*from-n n*)) (*Pair* (*from-n n*) −' *A*))
by(*auto intro*!: *summable-suminf-not-top*)
show *?lhs* = *?rhs*
proof(*subst ennreal-inj*[*symmetric*])
have *ennreal ?lhs* = ($\sum$*n*. *ennreal* (*measure* (*Mi* (*from-n n*)) (*Pair* (*from-n n*) −' *A*)))
by *fact*
also have ... = *ennreal ?rhs*
using *assms*(*3*) **by**(*auto intro*!: *suminf-ennreal2*)

finally show *ennreal ?lhs* = *ennreal ?rhs* **.**
qed(*simp-all add*: *suminf-nonneg*)
qed

lemmas *measure-coPiM-countable-infinite′* = *measure-coPiM-countable-infinite*[*OF bij-betw-from-nat-into*]
lemmas *measure-coPiM-nat* = *measure-coPiM-countable-infinite*[*OF bij-id*,*simplified id-apply*]

lemma *nn-integral-coPiM-countable-infinite*:
assumes [*measurable*]:*bij-betw from-n* (*UNIV* :: *nat set*) *I f* ∈ *borel-measurable* (*coPiM I Mi*)
shows ($\int^+ x.\ f\ x\ \partial(coPiM\ I\ Mi)$) = ($\sum n.\ (\int^+ x.\ f\ (from\text{-}n\ n,\ x)\ \partial(Mi\ (from\text{-}n\ n)))$) (**is** - = *?rhs*)
proof −
have ($\int^+ x.\ f\ x\ \partial(coPiM\ I\ Mi)$) = ($\sum_\infty i{\in}I.\ (\int^+ x.\ f\ (i,\ x)\ \partial Mi\ i)$)
by(*simp add*: *nn-integral-coPiM*)
also have ... = ($\sum_\infty i{\in}from\text{-}n$ ‘ $UNIV.\ (\int^+ x.\ f\ (i,\ x)\ \partial Mi\ i)$)
by(*rule arg-cong*[**where** *f*=*infsum* -]) (*metis assms*(*1*) *bij-betw-def*)
also have ... = ($\sum_\infty n{\in}UNIV.\ (\int^+ x.\ f\ (from\text{-}n\ n,\ x)\ \partial(Mi\ (from\text{-}n\ n)))$)
by(*rule infsum-reindex*[*simplified comp-def*]) (*use assms*(*1*) *bij-betw-imp-inj-on* **in** *blast*)
also have ... = *?rhs*
by(*auto intro*!: *infsum-eq-suminf nonneg-summable-on-complete*)
finally show *?thesis* **.**
qed
lemmas *nn-integral-coPiM-countable-infinite′* = *nn-integral-coPiM-countable-infinite*[*OF bij-betw-from-nat-into*]
lemmas *nn-integral-coPiM-nat* = *nn-integral-coPiM-countable-infinite*[*OF bij-id*,*simplified*]

lemma
fixes *f* :: - ⇒ ′*b*::{*banach*, *second-countable-topology*}
assumes *bij-betw from-n* (*UNIV* :: *nat set*) *I integrable* (*coPiM I Mi*) *f*
shows *integrable-coPiM-countable-infinite-dest-sum*:($\sum n.\ (\int^+ x.\ norm\ (f\ (from\text{-}n\ n,\ x))\ \partial(Mi\ (from\text{-}n\ n)))) < \infty$
and *integrable-coPiM-countable-infinite-dest′*: $\bigwedge n.$ *integrable* (*Mi* (*from-n n*)) ($\lambda x.\ f\ (from\text{-}n\ n,\ x)$)
using *ennreal-suminf-lessD assms*(*1*,*2*) *bij-betwE*[*OF assms*(*1*)]
by(*auto simp*: *integrable-iff-bounded nn-integral-coPiM-countable-infinite*)

lemmas *integrable-coPiM-countable-infinite-dest-sum′* = *integrable-coPiM-countable-infinite-dest-sum*[*OF bij-betw-from-nat-into*]
lemmas *integrable-coPiM-countable-infinite-dest′′* = *integrable-coPiM-countable-infinite-dest′*[*OF bij-betw-from-nat-into*]
lemmas *integrable-coPiM-nat-dest-sum* = *integrable-coPiM-countable-infinite-dest-sum*[*OF bij-id*,*simplified id-apply*]
lemmas *integrable-coPiM-nat-dest* = *integrable-coPiM-countable-infinite-dest′*[*OF bij-id*,*simplified id-apply*]

lemma
  fixes f :: - ⇒ $'b$::{*banach*, *second-countable-topology*}
  assumes *bij-betw from-n* (*UNIV* :: *nat set*) *I* *integrable* (*coPiM I Mi*) *f*
  shows *integrable-coPiM-countable-infinite-summable-norm*: *summable* ($\lambda n.$ ($\int x.$ *norm* (*f* (*from-n n*, *x*)) ∂(*Mi* (*from-n n*))))
    and *integrable-coPiM-countable-infinite-summable-norm'*: *summable* ($\lambda n.$ *norm* ($\int x.$ *f* (*from-n n*, *x*) ∂(*Mi* (*from-n n*))))
    and *integrable-coPiM-countable-infinite-summable*: *summable* ($\lambda n.$ ($\int x.$ *f* (*from-n n*, *x*) ∂(*Mi* (*from-n n*))))
proof −
  show ∗:*summable* ($\lambda n.$ ($\int x.$ *norm* (*f* (*from-n n*, *x*)) ∂(*Mi* (*from-n n*))))
    using *integrable-coPiM-countable-infinite-dest-sum*[*OF assms*]
    nn-integral-eq-integral[*OF integrable-norm*[*OF integrable-coPiM-countable-infinite-dest'*[*OF assms*]]]
    by (*auto intro*!: *summable-suminf-not-top*)
  show *summable* ($\lambda n.$ *norm* ($\int x.$ *f* (*from-n n*, *x*) ∂(*Mi* (*from-n n*))))
    by(*rule summable-comparison-test-ev*[*OF* - ∗]) *auto*
  thus *summable* ($\lambda n.$ ($\int x.$ *f* (*from-n n*, *x*) ∂(*Mi* (*from-n n*))))
    using *summable-norm-cancel* **by** *force*
qed

lemmas *integrable-coPiM-countable-infinite-summable-norm''*
  = *integrable-coPiM-countable-infinite-summable-norm*[*OF bij-betw-from-nat-into*]
lemmas *integrable-coPiM-countable-infinite-summable-norm'''*
  = *integrable-coPiM-countable-infinite-summable-norm'*[*OF bij-betw-from-nat-into*]
lemmas *integrable-coPiM-countable-infinite-summable'*
  = *integrable-coPiM-countable-infinite-summable*[*OF bij-betw-from-nat-into*]
lemmas *integrable-coPiM-nat-summable-norm*
  = *integrable-coPiM-countable-infinite-summable-norm*[*OF bij-id*,*simplified id-apply*]
lemmas *integrable-coPiM-nat-summable-norm'*
  = *integrable-coPiM-countable-infinite-summable-norm'*[*OF bij-id*,*simplified id-apply*]
lemmas *integrable-coPiM-nat-summable*
  = *integrable-coPiM-countable-infinite-summable*[*OF bij-id*,*simplified id-apply*]

lemma
  fixes f :: - ⇒ $'b$::{*banach*, *second-countable-topology*}
  assumes *countable I infinite I integrable* (*coPiM I Mi*) *f*
  shows *integrable-coPiM-countable-infinite-dest*:$\bigwedge i.$ $i \in I \Longrightarrow$ *integrable* (*Mi i*) ($\lambda x.$ *f* (*i*, *x*))
  using *integrable-coPiM-countable-infinite-dest'*[*OF bij-betw-from-nat-into*[*OF assms*(*1*,*2*)] *assms*(*3*)]
  by (*meson assms*(*1*) *countable-all*)

lemma *integrable-coPiM-countable-infiniteI*:
  fixes f :: - ⇒ $'b$::{*banach*, *second-countable-topology*}
  assumes *bij-betw from-n* (*UNIV* :: *nat set*) *I* $\bigwedge i.$ $i \in I \Longrightarrow$ ($\lambda x.$ *f* (*i*,*x*)) $\in$ *borel-measurable* (*Mi i*)
    and ($\sum n.$ ($\int^+ x.$ *norm* (*f* (*from-n n*, *x*)) ∂(*Mi* (*from-n n*)))) $< \infty$
  shows *integrable* (*coPiM I Mi*) *f*

using *nn-integral-coPiM-countable-infinite*[*OF assms*(*1*),*of* - *Mi*] *assms*(*2*,*3*)
by(*auto simp*: *measurable-copair-iff′ integrable-iff-bounded*)

lemmas *integrable-coPiM-countable-infiniteI′* = *integrable-coPiM-countable-infiniteI*[*OF bij-betw-from-nat-into*]
lemmas *integrable-coPiM-natI* = *integrable-coPiM-countable-infiniteI*[*OF bij-id*, *simplified id-apply*]

lemma *integral-coPiM-countable-infinite*:
fixes *f* :: - ⇒ ′b::{*banach*, *second-countable-topology*}
assumes *bij-betw from-n* (*UNIV* :: *nat set*) *I integrable* (*coPiM I Mi*) *f*
shows ($\int$ *x*. *f x* ∂(*coPiM I Mi*)) = ($\sum$ *n*. ($\int$ *x*. *f* (*from-n n*, *x*) ∂(*Mi* (*from-n n*)))) (**is** *?lhs* = *?rhs*)
proof −
have *?lhs* = ($\sum_\infty$ *i*∈*I*. ($\int$ *x*. *f* (*i*, *x*) ∂*Mi i*))
by(*simp add*: *integral-coPiM assms*)
also have ... = ($\sum_\infty$ *i*∈*from-n* ' *UNIV*. ($\int$ *x*. *f* (*i*, *x*) ∂*Mi i*))
by(*rule arg-cong*[**where** *f*=*infsum* -]) (*metis assms*(*1*) *bij-betw-def*)
also have ... = ($\sum_\infty$ *n*∈*UNIV*. ($\int$ *x*. *f* (*from-n n*, *x*) ∂(*Mi* (*from-n n*))))
by(*rule infsum-reindex*[*simplified comp-def*]) (*use assms*(*1*) *bij-betw-imp-inj-on* **in** *blast*)
also have ... = *?rhs*
by(*auto intro*!: *infsum-eq-suminf norm-summable-imp-summable-on integrable-coPiM-countable-infinite-sum assms*)
finally show *?thesis* **.**
qed

lemmas *integral-coPiM-countable-infinite′* = *integral-coPiM-countable-infinite*[*OF bij-betw-from-nat-into*]
lemmas *integral-coPiM-nat* = *integral-coPiM-countable-infinite*[*OF bij-id*,*simplified id-apply*]

3.8 Finiteness

lemma *finite-measure-coPiM*:
assumes *finite I* ⋀*i*. *i* ∈ *I* ⟹ *finite-measure* (*Mi i*)
shows *finite-measure* (*coPiM I Mi*)
by(*rule finite-measureI*) (*auto simp*: *emeasure-coPiM-finite finite-measure.emeasure-finite assms*)

3.9 σ-Finiteness

lemma *sigma-finite-measure-coPiM*:
assumes *countable I* ⋀*i*. *i* ∈ *I* ⟹ *sigma-finite-measure* (*Mi i*)
shows *sigma-finite-measure* (*coPiM I Mi*)
proof
have ∃*A*. *range A* ⊆ *sets* (*Mi i*) ∧ (⋃*n*. *A n*) = *space* (*Mi i*) ∧ (∀*n*::*nat*. *emeasure* (*Mi i*) (*A n*) ≠ ∞)
if *i* ∈ *I* **for** *i*
using *sigma-finite-measure.sigma-finite*[*OF assms*(*2*)[*OF that*]] **by** *metis*

hence $\exists A. \forall i{\in}I.$ *range* $(A\ i) \subseteq$ *sets* $(Mi\ i) \wedge (\bigcup n.\ A\ i\ n) =$ *space* $(Mi\ i) \wedge$ $(\forall n{::}nat.$ *emeasure* $(Mi\ i)\ (A\ i\ n) \neq \infty)$
by *metis*
then obtain *Ai*
where *Ai*[*measurable*]: $\bigwedge i\ n.\ i \in I \Longrightarrow Ai\ i\ n \in$ *sets* $(Mi\ i)$
$\bigwedge i.\ i \in I \Longrightarrow (\bigcup n{::}nat.\ (Ai\ i\ n)) =$ *space* $(Mi\ i)$
$\bigwedge i\ n.\ i \in I \Longrightarrow$ *emeasure* $(Mi\ i)\ (Ai\ i\ n) \neq \infty$
by (*metis UNIV-I sets-range*)
show $\exists A.$ *countable* $A \wedge A \subseteq$ *sets* (*coPiM I Mi*) $\wedge \bigcup A =$ *space* (*coPiM I Mi*) $\wedge$ $(\forall a{\in}A.$ *emeasure* (*coPiM I Mi*) $a \neq \infty)$
proof(*intro exI*[**where** $x{=}\bigcup n.\ (\bigcup i \in I.\ \{\{i\} \times Ai\ i\ n\})$] *conjI ballI*)
show *countable* $(\bigcup n.\ (\bigcup i \in I.\ \{\{i\} \times Ai\ i\ n\}))$
using *assms*(*1*) **by** *auto*
next
show $(\bigcup n.\ \bigcup i{\in}I.\ \{\{i\} \times Ai\ i\ n\}) \subseteq$ *sets* (*coPiM I Mi*)
by *auto*
next
show $\bigcup\ (\bigcup n.\ \bigcup i{\in}I.\ \{\{i\} \times Ai\ i\ n\}) =$ *space* (*coPiM I Mi*)
using *sets.sets-into-space*[*OF Ai*(*1*)] *Ai*(*2*) **by**(*fastforce simp*: *space-coPiM*)
next
fix *a*
assume $a \in (\bigcup n.\ \bigcup i{\in}I.\ \{\{i\} \times Ai\ i\ n\})$
then obtain *n i* **where** *a*: $i \in I$ $a = \{i\} \times Ai\ i\ n$
by *blast*
show *emeasure* (*coPiM I Mi*) $a \neq \infty$
using *a*(*1*) *Ai*(*3*) *assms* **by**(*auto simp*: *a*(*2*) *emeasure-coPiM-coproj*)
qed
qed

end

4 Additional Properties

theory *Coproduct-Measure-Additional*
imports *Coproduct-Measure*
Standard-Borel-Spaces.StandardBorel
S-Finite-Measure-Monad.Kernels
S-Finite-Measure-Monad.Measure-QuasiBorel-Adjunction
begin

4.1 S-Finiteness

lemma *s-finite-measure-copair-measure*:
assumes *s-finite-measure M s-finite-measure N*
shows *s-finite-measure* (*copair-measure M N*)
proof −
note [*measurable*] = *measurable-vimage-Inl*[*of - M N*] *measurable-vimage-Inr*[*of - M N*]
obtain *Mi Ni* **where** [*measurable-cong*]:

$\bigwedge i.$ *sets* $(Mi\ i) =$ *sets* M $\bigwedge i.$ *finite-measure* $(Mi\ i)$ $\bigwedge A.$ $M\ A = (\sum i.\ Mi\ i\ A)$
$\bigwedge i.$ *sets* $(Ni\ i) =$ *sets* N $\bigwedge i.$ *finite-measure* $(Ni\ i)$ $\bigwedge A.$ $N\ A = (\sum i.\ Ni\ i\ A)$
by (*metis assms*(*1*) *assms*(*2*) *s-finite-measure.finite-measures*′)
thus *?thesis*
by(*auto intro*!: *s-finite-measureI*[**where** $Mi=\lambda i.\ Mi\ i \bigoplus_M Ni\ i$] *finite-measure-copair-measure*
cong: *sets-copair-measure-cong simp*: *emeasure-copair-measure sum-inf-add*)
qed

lemma *s-finite-measure-coPiM*:
assumes *countable I* $\bigwedge i.\ i \in I \Longrightarrow$ *s-finite-measure* (*Mi i*)
shows *s-finite-measure* (*coPiM I Mi*)
proof −
note *measurable-Pair-vimage*[*measurable* (*raw*)]
consider *finite I* | *infinite I countable I*
using *assms* **by** *argo*
then show *?thesis*
proof *cases*
assume *I*:*finite I*
show *?thesis*
by(*auto intro*!: *s-finite-measure-finite-sumI*[**where** $Mi=\lambda i.$ *distr* (*Mi i*) (*coPiM I Mi*) (*Pair i*)
and *I*=*I*,*OF* - *s-finite-measure.s-finite-measure-distr*[*OF assms*(*2*)]]
simp: *emeasure-distr emeasure-coPiM-finite I*)
next
assume *I*:*infinite I countable I*
then have [*simp*]:$\bigwedge n.$ *from-nat-into I n* $\in I$
by (*simp add*: *from-nat-into infinite-imp-nonempty*)
show *?thesis*
by(*auto intro*!: *s-finite-measure-s-finite-sumI*[**where**
$Mi=\lambda n.$ *distr* (*Mi* (*from-nat-into I n*)) (*coPiM I Mi*) (*Pair* (*from-nat-into I n*)),
OF - *s-finite-measure.s-finite-measure-distr*[*OF assms*(*2*)]]
simp: *emeasure-distr I emeasure-coPiM-countable-infinite*′ *coPair-inverse-space-unit*[**where** *I*=*I*])
qed
qed

4.2 Standardness

lemma *standard-borel-copair-measure*:
assumes *standard-borel M standard-borel N*
shows *standard-borel* ($M \bigoplus_M N$)
proof −
obtain *A* **where** *A*[*measurable*]: $A \in$ *sets borel* $A \subseteq \{0{<}..{<}1{::}real\}$
M measurable-isomorphic restrict-space borel A
by (*meson assms*(*1*) *greaterThanLessThan-borel linorder-not-le not-one-le-zero standard-borel.isomorphic-subset-real uncountable-open-interval*)

then obtain $f\,f'$
where f[*measurable*]: $f \in M \to_M$ *restrict-space borel A*
$f' \in$ *restrict-space borel A* $\to_M M$
$\bigwedge x.\ x \in$ *space M* $\Longrightarrow f'\ (f\ x) = x$ $\bigwedge y.\ y \in A \Longrightarrow f\ (f'\ y) = y$
using *measurable-isomorphicD*[*OF A*(*3*)] **unfolding** *space-restrict-space* **by** *fastforce*
obtain B **where** B[*measurable*]:$B \in$ *sets borel* $B \subseteq \{1{<}..{<}2::real\}$
N measurable-isomorphic restrict-space borel B
by (*metis assms*(*2*) *greaterThanLessThan-borel linorder-not-le numeral-le-one-iff semiring-norm*(*69*) *standard-borel.isomorphic-subset-real uncountable-open-interval*)
then obtain $g\ g'$
where g[*measurable*]: $g \in N \to_M$ *restrict-space borel B*
$g' \in$ *restrict-space borel B* $\to_M N$
$\bigwedge x.\ x \in$ *space N* $\Longrightarrow g'\ (g\ x) = x$
$\bigwedge y.\ y \in B \Longrightarrow g\ (g'\ y) = y$
using *measurable-isomorphicD*[*OF B*(*3*)] **unfolding** *space-restrict-space* **by** *fastforce*
have *AB*:$A \cap B = \{\}$
using *A B* **by** *fastforce*
have [*measurable*]:$f \in M \to_M$ *restrict-space borel* $(A \cup B)$
using f(*1*) **unfolding** *measurable-restrict-space2-iff* **by** *blast*
have [*measurable*]:$g \in N \to_M$ *restrict-space borel* $(A \cup B)$
using g(*1*) **unfolding** *measurable-restrict-space2-iff* **by** *blast*

have *iso*:*restrict-space borel* $(A \cup B)$ *measurable-isomorphic* $M \bigoplus_M N$
proof(*safe intro*!: *measurable-isomorphic-byWitness*)
show *case-sum f g* $\in M \bigoplus_M N \to_M$ *restrict-space borel* $(A \cup B)$
by(*auto intro*!: *measurable-copair-Inl-Inr*)
show (λr. *if* $r \in A$ *then Inl* $(f'\ r)$ *else if* $r \in B$ *then Inr* $(g'\ r)$ *else undefined*)
$\in$ *restrict-space borel* $(A \cup B) \to_M M \bigoplus_M N$ (**is** *?f* $\in$ -)
proof −
have *1*:
restrict-space (*restrict-space borel* $(A \cup B)$) $\{r.\ r \in A\}$ = *restrict-space borel A*
restrict-space (*restrict-space borel* $(A \cup B)$) $\{r.\ r \notin A\}$ = *restrict-space borel B*
restrict-space (*restrict-space borel B*) $\{x.\ x \in B\}$ = *restrict-space borel B*
restrict-space (*restrict-space borel B*) $\{x.\ x \notin B\}$ = *count-space* $\{\}$
using *AB* **by**(*auto simp*: *restrict-restrict-space*
intro!: *arg-cong*[**where** *f=restrict-space borel*] *space-empty*)
have *2*:$\{r \in$ *space* (*restrict-space borel* $(A \cup B)$). $r \in A\} = A$
$\{x \in$ *space* (*restrict-space* (*restrict-space borel* $(A \cup B)$) $\{r.\ r \notin A\}$). $x \in B\} = B$
$\{x \in$ *space* (*restrict-space borel B*). $x \in B\} = B$
using *AB* **by** (*auto simp*: *space-restrict-space*)
show *?thesis*
by (*intro measurable-If-restrict-space-iff*[*THEN iffD2*] *conjI*)
(*unfold 1 2*, *simp-all add*: *sets-restrict-space-iff*)

qed
show $\bigwedge x.\ x \in$ *space* $(M \bigoplus_M N) \Longrightarrow$ *?f* (*case-sum f g x*) $= x$
$\bigwedge r.\ r \in$ *space* (*restrict-space borel* $(A \cup B)) \Longrightarrow$ *case-sum f g* (*?f r*) $= r$
using *measurable-space*[*OF f*(*1*)] *measurable-space*[*OF g*(*1*)] *AB*
by (*auto simp*: *space-copair-measure f g*)
qed
show *?thesis*
by(*auto intro*!: *standard-borel.measurable-isomorphic-standard*[*OF - iso*]
standard-borel.standard-borel-restrict-space[*OF standard-borel-ne.standard-borel*])
qed

corollary
shows *standard-borel-ne-copair-measure1*: *standard-borel-ne M* $\Longrightarrow$ *standard-borel* $N \Longrightarrow$ *standard-borel-ne* $(M \bigoplus_M N)$
and *standard-borel-ne-copair-measure2*: *standard-borel M* $\Longrightarrow$ *standard-borel-ne* $N \Longrightarrow$ *standard-borel-ne* $(M \bigoplus_M N)$
and *standard-borel-ne-copair-measure*: *standard-borel-ne M* $\Longrightarrow$ *standard-borel-ne* $N \Longrightarrow$ *standard-borel-ne* $(M \bigoplus_M N)$
by(*auto simp*: *standard-borel-ne-def standard-borel-ne-axioms-def standard-borel-copair-measure space-copair-measure*)

lemma *standard-borel-coPiM*:
assumes *countable I* $\bigwedge i.\ i \in I \Longrightarrow$ *standard-borel* (*Mi i*)
shows *standard-borel* (*coPiM I Mi*)
proof −
let *?I* $= \{i{\in}I.$ *space* (*Mi i*) $\neq \{\}\}$
have *countable-I*: *countable ?I*
using *assms* **by** *auto*
define I' **where** $I' \equiv$ *to-nat-on ?I* ‘ *?I*
define *Mn* **where** *Mn* $\equiv \lambda n.$ *Mi* (*from-nat-into ?I n*)
have I':*countable* I' $\bigwedge n.\ n \in I' \Longrightarrow$ *space* (*Mn n*) $\neq \{\}$
$\bigwedge n.\ n \in I' \Longrightarrow$ *standard-borel-ne* (*Mn n*)
using *countable-I from-nat-into-to-nat-on*[*OF countable-I*] *assms*(*2*)
by(*fastforce simp*: I'*-def Mn-def standard-borel-ne-def standard-borel-ne-axioms-def*
simp del: *from-nat-into-to-nat-on*)+
have *iso1*:*coPiM I Mi measurable-isomorphic coPiM* I' *Mn*
proof(*safe intro*!: *measurable-isomorphic-byWitness*[**where** $f{=}\lambda(i,x).$ (*to-nat-on ?I i*, *x*)
and $g{=}\lambda(n,x).$ (*from-nat-into ?I n*, *x*)])
show $(\lambda(i, x).$ (*to-nat-on ?I i*, *x*)) $\in$ *coPiM I Mi* $\to_M$ *coPiM* I' *Mn*
proof(*rule measurable-coPiM2*)
fix *i*
assume *i*:$i \in I$
show *Pair* (*to-nat-on ?I i*) $\in$ *Mi i* $\to_M$ *coPiM* I' *Mn*
proof(*cases space* (*Mi i*) $= \{\}$)
assume *space* (*Mi i*) $\neq \{\}$
then show *?thesis*
by(*intro measurable-compose*[*OF - measurable-Pair-coPiM*[**where** $I{=}I'$]])
(*use* I'*-def i countable-I Mn-def* **in** *auto*)

qed(*simp add*: *measurable-def*)
qed
show ($\lambda(n,x)$. (*from-nat-into ?I n*, *x*)) $\in$ *coPiM I′ Mn* $\to_M$ *coPiM I Mi*
proof(*rule measurable-coPiM2*)
fix *n*
assume *n* $\in$ *I′*
show *Pair* (*from-nat-into ?I n*) $\in$ *Mn n* $\to_M$ *coPiM I Mi*
by (*metis* (*no-types*, *lifting*) *Mn-def I′-def* ‹*n* $\in$ *I′*› *emptyE empty-is-image from-nat-into measurable-Pair-coPiM mem-Collect-eq*)
qed
qed(*auto intro*!: *from-nat-into-to-nat-on to-nat-on-from-nat-into simp*: *space-coPiM I′-def countable-I*)
have $\exists$ *A*. *A* $\in$ *sets borel* $\wedge$ *A* $\subseteq$ {*real n*<..*real n* + *1*} $\wedge$ *Mn n measurable-isomorphic* (*restrict-space borel A*)
if *n*:*n* $\in$ *I′* **for** *n*
using *standard-borel.isomorphic-subset-real*[*OF standard-borel-ne.standard-borel*[*OF I′*(*3*)[*OF n*]],*of* {*real n*<..*real n* + *1*}] *uncountable-half-open-interval-2*[*of real n real n* + *1*]
by *fastforce*
then obtain *An′*
where *An′*: $\bigwedge$*n*. *n* $\in$ *I′* $\Longrightarrow$ *An′ n* $\in$ *sets borel*
$\bigwedge$*n*. *n* $\in$ *I′* $\Longrightarrow$ *An′ n* $\subseteq$ {*real n*<..*real n* + *1*}
$\bigwedge$*n*. *n* $\in$ *I′* $\Longrightarrow$ *Mn n measurable-isomorphic* (*restrict-space borel* (*An′ n*))
by *metis*
define *An* **where** *An* $\equiv$ λ*n*. *if n* $\in$ *I′ then An′ n else* {*real n* + *1*}
have *An*[*measurable*]: $\bigwedge$*n*. *An n* $\in$ *sets borel*
$\bigwedge$*n*. *An n* $\subseteq$ {*real n*<..*real n* + *1*}
$\bigwedge$*n*. *n* $\in$ *I′* $\Longrightarrow$ *Mn n measurable-isomorphic* (*restrict-space borel* (*An n*))
using *An′* **by**(*auto simp*: *An-def*)
hence *disj-An*:*disjoint-family An*
unfolding *disjoint-family-on-def*
by *safe* (*metis* (*no-types*, *opaque-lifting*) *greaterThanAtMost-iff less-le nat-less-real-le not-less order-trans subset-eq*)
obtain *fn gn′*
where *fg*:$\bigwedge$*n*. *n* $\in$ *I′* $\Longrightarrow$ *fn n* $\in$ *Mn n* $\to_M$ *restrict-space borel* (*An n*)
$\bigwedge$*n*. *n* $\in$ *I′* $\Longrightarrow$ *gn′ n* $\in$ *restrict-space borel* (*An n*) $\to_M$ *Mn n*
$\bigwedge$*n x*. *n* $\in$ *I′* $\Longrightarrow$ *x* $\in$ *space* (*Mn n*) $\Longrightarrow$ *gn′ n* (*fn n x*) = *x*
$\bigwedge$*n r*. *n* $\in$ *I′* $\Longrightarrow$ *r* $\in$ *space* (*restrict-space borel* (*An n*)) $\Longrightarrow$ *fn n* (*gn′ n r*) = *r*
using *measurable-isomorphicD*[*OF An*(*3*)] **by** *metis*
define *gn* **where** *gn* $\equiv$ (λ*n r*. *if r* $\in$ *An n then gn′ n r else* (*SOME x*. *x* $\in$ *space* (*Mn n*)))
have *gn-meas*[*measurable*]:*gn n* $\in$ *borel* $\to_M$ *Mn n* **if** *n*:*n* $\in$ *I′* **for** *n*
unfolding *gn-def* **by**(*rule measurable-restrict-space-iff*[*THEN iffD1*,*OF* - - *fg*(*2*)[*OF n*]])
(*auto simp add*: *I′*(*2*) *some-in-eq that*)
have *fg′*:$\bigwedge$*n x*. *n* $\in$ *I′* $\Longrightarrow$ *x* $\in$ *space* (*Mn n*) $\Longrightarrow$ *gn n* (*fn n x*) = *x*

$\bigwedge n\ r.\ n \in I' \Longrightarrow r \in An\ n \Longrightarrow fn\ n\ (gn\ n\ r) = r$
using *fg measurable-space*[*OF fg*(*1*)] **by**(*auto simp*: *gn-def*)
have *fn*[*measurable*]:*fn n* $\in$ *Mn n* $\rightarrow_M$ *restrict-space borel* ($\bigcup n{\in}I'$. *An n*) **if** *n*:*n* $\in I'$ **for** *n*
using *measurable-restrict-space2-iff*[*THEN iffD1*,*OF fg*(*1*)[*OF n*]]
by(*auto intro*!: *measurable-restrict-space2 n*)
let *?f* = λ(*n*,*x*). *fn n x* **and** *?g* =λr. (*nat* $\lceil r \rceil - 1$, *gn* (*nat* $\lceil r \rceil - 1$) *r*)
have *iso2*:*coPiM* I' *Mn measurable-isomorphic restrict-space borel* ($\bigcup n{\in}I'$. *An n*)
proof(*safe intro*!: *measurable-isomorphic-byWitness*)
show *?f* $\in$ *coPiM* I' *Mn* $\rightarrow_M$ *restrict-space borel* ($\bigcup n{\in}I'$. *An n*)
by(*auto intro*!: *measurable-coPiM2*)
next
show *?g* $\in$ *restrict-space borel* ($\bigcup n{\in}I'$. *An n*) $\rightarrow_M$ *coPiM* I' *Mn*
proof(*safe intro*!: *measurable-coPiM1*)
have *1*:*restrict-space borel* ($\bigcup$ (*An* ‘ I')) $\rightarrow_M$ *count-space* I'
= *restrict-space borel* ($\bigcup$ (*An* ‘ I')) $\rightarrow_M$ *restrict-space* (*count-space UNIV*) I'
by (*simp add*: *restrict-count-space*)
show (λx. *nat* $\lceil x \rceil - 1$) $\in$ *restrict-space borel* ($\bigcup$ (*An* ‘ I')) $\rightarrow_M$ *count-space* I'
unfolding *1*
proof(*safe intro*!: *measurable-restrict-space3*)
fix *n r*
assume *n*:*n* $\in I'$ *r* $\in$ *An n*
then have *real n* < *r r* $\leq$ *real n* + *1*
using *An*(*2*) **by** *fastforce*+
thus *nat* $\lceil r \rceil - 1 \in I'$
by (*metis n*(*1*) *add.commute diff-Suc-1 le-SucE nat-ceiling-le-eq not-less of-nat-Suc*)
qed *simp*
qed(*auto simp*: *measurable-restrict-space1*)
next
fix *n x*
assume (*n*,*x*)$\in$*space* (*coPiM* I' *Mn*)
then have *nx*:*n* $\in I'$ *x* $\in$ *space* (*Mn n*)
by(*auto simp*: *space-coPiM*)
have *1*:*nat* $\lceil$ *?f* (*n*,*x*)$\rceil$ = *n* + *1*
using *measurable-space*[*OF fg*(*1*)[*OF nx*(*1*)] *nx*(*2*)] *An*(*2*)[*of n*]
by *simp*
(*metis add.commute greaterThanAtMost-iff le-SucE nat-ceiling-le-eq not-less of-nat-Suc subset-eq*)
show *?g* (*?f* (*n*,*x*)) = (*n*,*x*)
unfolding *1* **using** *fg*$'$(*1*)[*OF nx*] **by** *simp*
next
fix *y*
assume *y* $\in$ *space* (*restrict-space borel* ($\bigcup$ (*An* ‘ I')))
then obtain *n* **where** *n*: *n* $\in I'$ *y* $\in$ *An n*
by *auto*

then have [*simp*]:*nat* $\lceil y \rceil$ = *n* + *1*
using *An*(*2*)[*of n*]
by *simp* (*metis add.commute greaterThanAtMost-iff le-SucE nat-ceiling-le-eq not-less of-nat-Suc subset-eq*)
show *?f* (*?g y*) = *y*
using *fg*′(*2*)[*OF n*(*1*)] *n*(*2*) **by** *auto*
qed
have *standard-borel* (*restrict-space borel* ($\bigcup$ (*An* ‘ *I*′)))
by(*auto intro*!: *standard-borel-ne.standard-borel*[*THEN standard-borel.standard-borel-restrict-space*])
with *iso1 iso2* **show** *?thesis*
by (*meson measurable-isomorphic-sym standard-borel.measurable-isomorphic-standard*)
qed

lemma *standard-borel-ne-coPiM*:
assumes *countable I* $\bigwedge$*i*. *i* ∈ *I* ⟹ *standard-borel* (*Mi i*)
and *i* ∈ *I space* (*Mi i*) ≠ {}
shows *standard-borel-ne* (*coPiM I Mi*)
proof −
have *space* (*coPiM I Mi*) ≠ {}
using *assms*(*3*) *assms*(*4*) *space-coPiM* **by** *fastforce*
thus *?thesis*
by(*auto intro*!: *standard-borel-coPiM assms simp*: *standard-borel-ne-def standard-borel-ne-axioms-def*)
qed

4.3 Relationships with Quasi-Borel Spaces

Proposition19(3) [1]

lemma *r-preserve-copair*: *measure-to-qbs* (*copair-measure M N*) = *measure-to-qbs M* $\bigoplus_Q$ *measure-to-qbs N*
proof(*safe intro*!: *qbs-eqI*)
fix α
assume α ∈ *qbs-Mx* (*measure-to-qbs* (*M* $\bigoplus_M$ *N*))
then have *a*[*measurable*]: α ∈ *borel* $\to_M$ *M* $\bigoplus_M$ *N*
by(*simp add*: *qbs-Mx-R*)
have *s*[*measurable*]: α −‘ *Inr* ‘ *space N* ∈ *sets borel* α −‘ *Inl* ‘ *space M* ∈ *sets borel*
by(*auto intro*!: *measurable-sets-borel*[*OF a*])
consider α −‘ *Inl* ‘ *space M* ∩ *space borel* = *space borel*
| α −‘ *Inr* ‘ (*space N*) ∩ *space borel* = *space borel*
| α −‘ *Inl* ‘ *space M* ∩ *space borel* ⊂ *space borel*
α −‘ *Inr* ‘ (*space N*) ∩ *space borel* ⊂ *space borel*
by *blast*
then show α ∈ *qbs-Mx* (*measure-to-qbs M* $\bigoplus_Q$ *measure-to-qbs N*)
proof *cases*
assume *1*:α −‘ *Inl* ‘ *space M* ∩ *space borel* = *space borel*
then obtain *f*′ **where** *f*′ ∈ *borel* $\to_M$ *M* $\bigwedge$*x*. *x* ∈ *space borel* ⟹ α *x* = *Inl* (*f*′ *x*)
using *measurable-copair-dest1*[*OF a*] **by** *blast*

thus *?thesis*
using *1* **by**(*auto simp*: *copair-qbs-Mx copair-qbs-Mx-def qbs-Mx-R*
intro!: *bexI*[**where** *x*=α $-$‘ *Inr* ‘ *space N*] *bexI*[**where** *x*=f'])
next
assume *2*:α $-$‘ *Inr* ‘ *space N* $\cap$ *space borel* = *space borel*
then obtain f' **where** $f' \in$ *borel* $\rightarrow_M$ *N* $\bigwedge x$. $x \in$ *space borel* $\Longrightarrow \alpha\ x =$ *Inr* ($f'\ x$)
using *measurable-copair-dest2*[*OF a*] **by** *blast*
thus *?thesis*
using *2* **by**(*auto simp*: *copair-qbs-Mx copair-qbs-Mx-def qbs-Mx-R*
intro!: *bexI*[**where** *x*=α $-$‘ *Inr* ‘ *space N*] *bexI*[**where** *x*=f'])
next
case *3*
then obtain $f'\ f''$
where *f*[*measurable*]:$f' \in$ *borel* $\rightarrow_M$ *M*
$f'' \in$ *borel* $\rightarrow_M$ *N*
$\bigwedge x$. $x \in$ *space borel* $\Longrightarrow x \in \alpha$ $-$‘ *Inl* ‘ *space M* $\Longrightarrow \alpha\ x =$ *Inl* ($f'\ x$)
$\bigwedge x$. $x \in$ *space borel* $\Longrightarrow x \notin \alpha$ $-$‘ *Inl* ‘ *space M* $\Longrightarrow \alpha\ x =$ *Inr* ($f''\ x$)
using *measurable-copair-dest3*[*OF a*] **by** *metis*
moreover have α $-$‘ *Inl* ‘ *space M* $\neq$ *UNIV* α $-$‘ *Inl* ‘ *space M* $\neq$ {}
using *3 measurable-space*[*OF a*] **by**(*fastforce simp*: *space-copair-measure*)+
ultimately show *?thesis*
by(*auto simp*: *copair-qbs-Mx copair-qbs-Mx-def qbs-Mx-R simp del*: *vimage-eq*
intro!: *bexI*[**where** *x*=α $-$‘ *Inl* ‘ *space M*] *bexI*[**where** *x*=f'] *bexI*[**where** *x*=f''])
qed
qed(*auto simp*: *qbs-Mx-R copair-qbs-Mx copair-qbs-Mx-def*)

lemma *r-preserve-coproduct*:
assumes *countable I*
shows *measure-to-qbs* (*coPiM I M*) = ($\amalg_Q$ $i \in I$. *measure-to-qbs* (*M i*))
proof(*safe intro*!: *qbs-eqI*)
fix α
assume *h*:$\alpha \in$ *qbs-Mx* (*measure-to-qbs* (*coPiM I M*))
then obtain *a g*
where $a \in$ *borel* $\rightarrow_M$ *count-space I*
$\bigwedge i$. $i \in I \Longrightarrow$ *space* (*M i*) $\neq$ {} $\Longrightarrow g\ i \in$ *borel* $\rightarrow_M$ *M i*
$\alpha = (\lambda x.\ (a\ x,\ g\ (a\ x)\ x))$
using *measurable-coPiM1-elements*[*OF assms*] **unfolding** *qbs-Mx-R* **by** *blast*
thus $\alpha \in$ *qbs-Mx* ($\amalg_Q$ $i \in I$. *measure-to-qbs* (*M i*))
using *qbs-Mx-to-X*[*OF h*]
by(*safe intro*!: *coPiQ-MxI*) (*auto simp*: *qbs-Mx-R qbs-space-R space-coPiM*)
next
fix α
assume $\alpha \in$ *qbs-Mx* ($\amalg_Q$ $i \in I$. *measure-to-qbs* (*M i*))
then obtain *a g* **where** $a \in$ *borel* $\rightarrow_M$ *count-space I*
$\bigwedge i$. $i \in$ *range a* $\Longrightarrow g\ i \in$ *borel* $\rightarrow_M$ *M i* $\alpha = (\lambda x.\ (a\ x,\ g\ (a$

x) *x*))
  unfolding *coPiQ-Mx coPiQ-Mx-def qbs-Mx-R* **by** *blast*
 thus $\alpha \in$ *qbs-Mx* (*measure-to-qbs* (*coPiM I M*))
  by(*auto intro*!: *measurable-coPiM1′ simp*: *qbs-Mx-R assms*)
qed

end

References

[1] C. Heunen, O. Kammar, S. Staton, and H. Yang. A convenient category for higher-order probability theory. In *Proceedings of the 32nd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '17. IEEE Press, 2017.