

Formalizing MLTL in Isabelle/HOL

Zili Wang and Katherine Kosaian and Alec Rosentrater

March 4, 2025

Abstract

Building on the existing formalization of Mission-time Linear Temporal Logic (MLTL) [1], we formalize the notions of *language decomposition* and *language partition* for MLTL. More specifically, we formalize an algorithm to compute a language partition for MLTL and formally prove its correctness. Our algorithm is executable, and we export it Haskell via Isabelle/HOL's code generator.

Contents

1	Extended MLTL Data Structure with Interval Compositions	2
2	List Helper Functions and Properties	3
3	Decomposition Function	5
3.1	Examples	7
4	Properties of <code>convert nnf ext</code>	8
4.1	Cases where <code>to mltl</code> is bijective	14
5	Lemmas about Integer Composition	14
6	MLTL Decomposition Lemmas	26
7	Lemmas for MLTL operators that operate over lists of <code>mtl</code> formulas	26
7.1	Forward Direction Proofs	27
7.2	Converse Direction Proofs	29
7.3	Biconditional Lemmas	29
8	MLTL Decomposition Top Level Correctness	30
8.1	Helper Lemmas	30
8.2	Union Theorem	61

8.3 Disjointedness Theorem	104
8.4 Disjointedness Theorem (special case of k=1)	132

9 Pretty Parsing 155

10 Code Export 156

theory *MLTL-Language-Partition-Algorithm*

imports *Mission-Time-LTL.MLTL-Properties*

begin

1 Extended MLTL Data Structure with Interval Compositions

Extended datatype that has an additional nat list associated with the temporal operators F, U, R to represent integer compositions of the interval

```
datatype (atoms-mltl: 'a) mltl-ext =
  | True-mltl-ext (Truec)
  | False-mltl-ext (Falsec)
  | Prop-mltl-ext 'a (Propc '(-))
  | Not-mltl-ext 'a mltl-ext (Notc - [85] 85)
  | And-mltl-ext 'a mltl-ext 'a mltl-ext (- Andc - [82, 82] 81)
  | Or-mltl-ext 'a mltl-ext 'a mltl-ext (- Orc - [81, 81] 80)
  | Future-mltl-ext nat nat nat list 'a mltl-ext (Fc '['-, -'] '<->' - [88, 88, 88, 88] 87)
  | Global-mltl-ext nat nat nat list 'a mltl-ext (Gc '['-, -'] '<->' - [88, 88, 88, 88] 87)
  | Until-mltl-ext 'a mltl-ext nat nat nat list 'a mltl-ext (- Uc '['-, -'] '<->' - [84, 84, 84, 84] 83)
  | Release-mltl-ext 'a mltl-ext nat nat nat list 'a mltl-ext (- Rc '['-, -'] '<->' - [84, 84, 84, 84] 83)
```

Converts mltl ext formula to mltl by just dropping the nat list

```
fun to-mltl:: 'a mltl-ext  $\Rightarrow$  'a mltl where
  | to-mltl Truec = Truem
  | to-mltl Falsec = Falsem
  | to-mltl Propc (p) = Propm (p)
  | to-mltl (Notc  $\varphi$ ) = Notm (to-mltl  $\varphi$ )
  | to-mltl ( $\varphi$  Andc  $\psi$ ) = (to-mltl  $\varphi$ ) Andm (to-mltl  $\psi$ )
  | to-mltl ( $\varphi$  Orc  $\psi$ ) = (to-mltl  $\varphi$ ) Orm (to-mltl  $\psi$ )
  | to-mltl (Fc [a, b] <L>  $\varphi$ ) = (Fm [a, b] (to-mltl  $\varphi$ ))
  | to-mltl (Gc [a, b] <L>  $\varphi$ ) = (Gm [a, b] (to-mltl  $\varphi$ ))
  | to-mltl ( $\varphi$  Uc [a, b] <L>  $\psi$ ) = ((to-mltl  $\varphi$ ) Um [a, b] (to-mltl  $\psi$ ))
  | to-mltl ( $\varphi$  Rc [a, b] <L>  $\psi$ ) = ((to-mltl  $\varphi$ ) Rm [a, b] (to-mltl  $\psi$ ))
```

definition *semantics-mltl-ext*:: 'a set list \Rightarrow 'a mltl-ext \Rightarrow bool
 (- \models_c - [80,80] 80)
 where $\pi \models_c \varphi = \pi \models_m (to_mltl \ \varphi)$

definition *semantic-equiv-ext*:: 'a mltl-ext \Rightarrow 'a mltl-ext \Rightarrow bool
 (- \equiv_c - [80, 80] 80)
 where $\varphi \equiv_c \psi = (to_mltl \ \varphi) \equiv_m (to_mltl \ \psi)$

definition *language-mltl-r* :: 'a mltl \Rightarrow nat \Rightarrow 'a set list set
 where *language-mltl-r* φ $r =$
 $\{\pi. semantics_mltl \ \pi \ \varphi \wedge length \ \pi \geq r\}$

fun *convert-nnf-ext*:: 'a mltl-ext \Rightarrow 'a mltl-ext **where**
convert-nnf-ext $True_c = True_c$
 | *convert-nnf-ext* $False_c = False_c$
 | *convert-nnf-ext* $Prop_c \ (p) = Prop_c \ (p)$
 | *convert-nnf-ext* $(\varphi \ And_c \ \psi) = ((convert_nnf_ext \ \varphi) \ And_c \ (convert_nnf_ext \ \psi))$
 | *convert-nnf-ext* $(\varphi \ Or_c \ \psi) = ((convert_nnf_ext \ \varphi) \ Or_c \ (convert_nnf_ext \ \psi))$
 | *convert-nnf-ext* $(F_c \ [a,b] \ <L> \ \varphi) = (F_c \ [a,b] \ <L> \ (convert_nnf_ext \ \varphi))$
 | *convert-nnf-ext* $(G_c \ [a,b] \ <L> \ \varphi) = (G_c \ [a,b] \ <L> \ (convert_nnf_ext \ \varphi))$
 | *convert-nnf-ext* $(\varphi \ U_c \ [a,b] \ <L> \ \psi) = ((convert_nnf_ext \ \varphi) \ U_c \ [a,b] \ <L> \ (convert_nnf_ext \ \psi))$
 | *convert-nnf-ext* $(\varphi \ R_c \ [a,b] \ <L> \ \psi) = ((convert_nnf_ext \ \varphi) \ R_c \ [a,b] \ <L> \ (convert_nnf_ext \ \psi))$

 | *convert-nnf-ext* $(Not_c \ True_c) = False_c$
 | *convert-nnf-ext* $(Not_c \ False_c) = True_c$
 | *convert-nnf-ext* $(Not_c \ Prop_c \ (p)) = (Not_c \ Prop_c \ (p))$
 | *convert-nnf-ext* $(Not_c \ (Not_c \ \varphi)) = convert_nnf_ext \ \varphi$
 | *convert-nnf-ext* $(Not_c \ (\varphi \ And_c \ \psi)) = ((convert_nnf_ext \ (Not_c \ \varphi)) \ Or_c \ (convert_nnf_ext \ (Not_c \ \psi)))$
 | *convert-nnf-ext* $(Not_c \ (\varphi \ Or_c \ \psi)) = ((convert_nnf_ext \ (Not_c \ \varphi)) \ And_c \ (convert_nnf_ext \ (Not_c \ \psi)))$
 | *convert-nnf-ext* $(Not_c \ (F_c \ [a,b] \ <L> \ \varphi)) = (G_c \ [a,b] \ <L> \ (convert_nnf_ext \ (Not_c \ \varphi)))$
 | *convert-nnf-ext* $(Not_c \ (G_c \ [a,b] \ <L> \ \varphi)) = (F_c \ [a,b] \ <L> \ (convert_nnf_ext \ (Not_c \ \varphi)))$
 | *convert-nnf-ext* $(Not_c \ (\varphi \ U_c \ [a,b] \ <L> \ \psi)) = ((convert_nnf_ext \ (Not_c \ \varphi)) \ R_c \ [a,b] \ <L> \ (convert_nnf_ext \ (Not_c \ \psi)))$
 | *convert-nnf-ext* $(Not_c \ (\varphi \ R_c \ [a,b] \ <L> \ \psi)) = ((convert_nnf_ext \ (Not_c \ \varphi)) \ U_c \ [a,b] \ <L> \ (convert_nnf_ext \ (Not_c \ \psi)))$

2 List Helper Functions and Properties

Computes the partial sum of the first i elements of list

definition *partial-sum* :: [nat list, nat] \Rightarrow nat **where**
partial-sum $L \ i = sum_list \ (take \ i \ L)$

Given interval start time a, and a list of ints $L = [t1, t2, t3]$ Constructs

the list (of length 1 longer) of partial sums added to a: [a, a+t1, a+t1+t2, a+t1+t2+t3]

definition *interval-times* :: [nat, nat list] ⇒ nat list **where**
interval-times a L = map (λi. a + partial-sum L i) [0 ..< length L + 1]

value *interval-times* 3 [1, 2, 3, 4, 5] =
[3, 4, 6, 9, 13, 18]

This function checks that L is a composition of n. A composition of an integer n is a way of writing n as the sum of a sequence of (strictly) positive integers

definition *is-composition* :: [nat, nat list] ⇒ bool **where**
is-composition n L = ((∀ i. List.member L i → i > 0) ∧ (sum-list L = n))

Checks that every nat list in input of type mtl ext is a composition of its interval For example the formula F[2,7] has interval of length 7-2+1=6, and a valid composition would be L = [2, 3, 1]

fun *is-composition-MLTL*:: 'a mtl-ext ⇒ bool **where**
is-composition-MLTL (φ And_c ψ) = ((*is-composition-MLTL* φ) ∧ (*is-composition-MLTL* ψ))
| *is-composition-MLTL* (φ Or_c ψ) = ((*is-composition-MLTL* φ) ∧ (*is-composition-MLTL* ψ))
| *is-composition-MLTL* (G_c[a,b] <L> φ) = ((*is-composition* (b-a+1) L) ∧ (*is-composition-MLTL* φ))
| *is-composition-MLTL* (Not_c φ) = *is-composition-MLTL* φ
| *is-composition-MLTL* (F_c[a,b] <L> φ) = ((*is-composition* (b-a+1) L) ∧ (*is-composition-MLTL* φ))
| *is-composition-MLTL* (φ U_c[a,b] <L> ψ) = ((*is-composition* (b-a+1) L) ∧ (*is-composition-MLTL* φ) ∧ (*is-composition-MLTL* ψ))
| *is-composition-MLTL* (φ R_c[a,b] <L> ψ) = ((*is-composition* (b-a+1) L) ∧ (*is-composition-MLTL* φ) ∧ (*is-composition-MLTL* ψ))
| *is-composition-MLTL* - = True

definition *is-composition-allones*:: nat ⇒ nat list ⇒ bool **where**
is-composition-allones n L = ((*is-composition* n L) ∧ (∀ i < length L. L[i] = 1))

fun *is-composition-MLTL-allones*:: 'a mtl-ext ⇒ bool **where**
is-composition-MLTL-allones (φ And_c ψ) = ((*is-composition-MLTL-allones* φ) ∧ (*is-composition-MLTL-allones* ψ))
| *is-composition-MLTL-allones* (φ Or_c ψ) = ((*is-composition-MLTL-allones* φ) ∧ (*is-composition-MLTL-allones* ψ))
| *is-composition-MLTL-allones* (G_c[a,b] <L> φ) = ((*is-composition-allones* (b-a+1) L) ∧ *is-composition-MLTL-allones* φ)
| *is-composition-MLTL-allones* (Not_c φ) = *is-composition-MLTL-allones* φ
| *is-composition-MLTL-allones* (F_c[a,b] <L> φ) = ((*is-composition-allones* (b-a+1) L) ∧ (*is-composition-MLTL-allones* φ))
| *is-composition-MLTL-allones* (φ U_c[a,b] <L> ψ) = ((*is-composition-allones* (b-a+1) L) ∧ (*is-composition-MLTL-allones* φ) ∧ (*is-composition-MLTL-allones* ψ))

| *is-composition-MTLTl-allones* ($\varphi R_c[a,b] <L> \psi$) = (*is-composition-allones* ($b-a+1$)
 L) \wedge (*is-composition-MTLTl-allones* φ) \wedge (*is-composition-MTLTl-allones* ψ)
| *is-composition-MTLTl-allones* - = *True*

3 Decomposition Function

fun *pairs* :: 'a list \Rightarrow 'a list \Rightarrow ('a \times 'a) list **where**

pairs [] L2 = []

| *pairs* (h1#T1) L2 = (map ($\lambda x.$ (h1, x)) L2) @ (*pairs* T1 L2)

fun *And-mltl-list* :: 'a mltl-ext list \Rightarrow 'a mltl-ext list \Rightarrow 'a mltl-ext list **where**

And-mltl-list D- φ D- ψ = map ($\lambda x.$ *And-mltl-ext* (fst x) (snd x)) (*pairs* D- φ D- ψ)

fun *Global-mltl-list* :: 'a mltl-ext list \Rightarrow nat \Rightarrow nat \Rightarrow nat list \Rightarrow 'a mltl-ext list
where

Global-mltl-list D- φ a b L = map ($\lambda x.$ *Global-mltl-ext* a b L x) D- φ

fun *Future-mltl-list* :: 'a mltl-ext list \Rightarrow nat \Rightarrow nat \Rightarrow nat list \Rightarrow 'a mltl-ext list
where

Future-mltl-list D- φ a b L = map ($\lambda x.$ *Future-mltl-ext* a b L x) D- φ

fun *Until-mltl-list* :: 'a mltl-ext \Rightarrow 'a mltl-ext list \Rightarrow nat \Rightarrow nat \Rightarrow nat list \Rightarrow 'a
mltl-ext list **where**

Until-mltl-list φ D- ψ a b L = map ($\lambda x.$ *Until-mltl-ext* φ a b L x) D- ψ

fun *Release-mltl-list* :: 'a mltl-ext list \Rightarrow 'a mltl-ext \Rightarrow nat \Rightarrow nat \Rightarrow nat list \Rightarrow 'a
mltl-ext list **where**

Release-mltl-list D- φ ψ a b L = map ($\lambda x.$ *Release-mltl-ext* x a b L ψ) D- φ

fun *Mighty-Release-mltl-ext*:: 'a mltl-ext \Rightarrow 'a mltl-ext \Rightarrow nat \Rightarrow nat \Rightarrow nat list \Rightarrow
'a mltl-ext

where *Mighty-Release-mltl-ext* x ψ a b L =

(*And-mltl-ext* (*Release-mltl-ext* x a b L ψ))

(*Future-mltl-ext* a b L x))

fun *Mighty-Release-mltl-list* :: 'a mltl-ext list \Rightarrow 'a mltl-ext \Rightarrow nat \Rightarrow nat \Rightarrow nat
list \Rightarrow 'a mltl-ext list **where**

Mighty-Release-mltl-list D- φ ψ a b L = map ($\lambda x.$ *Mighty-Release-mltl-ext* x ψ a b
L) D- φ

fun *Global-mltl-decomp* :: 'a mltl-ext list \Rightarrow nat \Rightarrow nat \Rightarrow nat list \Rightarrow 'a mltl-ext
list **where**

Global-mltl-decomp D- φ a 0 L = *Global-mltl-list* D- φ a a [1]

| *Global-mltl-decomp* D- φ a len L = *And-mltl-list* (*Global-mltl-decomp* D- φ a (len-1)
L)

(*Global-mltl-list* D- φ (a+len) (a+len) [1])

value *Global-mltl-decomp* [*True-mltl-ext*, (*Prop-mltl-ext* (0::nat))] 0 2 [3] =

[(G_c [0,0] <[1]> *True_c* *And_c* G_c [1,1] <[1]> *True_c*) *And_c* G_c [2,2] <[1]> *True_c*,

(G_c [0,0] <[1]> *True_c* *And_c* G_c [1,1] <[1]> *True_c*) *And_c* G_c [2,2] <[1]>

$Prop_c (0),$
 $(G_c [0,0] \langle [1] \rangle \ True_c \ And_c \ G_c [1,1] \langle [1] \rangle \ Prop_c (0)) \ And_c \ G_c [2,2] \langle [1] \rangle$
 $True_c,$
 $(G_c [0,0] \langle [1] \rangle \ True_c \ And_c \ G_c [1,1] \langle [1] \rangle \ Prop_c (0)) \ And_c \ G_c [2,2] \langle [1] \rangle$
 $Prop_c (0),$
 $(G_c [0,0] \langle [1] \rangle \ Prop_c (0) \ And_c \ G_c [1,1] \langle [1] \rangle \ True_c) \ And_c \ G_c [2,2] \langle [1] \rangle$
 $True_c,$
 $(G_c [0,0] \langle [1] \rangle \ Prop_c (0) \ And_c \ G_c [1,1] \langle [1] \rangle \ True_c) \ And_c \ G_c [2,2] \langle [1] \rangle$
 $Prop_c (0),$
 $(G_c [0,0] \langle [1] \rangle \ Prop_c (0) \ And_c \ G_c [1,1] \langle [1] \rangle \ Prop_c (0)) \ And_c \ G_c [2,2]$
 $\langle [1] \rangle \ True_c,$
 $(G_c [0,0] \langle [1] \rangle \ Prop_c (0) \ And_c \ G_c [1,1] \langle [1] \rangle \ Prop_c (0)) \ And_c \ G_c [2,2]$
 $\langle [1] \rangle \ Prop_c (0)$

fun $LP\text{-}m\text{ltl}\text{-}aux :: 'a \ m\text{ltl}\text{-}ext \Rightarrow nat \Rightarrow 'a \ m\text{ltl}\text{-}ext \ list \ \mathbf{where}$

$LP\text{-}m\text{ltl}\text{-}aux \ \varphi \ 0 = [\varphi]$
 $| LP\text{-}m\text{ltl}\text{-}aux \ True_c \ (Suc \ k) = [True_c]$
 $| LP\text{-}m\text{ltl}\text{-}aux \ False_c \ (Suc \ k) = [False_c]$
 $| LP\text{-}m\text{ltl}\text{-}aux \ Prop_c \ (p) \ (Suc \ k) = [Prop_c \ (p)]$
 $| LP\text{-}m\text{ltl}\text{-}aux \ (Not_c \ (Prop_c \ (p))) \ (Suc \ k) = [Not_c \ (Prop_c \ (p))]$
 $| LP\text{-}m\text{ltl}\text{-}aux \ (\varphi \ And_c \ \psi) \ (Suc \ k) =$
 $\quad (let \ D\text{-}\varphi = (LP\text{-}m\text{ltl}\text{-}aux \ (convert\text{-}nnf\text{-}ext \ \varphi) \ k) \ in$
 $\quad (let \ D\text{-}\psi = (LP\text{-}m\text{ltl}\text{-}aux \ (convert\text{-}nnf\text{-}ext \ \psi) \ k) \ in$
 $\quad And\text{-}m\text{ltl}\text{-}list \ D\text{-}\varphi \ D\text{-}\psi))$
 $| LP\text{-}m\text{ltl}\text{-}aux \ (\varphi \ Or_c \ \psi) \ (Suc \ k) =$
 $\quad (let \ D\text{-}\varphi = (LP\text{-}m\text{ltl}\text{-}aux \ (convert\text{-}nnf\text{-}ext \ \varphi) \ k) \ in$
 $\quad (let \ D\text{-}\psi = (LP\text{-}m\text{ltl}\text{-}aux \ (convert\text{-}nnf\text{-}ext \ \psi) \ k) \ in$
 $\quad (And\text{-}m\text{ltl}\text{-}list \ D\text{-}\varphi \ D\text{-}\psi) \ @ \ (And\text{-}m\text{ltl}\text{-}list \ [Not_c \ \varphi] \ D\text{-}\psi) \ @$
 $\quad (And\text{-}m\text{ltl}\text{-}list \ D\text{-}\varphi \ [(Not_c \ \psi)]))$
 $| LP\text{-}m\text{ltl}\text{-}aux \ (G_c[a,b] \ \langle L \rangle \ \varphi) \ (Suc \ k) =$
 $\quad (let \ D\text{-}\varphi = (LP\text{-}m\text{ltl}\text{-}aux \ (convert\text{-}nnf\text{-}ext \ \varphi) \ k) \ in$
 $\quad (if \ (length \ D\text{-}\varphi \leq 1) \ then \ ([G_c[a,b] \ \langle L \rangle \ \varphi])$
 $\quad \quad \quad \text{else} \ (Global\text{-}m\text{ltl}\text{-}decomp \ D\text{-}\varphi \ a \ (b-a) \ L)))$
 $| LP\text{-}m\text{ltl}\text{-}aux \ (F_c[a,b] \ \langle L \rangle \ \varphi) \ (Suc \ k) =$
 $\quad (let \ D\text{-}\varphi = (LP\text{-}m\text{ltl}\text{-}aux \ (convert\text{-}nnf\text{-}ext \ \varphi) \ k) \ in$
 $\quad (let \ s = interval\text{-}times \ a \ L \ in$
 $\quad (Future\text{-}m\text{ltl}\text{-}list \ D\text{-}\varphi \ (s!0) \ ((s!1)-1) \ [(s!1)-(s!0)]) \ @ \ (concat \ (map$
 $\quad \quad (\lambda i. \ (And\text{-}m\text{ltl}\text{-}list \ [Global\text{-}m\text{ltl}\text{-}ext \ (s!0) \ ((s!i)-1) \ [s!i - s!0] \ (Not_c \ \varphi)]$
 $\quad \quad \quad (Future\text{-}m\text{ltl}\text{-}list \ D\text{-}\varphi \ (s!i) \ ((s!(i+1))-1) \ [s!(i+1)-(s!i)])))$
 $\quad \quad [1 \ ..< \ length \ L])))$
 $| LP\text{-}m\text{ltl}\text{-}aux \ (\varphi \ U_c[a,b] \ \langle L \rangle \ \psi) \ (Suc \ k) =$
 $\quad (let \ D\text{-}\psi = (LP\text{-}m\text{ltl}\text{-}aux \ (convert\text{-}nnf\text{-}ext \ \psi) \ k) \ in$
 $\quad (let \ s = interval\text{-}times \ a \ L \ in$
 $\quad (Until\text{-}m\text{ltl}\text{-}list \ \varphi \ D\text{-}\psi \ (s!0) \ ((s!1)-1) \ [(s!1)-(s!0)]) \ @ \ (concat \ (map$
 $\quad \quad (\lambda i. \ (And\text{-}m\text{ltl}\text{-}list \ [Global\text{-}m\text{ltl}\text{-}ext \ (s!0) \ ((s!i)-1) \ [s!i - s!0] \ (And\text{-}m\text{ltl}\text{-}ext \ \varphi$
 $\quad \quad \quad (Not_c \ \psi)]))$
 $\quad \quad \quad (Until\text{-}m\text{ltl}\text{-}list \ \varphi \ D\text{-}\psi \ (s!i) \ ((s!(i+1)-1) \ [s!(i+1)-(s!i)])))$
 $\quad \quad [1 \ ..< \ length \ L])))$
 $| LP\text{-}m\text{ltl}\text{-}aux \ (\varphi \ R_c[a,b] \ \langle L \rangle \ \psi) \ (Suc \ k) =$

```

(let D-φ = (LP-mltl-aux (convert-nnf-ext φ) k) in
(let s = interval-times a L in
[Global-mltl-ext a b L ((Notc φ) Andc ψ)] @
(Mighty-Release-mltl-list D-φ ψ (s!0) ((s!1)-1) [(s!1)-(s!0)]) @ (concat (map
(λi. (And-mltl-list [Global-mltl-ext (s!0) ((s!i)-1) [s!i - s!0] ((Notc φ) Andc
ψ)]
(Mighty-Release-mltl-list D-φ ψ (s!i) ((s!(i+1)-1)) [s!(i+1)-(s!i)])))
[1 ..< length L])))
| LP-mltl-aux - - = []

```

```

fun LP-mltl :: 'a mltl-ext ⇒ nat ⇒ 'a mltl list where
LP-mltl φ k = map (λx. to-mltl x)
(map (λx. convert-nnf-ext x) (LP-mltl-aux (convert-nnf-ext φ) k))

```

3.1 Examples

```

value LP-mltl-aux (Fc[0,9] <[3, 3, 3]> ((Propc (0::nat)) Orc (Propc (1::nat))))
1 =
[Fc [0,2] <[3]> (Propc (0) Orc Propc (1)),
Gc [0,2] <[3]> (Notc (Propc (0) Orc Propc (1))) Andc Fc [3,5] <[3]> (Propc
(0) Orc Propc (1)),
Gc [0,5] <[6]> (Notc (Propc (0) Orc Propc (1))) Andc Fc [6,8] <[3]> (Propc
(0) Orc Propc (1))]

```

```

value LP-mltl (Truec Orc (Propc (0::nat))) 1 =
[Truem Andm Propm (0), Falsem Andm Propm (0), Truem Andm Notm Propm
(0)]

```

```

value LP-mltl ((Propc (0::nat)) Uc [2,5] <[4]> (Propc (1))) 1 =
[Propm (0) Um [2,5] Propm (1)]

```

```

value LP-mltl ((Propc (0::nat)) Rc[2,5] <[2, 2]> (Propc (1))) 1 =
[Gm [2,5] (Notm Propm (0) Andm Propm (1)),
Propm (0) Rm [2,3] Propm (1) Andm Fm [2,3] Propm (0),
Gm [2,3] (Notm Propm (0) Andm Propm (1)) Andm (Propm (0) Rm [4,5] Propm
(1) Andm Fm [4,5] Propm (0))]

```

```

value LP-mltl ((Fc[0,3] <[1,1,1,1]> (Propc (0::nat))) Orc
(Gc[0,3] <[1,1,1,1]> (Propc (1)))) 3 =
[Fm [0,0] Propm (0) Andm Gm [0,3] Propm (1),
(Gm [0,0] (Notm Propm (0)) Andm Fm [1,1] Propm (0)) Andm Gm [0,3] Propm
(1),
(Gm [0,1] (Notm Propm (0)) Andm Fm [2,2] Propm (0)) Andm Gm [0,3] Propm
(1),
(Gm [0,2] (Notm Propm (0)) Andm Fm [3,3] Propm (0)) Andm Gm [0,3] Propm
(1),
Gm [0,3] (Notm Propm (0)) Andm Gm [0,3] Propm (1),
Fm [0,0] Propm (0) Andm Fm [0,3] (Notm Propm (1)),
(Gm [0,0] (Notm Propm (0)) Andm Fm [1,1] Propm (0)) Andm Fm [0,3] (Notm

```

$Prop_m (1)$,
 $(G_m [0,1] (Not_m Prop_m (0)) And_m F_m [2,2] Prop_m (0)) And_m F_m [0,3] (Not_m Prop_m (1))$,
 $Prop_m (1)$,
 $(G_m [0,2] (Not_m Prop_m (0)) And_m F_m [3,3] Prop_m (0)) And_m F_m [0,3] (Not_m Prop_m (1))]$

end

theory *MLTL-Language-Partition-Proof*

imports *MLTL-Language-Partition-Algorithm*

begin

4 Properties of convert nnf ext

lemma *convert-nnf-and-convert-nnf-ext:*

shows $to_mttl (convert_nnf_ext \varphi) =$
 $convert_nnf (to_mttl \varphi)$

proof (*induct depth-mttl (to-mttl φ) arbitrary: φ rule: less-induct*)

case *less*

have *not:* $(\bigwedge \varphi. depth_mttl (to_mttl \varphi)$
 $< Suc (depth_mttl (to_mttl \psi)) \implies$
 $to_mttl (convert_nnf_ext \varphi) =$
 $convert_nnf (to_mttl \varphi) \implies$
 $\varphi = Not_c \psi \implies$
 $to_mttl (convert_nnf_ext (Not_c \psi)) =$
 $convert_nnf (Not_m (to_mttl \psi))$ **for** ψ

proof–

assume *ih:* $(\bigwedge \varphi. depth_mttl (to_mttl \varphi)$
 $< Suc (depth_mttl (to_mttl \psi)) \implies$
 $to_mttl (convert_nnf_ext \varphi) =$
 $convert_nnf (to_mttl \varphi)$

assume *shape:* $\varphi = Not_c \psi$

show *?thesis*

using *less ih shape* **by** (*induct ψ) simp-all*

qed

show *?case* **using** *less not*

by(*cases φ) auto*

qed

lemma *convert-nnf-ext-to-mttl-commute:*

shows $(convert_nnf (to_mttl \varphi)) = (to_mttl (convert_nnf_ext \varphi))$

proof(*induct depth-mttl (to-mttl φ) arbitrary: φ rule: less-induct*)

case *less*

then show *?case*

proof (*cases φ)*

case *True-mttl-ext*

then show *?thesis*


```

unfolding True-mltl-ext convert-nnf.simps convert-nnf-ext.simps to-mltl.simps
semantic-equiv-def
  by simp
next
  case False-mltl-ext
  then show ?thesis
  unfolding False-mltl-ext convert-nnf.simps convert-nnf-ext.simps to-mltl.simps
semantic-equiv-def
  by simp
next
  case (Prop-mltl-ext p)
  then show ?thesis
  unfolding Prop-mltl-ext convert-nnf.simps convert-nnf-ext.simps to-mltl.simps
semantic-equiv-def
  by simp
next
  case (Not-mltl-ext F)
  then have  $\varphi$ -is:  $\varphi = \text{Not}_c F$ 
  by blast
  show ?thesis
  proof(cases F)
  case True-mltl-ext
  then show ?thesis using  $\varphi$ -is less semantic-equiv-def by auto
next
  case False-mltl-ext
  then show ?thesis using  $\varphi$ -is less semantic-equiv-def by auto
next
  case (Prop-mltl-ext p)
  then show ?thesis using  $\varphi$ -is less semantic-equiv-def by auto
next
  case (Not-mltl-ext F1)
  then show ?thesis using  $\varphi$ -is less semantic-equiv-def by auto
next
  case (And-mltl-ext F1 F2)
  have  $r1$ :  $\text{Not}_m (\text{to-mltl } F1) = \text{to-mltl } (\text{Not}_c F1)$ 
  by simp
  have  $r2$ :  $\text{Not}_m (\text{to-mltl } F2) = \text{to-mltl } (\text{Not}_c F2)$ 
  by simp
  have rewrite:  $(\text{Or-mltl } (\text{convert-nnf } (\text{Not}_m (\text{to-mltl } F1)))$ 
     $(\text{convert-nnf } (\text{Not}_m (\text{to-mltl } F2)))) =$ 
     $(\text{Or-mltl } (\text{convert-nnf } (\text{to-mltl } (\text{Not}_c F1)))$ 
     $(\text{convert-nnf } (\text{to-mltl } (\text{Not}_c F2))))$ 
  using  $r1$   $r2$  by simp
  have  $ih1$ :  $(\text{convert-nnf } (\text{to-mltl } (\text{Not}_c F1))) =$ 
     $(\text{to-mltl } (\text{convert-nnf-ext } (\text{Not}_c F1)))$ 
  using less[of Notc F1] unfolding And-mltl-ext  $\varphi$ -is by simp
  have  $ih2$ :  $(\text{convert-nnf } (\text{to-mltl } (\text{Not}_c F2))) =$ 
     $(\text{to-mltl } (\text{convert-nnf-ext } (\text{Not}_c F2)))$ 
  using less[of Notc F2] unfolding And-mltl-ext  $\varphi$ -is by simp

```

```

have (Or-mltl (convert-nnf (to-mltl (Notc F1)))
      (convert-nnf (to-mltl (Notc F2))))
= (Or-mltl (to-mltl (convert-nnf-ext (Notc F1)))
  (to-mltl (convert-nnf-ext (Notc F2))))
  using ih1 ih2 unfolding semantic-equiv-def by auto
then show ?thesis
  unfolding φ-is And-mltl-ext to-mltl.simps convert-nnf.simps
  unfolding convert-nnf-ext.simps to-mltl.simps
  by simp
next
case (Or-mltl-ext F1 F2)
have r1: Notm (to-mltl F1) = to-mltl (Notc F1)
  by simp
have r2: Notm (to-mltl F2) = to-mltl (Notc F2)
  by simp
have rewrite: (Or-mltl (convert-nnf (Notm (to-mltl F1)))
  (convert-nnf (Notm (to-mltl F2)))) =
  (Or-mltl (convert-nnf (to-mltl (Notc F1)))
  (convert-nnf (to-mltl (Notc F2))))
  using r1 r2 by simp
have ih1: (convert-nnf (to-mltl (Notc F1))) =
  (to-mltl (convert-nnf-ext (Notc F1)))
  using less[of Notc F1] unfolding Or-mltl-ext φ-is by simp
have ih2: (convert-nnf (to-mltl (Notc F2))) =
  (to-mltl (convert-nnf-ext (Notc F2)))
  using less[of Notc F2] unfolding Or-mltl-ext φ-is by simp
have
(And-mltl (convert-nnf (to-mltl (Notc F1)))
  (convert-nnf (to-mltl (Notc F2)))) =
(And-mltl (to-mltl (convert-nnf-ext (Notc F1)))
  (to-mltl (convert-nnf-ext (Notc F2))))
  using ih1 ih2 unfolding semantic-equiv-def by auto
then show ?thesis
  unfolding φ-is Or-mltl-ext to-mltl.simps convert-nnf.simps
  unfolding convert-nnf-ext.simps to-mltl.simps
  by blast
next
case (Future-mltl-ext a b L F)
have r1: Notm (to-mltl F) = to-mltl (Notc F)
  by simp
then have rewrite: (Global-mltl a b (convert-nnf (Notm (to-mltl F)))) =
  (Global-mltl a b (convert-nnf (to-mltl (Notc F))))
  by simp
have ih: (convert-nnf (to-mltl (Notc F))) =
  (to-mltl (convert-nnf-ext (Notc F)))
  using less[of Notc F] φ-is unfolding Future-mltl-ext by simp
have (Global-mltl a b (convert-nnf (to-mltl (Notc F)))) =
  (Global-mltl a b (to-mltl (convert-nnf-ext (Notc F))))
  using ih unfolding semantic-equiv-def by auto

```

```

then show ?thesis
  unfolding  $\varphi$ -is Future-mltl-ext to-mltl.simps convert-nnf.simps
  unfolding convert-nnf-ext.simps to-mltl.simps
  using rewrite by blast
next
case (Global-mltl-ext a b L F)
have r1:  $\text{Not}_m$  (to-mltl F) = to-mltl ( $\text{Not}_c$  F)
  by simp
then have rewrite: (Global-mltl a b (convert-nnf ( $\text{Not}_m$  (to-mltl F)))) =
  (Global-mltl a b (convert-nnf (to-mltl ( $\text{Not}_c$  F))))
  by simp
have ih: (convert-nnf (to-mltl ( $\text{Not}_c$  F))) =
  (to-mltl (convert-nnf-ext ( $\text{Not}_c$  F)))
  using less[of  $\text{Not}_c$  F]  $\varphi$ -is unfolding Global-mltl-ext by simp
have (Future-mltl a b (convert-nnf (to-mltl ( $\text{Not}_c$  F)))) =
(Future-mltl a b (to-mltl (convert-nnf-ext ( $\text{Not}_c$  F))))
  using ih unfolding semantic-equiv-def by auto
then show ?thesis
  unfolding  $\varphi$ -is Global-mltl-ext to-mltl.simps convert-nnf.simps
  unfolding convert-nnf-ext.simps to-mltl.simps
  using rewrite by simp
next
case (Until-mltl-ext F1 a b L F2)
have r1:  $\text{Not}_m$  (to-mltl F1) = to-mltl ( $\text{Not}_c$  F1)
  by simp
have r2:  $\text{Not}_m$  (to-mltl F2) = to-mltl ( $\text{Not}_c$  F2)
  by simp
have rewrite: (Or-mltl (convert-nnf ( $\text{Not}_m$  (to-mltl F1)))
  (convert-nnf ( $\text{Not}_m$  (to-mltl F2)))) =
  (Or-mltl (convert-nnf (to-mltl ( $\text{Not}_c$  F1)))
  (convert-nnf (to-mltl ( $\text{Not}_c$  F2))))
  using r1 r2 by simp
have ih1: (convert-nnf (to-mltl ( $\text{Not}_c$  F1))) =
  (to-mltl (convert-nnf-ext ( $\text{Not}_c$  F1)))
  using less[of  $\text{Not}_c$  F1] unfolding Until-mltl-ext  $\varphi$ -is by simp
have ih2: (convert-nnf (to-mltl ( $\text{Not}_c$  F2))) =
  (to-mltl (convert-nnf-ext ( $\text{Not}_c$  F2)))
  using less[of  $\text{Not}_c$  F2] unfolding Until-mltl-ext  $\varphi$ -is by simp
have
  (Release-mltl (convert-nnf (to-mltl ( $\text{Not}_c$  F1))) a b
  (convert-nnf (to-mltl ( $\text{Not}_c$  F2)))) =
  (Release-mltl (to-mltl (convert-nnf-ext ( $\text{Not}_c$  F1))) a b
  (to-mltl (convert-nnf-ext ( $\text{Not}_c$  F2))))
  using ih1 ih2 unfolding semantic-equiv-def by auto
then show ?thesis
  unfolding  $\varphi$ -is Until-mltl-ext to-mltl.simps convert-nnf.simps
  unfolding convert-nnf-ext.simps to-mltl.simps
  by blast
next

```

```

case (Release-mltl-ext  $F1$   $a$   $b$   $L$   $F2$ )
have  $r1$ :  $Not_m$  (to-mltl  $F1$ ) = to-mltl ( $Not_c$   $F1$ )
  by simp
have  $r2$ :  $Not_m$  (to-mltl  $F2$ ) = to-mltl ( $Not_c$   $F2$ )
  by simp
have rewrite: (Or-mltl (convert-nnf ( $Not_m$  (to-mltl  $F1$ )))
  (convert-nnf ( $Not_m$  (to-mltl  $F2$ )))) =
  (Or-mltl (convert-nnf (to-mltl ( $Not_c$   $F1$ )))
  (convert-nnf (to-mltl ( $Not_c$   $F2$ ))))
  using  $r1$   $r2$  by simp
have  $ih1$ : (convert-nnf (to-mltl ( $Not_c$   $F1$ ))) =
  (to-mltl (convert-nnf-ext ( $Not_c$   $F1$ )))
  using less[of  $Not_c$   $F1$ ] unfolding Release-mltl-ext  $\varphi$ -is by simp
have  $ih2$ : (convert-nnf (to-mltl ( $Not_c$   $F2$ ))) =
  (to-mltl (convert-nnf-ext ( $Not_c$   $F2$ )))
  using less[of  $Not_c$   $F2$ ] unfolding Release-mltl-ext  $\varphi$ -is by simp
have
  (Until-mltl (convert-nnf (to-mltl ( $Not_c$   $F1$ )))  $a$   $b$ 
  (convert-nnf (to-mltl ( $Not_c$   $F2$ )))) =
  (Until-mltl (to-mltl (convert-nnf-ext ( $Not_c$   $F1$ )))  $a$   $b$ 
  (to-mltl (convert-nnf-ext ( $Not_c$   $F2$ ))))
  using  $ih1$   $ih2$  unfolding semantic-equiv-def by auto
then show ?thesis
  unfolding  $\varphi$ -is Release-mltl-ext to-mltl.simps convert-nnf.simps
  unfolding convert-nnf-ext.simps to-mltl.simps
  by blast
qed
next
case (And-mltl-ext  $F1$   $F2$ )
show ?thesis
  unfolding And-mltl-ext to-mltl.simps convert-nnf.simps convert-nnf-ext.simps
semantic-equiv-def
  using less[of  $F1$ ] less[of  $F2$ ] And-mltl-ext unfolding semantics-mltl.simps
semantic-equiv-def by auto
next
case (Or-mltl-ext  $F1$   $F2$ )
then show ?thesis
  unfolding Or-mltl-ext to-mltl.simps convert-nnf.simps convert-nnf-ext.simps
semantic-equiv-def
  using less[of  $F1$ ] less[of  $F2$ ] Or-mltl-ext unfolding semantics-mltl.simps
semantic-equiv-def by simp
next
case (Future-mltl-ext  $a$   $b$   $L$   $F$ )
show ?thesis
  unfolding Future-mltl-ext to-mltl.simps convert-nnf.simps convert-nnf-ext.simps
to-mltl.simps
  using less[of  $F$ ] Future-mltl-ext unfolding semantic-equiv-def semantics-mltl.simps
by simp
next

```

```

    case (Global-mltl-ext a b L F)
    then show ?thesis
    unfolding Global-mltl-ext to-mltl.simps convert-nnf.simps convert-nnf-ext.simps
to-mltl.simps
    using less[of F] Global-mltl-ext unfolding semantic-equiv-def semantics-mltl.simps
by simp
next
    case (Until-mltl-ext F1 a b L F2)
    then show ?thesis
    unfolding Until-mltl-ext to-mltl.simps convert-nnf.simps convert-nnf-ext.simps
to-mltl.simps
    using less[of F1] less[of F2] Until-mltl-ext unfolding semantic-equiv-def
semantics-mltl.simps by simp
next
    case (Release-mltl-ext F1 a b L F2)
    then show ?thesis
    unfolding Release-mltl-ext to-mltl.simps convert-nnf.simps convert-nnf-ext.simps
to-mltl.simps
    using less[of F1] less[of F2] Release-mltl-ext unfolding semantic-equiv-def
semantics-mltl.simps by simp
qed
qed

```

lemma *convert-nnf-ext-preserves-semantics:*

```

    assumes intervals-welldef (to-mltl  $\varphi$ )
    shows (convert-nnf-ext  $\varphi$ )  $\equiv_c$   $\varphi$ 
proof –
    have semantic-equiv (convert-nnf (to-mltl  $\varphi$ )) (to-mltl  $\varphi$ )
    using assms convert-nnf-preserves-semantics[of (to-mltl  $\varphi$ )]
    unfolding semantic-equiv-ext-def semantic-equiv-def by blast
    then show ?thesis
    using convert-nnf-ext-to-mltl-commute
    unfolding semantic-equiv-ext-def semantic-equiv-def by metis
qed

```

lemma *convert-nnf-ext-convert-nnf-ext:*

```

    shows convert-nnf-ext  $\varphi$  = convert-nnf-ext (convert-nnf-ext  $\varphi$ )
proof(induction depth-mltl (to-mltl  $\varphi$ ) arbitrary:  $\varphi$  rule: less-induct)
    case less
    have not-case: ( $\bigwedge F$ . depth-mltl (to-mltl F) <
    Suc (depth-mltl (to-mltl G))  $\implies$ 
    convert-nnf-ext (convert-nnf-ext F) = convert-nnf-ext F)  $\implies$ 
 $\varphi$  = Notc G  $\implies$ 
    convert-nnf-ext (convert-nnf-ext (Notc G)) =
    convert-nnf-ext (Notc G) for G
proof –
    assume ind-h: ( $\bigwedge F$ . depth-mltl (to-mltl F) <
    Suc (depth-mltl (to-mltl G))  $\implies$ 

```

```

      convert-nnf-ext (convert-nnf-ext F) = convert-nnf-ext F)
    assume  $\varphi$ -is:  $\varphi = \text{Not}_c G$ 
    show ?thesis using less  $\varphi$ -is by (cases G) simp-all
  qed
  show ?case using less not-case
  by (cases  $\varphi$ ) fastforce+
qed

```

4.1 Cases where to mttl is bijective

```

lemma to-mltl-true-bijective:
  assumes to-mltl  $\varphi = \text{True}_m$ 
  shows  $\varphi = \text{True}_c$ 
  using assms by (cases  $\varphi$ ) simp-all

```

```

lemma to-mltl-false-bijective:
  assumes to-mltl  $\varphi = \text{False}_m$ 
  shows  $\varphi = \text{False}_c$ 
  using assms by (cases  $\varphi$ ) simp-all

```

```

lemma to-mltl-prop-bijective:
  assumes to-mltl  $\varphi = \text{Prop}_m (p)$ 
  shows  $\varphi = \text{Prop}_c (p)$ 
  using assms by (cases  $\varphi$ ) simp-all

```

```

lemma to-mltl-not-prop-bijective:
  assumes to-mltl  $\varphi = \text{Not}_m (\text{Prop}_m (p))$ 
  shows  $\varphi = \text{Not}_c (\text{Prop}_c (p))$ 
  using assms by (cases  $\varphi$ ) (simp-all add: to-mltl-prop-bijective)

```

5 Lemmas about Integer Composition

```

lemma composition-length-ub:
  fixes  $n::\text{nat}$  and  $L::\text{nat list}$ 
  assumes is-composition  $n L$ 
  shows length  $L \leq n$ 
  using assms unfolding is-composition-def
proof (induct L arbitrary: n)
  case Nil
  then show ?case by simp
next
  case (Cons a L)
  have listsum: sum-list (a # L) = a + sum-list L
  by simp
  then have ls-L: sum-list L = n - a
  using Cons(2) by auto
  then have Lprop: ( $\forall i. \text{List.member } L i \longrightarrow 0 < i$ )  $\wedge$  sum-list L = n - a
  using Cons(2)
  by (meson member-rec(1))

```

```

then have len-leq:  $\text{length } L \leq n - a$ 
  using Cons(1)[OF Lprop]
  by auto
have  $a > 0$ 
  using Cons(2)
  by (meson member-rec(1))
then show ?case using len-leq
  using Cons.premis listsum by auto
qed

```

```

lemma composition-length-lb:
  fixes  $n::\text{nat}$  and  $L::\text{nat list}$ 
  assumes is-composition  $n L$ 
  assumes  $n > 0$ 
  shows  $0 < \text{length } L$ 
proof -
  have  $\neg(0 < \text{length } L) \implies \text{False}$ 
  proof -
    assume  $\neg(0 < \text{length } L)$ 
    then have  $\text{length } L = 0$ 
      by simp
    then have  $\text{sum-list } L = 0$ 
      by simp
    then show ?thesis
      using assms unfolding is-composition-def
      by simp
  qed
then show ?thesis using assms by blast
qed

```

```

lemma interval-times-length:
  fixes  $a::\text{nat}$  and  $L::\text{nat list}$ 
  shows  $\text{length } (\text{interval-times } a L) = \text{length } L + 1$ 
  unfolding interval-times-def by auto

```

```

lemma interval-times-first:
  fixes  $a::\text{nat}$  and  $L::\text{nat list}$ 
  shows  $(\text{interval-times } a L)!0 = a$ 
proof -
  have  $\text{map } (\lambda i. a + \text{partial-sum } L i) [0..<\text{length } L + 1] ! 0 =$ 
     $(\lambda i. a + \text{partial-sum } L i) 0$ 
  by (metis Nat.add-0-right add-gr-0 less-numeral-extra(1) nth-map-upt zero-less-diff)

then have  $\text{map } (\lambda i. a + \text{partial-sum } L i) [0..<\text{length } L + 1] ! 0 = a$ 
  unfolding partial-sum-def by auto
then show ?thesis
  unfolding interval-times-def by blast

```

qed

lemma *interval-times-last*:

fixes $a b :: \text{nat}$ **and** $L :: \text{nat list}$

assumes *int-welldef*: $a \leq b$

assumes *composition*: *is-composition* $(b-a+1)$ L

shows $(\text{interval-times } a \ L)!(\text{length } L) = b+1$

proof –

have *partial-sum* L $(\text{length } L) = \text{sum-list } L$

unfolding *partial-sum-def* **by** *auto*

then have $a + \text{partial-sum } L$ $(\text{length } L) = b+1$

using *assms* **unfolding** *is-composition-def*

by *simp*

then show *?thesis*

unfolding *interval-times-def*

by (*metis* *add-0* *add-diff-cancel-left'* *less-add-one* *nth-map-upt*)

qed

lemma *interval-times-diff*:

fixes $a b i :: \text{nat}$ **and** $L :: \text{nat list}$

assumes *int-welldef*: $a \leq b$

assumes *composition*: *is-composition* $(b-a+1)$ L

assumes *i-index*: $i < \text{length } L$

assumes *s-is*: $s = \text{interval-times } a \ L$

shows $s!(i+1) - s!(i) = L!i$

proof –

have *ip1*: $s!(i+1) = a + \text{partial-sum } L$ $(i+1)$

using *s-is* *i-index* **unfolding** *interval-times-def*

by (*metis* (*no-types*, *lifting*) *add-0* *add-mono1* *diff-zero* *nth-map-upt*)

have *i*: $s!i = a + \text{partial-sum } L$ i

using *s-is* *i-index* **unfolding** *interval-times-def*

by (*metis* (*no-types*, *lifting*) *add commute* *add-0* *add-strict-increasing* *diff-zero*

less-numeral-extra(1) *less-or-eq-imp-le* *nth-map-upt*)

have *s-iat*: $s!(i+1) - s!i = \text{partial-sum } L$ $(i+1) - \text{partial-sum } L$ i

using *ip1* *i*

by *auto*

have *take-is*: $\text{take } (i+1) \ L = (\text{take } i \ L) @ [L!i]$

by (*simp* *add*: *i-index* *take-Suc-conv-app-nth*)

have *li*: $\text{foldr } (+) [L!i] \ 0 = L!i$

by *force*

have $\bigwedge a :: \text{nat}.$ $\text{foldr } (+) \ L \ a = a + \text{foldr } (+) \ L \ 0$ **for** $L :: \text{nat list}$

proof (*induct* L)

case *Nil*

then show *?case* **by** *auto*

next

case (*Cons* $h \ T$)

then show *?case*

by (*metis* *add.left-commute* *foldr.simps*(2) *o-apply*)


```

qed
then have foldr (+) (take i L) (L!i) = L ! i + foldr (+) (take i L) 0
  by blast
then have foldr (+) ((take i L) @ [L ! i]) 0 - foldr (+) (take i L) 0 = L ! i
  using foldr-append[of (+) take i L [L ! i] 0] li
  by simp
then have sum-list (take (i + 1) L) - sum-list (take i L) = L ! i
  using i-index take-is by simp
then show ?thesis
  using i-index composition unfolding is-composition-def
  partial-sum-def s-iat by blast
qed

```

```

lemma interval-times-diff-ge:
  fixes a b i::nat and L::nat list
  assumes int-welldef: a ≤ b
  assumes composition: is-composition (b-a+1) L
  assumes i-index: i < length L
  assumes s-is: s = interval-times a L
  shows s!(i+1) > s!(i)
proof -
  have diff: s!(i+1) - s!(i) = L!i
    using assms interval-times-diff by blast
  have gap: L!i > 0 using assms(2) unfolding is-composition-def
    by (meson i-index in-set-member nth-mem)
  show ?thesis using diff gap by simp
qed

```

```

lemma interval-times-diff-ge-general:
  fixes a b i j::nat and L::nat list
  assumes int-welldef: a ≤ b
  assumes composition: is-composition (b-a+1) L
  assumes j-index: j ≤ length L
  assumes i-le-j: i < j
  assumes s-is: s = interval-times a L
  shows s!j > s!i
  using assms
proof (induct j-1 arbitrary: i j)
  case 0
  then have i = 0 and j = 1
    by simp-all
  then show ?case
    using interval-times-diff-ge 0 by fastforce
next
  case (Suc x)
  then have j-eq: j = x+2
    by simp
  have high: s ! (x + 1) < s ! (x + 2)
    using interval-times-diff-ge[of a b L x+1 s] Suc by simp

```

```

{
  assume i-eq:  $i = x+1$ 
  then have ?case unfolding i-eq j-eq
    using high by simp
} moreover {
  assume i-eq:  $i \leq x$ 
  then have  $s! i < s! (x+1)$ 
    using Suc.hyps(1)[of x+1 i] Suc by force
  then have ?case using high i-eq j-eq by simp
}
ultimately show ?case using Suc j-eq by linarith
qed

```

lemma *trivial-composition*:

```

assumes  $n > 0$ 
shows is-composition  $n [n]$ 
proof -
  have pos:  $(\forall i. \text{List.member } [n] i \longrightarrow 0 < i)$ 
    unfolding List.member-def
    by (simp add: asms)
  have sum:  $\text{sum-list } [n] = n$ 
    by simp
  show ?thesis unfolding is-composition-def
    using pos sum by blast
qed

```

lemma *sum-list-pos*: $(\bigwedge x. x \in \text{set } (xs::\text{nat list}) \implies 0 < x) \implies \text{length } xs > 0 \implies 0 < \text{sum-list } xs$

by (*induction xs*) *auto*

lemma *take-prefix*:

```

assumes  $L = H@[t]$ 
assumes  $k \leq \text{length } L - 1$ 
shows  $\text{take } k H = \text{take } k L$ 
using asms by auto

```

lemma *take-interval-times*:

```

assumes  $\text{length } L \geq k$ 
shows  $\text{take } (k+1) (\text{interval-times } a L) = \text{interval-times } a (\text{take } k L)$ 
using asms
proof(induct length L arbitrary: L)
  case 0
  then show ?case
    by (simp add: interval-times-length)
next
  case (Suc x)
  then obtain  $H t$  where  $L\text{-eq}: L = H@[t]$ 
    by (metis length-Suc-conv-rev)

```

```

have ih: take (k + 1) (interval-times a H) = interval-times a (take k H)
  using Suc.hyps(1)[of H] Suc L-eq
  by (metis Suc-eq-plus1 add-left-cancel interval-times-length le-SucE le-add1
length-append-singleton plus-1-eq-Suc take-all-iff)
have length-it: length (interval-times a L) = length L + 1
  unfolding interval-times-def by auto
{
  assume *: k ≤ length L - 1
  then have eq1: (take k H) = (take k L)
    by (simp add: L-eq)
  have (interval-times a H)@[a+(sum-list L)] = interval-times a L
    using L-eq unfolding interval-times-def partial-sum-def by auto
  then have eq2: take (k + 1) (interval-times a H) = take (k + 1) (interval-times
a L)
    using take-prefix[of interval-times a L interval-times a H a + sum-list L]
    by (metis Suc-eq-plus1 diff-Suc-1 eq1 ih interval-times-length not-less-eq-eq
take-all)
  have ?case using eq1 eq2 ih by argo
} moreover {
  assume *: k = length L
  then have ?case
    by (simp add: length-it)
}
ultimately show ?case using Suc by linarith
qed

```

```

lemma index-list-index:
  fixes k::nat
  assumes j < k
  shows [0 ..< k] ! j = j
  using assms by simp

```

```

lemma interval-times-obtain-aux:
  assumes a ≤ b
  assumes is-composition (b - a + 1) L
  assumes s = interval-times a L
  assumes (s ! 1) ≤ t ∧ t ≤ b
  shows ∃ i. s ! i ≤ t ∧ t ≤ s ! (i + 1) - 1 ∧ 1 ≤ i ∧ i < length L
proof -
  have length-s: length s = length L + 1
    using assms interval-times-length by auto
  have first: s!0 = a
    using interval-times-first assms by blast
  have last: s!(length L) = b+1
    using interval-times-last assms by blast
  {
    assume length-L: length L = 0
    then have ?thesis using assms
  }

```

```

    by (metis first last less-add-one verit-comp-simplify1(3))
  } moreover {
    assume length-L: length L ≥ 1
    have ?thesis using assms first last length-s length-L
    proof(induct length L - 1 arbitrary: s L a b t)
      case 0
      then show ?case by auto
    next
      case (Suc x)
      then have length-L: length L ≥ 2 by linarith
      then have length-s: length s ≥ 3 using Suc by linarith
      {
        assume *: t < s!(length L-1)
        let ?L' = take (length L-1) L
        let ?s' = take (length L) s
        let ?b' = b - (List.last L)
        have pos-L: (∀ i. List.member L i → 0 < i) and
          sum-L: sum-list L = b - a + 1
          using Suc(4) unfolding is-composition-def by auto
        have List.member L (last L) unfolding List.member-def
          by (metis Suc.prem1(8) last-in-set length-0-conv not-one-le-zero)
        have sum-list-eq: sum-list L = sum-list (take (length L-1) L) + last L
          using length-L
        proof(induct length L arbitrary: L)
          case 0
          then show ?case by auto
        next
          case (Suc xa)
          then obtain h T where L-eq: L = h#T
            by (meson Suc-length-conv)
          then have L-decomp: sum-list L = sum-list T + h by simp
          {
            assume length L = 2
            then obtain x1 x2 where L = [x1, x2]
              by (metis Suc-1 Suc-length-conv gen-length-code(1) gen-length-def
                impossible-Cons le-add2 list.exhaust plus-1-eq-Suc)
            then have ?case by auto
          } moreover {
            assume length-L: length L > 2
            then have last: last T = last L
              using L-eq by auto
            have *: sum-list T = sum-list (take (length T - 1) T) + last T
              using Suc.hyps(1)[of T] L-decomp L-eq length-L
              by (metis Suc.hyps(2) add-diff-cancel-left' length-Cons less-Suc-eq-le
                plus-1-eq-Suc)
            have **: h + sum-list (take (length T - 1) T) = sum-list (take (length
              L - 1) L)
              using L-eq
            by (metis (no-types, opaque-lifting) Suc.prem1 Suc-1 Suc-eq-plus1 Suc-le-D

```

add-diff-cancel-right' add-le-same-cancel2 length-Cons not-less-eq-eq sum-list.Cons take-Suc-Cons)

```

    have ?case using * ** last
      using L-decomp by presburger
  }
  ultimately show ?case using Suc.prem1 by fastforce
qed
have pos-preL: ( $\bigwedge x. x \in \text{set } (\text{take } (\text{length } L - 1) L) \implies 0 < x$ )
  using pos-L
  by (metis in-set-member in-set-takeD)
have length-preL:  $0 < \text{length } (\text{take } (\text{length } L - 1) L)$ 
  using length-L by auto
have sum-preL-pos:  $\text{sum-list } (\text{take } (\text{length } L - 1) L) > 0$ 
  using sum-list-pos[of take (length L - 1) L]
  using pos-preL length-preL by blast
then have sum-last:  $\text{sum-list } L > \text{last } L$  using pos-L length-L
  using sum-list-pos sum-list-eq by linarith
then have b-lastL:  $b \geq \text{last } L$ 
  using sum-L by auto
then have ba-lastL:  $\text{last } L \leq b - a$ 
  using sum-L sum-last by auto
have first:  $s!0 = a$ 
  using Suc interval-times-first by blast
have last:  $s!(\text{length } L) = b+1$ 
  using Suc interval-times-last by blast
have c1:  $x = \text{length } (\text{take } (\text{length } L - 1) L) - 1$ 
  using Suc by auto
have c2:  $a \leq b - \text{last } L$ 
  using Suc(3) b-lastL ba-lastL by auto
have c3 :is-composition (b - last L - a + 1) (take (length L - 1) L)
  using Suc.prem2 unfolding is-composition-def
  by (metis Suc-diff-1 Suc-eq-plus1 <0 < sum-list (take (length L - 1) L)
L)
have c4:  $\text{take } (\text{length } L) s = \text{interval-times } a (\text{take } (\text{length } L - 1) L)$ 
  unfolding Suc(5) using length-L take-interval-times
  by (metis Suc.prem8 diff-add diff-le-self)
have c5:  $\text{take } (\text{length } L) s ! 1 \leq t \wedge t \leq b - \text{last } L$ 
proof-
  have s!(length L-1) = a + sum-list (take (length L-1) L)
    unfolding Suc(5) interval-times-def partial-sum-def
  by (metis (no-types, lifting) Suc.prem8 add commute add-0 add-mono-thms-linordered-field(3)
le-add-same-cancel2 less-numeral-extra(1) nth-map-upt ordered-cancel-comm-monoid-diff-class.add-diff-inverse
zero-le)
  then have part1:  $(s ! (\text{length } L - 1)) - 1 \leq b - \text{last } L$ 
    using last sum-list-eq
  by (metis (no-types, lifting) One-nat-def Suc-leI sum-preL-pos c2 c3
diff-add-inverse2 eq-imp-le is-composition-def order-eq-refl ordered-cancel-comm-monoid-diff-class.add-diff-inverse
ordered-cancel-comm-monoid-diff-class.diff-add-assoc)

```

```

have part2: take (length L) s ! 1 ≤ t
  using Suc.hyps(2) Suc.prem(4) by auto
then show ?thesis using * part1 part2
  by linarith
qed
have c6: take (length L) s ! 0 = a
  by (simp add: c4 interval-times-first)
have c7: take (length L) s ! length (take (length L - 1) L) = b - last L + 1
proof-
  have idx: length (take (length L - 1) L) = length L - 1 by simp
  have p1: a + partial-sum L (length L - 1) = b - last L + 1
    unfolding partial-sum-def
  by (metis add.assoc c2 c3 is-composition-def ordered-cancel-comm-monoid-diff-class.add-diff-inverse)

  have p2: take (length L) (map (λi. a + partial-sum L i) [0..<length L +
1]) ! (length L - 1)
    = (map (λi. a + partial-sum L i) [0..<length L + 1]) ! (length L - 1)
    by (meson Suc.prem(2) add-gr-0 composition-length-lb diff-less nth-take
zero-less-one)
  have p3: (map (λi. a + partial-sum L i) [0..<length L + 1]) ! (length L
- 1)
    = a + partial-sum L (length L - 1)
proof-
  have fact1: map (λi. a + partial-sum L i) [0..<length L + 1] ! (length
L - 1) =
    a + partial-sum L ([0..<length L + 1] ! (length L - 1))
    using nth-map[of (length L - 1) [0..<length L + 1] (λi. a + partial-sum
L i)]
    by simp
  have length L ≥ 0
    using Suc(2) by auto
  then have fact2: ((0::nat)..<length L + 1] ! (length L - 1)) = length
L - 1
    using index-list-index[of length L - 1 length L + 1] by simp
  then show ?thesis using fact1 fact2 by argo
qed
then have take (length L) s ! (length L - 1) = b - last L + 1
  unfolding Suc(5) interval-times-def
  using p1 p2 p3 by argo
then show ?thesis using idx by argo
qed
have c8: length (take (length L) s) = length (take (length L - 1) L) + 1
  using c4 interval-times-length by presburger
have c9: 1 ≤ length (take (length L - 1) L)
  using length-preL by linarith
have ih: ∃ i. take (length L) s ! i ≤ t ∧ t ≤ take (length L) s ! (i + 1) - 1
  ∧ 1 ≤ i ∧ i < length (take (length L - 1) L)
  using Suc(1)[of (take (length L - 1) L) a b - last L take (length L) s t,
OF c1 c2 c3 c4 c5 c6 c7 c8 c9] by blast

```

```

    then obtain  $i$  where  $t$ -bound: take (length  $L$ )  $s ! i \leq t \wedge t \leq$  take (length
 $L$ )  $s ! (i + 1) - 1$ 
      and  $i$ -bound:  $1 \leq i \wedge i < \text{length (take (length } L - 1) L)$ 
    by blast
    have  $i$ -bound- $L$ :  $1 \leq i \wedge i < \text{length } L$ 
    using  $i$ -bound by auto
    then have  $t$ -bound- $L$ :  $s ! i \leq t \wedge t \leq s ! (i + 1) - 1$ 
    using  $t$ -bound
      by (metis Suc.hyps(2) c1 c9  $i$ -bound le-add-diff-inverse less-diff-conv
nth-take plus-1-eq-Suc)
    then have ?case using  $i$ -bound- $L$   $t$ -bound by auto
  } moreover {
    assume *:  $t \geq s!(\text{length } L - 1)$ 
    then have ?case
      by (metis Suc.hyps(2) Suc.prem(4) Suc.prem(6) Suc.prem(8) add-diff-cancel-right'
diff-less le-add1 le-add-diff-inverse2 less-numeral-extra(1) order-less-le-trans plus-1-eq-Suc)
  }
  ultimately show ?case by fastforce
qed
}
ultimately show ?thesis
  by (meson less-one verit-comp-simplify1(3))
qed

```

lemma *interval-times-obtain*:

```

assumes  $a \leq b$ 
assumes  $is$ -composition  $(b - a + 1) L$ 
assumes  $s = interval$ -times  $a L$ 
assumes  $a \leq t \wedge t \leq b$ 
shows  $\exists i. s ! i \leq t \wedge t \leq s ! (i + 1) - 1 \wedge 0 \leq i \wedge i < \text{length } L$ 
proof -
{
  assume *:  $(s ! 1) \leq t$ 
  from  $interval$ -times-obtain-aux[OF assms(1-3), of  $t$ ] * assms(4)
  obtain  $i$  where  $s ! i \leq t \wedge t \leq s ! (i + 1) - 1 \wedge 1 \leq i \wedge i < \text{length } L$ 
    by auto
  then have ?thesis by blast
} moreover {
  assume *:  $t < s!1$ 
  have  $s$ first:  $s!0 = a$ 
    using  $interval$ -times-first unfolding assms by auto
  have  $length$ - $L$ :  $0 < \text{length } L$ 
    using  $composition$ -length-lb[OF assms(2)] using assms by auto
  have  $t \leq s ! 1 - 1$ 
    using * by simp
  then have  $s ! 0 \leq t \wedge t \leq s ! 1 - 1 \wedge 0 \leq (0::nat) \wedge 0 < \text{length } L$ 
    using * assms unfolding  $s$ first using  $length$ - $L$  by blast
}

```

```

    then have ?thesis by auto
  }
  ultimately show ?thesis by force
qed

```

```

lemma list-allones:
  assumes  $\forall i < \text{length } L. L!i = 1$ 
  shows  $L = \text{map } (\lambda i. 1) [0 .. < \text{length } L]$ 
  using assms
proof(induct L)
  case Nil
  then show ?case by simp
next
  case (Cons a L)
  then show ?case
    by (metis (no-types, lifting) length-map list-eq-iff-nth-eq map-nth nth-map)
qed

```

```

lemma sum-list-constants:
  fixes  $L::\text{nat list}$  and  $k::\text{nat}$ 
  assumes  $\forall i < \text{length } L. L!i = k$ 
  shows  $\text{sum-list } L = k * (\text{length } L)$ 
  using assms by(induct L) force+

```

```

lemma length-is-composition-allones:
  assumes is-composition-allones  $n$   $L$ 
  shows  $\text{length } L = n$ 
  using assms unfolding is-composition-allones-def is-composition-def
  by (metis mult-1 sum-list-constants)

```

```

lemma partial-sum-allones:
  assumes  $(\forall i < \text{length } L. L!i = 1)$ 
  assumes  $i \leq \text{length } L$ 
  shows  $\text{partial-sum } L\ i = i$ 
  using assms
proof(induct length L arbitrary: i L)
  case 0
  then have  $i0: i = 0$  by auto
  have  $L\text{-empty}: L = []$  using 0 by auto
  show ?case using  $L\text{-empty } i0$ 
    unfolding partial-sum-def by simp
next
  case (Suc x)
  then obtain  $H\ t$  where  $L\text{-is}: L = H@[t]$ 
    by (metis length-Suc-conv-rev)
  have  $L\text{-ones}: L = \text{map } (\lambda i. 1) [0 .. < \text{length } L]$ 
    using list-allones Suc by blast
  {

```



```

assume *:  $i = \text{length } L$ 
then have takeall:  $\text{take } i \ L = L$ 
  using take-all[of  $L$   $i$ ] by simp
have ?case unfolding takeall partial-sum-def
  using  $\text{Suc}(3) * \text{sum-list-constants}$ [of  $L$   $1$ ] by simp
} moreover {
  assume *:  $i < \text{length } L$ 
  have cond1:  $x = \text{length } H$ 
    using Suc L-is by simp
  have cond2:  $\forall i < \text{length } H. H ! i = 1$ 
    using  $\text{Suc}(3)$  unfolding L-is
    by (metis L-is Suc.hyps(2) Suc-lessD Suc-mono butlast-snoc cond1 nth-butlast)

  have cond3:  $i \leq \text{length } H$ 
    using * L-is by auto
  then have ?case
    using  $\text{Suc}(1)$ [of  $H$   $i$ , OF cond1 cond2 cond3]
    unfolding partial-sum-def L-is by simp
  }
ultimately show ?case using L-is Suc by fastforce
qed

```

lemma *interval-times-allones*:

```

assumes  $a \leq b$ 
assumes is-composition-allones ( $b - a + 1$ )  $L$ 
assumes  $i < \text{length } (\text{interval-times } a \ L)$ 
shows  $(\text{interval-times } a \ L) ! i = a + i$ 
proof -
  have *:  $\text{map } (\lambda i. a + \text{partial-sum } L \ i) \ [0..<\text{length } L + 1] ! i = a + \text{partial-sum } L \ i$ 
    using assms
    by (metis interval-times-def length-map length-upt nth-map-upt plus-nat.add-0)

  have allones:  $\forall i < \text{length } L. L ! i = 1$ 
    using assms(2) unfolding is-composition-allones-def
    by blast
  have  $\text{length } (\text{interval-times } a \ L) = \text{length } L + 1$ 
    using interval-times-length by simp
  then have  $\text{partial-sum } L \ i = i$ 
    using partial-sum-allones[of  $L$   $i$ ]
    using allones assms by simp
  then have  $a + \text{partial-sum } L \ i = a + i$ 
    by auto
  then show ?thesis
    unfolding interval-times-def
    using * by auto
qed

```

lemma *allones-implies-is-composition*:

assumes *is-composition-allones* n L
shows *is-composition* n L
using *assms unfolding is-composition-allones-def* **by** *blast*

lemma *allones-implies-is-composition-MTLT*:
assumes *is-composition-MTLT-allones* φ
shows *is-composition-MTLT* φ
using *assms allones-implies-is-composition*
by (*induct* φ) *simp-all*

6 MTLT Decomposition Lemmas

lemma *LP-mttl-nnf*:
fixes $\varphi::'a$ *mttl-ext* **and** $\psi::'a$ *mttl* **and** $k::nat$
assumes ψ -*coformula*: $\psi \in set (LP\text{-}mttl\ \varphi\ k)$
shows $\exists \psi$ -*init*. $\psi = convert\text{-}nnf\ \psi$ -*init*
proof –
obtain ψ -*init* **where** $\psi = to\text{-}mttl\ (convert\text{-}nnf\text{-}ext\ \psi$ -*init*)
using *assms unfolding LP-mttl.simps* **by** *auto*
then have $\psi = convert\text{-}nnf\ (to\text{-}mttl\ \psi$ -*init*)
using *convert-nnf-ext-to-mttl-commute* **by** *metis*
then show *?thesis*
by *blast*
qed

lemma *LP-mttl-element*:
fixes $\psi::'a$ *mttl* **and** $\varphi::'a$ *mttl-ext*
shows $\psi \in set (LP\text{-}mttl\ \varphi\ k) \longleftrightarrow$
 $(\exists \psi$ -*ext* $\in set (LP\text{-}mttl\text{-}aux\ (convert\text{-}nnf\text{-}ext\ \varphi)\ k).$
 $\psi = to\text{-}mttl\ (convert\text{-}nnf\text{-}ext\ \psi$ -*ext*)
unfolding *LP-mttl.simps* **by** *auto*

7 Lemmas for MTLT operators that operate over lists of mtl formulas

lemma *pairs-alt*:
shows $set (pairs\ L1\ (h2\ \#\ T2)) =$
 $set ((map\ (\lambda x. (x, h2))\ L1)\ @\ (pairs\ L1\ T2))$
proof(*induct* $L1$ *arbitrary*: $h2\ T2$)
case *Nil*
then show *?case* **by** *simp*
next
case (*Cons* $a\ L1$)
have *pairs-fact*: $set (pairs\ (a\ \#\ L1)\ (h2\ \#\ T2)) = set (map\ (Pair\ a)\ (h2\ \#\ T2))$
 $@\ pairs\ L1\ (h2\ \#\ T2))$
unfolding *pairs.simps* **by** *auto*
have *ih*: $set (pairs\ L1\ (h2\ \#\ T2)) = set (map\ (\lambda x. (x, h2))\ L1\ @\ pairs\ L1\ T2)$
using *Cons.hyps*[*of* $h2\ T2$] **by** *simp*

```

have *: set (pairs (a#L1) (h2#T2)) =
  set (map (Pair a) (h2 # T2)) ∪ set (map (λx. (x, h2)) L1 @ pairs L1 T2)
  using pairs-fact ih by auto
have **: set (pairs (a # L1) T2) = set (map (Pair a) T2 @ pairs L1 T2)
  using pairs.simps by simp
then show ?case using * ** by auto
qed

```

```

lemma list-concat-set-union:
  shows set(A@B) = set A ∪ set B
  by simp

```

```

lemma pairs-empty-list:
  shows pairs A [] = []
proof(induct A)
  case Nil
  then show ?case by simp
next
  case (Cons a A)
  then show ?case by auto
qed

```

7.1 Forward Direction Proofs

```

lemma pairs-member-fst-forward:
  assumes List.member (pairs A B) x
  shows List.member A (fst x)
  using assms
proof(induct A)
  case Nil
  then have pairs [] B = [] unfolding pairs.simps by simp
  then show ?case using member-rec(2)
    by (metis Nil)
next
  case (Cons a A)
  {assume fst-x-is-a: fst x = a
    then have ?case
      using Cons member-rec(1) by metis
  } moreover {
    assume fst-x-not-a: fst x ≠ a
    then have ¬(List.member (map (Pair a) B) x)
      using in-set-member by force
    then have List.member (pairs A B) x
      using Cons(2) unfolding pairs.simps List.member-def by auto
    then have ih: List.member A (fst x)
      using Cons.hyps by blast
    then have List.member (a # A) (fst x)
      unfolding List.member-def by simp
    then have ?case

```

```

    using ih by blast
  }
  ultimately show ?case by blast
qed

```

```

lemma pairs-member-snd-forward:
  assumes List.member (pairs A B) x
  shows List.member B (snd x)
  using assms

```

```

proof(induct B)
  case Nil
  have pairs A [] = []
    using pairs-empty-list by blast
  then show ?case
    by (metis local.Nil member-rec(2))
next
  case (Cons b B)
  have pairs-rec: set (pairs A (b # B)) = set (map (λx. (x, b)) A @ pairs A B)
    using pairs-alt[of A b B] by blast
  {assume snd-x-is-b: snd x = b
   then have ?case
     using Cons member-rec(1) by metis
  } moreover {
    assume snd-x-not-b: snd x ≠ b
    then have  $\neg(List.member (map (\lambda x. (x, b)) A) x)$ 
      using in-set-member pairs-rec by force
    then have List.member (pairs A B) x
      using Cons(2) unfolding pairs-rec List.member-def by simp
    then have ih: List.member B (snd x)
      using Cons.hyps by blast
    then have List.member (b # B) (snd x)
      unfolding List.member-def by simp
    then have ?case
      using ih by blast
  }
  ultimately show ?case by blast
qed

```

```

lemma pairs-member-forward:
  assumes List.member (pairs A B) x
  shows List.member A (fst x) ∧ List.member B (snd x)
  using assms pairs-member-fst-forward pairs-member-snd-forward by blast

```

```

lemma And-mltl-list-member-forward:
  assumes List.member (And-mltl-list D-x D-y) ψ
  shows  $\exists \psi1 \psi2. \psi = \text{And-mltl-ext } \psi1 \psi2$ 
   $\wedge List.member D-x \psi1 \wedge List.member D-y \psi2$ 
proof –
  obtain x where  $\psi = \text{And-mltl-ext } (fst\ x) (snd\ x) \wedge x \in \text{set } (pairs\ D-x\ D-y)$ 

```

```

    using assms unfolding And-mltl-list.simps List.member-def by auto
  then show ?thesis
    using pairs-member-forward[of D-x D-y x]
    by (simp add: in-set-member)
qed

```

7.2 Converse Direction Proofs

```

lemma pairs-member-converse:
  assumes List.member A (fst x)
  assumes List.member B (snd x)
  shows List.member (pairs A B) x
  using assms
proof(induct A)
  case Nil
  then show ?case unfolding List.member-def by simp
next
  case (Cons a A)
  {assume *: fst x = a
   then have ?case using Cons
     unfolding pairs.simps List.member-def by force
   } moreover {
   assume *: fst x ∈ set A
   then have List.member (pairs A B) x
     using Cons.hyps Cons(3) unfolding List.member-def by simp
   then have ?case unfolding pairs.simps List.member-def by simp
   }
  ultimately show ?case using Cons(2) unfolding List.member-def by force
qed

```

```

lemma And-mltl-list-member-converse:
  assumes  $\exists \psi_1 \psi_2. \psi = \text{And-mltl-ext } \psi_1 \psi_2$ 
  ∧ List.member D-x \psi_1 ∧ List.member D-y \psi_2
  shows List.member (And-mltl-list D-x D-y) \psi
proof–
  from assms obtain  $\psi_1 \psi_2$  where  $\psi = \text{And-mltl-ext } \psi_1 \psi_2 \wedge \text{List.member } D-x$ 
   $\psi_1 \wedge \text{List.member } D-y \psi_2$ 
  by blast
  then show ?thesis using pairs-member-converse
  unfolding And-mltl-list.simps List.member-def by force
qed

```

7.3 Biconditional Lemmas

```

lemma pairs-member:
  shows (List.member A (fst x) ∧ List.member B (snd x))  $\longleftrightarrow$ 
  List.member (pairs A B) x
  using pairs-member-forward pairs-member-converse by blast

```

lemma *And-mltl-list-member*:

shows $(\exists \psi 1 \ \psi 2. \ \psi = \text{And-mltl-ext } \psi 1 \ \psi 2$

$\wedge \text{List.member } D\text{-}x \ \psi 1 \ \wedge \text{List.member } D\text{-}y \ \psi 2) \longleftrightarrow$

$\text{List.member } (\text{And-mltl-list } D\text{-}x \ D\text{-}y) \ \psi$

using *And-mltl-list-member-forward And-mltl-list-member-converse* **by** *blast*

8 MLTL Decomposition Top Level Correctness

fun *wpd-mltl*:: 'a mltl \Rightarrow nat

where *wpd-mltl* *False_m* = 1

| *wpd-mltl* *True_m* = 1

| *wpd-mltl* (*Prop_m* (*p*)) = 1

| *wpd-mltl* (*Not_m* φ) = *wpd-mltl* φ

| *wpd-mltl* (φ *And_m* ψ) = *max* (*wpd-mltl* φ) (*wpd-mltl* ψ)

| *wpd-mltl* (φ *Or_m* ψ) = *max* (*wpd-mltl* φ) (*wpd-mltl* ψ)

| *wpd-mltl* (*G_m*[*a*,*b*] φ) = *b* + (*wpd-mltl* φ)

| *wpd-mltl* (*F_m*[*a*,*b*] φ) = *b* + (*wpd-mltl* φ)

| *wpd-mltl* (φ *R_m* [*a*,*b*] ψ) = *b* + (*max* ((*wpd-mltl* φ)) (*wpd-mltl* ψ))

| *wpd-mltl* (φ *U_m* [*a*,*b*] ψ) = *b* + (*max* ((*wpd-mltl* φ)) (*wpd-mltl* ψ))

8.1 Helper Lemmas

lemma *wpd-geq-one*:

shows *wpd-mltl* $\varphi \geq 1$

by (*induct* φ) *simp-all*

lemma *wpd-convert-nnf*:

fixes φ ::'a mltl

shows *wpd-mltl* (*convert-nnf* φ) = *wpd-mltl* φ

proof(*induction* *depth-mltl* φ *arbitrary*: φ *rule*: *less-induct*)

case *less*

have *not*: $(\bigwedge \varphi. \text{depth-mltl } \varphi < \text{Suc } (\text{depth-mltl } p) \implies$

$\text{wpd-mltl } (\text{convert-nnf } \varphi) = \text{wpd-mltl } \varphi) \implies$

$\varphi = \text{Not}_m p \implies$

$\text{wpd-mltl } (\text{convert-nnf } (\text{Not}_m p)) = \text{wpd-mltl } p$ **for** *p*

proof –

assume *ih*: $\bigwedge \varphi. \text{depth-mltl } \varphi < \text{Suc } (\text{depth-mltl } p) \implies$

$\text{wpd-mltl } (\text{convert-nnf } \varphi) = \text{wpd-mltl } \varphi$

assume *notcase*: $\varphi = \text{Not}_m p$

show *?thesis* **using** *ih notcase less* **by** (*induct* *p*) *simp-all*

qed

show *?case* **using** *less not* **by** (*cases* φ) *auto*

qed

lemma *convert-nnf-ext-preserves-wpd*:

shows *wpd-mltl* (*to-mltl* (*convert-nnf-ext* φ)) =

wpd-mltl (*to-mltl* φ)

proof(*induction* *depth-mltl* (*to-mltl* φ) *arbitrary*: φ *rule*: *less-induct*)

case *less*

have *not*: $(\bigwedge \varphi. \text{depth-mltl } (to\text{-mltl } \varphi)$
 $< \text{Suc } (\text{depth-mltl } (to\text{-mltl } x)) \implies$
 $\text{wpd-mltl } (to\text{-mltl } (\text{convert-nnf-ext } \varphi)) =$
 $\text{wpd-mltl } (to\text{-mltl } \varphi) \implies$
 $\varphi = \text{Not}_c x \implies$
 $\text{wpd-mltl } (to\text{-mltl } (\text{convert-nnf-ext } (\text{Not}_c x))) =$
 $\text{wpd-mltl } (to\text{-mltl } x)$ **for** x
proof –
assume *ih*: $(\bigwedge \varphi. \text{depth-mltl } (to\text{-mltl } \varphi)$
 $< \text{Suc } (\text{depth-mltl } (to\text{-mltl } x)) \implies$
 $\text{wpd-mltl } (to\text{-mltl } (\text{convert-nnf-ext } \varphi)) =$
 $\text{wpd-mltl } (to\text{-mltl } \varphi)$
assume *shape*: $\varphi = \text{Not}_c x$
show *?thesis* **using** *ih* *shape* **less by** $(\text{induct } x)$ *simp-all*
qed
show *?case* **using** *less not*
by $(\text{cases } \varphi)$ *auto*
qed

lemma *nnf-intervals-welldef*:
assumes *intervals-welldef F1*
shows *intervals-welldef* $(\text{convert-nnf } F1)$
using *assms*
proof $(\text{induct } \text{depth-mltl } F1)$ *arbitrary: F1 rule: less-induct*
case *less*
have *iwd*: *intervals-welldef F2* \implies
 $F1 = \text{Not}_m F2 \implies$
 $\text{intervals-welldef } (\text{convert-nnf } (\text{Not}_m F2))$
for $F2$ **using** *less* **by** $(\text{cases } F2)$ *simp-all*
then show *?case* **using** *less* **by** $(\text{cases } F1)$ *simp-all*
qed

lemma *is-composition-convert-nnf-ext*:
fixes $\varphi::'a \text{ mltl-ext}$
assumes *intervals-welldef* $(to\text{-mltl } \varphi)$
assumes *is-composition-MLTL* φ
shows *is-composition-MLTL* $(\text{convert-nnf-ext } \varphi)$
using *assms*
proof $(\text{induct } \text{depth-mltl } (to\text{-mltl } \varphi))$ *arbitrary: φ rule: less-induct*
case *less*
have *not-case*: $(\bigwedge \varphi. \text{depth-mltl } (to\text{-mltl } \varphi)$
 $< \text{Suc } (\text{depth-mltl } (to\text{-mltl } x_4)) \implies$
 $\text{intervals-welldef } (to\text{-mltl } \varphi) \implies$
 $\text{is-composition-MLTL } \varphi \implies$
 $\text{is-composition-MLTL } (\text{convert-nnf-ext } \varphi) \implies$
 $\text{intervals-welldef } (to\text{-mltl } x_4) \implies$
 $\text{is-composition-MLTL } x_4 \implies$
 $\varphi = \text{Not}_c x_4 \implies$

```

      is-composition-MLTL (convert-nnf-ext (Notc x4)) for x4
    using less by (induct x4) simp-all
    show ?case using less not-case by (cases φ) auto
  qed

```

```

lemma is-composition-allones-convert-nnf-ext:
  fixes φ: 'a mttl-ext
  assumes intervals-welldef (to-mltl φ)
  assumes is-composition-MLTL-allones φ
  shows is-composition-MLTL-allones (convert-nnf-ext φ)
  using assms
proof(induct depth-mltl (to-mltl φ) arbitrary: φ rule: less-induct)
  case less
  have not-case: (∧ φ. depth-mltl (to-mltl φ)
    < Suc (depth-mltl (to-mltl x4)) ⇒
    intervals-welldef (to-mltl φ) ⇒
    is-composition-MLTL-allones φ ⇒
    is-composition-MLTL-allones (convert-nnf-ext φ)) ⇒
    intervals-welldef (to-mltl x4) ⇒
    is-composition-MLTL-allones x4 ⇒
    φ = Notc x4 ⇒
    is-composition-MLTL-allones (convert-nnf-ext (Notc x4)) for x4
  using less by (induct x4) simp-all
  show ?case using less not-case
  by (cases φ) auto
qed

```

```

function Ands-mltl-ext:: 'a mttl-ext list ⇒ 'a mttl-ext
  where Ands-mltl-ext [] = True-mltl-ext
  | Ands-mltl-ext (H@[t]) = (if (length H = 0) then t
    else (And-mltl-ext (Ands-mltl-ext H) t))
  using rev-exhaust by auto
termination by (relation measure (λL. length L)) auto

```

```

lemma Ands-mltl-antics:
  assumes length X ≥ 1
  shows semantics-mltl-ext π (Ands-mltl-ext X) ↔
    (∀ x ∈ set X. semantics-mltl-ext π x)
  using assms
proof(induct length X-1 arbitrary: X)
  case 0
  then obtain x where X-is: X = [x]
  by (metis butlast-snoc diff-is-0-eq le-antisym length-0-conv length-butlast list.exhaust
    rotate1.simps(2) rotate1-length01 zero-neq-one)
  then show ?case unfolding X-is

```



```

    using Ands-mltl-ext.simps(2)[of [] x] by simp
next
case (Suc n)
then obtain H t where X-is:  $X = H@[t]$ 
  by (metis Ands-mltl-ext.cases One-nat-def Suc-n-not-le-n gen-length-code(1)
length-code)
then have length-H:  $\text{length } H = n+1$  using Suc by auto
then have cond1:  $n = \text{length } H - 1$  by simp
have cond2:  $\text{length } H \geq 1$  using length-H by simp
have semantics-H:  $\text{semantics-mltl-ext } \pi (\text{Ands-mltl-ext } H) =$ 
   $(\forall x. x \in \text{set } H \longrightarrow \text{semantics-mltl-ext } \pi x)$ 
  using Suc(1)[OF cond1 cond2] unfolding Ball-def by simp
have  $(\text{semantics-mltl-ext } \pi (\text{Ands-mltl-ext } H) \wedge$ 
   $\text{semantics-mltl-ext } \pi t) \longleftrightarrow$ 
   $(\forall x. x \in \text{set } (H @ [t]) \longrightarrow \text{semantics-mltl-ext } \pi x)$ 
  using semantics-H by auto
then have  $\text{semantics-mltl-ext } \pi (\text{And-mltl-ext } (\text{Ands-mltl-ext } H) t) =$ 
   $(\forall x. x \in \text{set } (H @ [t]) \longrightarrow \text{semantics-mltl-ext } \pi x)$ 
  unfolding semantics-mltl-ext-def to-mltl.simps by simp
then show ?case unfolding Ball-def X-is Ands-mltl-ext.simps
  using length-H by simp
qed

```

lemma *in-Global-mltl-decomp*:

```

  assumes length D-φ > 1
  assumes  $\psi \in \text{set } (\text{Global-mltl-decomp } D-\varphi \ a \ n \ L)$ 
  shows  $\exists X. ((\psi = \text{Ands-mltl-ext } X \wedge$ 
     $(\forall x. \text{List.member } X \ x \longrightarrow$ 
     $(\exists y \in \text{set } D-\varphi. (\exists k. a \leq k \wedge k \leq (a+n) \wedge x = \text{Global-mltl-ext } k \ k \ [1]$ 
     $y)))) \wedge$ 
     $(\text{length } X = \text{Suc } n))$ 
  using assms
proof (induct n arbitrary: D-φ ψ a)
  case 0
  then obtain x where x-in:  $x \in \text{set } D-\varphi$  and
    ψ-is:  $\psi = \text{Global-mltl-ext } a \ a \ [1] \ x$ 
    unfolding Global-mltl-decomp.simps Global-mltl-list.simps by auto
  then have  $\psi = \text{Ands-mltl-ext } [\text{Global-mltl-ext } a \ a \ [1] \ x]$ 
    using Ands-mltl-ext.simps(2)[of [] Global-mltl-ext a a [1] x] by auto
  then show ?case
    by (metis add.right-neutral length-Cons list.size(3) member-rec(1) member-rec(2)
order-refl x-in)
  next
  case (Suc x)
  then have  $\psi \in \text{set } (\text{And-mltl-list } (\text{Global-mltl-decomp } D-\varphi \ a \ x \ L)$ 
     $(\text{Global-mltl-list } D-\varphi \ (a + \text{Suc } x) \ (a + \text{Suc } x) \ [1]))$ 
    unfolding Global-mltl-decomp.simps by force
  then obtain first second where ψ-is:  $\psi = \text{And-mltl-ext } \text{first } \text{second}$ 
    and first-in:  $\text{first} \in \text{set } (\text{Global-mltl-decomp } D-\varphi \ a \ x \ L)$ 

```

and *second-in*: $second \in set (Global\text{-}mltl\text{-}list\ D\text{-}\varphi (a + Suc\ x) (a + Suc\ x)$
 $[1])$
using *And-mltl-list-member* **by** (*metis in-set-member*)
from *Suc.hyps*[*OF Suc.prem*s(1) *first-in*] **obtain** *X* **where**
X1: $first = And\text{-}mltl\text{-}ext\ X$ **and**
X2: $(\forall xa. List.member\ X\ xa \longrightarrow$
 $(\exists y \in set\ D\text{-}\varphi. \exists k \geq a. k \leq a + x \wedge xa = Global\text{-}mltl\text{-}ext\ k\ k\ [1]\ y))$ **and**
X3: $length\ X = (Suc\ x)$
by *blast*
from *second-in* **obtain** *x-second* **where**
second-is: $second = Global\text{-}mltl\text{-}ext\ (a + Suc\ x) (a + Suc\ x)\ [1]\ x\text{-}second$
and *x-second-in*: $x\text{-}second \in set\ D\text{-}\varphi$ **by** *auto*
have *prop1*: $\psi = And\text{-}mltl\text{-}ext\ (X@[second])$ **using** *ψ -is X1*
unfolding *And-mltl-ext.simps* **using** *X3* **by** *auto*
have *prop2*: $(\exists y \in set\ D\text{-}\varphi. \exists k \geq a. k \leq a + Suc\ x \wedge xa = Global\text{-}mltl\text{-}ext\ k\ k\ [1]$
 $y)$
if *prem*: $List.member\ (X@[second])\ xa$ **for** *xa*
using *X2 second-is*
proof–
have *split*: $(List.member\ X\ xa) \vee xa = second$
using *prem*
by (*metis in-set-member member-rec*(1) *rotate1.simps*(2) *set-rotate1*)
{assume *in-X*: $List.member\ X\ xa$
have *?thesis* **using** *X2 in-X* **by** *force*
} **moreover** **{**
assume *in-second*: $xa = second$
have *?thesis* **using** *in-second second-is*
by (*simp add: x-second-in*)
}
ultimately show *?thesis* **using** *split* **by** *blast*
qed
have *prop3*: $length\ (X@[second]) = Suc\ (Suc\ x)$
using *X3* **by** *simp*
then show *?case*
using *prop1 prop2 prop3* **by** *blast*
qed

lemma *in-Global-mltl-decomp-exact-forward*:

assumes $length\ D\text{-}\varphi > 1$
assumes $\psi \in set (Global\text{-}mltl\text{-}decomp\ D\text{-}\varphi\ a\ n\ L)$
shows $\exists X. ((\psi = And\text{-}mltl\text{-}ext\ X \wedge$
 $(\forall i < length\ X. (\exists y \in set\ D\text{-}\varphi. (X!i) = Global\text{-}mltl\text{-}ext\ (a+i)\ (a+i)\ [1]$
 $y)))) \wedge$
 $(length\ X = Suc\ n)$
using *assms*
proof(*induct n arbitrary: D- φ ψ a*)
case 0
then obtain *x* **where** *x-in*: $x \in set\ D\text{-}\varphi$ **and**

```

       $\psi$ -is:  $\psi = \text{Global-mltl-ext } a \ a \ [1] \ x$ 
    unfolding Global-mltl-decomp.simps Global-mltl-list.simps by auto
  then have  $\psi = \text{And-s-mltl-ext } [\text{Global-mltl-ext } a \ a \ [1] \ x]$ 
    using And-s-mltl-ext.simps(2)[of [] Global-mltl-ext a a [1] x] by auto
  then show ?case
    using x-in by auto
next
case (Suc n)
obtain H t where  $\psi$ -is:  $\psi = \text{And-mltl-ext } H \ t$ 
  and H-in:  $H \in \text{set } (\text{Global-mltl-decomp } D\text{-}\varphi \ a \ n \ L)$ 
  and t-in:  $t \in \text{set } (\text{Global-mltl-list } D\text{-}\varphi \ (a + \text{Suc } n) \ (a + \text{Suc } n) \ [1])$ 
  using Suc(3) unfolding Global-mltl-decomp.simps
  using And-mltl-list-member unfolding List.member-def
  by (metis add-diff-cancel-left' plus-1-eq-Suc)
obtain x where t-is:  $t = \text{Global-mltl-ext } (a + \text{Suc } n) \ (a + \text{Suc } n) \ [1] \ x$ 
  and x-in:  $x \in \text{set } D\text{-}\varphi$ 
  using t-in unfolding Global-mltl-list.simps by auto
have  $\exists X. (H = \text{And-s-mltl-ext } X \wedge$ 
   $(\forall i < \text{length } X. \exists y \in \text{set } D\text{-}\varphi. X \ ! \ i = \text{Global-mltl-ext } (a + i) \ (a + i) \ [1] \ y)) \wedge$ 
   $\text{length } X = \text{Suc } n$ 
  using Suc.hyps[of D-φ H a] Suc.prem1 H-in by blast
then obtain X where H-is:  $H = \text{And-s-mltl-ext } X$ 
  and X-prop:  $\forall i < \text{length } X. \exists y \in \text{set } D\text{-}\varphi. X \ ! \ i = \text{Global-mltl-ext } (a$ 
   $+ i) \ (a + i) \ [1] \ y$ 
  and length-X:  $\text{length } X = \text{Suc } n$ 
  by blast
have  $\psi$ -is:  $\psi = \text{And-s-mltl-ext } (X@[t])$ 
  unfolding And-s-mltl-ext.simps using length-X  $\psi$ -is
  by (simp add: H-is)
have property:  $\exists y \in \text{set } D\text{-}\varphi. (X@[t]) \ ! \ i = \text{Global-mltl-ext } (a + i) \ (a + i) \ [1] \ y$ 
  if i-bound:  $i < \text{length } (X@[t])$  for i
proof-
{
  assume *:  $i < \text{length } X$ 
  then have  $X \ ! \ i = (X@[t]) \ ! \ i$  using length-X
    by (simp add: nth-append)
  then have ?thesis using X-prop length-X * by metis
} moreover {
  assume *:  $i = \text{length } X$ 
  have  $(X@[t]) \ ! \ i = t$ 
    using length-X *
    by (metis nth-append-length)
  then have ?thesis using t-is * length-X
    by (simp add: x-in)
}
ultimately show ?thesis using i-bound by fastforce
qed
have len:  $\text{length } (X@[t]) = \text{Suc } (\text{Suc } n)$ 
  using length-X by auto

```

```

then show ?case
  using  $\psi$ -is property len by blast
qed

lemma in-Global-mltl-decomp-exact-converse:
  fixes  $n::\text{nat}$  and  $X::'a$  mltl-ext list
  assumes length  $D-\varphi > 1$ 
  assumes  $\psi = \text{And-s-mltl-ext } X$ 
  assumes  $(\forall i < \text{length } X. (\exists y \in \text{set } D-\varphi. (X!i) = \text{Global-mltl-ext } (a+i) (a+i) [1] y))$ 
  assumes length  $X = n+1$ 
  shows  $\psi \in \text{set } (\text{Global-mltl-decomp } D-\varphi a n L)$ 
  using assms
proof(induct n arbitrary: X  $\psi$  a)
  case 0
  then have length- $X$ : length  $X = 1$  by auto
  then have  $\exists x. X = [x]$ 
  by (metis Suc-eq-plus1 add-cancel-right-left length-Cons list.size(3) neq-Nil-conv zero-eq-add-iff-both-eq-0 zero-neq-one)
  then obtain  $x$  where  $X$ -is:  $X = [x]$  by blast
  then obtain  $y$  where  $x$ -is:  $x = \text{Global-mltl-ext } a a [1] y$ 
  and  $y$ -in:  $y \in \text{set } D-\varphi$ 
  using 0 by auto
  then show ?case unfolding 0(2)  $X$ -is
  using And-s-mltl-ext.simps(2)[of [] x] by simp
next
  case (Suc n)
  then have length- $X$ : length  $X = n+2$  by simp
  then obtain  $H t$  where  $X$ -is:  $X = H@[t]$ 
  by (metis Suc.prem(4) Suc-eq-plus1 length-Suc-conv-rev)
  have length- $H$ : length  $H = n+1$  using length- $X$   $X$ -is by auto
  have  $\psi$ -is:  $\psi = \text{And-mltl-ext } (\text{And-s-mltl-ext } H) t$ 
  using Suc(3) unfolding X-is And-s-mltl-ext.simps
  using length- $H$  by simp
  have  $H$ -prop:  $\exists y \in \text{set } D-\varphi. H ! i = \text{Global-mltl-ext } (a + i) (a + i) [1] y$ 
  if  $i$ -bound:  $i < \text{length } H$  for  $i$ 
  proof–
  have index:  $(H @ [t]) ! i = H ! i$ 
  using  $i$ -bound by (simp add: nth-append)
  then have  $\exists y \in \text{set } D-\varphi. (H @ [t]) ! i = \text{Global-mltl-ext } (a + i) (a + i) [1] y$ 
  using  $i$ -bound Suc(4) unfolding X-is
  by (metis Suc.prem(4) Suc-eq-plus1 X-is length-H plus-1-eq-Suc trans-less-add2)

  then show ?thesis
  using index by auto
qed
then have  $H$ -prop:  $\forall i < \text{length } H. \exists y \in \text{set } D-\varphi. H ! i = \text{Global-mltl-ext } (a + i) (a + i) [1] y$ 
by blast

```

```

have H-in: Ands-mltl-ext  $H \in \text{set } (\text{Global-mltl-decomp } D\text{-}\varphi \text{ a n } L)$ 
  using Suc(1)[OF Suc(2) - H-prop, of (Ands-mltl-ext H)]
  using length-H by blast
have t-is:  $\exists y \in \text{set } D\text{-}\varphi. t = \text{Global-mltl-ext } (a + n + 1) (a + n + 1) [1] y$ 
  using Suc(4) unfolding X-is using length-X
  by (metis X-is add.assoc length-H less-add-one nth-append-length one-add-one)
then obtain y where t-is:  $t = \text{Global-mltl-ext } (a + n + 1) (a + n + 1) [1] y$ 
  and y-in:  $y \in \text{set } D\text{-}\varphi$ 
  by blast
have t-in:  $t \in \text{set } (\text{Global-mltl-list } D\text{-}\varphi (a + \text{Suc } n) (a + \text{Suc } n) [1])$ 
  using y-in t-is by simp
show ?case unfolding  $\psi$ -is Global-mltl-decomp.simps
  using t-in H-in And-mltl-list-member[of  $\psi$  (Global-mltl-decomp } D\text{-}\varphi \text{ a n) } L
(Global-mltl-list } D\text{-}\varphi (a + \text{Suc } n) (a + \text{Suc } n) [1])]
  unfolding List.member-def  $\psi$ -is by auto
qed

```

```

lemma case-split-helper:
  assumes  $x \in A \cup B \cup C$ 
  assumes  $x \in A \implies P x$  and  $x \in B \implies P x$  and  $x \in C \implies P x$ 
  shows  $P x$ 
  using assms by blast

```

```

lemma LP-mltl-aux-intervals-welldef:
  fixes  $\varphi \psi :: 'a \text{ mltl-ext}$ 
  assumes intervals-welldef (to-mltl } \varphi)
  assumes  $\psi \in \text{set } (\text{LP-mltl-aux } (\text{convert-nnf-ext } \varphi) k)$ 
  assumes is-composition-MLTL } \varphi
  shows intervals-welldef (to-mltl } \psi)
  using assms
proof(induct k arbitrary: } \varphi } \psi)
  case 0
  then show ?case unfolding LP-mltl-aux.simps
  by (simp add: convert-nnf-and-convert-nnf-ext nnf-intervals-welldef)
next
  case (Suc k)
  then show ?case
  proof(cases convert-nnf-ext } \varphi)
  case True-mltl-ext
  then show ?thesis using Suc by simp
  next
  case False-mltl-ext
  then show ?thesis using Suc by simp
  next
  case (Prop-mltl-ext p)
  then show ?thesis using Suc by simp
  next
  case (Not-mltl-ext q)
  then have  $\exists p. q = \text{Prop-mltl-ext } p$ 

```

```

using convert-nnf-form-Not-Implies-Prop
by (metis convert-nnf-ext-to-mltl-commute to-mltl.simps(4) to-mltl-prop-bijective)

then obtain  $p$  where  $q = \text{Prop-mltl-ext } p$  by auto
then show ?thesis using Suc
  by (simp add: Not-mltl-ext)
next
case (And-mltl-ext  $\alpha$   $\beta$ )
obtain  $x$   $y$  where  $\psi$ -is:  $\psi = \text{And-mltl-ext } x$   $y$ 
  and  $x$ -in:  $x \in \text{set } (LP\text{-mltl-aux } (\text{convert-nnf-ext } \alpha) k)$ 
  and  $y$ -in:  $y \in \text{set } (LP\text{-mltl-aux } (\text{convert-nnf-ext } \beta) k)$ 
  using Suc(3) unfolding And-mltl-ext LP-mltl-aux.simps
  by (meson And-mltl-list-member in-set-member)
then show ?thesis unfolding  $\psi$ -is to-mltl.simps intervals-welldef.simps
  using Suc.hyps x-in y-in
  by (metis And-mltl-ext Suc.prem(1) Suc.prem(3) convert-nnf-ext-to-mltl-commute
intervals-welldef.simps(5) nnf-intervals-welldef is-composition-MLTL.simps(1) is-composition-convert-nnf-ext
to-mltl.simps(5))
next
case (Or-mltl-ext  $\alpha$   $\beta$ )
let  $?Dx = LP\text{-mltl-aux } (\text{convert-nnf-ext } \alpha) k$ 
let  $?Dy = LP\text{-mltl-aux } (\text{convert-nnf-ext } \beta) k$ 
{assume  $*$ :  $\psi \in \text{set } (And\text{-mltl-list } ?Dx ?Dy)$ 
  then obtain  $x$   $y$  where  $\psi$ -is:  $\psi = \text{And-mltl-ext } x$   $y$ 
  and  $x$ -in:  $x \in \text{set } ?Dx$  and  $y$ -in:  $y \in \text{set } ?Dy$ 
  using Suc(3) LP-mltl-aux.simps
  by (meson And-mltl-list-member in-set-member)
then have ?thesis unfolding Or-mltl-ext
  by (metis Or-mltl-ext Suc.hyps Suc.prem(1) Suc.prem(3) convert-nnf-ext-to-mltl-commute
intervals-welldef.simps(5) intervals-welldef.simps(6) nnf-intervals-welldef is-composition-MLTL.simps(2)
is-composition-convert-nnf-ext to-mltl.simps(5) to-mltl.simps(6))
} moreover {
  assume  $*$ :  $\psi \in \text{set } (And\text{-mltl-list } [Not_c \alpha] ?Dy)$ 
  then obtain  $y$  where  $\psi$ -is:  $\psi = \text{And-mltl-ext } (Not_c \alpha) y$ 
  and  $y$ -in:  $y \in \text{set } ?Dy$ 
  using Suc(3)
  using And-mltl-list-member[of  $\psi$   $?Dy$   $[Not_c \alpha]$  by auto
  have lhs-welldef: intervals-welldef (to-mltl  $\alpha$ )
  by (metis Or-mltl-ext Suc.prem(1) convert-nnf-ext-to-mltl-commute inter-
vals-welldef.simps(6) nnf-intervals-welldef to-mltl.simps(6))
  have rhs-welldef: intervals-welldef (to-mltl  $y$ )
  using  $y$ -in Suc.prem unfolding Or-mltl-ext
  by (metis Or-mltl-ext Suc.hyps convert-nnf-ext-to-mltl-commute inter-
vals-welldef.simps(6) nnf-intervals-welldef is-composition-MLTL.simps(2) is-composition-convert-nnf-ext
to-mltl.simps(6))
  then have ?thesis
  unfolding  $\psi$ -is to-mltl.simps intervals-welldef.simps
  using lhs-welldef rhs-welldef by blast
} moreover {

```

```

assume *:  $\psi \in \text{set } (\text{And-mltl-list } ?Dx \text{ [Not}_c \beta])$ 
then obtain  $x$  where  $\psi$ -is:  $\psi = \text{And-mltl-ext } x \text{ (Not}_c \beta)$ 
  and  $x$ -in:  $x \in \text{set } ?Dx$ 
  using  $\text{Suc}(3)$   $\text{And-mltl-list-member}$ [of  $\psi$   $?Dx$   $[\text{Not}_c \beta]$ ]
  by ( $\text{metis in-set-member member-rec(1) member-rec(2)}$ )
have  $\text{lhs-welldef}$ :  $\text{intervals-welldef } (\text{to-mltl } \beta)$ 
  by ( $\text{metis Or-mltl-ext Suc.prem(1) convert-nnf-ext-to-mltl-commute intervals-welldef.simps(6) nnf-intervals-welldef to-mltl.simps(6)}$ )
have  $\text{rhs-welldef}$ :  $\text{intervals-welldef } (\text{to-mltl } x)$ 
  using  $x$ -in  $\text{Suc.prem}$  unfolding  $\text{Or-mltl-ext}$ 
  by ( $\text{metis Or-mltl-ext Suc.hyps convert-nnf-ext-to-mltl-commute intervals-welldef.simps(6) nnf-intervals-welldef is-composition-MLTL.simps(2) is-composition-convert-nnf-ext to-mltl.simps(6)}$ )
  then have  $?thesis$ 
    unfolding  $\psi$ -is  $\text{to-mltl.simps intervals-welldef.simps}$ 
    using  $\text{lhs-welldef rhs-welldef}$  by  $\text{blast}$ 
  }
ultimately show  $?thesis$ 
  using  $\text{Suc}(3)$  unfolding  $\text{Or-mltl-ext LP-mltl-aux.simps}$ 
  using  $\text{list-concat-set-union}$ 
  by ( $\text{metis UnE}$ )
next
case ( $\text{Future-mltl-ext } a \ b \ L \ \alpha$ )
let  $?D = \text{LP-mltl-aux } (\text{convert-nnf-ext } \alpha) \ k$ 
let  $?s = \text{interval-times } a \ L$ 
have  $\text{convert-nnf } (\text{to-mltl } \varphi) = \text{Future-mltl } a \ b \ (\text{to-mltl } \alpha)$ 
  using  $\text{Future-mltl-ext convert-nnf-and-convert-nnf-ext}$ 
  by ( $\text{simp add: convert-nnf-ext-to-mltl-commute}$ )
then have  $a$ -leq- $b$ :  $a \leq b$ 
  using  $\text{Suc } (2)$   $\text{Future-mltl-ext nnf-intervals-welldef}$ 
  by  $\text{fastforce}$ 
from  $\text{is-composition-convert-nnf-ext}[OF \ \text{Suc}(2) \ \text{Suc}(4)]$ 
  have  $\text{is-composition-MLTL } (\text{convert-nnf-ext } \varphi)$ 
  .
  then have  $\text{is-comp}$ :  $\text{is-composition } (b-a+1) \ L$ 
  unfolding  $\text{Future-mltl-ext is-composition-MLTL.simps}$  by  $\text{blast}$ 
{assume *:  $\psi \in \text{set } (\text{Future-mltl-list } ?D \ (?s ! 0) \ (?s ! 1 - 1) \ [?s ! 1 - ?s ! 0])$ 
  then obtain  $x$  where  $\psi$ -is:  $\psi = \text{Future-mltl-ext } (?s ! 0) \ (?s ! 1 - 1) \ [?s ! 1 - ?s ! 0] \ x$ 
  and  $x$ -in:  $x \in \text{set } ?D$ 
  unfolding  $\text{Future-mltl-list.simps}$  by  $\text{fastforce}$ 
from  $\text{is-comp}$  have  $\text{welldef}$ :  $?s ! 0 \leq ?s ! 1 - 1$ 
  using  $\text{interval-times-diff-ge}[OF \ a$ -leq- $b \ \text{is-comp } - \ , \ \text{of } 0 \ ?s]$ 
  by ( $\text{metis } a$ -leq- $b \ \text{add-0} \ \text{add-le-imp-le-diff} \ \text{gr-zeroI} \ \text{interval-times-first interval-times-last less-iff-succ-less-eq} \ \text{order-less-irrefl}$ )
  have  $ih$ :  $\text{intervals-welldef } (\text{to-mltl } x)$ 
  using  $\text{Suc } x$ -in
  by ( $\text{metis Future-mltl-ext convert-nnf-ext-to-mltl-commute intervals-welldef.simps}(7)$ )

```

nnf-intervals-welldef is-composition-MLTL.simps(5) is-composition-convert-nnf-ext to-mltl.simps(7)

```

then have ?thesis
  unfolding  $\psi$ -is to-mltl.simps intervals-welldef.simps
  using welldef ih by blast
} moreover {
assume *:  $\psi \in \text{set } (\text{concat } (\text{map } (\lambda i. \text{And-mltl-list }
  [\text{Global-mltl-ext } (?s ! 0)
  (?s ! i - 1) [?s!i-?s!0] (\text{Not}_c \alpha)]
  (\text{Future-mltl-list } ?D (?s ! i) (?s ! (i + 1) - 1)
  [?s ! (i + 1) - ?s ! i]))
  [1..<\text{length } L]))$ 
then obtain  $i$  where  $\psi$ -is:  $\psi \in \text{set } ((\text{And-mltl-list }
  [\text{Global-mltl-ext } (?s ! 0)
  (?s ! i - 1) [?s!i-?s!0] (\text{Not}_c \alpha)]
  (\text{Future-mltl-list } ?D (?s ! i) (?s ! (i + 1) - 1)
  [?s ! (i + 1) - ?s ! i])
  ))$ 
and  $i$ -in:  $i \in \{1..<\text{length } L\}$ 
by force
then obtain  $x$  where  $\psi$ -is:  $\psi = ((\text{And-mltl-ext }
  (\text{Global-mltl-ext } (?s ! 0)
  (?s ! i - 1) [?s!i-?s!0] (\text{Not}_c \alpha))
  (\text{Future-mltl-ext } (?s ! i) (?s ! (i + 1) - 1)
  [?s ! (i + 1) - ?s ! i] x)))$ 
and  $x$ -in:  $x \in \text{set } ?D$ 
by auto
from is-comp have welldef1:  $\text{interval-times } a L ! 0 \leq \text{interval-times } a L ! i$ 
- 1
using  $i$ -in
using interval-times-diff-ge-general[OF a-leq-b is-comp -, of  $i 0 ?s$ ]
by force
have welldef2:  $\text{interval-times } a L ! i \leq \text{interval-times } a L ! (i + 1) - 1$ 
using  $i$ -in
by (metis a-leq-b add commute add-le-imp-le-diff atLeastLessThan-iff inter-
val-times-diff-ge is-comp less-eq-Suc-le plus-1-eq-Suc)

have ih1: intervals-welldef (to-mltl  $\alpha$ )
using Suc  $x$ -in
by (metis <convert-nnf (to-mltl  $\varphi$ ) = Future-mltl a b (to-mltl  $\alpha$ )> inter-
vals-welldef.simps(7) nnf-intervals-welldef)
have ih2: intervals-welldef (to-mltl  $x$ )
using Suc
by (metis Future-mltl-ext <is-composition-MLTL (convert-nnf-ext  $\varphi$ )> ih1
is-composition-MLTL.simps(5)  $x$ -in)
have ?thesis unfolding  $\psi$ -is to-mltl.simps intervals-welldef.simps
using ih1 ih2 welldef1 welldef2
by auto
}

```



```

ultimately show ?thesis
  using Suc(3) unfolding Future-mltl-ext LP-mltl-aux.simps
  using list-concat-set-union
  by (metis (no-types, lifting) Un-iff)
next
case (Global-mltl-ext a b L α)
let ?D-φ = LP-mltl-aux (convert-nnf-ext α) k
have nnf-φ: convert-nnf (to-mltl φ) = Global-mltl a b (to-mltl α)
  using Global-mltl-ext convert-nnf-and-convert-nnf-ext
  by (simp add: convert-nnf-ext-to-mltl-commute)
then have a-leq-b: a ≤ b
  using Suc (2) Global-mltl-ext nnf-intervals-welldef
  by fastforce
have α-composition: is-composition-MLTL α
  using Suc(4) Global-mltl-ext Suc.premis(1) is-composition-convert-nnf-ext by
fastforce
have L-composition: is-composition (b-a+1) L
  by (metis Global-mltl-ext Suc.premis(1) Suc.premis(3) is-composition-MLTL.simps(3)
is-composition-convert-nnf-ext)
{assume *: length ?D-φ ≤ 1
  then have ψ: ψ = Global-mltl-ext a b L α
    using Suc(3)
    unfolding Global-mltl-ext LP-mltl-aux.simps
    by simp
  have ih1: intervals-welldef (to-mltl α)
    using Suc nnf-φ
    by (metis intervals-welldef.simps(8) nnf-intervals-welldef)
  then have ?thesis
    using a-leq-b unfolding ψ to-mltl.simps
    intervals-welldef.simps by auto
} moreover {assume *: length ?D-φ > 1
  then have ψ-in: ψ ∈ set (Global-mltl-decomp ?D-φ a (b - a) L)
    using Suc(3)
    unfolding Global-mltl-ext LP-mltl-aux.simps
    by simp
  then obtain X where ψ-is: ψ = Ands-mltl-ext X and
    X-fact: (∀ x ∈ set X.
      (∃ y ∈ set (LP-mltl-aux (convert-nnf-ext α) k).
        ∃ k ≥ a. k ≤ a + (b - a) ∧ x = Global-mltl-ext k k [1] y))
    and length-X: length X = Suc (b - a)
    using in-Global-mltl-decomp[OF * ψ-in]
    unfolding List.member-def by blast
  have X-ih: intervals-welldef (to-mltl x)
    if x-in: x ∈ set X for x
  proof-
  obtain y k where y-in: y ∈ set ?D-φ
    and k-bound: a ≤ k ∧ k ≤ b
    and x-is: x = Global-mltl-ext k k [1] y
    using X-fact a-leq-b x-in by fastforce

```

```

    show ?thesis using y-in Suc
      unfolding x-is to-mltl.simps intervals-welldef.simps
    by (metis Global-mltl-ext intervals-welldef.simps(8) is-composition-MLTL.simps(3)
is-composition-convert-nnf-ext nnf-φ nnf-intervals-welldef order-refl)
  qed
  have ?thesis
    using ψ-is X-ih length-X
  proof(induct b-a arbitrary: b a ψ X)
    case 0
    then obtain x where X-is: X = [x]
      by (metis length-0-conv length-Suc-conv)
    have ψ = x
      using Ands-mltl-ext.simps(2) 0
      by (metis X-is append-self-conv2 length-0-conv)
    then show ?case using 0(3)[of x] unfolding X-is by auto
  next
    case (Suc n)
    then have length X = n + 2 by linarith
    then obtain H t where X-is: X = H@[t] and length-H: length H = length
X-1
      by (metis Suc.prem(3) diff-Suc-1 length-Suc-conv-rev)
    have ψ-is: ψ = And-mltl-ext (Ands-mltl-ext H) t
      using Suc(3) unfolding X-is Ands-mltl-ext.simps using length-H
    by (metis One-nat-def Suc.hyps(2) Suc.prem(3) diff-Suc-1' nat.distinct(1))

    have t-ih: intervals-welldef (to-mltl t)
      using X-is Suc by force
    have (∧x. x ∈ set H ⇒ intervals-welldef (to-mltl x))
      using Suc.prem unfolding X-is by auto
    then have H-ih: intervals-welldef (to-mltl (Ands-mltl-ext H))
      using Suc.hyps(1)[of - - Ands-mltl-ext H H]
      by (metis Suc.hyps(2) Suc.prem(3) diff-Suc-1 length-H)
    show ?case unfolding ψ-is to-mltl.simps
      using t-ih H-ih by simp
  qed
}
ultimately show ?thesis
  by linarith
next
case (Until-mltl-ext α a b L β)
let ?D-β = LP-mltl-aux (convert-nnf-ext β) k
let ?s = interval-times a L
have a-leq-b: a ≤ b using Suc(2)
  by (metis Until-mltl-ext convert-nnf-ext-to-mltl-commute intervals-welldef.simps(9)
to-mltl.simps(9) nnf-intervals-welldef)
have composition: is-composition-MLTL (Until-mltl-ext α a b L β)
  using Suc(4) Until-mltl-ext
  by (metis Suc.prem(1) is-composition-convert-nnf-ext)
have interval-composition: is-composition (b - a + 1) L

```

```

    using composition by simp
  have length-L: 0 < length L
    using interval-composition
    by (meson add-gr-0 composition-length-lb less-numeral-extra(1))
  have  $\alpha$ -ih: intervals-welldef (to-mltl  $\alpha$ )
    using Suc Until-mltl-ext convert-nnf-ext-to-mltl-commute
    by (metis intervals-welldef.simps(9) to-mltl.simps(9) nnf-intervals-welldef)
  have  $\beta$ -ih: intervals-welldef (to-mltl  $\beta$ )
    using Suc(2) Until-mltl-ext
    by (metis convert-nnf-ext-to-mltl-commute intervals-welldef.simps(9) to-mltl.simps(9)
    nnf-intervals-welldef)
  {assume *:  $\psi \in \text{set } (\text{Until-mltl-list } \alpha \text{ ?D-}\beta \text{ (?s! 0) (?s! 1 - 1) [?s! 1 - ?s! 0])$ }
  then obtain x where  $\psi$ -is:  $\psi = \text{Until-mltl-ext } \alpha \text{ (?s!0) (?s!1-1) [?s!1-?s!0]$ 
  x
    and x-in:  $x \in \text{set } (?D\text{-}\beta)$ 
    by auto
  have fact1: interval-times a L ! 0  $\leq$  interval-times a L ! 1 - 1
    unfolding is-composition-def
    using interval-times-diff-ge[OF a-leq-b interval-composition length-L, of ?s]
    by auto
  have x-ih: intervals-welldef (to-mltl x)
    using x-in Suc.hyps[of  $\beta$  x] Suc.premis
    using  $\beta$ -ih composition is-composition-MLTL.simps(6) by blast
  have ?thesis unfolding  $\psi$ -is unfolding to-mltl.simps
    unfolding intervals-welldef.simps
    using fact1  $\alpha$ -ih x-ih by blast
} moreover {
  assume *:  $\psi \in \text{set } (\text{concat } (\text{map } (\lambda i. \text{And-mltl-list } [\text{Global-mltl-ext } (?s! 0) (?s! i - 1) [?s!i - ?s!0] (\text{And-mltl-ext } \alpha (\text{Not}_c \beta))])$ 
  ( $\text{Until-mltl-list } \alpha \text{ ?D-}\beta \text{ (?s! i) (?s! (i + 1) - 1) [?s! (i + 1) - ?s! i]$ ))
  [1.. $\text{length } L$ ]))
  then obtain i x where
 $\psi$ -is:  $\psi = \text{And-mltl-ext } (\text{Global-mltl-ext } (?s!0) (?s!i-1) [?s!i - ?s!0] (\text{And-mltl-ext } \alpha (\text{Not}_c \beta)))$ 
  ( $\text{Until-mltl-ext } \alpha \text{ (?s!i) (?s!(i+1)-1) [(?s!(i+1)) - (?s!i)] x$ )
  and i-bound:  $1 \leq i \wedge i < \text{length } L$ 
  and x-in:  $x \in \text{set } ?D\text{-}\beta$ 
  by auto
  have fact1: interval-times a L ! 0  $\leq$  interval-times a L ! i - 1
    using i-bound a-leq-b
    using interval-times-diff-ge-general[OF a-leq-b interval-composition, of i 0
  ?s]
  by force
  have fact2: interval-times a L ! i  $\leq$  interval-times a L ! (i + 1) - 1

```

```

    using i-bound
    using interval-times-diff-ge[OF a-leq-b interval-composition, of i ?s]
    by auto
  have x-ih: intervals-welldef (to-mltl x)
    using Suc.hyps β-ih composition is-composition-MLTL.simps(6) x-in by
blast
  have ?thesis unfolding ψ-is to-mltl.simps
    unfolding intervals-welldef.simps
    using fact1 fact2 α-ih β-ih x-ih by blast
}
ultimately show ?thesis using Suc(3) list-concat-set-union
  unfolding Until-mltl-ext LP-mltl-aux.simps
  by (metis (mono-tags, lifting) UnE)
next
case (Release-mltl-ext α a b L β)
let ?D = LP-mltl-aux (convert-nnf-ext α) k
let ?s = interval-times a L
have α-ih: intervals-welldef (to-mltl α)
  using Suc(2) Release-mltl-ext convert-nnf-ext-to-mltl-commute
  by (metis intervals-welldef.simps(10) to-mltl.simps(10) nnf-intervals-welldef)
have β-ih: intervals-welldef (to-mltl β)
  using Suc(2) Release-mltl-ext convert-nnf-ext-to-mltl-commute
  by (metis intervals-welldef.simps(10) to-mltl.simps(10) nnf-intervals-welldef)
have a-leq-b: a ≤ b using Suc(2) Release-mltl-ext
by (metis convert-nnf-ext-to-mltl-commute intervals-welldef.simps(10) to-mltl.simps(10)
nnf-intervals-welldef)
have composition: is-composition-MLTL (Release-mltl-ext α a b L β)
  using Suc.prem(3) Release-mltl-ext
  by (metis Suc.prem(1) is-composition-convert-nnf-ext)
then have composition-L: is-composition (b-a+1) L
  and composition-α: is-composition-MLTL α
  and composition-β: is-composition-MLTL β
  unfolding is-composition-MLTL.simps by simp-all
have length-L: length L > 0
  using composition-length-lb composition-L by auto
have sfirst: ?s!0 = a
  using interval-times-first by simp
have slast: ?s!(length L) = b+1
  using interval-times-last[OF a-leq-b composition-L] by blast
let ?front = set [Global-mltl-ext a b L (And-mltl-ext (Notc α) β)]
let ?middle = set (Mighty-Release-mltl-list ?D β (?s!0) (?s!1 - 1)
  [?s!1 - ?s!0])
let ?back = set (concat (map (λi. And-mltl-list
  [Global-mltl-ext
    (?s!0)
    (?s!i - 1) [?s!i - ?s!0] (And-mltl-ext (Notc α) β)]
  (Mighty-Release-mltl-list ?D β (?s!i)
    (?s!(i + 1) - 1) [?s!(i + 1) - ?s!i]))
  [1..length L]))

```

```

have split:  $\psi \in ?front \cup ?middle \cup ?back$ 
  using Suc( $\beta$ ) unfolding Release-mltl-ext LP-mltl-aux.simps
  using list-concat-set-union
  by (metis append.assoc)
{
  assume *:  $\psi \in ?front$ 
  then have  $\psi$ -is:  $\psi = \text{Global-mltl-ext } a \ b \ L \ (\text{And-mltl-ext } (\text{Not}_c \ \alpha) \ \beta)$ 
    by auto
  have ?thesis unfolding  $\psi$ -is to-mltl.simps intervals-welldef.simps
    using  $\alpha$ -ih  $\beta$ -ih a-leq-b by blast
} moreover {
  assume *:  $\psi \in ?middle$ 
  then obtain  $x$  where  $\psi$ -is:  $\psi = \text{Mighty-Release-mltl-ext } x \ \beta$ 
    (interval-times  $a \ L \ ! \ 0$ ) (interval-times  $a \ L \ ! \ 1 - 1$ )
    [interval-times  $a \ L \ ! \ 1 - \text{interval-times } a \ L \ ! \ 0$ ]
    and  $x$ -in:  $x \in \text{set } ?D$ 
  by auto
  have  $x$ -ih: intervals-welldef (to-mltl  $x$ )
    using Suc(1)[OF  $\alpha$ -ih  $x$ -in composition- $\alpha$ ] by blast
  have welldef: interval-times  $a \ L \ ! \ 0 \leq \text{interval-times } a \ L \ ! \ 1 - 1$ 
    using interval-times-diff-ge[OF a-leq-b composition-L, of 0 ?s]
    using length-L by auto
  then have ?thesis unfolding  $\psi$ -is to-mltl.simps Mighty-Release-mltl-ext.simps
    intervals-welldef.simps
    using  $x$ -ih  $\alpha$ -ih  $\beta$ -ih by blast
} moreover {
  assume *:  $\psi \in ?back$ 
  then obtain  $i \ x$  where  $\psi$ -is:  $\psi = \text{And-mltl-ext}$ 
    (Global-mltl-ext
    (interval-times  $a \ L \ ! \ 0$ )
    (interval-times  $a \ L \ ! \ i - 1$ ) [?s!i - ?s!0] (And-mltl-ext (Notc
 $\alpha$ )  $\beta$ ))
    (Mighty-Release-mltl-ext  $x \ \beta$ 
    (interval-times  $a \ L \ ! \ i$ )
    (interval-times  $a \ L \ ! \ (i + 1) - 1$ )
    [interval-times  $a \ L \ ! \ (i + 1) -$ 
    interval-times  $a \ L \ ! \ i$ ])
    and  $x$ -in:  $x \in \text{set } ?D$ 
    and  $i$ -bound:  $1 \leq i \wedge i < \text{length } L$ 
  by auto
  have lb:  $a < ?s!i$ 
    using interval-times-diff-ge-general[OF a-leq-b composition-L, of i 0 ?s]
    using sfirst  $i$ -bound by simp
  have welldef: (interval-times  $a \ L \ ! \ i$ ) < (interval-times  $a \ L \ ! \ (i + 1)$ )
    using interval-times-diff-ge[OF a-leq-b composition-L, of i ?s]
    using  $i$ -bound by simp
  have ub: ?s!( $i+1$ )  $\leq b+1$ 
    using slast  $i$ -bound
    using interval-times-diff-ge-general[OF a-leq-b composition-L, of length L

```

```

i+1 ?s]
  by (metis Orderings.order-eq-iff less-iff-succ-less-eq order-le-imp-less-or-eq
order-less-imp-le)
  have x-ih: intervals-welldef (to-mltl x)
    using Suc(1)
    using  $\alpha$ -ih composition- $\alpha$  x-in by blast
  have ?thesis unfolding  $\psi$ -is to-mltl.simps intervals-welldef.simps Mighty-Release-mltl-ext.simps
    using x-ih  $\alpha$ -ih  $\beta$ -ih ub lb welldef
    by (simp add: add-le-imp-le-diff sfirst)
}
ultimately show ?thesis
  using Suc(3) unfolding Release-mltl-ext LP-mltl-aux.simps
  using split by blast
qed
qed

```

lemma *LP-mltl-aux-wpd*:

```

assumes  $\exists \varphi$ -init.  $\varphi = \text{convert-nnf-ext } \varphi\text{-init}$ 
assumes intervals-welldef (to-mltl  $\varphi$ )
assumes  $\psi \in \text{set } (LP\text{-mltl-aux } \varphi \ k)$ 
assumes is-composition-MLTL  $\varphi$ 
shows wpd-mltl (to-mltl  $\psi$ )  $\leq$  wpd-mltl (to-mltl  $\varphi$ )
  using assms
proof(induct k arbitrary:  $\varphi \ \psi$ )
  case 0
  then show ?case by auto
next
  case (Suc k)
  then show ?case
  proof(cases  $\varphi$ )
    case True-mltl-ext
    then show ?thesis using Suc by auto
  next
    case False-mltl-ext
    then show ?thesis using Suc by auto
  next
    case (Prop-mltl-ext p)
    then show ?thesis using Suc by auto
  next
    case (Not-mltl-ext q)
    then have  $\exists p$ .  $q = \text{Prop-mltl-ext } p$ 
      using convert-nnf-form-Not-Implies-Prop Suc
    by (metis convert-nnf-ext-to-mltl-commute to-mltl.simps(4) to-mltl-prop-bijective)

  then obtain p where  $q = \text{Prop-mltl-ext } p$  by blast
  then show ?thesis
    using Not-mltl-ext Suc.prem(3) by fastforce
  next

```

```

case (And-mltl-ext  $\alpha$   $\beta$ )
obtain  $x$   $y$  where  $\psi$ -is:  $\psi = \text{And-mltl-ext } x \ y$ 
      and  $x$ -in:  $x \in \text{set } (LP\text{-mltl-aux } \alpha \ k)$ 
      and  $y$ -in:  $y \in \text{set } (LP\text{-mltl-aux } \beta \ k)$ 
      using Suc unfolding And-mltl-ext LP-mltl-aux.simps
      by (metis And-mltl-list-member convert-nnf-ext.simps(4) convert-nnf-ext-convert-nnf-ext
in-set-member mltl-ext.inject(3))
      have  $\alpha$ -nnf:  $\exists \varphi\text{-init. } \alpha = \text{convert-nnf-ext } \varphi\text{-init}$ 
      using Suc(2) unfolding And-mltl-ext
      by (metis convert-nnf-ext.simps(4) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(3))

      have  $\beta$ -nnf:  $\exists \varphi\text{-init. } \beta = \text{convert-nnf-ext } \varphi\text{-init}$ 
      using Suc(2) unfolding And-mltl-ext
      by (metis convert-nnf-ext.simps(4) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(3))

      have  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ ) and
         $\beta$ -welldef: intervals-welldef (to-mltl  $\beta$ )
      using Suc(3) unfolding And-mltl-ext by simp-all
      have  $\alpha$ -composition: is-composition-MLTL  $\alpha$  and
         $\beta$ -composition: is-composition-MLTL  $\beta$ 
      using Suc(5) unfolding And-mltl-ext is-composition-MLTL.simps by simp-all
      have  $x$ -ih: wpd-mltl (to-mltl  $x$ )  $\leq$  wpd-mltl (to-mltl  $\alpha$ )
      using Suc.hyps[of  $\alpha$   $x$ , OF  $\alpha$ -nnf  $\alpha$ -welldef  $x$ -in  $\alpha$ -composition] by blast
      have  $y$ -ih: wpd-mltl (to-mltl  $y$ )  $\leq$  wpd-mltl (to-mltl  $\beta$ )
      using Suc.hyps[of  $\beta$   $y$ , OF  $\beta$ -nnf  $\beta$ -welldef  $y$ -in  $\beta$ -composition] by blast
      show ?thesis
      unfolding And-mltl-ext  $\psi$ -is to-mltl.simps wpd-mltl.simps
      using  $x$ -ih  $y$ -ih by linarith
next
case (Or-mltl-ext  $\alpha$   $\beta$ )
let  $?Dx = LP\text{-mltl-aux } \alpha \ k$ 
let  $?Dy = LP\text{-mltl-aux } \beta \ k$ 
have  $\alpha$ -nnf:  $\exists \varphi\text{-init. } \alpha = \text{convert-nnf-ext } \varphi\text{-init}$ 
      using Suc(2) unfolding Or-mltl-ext
      by (metis convert-nnf-ext.simps(5) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(4))

      have  $\beta$ -nnf:  $\exists \varphi\text{-init. } \beta = \text{convert-nnf-ext } \varphi\text{-init}$ 
      using Suc(2) unfolding Or-mltl-ext
      by (metis convert-nnf-ext.simps(5) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(4))

      have  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ ) and
         $\beta$ -welldef: intervals-welldef (to-mltl  $\beta$ )
      using Suc(3) unfolding Or-mltl-ext by simp-all
      have  $\alpha$ -composition: is-composition-MLTL  $\alpha$  and
         $\beta$ -composition: is-composition-MLTL  $\beta$ 
      using Suc(5) unfolding Or-mltl-ext is-composition-MLTL.simps by simp-all
      {
      assume  $*$ :  $\psi \in \text{set } (\text{And-mltl-list } ?Dx \ ?Dy)$ 
      then obtain  $x$   $y$  where  $\psi$ -is:  $\psi = \text{And-mltl-ext } x \ y$ 

```

```

      and  $x$ -in:  $x \in \text{set } ?Dx$  and  $y$ -in:  $y \in \text{set } ?Dy$ 
      using And-mltl-list-member[of  $\psi$   $?Dx$   $?Dy$ ]
      by (metis in-set-member)
    have  $x$ -ih: wpd-mltl (to-mltl  $x$ )  $\leq$  wpd-mltl (to-mltl  $\alpha$ )
      using Suc.hyps[of  $\alpha$   $x$ , OF  $\alpha$ -nnf  $\alpha$ -welldef  $x$ -in  $\alpha$ -composition] by blast
    have  $y$ -ih: wpd-mltl (to-mltl  $y$ )  $\leq$  wpd-mltl (to-mltl  $\beta$ )
      using Suc.hyps[of  $\beta$   $y$ , OF  $\beta$ -nnf  $\beta$ -welldef  $y$ -in  $\beta$ -composition] by blast

  have ?thesis
    unfolding Or-mltl-ext  $\psi$ -is to-mltl.simps wpd-mltl.simps
    using  $x$ -ih  $y$ -ih by linarith
} moreover {
  assume *:  $\psi \in \text{set } (\text{And-mltl-list } [\text{Not}_c \alpha] ?Dy)$ 
  then obtain  $y$  where  $\psi$ -is:  $\psi = \text{And-mltl-ext } (\text{Not}_c \alpha) y$ 
    and  $y$ -in:  $y \in \text{set } ?Dy$ 
    using And-mltl-list-member[of  $\psi$   $[\text{Not}_c \alpha] ?Dy$ ]
    by auto
  have  $y$ -ih: wpd-mltl (to-mltl  $y$ )  $\leq$  wpd-mltl (to-mltl  $\beta$ )
    using Suc.hyps[of  $\beta$   $y$ , OF  $\beta$ -nnf  $\beta$ -welldef  $y$ -in  $\beta$ -composition] by blast

  have ?thesis
    unfolding Or-mltl-ext  $\psi$ -is to-mltl.simps wpd-mltl.simps
    using  $y$ -ih by auto
} moreover {
  assume *:  $\psi \in \text{set } (\text{And-mltl-list } ?Dx [\text{Not}_c \beta])$ 
  then obtain  $x$  where  $\psi$ -is:  $\psi = \text{And-mltl-ext } x (\text{Not}_c \beta)$ 
    and  $x$ -in:  $x \in \text{set } ?Dx$ 
    using And-mltl-list-member[of  $\psi$   $?Dx$   $[\text{Not}_c \beta]$ ]
    by (metis in-set-member member-rec(1) member-rec(2))
  have  $x$ -ih: wpd-mltl (to-mltl  $x$ )  $\leq$  wpd-mltl (to-mltl  $\alpha$ )
    using Suc.hyps[of  $\alpha$   $x$ , OF  $\alpha$ -nnf  $\alpha$ -welldef  $x$ -in  $\alpha$ -composition] by blast
  have ?thesis
    unfolding Or-mltl-ext  $\psi$ -is to-mltl.simps wpd-mltl.simps
    using  $x$ -ih by auto
}
}
ultimately show ?thesis
  using Suc unfolding Or-mltl-ext LP-mltl-aux.simps
  using list-concat-set-union
  by (metis UnE  $\alpha$ -nnf  $\beta$ -nnf convert-nnf-ext-convert-nnf-ext)
next
case (Future-mltl-ext  $a$   $b$   $L$   $\alpha$ )
let ?D = LP-mltl-aux  $\alpha$   $k$ 
let ?s = interval-times  $a$   $L$ 
let ?front = (Future-mltl-list ?D (?s ! 0) (?s ! 1 - 1) [?s ! 1 - ?s ! 0])
let ?back = (concat (map ( $\lambda i$ . And-mltl-list
  [Global-mltl-ext (?s ! 0)
  (?s ! i - 1) [?s!i - ?s!0] (Notc  $\alpha$ )]
  (Future-mltl-list ?D (?s ! i) (?s ! (i + 1) - 1)
  [?s ! (i + 1) - ?s ! i]))

```



```

[1..<length L])
have a-leq-b: a ≤ b using Suc(3)
  unfolding Future-mltl-ext to-mltl.simps intervals-welldef.simps
  by blast
have composition-L: is-composition (b-a+1) L and
  composition-α: is-composition-MLTL α using Suc(5)
  unfolding Future-mltl-ext is-composition-MLTL.simps by simp-all
have α-nnf: ∃ φ-init. α = convert-nnf-ext φ-init
  using Suc(2) unfolding Future-mltl-ext
by (metis convert-nnf-ext.simps(6) convert-nnf-ext-convert-nnf-ext mtl-ext.inject(5))

have α-welldef: intervals-welldef (to-mltl α)
  using Suc(3) unfolding Future-mltl-ext by simp
have nnf: convert-nnf-ext α = α
  using α-nnf convert-nnf-ext-convert-nnf-ext by metis
have slast: interval-times a L ! (length L) = b+1
  using interval-times-last[OF a-leq-b composition-L] by blast
then have split: ψ ∈ (set ?front) ∪ (set ?back)
  using Suc(4) unfolding Future-mltl-ext LP-mltl-aux.simps nnf
  using list-concat-set-union[of ?front ?back] by metis
{
  assume *: ψ ∈ set ?front
  then obtain x where ψ-is: ψ = Future-mltl-ext (?s ! 0) (?s ! 1 - 1) [?s ! 1
- ?s ! 0] x
    and x-in: x ∈ set ?D
    unfolding Future-mltl-list.simps by fastforce
  have length-s: 1 < length ?s using ψ-is
    by (metis One-nat-def add commute add-gr-0 add-less-cancel-right com-
position-L composition-length-lb interval-times-length plus-1-eq-Suc zero-less-one)

  then have length-L: 1 ≤ length L
    unfolding interval-times-def
    by (simp add: less-eq-iff-succ-less)
  have interval-times a L ! 1 ≤ interval-times a L ! (length L)
    using interval-times-diff-ge-general[OF a-leq-b composition-L, of length L 1
?s]
    using length-L by force
  then have bound: interval-times a L ! 1 - 1 ≤ b
    using slast by auto
  have ih: wpd-mltl (to-mltl x) ≤ wpd-mltl (to-mltl α)
    using Suc(1)[OF α-nnf α-welldef x-in composition-α] by blast
  have ?thesis
    unfolding ψ-is Future-mltl-ext to-mltl.simps wpd-mltl.simps
    using bound ih by simp
} moreover {
  assume *: ψ ∈ set ?back
  then obtain i where ψ-is: ψ ∈ set ((And-mltl-list
[Global-mltl-ext (?s ! 0)
(?s ! i - 1) [?s!i - ?s!0] (Notc α)]

```

```

      (Future-mltl-list ?D (?s ! i) (?s ! (i + 1) - 1)
        [?s ! (i + 1) - ?s ! i])
    ))
  and i-in: i ∈ {1..<length L}
  by force
then obtain x where ψ-is: ψ = ((And-mltl-ext
  (Global-mltl-ext (?s ! 0)
    (?s ! i - 1) [?s!i - ?s!0] (Notc α))
  (Future-mltl-ext (?s ! i) (?s ! (i + 1) - 1)
    [?s ! (i + 1) - ?s ! i] x)))
  and x-in: x ∈ set ?D
  by auto
have ih: wpd-mltl (to-mltl x) ≤ wpd-mltl (to-mltl α)
  using Suc.hyps(1)[OF α-nnf α-welldef x-in composition-α] by blast
have bound: interval-times a L ! i < interval-times a L ! (i + 1)
  using interval-times-diff-ge[OF a-leq-b composition-L, of i ?s]
  using i-in by simp
have (interval-times a L ! (i + 1) - 1) ≤ b using slast
  using interval-times-diff-ge-general[OF a-leq-b composition-L, of length L
i+1 ?s] i-in
  by (metis Suc-eq-plus1 atLeastLessThan-iff le-Suc-eq le-diff-conv linorder-not-less
order-less-imp-le verit-comp-simplify1(2))
  then have ?thesis
    unfolding ψ-is Future-mltl-ext to-mltl.simps wpd-mltl.simps
    using ih bound by linarith
}
ultimately show ?thesis using split by blast
next
case (Global-mltl-ext a b L α)
let ?D-α = LP-mltl-aux α k
have a-leq-b: a ≤ b and α-welldef: intervals-welldef (to-mltl α)
  using Suc(3)
  unfolding Global-mltl-ext to-mltl.simps intervals-welldef.simps
  by simp-all
have composition-α: is-composition-MLTL α using Suc(5)
  unfolding Global-mltl-ext is-composition-MLTL.simps by simp-all
have α-nnf: ∃ φ-init. α = convert-nnf-ext φ-init
  using Suc(2) unfolding Global-mltl-ext
  by (metis convert-nnf-ext.simps(7) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(6))

have α-welldef: intervals-welldef (to-mltl α)
  using Suc(3) unfolding Global-mltl-ext by simp
have nnf: convert-nnf-ext α = α
  using α-nnf convert-nnf-ext-convert-nnf-ext by metis
{
  assume *: length ?D-α ≤ 1
  then have ψ-is: ψ = Global-mltl-ext a b L α
    using Suc unfolding Global-mltl-ext LP-mltl-aux.simps
    using nnf by fastforce
}

```

```

have ?thesis unfolding  $\psi$ -is Global-mltl-ext by simp
} moreover {
  assume *: length ?D- $\alpha$  > 1
  then have  $\psi$ -in:  $\psi \in \text{set } (\text{Global-mltl-decomp } ?D\text{-}\alpha \ a \ (b - a) \ L)$ 
    using Suc nnf unfolding Global-mltl-ext LP-mltl-aux.simps
    by simp
  then obtain X where  $\psi$ -is:  $\psi = \text{And-s-mltl-ext } X$ 
    and X-fact:  $\forall i < \text{length } X. \exists y \in \text{set } (\text{LP-mltl-aux } \alpha \ k).$ 
       $X ! i = \text{Global-mltl-ext } (a + i) \ (a + i) \ [1] \ y$ 
    and length-X: length X = Suc (b - a)
    using in-Global-mltl-decomp-exact-forward[OF *  $\psi$ -in] nnf a-leq-b
    unfolding List.member-def by blast
  have X-ih: wpd-mltl (to-mltl (X!i))  $\leq$  b+wpd-mltl (to-mltl  $\alpha$ )
    if i-bound:  $i < \text{length } X$  for i
  proof-
    obtain x where x-in:  $x \in \text{set } ?D\text{-}\alpha$ 
      and Xi-is:  $X!i = \text{Global-mltl-ext } (a+i) \ (a+i) \ [1] \ x$ 
      using X-fact a-leq-b i-bound by blast
    have wpd-mltl (to-mltl x)  $\leq$  wpd-mltl (to-mltl  $\alpha$ )
      using Suc.hyps[OF  $\alpha$ -nnf  $\alpha$ -welldef x-in composition- $\alpha$ ] by simp
    then show ?thesis unfolding Xi-is to-mltl.simps wpd-mltl.simps
      using a-leq-b length-X i-bound by auto
  qed
  have ?thesis
    unfolding  $\psi$ -is Global-mltl-ext to-mltl.simps wpd-mltl.simps
    using X-ih length-X X-fact Suc(1)
  proof(induct b-a arbitrary:X a b)
    case 0
    then have length X = 1
      by simp
    then obtain x where X-is:  $X = [x]$ 
      by (metis One-nat-def Suc-length-conv length-0-conv)
    show ?case using 0(2)[of 0] unfolding X-is
      using And-s-mltl-ext.simps(2)
    by (metis X-is  $\langle \text{length } X = 1 \rangle$  length-0-conv less-one nth-Cons' self-append-conv2)

  next
    case (Suc n)
    then have length-X: length X = n + 2 by linarith
    then obtain H t where X-is:  $X = H@[t]$  and length-H: length H = length
X-1
      by (metis Suc.prem(2) diff-Suc-1 length-Suc-conv-rev)
    have And-s: And-s-mltl-ext X = And-s-mltl-ext (And-s-mltl-ext H) t
      unfolding X-is And-s-mltl-ext.simps using length-H length-X by simp
    have t-bound: (wpd-mltl (to-mltl t))  $\leq$  b + wpd-mltl (to-mltl  $\alpha$ )
      using Suc(3)[of length X-1] unfolding X-is by auto
    have cond1:  $n = b - 1 - a$  using Suc by auto
    have cond2: wpd-mltl (to-mltl (H ! i))
       $\leq$  b + wpd-mltl (to-mltl  $\alpha$ )-1

```

```

    if i-bound: i < length H for i
  proof-
    have Hi-is: H!i = X!i using X-is i-bound
      by (simp add: nth-append)
    have  $\exists y \in \text{set } (LP\text{-mttl-aux } \alpha \ k). \ X \ ! \ i = \text{Global-mttl-ext } (a + i) \ (a + i)$ 
  [1] y
      using Suc(3)[of i] Suc(5) i-bound
    by (metis Suc.prem(2) add-diff-cancel-left' length-H less-Suc-eq plus-1-eq-Suc)

  then obtain y where Xi-is: X ! i = Global-mttl-ext (a + i) (a + i) [1] y
    and y-in: y ∈ set (LP-mttl-aux  $\alpha$  k)
      by auto
    have ih: wpd-mttl (to-mttl (X ! i)) ≤ b + wpd-mttl (to-mttl  $\alpha$ )
      using i-bound Suc(3)[of i] X-is by auto
    have bound: a+i < b
      using i-bound length-H length-X
      by (simp add: Suc.prem(2))
    have wpd-mttl (to-mttl y) ≤ wpd-mttl (to-mttl  $\alpha$ )
      using Suc(6)[OF  $\alpha$ -nnf  $\alpha$ -welldef y-in composition- $\alpha$ ] by blast
    then show ?thesis unfolding Hi-is Xi-is to-mttl.simps wpd-mttl.simps
      using bound by simp
  qed
  have cond3: length H = Suc (b - 1 - a)
    using length-H length-X Suc.hyps(2) by simp
  have cond4:  $\exists y \in \text{set } (LP\text{-mttl-aux } \alpha \ k). \ H \ ! \ i = \text{Global-mttl-ext } (a + i) \ (a$ 
+ i) [1] y
    if i-bound: i < length H for i
  proof-
    have  $\exists y \in \text{set } (LP\text{-mttl-aux } \alpha \ k). \ X \ ! \ i = \text{Global-mttl-ext } (a + i) \ (a + i)$ 
  [1] y
      using Suc(5) i-bound length-H by auto
    then obtain y where y-in: y ∈ set (LP-mttl-aux  $\alpha$  k) and
      Xi-is: X ! i = Global-mttl-ext (a + i) (a + i) [1] y
      by blast
    then have Hi-is: H!i = X!i using i-bound length-H
      by (metis X-is nth-append)
    then show ?thesis unfolding Xi-is using y-in by blast
  qed
  have ih: wpd-mttl (to-mttl (Ands-mttl-ext H))
≤ b - 1 + wpd-mttl (to-mttl  $\alpha$ )
    using Suc.hyps(1)[of b-1 a H, OF cond1 - cond3] cond2 cond4 Suc.prem(4)
    by force
  show ?case unfolding Ands wpd-mttl.simps to-mttl.simps
    using t-bound ih by simp
  qed
}
ultimately show ?thesis by linarith
next
case (Until-mttl-ext  $\alpha$  a b L  $\beta$ )

```

```

let ?D- $\alpha$  = LP-mltl-aux  $\alpha$  k
let ?D- $\beta$  = LP-mltl-aux  $\beta$  k
let ?s = interval-times a L
have a-leq-b:  $a \leq b$  and  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ )
    and  $\beta$ -welldef: intervals-welldef (to-mltl  $\alpha$ )
  using Suc(3)
  unfolding Until-mltl-ext to-mltl.simps intervals-welldef.simps
  by simp-all
have composition- $\alpha$ : is-composition-MTLT  $\alpha$  and
  composition- $\beta$ : is-composition-MTLT  $\beta$  and
  composition-L: is-composition (b-a+1) L using Suc(5)
  unfolding Until-mltl-ext is-composition-MTLT.simps by simp-all
have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
  using Suc(2) unfolding Until-mltl-ext
by (metis convert-nnf-ext.simps(8) convert-nnf-ext-convert-nnf-ext mtl-ext.inject(7))

have  $\beta$ -nnf:  $\exists \varphi$ -init.  $\beta = \text{convert-nnf-ext } \varphi$ -init
  using Suc(2) unfolding Until-mltl-ext
by (metis convert-nnf-ext.simps(8) convert-nnf-ext-convert-nnf-ext mtl-ext.inject(7))
have  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ ) and
   $\beta$ -welldef: intervals-welldef (to-mltl  $\beta$ )
  using Suc(3) unfolding Until-mltl-ext by simp-all
have convert- $\alpha$ : convert-nnf-ext  $\alpha = \alpha$ 
  by (metis  $\alpha$ -nnf convert-nnf-ext-convert-nnf-ext)
have convert- $\beta$ : convert-nnf-ext  $\beta = \beta$ 
by (metis Suc.prem(1) Until-mltl-ext convert-nnf-ext.simps(8) convert-nnf-ext-convert-nnf-ext
mtl-ext.inject(7))
have slast: interval-times a L ! (length L) = b+1
  using interval-times-last[OF a-leq-b composition-L] by blast
let ?front = (Until-mltl-list  $\alpha$  ?D- $\beta$  (?s ! 0) (?s ! 1 - 1) [?s ! 1 - ?s ! 0])
let ?back = (concat (map ( $\lambda i$ . And-mltl-list
  [Global-mltl-ext
    (?s ! 0) (?s ! i - 1) [?s!i - ?s!0] (And-mltl-ext  $\alpha$  (Not_c
 $\beta$ ))])
  (Until-mltl-list  $\alpha$  ?D- $\beta$  (?s ! i) (?s ! (i + 1) - 1)
    [?s ! (i + 1) - ?s ! i]) [1..<length L]))
have split:  $\psi \in (\text{set } ?\text{front}) \cup (\text{set } ?\text{back})$ 
  using Suc(4) unfolding Until-mltl-ext LP-mltl-aux.simps
  using convert- $\alpha$  convert- $\beta$  list-concat-set-union by metis
{
  assume *:  $\psi \in \text{set } ?\text{front}$ 
  then obtain y where  $\psi$ -is:  $\psi = \text{Until-mltl-ext } \alpha$  (interval-times a L ! 0)
    (interval-times a L ! 1 - 1) [interval-times a L ! 1 - interval-times
a L ! 0] y
    and y-in:  $y \in \text{set } ?\text{D-}\beta$ 
  by auto
  have length-s:  $1 < \text{length } ?s$  using  $\psi$ -is
    by (metis One-nat-def add commute add-gr-0 add-less-cancel-right com-
position-L composition-length-lb interval-times-length plus-1-eq-Suc zero-less-one)

```

```

then have length-L:  $1 \leq \text{length } L$ 
  unfolding interval-times-def
  by (simp add: less-eq-iff-succ-less)
have interval-times a L ! 1 ≤ interval-times a L ! (length L)
  using interval-times-diff-ge-general[OF a-leq-b composition-L, of length L 1
?S]
  using length-L by force
then have bound: interval-times a L ! 1 - 1 ≤ b
  using slast by auto
have  $\beta$ -ih:  $\text{wpd-mltl } (to\text{-mltl } y) \leq \text{wpd-mltl } (to\text{-mltl } \beta)$ 
  using Suc.hyps(1)[OF  $\beta$ -nnf  $\beta$ -welldef  $y$ -in composition- $\beta$ ] by blast
have ?thesis
  unfolding  $\psi$ -is Until-mltl-ext to-mltl.simps wpd-mltl.simps
  using  $\beta$ -ih bound by linarith
} moreover {
  assume  $*$ :  $\psi \in \text{set } ?back$ 
  then obtain  $i$   $y$  where
   $\psi$ -is:  $\psi = \text{And-mltl-ext } (Global\text{-mltl-ext } (?s!0) (?s!i-1) [?s!i - ?s!0]) (\text{And-mltl-ext}$ 
 $\alpha$  (Notc  $\beta$ )))
  (Until-mltl-ext  $\alpha$  ( $?s!i$ ) ( $?s!(i+1)-1$ ) [ $(?s!(i+1)) - (?s!i)$ ]  $y$ )
  and  $i$ -bound:  $1 \leq i \wedge i < \text{length } L$ 
  and  $y$ -in:  $y \in \text{set } ?D$ - $\beta$ 
  by auto
  have bound1:  $\text{interval-times } a \ L \ ! \ i < \text{interval-times } a \ L \ ! \ (i+1)$ 
  using interval-times-diff-ge[OF a-leq-b composition-L, of i ?s]
  using  $i$ -bound by blast
  have  $\text{interval-times } a \ L \ ! \ (i + 1) \leq \text{interval-times } a \ L \ ! \ (\text{length } L)$ 
  using interval-times-diff-ge-general[OF a-leq-b composition-L, of length L
i+1 ?s]
  using  $i$ -bound by (metis less-iff-succ-less-eq order-le-less)
  then have bound2:  $\text{interval-times } a \ L \ ! \ (i+1) \leq b+1$ 
  using slast by simp
  have  $\beta$ -ih:  $\text{wpd-mltl } (to\text{-mltl } y) \leq \text{wpd-mltl } (to\text{-mltl } \beta)$ 
  using Suc.hyps(1)[OF  $\beta$ -nnf  $\beta$ -welldef  $y$ -in composition- $\beta$ ] by blast
  have  $\text{interval-times } a \ L \ ! \ i > \text{interval-times } a \ L \ ! \ 0$ 
  using  $i$ -bound interval-times-diff-ge-general[OF a-leq-b composition-L, of i 0
?S]
  by auto
  then have  $\text{interval-times } a \ L \ ! \ i > 0$ 
  unfolding interval-times-def by simp
  then have  $b > \text{interval-times } a \ L \ ! \ i - 1$ 
  using bound1 bound2 by simp
  then have case1:  $(\text{interval-times } a \ L \ ! \ i - 1 +$ 
 $\max (\text{wpd-mltl } (to\text{-mltl } \alpha))$ 
 $(\text{wpd-mltl } (to\text{-mltl } \beta))) \leq$ 
 $b + \max (\text{wpd-mltl } (to\text{-mltl } \alpha))$ 
 $(\text{wpd-mltl } (to\text{-mltl } \beta))$ 
  using bound1 bound2  $\beta$ -ih by linarith

```

```

have case2: (interval-times a L ! (i + 1) - 1 +
max (wpd-mltl (to-mltl  $\alpha$ ))
(wpd-mltl (to-mltl y)))  $\leq$ 
    b + max (wpd-mltl (to-mltl  $\alpha$ ))
    (wpd-mltl (to-mltl  $\beta$ ))
using bound1 bound2  $\beta$ -ih by linarith
have ?thesis
unfolding Until-mltl-ext  $\psi$ -is to-mltl.simps wpd-mltl.simps
using case1 case2
by presburger
}
ultimately show ?thesis using split by blast
next
case (Release-mltl-ext  $\alpha$  a b L  $\beta$ )
let ?D = LP-mltl-aux  $\alpha$  k
let ?s = interval-times a L
have a-leq-b: a  $\leq$  b and  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ )
and  $\beta$ -welldef: intervals-welldef (to-mltl  $\alpha$ )
using Suc(3)
unfolding Release-mltl-ext to-mltl.simps intervals-welldef.simps
by simp-all
have composition- $\alpha$ : is-composition-MLTL  $\alpha$  and
    composition- $\beta$ : is-composition-MLTL  $\beta$  and
    composition-L: is-composition (b-a+1) L using Suc(5)
unfolding Release-mltl-ext is-composition-MLTL.simps by simp-all
have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha$  = convert-nnf-ext  $\varphi$ -init
using Suc(2) unfolding Release-mltl-ext
by (metis convert-nnf-ext.simps(9) convert-nnf-ext-convert-nnf-ext mtl-ext.inject(8))

have  $\beta$ -nnf:  $\exists \varphi$ -init.  $\alpha$  = convert-nnf-ext  $\varphi$ -init
using Suc(2) unfolding Release-mltl-ext
by (metis convert-nnf-ext.simps(9) convert-nnf-ext-convert-nnf-ext mtl-ext.inject(8))
have  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ ) and
     $\beta$ -welldef: intervals-welldef (to-mltl  $\alpha$ )
using Suc(3) unfolding Release-mltl-ext by simp-all
have convert- $\alpha$ : convert-nnf-ext  $\alpha$  =  $\alpha$ 
by (metis  $\alpha$ -nnf convert-nnf-ext-convert-nnf-ext)
have convert- $\beta$ : convert-nnf-ext  $\beta$  =  $\beta$ 
by (metis Suc.prem(1) Release-mltl-ext convert-nnf-ext.simps(9) convert-nnf-ext-convert-nnf-ext
mtl-ext.inject(8))
have slast: interval-times a L ! (length L) = b+1
using interval-times-last[OF a-leq-b composition-L] by blast
have sfirst: ?s!0 = a
using interval-times-first by blast
have length-L: length L > 0
using composition-length-lb composition-L by simp
let ?front = set [Global-mltl-ext a b L (And-mltl-ext (Notc  $\alpha$ )  $\beta$ )]
let ?middle = set (Mighty-Release-mltl-list ?D  $\beta$  (?s ! 0) (?s ! 1 - 1)
[?s ! 1 - ?s ! 0])

```

```

let ?back = set (concat
  (map ( $\lambda i$ . And-mltl-list
    [Global-mltl-ext
      (?s ! 0)
      (?s ! i - 1) [?s!i - ?s!0] (And-mltl-ext (Notc  $\alpha$ )  $\beta$ )]
    (Mighty-Release-mltl-list ?D  $\beta$  (?s ! i)
      (?s ! (i + 1) - 1) [?s ! (i + 1) - ?s ! i]))
    [1.. $\text{length } L$ ]))
have split:  $\psi \in ?\text{front} \cup ?\text{middle} \cup ?\text{back}$ 
  using Suc(4) unfolding Release-mltl-ext LP-mltl-aux.simps
  using list-concat-set-union
  by (metis append.assoc convert- $\alpha$ )
{
  assume *:  $\psi \in ?\text{front}$ 
  then have  $\psi$ -is:  $\psi = \text{Global-mltl-ext } a \ b \ L \ (\text{And-mltl-ext } (\text{Not}_c \ \alpha) \ \beta)$ 
    by simp
  have ?thesis unfolding Release-mltl-ext  $\psi$ -is to-mltl.simps wpd-mltl.simps
    by linarith
} moreover {
  assume *:  $\psi \in ?\text{middle}$ 
  then obtain  $x$  where  $\psi$ -is:  $\psi = \text{Mighty-Release-mltl-ext } x \ \beta \ (\text{interval-times } a \ L ! 0)$ 
    (interval-times  $a \ L ! 1 - 1$ )
    [interval-times  $a \ L ! 1 - \text{interval-times } a \ L ! 0$ ]
    and  $x$ -in:  $x \in \text{set } ?D$ 
  by auto
  have ub: interval-times  $a \ L ! 1 - 1 \leq b$ 
    using interval-times-diff-ge-general[OF  $a$ -leq- $b$  composition- $L$ , of length  $L \ 1$ 
    ?s]
    using slast length- $L$ 
  by (metis diff-add-inverse2 diff-le-self dual-order.strict-iff-order dual-order.trans
  less-eq-iff-succ-less zero-less-diff)
  have  $x$ -ih: wpd-mltl (to-mltl  $x$ )  $\leq$  wpd-mltl (to-mltl  $\alpha$ )
    using Suc(1)[OF  $\alpha$ -nnf  $\alpha$ -welldef  $x$ -in composition- $\alpha$ ]
    by blast
  then have ?thesis unfolding  $\psi$ -is Release-mltl-ext to-mltl.simps wpd-mltl.simps
  Mighty-Release-mltl-ext.simps
    using ub by auto
} moreover {
  assume *:  $\psi \in ?\text{back}$ 
  then obtain  $x \ i$  where  $\psi$ -is:  $\psi = \text{And-mltl-ext } ( \text{Global-mltl-ext } ( \text{interval-times } a \ L ! 0 ) ( \text{interval-times } a \ L ! i - 1 ) [?s!i - ?s!0] ( \text{And-mltl-ext } (\text{Not}_c \ \alpha) \ \beta ) ) ( \text{Mighty-Release-mltl-ext } x \ \beta ( \text{interval-times } a \ L ! i ) ( \text{interval-times } a \ L ! (i + 1) - 1 ) [ \text{interval-times } a \ L ! (i + 1) -$ 

```



```

      interval-times a L ! i])
    and x-in: x ∈ set ?D
    and i-bound: 1 ≤ i ∧ i < length L
  by auto
  have x-ih: wpd-mltl (to-mltl x) ≤ wpd-mltl (to-mltl α)
    using Suc(1)[OF α-nnf α-welldef x-in composition-α] by blast
  have lb: a < ?s!i
    using interval-times-diff-ge-general sfirst
    by (smt (verit, cfv-SIG) a-leq-b composition-L i-bound less-or-eq-imp-le
order-less-le-trans zero-less-one)
  have welldef: ?s!i < ?s!(i+1)
    using interval-times-diff-ge[OF a-leq-b composition-L]
    using i-bound length-L by blast
  have ub: ?s!(i+1) ≤ b+1
    using interval-times-diff-ge-general[OF a-leq-b composition-L, of length L
i+1 ?s]
    using i-bound slast
    by (metis less-iff-succ-less-eq order-le-imp-less-or-eq order-less-imp-le or-
der-refl)
  have ?thesis unfolding Release-mltl-ext ψ-is to-mltl.simps wpd-mltl.simps
Mighty-Release-mltl-ext.simps
    using lb welldef ub x-ih by auto
  }
  ultimately show ?thesis
    using split by blast
qed
qed

```

```

lemma And-mltl-list-nonempty:
  assumes A ≠ [] and B ≠ []
  shows And-mltl-list A B ≠ []
proof-
  have length A > 0
    using assms by blast
  then obtain ha Ta where A: A = ha#Ta
    using list.exhaust by auto
  have length B > 0
    using assms by blast
  then obtain hb Tb where B: B = hb#Tb
    using list.exhaust by auto
  show ?thesis
    using assms unfolding And-mltl-list.simps A B pairs.simps
    by blast
qed

```

```

lemma Global-mltl-decomp-nonempty:
  assumes D ≠ []
  shows Global-mltl-decomp D a n L ≠ []
  using assms

```

```

proof(induct n)
  case 0
  then show ?case by simp
next
  case (Suc n)
  then show ?case unfolding Global-mltl-decomp.simps Global-mltl-list.simps
    using And-mltl-list-nonempty by auto
qed

lemma LP-mltl-aux-nonempty:
  assumes  $\exists \varphi$ -init.  $\varphi = \text{convert-nnf-ext } \varphi$ -init
  assumes intervals-welldef (to-mltl  $\varphi$ )
  assumes is-composition-MLTL  $\varphi$ 
  shows LP-mltl-aux  $\varphi$   $k \neq []$ 
  using assms
proof(induct k arbitrary: \varphi)
  case 0
  then show ?case by simp
next
  case (Suc k)
  then show ?case
  proof(cases \varphi)
    case True-mltl-ext
    then show ?thesis by simp
  next
    case False-mltl-ext
    then show ?thesis by simp
  next
    case (Prop-mltl-ext p)
    then show ?thesis by simp
  next
    case (Not-mltl-ext q)
    then have  $\exists p$ .  $q = \text{Prop-mltl-ext } p$ 
      using convert-nnf-form-Not-Implies-Prop Suc
    by (metis convert-nnf-ext-to-mltl-commute to-mltl.simps(4) to-mltl-prop-bijective)

  then obtain  $p$  where  $q = \text{Prop-mltl-ext } p$  by blast
  then show ?thesis
    unfolding Not-mltl-ext by simp
next
  case (And-mltl-ext \alpha \beta)
  have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
    using Suc(2) unfolding And-mltl-ext
  by (metis convert-nnf-ext.simps(4) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(3))

  have  $\beta$ -nnf:  $\exists \varphi$ -init.  $\beta = \text{convert-nnf-ext } \varphi$ -init
    using Suc(2) unfolding And-mltl-ext
  by (metis convert-nnf-ext.simps(4) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(3))
  have  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ ) and

```

```

       $\beta$ -welldef: intervals-welldef (to-mltl  $\beta$ )
    using Suc(3) unfolding And-mltl-ext by simp-all
  have  $\alpha$ -composition: is-composition-MLTL  $\alpha$  and
       $\beta$ -composition: is-composition-MLTL  $\beta$ 
    using Suc(4) unfolding And-mltl-ext is-composition-MLTL.simps
    by simp-all
  have  $\alpha$ -ih: LP-mltl-aux  $\alpha$   $k \neq []$ 
    using Suc(1)[OF  $\alpha$ -nnf  $\alpha$ -welldef  $\alpha$ -composition] by simp
  have  $\beta$ -ih: LP-mltl-aux  $\beta$   $k \neq []$ 
    using Suc(1)[OF  $\beta$ -nnf  $\beta$ -welldef  $\beta$ -composition] by simp
  show ?thesis
    unfolding And-mltl-ext LP-mltl-aux.simps And-mltl-list.simps
    using pairs.simps(2)  $\alpha$ -ih  $\beta$ -ih
    by (metis (no-types, lifting)  $\alpha$ -nnf  $\beta$ -nnf append-is-Nil-conv convert-nnf-ext-convert-nnf-ext
list.map-disc-iff pairs.elims)
  next
  case (Or-mltl-ext  $\alpha$   $\beta$ )
  have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
    using Suc(2) unfolding Or-mltl-ext
  by (metis convert-nnf-ext.simps(5) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(4))

  have  $\beta$ -nnf:  $\exists \varphi$ -init.  $\beta = \text{convert-nnf-ext } \varphi$ -init
    using Suc(2) unfolding Or-mltl-ext
  by (metis convert-nnf-ext.simps(5) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(4))
  have  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ ) and
       $\beta$ -welldef: intervals-welldef (to-mltl  $\beta$ )
    using Suc(3) unfolding Or-mltl-ext by simp-all
  have  $\alpha$ -composition: is-composition-MLTL  $\alpha$  and
       $\beta$ -composition: is-composition-MLTL  $\beta$ 
    using Suc(4) unfolding Or-mltl-ext is-composition-MLTL.simps
    by simp-all
  have  $\alpha$ -ih: LP-mltl-aux  $\alpha$   $k \neq []$ 
    using Suc(1)[OF  $\alpha$ -nnf  $\alpha$ -welldef  $\alpha$ -composition] by simp
  have  $\beta$ -ih: LP-mltl-aux  $\beta$   $k \neq []$ 
    using Suc(1)[OF  $\beta$ -nnf  $\beta$ -welldef  $\beta$ -composition] by simp
  then show ?thesis
    unfolding Or-mltl-ext LP-mltl-aux.simps And-mltl-list.simps
    by (metis (no-types, lifting)  $\alpha$ -ih  $\alpha$ -nnf concat.simps(1) concat-eq-append-conv
convert-nnf-ext-convert-nnf-ext list.map-disc-iff not-Cons-self2 pairs.elims)
  next
  case (Future-mltl-ext a b L  $\alpha$ )
  have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
    using Suc(2) unfolding Future-mltl-ext
  by (metis convert-nnf-ext.simps(6) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(5))

  have  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ )
    using Suc(3) unfolding Future-mltl-ext by simp-all
  have  $\alpha$ -composition: is-composition-MLTL  $\alpha$ 
    using Suc(4) unfolding Future-mltl-ext is-composition-MLTL.simps

```

```

    by simp-all
  have  $\alpha$ -ih: LP-mltl-aux  $\alpha$   $k \neq []$ 
    using Suc(1)[OF  $\alpha$ -nnf  $\alpha$ -welldef  $\alpha$ -composition] by simp
  then show ?thesis
    unfolding Future-mltl-ext LP-mltl-aux.simps And-mltl-list.simps
    by (metis (no-types, lifting) Future-mltl-list.elims  $\alpha$ -nnf append-is-Nil-conv
convert-nnf-ext-convert-nnf-ext map-is-Nil-conv)
  next
  case (Global-mltl-ext a b L  $\alpha$ )
  have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
    using Suc(2) unfolding Global-mltl-ext
  by (metis convert-nnf-ext.simps(7) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(6))

  then have  $\alpha$ -convert: convert-nnf-ext  $\alpha = \alpha$ 
    using convert-nnf-ext-convert-nnf-ext by metis
  have  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ )
    using Suc(3) unfolding Global-mltl-ext by simp-all
  have  $\alpha$ -composition: is-composition-MLTL  $\alpha$ 
    using Suc(4) unfolding Global-mltl-ext is-composition-MLTL.simps
    by simp-all
  have  $\alpha$ -ih: LP-mltl-aux  $\alpha$   $k \neq []$ 
    using Suc(1)[OF  $\alpha$ -nnf  $\alpha$ -welldef  $\alpha$ -composition] by simp
  let ?D = LP-mltl-aux  $\alpha$   $k$ 
  {
    assume *: length ?D  $\leq 1$ 
    then have ?thesis unfolding Global-mltl-ext LP-mltl-aux.simps
      using  $\alpha$ -ih  $\alpha$ -convert by simp
  } moreover {
    assume *: length ?D  $> 1$ 
    have D-is: LP-mltl-aux  $\varphi$  (Suc  $k$ ) = Global-mltl-decomp ?D a (b - a) L
      unfolding Global-mltl-ext LP-mltl-aux.simps
      using *  $\alpha$ -convert by auto
    have ?thesis unfolding D-is
      using Global-mltl-decomp-nonempty  $\alpha$ -ih by blast
  }
  ultimately show ?thesis by linarith
  next
  case (Until-mltl-ext  $\alpha$  a b L  $\beta$ )
  have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
    using Suc(2) unfolding Until-mltl-ext
  by (metis convert-nnf-ext.simps(8) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(7))

  have  $\beta$ -nnf:  $\exists \varphi$ -init.  $\beta = \text{convert-nnf-ext } \varphi$ -init
    using Suc(2) unfolding Until-mltl-ext
  by (metis convert-nnf-ext.simps(8) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(7))
  have  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ ) and
     $\beta$ -welldef: intervals-welldef (to-mltl  $\beta$ ) and
    a-leq-b:  $a \leq b$ 
  using Suc(3) unfolding Until-mltl-ext by simp-all

```

```

have  $\alpha$ -composition: is-composition-MLTL  $\alpha$  and
   $\beta$ -composition: is-composition-MLTL  $\beta$  and
  L-composition: is-composition  $(b-a+1)$  L
using Suc(4) unfolding Until-mltl-ext is-composition-MLTL.simps
by simp-all
have  $\alpha$ -ih: LP-mltl-aux  $\alpha$   $k \neq []$ 
using Suc(1)[OF  $\alpha$ -nnf  $\alpha$ -welldef  $\alpha$ -composition] by simp
have  $\beta$ -ih: LP-mltl-aux  $\beta$   $k \neq []$ 
using Suc(1)[OF  $\beta$ -nnf  $\beta$ -welldef  $\beta$ -composition] by simp
show ?thesis unfolding Until-mltl-ext LP-mltl-aux.simps
using  $\alpha$ -ih  $\beta$ -ih
by (metis (no-types, lifting) Until-mltl-list.elims  $\beta$ -nnf append-is-Nil-conv
convert-nnf-ext-convert-nnf-ext map-is-Nil-conv)
next
case (Release-mltl-ext  $\alpha$  a b L  $\beta$ )
show ?thesis unfolding LP-mltl-aux.simps Release-mltl-ext
by (meson append-is-Nil-conv not-Cons-self2)
qed
qed

```

8.2 Union Theorem

```

Forward Direction lemma exist-first:
  fixes lb  $i::nat$ 
  assumes lowerbound:  $lb \leq i$  and iprop:  $(P\ i)$ 
  shows  $\exists j. (lb \leq j \wedge j \leq i \wedge (P\ j))$ 
     $\wedge (\forall l. (lb \leq l \wedge l < j) \longrightarrow \neg(P\ l))$ 
  using lowerbound iprop
proof(induct  $i-lb$  arbitrary:  $i$  rule: less-induct)
case less
  {
    assume *:  $\forall l \geq lb. l < i \longrightarrow \neg(P\ l)$ 
    then have ?case
      using less by blast
  } moreover {
    assume *:  $\exists i' \geq lb. i' < i \wedge (P\ i')$ 
    then obtain  $i'$  where  $lb \leq i' \wedge i' < i \wedge P\ i'$ 
      by blast
    then have ?case
      using less.hyps(1)[of  $i'$ ] by fastforce
  }
  ultimately show ?case by blast
qed

```

```

lemma exist-bound-split:
  fixes a m b::nat
  assumes  $a \leq b$ 
  assumes  $\exists i. a \leq i \wedge i \leq b \wedge P\ i$ 

```

shows $(\exists i. a \leq i \wedge i \leq m-1 \wedge P i) \vee$
 $(\exists i. m \leq i \wedge i \leq b \wedge P i \wedge \neg(\exists j. a \leq j \wedge j < m \wedge P j))$
 using *assms* by *fastforce*

lemma *Global-mltl-ext-obtain:*

fixes $D::'a$ *mltl-ext list* **and** $\pi::'a$ *set list*

and $\alpha::'a$ *mltl-ext* **and** $a b k::nat$

assumes *a-leq-b*: $a \leq b$

assumes *length- π* : $length \pi \geq b + wpd_mltl (to_mltl \alpha)$

assumes *semantics*: *semantics-mltl-ext* π (*Global-mltl-ext* $a b L \alpha$)

assumes *ih*: $\bigwedge trace. semantics_mltl_ext \text{ trace } \alpha \implies$

$wpd_mltl (to_mltl \alpha) \leq length \text{ trace} \implies$

$\exists x \in set D. semantics_mltl_ext \text{ trace } x$

shows $\exists X. (length X = b - a + 1) \wedge$

$(\forall i < length X. (X!i \in set D) \wedge semantics_mltl_ext (drop (a+i) \pi) (X!i))$

proof –

have *semantics*: $\bigwedge i. a \leq i \wedge i \leq b \implies semantics_mltl_ext (drop i \pi) \alpha$

using *semantics length- π a-leq-b*

unfolding *semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps*

by (*metis add-diff-cancel-left' wpd-geq-one diff-add-zero le-less-Suc-eq le-trans*

less-add-Suc1 not-one-le-zero)

have *ih*: $\exists x \in set D. semantics_mltl_ext (drop i \pi) x$

if *i-bound*: $a \leq i \wedge i \leq b$ **for** i

proof –

have *cond1*: *semantics-mltl-ext* $(drop i \pi) \alpha$

using *semantics[of i] i-bound* **by** *blast*

have *cond2*: $wpd_mltl (to_mltl \alpha) \leq length (drop i \pi)$

using *length- π a-leq-b i-bound* **by** *auto*

show *?thesis*

using *ih[OF cond1 cond2]* **by** *blast*

qed

show *?thesis* **using** *ih a-leq-b*

proof(*induct b-a arbitrary: a b*)

case 0

then **have** *aeqb*: $a = b$ **by** *simp*

then **obtain** x **where** *semantics-x*: *semantics-mltl-ext* $(drop a \pi) x$

and *x-in*: $x \in set D$

using $0(2)[of a]$ **by** *blast*

let $?X = [x]$

have *length-X*: $length ?X = b - a + 1$ **using** *aeqb* **by** *simp*

have $?X ! i \in set D \wedge semantics_mltl_ext (drop (a+i) \pi) (?X ! i)$

if *i-bound*: $i < length ?X$ **for** i

using *semantics-x that x-in* **by** *force*

then **show** *?case* **using** *length-X* **by** *blast*

next

case (*Suc n*)

then **have** *n-eq*: $n = b - 1 - a$ **by** *simp*

have $\exists X. length X = b - 1 - a + 1 \wedge$

$(\forall i < length X.$

$X ! i \in \text{set } D \wedge \text{semantics-mltl-ext } (\text{drop } (a + i) \pi) (X ! i)$
using $\text{Suc}(1)[\text{OF } n\text{-eq}]$ **unfolding** Bex-def
using $\text{Suc.hyps}(2)$ $\text{Suc.prem}(1)$ diff-diff-left diff-le-self plus-1-eq-Suc **by**
fastforce
then obtain X **where** $\text{length-}X$: $\text{length } X = b - a$ **and**
 $X\text{-prop}$: $\forall i < \text{length } X. X ! i \in \text{set } D \wedge \text{semantics-mltl-ext } (\text{drop } (a + i) \pi) (X ! i)$
by $(\text{metis } \text{Suc.hyps}(2) \text{Suc-eq-plus1 } n\text{-eq})$
obtain x **where** $x\text{-in}$: $x \in \text{set } D$
and $\text{semantics-}x$: $\text{semantics-mltl-ext } (\text{drop } b \pi) x$
using $\text{Suc}(3)[\text{of } b]$ **unfolding** Bex-def **using** $\text{Suc}(4)$ **by** *blast*
let $?L = X @ [x]$
have $\text{length-}L$: $\text{length } ?L = b - a + 1$
using $\text{length-}X$ **by** *simp*
have $?L ! i \in \text{set } D \wedge \text{semantics-mltl-ext } (\text{drop } (a + i) \pi) (?L ! i)$
if $i\text{-bound}$: $i < \text{length } ?L$ **for** i
proof –
{
assume $*$: $i < b - a$
have $?thesis$
using $X\text{-prop}$ $\text{length-}X$
by $(\text{metis } * \text{nth-append})$
} **moreover** {
assume $*$: $i = b - a$
then have $x\text{-is}$: $(X @ [x]) ! i = x$
using $\text{length-}L$ **by** $(\text{metis } \text{length-}X \text{nth-append-length})$
have $?thesis$ **unfolding** $x\text{-is}$
using $x\text{-in}$ Suc $\text{semantics-}x$ **unfolding** $*$ **by** *simp*
}
ultimately show $?thesis$ **using** $i\text{-bound}$ $\text{length-}L$ **by** *fastforce*
qed
then show $?case$ **using** $\text{length-}L$ **by** *blast*
qed
qed

lemma *Release-semantics-split*:

assumes $(\forall i. a \leq i \wedge i \leq b \longrightarrow \text{semantics-mltl } (\text{drop } i \pi) (\text{to-mltl } \beta)) \vee$
 $(\exists j \geq a. j \leq b - 1 \wedge \text{semantics-mltl } (\text{drop } j \pi) (\text{to-mltl } \alpha) \wedge$
 $(\forall k. a \leq k \wedge k \leq j \longrightarrow$
 $\text{semantics-mltl } (\text{drop } k \pi) (\text{to-mltl } \beta)))$
shows $((\forall i. a \leq i \wedge i \leq b \longrightarrow \text{semantics-mltl } (\text{drop } i \pi) (\text{to-mltl } \beta))$
 $\wedge (\forall i. a \leq i \wedge i \leq b \longrightarrow \text{semantics-mltl } (\text{drop } i \pi) (\text{Not}_m (\text{to-mltl } \alpha))))$
 $\vee (\exists j \geq a. j \leq b \wedge$
 $\text{semantics-mltl } (\text{drop } j \pi) (\text{to-mltl } \alpha) \wedge$
 $(\forall k. a \leq k \wedge k \leq j \longrightarrow$
 $\text{semantics-mltl } (\text{drop } k \pi) (\text{to-mltl } \beta)))$

proof –

{**assume** $*$: $(\forall i. a \leq i \wedge i \leq b \longrightarrow \text{semantics-mltl } (\text{drop } i \pi) (\text{to-mltl } \beta)) \wedge$

```

       $\neg(\exists j \geq a. j \leq b - 1 \wedge \text{semantics-mltl } (\text{drop } j \pi) (\text{to-mltl } \alpha) \wedge$ 
       $(\forall k. a \leq k \wedge k \leq j \longrightarrow$ 
       $\text{semantics-mltl } (\text{drop } k \pi) (\text{to-mltl } \beta)))$ 
    then have semantics:  $\forall j. a \leq j \wedge j \leq b-1 \longrightarrow \neg \text{semantics-mltl } (\text{drop } j \pi)$ 
     $(\text{to-mltl } \alpha) \vee$ 
       $\neg(\forall k. a \leq k \wedge k \leq j \longrightarrow$ 
       $\text{semantics-mltl } (\text{drop } k \pi) (\text{to-mltl } \beta))$ 
    by blast
    then have  $\neg \text{semantics-mltl } (\text{drop } j \pi) (\text{to-mltl } \alpha)$ 
    if j-bound:  $a \leq j \wedge j \leq b-1$  for j
    proof-
      have semantics-mltl  $(\text{drop } k \pi) (\text{to-mltl } \beta)$ 
      if k-bound:  $a \leq k \wedge k \leq j$  for k
      using k-bound j-bound * by auto
      then show ?thesis using semantics j-bound by blast
    qed
    then have ?thesis using *
      by (metis dual-order.trans semantics-mltl.simps(4))
  } moreover {
    assume  $(\forall i. a \leq i \wedge i \leq b \longrightarrow \text{semantics-mltl } (\text{drop } i \pi) (\text{to-mltl } \beta)) \wedge$ 
       $(\exists j \geq a. j \leq b - 1 \wedge \text{semantics-mltl } (\text{drop } j \pi) (\text{to-mltl } \alpha) \wedge$ 
       $(\forall k. a \leq k \wedge k \leq j \longrightarrow$ 
       $\text{semantics-mltl } (\text{drop } k \pi) (\text{to-mltl } \beta)))$ 
    then have ?thesis
      by (meson diff-le-self le-trans)
  } moreover {
    assume  $(\exists j \geq a. j \leq b - 1 \wedge \text{semantics-mltl } (\text{drop } j \pi) (\text{to-mltl } \alpha) \wedge$ 
       $(\forall k. a \leq k \wedge k \leq j \longrightarrow$ 
       $\text{semantics-mltl } (\text{drop } k \pi) (\text{to-mltl } \beta)))$ 
    then have ?thesis
      by (meson diff-le-self le-trans)
  }
  }
  ultimately show ?thesis using assms
  by blast
qed

```

theorem *LP-mltl-aux-language-union-forward*:

```

  fixes  $\varphi::'a \text{ mltl-ext}$  and  $k::\text{nat}$  and  $\pi::'a \text{ set list}$ 
  assumes intervals-welldef: intervals-welldef  $(\text{to-mltl } \varphi)$ 
  assumes is-nnf:  $\exists \varphi\text{-init}. \varphi = \text{convert-nnf-ext } \varphi\text{-init}$ 
  assumes composition: is-composition-MLTL  $\varphi$ 
  assumes D-is:  $D = \text{LP-mltl-aux } \varphi \ k$ 
  assumes semantics: semantics-mltl-ext  $\pi \ \varphi$ 
  assumes trace-length: length  $\pi \geq \text{wpd-mltl } (\text{to-mltl } \varphi)$ 
  shows  $\exists \psi \in \text{set } D. \text{semantics-mltl-ext } \pi \ \psi$ 
  using assms
proof(induct k arbitrary: \varphi D \pi)
  case 0

```



```

then show ?case by auto
next
case (Suc k)
then show ?case
proof(cases  $\varphi$ )
  case True-mltl-ext
  then show ?thesis using Suc by simp
next
  case False-mltl-ext
  then show ?thesis using Suc by simp
next
  case (Prop-mltl-ext  $x_3$ )
  then show ?thesis using Suc by simp
next
  case (Not-mltl-ext  $x_4$ )
  then have  $\exists p. x_4 = \text{Prop-mltl-ext } p$ 
    using convert-nnf-form-Not-Implies-Prop Suc(3)
  by (metis convert-nnf-ext-to-mltl-commute to-mltl.simps(4) to-mltl-prop-bijective)

then show ?thesis using Suc
  by (metis LP-mltl-aux.simps(5) ListMem-iff Not-mltl-ext elem)
next
case (And-mltl-ext  $\alpha \beta$ )
have  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ ) and
   $\beta$ -welldef: intervals-welldef (to-mltl  $\beta$ )
  using Suc(2) unfolding And-mltl-ext by simp-all
have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
  using Suc(3) unfolding And-mltl-ext
  by (metis convert-nnf-ext.simps(4) convert-nnf-ext-convert-nnf-ext mtl-ext.inject(3))
have  $\beta$ -nnf:  $\exists \varphi$ -init.  $\beta = \text{convert-nnf-ext } \varphi$ -init
  by (metis And-mltl-ext Suc.prem(2) convert-nnf-ext.simps(4) convert-nnf-ext-convert-nnf-ext
  mtl-ext.inject(3))
have  $\alpha$ -composition: is-composition-MLTL  $\alpha$  and
   $\beta$ -composition: is-composition-MLTL  $\beta$ 
  using Suc(4) unfolding And-mltl-ext is-composition-MLTL.simps
  by simp-all
have  $\alpha$ -semantics: semantics-mltl-ext  $\pi \alpha$  and
   $\beta$ -semantics: semantics-mltl-ext  $\pi \beta$ 
  using Suc(6) unfolding And-mltl-ext semantics-mltl-ext-def
  by simp-all
have  $\alpha$ -wpd: wpd-mltl (to-mltl  $\alpha$ )  $\leq$  length  $\pi$  and
   $\beta$ -wpd: wpd-mltl (to-mltl  $\beta$ )  $\leq$  length  $\pi$ 
  using Suc(7) unfolding And-mltl-ext to-mltl.simps wpd-mltl.simps
  by simp-all
have  $\alpha$ -ih:  $\exists xa \in \text{set } (LP\text{-mltl-aux } \alpha \ k). \text{ semantics-mltl-ext } \pi \ xa$ 
  using Suc(1)[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition -  $\alpha$ -semantics  $\alpha$ -wpd] by
blast
have  $\beta$ -ih:  $\exists xb \in \text{set } (LP\text{-mltl-aux } \beta \ k). \text{ semantics-mltl-ext } \pi \ xb$ 
  using Suc(1)[OF  $\beta$ -welldef  $\beta$ -nnf  $\beta$ -composition -  $\beta$ -semantics  $\beta$ -wpd] by

```

blast
then obtain xa **where** $xa\text{-in}: xa \in \text{set } (LP\text{-mttl-aux } \alpha \ k)$ **and** $xa\text{-semantics}: \text{semantics-mltl-ext } \pi \ xa$
using $\alpha\text{-ih}$ **by** *blast*
then obtain xb **where** $xb\text{-in}: xb \in \text{set } (LP\text{-mttl-aux } \beta \ k)$ **and** $xb\text{-semantics}: \text{semantics-mltl-ext } \pi \ xb$
using $\beta\text{-ih}$ **by** *blast*
have $xab\text{-in}: \text{And-mltl-ext } xa \ xb \in \text{set } D$
unfolding $Suc(5)$ *And-mltl-ext LP-mltl-aux.simps*
using $xa\text{-in } xb\text{-in}$ *And-mltl-list-member*
by $(metis \ \alpha\text{-nnf } \beta\text{-nnf } \text{convert-nnf-ext-convert-nnf-ext } \text{in-set-member})$
have $xab\text{-semantics}: \text{semantics-mltl-ext } \pi \ (\text{And-mltl-ext } xa \ xb)$
using $xa\text{-semantics } xb\text{-semantics}$ **unfolding** *semantics-mltl-ext-def*
by *simp*
show *?thesis* **using** $xab\text{-in } xab\text{-semantics}$ **by** *blast*
next
case $(Or\text{-mltl-ext } \alpha \ \beta)$
have $\alpha\text{-welldef}: \text{intervals-welldef } (to\text{-mltl } \alpha)$ **and**
 $\beta\text{-welldef}: \text{intervals-welldef } (to\text{-mltl } \beta)$
using $Suc(2)$ **unfolding** *Or-mltl-ext* **by** *simp-all*
have $\alpha\text{-nnf}: \exists \varphi\text{-init}. \alpha = \text{convert-nnf-ext } \varphi\text{-init}$
using $Suc(3)$ **unfolding** *Or-mltl-ext*
by $(metis \ \text{convert-nnf-ext.simps}(5) \ \text{convert-nnf-ext-convert-nnf-ext } \text{mltl-ext.inject}(4))$

have $\beta\text{-nnf}: \exists \varphi\text{-init}. \beta = \text{convert-nnf-ext } \varphi\text{-init}$
by $(metis \ Or\text{-mltl-ext } Suc.prem(2) \ \text{convert-nnf-ext.simps}(5) \ \text{convert-nnf-ext-convert-nnf-ext } \text{mltl-ext.inject}(4))$
have $\alpha\text{-composition}: \text{is-composition-MLTL } \alpha$ **and**
 $\beta\text{-composition}: \text{is-composition-MLTL } \beta$
using $Suc(4)$ **unfolding** *Or-mltl-ext is-composition-MLTL.simps*
by *simp-all*
have $\alpha\text{-wpd}: \text{wpd-mltl } (to\text{-mltl } \alpha) \leq \text{length } \pi$ **and**
 $\beta\text{-wpd}: \text{wpd-mltl } (to\text{-mltl } \beta) \leq \text{length } \pi$
using $Suc(7)$ **unfolding** *Or-mltl-ext to-mltl.simps wpd-mltl.simps*
by *simp-all*
have $\alpha\beta\text{-semantics}: \text{semantics-mltl-ext } \pi \ \alpha \vee \text{semantics-mltl-ext } \pi \ \beta$
using $Suc(6)$ **unfolding** *Or-mltl-ext semantics-mltl-ext-def*
by *simp*
let $?D\text{-}\alpha = LP\text{-mttl-aux } \alpha \ k$ **and** $?D\text{-}\beta = LP\text{-mttl-aux } \beta \ k$
{
assume $*$: $\text{semantics-mltl-ext } \pi \ \alpha \wedge \neg \text{semantics-mltl-ext } \pi \ \beta$
have $\alpha\text{-ih}: \exists xa \in \text{set } (LP\text{-mttl-aux } \alpha \ k). \text{semantics-mltl-ext } \pi \ xa$
using $*$ $Suc(1)$ *[OF $\alpha\text{-welldef } \alpha\text{-nnf } \alpha\text{-composition} \ - \ - \ \alpha\text{-wpd}$]* **by** *blast*
then obtain xa **where** $xa\text{-in}: xa \in \text{set } ?D\text{-}\alpha$ **and** $xa\text{-semantics}: \text{semantics-mltl-ext } \pi \ xa$
using $\alpha\text{-ih}$ **by** *blast*
let $? \psi = \text{And-mltl-ext } xa \ (\text{Not}_c \ \beta)$
have $xa\beta\text{-in}: ? \psi \in \text{set } (\text{And-mltl-list } ?D\text{-}\alpha \ [\text{Not}_c \ \beta])$
using $xa\text{-in}$ *And-mltl-list-member* **unfolding** *List.member-def*

```

    by (metis list.set-intros(1))
  then have  $\alpha\beta$ -in:  $?\psi \in \text{set } D$ 
    unfolding Suc(5) Or-mltl-ext LP-mltl-aux.simps
    using list-concat-set-union
    [of And-mltl-list  $?D$ - $\alpha$   $?D$ - $\beta$  @ And-mltl-list [Notc  $\alpha$ ]  $?D$ - $\beta$ 
      And-mltl-list (LP-mltl-aux  $\alpha$   $k$ ) [Notc  $\beta$ ]]
  by (metis UnCI  $\alpha$ -nnf  $\beta$ -nnf append-assoc convert-nnf-ext-convert-nnf-ext)
  have  $\alpha\beta$ -semantics: semantics-mltl-ext  $\pi$   $? \psi$  using *  $\alpha$ -semantics
    unfolding semantics-mltl-ext-def semantics-mltl.simps to-mltl.simps
    by simp
  have ?thesis using  $\alpha\beta$ -in  $\alpha\beta$ -semantics by blast
} moreover {
  assume *:  $\neg$ semantics-mltl-ext  $\pi$   $\alpha \wedge$  semantics-mltl-ext  $\pi$   $\beta$ 
  have  $\beta$ -ih:  $\exists xb \in \text{set } (LP\text{-mltl-aux } \beta \ k)$ . semantics-mltl-ext  $\pi$   $xb$ 
    using * Suc(1)[OF  $\beta$ -welldef  $\beta$ -nnf  $\beta$ -composition - -  $\beta$ -wpd] by blast
  then obtain  $xb$  where  $\alpha$ -in:  $xb \in \text{set } ?D$ - $\beta$  and  $\alpha$ -semantics: semantics-mltl-ext  $\pi$   $xb$ 
    using  $\beta$ -ih by blast
  let  $? \psi = \text{And-mltl-ext } (Not_c \ \alpha) \ xb$ 
  have  $\alpha xb$ -in:  $? \psi \in \text{set } (And\text{-mltl-list } [Not_c \ \alpha] \ ?D\text{-}\beta)$ 
    using  $\alpha$ -in And-mltl-list-member unfolding List.member-def
    by (metis list.set-intros(1))
  then have  $\alpha xb$ -in:  $? \psi \in \text{set } (And\text{-mltl-list } ?D\text{-}\alpha \ ?D\text{-}\beta \ @ \ And\text{-mltl-list } [Not_c \ \alpha] \ ?D\text{-}\beta)$ 
    using list-concat-set-union[of And-mltl-list  $?D$ - $\alpha$   $?D$ - $\beta$  And-mltl-list [Notc  $\alpha$ ]  $?D$ - $\beta$ ]
    by blast
  then have  $\alpha xb$ -in:  $? \psi \in \text{set } D$ 
    unfolding Suc(5) Or-mltl-ext LP-mltl-aux.simps
    using list-concat-set-union
    [of And-mltl-list  $?D$ - $\alpha$   $?D$ - $\beta$  @ And-mltl-list [Notc  $\alpha$ ]  $?D$ - $\beta$ 
      And-mltl-list (LP-mltl-aux  $\alpha$   $k$ ) [Notc  $\beta$ ]]
  by (metis UnCI  $\alpha$ -nnf  $\beta$ -nnf append-assoc convert-nnf-ext-convert-nnf-ext)
  have  $\alpha xb$ -semantics: semantics-mltl-ext  $\pi$   $? \psi$  using *  $\alpha$ -semantics
    unfolding semantics-mltl-ext-def semantics-mltl.simps to-mltl.simps
    by simp
  have ?thesis using  $\alpha xb$ -in  $\alpha xb$ -semantics by blast
} moreover {
  assume *: semantics-mltl-ext  $\pi$   $\alpha \wedge$  semantics-mltl-ext  $\pi$   $\beta$ 
  have  $\alpha$ -ih:  $\exists xa \in \text{set } (LP\text{-mltl-aux } \alpha \ k)$ . semantics-mltl-ext  $\pi$   $xa$ 
    using * Suc(1)[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition - -  $\alpha$ -wpd] by blast
  have  $\beta$ -ih:  $\exists xb \in \text{set } (LP\text{-mltl-aux } \beta \ k)$ . semantics-mltl-ext  $\pi$   $xb$ 
    using * Suc(1)[OF  $\beta$ -welldef  $\beta$ -nnf  $\beta$ -composition - -  $\beta$ -wpd] by blast
  then obtain  $xa$  where  $\alpha$ -in:  $xa \in \text{set } (LP\text{-mltl-aux } \alpha \ k)$  and  $\alpha$ -semantics: semantics-mltl-ext  $\pi$   $xa$ 
    using  $\alpha$ -ih by blast
  then obtain  $xb$  where  $\beta$ -in:  $xb \in \text{set } (LP\text{-mltl-aux } \beta \ k)$  and  $\beta$ -semantics: semantics-mltl-ext  $\pi$   $xb$ 
    using  $\beta$ -ih by blast

```

```

have xab-in: And-mltl-ext xa xb ∈ set D
  unfolding Suc(5) Or-mltl-ext LP-mltl-aux.simps
  using xa-in xb-in And-mltl-list-member list-concat-set-union
  unfolding List.member-def
  by (metis UnCI α-nnf β-nnf convert-nnf-ext-convert-nnf-ext)
have xab-antics: semantics-mltl-ext π (And-mltl-ext xa xb)
  using xa-semantics xb-semantics unfolding semantics-mltl-ext-def
  by simp
have ?thesis using xab-in xab-semantics by blast
}
ultimately show ?thesis using αβ-semantics by blast
next
case (Future-mltl-ext a b L α)
have α-welldef: intervals-welldef (to-mltl α)
  using Suc(2) unfolding Future-mltl-ext by auto
have α-nnf:  $\exists \varphi\text{-init. } \alpha = \text{convert-nnf-ext } \varphi\text{-init}$ 
  using Suc(3) unfolding Future-mltl-ext
by (metis convert-nnf-ext.simps(6) convert-nnf-ext-convert-nnf-ext mtl-ext.inject(5))
have α-composition: is-composition-MLTL α
  using Suc(4) unfolding Future-mltl-ext is-composition-MLTL.simps by blast
have α-wpd:  $b + \text{wpd-mltl} (\text{to-mltl } \alpha) \leq \text{length } \pi$ 
  using Suc(7) unfolding Future-mltl-ext to-mltl.simps wpd-mltl.simps
  by simp
have a-leq-b:  $a \leq b$  and length-π-geq-b:  $b < \text{length } \pi$  and length-π-ge-a:  $a < \text{length } \pi$ 
and antics:  $\exists i. (a \leq i \wedge i \leq b) \wedge \text{semantics-mltl} (\text{drop } i \pi) (\text{to-mltl } \alpha)$ 
  using Suc(6) α-wpd
unfolding Future-mltl-ext semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps
  using wpd-geq-one[of (to-mltl α)]
  by simp-all
have composition-L: is-composition ( $b - a + 1$ ) L
  using Suc(4) unfolding Future-mltl-ext is-composition-MLTL.simps by blast
then have s0:  $(\text{interval-times } a \text{ L} ! 0) = a$ 
  using interval-times-first by auto
have slast:  $\text{interval-times } a \text{ L} ! (\text{length } L) = b+1$ 
  using interval-times-last[OF a-leq-b composition-L] by blast
have length-L:  $\text{length } L \geq 0$ 
  using composition-L composition-length-lb by blast
let ?s = interval-times a L
let ?D-α = LP-mltl-aux α k
let ?decomp = (concat(map ( $\lambda i. \text{And-mltl-list}$ 
  [Global-mltl-ext (?s ! 0)
  (?s ! i - 1) [?s!i - ?s!0] (Notc α)]
  (Future-mltl-list ?D-α (?s ! i) (?s ! (i + 1) - 1)
  [?s ! (i + 1) - ?s ! i]))
  [ $1..<\text{length } L$ ]))
{
  assume *:  $\exists i. (a \leq i \wedge i \leq (?s!1-1)) \wedge \text{semantics-mltl} (\text{drop } i \pi) (\text{to-mltl}$ 
α)

```

```

then obtain  $i$  where  $i$ -bounds:  $a \leq i \wedge i \leq (?s!1-1)$  and
  semantics: semantics-mltl (drop  $i$   $\pi$ ) (to-mltl  $\alpha$ ) by blast
have length-s: length  $?s \geq 2$ 
  using  $i$ -bounds
  by (metis a-leq-b add-less-same-cancel2 antisym-conv3 interval-times-first in-
interval-times-length less-eq-iff-succ-less less-iff-succ-less-eq less-nat-zero-code one-add-one
slast verit-comp-simplify1 (1))
  have dropi-length: wpd-mltl (to-mltl  $\alpha$ )  $\leq$  length (drop  $i$   $\pi$ )
proof–
  have  $1 \leq$  length  $L$ 
    using length-s unfolding interval-times-def by simp
  then have interval-times  $a L ! 1 \leq$  interval-times  $a L !$  length  $L$ 
    using interval-times-diff-ge-general[OF a-leq-b composition-L, of length  $L$ 
1 ?s]
    by fastforce
  then have interval-times  $a L ! 1 - 1 \leq b$ 
    using slast by auto
  then show ?thesis
    using  $\alpha$ -wpd  $i$ -bounds by force
qed
have  $\exists x \in \text{set } (LP\text{-mltl-aux } \alpha k)$ . semantics-mltl-ext (drop  $i$   $\pi$ )  $x$ 
using Suc(1)[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition, of ?D- $\alpha$  drop  $i$   $\pi$ ] semantics
  using semantics-mltl-ext-def  $\alpha$ -wpd dropi-length by blast
then obtain  $x$  where  $x$ -in:  $x \in \text{set } (LP\text{-mltl-aux } \alpha k)$  and
   $x$ -semantics: semantics-mltl-ext (drop  $i$   $\pi$ )  $x$ 
  by blast
let ? $\psi$  = Future-mltl-ext (?s!0) (?s!1-1) [?s!1 - ?s!0]  $x$ 
have  $\psi$ -in: ? $\psi \in \text{set } (Future\text{-mltl-list } ?D\text{-}\alpha (?s!0) (?s!1-1) [?s!1 - ?s!0])$ 
  unfolding Future-mltl-list.simps using  $x$ -in by simp
  then have  $\psi$ -in: ? $\psi \in \text{set } ((Future\text{-mltl-list } ?D\text{-}\alpha (?s!0) (?s!1-1) [?s!1 -$ 
?s!0]) @
  (concat
    (map ( $\lambda i$ . And-mltl-list
      [Global-mltl-ext (?s ! 0) (?s !  $i - 1$ ) [?s! $i - ?s!0$ ] (Not $_c$   $\alpha$ )
      (Future-mltl-list ?D- $\alpha$  (?s !  $i$ ) (?s ! ( $i + 1$ ) - 1)
      [?s ! ( $i + 1$ ) - ?s !  $i$ ]))
      [1.. $\text{length } L$ ]))))
  by force
have  $\psi$ -semantics: semantics-mltl-ext  $\pi$  ? $\psi$ 
  using  $x$ -semantics unfolding s0 semantics-mltl-ext-def
  unfolding semantics-mltl.simps to-mltl.simps
  using  $i$ -bounds length- $\pi$ -geq-b length- $\pi$ -ge-a by auto
have ?thesis unfolding Suc(5) Future-mltl-ext LP-mltl-aux.simps
  using  $\psi$ -in  $\psi$ -semantics
proof –
  have convert- $nnf$ -ext  $\alpha = \alpha$ 
    by (metis (full-types)  $\alpha$ -nnf convert- $nnf$ -ext-convert- $nnf$ -ext)
  then have Future-mltl-ext (interval-times  $a L ! 0$ )
(interval-times  $a L ! 1 - 1$ ) [interval-times  $a L ! 1 -$  interval-times  $a L ! 0$ ]  $x \in$ 

```

set (*Future-mltl-list* (*LP-mltl-aux* (*convert-nnf-ext* α) k)
(interval-times $a L ! 0$) (*interval-times* $a L ! 1 - 1$)
[interval-times $a L ! 1 - interval-times$ $a L ! 0]$ @
concat (*map* ($\lambda n.$ *And-mltl-list* [*Global-mltl-ext*
(interval-times $a L ! 0$) (*interval-times* $a L ! n - 1$) [$?s!n - ?s!0$] (*Not_c* α)])
(Future-mltl-list (*LP-mltl-aux* (*convert-nnf-ext* α) k)
(interval-times $a L ! n$) (*interval-times* $a L ! (n + 1) - 1$)
[interval-times $a L ! (n + 1) - interval-times$ $a L ! n]$)) [$1..<length L$])
using ψ -*in* **by** *presburger*
then show $\exists m \in set$ (*let* $ms = LP\text{-mltl-aux}$ (*convert-nnf-ext* α) k ; $ns =$
interval-times $a L$ *in* *Future-mltl-list* ms ($ns ! 0$) ($ns ! 1 - 1$) [$ns ! 1 - ns ! 0$] @
concat (*map* ($\lambda n.$ *And-mltl-list* [*Global-mltl-ext* ($ns ! 0$) ($ns ! n - 1$) [$ns!n - ns!0$]
(Not_c α)]) (*Future-mltl-list* ms ($ns ! n$) ($ns ! (n + 1) - 1$) [$ns ! (n + 1) - ns !$
 n])) [$1..<length L$]). *semantics-mltl-ext* πm
by (*meson* ψ -*semantics*)
qed
} moreover {
assume $*$: $\exists i. ((?s!1) \leq i \wedge i \leq b) \wedge semantics\text{-mltl}$ (*drop* $i \pi$) (*to-mltl* α) \wedge
 $\neg(\exists i. (a \leq i \wedge i \leq (?s!1 - 1)) \wedge semantics\text{-mltl}$ (*drop* $i \pi$) (*to-mltl*
 α))
obtain t' **where** t' -*facts*: $((?s!1) \leq t' \wedge t' \leq b) \wedge semantics\text{-mltl}$ (*drop* $t' \pi$)
(*to-mltl* α)
using $*$ **by** *blast*
then have $\exists j. (interval\text{-times}$ $a L ! 1 \leq j \wedge j \leq t') \wedge$
semantics-mltl (*drop* $j \pi$) (*to-mltl* α) \wedge
 $(\forall l. (interval\text{-times}$ $a L ! 1 \leq l \wedge l < j) \longrightarrow$
 $\neg semantics\text{-mltl}$ (*drop* $l \pi$) (*to-mltl* α))
using *exist-first*[*of* ($?s!1$) t' $\lambda i.$ *semantics-mltl* (*drop* $i \pi$) (*to-mltl* α)]
by *simp*
then obtain t **where**
 t -*bounds*: $(interval\text{-times}$ $a L ! 1 \leq t \wedge t \leq t')$ **and**
 t -*semantics*: *semantics-mltl* (*drop* $t \pi$) (*to-mltl* α) **and**
 t -*minimal*: $(\forall l. (interval\text{-times}$ $a L ! 1 \leq l \wedge l < t) \longrightarrow$
 $\neg semantics\text{-mltl}$ (*drop* $l \pi$) (*to-mltl* α)) **by** *auto*
have *dropt-length*: *wpd-mltl* (*to-mltl* α) $\leq length$ (*drop* $t \pi$)
proof-
have $t' \leq b$
using t' -*facts* **by** *blast*
then show *?thesis*
using α -*wpd* t -*bounds* **by** *auto*
qed
have $\exists i. interval\text{-times}$ $a L ! i \leq t \wedge$
 $t \leq interval\text{-times}$ $a L ! (i + 1) - 1 \wedge 1 \leq i \wedge i < length L$
using *interval-times-obtain-aux*[*of* $a b L ?s t$]
using a -*leq-b* *composition-L* t -*bounds* t -*semantics*
using le -*trans* t' -*facts* **by** *blast*
then obtain i **where** t -*bound*: $interval\text{-times}$ $a L ! i \leq t \wedge t \leq interval\text{-times}$
 $a L ! (i + 1) - 1$
and i -*bound*: $1 \leq i \wedge i < length L$

by *blast*
have $\exists x \in \text{set } (LP\text{-mttl-aux } \alpha \ k)$. *semantics-mttl-ext* (*drop t* π) x
using *Suc(1)*[*OF* α -*welldef* α -*nnf* α -*composition*, *of* $?D\text{-}\alpha$ *drop t* π]
using *semantics-mttl-ext-def* *t-semantics* *dropt-length* **by** *blast*
then obtain x **where** $x\text{-in}$: $x \in \text{set } (LP\text{-mttl-aux } \alpha \ k)$ **and**
 $x\text{-semantics}$: *semantics-mttl-ext* (*drop t* π) x
by *blast*
let $? \psi = \text{And-mttl-ext}$
 $(\text{Global-mttl-ext } (?s ! 0) (?s ! i - 1) [?s!i - ?s!0] (\text{Not}_c \alpha))$
 $(\text{Future-mttl-ext } (?s ! i) (?s ! (i + 1) - 1) [?s ! (i + 1) - ?s ! i] x)$
have $? \psi \in \text{set } ?\text{decomp}$
proof–
have $? \psi \in \text{set } (\text{And-mttl-list}$
 $[\text{Global-mttl-ext } (?s ! 0)$
 $(?s ! i - 1) [?s!i - ?s!0] (\text{Not}_c \alpha)]$
 $(\text{Future-mttl-list } ?D\text{-}\alpha (?s ! i) (?s ! (i + 1) - 1)$
 $[?s ! (i + 1) - ?s ! i])$
using $x\text{-in}$ **unfolding** *Future-mttl-list.simps* **by** *auto*
then have $? \psi \in \text{set } ((\text{map } (\lambda i. \text{And-mttl-list}$
 $[\text{Global-mttl-ext}$
 $(\text{interval-times } a \ L ! 0)$
 $(\text{interval-times } a \ L ! i - 1) [?s!i - ?s!0] (\text{Not}_c \alpha)]$
 $(\text{Future-mttl-list } (LP\text{-mttl-aux } \alpha \ k)$
 $(\text{interval-times } a \ L ! i)$
 $(\text{interval-times } a \ L ! (i + 1) - 1)$
 $[\text{interval-times } a \ L ! (i + 1) -$
 $\text{interval-times } a \ L ! i])$
 $[1..<\text{length } L])!(i-1))$ **using** $i\text{-bound}$ **by** *auto*
then show $?thesis$
using *set-concat* $i\text{-bound}$ **by** *fastforce*
qed
then have $\psi\text{-in}$: $? \psi \in \text{set } (\text{Future-mttl-list } ?D\text{-}\alpha (?s ! 0) (?s ! 1 - 1) [?s ! 1$
 $- ?s ! 0] @$
 $\text{concat}(\text{map } (\lambda i. \text{And-mttl-list}$
 $[\text{Global-mttl-ext } (?s ! 0)$
 $(?s ! i - 1) [?s!i - ?s!0] (\text{Not}_c \alpha)]$
 $(\text{Future-mttl-list } ?D\text{-}\alpha (?s ! i) (?s ! (i + 1) - 1)$
 $[?s ! (i + 1) - ?s ! i])$
 $[1..<\text{length } L]))$
by *simp*
have $\psi\text{-semantics}$: *semantics-mttl-ext* π $? \psi$
proof–
have *bound*: $\text{interval-times } a \ L ! 0 \leq \text{interval-times } a \ L ! i - 1$
using *interval-times-diff-ge-general*[*OF* $a\text{-leq-}b$ *composition-L*, *of* $- 0]$
 $\text{length-L } i\text{-bound}$
by (*simp* *add*: *add-le-imp-le-diff* *less-iff-succ-less-eq*)
have *not-semantics*: $\neg \text{semantics-mttl } (\text{drop } ia \ \pi) (\text{to-mttl } \alpha)$
if $ia\text{-bound}$: $(\text{interval-times } a \ L ! 0 \leq ia \wedge ia \leq \text{interval-times } a \ L ! i -$
 $1)$ **for** ia

```

proof-
  {
    assume ia-location:  $ia \leq \text{interval-times } a \ L ! \ 1 - 1$ 
    have ?thesis using * ia-bound
      using ia-location s0 by auto
  } moreover {
    assume ia-location:  $ia > \text{interval-times } a \ L ! \ 1 - 1$ 
    have  $\text{interval-times } a \ L ! \ i - 1 < \text{interval-times } a \ L ! \ i$ 
      using interval-times-diff-ge[OF a-leq-b composition-L, of i-1 ?s]
      using i-bound by fastforce
    then have  $ia < t$ 
      using t-bound ia-bound by auto
    then have ia-cond:  $\text{interval-times } a \ L ! \ 1 \leq ia \wedge ia < t$ 
      using ia-location by simp
    then have ?thesis using t-minimal by blast
  }
  ultimately show ?thesis by linarith
qed
then have global-not: semantics-mltl-ext  $\pi$ 
  (Global-mltl-ext ( $\text{interval-times } a \ L ! \ 0$ ) ( $\text{interval-times } a \ L ! \ i - 1$ ) [?s!i
- ?s!0] (Notc  $\alpha$ ))
  unfolding semantics-mltl-ext-def semantics-mltl.simps to-mltl.simps
  using bound not-semantics by blast
  have future: semantics-mltl-ext  $\pi$  (Future-mltl-ext ( $\text{interval-times } a \ L ! \ i$ )
( $\text{interval-times } a \ L ! \ (i + 1) - 1$ ) [ $\text{interval-times } a \ L ! \ (i + 1) - \text{interval-times}$ 
 $a \ L ! \ i$ ] x)
  proof-
    have  $\text{interval-times } a \ L ! \ i \leq b$ 
      using interval-times-diff-ge-general[OF a-leq-b composition-L, of length
 $L \ i \ ?s$ ]
    unfolding slast using i-bound by auto
    then have trace-length:  $\text{interval-times } a \ L ! \ i < \text{length } \pi$ 
      using length- $\pi$ -geq-b by auto
    have semantics:  $(\exists ia. (\text{interval-times } a \ L ! \ i \leq ia \wedge$ 
 $ia \leq \text{interval-times } a \ L ! \ (i + 1) - 1) \wedge$ 
semantics-mltl (drop ia  $\pi$ ) (to-mltl x))
      using x-semantics t-bound semantics-mltl-ext-def
      by auto
    have  $\text{interval-times } a \ L ! \ i \leq \text{interval-times } a \ L ! \ (i + 1) - 1$ 
      using interval-times-diff-ge[OF a-leq-b composition-L, of i ?s]
      using i-bound by simp
    then show ?thesis unfolding semantics-mltl-ext-def semantics-mltl.simps
to-mltl.simps
      using trace-length semantics by blast
  qed
  show ?thesis using global-not future
  unfolding semantics-mltl-ext-def semantics-mltl.simps by simp
qed
have ?thesis

```



```

unfolding Suc(5) Future-mltl-ext LP-mltl-aux.simps
using  $\psi$ -in  $\psi$ -semantics
proof –
  have convert-nnf-ext  $\alpha = \alpha$ 
    by (metis  $\alpha$ -nnf convert-nnf-ext-convert-nnf-ext)
  then have And-mltl-ext (Global-mltl-ext (interval-times a L ! 0) (interval-times
a L ! i – 1) [?s!i – ?s!0] (Notc  $\alpha$ ))
(Future-mltl-ext (interval-times a L ! i) (interval-times a L ! (i + 1) – 1)
[interval-times a L ! (i + 1) – interval-times a L ! i] x)  $\in$ 
set (Future-mltl-list (LP-mltl-aux (convert-nnf-ext  $\alpha$ ) k) (interval-times a L ! 0)
(interval-times a L ! 1 – 1)
[interval-times a L ! 1 – interval-times a L ! 0])
@ concat (map ( $\lambda n$ . And-mltl-list [Global-mltl-ext (interval-times a L ! 0) (interval-times
a L ! n – 1) [?s!n – ?s!0] (Notc  $\alpha$ )]
(Future-mltl-list (LP-mltl-aux (convert-nnf-ext  $\alpha$ ) k) (interval-times a L ! n) (interval-times
a L ! (n + 1) – 1) [interval-times a L ! (n + 1) – interval-times a L ! n]))
[1.. $\text{length } L$ ]))
using  $\psi$ -in by presburger
  then show  $\exists m \in \text{set}$  (let ms = LP-mltl-aux (convert-nnf-ext  $\alpha$ ) k;
ns = interval-times a L in Future-mltl-list ms (ns ! 0) (ns ! 1 – 1)
[ns ! 1 – ns ! 0] @ concat (map ( $\lambda n$ . And-mltl-list
[Global-mltl-ext (ns ! 0) (ns ! n – 1) [ns!n – ns!0] (Notc  $\alpha$ )] (Future-mltl-list ms
(ns ! n) (ns ! (n + 1) – 1) [ns ! (n + 1) – ns ! n])) [1.. $\text{length } L$ ])). semantics-mltl-ext  $\pi$  m
by (meson  $\psi$ -semantics)
qed
}
ultimately show ?thesis using semantics by force
next
case (Global-mltl-ext a b L  $\alpha$ )
have  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ )
using Suc(2) unfolding Global-mltl-ext by auto
have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
using Suc(3) unfolding Global-mltl-ext
by (metis convert-nnf-ext.simps(7) convert-nnf-ext-convert-nnf-ext mtl-ext.inject(6))
have  $\alpha$ -composition: is-composition-MLTL  $\alpha$ 
using Suc(4) unfolding Global-mltl-ext is-composition-MLTL.simps by blast
have  $\alpha$ -wpd: b + wpd-mltl (to-mltl  $\alpha$ )  $\leq$  length  $\pi$ 
using Suc(7) unfolding Global-mltl-ext to-mltl.simps wpd-mltl.simps
by simp
have a-leq-b: a  $\leq$  b
using Suc(6)  $\alpha$ -wpd unfolding Global-mltl-ext semantics-mltl-ext-def to-mltl.simps
semantics-mltl.simps
by blast
have length- $\pi$ -geq-b: b  $<$  length  $\pi$ 
and semantics:  $\forall i$ . a  $\leq$  i  $\wedge$  i  $\leq$  b  $\longrightarrow$  semantics-mltl (drop i  $\pi$ ) (to-mltl  $\alpha$ )
using Suc(6)  $\alpha$ -wpd unfolding Global-mltl-ext semantics-mltl-ext-def to-mltl.simps
semantics-mltl.simps
using wpd-geq-one[of (to-mltl  $\alpha$ )] by auto

```

```

let ?D- $\alpha$  = LP-mltl-aux  $\alpha$  k
{
  assume *: length ?D- $\alpha$   $\leq$  1
  let ? $\psi$  = Global-mltl-ext a b L  $\alpha$ 
  have semantics: semantics-mltl  $\pi$  (to-mltl ? $\psi$ )
    using Suc(6) unfolding Global-mltl-ext semantics-mltl-ext-def
    by blast
  have  $\psi$ -in: ? $\psi$   $\in$  set D using Suc(5) *
    unfolding Global-mltl-ext LP-mltl-aux.simps
    by (metis (full-types)  $\alpha$ -nnf convert- $\text{nnf-ext}$ -convert- $\text{nnf-ext}$  list.set-intros(1))

  have ?thesis
    using semantics  $\psi$ -in Global-mltl-ext Suc.prem(5) by auto
} moreover {
  assume *: length ?D- $\alpha$   $>$  1
  then have D-is: D = Global-mltl-decomp ?D- $\alpha$  a (b - a) L
    using Suc(5) * unfolding Global-mltl-ext LP-mltl-aux.simps
    by (metis (full-types)  $\alpha$ -nnf convert- $\text{nnf-ext}$ -convert- $\text{nnf-ext}$  leD)
  have semantics-global: semantics-mltl-ext  $\pi$  (Global-mltl-ext a b L  $\alpha$ )
    using Suc(6) unfolding Global-mltl-ext by blast
  have length- $\pi$ : length  $\pi$   $\geq$  b + wpd-mltl (to-mltl  $\alpha$ )
    using Suc(6)  $\alpha$ -wpd unfolding Global-mltl-ext semantics-mltl-ext-def
    to-mltl.simps semantics-mltl.simps
    using wpd-geq-one[of (to-mltl  $\alpha$ )] by blast
  have ih:  $\bigwedge$  trace. semantics-mltl-ext trace  $\alpha$   $\implies$ 
    wpd-mltl (to-mltl  $\alpha$ )  $\leq$  length trace  $\implies$ 
     $\exists a \in \text{set } (LP\text{-mltl-aux } \alpha \text{ } k). \text{ semantics-mltl-ext trace } a$ 
    using Suc(1)[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition, of ?D- $\alpha$ ] by blast
  have  $\exists X. \text{ length } X = b - a + 1 \wedge$ 
    ( $\forall i < \text{length } X. X ! i \in \text{set } (LP\text{-mltl-aux } \alpha \text{ } k) \wedge$ 
    semantics-mltl-ext (drop (a+i)  $\pi$ ) (X ! i))
    using Global-mltl-ext-obtain[OF a-leq-b length- $\pi$  semantics-global ih]
    by blast
  then obtain Y where length-Y: length Y = b - a + 1
    and Y-prop:  $\forall i < \text{length } Y. Y ! i \in \text{set } ?D\text{-}\alpha \wedge$ 
    semantics-mltl-ext (drop (a+i)  $\pi$ ) (Y ! i)
    by blast
  let ?X = map ( $\lambda i. \text{ Global-mltl-ext } (a+i) (a+i) [1] (Y ! i)$ ) [0.. $\text{length } Y$ ]
  let ? $\psi$  = Ands-mltl-ext ?X
  have cond1: ? $\psi$  = ? $\psi$  by auto
  have length-X: length ?X = b - a + 1
    using length-Y by simp
  have cond2:  $\forall i < \text{length } ?X.$ 
     $\exists y \in \text{set } ?D\text{-}\alpha. ?X ! i = \text{ Global-mltl-ext } (a + i) (a + i) [1] y$ 
    using Y-prop by simp
  have  $\psi$ -in: ? $\psi$   $\in$  set D
    using in-Global-mltl-decomp-exact-converse[OF * cond1 cond2 length-X]
    unfolding D-is by blast
  have  $\psi$ -semantics: semantics-mltl-ext  $\pi$  ? $\psi$ 

```

```

proof–
  have cond1: length ?X ≥ 1 using length-X by simp
  have semantics-mltl-ext  $\pi$  (?X!i)
    if i-bound: i < length ?X for i
  proof–
    have Xi-is: ?X!i = Global-mltl-ext (a + i) (a + i) [1] (Y ! i)
      using i-bound by auto
    show ?thesis unfolding Xi-is
      using Y-prop i-bound unfolding semantics-mltl-ext-def
      unfolding semantics-mltl.simps by auto
  qed
  then have ( $\forall x \in \text{set } ?X. \text{ semantics-mltl-ext } \pi x$ )
    by auto
  then show ?thesis
    using Ands-mltl-semantics[of ?X  $\pi$ , OF cond1] by blast
  qed
  have ?thesis using D-is  $\psi$ -in  $\psi$ -semantics by blast
}
ultimately show ?thesis by linarith
next
case (Until-mltl-ext  $\alpha$  a b L  $\beta$ )
have  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ )
and  $\beta$ -welldef: intervals-welldef (to-mltl  $\beta$ )
  using Suc(2) unfolding Until-mltl-ext by simp-all
have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha$  = convert-nnf-ext  $\varphi$ -init
  using Suc(3) unfolding Until-mltl-ext
by (metis convert-nnf-ext.simps(8) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(7))
have  $\beta$ -nnf:  $\exists \varphi$ -init.  $\beta$  = convert-nnf-ext  $\varphi$ -init
  using Suc(3) unfolding Until-mltl-ext
by (metis convert-nnf-ext.simps(8) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(7))
have  $\alpha$ -convert: convert-nnf-ext  $\alpha$  =  $\alpha$ 
  using  $\alpha$ -nnf convert-nnf-ext-convert-nnf-ext by metis
have  $\beta$ -convert: convert-nnf-ext  $\beta$  =  $\beta$ 
  using  $\beta$ -nnf convert-nnf-ext-convert-nnf-ext by metis
have  $\alpha$ -composition: is-composition-MLTL  $\alpha$ 
and  $\beta$ -composition: is-composition-MLTL  $\beta$ 
and L-composition: is-composition (b – a + 1) L
  using Suc(4) unfolding Until-mltl-ext is-composition-MLTL.simps
  by simp-all
have  $\alpha$ -wpd: b + wpd-mltl (to-mltl  $\alpha$ ) – 1 ≤ length  $\pi$ 
and  $\beta$ -wpd: b + wpd-mltl (to-mltl  $\beta$ ) ≤ length  $\pi$ 
  using Suc(7) unfolding Until-mltl-ext to-mltl.simps wpd-mltl.simps
  by simp-all
have a-leq-b: a ≤ b and length- $\pi$ -ge-b: b < length  $\pi$ 
and semantics: ( $\exists i. (a \leq i \wedge i \leq b) \wedge$ 
  semantics-mltl (drop i  $\pi$ ) (to-mltl  $\beta$ )  $\wedge$ 
  ( $\forall j. a \leq j \wedge j < i \longrightarrow$ 
  semantics-mltl (drop j  $\pi$ ) (to-mltl  $\alpha$ )))
using Suc(6)  $\alpha$ -wpd unfolding Until-mltl-ext semantics-mltl-ext-def to-mltl.simps

```

```

semantics-mltl.simps
  using wpd-geq-one[of to-mltl  $\beta$ ]  $\beta$ -wpd
  by simp-all
  let  $?D\text{-}\beta = LP\text{-}mltl\text{-}aux\ \beta\ k$ 
  let  $?s = interval\text{-}times\ a\ L$ 
  have sfirst:  $?s!0 = a$ 
    using interval-times-first by auto
  have slast:  $?s!(length\ L) = b+1$ 
    using interval-times-last[OF a-leq-b L-composition] by auto
  have length-L:  $length\ L \geq 1$ 
    using composition-length-lb[OF L-composition] by linarith
  have s-second-lb:  $a \leq interval\text{-}times\ a\ L\ !\ 1 - 1$ 
    using sfirst interval-times-diff-ge[OF a-leq-b L-composition, of 0 ?s]
    using length-L by force
  have s-second-ub:  $interval\text{-}times\ a\ L\ !\ 1 - 1 \leq b$ 
    using slast length-L
    using interval-times-diff-ge-general[OF a-leq-b L-composition, of length L 1
?s]
    by force
  let ?front = (Until-mltl-list  $\alpha\ ?D\text{-}\beta\ (?s!\ 0)\ (?s!\ 1 - 1)\ [?s!\ 1 - ?s!\ 0]$ )
  let ?back = (concat (map ( $\lambda i.$  And-mltl-list
    [Global-mltl-ext
      ( $?s!\ 0)\ (?s!\ i - 1)\ [?s!\ i - ?s!\ 0]$ ) (And-mltl-ext  $\alpha\ (Not_c$ 
 $\beta)$ ))
    (Until-mltl-list  $\alpha\ ?D\text{-}\beta\ (?s!\ i)\ (?s!\ (i + 1) - 1)$ 
    [ $?s!\ (i + 1) - ?s!\ i$ ]) [1..<length L]))
  have D-union:  $set\ D = (set\ ?front) \cup (set\ ?back)$ 
    unfolding Suc(5) Until-mltl-ext LP-mltl-aux.simps
    using  $\alpha$ -convert  $\beta$ -convert list-concat-set-union by metis
  let  $?P = \lambda i.$  semantics-mltl (drop  $i\ \pi$ ) (to-mltl  $\beta$ )  $\wedge$ 
    ( $\forall j.$   $a \leq j \wedge j < i \longrightarrow semantics\text{-}mltl\ (drop\ j\ \pi)\ (to\text{-}mltl\ \alpha)$ )
  {
    assume *:  $\exists i.$  ( $a \leq i$ )  $\wedge$  ( $i \leq (?s!1) - 1$ )  $\wedge$   $?P\ i$ 
    then obtain i where i-bound: ( $a \leq i \wedge i \leq (?s!1) - 1$ ) and
      semantics: semantics-mltl (drop  $i\ \pi$ ) (to-mltl  $\beta$ )  $\wedge$ 
      ( $\forall j.$   $a \leq j \wedge j < i \longrightarrow semantics\text{-}mltl\ (drop\ j\ \pi)\ (to\text{-}mltl\ \alpha)$ )
    by blast
    have semantics-dropi: semantics-mltl-ext (drop  $i\ \pi$ )  $\beta$ 
      using semantics unfolding semantics-mltl-ext-def by blast
    have length-dropi: wpd-mltl (to-mltl  $\beta$ )  $\leq length\ (drop\ i\ \pi)$ 
      using  $\beta$ -wpd length- $\pi$ -ge-b i-bound a-leq-b s-second-ub by auto
    obtain x where x-semantics: semantics-mltl-ext (drop  $i\ \pi$ )  $x$ 
      and x-in:  $x \in set\ ?D\text{-}\beta$ 
    using Suc(1)[OF  $\beta$ -welldef  $\beta$ -nnf  $\beta$ -composition - semantics-dropi length-dropi,
of ?D- $\beta$ ]
    by blast
    let  $? $\psi$  = (Until-mltl-ext$   $\alpha\ a\ ((?s!1) - 1)\ [(?s!1) - a]$   $x$ )
    have  $\psi$ -semantics: semantics-mltl-ext  $\pi\ ?\psi$ 
      using semantics length- $\pi$ -ge-b a-leq-b i-bound x-semantics

```

```

    unfolding semantics-mltl-ext-def semantics-mltl.simps to-mltl.simps
  by auto
have ?ψ ∈ set ?front
  using x-in unfolding Until-mltl-list.simps sfirst by auto
then have ψ-in: ?ψ ∈ set D
  unfolding D-union by blast
have ?thesis
  using ψ-semantics ψ-in by blast
} moreover {
  assume *: ∃ i. ((?s!1) ≤ i) ∧ (i ≤ b) ∧ ?P i ∧
    ¬(∃ j. a ≤ j ∧ j < (?s!1) ∧ ?P j)
  then obtain t' where t'-bound: ((?s!1) ≤ t') ∧ (t' ≤ b) and
    semantics: ?P t' and not-semantics: ¬(∃ j. a ≤ j ∧ j < (?s!1) ∧ ?P j)
  by blast
  have ∃ j ≥ interval-times a L ! 1. j ≤ t' ∧
    ?P j ∧ (∀ l. interval-times a L ! 1 ≤ l ∧ l < j → ¬ ?P l)
  proof-
    have cond1: interval-times a L ! 1 ≤ t'
      using t'-bound by auto
    show ?thesis
      using exist-first[of ?s!1 t' ?P, OF cond1 semantics] by blast
  qed
  then obtain t where
    t-bound: interval-times a L ! 1 ≤ t ∧ t ≤ t' and
    t-semantics: ?P t and
    t-minimal: ∀ l. interval-times a L ! 1 ≤ l ∧ l < t → ¬ ?P l
  by blast
  have ∃ i. interval-times a L ! i ≤ t ∧
    t ≤ interval-times a L ! (i + 1) - 1 ∧ 1 ≤ i ∧ i < length L
    using interval-times-obtain-aux[OF a-leq-b L-composition, of ?s t]
    using t-bound t'-bound by simp
  then obtain i where t-bound: interval-times a L ! i ≤ t
    ∧ t ≤ interval-times a L ! (i + 1) - 1
    and i-bound: 1 ≤ i ∧ i < length L
  by blast
  have bound1: interval-times a L ! i < interval-times a L ! (i+1)
    using interval-times-diff-ge[OF a-leq-b L-composition, of i ?s]
    using i-bound by blast
  have bound2: a ≤ interval-times a L ! i - 1
    using interval-times-diff-ge-general[OF a-leq-b L-composition, of i 0 ?s]
    using i-bound sfirst by simp
  have positive-i: interval-times a L ! i > 0
    using i-bound sfirst
    using interval-times-diff-ge-general[OF a-leq-b L-composition, of i 0 ?s]
    by auto
  have global-α: semantics-mltl-ext π (Global-mltl-ext a (?s ! i - 1) [?s!i -
?s!0] α)
  proof-
    have semantics-mltl (drop ia π) (to-mltl α)

```

```

    if ia-bound:  $a \leq ia \wedge ia \leq \text{interval-times } a \ L \ ! \ i - 1$  for ia
  proof-
    have  $a \leq ia \wedge ia < t$ 
      using ia-bound t-bound positive-i by auto
    then show ?thesis
      using t-semantic by blast
  qed
  then show ?thesis
    using bound2
    unfolding semantic-mltl-ext-def semantic-mltl.simps to-mltl.simps
    by blast
  qed
  have global-not-β: semantic-mltl-ext  $\pi$  (Global-mltl-ext  $a$  ( $?s \ ! \ i - 1$ ) [ $?s!i - ?s!0$ ] (Notc  $\beta$ ))
  proof-
    have  $\neg \text{semantic-mltl} (\text{drop } ia \ \pi) (\text{to-mltl } \beta)$ 
      if ia-bound:  $a \leq ia \wedge ia \leq \text{interval-times } a \ L \ ! \ i - 1$  for ia
    proof-
      have globally:  $(\forall j. a \leq j \wedge j < ia \longrightarrow \text{semantic-mltl} (\text{drop } j \ \pi) (\text{to-mltl } \alpha))$ 
        using global-α unfolding semantic-mltl-ext-def semantic-mltl.simps to-mltl.simps
        using length-π-ge-b a-leq-b
        using antisym dual-order.trans that by auto
      have  $a \leq ia \wedge ia < t$ 
        using ia-bound t-bound positive-i by auto
      then show ?thesis
        using t-minimal globally
        by (meson linorder-le-less-linear not-semantic)
    qed
  then show ?thesis
    unfolding semantic-mltl-ext-def semantic-mltl.simps to-mltl.simps
    using bound2 by blast
  qed
  let  $?ψ1 = \text{Global-mltl-ext } (?s \ ! \ 0) (?s \ ! \ i - 1) [?s!i - ?s!0] (\text{And-mltl-ext } \alpha (\text{Not}_c \ \beta))$ 
  have ψ1-semantic: semantic-mltl-ext  $\pi$   $?ψ1$ 
  proof-
    have  $p1: \text{semantic-mltl } \pi (\text{Global-mltl } (?s \ ! \ 0) (?s \ ! \ i - 1) (\text{to-mltl } \alpha))$ 
      using global-α unfolding semantic-mltl-ext-def to-mltl.simps first by
      blast
    have  $p2: \text{semantic-mltl } \pi (\text{Global-mltl } (?s \ ! \ 0) (?s \ ! \ i - 1) (\text{Not}_m (\text{to-mltl } \beta)))$ 
      using global-not-β unfolding semantic-mltl-ext-def to-mltl.simps first
by blast
    show ?thesis unfolding semantic-mltl-ext-def to-mltl.simps
      using  $p1 \ p2$  global-and-distribute by auto
  qed
  have interval-times  $a \ L \ ! \ (i + 1) \leq ?s!(\text{length } L)$ 

```

```

    using interval-times-diff-ge-general[OF a-leq-b L-composition, of length L
i+1 ?s]
    using i-bound
    by (metis le-eq-less-or-eq less-iff-succ-less-eq)
    then have interval-times a L ! (i + 1) - 1 ≤ b
    using slast by auto
    then have t ≤ b
    using t-bound by simp
    then have upd-mltl (to-mltl β) ≤ length (drop t π)
    using β-upd by simp
    then obtain x where x-antics: semantics-mltl-ext (drop t π) x
    and x-in: x ∈ set ?D-β
    using t-antics
    using Suc(1)[OF β-welldef β-nnf β-composition, of ?D-β (drop t π)]
    unfolding semantics-mltl-ext-def by blast
    let ?ψ2 = Until-mltl-ext α (?s ! i) (?s ! (i + 1) - 1) [?s ! (i + 1) - ?s ! i]
x
    have ψ2-antics: semantics-mltl-ext π ?ψ2
    proof-
    have (∀ j. interval-times a L ! i ≤ j ∧ j < t →
semantics-mltl (drop j π) (to-mltl α))
    using t-minimal not-antics
    by (metis bound2 diff-less dual-order.strict-trans1 dual-order.strict-trans2
less-numeral-extra(1) nless-le positive-i t-antics)
    then have semantics-mltl (drop t π) (to-mltl x) ∧
(∀ j. interval-times a L ! i ≤ j ∧ j < t →
semantics-mltl (drop j π) (to-mltl α))
    using x-antics unfolding semantics-mltl-ext-def by blast
    then have (∃ ia. (interval-times a L ! i ≤ ia ∧
ia ≤ interval-times a L ! (i + 1) - 1) ∧
semantics-mltl (drop ia π) (to-mltl x) ∧
(∀ j. interval-times a L ! i ≤ j ∧ j < ia →
semantics-mltl (drop j π) (to-mltl α)))
    using t-bound by blast
    then show ?thesis
    unfolding semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps
    using bound1
    by (smt (verit) ⟨interval-times a L ! (i + 1) - 1 ≤ b⟩ le-antisym
le-neq-implies-less le-trans length-π-ge-b less-or-eq-imp-le)
    qed
    let ?ψ = And-mltl-ext ?ψ1 ?ψ2
    have ψ-antics: semantics-mltl-ext π ?ψ
    using ψ1-antics ψ2-antics unfolding semantics-mltl-ext-def by
simp
    have ?ψ ∈ set ?back
    using x-in i-bound
    unfolding Until-mltl-list.simps by auto
    then have ψ-in: ?ψ ∈ set D
    using D-union by blast

```

```

  have ?thesis using  $\psi$ -semantics  $\psi$ -in by auto
}
ultimately show ?thesis
  using exist-bound-split[OF a-leq-b, of ?P ?s!1] semantics by blast
next
case (Release-mltl-ext  $\alpha$  a b L  $\beta$ )
have  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ )
and  $\beta$ -welldef: intervals-welldef (to-mltl  $\beta$ )
  using Suc(2) unfolding Release-mltl-ext by simp-all
have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
  using Suc(3) unfolding Release-mltl-ext
by (metis convert-nnf-ext.simps(9) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(8))
have  $\beta$ -nnf:  $\exists \varphi$ -init.  $\beta = \text{convert-nnf-ext } \varphi$ -init
  using Suc(3) unfolding Release-mltl-ext
by (metis convert-nnf-ext.simps(9) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(8))
have  $\alpha$ -convert: convert-nnf-ext  $\alpha = \alpha$ 
  using  $\alpha$ -nnf convert-nnf-ext-convert-nnf-ext by metis
have  $\beta$ -convert: convert-nnf-ext  $\beta = \beta$ 
  using  $\beta$ -nnf convert-nnf-ext-convert-nnf-ext by metis
have  $\alpha$ -composition: is-composition-MLTL  $\alpha$ 
and  $\beta$ -composition: is-composition-MLTL  $\beta$ 
and L-composition: is-composition (b-a+1) L
  using Suc(4) unfolding Release-mltl-ext is-composition-MLTL.simps
  by simp-all
have  $\alpha$ -wpd: b + wpd-mltl (to-mltl  $\alpha$ )  $\leq$  length  $\pi$ 
and  $\beta$ -wpd: b + wpd-mltl (to-mltl  $\beta$ )  $\leq$  length  $\pi$ 
  using Suc(7) unfolding Release-mltl-ext to-mltl.simps wpd-mltl.simps
  by simp-all
have length- $\pi$ -ge-b: b < length  $\pi$ 
  using wpd-geq-one[of to-mltl  $\beta$ ]  $\beta$ -wpd
  by auto
have a-leq-b: a  $\leq$  b
  using Suc(6)  $\alpha$ -wpd unfolding Release-mltl-ext semantics-mltl-ext-def to-mltl.simps
  semantics-mltl.simps
  by blast
have semantics: ( $\forall i. a \leq i \wedge i \leq b \longrightarrow$ 
  semantics-mltl (drop i  $\pi$ ) (to-mltl  $\beta$ ))  $\vee$ 
( $\exists j \geq a. j \leq b - 1 \wedge$ 
  semantics-mltl (drop j  $\pi$ ) (to-mltl  $\alpha$ )  $\wedge$ 
( $\forall k. a \leq k \wedge k \leq j \longrightarrow$ 
  semantics-mltl (drop k  $\pi$ ) (to-mltl  $\beta$ )))
  using Suc(6) unfolding Release-mltl-ext semantics-mltl-ext-def to-mltl.simps
  semantics-mltl.simps
  using length- $\pi$ -ge-b by auto
let ?D = LP-mltl-aux  $\alpha$  k
let ?s = interval-times a L
have sfirst: ?s!0 = a
  using interval-times-first by auto
have slast: ?s!(length L) = b+1

```



```

using interval-times-last[OF a-leq-b L-composition] by auto
let ?front = set [Global-mltl-ext a b L (And-mltl-ext (Notc α) β)]
let ?middle = set (Mighty-Release-mltl-list ?D β (?s ! 0) (?s ! 1 - 1)
  [?s ! 1 - ?s ! 0])
let ?back = set (concat (map ( $\lambda i$ . And-mltl-list
  [Global-mltl-ext
    (?s ! 0) (?s ! i - 1) [?s!i - ?s!0] (And-mltl-ext (Notc α)
β)]
    (Mighty-Release-mltl-list ?D β (?s ! i)
    (?s ! (i + 1) - 1) [?s ! (i + 1) - ?s ! i]))
  [1..<length L]))
let ?P =  $\lambda j$ . (semantics-mltl (drop j π) (to-mltl α)  $\wedge$ 
  ( $\forall k$ .  $a \leq k \wedge k \leq j \longrightarrow$ 
    semantics-mltl (drop k π) (to-mltl β)))
have D-is: set D = ?front  $\cup$  ?middle  $\cup$  ?back
unfolding Suc(5) Release-mltl-ext LP-mltl-ax.simps
using  $\alpha$ -convert list-concat-set-union
by (metis append-assoc)
{
  assume *: ( $\forall i$ .  $a \leq i \wedge i \leq b \longrightarrow$  semantics-mltl (drop i π) (to-mltl β))
     $\wedge$  ( $\forall i$ .  $a \leq i \wedge i \leq b \longrightarrow$  semantics-mltl (drop i π) (Notm (to-mltl
α)))
  let ?ψ = Global-mltl-ext a b L (And-mltl-ext (Notc α) β)
  have ψ-in: ?ψ  $\in$  set D
  using D-is by auto
  have semantics-mltl-ext π ?ψ
  unfolding semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps
  using a-leq-b * by auto
  then have ?thesis using ψ-in by blast
} moreover {
  assume *:  $\exists i$ .  $a \leq i \wedge i \leq b \wedge ?P i$ 
  then obtain t' where t'-semantics: ?P t'
    and t'-bound:  $a \leq t' \wedge t' \leq b$ 
  by blast
  then obtain t where t-semantics: ?P t
    and t-bound:  $a \leq t \wedge t \leq t'$ 
    and t-minimal:  $\forall j$ . ( $a \leq j \wedge j < t$ )  $\longrightarrow \neg ?P j$ 
  using exist-first[of a t' ?P] by blast
  have globally-notα:  $\forall i$ . ( $a \leq i \wedge i < t$ )  $\longrightarrow$ 
     $\neg$  (semantics-mltl-ext (drop i π) α)
  using t-minimal t-semantics unfolding semantics-mltl-ext-def by auto
  have α-semantics: semantics-mltl-ext (drop t π) α
  using t-semantics unfolding semantics-mltl-ext-def by blast
  have globally-β:  $\forall i$ . ( $a \leq i \wedge i \leq t$ )  $\longrightarrow$  (semantics-mltl-ext (drop i π) β)
  using t-semantics unfolding semantics-mltl-ext-def by blast
  obtain i where t-bound: ?s!i  $\leq t \wedge t \leq ?s!(i+1)-1$ 
    and i-bound:  $0 \leq i \wedge i < \text{length } L$ 
  using interval-times-obtain[OF a-leq-b L-composition, of ?s t]
  using t-bound t'-bound by auto

```

```

have lb:  $a \leq ?s!i$ 
  using i-bound sfirst interval-times-diff-ge-general[OF a-leq-b L-composition,
of i 0 ?s]
  by force
have welldef:  $?s!i < ?s!(i+1)$ 
  using i-bound
  using interval-times-diff-ge[OF a-leq-b L-composition, of i ?s]
  by blast
have ub:  $?s!(i+1) \leq b+1$ 
  using i-bound slast interval-times-diff-ge-general[OF a-leq-b L-composition,
of length L i+1 ?s]
  by (metis Orderings.order-eq-iff less-iff-succ-less-eq order-le-imp-less-or-eq
order-less-imp-le)
have wpd-mttl (to-mttl  $\alpha$ )  $\leq$  length (drop t  $\pi$ )
  using  $\alpha$ -wpd t-bound i-bound sfirst welldef ub by auto
then obtain x where x-antics: semantics-mttl-ext (drop t  $\pi$ ) x
  and x-in:  $x \in$  set (LP-mttl-aux  $\alpha$  k)
  using Suc(1)[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition -  $\alpha$ -semantics, of ?D]
  by blast
{
  assume i-bound:  $i = 0$ 
  let  $?\psi =$  Mighty-Release-mttl-ext  $x \beta a$  (interval-times a L ! 1 - 1)
[interval-times a L ! 1 - a]
  have  $\psi$ -in:  $? \psi \in$  ?middle using x-in unfolding sfirst by auto
  then have  $\psi$ -in:  $? \psi \in$  set D using D-is by blast
  have semantics-mttl-ext  $\pi ?\psi$ 
  proof-
    have sem1: ( $\forall i. a \leq i \wedge i \leq$  interval-times a L ! 1 - 1  $\longrightarrow$ 
      semantics-mttl (drop i  $\pi$ ) (to-mttl  $\beta$ ))  $\vee$ 
      ( $\exists j \geq a. j \leq$  interval-times a L ! 1 - 1 - 1  $\wedge$ 
        semantics-mttl (drop j  $\pi$ ) (to-mttl  $x$ )  $\wedge$ 
        ( $\forall k. a \leq k \wedge k \leq j \longrightarrow$ 
          semantics-mttl (drop k  $\pi$ ) (to-mttl  $\beta$ )))
    proof-
      {
        assume t-loc:  $t = ?s!(i+1) - 1$ 
        then have ?thesis
          using globally- $\beta$ 
          by (simp add: i-bound t-semantics)
      } moreover {
        assume t-loc:  $?s!i \leq t \wedge t \leq ?s!(i+1) - 1 - 1$ 
        then have ?thesis
          using t-semantics i-bound globally- $\beta$ 
          by (metis add-cancel-right-left semantics-mttl-ext-def sfirst x-semantics)
      }
    ultimately show ?thesis using t-bound by fastforce
  qed
  have sem2: ( $\exists i. (a \leq i \wedge i \leq$  interval-times a L ! 1 - 1)  $\wedge$ 

```

```

    semantics-mltl (drop i  $\pi$ ) (to-mltl x))
      using x-semantic t-bound ub lb welldef unfolding semantics-mltl-ext-def
      using i-bound sfirst by auto
    show ?thesis unfolding Mighty-Release-mltl-ext.simps semantics-mltl-ext-def
    to-mltl.simps semantics-mltl.simps
      using welldef i-bound sem1 sem2 length- $\pi$ -ge-b a-leq-b by auto
    qed
    then have ?thesis
      using  $\psi$ -in by auto
  } moreover {
    assume i-bound:  $0 < i \wedge i < \text{length } L$ 
    have lb:  $a < ?s!i$ 
    using i-bound sfirst interval-times-diff-ge-general[OF a-leq-b L-composition,
of i 0 ?s]
      by force
    let ? $\psi$  = And-mltl-ext
      (Global-mltl-ext
        a (interval-times a L ! i - 1) [?s!i - ?s!0] (And-mltl-ext (Notc
 $\alpha$ )  $\beta$ ))
      (Mighty-Release-mltl-ext x  $\beta$ 
        (interval-times a L ! i) (interval-times a L ! (i + 1) - 1)
        [interval-times a L ! (i + 1) - interval-times a L ! i])
    have ? $\psi$   $\in$  ?back
      using x-in i-bound sfirst by auto
    then have  $\psi$ -in: ? $\psi$   $\in$  set D using D-is by blast
    have semantics-mltl-ext  $\pi$  ? $\psi$ 
    proof -
      have p1: ( $\forall ia. a \leq ia \wedge ia \leq \text{interval-times } a L ! i - 1 \longrightarrow$ 
         $\neg \text{semantics-mltl } (\text{drop } ia \ \pi) \ (\text{to-mltl } \alpha) \wedge$ 
         $\text{semantics-mltl } (\text{drop } ia \ \pi) \ (\text{to-mltl } \beta)$ )
        using globally-not $\alpha$  globally- $\beta$  t-bound lb ub welldef
        unfolding semantics-mltl-ext-def by auto
      have p2: ( $\forall ia. \text{interval-times } a L ! i \leq ia \wedge$ 
         $ia \leq \text{interval-times } a L ! (i + 1) - 1 \longrightarrow$ 
         $\text{semantics-mltl } (\text{drop } ia \ \pi) \ (\text{to-mltl } \beta)$ )  $\vee$ 
        ( $\exists j \geq \text{interval-times } a L ! i.$ 
         $j \leq \text{interval-times } a L ! (i + 1) - 1 - 1 \wedge$ 
         $\text{semantics-mltl } (\text{drop } j \ \pi) \ (\text{to-mltl } x) \wedge$ 
        ( $\forall k. \text{interval-times } a L ! i \leq k \wedge k \leq j \longrightarrow$ 
         $\text{semantics-mltl } (\text{drop } k \ \pi) \ (\text{to-mltl } \beta)$ ))
    proof -
      {
        assume t-loc:  $t = \text{interval-times } a L ! (i + 1) - 1$ 
        then have ?thesis
          using globally- $\beta$  t-bound ub lb welldef
          by (metis le-trans less-or-eq-imp-le t-semantic)
      } moreover {
        assume t-loc:  $t \leq \text{interval-times } a L ! (i + 1) - 1 - 1$ 
        then have ?thesis

```

```

    using x-semantic globally-β t-bound ub lb welldef
    by (meson le-trans less-imp-le-nat semantics-mltl-ext-def)
  }
  ultimately show ?thesis using t-bound by fastforce
qed
have p3: (∃ ia. (interval-times a L ! i ≤ ia ∧
ia ≤ interval-times a L ! (i + 1) - 1) ∧
semantics-mltl (drop ia π) (to-mltl x))
  using x-semantic i-bound lb ub welldef
  unfolding semantics-mltl-ext-def
  using t-bound by auto
have tracelen: interval-times a L ! i < length π
  using length-π-ge-b ub welldef by simp
then show ?thesis unfolding semantics-mltl-ext-def to-mltl.simps Mighty-Release-mltl-ext.simps
semantics-mltl.simps
  using lb ub welldef p1 p2 p3 by auto
qed
then have ?thesis
  using ψ-in by auto
}
ultimately have ?thesis using i-bound by blast
}
ultimately show ?thesis using semantics Release-semantics-split
by blast
qed
qed

```

Converse Direction lemma *LP-mltl-aux-language-union-converse:*

```

fixes φ::'a mltl-ext and k::nat and π::'a set list
assumes intervals-welldef: intervals-welldef (to-mltl φ)
assumes is-nnf: ∃ φ-init. φ = convert-nnf-ext φ-init
assumes composition: is-composition-MTLT φ
assumes trace-length: length π ≥ wpd-mltl (to-mltl φ)
assumes D-is: D = LP-mltl-aux φ k
assumes ∃ ψ ∈ set D. semantics-mltl-ext π ψ
shows semantics-mltl-ext π φ
using assms
proof(induct k arbitrary: D φ π)
  case 0
  then show ?case by simp
next
  case (Suc k)
  then show ?case
proof(cases φ)
  case True-mltl-ext
  then show ?thesis unfolding semantics-mltl-ext-def by simp
next
  case False-mltl-ext
  then show ?thesis using assms unfolding semantics-mltl-ext-def

```

by (*metis LP-mltl-aux.simps(3) Suc.prem(5) Suc.prem(6) empty-iff empty-set semantics-mltl-ext-def set-ConsD*)
next
case (*Prop-mltl-ext p*)
then show *?thesis* **using** *Suc*
unfolding *semantics-mltl-ext-def* **by** *simp*
next
case (*Not-mltl-ext q*)
then have $\exists p. q = \text{Prop-mltl-ext } p$
using *convert-nnf-form-Not-Implies-Prop Suc*
by (*metis convert-nnf-ext-to-mltl-commute to-mltl.simps(4) to-mltl-prop-bijective*)

then obtain *p* **where** $q = \text{Prop-mltl-ext } p$ **by** *blast*
then show *?thesis*
using *Not-mltl-ext Suc* **by** *simp*
next
case (*And-mltl-ext α β*)
have α -*welldef: intervals-welldef (to-mltl α)* **and**
 β -*welldef: intervals-welldef (to-mltl β)*
using *Suc(2) unfolding And-mltl-ext* **by** *simp-all*
have α -*nnf: $\exists \varphi$ -init. $\alpha = \text{convert-nnf-ext } \varphi$ -init*
using *Suc(3) unfolding And-mltl-ext*
by (*metis convert-nnf-ext.simps(4) convert-nnf-ext-convert-nnf-ext mtl-ext.inject(3)*)

have β -*nnf: $\exists \varphi$ -init. $\beta = \text{convert-nnf-ext } \varphi$ -init*
using *Suc(3) unfolding And-mltl-ext*
by (*metis convert-nnf-ext.simps(4) convert-nnf-ext-convert-nnf-ext mtl-ext.inject(3)*)

have α -*composition: is-composition-MLTL α* **and**
 β -*composition: is-composition-MLTL β*
using *Suc(4) unfolding And-mltl-ext is-composition-MLTL.simps* **by** *simp-all*
have α -*convert: convert-nnf-ext $\alpha = \alpha$*
using α -*nnf* β -*nnf* *convert-nnf-ext-convert-nnf-ext* **by** *metis*
have β -*convert: convert-nnf-ext $\beta = \beta$*
using α -*nnf* β -*nnf* *convert-nnf-ext-convert-nnf-ext* **by** *metis*
have α -*wpd: length $\pi \geq \text{wpd-mltl (to-mltl } \alpha)$* **and**
 β -*wpd: length $\pi \geq \text{wpd-mltl (to-mltl } \beta)$*
using *Suc(5) unfolding And-mltl-ext to-mltl.simps wpd-mltl.simps*
by *simp-all*
obtain ψ **where** ψ -*in: $\psi \in \text{set } D$*
and ψ -*semantics: semantics-mltl-ext π ψ*
using *Suc(7) by blast*
let *?Da = LP-mltl-aux α k*
let *?Db = LP-mltl-aux β k*
obtain x y **where** ψ -*is: $\psi = \text{And-mltl-ext } x$ y*
and x -*in: $x \in \text{set } ?Da$*
and y -*in: $y \in \text{set } ?Db$*
using ψ -*in* *unfolding Suc(6) And-mltl-ext LP-mltl-aux.simps*
using *And-mltl-list-member unfolding List.member-def*

```

    using  $\alpha$ -convert  $\beta$ -convert by metis
  have  $x$ -semantics: semantics-mltl-ext  $\pi$   $x$  and
     $y$ -semantics: semantics-mltl-ext  $\pi$   $y$ 
    using  $\psi$ -semantics unfolding semantics-mltl-ext-def  $\psi$ -is to-mltl.simps
    by simp-all
  have  $\alpha$ -ih: semantics-mltl-ext  $\pi$   $\alpha$ 
    using Suc(1)[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition  $\alpha$ -wpd, of ?Da]
    using  $x$ -in  $x$ -semantics by blast
  have  $\beta$ -ih: semantics-mltl-ext  $\pi$   $\beta$ 
    using Suc(1)[OF  $\beta$ -welldef  $\beta$ -nnf  $\beta$ -composition  $\beta$ -wpd, of ?Db]
    using  $y$ -in  $y$ -semantics by blast
  show ?thesis
    using  $\alpha$ -ih  $\beta$ -ih unfolding And-mltl-ext semantics-mltl-ext-def by auto
next
case (Or-mltl-ext  $\alpha$   $\beta$ )
have  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ ) and
   $\beta$ -welldef: intervals-welldef (to-mltl  $\beta$ )
  using Suc(2) unfolding Or-mltl-ext by simp-all
have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
  using Suc(3) unfolding Or-mltl-ext
by (metis convert-nnf-ext.simps(5) convert-nnf-ext-convert-nnf-ext mtl-ext.inject(4))

have  $\beta$ -nnf:  $\exists \varphi$ -init.  $\beta = \text{convert-nnf-ext } \varphi$ -init
  using Suc(3) unfolding Or-mltl-ext
by (metis convert-nnf-ext.simps(5) convert-nnf-ext-convert-nnf-ext mtl-ext.inject(4))

have  $\alpha$ -composition: is-composition-MLTL  $\alpha$  and
   $\beta$ -composition: is-composition-MLTL  $\beta$ 
  using Suc(4) unfolding Or-mltl-ext is-composition-MLTL.simps by simp-all
have  $\alpha$ -convert: convert-nnf-ext  $\alpha = \alpha$ 
  using  $\alpha$ -nnf  $\beta$ -nnf convert-nnf-ext-convert-nnf-ext by metis
have  $\beta$ -convert: convert-nnf-ext  $\beta = \beta$ 
  using  $\alpha$ -nnf  $\beta$ -nnf convert-nnf-ext-convert-nnf-ext by metis
have  $\alpha$ -wpd: length  $\pi \geq \text{wpd-mltl (to-mltl } \alpha)$  and
   $\beta$ -wpd: length  $\pi \geq \text{wpd-mltl (to-mltl } \beta)$ 
  using Suc(5) unfolding Or-mltl-ext to-mltl.simps wpd-mltl.simps
  by simp-all
obtain  $\psi$  where  $\psi$ -in:  $\psi \in \text{set } D$ 
  and  $\psi$ -semantics: semantics-mltl-ext  $\pi$   $\psi$ 
  using Suc(7) by blast
let ?Da = LP-mltl-aux  $\alpha$   $k$ 
let ?Db = LP-mltl-aux  $\beta$   $k$ 
let ?front = And-mltl-list ?Da ?Db
let ?middle = And-mltl-list [Notc  $\alpha$ ] ?Db
let ?back = And-mltl-list ?Da [Notc  $\beta$ ]
have cases:  $\psi \in (\text{set } ?front) \cup (\text{set } ?middle) \cup (\text{set } ?back)$ 
  using Suc(6) unfolding Or-mltl-ext LP-mltl-aux.simps using  $\psi$ -in
  by (metis  $\alpha$ -convert  $\beta$ -convert boolean-algebra-cancel.sup1 set-append)
{

```

```

assume *:  $\psi \in \text{set } ?\text{front}$ 
obtain  $x\ y$  where  $\psi\text{-is: } \psi = \text{And-mltl-ext } x\ y$ 
  and  $x\text{-in: } x \in \text{set } ?Da$ 
  and  $y\text{-in: } y \in \text{set } ?Db$ 
  using  $\psi\text{-in} * \text{unfolding } \text{Or-mltl-ext } \text{LP-mltl-aux.simps}$ 
  using  $\text{And-mltl-list-member unfolding } \text{List.member-def}$ 
  using  $\alpha\text{-convert } \beta\text{-convert by } \text{metis}$ 
have  $x\text{-semantics: semantics-mltl-ext } \pi\ x$  and
   $y\text{-semantics: semantics-mltl-ext } \pi\ y$ 
  using  $\psi\text{-semantics unfolding semantics-mltl-ext-def } \psi\text{-is to-mltl.simps}$ 
  by  $\text{simp-all}$ 
have  $\alpha\text{-ih: semantics-mltl-ext } \pi\ \alpha$ 
  using  $\text{Suc}(1)[\text{OF } \alpha\text{-welldef } \alpha\text{-nnf } \alpha\text{-composition } \alpha\text{-wpd, of } ?Da]$ 
  using  $x\text{-in } x\text{-semantics by } \text{blast}$ 
have  $\beta\text{-ih: semantics-mltl-ext } \pi\ \beta$ 
  using  $\text{Suc}(1)[\text{OF } \beta\text{-welldef } \beta\text{-nnf } \beta\text{-composition } \beta\text{-wpd, of } ?Db]$ 
  using  $y\text{-in } y\text{-semantics by } \text{blast}$ 
have  $?thesis$ 
  using  $\alpha\text{-ih } \beta\text{-ih unfolding } \text{Or-mltl-ext semantics-mltl-ext-def by } \text{auto}$ 
} moreover {
  assume *:  $\psi \in \text{set } ?\text{middle}$ 
  obtain  $y$  where  $\psi\text{-is: } \psi = \text{And-mltl-ext } (\text{Not}_c\ \alpha)\ y$ 
    and  $y\text{-in: } y \in \text{set } ?Db$ 
    using  $\psi\text{-in} * \text{unfolding } \text{Or-mltl-ext } \text{LP-mltl-aux.simps}$ 
    using  $\text{And-mltl-list-member unfolding } \text{List.member-def}$ 
    using  $\alpha\text{-convert } \beta\text{-convert by } \text{auto}$ 
  have  $x\text{-semantics: semantics-mltl-ext } \pi\ (\text{Not}_c\ \alpha)$  and
     $y\text{-semantics: semantics-mltl-ext } \pi\ y$ 
    using  $\psi\text{-semantics unfolding semantics-mltl-ext-def } \psi\text{-is to-mltl.simps}$ 
    by  $\text{simp-all}$ 
  have  $\beta\text{-ih: semantics-mltl-ext } \pi\ \beta$ 
    using  $\text{Suc}(1)[\text{OF } \beta\text{-welldef } \beta\text{-nnf } \beta\text{-composition } \beta\text{-wpd, of } ?Db]$ 
    using  $y\text{-in } y\text{-semantics by } \text{blast}$ 
  have  $?thesis$ 
    using  $\beta\text{-ih unfolding } \text{Or-mltl-ext semantics-mltl-ext-def by } \text{auto}$ 
} moreover {
  assume *:  $\psi \in \text{set } ?\text{back}$ 
  obtain  $x$  where  $\psi\text{-is: } \psi = \text{And-mltl-ext } x\ (\text{Not}_c\ \beta)$ 
    and  $x\text{-in: } x \in \text{set } ?Da$ 
    using  $\psi\text{-in} * \text{unfolding } \text{Or-mltl-ext } \text{LP-mltl-aux.simps}$ 
    using  $\text{And-mltl-list-member unfolding } \text{List.member-def}$ 
    using  $\alpha\text{-convert } \beta\text{-convert}$ 
    by  $(\text{metis } \text{empty-iff } \text{empty-set } \text{set-ConsD})$ 
  have  $x\text{-semantics: semantics-mltl-ext } \pi\ x$  and
     $y\text{-semantics: semantics-mltl-ext } \pi\ (\text{Not}_c\ \beta)$ 
    using  $\psi\text{-semantics unfolding semantics-mltl-ext-def } \psi\text{-is to-mltl.simps}$ 
    by  $\text{simp-all}$ 
  have  $\alpha\text{-ih: semantics-mltl-ext } \pi\ \alpha$ 
    using  $\text{Suc}(1)[\text{OF } \alpha\text{-welldef } \alpha\text{-nnf } \alpha\text{-composition } \alpha\text{-wpd, of } ?Da]$ 

```

```

    using x-in x-semantics by blast
  have ?thesis
    using  $\alpha$ -ih unfolding Or-mltl-ext semantics-mltl-ext-def by auto
  }
  ultimately show ?thesis using cases by blast
next
case (Future-mltl-ext a b L  $\alpha$ )
have  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ ) and
  a-leq-b:  $a \leq b$ 
  using Suc(2) unfolding Future-mltl-ext by simp-all
have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
  using Suc(3) unfolding Future-mltl-ext
by (metis convert-nnf-ext.simps(6) convert-nnf-ext-convert-nnf-ext mtl-ext.inject(5))

have  $\alpha$ -composition: is-composition-MLTL  $\alpha$  and
  L-composition: is-composition (b-a+1) L
  using Suc(4) unfolding Future-mltl-ext is-composition-MLTL.simps by
simp-all
have  $\alpha$ -convert: convert-nnf-ext  $\alpha = \alpha$ 
  using  $\alpha$ -nnf convert-nnf-ext-convert-nnf-ext by metis
have  $\alpha$ -wpd: length  $\pi \geq b + \text{wpd-mltl (to-mltl } \alpha)$ 
  using Suc(5) unfolding Future-mltl-ext to-mltl.simps wpd-mltl.simps
  by simp-all
then have length- $\pi$ -ge-b: length  $\pi > b$ 
  using wpd-geq-one[of to-mltl  $\alpha$ ] by auto
obtain  $\psi$  where  $\psi$ -in:  $\psi \in \text{set } D$ 
  and  $\psi$ -semantics: semantics-mltl-ext  $\pi \psi$ 
  using Suc(7) by blast
let ?D = LP-mltl-aux  $\alpha k$ 
let ?s = interval-times a L
have length-L: 1  $\leq$  length L
  using composition-length-lb[OF L-composition] a-leq-b by linarith
have sfirst: ?s!0 = a
  using interval-times-first by simp
have slast: ?s!(length L) = b+1
  using interval-times-last[OF a-leq-b L-composition] by blast
let ?front = (Future-mltl-list ?D (?s!0) (?s!1-1) [?s!1-?s!0])
let ?back = (concat (map ( $\lambda i$ . And-mltl-list
  [Global-mltl-ext (?s!0) (?s!i-1) [?s!i-?s!0] (Notc  $\alpha$ )]
  (Future-mltl-list ?D (?s!i) (?s!(i+1)-1) [?s!(i+1)
- ?s!i]))
  [1..<length L]))
have cases:  $\psi \in (\text{set } ?\text{front}) \cup (\text{set } ?\text{back})$ 
  using  $\psi$ -in using Suc(6) unfolding Future-mltl-ext LP-mltl-aux.simps
  using list-concat-set-union[of ?front ?back]  $\alpha$ -convert by metis
{
  assume *:  $\psi \in \text{set } ?\text{front}$ 
  then obtain  $x$  where  $\psi$ -is:  $\psi = \text{Future-mltl-ext } (?s!0) (?s!1-1) [?s!1$ 
- ?s!0] x

```


and $x\text{-in}: x \in \text{set } ?D$
unfolding *Future-mltl-list.simps* **by** *fastforce*
obtain l **where** $x\text{-semantics}: \text{semantics-mltl } (\text{drop } l \ \pi) \ (\text{to-mltl } x)$ **and**
 $l\text{-bound}: a \leq l \wedge l \leq \text{interval-times } a \ L \ ! \ 1 - 1$
using $\psi\text{-semantics}$
unfolding $\psi\text{-is semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps}$
sfirst
by *blast*
have $\text{bound}: \text{interval-times } a \ L \ ! \ 1 - 1 \leq b$
using *slast length-L l-bound*
using *interval-times-diff-ge-general[OF a-leq-b L-composition, of length L 1*
 $?s]$
by *force*
then **have** $\text{wpd-mltl } (\text{to-mltl } \alpha) \leq \text{length } (\text{drop } l \ \pi)$
using $\alpha\text{-wpd } l\text{-bound}$ **by** *auto*
then **have** $\alpha\text{-ih}: \text{semantics-mltl-ext } (\text{drop } l \ \pi) \ \alpha$
using *Suc(1)[OF $\alpha\text{-welldef } \alpha\text{-nnf } \alpha\text{-composition, of drop } l \ \pi \ ?D]$*
using $x\text{-in } x\text{-semantics semantics-mltl-ext-def}$ **by** *auto*
then **have** $?thesis$ **unfolding** *Future-mltl-ext semantics-mltl-ext-def*
unfolding *to-mltl.simps semantics-mltl.simps*
using $\text{length-}\pi\text{-ge-}b \ a\text{-leq-}b \ l\text{-bound bound}$ **by** *auto*
} **moreover** **{**
assume $*$: $\psi \in \text{set } ?back$
then **obtain** i **where** $\psi\text{-is}: \psi \in \text{set } (\text{And-mltl-list}$
 $[\text{Global-mltl-ext } (?s \ ! \ 0) \ (?s \ ! \ i - 1) \ [?s!i - ?s!0] \ (\text{Not}_c \ \alpha)]$
 $(\text{Future-mltl-list } ?D \ (?s \ ! \ i) \ (?s \ ! \ (i + 1) - 1) \ [?s \ ! \ (i + 1)$
 $- \ ?s \ ! \ i]))$
and $i\text{-bound}: 1 \leq i \wedge i < \text{length } L$
by *force*
obtain x **where** $\psi\text{-is}: \psi = \text{And-mltl-ext}$
 $(\text{Global-mltl-ext } (?s \ ! \ 0) \ (?s \ ! \ i - 1) \ [?s!i - ?s!0] \ (\text{Not}_c \ \alpha))$
 $(\text{Future-mltl-ext } (?s \ ! \ i) \ (?s \ ! \ (i + 1) - 1) \ [?s \ ! \ (i + 1) -$
 $?s \ ! \ i] \ x)$
and $x\text{-in}: x \in \text{set } ?D$
using $\psi\text{-is unfolding Future-mltl-list.simps}$ **by** *auto*
obtain l **where** $x\text{-semantics}: \text{semantics-mltl } (\text{drop } l \ \pi) \ (\text{to-mltl } x)$ **and**
 $l\text{-bound}: ?s \ ! \ i \leq l \wedge l \leq ?s \ ! \ (i + 1) - 1$
using $\psi\text{-semantics unfolding } \psi\text{-is semantics-mltl-ext-def to-mltl.simps}$
 $\text{semantics-mltl.simps}$
by *auto*
have $\text{interval-times } a \ L \ ! \ (i + 1) \leq \text{interval-times } a \ L \ ! \ \text{length } L$
using *interval-times-diff-ge-general[OF a-leq-b L-composition, of length L*
 $i+1 \ ?s]$
using $i\text{-bound}$
by *(metis less-iff-succ-less-eq order-le-less)*
then **have** $\text{bound}: \text{interval-times } a \ L \ ! \ (i + 1) \leq b+1$
unfolding *slast* **by** *blast*
then **have** $l \leq b$
using $l\text{-bound slast}$ **by** *auto*

```

then have wpd-mltl (to-mltl  $\alpha$ )  $\leq$  length (drop l  $\pi$ )
  using l-bound  $\alpha$ -wpd by simp
then have  $\alpha$ -ih: semantics-mltl-ext (drop l  $\pi$ )  $\alpha$ 
  using Suc(1)[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition, of drop l  $\pi$  ?D]
  using x-in x-semantics semantics-mltl-ext-def by blast
have lb:  $a \leq$  interval-times  $a$   $L$  !  $i$ 
  using interval-times-diff-ge-general[OF  $a$ -leq-b  $L$ -composition, of i 0 ?s]
  using sfirst i-bound by auto
have ?thesis unfolding Future-mltl-ext semantics-mltl-ext-def
  unfolding to-mltl.simps semantics-mltl.simps
  using length- $\pi$ -ge-b a-leq-b l-bound  $\alpha$ -ih lb bound unfolding semantics-mltl-ext-def
  by (metis  $\langle l \leq b \rangle$  dual-order.trans order-le-less-trans)
}
ultimately show ?thesis using cases by blast
next
case (Global-mltl-ext a b L  $\alpha$ )
have  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ ) and
   $a$ -leq-b:  $a \leq b$ 
  using Suc(2) unfolding Global-mltl-ext by simp-all
have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha =$  convert- $\alpha$ -nnf-ext  $\varphi$ -init
  using Suc(3) unfolding Global-mltl-ext
by (metis convert- $\alpha$ -nnf-ext.simps(7) convert- $\alpha$ -nnf-ext-convert- $\alpha$ -nnf-ext mtl-ext.inject(6))

have  $\alpha$ -composition: is-composition-MLTL  $\alpha$ 
  using Suc(4) unfolding Global-mltl-ext is-composition-MLTL.simps by
simp-all
have  $\alpha$ -convert: convert- $\alpha$ -nnf-ext  $\alpha = \alpha$ 
  using  $\alpha$ -nnf convert- $\alpha$ -nnf-ext-convert- $\alpha$ -nnf-ext by metis
have  $\alpha$ -wpd: length  $\pi \geq b +$  wpd-mltl (to-mltl  $\alpha$ )
  using Suc(5) unfolding Global-mltl-ext to-mltl.simps wpd-mltl.simps
  by simp-all
then have length- $\pi$ -ge-b: length  $\pi > b$ 
  using wpd-geq-one[of to-mltl  $\alpha$ ] by auto
obtain  $\psi$  where  $\psi$ -in:  $\psi \in$  set D
  and  $\psi$ -semantics: semantics-mltl-ext  $\pi$   $\psi$ 
  using Suc(7) by blast
let ?D = LP-mltl-aux  $\alpha$   $k$ 
{
  assume *: length ?D  $\leq 1$ 
  have  $D =$  [Global-mltl-ext a b L  $\alpha$ ]
    using Suc(6) unfolding Global-mltl-ext LP-mltl-aux.simps
    using *  $\alpha$ -convert by auto
  then have ?thesis using Suc
    by (simp add: Global-mltl-ext)
} moreover {
  assume *: length ?D  $> 1$ 
  then have  $D$ -is:  $D =$  (Global-mltl-decomp ?D a (b - a) L)
    using Suc  $\alpha$ -nnf  $\alpha$ -convert unfolding Global-mltl-ext LP-mltl-aux.simps

```

```

    by simp
  obtain  $\psi$  where  $\psi$ -in:  $\psi \in \text{set } (\text{Global-mltl-decomp } ?D \ a \ (b - a) \ L)$ 
    and  $\psi$ -semantics:  $\text{semantics-mltl-ext } \pi \ \psi$ 
  using  $\text{Suc}(\gamma)$  unfolding  $D$ -is by blast
  then obtain  $X$  where  $\psi$ -is:  $\psi = \text{Ands-mltl-ext } X$ 
    and  $X$ -fact:  $\forall i < \text{length } X. \exists y \in \text{set } (\text{LP-mltl-aux } \alpha \ k).$ 
       $X ! i = \text{Global-mltl-ext } (a + i) \ (a + i) \ [1] \ y$ 
    and  $\text{length-}X$ :  $\text{length } X = \text{Suc } (b - a)$ 
  using  $\text{in-Global-mltl-decomp-exact-forward}[OF * \psi\text{-in}]$  by blast
  have  $\text{semantics-mltl } (\text{drop } i \ \pi) \ (\text{to-mltl } \alpha)$ 
  if  $i$ -bound:  $a \leq i \wedge i \leq b$  for  $i$ 
  proof-
  have  $i - a < \text{length } X$ 
    using  $i$ -bound  $\text{length-}X \ a\text{-leq-}b$  by linarith
  then obtain  $y$  where  $y$ -in:  $y \in \text{set } ?D$ 
    and  $Xi$ -is:  $X!(i - a) = \text{Global-mltl-ext } (a + i - a) \ (a + i - a) \ [1] \ y$ 
  using  $X$ -fact  $i$ -bound by auto
  have  $\text{semantics-mltl-ext } (\text{drop } i \ \pi) \ y$ 
  proof-
  have  $i$ -length-trace:  $i < \text{length } \pi$ 
    using  $i$ -bound  $\text{length-}\pi\text{-ge-}b$  by auto
  have  $\text{Ands-semantics}$ :  $(\forall x \in \text{set } X. \text{semantics-mltl-ext } \pi \ x)$ 
    using  $\psi$ -semantics unfolding  $\psi$ -is
    using  $\text{Ands-mltl-semantics}[of \ X \ \pi] \ \text{length-}X$  by auto
  have  $(\text{Global-mltl-ext } i \ i \ [1] \ y) \in \text{set } X$ 
    using  $Xi$ -is  $i$ -bound  $\langle i - a < \text{length } X \rangle \ \text{nth-mem}$  by fastforce
  then have  $\text{semantics-mltl-ext } \pi \ (\text{Global-mltl-ext } i \ i \ [1] \ y)$ 
    using  $\text{Ands-semantics}$  by blast
  then show  $?thesis$  unfolding  $\text{semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps}$ 
    using  $i$ -length-trace by simp
  qed
  then have  $\text{semantics}$ :  $\exists a \in \text{set } ?D. \text{semantics-mltl-ext } (\text{drop } i \ \pi) \ a$ 
    using  $y$ -in by blast
  have  $\text{wpd}$ :  $\text{wpd-mltl } (\text{to-mltl } \alpha) \leq \text{length } (\text{drop } i \ \pi)$ 
    using  $\text{length-}\pi\text{-ge-}b \ \alpha\text{-wpd } i$ -bound by auto
  show  $?thesis$ 
    using  $\text{Suc}(1)[OF \ \alpha\text{-welldef } \alpha\text{-nnf } \alpha\text{-composition, of drop } i \ \pi \ ?D]$ 
    using  $\text{wpd semantics}$  unfolding  $\text{semantics-mltl-ext-def}$  by blast
  qed
  then have  $?thesis$  unfolding  $\text{Global-mltl-ext semantics-mltl-ext-def semantics-mltl.simps to-mltl.simps}$ 
    using  $a$ -leq- $b \ \text{length-}\pi\text{-ge-}b$  by blast
}
ultimately show  $?thesis$  by linarith
next
case  $(\text{Until-mltl-ext } \alpha \ a \ b \ L \ \beta)$ 
have  $\alpha$ -welldef:  $\text{intervals-welldef } (\text{to-mltl } \alpha)$  and
   $\beta$ -welldef:  $\text{intervals-welldef } (\text{to-mltl } \beta)$  and

```

```

    a-leq-b:  $a \leq b$ 
    using Suc(2) unfolding Until-mltl-ext by simp-all
  have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
    using Suc(3) unfolding Until-mltl-ext
  by (metis convert-nnf-ext.simps(8) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(7))
  have  $\beta$ -nnf:  $\exists \varphi$ -init.  $\beta = \text{convert-nnf-ext } \varphi$ -init
    using Suc(3) unfolding Until-mltl-ext
  by (metis convert-nnf-ext.simps(8) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(7))
  have  $\alpha$ -composition: is-composition-MLTL  $\alpha$  and
     $\beta$ -composition: is-composition-MLTL  $\beta$  and
    L-composition: is-composition  $(b-a+1)$  L
  using Suc(4) unfolding Until-mltl-ext is-composition-MLTL.simps by simp-all
  have  $\alpha$ -convert: convert-nnf-ext  $\alpha = \alpha$ 
    using  $\alpha$ -nnf convert-nnf-ext-convert-nnf-ext by metis
  have  $\beta$ -convert: convert-nnf-ext  $\beta = \beta$ 
    using  $\beta$ -nnf convert-nnf-ext-convert-nnf-ext by metis
  have  $\alpha$ -wpd: length  $\pi \geq b + \text{wpd-mltl } (\text{to-mltl } \alpha) - 1$  and
     $\beta$ -wpd: length  $\pi \geq b + \text{wpd-mltl } (\text{to-mltl } \beta)$ 
    using Suc(5) unfolding Until-mltl-ext to-mltl.simps wpd-mltl.simps
    by simp-all
  then have length- $\pi$ -ge-b: length  $\pi > b$ 
    using wpd-geq-one[of to-mltl  $\beta$ ] by auto
  obtain  $\psi$  where  $\psi$ -in:  $\psi \in \text{set } D$ 
    and  $\psi$ -semantics: semantics-mltl-ext  $\pi$   $\psi$ 
    using Suc(7) by blast
  let ?D = LP-mltl-aux  $\beta$  k
  let ?s = interval-times a L
  have length-L:  $1 \leq \text{length } L$ 
    using composition-length-lb[OF L-composition] a-leq-b by linarith
  have sfirst:  $?s!0 = a$ 
    using interval-times-first by simp
  have slast:  $?s!(\text{length } L) = b+1$ 
    using interval-times-last[OF a-leq-b L-composition] by blast
  let ?front = (Until-mltl-list  $\alpha$  ?D (?s!0) (?s!1 - 1) [ $?s!1 - ?s!0$ ])
  let ?back = (concat (map ( $\lambda i$ . And-mltl-list
    [Global-mltl-ext
      ( $?s!0$ ) ( $?s!i - 1$ ) [ $?s!i - ?s!0$ ] (And-mltl-ext  $\alpha$  (Not_c
     $\beta$ ))])
    (Until-mltl-list  $\alpha$  ?D (?s!i) (?s!(i+1) - 1)
      [ $?s!(i+1) - ?s!i$ ] [ $1..<\text{length } L$ ]))
  have D-union: set D = (set ?front)  $\cup$  (set ?back)
    using Suc(6) unfolding Until-mltl-ext LP-mltl-aux.simps
    using  $\alpha$ -convert  $\beta$ -convert list-concat-set-union by metis
  obtain  $\psi$  where  $\psi$ -in:  $\psi \in \text{set } D$  and  $\psi$ -semantics: semantics-mltl-ext  $\pi$   $\psi$ 
    using Suc(7) by blast
  {
    assume *:  $\psi \in \text{set } ?\text{front}$ 
    then obtain y where  $\psi$ -is:  $\psi = \text{Until-mltl-ext } \alpha$  (interval-times a L!0)
      (interval-times a L!1 - 1) [interval-times a L!1 - interval-times

```

$a \ L \ ! \ 0] \ y$
and $y\text{-in: } y \in \text{set } ?D$
by *auto*
have $\text{length-}s: 1 < \text{length } ?s$ **using** $\psi\text{-is}$
by (*metis One-nat-def add commute add-gr-0 add-less-cancel-right L-composition composition-length-lb interval-times-length plus-1-eq-Suc zero-less-one*)
then **have** $\text{length-}L: 1 \leq \text{length } L$
unfolding *interval-times-def*
by (*simp add: less-eq-iff-succ-less*)
have $\text{interval-times } a \ L \ ! \ 1 \leq \text{interval-times } a \ L \ ! \ (\text{length } L)$
using *interval-times-diff-ge-general*[*OF a-leq-b L-composition, of length L 1*
 $?s]$
using $\text{length-}L$ **by** *force*
then **have** $ub: \text{interval-times } a \ L \ ! \ 1 - 1 \leq b$
using *slast* **by** *auto*
obtain l **where** $y\text{-semantics: semantics-mltl-ext } (\text{drop } l \ \pi) \ y$
and $\alpha\text{-global: } (\forall j. \text{interval-times } a \ L \ ! \ 0 \leq j \wedge j < l \longrightarrow$
 $\text{semantics-mltl } (\text{drop } j \ \pi) \ (\text{to-mltl } \alpha))$
and $l\text{-bound: } ?s \ ! \ 0 \leq l \wedge l \leq ?s \ ! \ 1 - 1$
using $\psi\text{-semantics unfolding } \psi\text{-is semantics-mltl-ext-def to-mltl.simps}$
 $\text{semantics-mltl.simps}$
by *blast*
have $l\text{-ab: } a \leq l \wedge l \leq b$
using $l\text{-bound sfirst } ub$ **by** *simp*
have $sem: \exists a \in \text{set } (LP\text{-mltl-aux } \beta \ k). \text{ semantics-mltl-ext } (\text{drop } l \ \pi) \ a$
using $y\text{-in } y\text{-semantics}$ **by** *blast*
have $\text{wpd-mltl } (\text{to-mltl } \beta) \leq \text{length } (\text{drop } l \ \pi)$
using $l\text{-bound length-}\pi\text{-ge-}b \ \beta\text{-wpd } ub$ **by** *auto*
then **have** $ih: \text{ semantics-mltl-ext } (\text{drop } l \ \pi) \ \beta$
using *Suc(1)*[*OF* $\beta\text{-welldef } \beta\text{-nnf } \beta\text{-composition, of drop } l \ \pi \ ?D]$
using sem **by** *blast*
have $\text{ semantics-mltl } (\text{drop } j \ \pi) \ (\text{to-mltl } \alpha)$
if $j\text{-bound: } a \leq j \wedge j < l$ **for** j
using $\alpha\text{-global unfolding sfirst using } j\text{-bound } l\text{-bound } ub$ **by** *blast*
then **have** $(\exists i. (a \leq i \wedge i \leq b) \wedge$
 $\text{ semantics-mltl } (\text{drop } i \ \pi) \ (\text{to-mltl } \beta) \wedge$
 $(\forall j. a \leq j \wedge j < i \longrightarrow$
 $\text{ semantics-mltl } (\text{drop } j \ \pi) \ (\text{to-mltl } \alpha)))$
using $ih \ l\text{-ab unfolding semantics-mltl-ext-def}$ **by** *blast*
then **have** $?thesis$ **unfolding** *Until-mltl-ext semantics-mltl-ext-def to-mltl.simps*
 $\text{ semantics-mltl.simps}$
using $a\text{-leq-}b \ \text{length-}\pi\text{-ge-}b$ **by** *simp*
} **moreover** **{**
assume $*$: $\psi \in \text{set } ?back$
then **obtain** $i \ y$ **where**
 $\psi\text{-is: } \psi = \text{And-mltl-ext } (Global\text{-mltl-ext } (?s!0) \ (?s!i-1) \ [?s!i - ?s!0]) \ (\text{And-mltl-ext}$
 $\alpha \ (\text{Not}_c \ \beta))$
 $(\text{Until-mltl-ext } \alpha \ (?s!i) \ (?s!(i+1)-1) \ [(?s!(i+1)) - (?s!i)]) \ y$
and $i\text{-bound: } 1 \leq i \wedge i < \text{length } L$

```

and  $y$ -in:  $y \in \text{set } ?D$ 
  by auto
have bound1:  $\text{interval-times } a \ L \ ! \ i < \text{interval-times } a \ L \ ! \ (i+1)$ 
  using interval-times-diff-ge[OF a-leq-b L-composition, of i ?s]
  using i-bound by blast
have  $\text{interval-times } a \ L \ ! \ (i + 1) \leq \text{interval-times } a \ L \ ! \ (\text{length } L)$ 
  using interval-times-diff-ge-general[OF a-leq-b L-composition, of length L
 $i+1 \ ?s$ ]
  using i-bound by (metis less-iff-succ-less-eq order-le-less)
then have bound2:  $\text{interval-times } a \ L \ ! \ (i+1) \leq b+1$ 
  using slast by simp
have  $\text{interval-times } a \ L \ ! \ i > \text{interval-times } a \ L \ ! \ 0$ 
  using i-bound interval-times-diff-ge-general[OF a-leq-b L-composition, of i 0
 $?s$ ]
  by auto
then have  $\text{interval-times } a \ L \ ! \ i > 0$ 
  unfolding interval-times-def by simp
then have  $\text{interval-times } a \ L \ ! \ i \leq b$ 
  using bound1 bound2 by simp
have  $\alpha\beta$ -global:  $(\forall ia. a \leq ia \wedge ia \leq \text{interval-times } a \ L \ ! \ i - 1 \longrightarrow$ 
   $\text{semantics-mltl } (\text{drop } ia \ \pi) \ (\text{to-mltl } \alpha) \wedge$ 
   $\neg \text{semantics-mltl } (\text{drop } ia \ \pi) \ (\text{to-mltl } \beta))$ 
  using  $\psi$ -semantics unfolding  $\psi$ -is semantics-mltl-ext-def to-mltl.simps
semantics-mltl.simps
  unfolding sfirst by auto
have until:  $(\exists ia. (\text{interval-times } a \ L \ ! \ i \leq ia \wedge$ 
   $ia \leq \text{interval-times } a \ L \ ! \ (i + 1) - 1) \wedge$ 
   $\text{semantics-mltl } (\text{drop } ia \ \pi) \ (\text{to-mltl } y) \wedge$ 
   $(\forall j. \text{interval-times } a \ L \ ! \ i \leq j \wedge j < ia \longrightarrow$ 
   $\text{semantics-mltl } (\text{drop } j \ \pi) \ (\text{to-mltl } \alpha)))$ 
  using  $\psi$ -semantics unfolding  $\psi$ -is semantics-mltl-ext-def to-mltl.simps
semantics-mltl.simps
  unfolding sfirst by auto
obtain  $l$  where  $y$ -semantics: semantics-mltl-ext  $(\text{drop } l \ \pi) \ y$ 
  and  $\alpha$ -global:  $(\forall j. ?s \ ! \ i \leq j \wedge j < l \longrightarrow$ 
   $\text{semantics-mltl } (\text{drop } j \ \pi) \ (\text{to-mltl } \alpha))$ 
  and  $l$ -bound:  $?s \ ! \ i \leq l \wedge l \leq ?s \ ! \ (i+1) - 1$ 
  using until unfolding semantics-mltl-ext-def by blast
have  $ub$ :  $?s \ ! \ (i+1) - 1 \leq b$ 
  using i-bound bound2 by auto
have  $lb$ :  $a < ?s!i$ 
  using i-bound interval-times-diff-ge-general[OF a-leq-b L-composition, of i 0
 $?s$ ]
  using sfirst by auto
have  $l$ -ab:  $a \leq l \wedge l \leq b$ 
  using  $l$ -bound using  $ub \ lb$  by simp
have sem:  $\exists a \in \text{set } (LP\text{-mltl-aux } \beta \ k). \text{semantics-mltl-ext } (\text{drop } l \ \pi) \ a$ 
  using  $y$ -in  $y$ -semantics by blast
have  $wpd\text{-mltl } (\text{to-mltl } \beta) \leq \text{length } (\text{drop } l \ \pi)$ 

```

```

    using  $\beta$ -wpd l-bound length- $\pi$ -ge-b ub by auto
  then have ih: semantics-mltl-ext (drop l  $\pi$ )  $\beta$ 
    using Suc(1)[OF  $\beta$ -welldef  $\beta$ -nnf  $\beta$ -composition - - sem] by blast
  have l-ab:  $a \leq l \wedge l \leq b$ 
    using l-bound lb ub by simp
  have semantics-mltl (drop j  $\pi$ ) (to-mltl  $\alpha$ )
    if j-bound:  $a \leq j \wedge j < l$  for j
  proof-
    have case1:  $\forall ia. a \leq ia \wedge ia \leq ?s ! i - 1 \longrightarrow$ 
      semantics-mltl (drop ia  $\pi$ ) (to-mltl  $\alpha$ )
      using  $\alpha$  $\beta$ -global by blast
    {
      assume *:  $a \leq j \wedge j \leq ?s ! i - 1$ 
      then have ?thesis
        using case1 by blast
    } moreover {
      assume *:  $?s ! i \leq j \wedge j < l$ 
      then have ?thesis
        using  $\alpha$ -global by blast
    }
    ultimately show ?thesis using j-bound by linarith
  qed
  then have ( $\exists i. (a \leq i \wedge i \leq b) \wedge$ 
    semantics-mltl (drop i  $\pi$ ) (to-mltl  $\beta$ )  $\wedge$ 
    ( $\forall j. a \leq j \wedge j < i \longrightarrow$ 
      semantics-mltl (drop j  $\pi$ ) (to-mltl  $\alpha$ )))
    using ih l-ab semantics-mltl-ext-def by auto
  then have ?thesis unfolding Until-mltl-ext semantics-mltl-ext-def to-mltl.simps
    semantics-mltl.simps
    using a-leq-b length- $\pi$ -ge-b by simp
  }
  ultimately show ?thesis using D-union  $\psi$ -in by blast
next
case (Release-mltl-ext  $\alpha$  a b L  $\beta$ )
have  $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ ) and
   $\beta$ -welldef: intervals-welldef (to-mltl  $\beta$ ) and
  a-leq-b:  $a \leq b$ 
  using Suc(2) unfolding Release-mltl-ext by simp-all
have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
  using Suc(3) unfolding Release-mltl-ext
  by (metis convert-nnf-ext.simps(9) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(8))
have  $\beta$ -nnf:  $\exists \varphi$ -init.  $\beta = \text{convert-nnf-ext } \varphi$ -init
  using Suc(3) unfolding Release-mltl-ext
  by (metis convert-nnf-ext.simps(9) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(8))
have  $\alpha$ -composition: is-composition-MLTL  $\alpha$  and
   $\beta$ -composition: is-composition-MLTL  $\beta$  and
  L-composition: is-composition (b-a+1) L
  using Suc(4) unfolding Release-mltl-ext is-composition-MLTL.simps by
simp-all

```

```

have  $\alpha$ -convert: convert-nnf-ext  $\alpha = \alpha$ 
  using  $\alpha$ -nnf convert-nnf-ext-convert-nnf-ext by metis
have  $\beta$ -convert: convert-nnf-ext  $\beta = \beta$ 
  using  $\beta$ -nnf convert-nnf-ext-convert-nnf-ext by metis
have  $\alpha$ -wpd: length  $\pi \geq b + \text{wpd-mltl}$  (to-mltl  $\alpha$ ) and
   $\beta$ -wpd: length  $\pi \geq b + \text{wpd-mltl}$  (to-mltl  $\beta$ )
  using Suc(5) unfolding Release-mltl-ext to-mltl.simps wpd-mltl.simps
  by simp-all
then have length- $\pi$ -ge- $b$ : length  $\pi > b$ 
  using wpd-geq-one[of to-mltl  $\beta$ ] by auto
obtain  $\psi$  where  $\psi$ -in:  $\psi \in \text{set } D$ 
  and  $\psi$ -semantics: semantics-mltl-ext  $\pi \ \psi$ 
  using Suc(7) by blast
let ?D = LP-mltl-aux  $\alpha \ k$ 
let ?s = interval-times  $a \ L$ 
have length-L:  $1 \leq \text{length } L$ 
  using composition-length-lb[OF L-composition] a-leq-b by linarith
have sfirst: ?s!0 =  $a$ 
  using interval-times-first by simp
have slast: ?s!(length L) =  $b + 1$ 
  using interval-times-last[OF a-leq-b L-composition] by blast
let ?front = set [Global-mltl-ext  $a \ b \ L$  (And-mltl-ext (Notc  $\alpha$ )  $\beta$ )]
let ?middle = set (Mighty-Release-mltl-list ?D  $\beta$  (?s! 0) (?s! 1 - 1)
  [?s! 1 - ?s! 0])
let ?back = set (concat (map ( $\lambda i$ . And-mltl-list
  [Global-mltl-ext
  (?s! 0) (?s! i - 1) [?s!i - ?s!0] (And-mltl-ext (Notc  $\alpha$ )
   $\beta$ )]
  (Mighty-Release-mltl-list ?D  $\beta$  (?s! i)
  (?s! (i + 1) - 1) [?s! (i + 1) - ?s! i]))
  [1..<length L]))
let ?P =  $\lambda j$ . (semantics-mltl (drop j  $\pi$ ) (to-mltl  $\alpha$ )  $\wedge$ 
  ( $\forall k$ .  $a \leq k \wedge k \leq j \longrightarrow$ 
  semantics-mltl (drop k  $\pi$ ) (to-mltl  $\beta$ )))
have D-is: set  $D = ?\text{front} \cup ?\text{middle} \cup ?\text{back}$ 
  unfolding Suc(6) Release-mltl-ext LP-mltl-aux.simps
  using  $\alpha$ -convert list-concat-set-union
  by (metis append-assoc)
have split:  $\psi \in ?\text{front} \cup ?\text{middle} \cup ?\text{back}$ 
  using  $\psi$ -in D-is by blast
{
  assume *:  $\psi \in ?\text{front}$ 
  then have  $\psi$ -is:  $\psi = \text{Global-mltl-ext } a \ b \ L$  (And-mltl-ext (Notc  $\alpha$ )  $\beta$ )
  by auto
  then have ?thesis using  $\psi$ -semantics unfolding  $\psi$ -is
  unfolding Release-mltl-ext semantics-mltl-ext-def to-mltl.simps seman-
  tics-mltl.simps
  by blast
} moreover {

```



```

assume *:  $\psi \in ?middle$ 
then obtain  $x$  where  $\psi$ -is:  $\psi = \text{Mighty-Release-mltl-ext } x \beta a (?s ! 1 - 1)$ 
[ $?s ! 1 - a$ ]
    and  $x$ -in:  $x \in \text{set } ?D$ 
    using sfirst by auto
have welldef:  $a < ?s!1$  using sfirst
    using interval-times-diff-ge[OF a-leq-b L-composition, of 0 ?s]
    using length-L by force
have ub:  $?s!1 \leq b+1$ 
    using length-L slast
    using interval-times-diff-ge-general[OF a-leq-b L-composition, of length L 1
?s]
    by force
obtain  $i$  where  $i$ -bound:  $a \leq i \wedge i \leq \text{interval-times } a L ! 1 - 1$ 
    and  $x$ -semantics: semantics-mltl (drop  $i \pi$ ) (to-mltl  $x$ )
    using  $\psi$ -semantics unfolding  $\psi$ -is Mighty-Release-mltl-ext.simps
    unfolding Release-mltl-ext semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps
    by auto
have wpd-mltl (to-mltl  $\alpha$ )  $\leq \text{length}$  (drop  $i \pi$ )
    using  $\alpha$ -wpd  $i$ -bound ub by auto
then have  $\alpha$ -semantics: semantics-mltl-ext (drop  $i \pi$ )  $\alpha$ 
    using Suc(1)[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition, of drop  $i \pi$  ?D]
    using  $x$ -in  $x$ -semantics unfolding semantics-mltl-ext-def by blast
let  $?globally$ - $\beta = (\forall i. a \leq i \wedge i \leq \text{interval-times } a L ! 1 - 1 \longrightarrow$ 
    semantics-mltl (drop  $i \pi$ ) (to-mltl  $\beta$ ))
let  $?release = (\exists j \geq a. j \leq \text{interval-times } a L ! 1 - 1 - 1 \wedge$ 
    semantics-mltl (drop  $j \pi$ ) (to-mltl  $x$ )  $\wedge$ 
    ( $\forall k. a \leq k \wedge k \leq j \longrightarrow$ 
    semantics-mltl (drop  $k \pi$ ) (to-mltl  $\beta$ )))
have eo:  $?globally$ - $\beta \vee ?release$ 
    using  $\psi$ -semantics unfolding  $\psi$ -is Mighty-Release-mltl-ext.simps
    unfolding Release-mltl-ext semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps
    by auto
    {
    assume **:  $?globally$ - $\beta$ 
    {
    assume interval-times  $a L ! 1 - 1 = b$ 
    then have  $?thesis$  unfolding Release-mltl-ext semantics-mltl-ext-def
to-mltl.simps semantics-mltl.simps
    using ** a-leq-b by simp
    } moreover {
    assume  $s1$ -bound: interval-times  $a L ! 1 - 1 < b$ 
    have semantics-mltl (drop  $k \pi$ ) (to-mltl  $\beta$ )
    if  $k$ -bound:  $a \leq k \wedge k \leq i$  for  $k$ 
    using **  $k$ -bound  $i$ -bound  $s1$ -bound by auto
    then have  $?thesis$  using **  $\alpha$ -semantics  $i$ -bound ub a-leq-b
    unfolding semantics-mltl-ext-def Release-mltl-ext to-mltl.simps semantics-

```

```

tics-mltl.simps
  using s1-bound by force
}
ultimately have ?thesis using ub by linarith
} moreover {
  assume **: ?release
  have bound: interval-times a L ! 1 - 1 - 1 ≤ b-1
  using ub by simp
  then obtain j where sem: a ≤ j ∧ j ≤ interval-times a L ! 1 - 1 - 1 ∧
    semantics-mltl (drop j π) (to-mltl x) ∧
    (∀ k. a ≤ k ∧ k ≤ j →
      semantics-mltl (drop k π) (to-mltl β))
  using ** by blast
  have wpd-mltl (to-mltl α) ≤ length (drop j π)
  using α-wpd sem ub by auto
  then have semantics-mltl (drop j π) (to-mltl α)
  using Suc(1)[OF α-welldef α-nnf α-composition, of drop j π ?D]
  using sem x-in unfolding semantics-mltl-ext-def by blast
  then have (∃ j ≥ a. j ≤ b - 1 ∧
    semantics-mltl (drop j π) (to-mltl α) ∧
    (∀ k. a ≤ k ∧ k ≤ j →
      semantics-mltl (drop k π) (to-mltl β)))
  using sem ub by auto
  then have ?thesis
  unfolding semantics-mltl-ext-def Release-mltl-ext to-mltl.simps seman-
tics-mltl.simps
  using a-leq-b by blast
}
ultimately have ?thesis using eo by blast
} moreover {
  assume *: ψ ∈ ?back
  then obtain i x where ψ-is: ψ = And-mltl-ext
    (Global-mltl-ext
      (interval-times a L ! 0)
      (interval-times a L ! i - 1) [?s!i - ?s!0] (And-mltl-ext (Notc
α) β))
    (Mighty-Release-mltl-ext x β
      (interval-times a L ! i)
      (interval-times a L ! (i + 1) - 1)
      [interval-times a L ! (i + 1) -
        interval-times a L ! i])
  and x-in: x ∈ set ?D
  and i-bound: 1 ≤ i ∧ i < length L
  by auto
  have lb: a < ?s!i
  using interval-times-diff-ge-general[OF a-leq-b L-composition, of i 0 ?s]
  using sfirst i-bound by simp
  have welldef: (interval-times a L ! i) < (interval-times a L ! (i + 1))
  using interval-times-diff-ge[OF a-leq-b L-composition, of i ?s]

```

using *i-bound by simp*
have *ub*: $?s!(i+1) \leq b+1$
using *slast i-bound*
using *interval-times-diff-ge-general*[*OF a-leq-b L-composition, of length L*
i+1 ?s]
by (*metis Orderings.order-eq-iff less-iff-succ-less-eq order-le-imp-less-or-eq*
order-less-imp-le)

have *globally-before*: $\forall ia. \text{interval-times } a \ L \ ! \ 0 \leq ia \wedge ia \leq \text{interval-times } a \ L \ ! \ i - 1 \longrightarrow$
 $\neg \text{semantics-mltl } (\text{drop } ia \ \pi) \ (to\text{-mltl } \alpha) \wedge$
 $\text{semantics-mltl } (\text{drop } ia \ \pi) \ (to\text{-mltl } \beta)$
using *ψ -semantics unfolding ψ -is semantics-mltl-ext-def to-mltl.simps*
semantics-mltl.simps Mighty-Release-mltl-ext.simps
using *length- π -ge-b a-leq-b sfirst by auto*
have *release*: $(\forall ia. \text{interval-times } a \ L \ ! \ i \leq ia \wedge$
 $ia \leq \text{interval-times } a \ L \ ! \ (i + 1) - 1 \longrightarrow$
 $\text{semantics-mltl } (\text{drop } ia \ \pi) \ (to\text{-mltl } \beta)) \vee$
 $(\exists j \geq \text{interval-times } a \ L \ ! \ i.$
 $j \leq \text{interval-times } a \ L \ ! \ (i + 1) - 1 - 1 \wedge$
 $\text{semantics-mltl } (\text{drop } j \ \pi) \ (to\text{-mltl } x) \wedge$
 $(\forall k. \text{interval-times } a \ L \ ! \ i \leq k \wedge k \leq j \longrightarrow$
 $\text{semantics-mltl } (\text{drop } k \ \pi) \ (to\text{-mltl } \beta)))$
using *ψ -semantics unfolding ψ -is semantics-mltl-ext-def to-mltl.simps*
semantics-mltl.simps Mighty-Release-mltl-ext.simps
by *auto*
obtain *ia where ia-bound*: $\text{interval-times } a \ L \ ! \ i \leq ia \wedge$
 $ia \leq \text{interval-times } a \ L \ ! \ (i + 1) - 1$
and *x-semantics*: $\text{semantics-mltl } (\text{drop } ia \ \pi) \ (to\text{-mltl } x)$
using *ψ -semantics unfolding ψ -is semantics-mltl-ext-def to-mltl.simps*
semantics-mltl.simps Mighty-Release-mltl-ext.simps
by *blast*
have *wpd-mltl* $(to\text{-mltl } \alpha) \leq \text{length } (\text{drop } ia \ \pi)$
using *α -wpd ia-bound ub by auto*
then have *α -semantics*: $\text{semantics-mltl } (\text{drop } ia \ \pi) \ (to\text{-mltl } \alpha)$
using *Suc(1)*[*OF α -welldef α -nnf α -composition, of drop ia π ?D*]
using *x-semantics x-in unfolding semantics-mltl-ext-def by blast*
{
assume *global- β* : $(\forall ia. \text{interval-times } a \ L \ ! \ i \leq ia \wedge$
 $ia \leq \text{interval-times } a \ L \ ! \ (i + 1) - 1 \longrightarrow$
 $\text{semantics-mltl } (\text{drop } ia \ \pi) \ (to\text{-mltl } \beta))$
{
assume *eq*: $\text{interval-times } a \ L \ ! \ (i + 1) - 1 = b$
have *semantics-mltl* $(\text{drop } j \ \pi) \ (to\text{-mltl } \beta)$
if *j-bound*: $a \leq j \wedge j \leq b$ **for** *j*
proof –
have *1*: $j \leq \text{interval-times } a \ L \ ! \ i - 1 \implies ?thesis$
using *globally-before j-bound unfolding sfirst by blast*
have *2*: $j \geq \text{interval-times } a \ L \ ! \ i \implies ?thesis$

```

    using global- $\beta$  j-bound eq by blast
  show ?thesis
    using 1 2 by linarith
qed
then have ?thesis
  unfolding Release-mltl-ext semantics-mltl-ext-def to-mltl.simps seman-
tics-mltl.simps
    using a-leq-b by blast
} moreover {
  assume le: interval-times a L ! (i + 1) - 1 < b
  have 1: semantics-mltl (drop k  $\pi$ ) (to-mltl  $\beta$ )
    if k-bound: a  $\leq$  k  $\wedge$  k  $\leq$  ia for k
  proof -
    have 1: k  $\leq$  interval-times a L ! i - 1  $\implies$  ?thesis
      using globally-before k-bound sfirst ia-bound by auto
    have 2: k  $\geq$  interval-times a L ! i  $\implies$  ?thesis
      using global- $\beta$  ia-bound k-bound by auto
    show ?thesis
      using 1 2 by linarith
  qed
  have 2: a  $\leq$  ia  $\wedge$  ia  $\leq$  b - 1
    using ia-bound ub lb le by auto
  then have ( $\exists j \geq a. j \leq b - 1 \wedge$ 
    semantics-mltl (drop j  $\pi$ ) (to-mltl  $\alpha$ )  $\wedge$ 
    ( $\forall k. a \leq k \wedge k \leq j \implies$ 
      semantics-mltl (drop k  $\pi$ ) (to-mltl  $\beta$ )))
    using  $\alpha$ -semantics ia-bound le ub lb welldef 1 2 by blast
  then have ?thesis
    unfolding Release-mltl-ext semantics-mltl-ext-def to-mltl.simps seman-
tics-mltl.simps
      using a-leq-b by auto
}
ultimately have ?thesis using ub by linarith
} moreover {
  assume ( $\exists j \geq$  interval-times a L ! i.
    j  $\leq$  interval-times a L ! (i + 1) - 1 - 1  $\wedge$ 
    semantics-mltl (drop j  $\pi$ ) (to-mltl x)  $\wedge$ 
    ( $\forall k. interval-times a L ! i \leq k \wedge k \leq j \implies$ 
      semantics-mltl (drop k  $\pi$ ) (to-mltl  $\beta$ )))
  then obtain j where j-bound: interval-times a L ! i  $\leq$  j  $\wedge$  j  $\leq$  interval-times
a L ! (i + 1) - 1 - 1
    and x-semantics: semantics-mltl (drop j  $\pi$ ) (to-mltl x)
    and global:  $\forall k. interval-times a L ! i \leq k \wedge k \leq j \implies$ 
      semantics-mltl (drop k  $\pi$ ) (to-mltl  $\beta$ )
  by blast
  have wpd-mltl (to-mltl  $\alpha$ )  $\leq$  length (drop j  $\pi$ )
    using  $\alpha$ -wpd j-bound ub by auto
  then have  $\alpha$ -semantics: semantics-mltl (drop j  $\pi$ ) (to-mltl  $\alpha$ )
    using Suc(1)[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition, of drop j  $\pi$  ?D]

```

```

    using x-in x-antics unfolding semantics-mltl-ext-def by blast
  have g: semantics-mltl (drop k π) (to-mltl β)
  if k-bound: a ≤ k ∧ k ≤ j for k
  proof -
    have 1: k ≤ interval-times a L ! i - 1 ⇒ ?thesis
      using globally-before k-bound sfirst ia-bound by auto
    have 2: k ≥ interval-times a L ! i ⇒ ?thesis
      using global ia-bound k-bound by auto
    show ?thesis
      using 1 2 by linarith
  qed
  have a ≤ j ∧ j ≤ b - 1
    using j-bound ub lb by auto
  then have  $(\exists j \geq a. j \leq b - 1 \wedge$ 
    semantics-mltl (drop j π) (to-mltl α) ∧
     $(\forall k. a \leq k \wedge k \leq j \longrightarrow$ 
      semantics-mltl (drop k π) (to-mltl β)))
    using α-semantics g by blast
  then have ?thesis
    unfolding Release-mltl-ext semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps
    using a-leq-b by blast
}
ultimately have ?thesis using release by blast
}
ultimately show ?thesis using split by blast
qed
qed

```

Top Level Union Theorem lemma *LP-mltl-aux-language-union:*

```

fixes  $\varphi::'a$  mltl-ext and  $k::\text{nat}$  and  $\pi::'a$  set list
assumes intervals-welldef: intervals-welldef (to-mltl φ)
assumes is-nnf: ∃ φ-init. φ = convert-nnf-ext φ-init
assumes trace-length: length π ≥ wpd-mltl (to-mltl φ)
assumes composition: is-composition-MLTL φ
assumes D-is: D = LP-mltl-aux φ k
shows semantics-mltl-ext π φ ↔
   $(\exists \psi \in \text{set } D. \text{ semantics-mltl-ext } \pi \psi)$ 
using assms
using LP-mltl-aux-language-union-converse
using LP-mltl-aux-language-union-forward by fast

```

theorem *LP-mltl-language-union-explicit:*

```

fixes  $\varphi::'a$  mltl-ext and  $k::\text{nat}$  and  $\pi::'a$  set list
assumes intervals-welldef: intervals-welldef (to-mltl φ)
assumes composition: is-composition-MLTL φ
assumes D-is: D = set (LP-mltl φ k)
assumes trace-length: length π ≥ wpd-mltl (to-mltl φ)
shows semantics-mltl-ext π φ ↔  $(\exists \psi \in D. \text{ semantics-mltl } \pi \psi)$ 

```

```

proof–
  have  $D = \text{set } (\text{map } \text{to-mltl } (\text{map } \text{convert-nnf-ext } (\text{LP-mltl-aux } (\text{convert-nnf-ext } \varphi) k)))$ 
    using  $D\text{-is unfolding LP-mltl.simps by blast}$ 
  let  $?D\text{-aux} = \text{LP-mltl-aux } (\text{convert-nnf-ext } \varphi) k$ 
  let  $?\varphi\text{-nnf} = \text{convert-nnf-ext } \varphi$ 
  have  $\text{wpd-decomp: wpd-mltl } \psi \leq \text{wpd-mltl } (\text{to-mltl } \varphi)$ 
    if  $\psi\text{-in} : \psi \in D$  for  $\psi$ 
  proof–
    obtain  $x$  where  $\psi\text{-is: } \psi = \text{to-mltl } (\text{convert-nnf-ext } x)$ 
      and  $x\text{-in: } x \in \text{set } (\text{LP-mltl-aux } (\text{convert-nnf-ext } \varphi) k)$ 
    using  $\psi\text{-in unfolding D-is LP-mltl.simps by auto}$ 
    have  $x\text{phi: wpd-mltl } (\text{to-mltl } x) \leq \text{wpd-mltl } (\text{to-mltl } \varphi)$ 
      using  $\text{LP-mltl-aux-wpd[of } (\text{convert-nnf-ext } \varphi) x k]$ 
    by  $(\text{metis composition convert-nnf-ext-to-mltl-commute intervals-welldef is-composition-convert-nnf-ext}$ 
 $\text{nnf-intervals-welldef wpd-convert-nnf } x\text{-in})$ 
    have  $\text{wpd-mltl } (\text{to-mltl } x) = \text{wpd-mltl } \psi$ 
      unfolding  $\psi\text{-is using convert-nnf-ext-to-mltl-commute}$ 
      by  $(\text{metis wpd-convert-nnf})$ 
    then show  $?thesis$  using  $x\text{phi by auto}$ 
  qed
  have  $\text{len-biconditional: } \bigwedge \pi. \text{length } \pi \geq \text{wpd-mltl } (\text{to-mltl } \varphi) \implies$ 
     $(\text{semantics-mltl } \pi (\text{to-mltl } \varphi) \longleftrightarrow (\exists \psi \in D. \text{semantics-mltl } \pi \psi))$ 
  proof–
    fix  $\pi :: 'a \text{ set list}$ 
    assume  $*$ :  $\text{length } \pi \geq \text{wpd-mltl } (\text{to-mltl } \varphi)$ 
    let  $?thesis = \text{semantics-mltl } \pi (\text{to-mltl } \varphi) \longleftrightarrow$ 
       $(\exists \psi \in D. \text{semantics-mltl } \pi \psi)$ 
    have  $\text{intervals-welldef } (\text{convert-nnf } (\text{to-mltl } \varphi))$ 
      using  $\text{intervals-welldef nnf-intervals-welldef by blast}$ 
    then have  $\text{cond1: intervals-welldef } (\text{to-mltl } (\text{convert-nnf-ext } \varphi))$ 
      by  $(\text{simp add: convert-nnf-ext-to-mltl-commute})$ 
    have  $? \varphi\text{-nnf} = \text{convert-nnf-ext } (? \varphi\text{-nnf})$ 
      using  $\text{convert-nnf-ext-convert-nnf-ext by blast}$ 
    then have  $\text{cond2: } \exists \varphi\text{-init. convert-nnf-ext } \varphi = \text{convert-nnf-ext } \varphi\text{-init}$ 
      by blast
    have  $\text{cond3: wpd-mltl } (\text{to-mltl } (\text{convert-nnf-ext } \varphi)) \leq \text{length } \pi$ 
    proof–
      have  $\text{wpd-mltl } (\text{convert-nnf } (\text{to-mltl } \varphi)) \leq \text{length } \pi$ 
        using  $*$  by  $(\text{simp add: wpd-convert-nnf})$ 
      then show  $?thesis$ 
        using  $\text{convert-nnf-ext-to-mltl-commute by metis}$ 
    qed
    have  $\text{cond4: is-composition-MLTL } (\text{convert-nnf-ext } \varphi)$ 
      using  $\text{composition intervals-welldef is-composition-convert-nnf-ext}$ 
      by blast
    have  $\text{aux-fact: semantics-mltl-ext } \pi (\text{convert-nnf-ext } \varphi) =$ 
 $(\exists \psi \in \text{set } (\text{LP-mltl-aux } (\text{convert-nnf-ext } \varphi) k). \text{semantics-mltl-ext } \pi \psi)$ 
      using  $\text{LP-mltl-aux-language-union[OF cond1 cond2 cond3 cond4] by blast}$ 

```

have forward: $(\exists \psi \in \text{set } (LP\text{-mttl-aux } (\text{convert-nnf-ext } \varphi) k)).$
 $\text{semantics-mttl } \pi (\text{to-mttl } \psi) \implies$
 $(\exists \psi \in \text{set } (\text{map to-mttl}$
 $\quad (\text{map convert-nnf-ext } (LP\text{-mttl-aux } (\text{convert-nnf-ext } \varphi) k))).$
 $\text{semantics-mttl } \pi \psi)$

proof –
assume $\exists \psi \in \text{set } (LP\text{-mttl-aux } (\text{convert-nnf-ext } \varphi) k).$
 $\text{semantics-mttl } \pi (\text{to-mttl } \psi)$
then obtain ψ **where** $*$: $\psi \in \text{set } (LP\text{-mttl-aux } (\text{convert-nnf-ext } \varphi) k)$ **and**
 $**$: $\text{semantics-mttl } \pi (\text{to-mttl } \psi)$

by blast
have in-set: $(\text{to-mttl } (\text{convert-nnf-ext } \psi)) \in \text{set } (\text{map to-mttl}$
 $\quad (\text{map convert-nnf-ext } (LP\text{-mttl-aux } (\text{convert-nnf-ext } \varphi) k)))$
using $*$ **by auto**
have intervals-welldef $(\text{to-mttl } \psi)$
using $\text{intervals-welldef } *$
using $LP\text{-mttl-aux-intervals-welldef}$
using $\text{composition by auto}$
then have semantics-mttl $\pi (\text{convert-nnf } (\text{to-mttl } \psi))$
using $**$ $\text{convert-nnf-preserves-semantics[of to-mttl } \psi \pi]$
by blast
then have semantics: $\text{semantics-mttl } \pi (\text{to-mttl } (\text{convert-nnf-ext } \psi))$
by $(\text{simp add: convert-nnf-ext-to-mttl-commute})$
show ?thesis using in-set semantics by blast

qed
have converse: $(\exists \psi \in \text{set } (\text{map to-mttl}$
 $\quad (\text{map convert-nnf-ext } (LP\text{-mttl-aux } (\text{convert-nnf-ext } \varphi) k))).$
 $\text{semantics-mttl } \pi \psi) \implies (\exists \psi \in \text{set } (LP\text{-mttl-aux } (\text{convert-nnf-ext } \varphi) k)).$
 $\text{semantics-mttl } \pi (\text{to-mttl } \psi))$

proof –
assume $\exists \psi \in \text{set } (\text{map to-mttl}$
 $\quad (\text{map convert-nnf-ext } (LP\text{-mttl-aux } (\text{convert-nnf-ext } \varphi) k))).$
 $\text{semantics-mttl } \pi \psi$
then obtain ψ **where** $*$: $\psi \in \text{set } (\text{map to-mttl}$
 $\quad (\text{map convert-nnf-ext } (LP\text{-mttl-aux } (\text{convert-nnf-ext } \varphi) k)))$
and $**$: $\text{semantics-mttl } \pi \psi$

by blast
obtain $\psi\text{-aux}$ **where** aux-in : $\psi\text{-aux} \in \text{set } (LP\text{-mttl-aux } (\text{convert-nnf-ext } \varphi)$
 $k)$ **and**
 $\text{is-aux: } \psi = \text{to-mttl } (\text{convert-nnf-ext } \psi\text{-aux})$
using $*$ $D\text{-is } LP\text{-mttl-element } \langle D = \text{set } (\text{map to-mttl } (\text{map convert-nnf-ext}$
 $(LP\text{-mttl-aux } (\text{convert-nnf-ext } \varphi) k))) \rangle$ **by blast**
have semantics: $\text{semantics-mttl } \pi (\text{to-mttl } \psi\text{-aux})$
using $**$ **unfolding** is-aux
by $(\text{metis } LP\text{-mttl-aux-intervals-welldef aux-in composition convert-nnf-ext-to-mttl-commute}$
 $\text{convert-nnf-preserves-semantics intervals-welldef})$
show ?thesis using aux-in semantics by blast

qed
have $(\exists \psi \in \text{set } (LP\text{-mttl-aux } (\text{convert-nnf-ext } \varphi) k)).$

```

    semantics-mltl  $\pi$  (to-mltl  $\psi$ ) =
    ( $\exists \psi \in \text{set}$  (map to-mltl
      (map convert-nnf-ext (LP-mltl-aux (convert-nnf-ext  $\varphi$ )  $k$ ))).
      semantics-mltl  $\pi$   $\psi$ )
  using forward converse by blast
  then show ?thesis
  unfolding D-is LP-mltl.simps semantics-mltl-ext-def
  using aux-fact convert-nnf-ext-to-mltl-commute convert-nnf-preserves-semantics
  by (metis intervals-welldef semantics-mltl-ext-def)
qed
show ?thesis
  using len-biconditional[of  $\pi$ ] assms(4)
  unfolding semantics-mltl-ext-def by blast
qed

```

theorem *LP-mltl-language-union:*

```

  fixes  $\varphi::'a$  mltl-ext and  $k::\text{nat}$ 
  assumes intervals-welldef: intervals-welldef (to-mltl  $\varphi$ )
  assumes composition: is-composition-MLTL  $\varphi$ 
  assumes D-is:  $D = \text{set}$  (LP-mltl  $\varphi$   $k$ )
  assumes  $r$ :  $r = \text{wpd-mltl}$  (to-mltl  $\varphi$ )
  shows language-mltl-r (to-mltl  $\varphi$ )  $r$ 
    = ( $\bigcup \psi \in D$ . language-mltl-r  $\psi$   $r$ )
  proof -
  have  $\pi \in \text{language-mltl-r}$  (to-mltl  $\varphi$ )  $r$   $\longleftrightarrow$ 
     $\pi \in (\bigcup \psi \in D$ . language-mltl-r  $\psi$   $r$ )
  if length: length  $\pi \geq r$  for  $\pi$ 
  proof -
  have equiv: ( $\exists \psi \in D$ . semantics-mltl  $\pi$   $\psi$ )  $\longleftrightarrow$   $\pi \in (\bigcup \psi \in D$ . language-mltl-r  $\psi$ 
 $r$ )
  unfolding language-mltl-r-def using length by blast
  have semantics-mltl-ext  $\pi$   $\varphi = (\exists \psi \in D$ . semantics-mltl  $\pi$   $\psi$ )
  using LP-mltl-language-union-explicit[of  $\varphi$   $D$   $k$   $\pi$ ]
  using assms length by blast
  then show ?thesis
  using equiv length
  unfolding language-mltl-r-def semantics-mltl-ext-def by blast
  qed
  then show ?thesis unfolding language-mltl-r-def
  by blast
qed

```

8.3 Disjointness Theorem

lemma *LP-mltl-language-disjoint-aux-helper:*

```

  fixes  $\varphi$   $\psi_1$   $\psi_2::'a$  mltl-ext and  $k::\text{nat}$  and  $\pi::'a$  set list
  assumes intervals-welldef: intervals-welldef (to-mltl  $\varphi$ )
  assumes is-nnf:  $\exists \varphi$ -init.  $\varphi = \text{convert-nnf-ext}$   $\varphi$ -init
  assumes composition-allones: is-composition-MLTL-allones  $\varphi$ 

```



```

assumes tracelen:  $length\ \pi \geq wpd\text{-}mttl\ (to\text{-}mttl\ \varphi)$ 
assumes D-decomp:  $D = set\ (LP\text{-}mttl\text{-}aux\ \varphi\ k)$ 
assumes diff-formulas:  $(\psi1 \in D) \wedge (\psi2 \in D) \wedge \psi1 \neq \psi2$ 
assumes sat1: semantics-mttl-ext  $\pi\ \psi1$ 
assumes sat2: semantics-mttl-ext  $\pi\ \psi2$ 
shows False
using assms
proof(induction k arbitrary: D \varphi \psi1 \psi2 \pi)
  case 0
  then show ?case unfolding LP-mttl.simps LP-mttl-aux.simps
    by auto
next
  case (Suc k)
  then show ?case
  proof(cases \varphi)
    case True-mttl-ext
    then show ?thesis using Suc
      unfolding True-mttl-ext LP-mttl.simps LP-mttl-aux.simps
      by auto
    next
    case False-mttl-ext
    then show ?thesis using Suc
      unfolding False-mttl-ext LP-mttl.simps LP-mttl-aux.simps
      by auto
    next
    case (Prop-mttl-ext p)
    then show ?thesis using Suc
      unfolding Prop-mttl-ext LP-mttl.simps LP-mttl-aux.simps
      by auto
    next
    case (Not-mttl-ext q)
    then have  $\exists p. q = Prop\text{-}mttl\text{-}ext\ p$ 
      using convert-nnf-form-Not-Implies-Prop Suc
    by (metis convert-nnf-ext-to-mttl-commute to-mttl.simps(4) to-mttl-prop-bijective)

    then obtain p where q = Prop-mttl-ext p by blast
    then show ?thesis
      using Suc unfolding Not-mttl-ext LP-mttl.simps LP-mttl-aux.simps
      by auto
    next
    case (And-mttl-ext \alpha \beta)
    let ?Dx = LP-mttl-aux \alpha k
    let ?Dy = LP-mttl-aux \beta k
    obtain x1 y1 where \psi1-is: \psi1 = And-mttl-ext x1 y1
      and x1-in: x1 \in set ?Dx and y1-in: y1 \in set ?Dy
      using And-mttl-list-member Suc.prem
    by (metis (no-types, lifting) And-mttl-ext LP-mttl-aux.simps(6) convert-nnf-ext.simps(4))
    convert-nnf-ext-convert-nnf-ext in-set-member mttl-ext.inject(3)
    obtain x2 y2 where \psi2-is: \psi2 = And-mttl-ext x2 y2

```

```

    and  $x2$ -in:  $x2 \in \text{set } ?Dx$  and  $y2$ -in:  $y2 \in \text{set } ?Dy$ 
    using And-mltl-list-member Suc.prems
  by (metis (no-types, lifting) And-mltl-ext LP-mltl-aux.simps(6) convert-nnf-ext.simps(4))
  convert-nnf-ext-convert-nnf-ext in-set-member mltl-ext.inject(3))
  have  $eo$ :  $x1 \neq x2 \vee y1 \neq y2$ 
    using Suc(7)  $\psi1$ -is  $\psi2$ -is by blast
  have  $\alpha iwd$ : intervals-welldef (to-mltl  $\alpha$ ) and
     $\beta iwd$ : intervals-welldef (to-mltl  $\beta$ )
    using Suc(2) unfolding And-mltl-ext by simp-all
  have  $\alpha nnf$ :  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
    using Suc(3) unfolding And-mltl-ext
  by (metis convert-nnf-ext.simps(4) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(3))
  have  $\beta nnf$ :  $\exists \varphi$ -init.  $\beta = \text{convert-nnf-ext } \varphi$ -init
    using Suc(3) unfolding And-mltl-ext
  by (metis convert-nnf-ext.simps(4) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(3))
  have  $\alpha$ -is-comp-allones: is-composition-MLTL-allones  $\alpha$  and
     $\beta$ -is-comp-allones: is-composition-MLTL-allones  $\beta$ 
    using Suc(4) unfolding And-mltl-ext is-composition-MLTL-allones.simps
  by simp-all
  have  $\alpha$ -is-comp: is-composition-MLTL  $\alpha$ 
    using  $\alpha$ -is-comp-allones allones-implies-is-composition-MLTL
    by blast
  have  $\beta$ -is-comp: is-composition-MLTL  $\beta$ 
    using  $\beta$ -is-comp-allones allones-implies-is-composition-MLTL
    by blast
  have  $\alpha wpd$ : wpd-mltl (to-mltl  $\alpha$ )  $\leq$  length  $\pi$  and
     $\beta wpd$ : wpd-mltl (to-mltl  $\beta$ )  $\leq$  length  $\pi$ 
    using Suc(5) unfolding And-mltl-ext by simp-all
  let  $?r = \text{wpd-mltl (to-mltl } \alpha)$ 
  {
    assume  $xs$ -neq:  $x1 \neq x2$ 
    have  $x1$ -semantics: semantics-mltl-ext  $\pi$   $x1$ 
      using Suc(8) unfolding  $\psi1$ -is semantics-mltl-ext-def by simp
    have  $x2$ -semantics: semantics-mltl-ext  $\pi$   $x2$ 
      using Suc(9) unfolding  $\psi2$ -is semantics-mltl-ext-def by simp
    have  $?thesis$ 
      using Suc(1)[OF  $\alpha iwd$   $\alpha nnf$   $\alpha$ -is-comp-allones  $\alpha wpd$ , of set  $?Dx$   $x1$   $x2$ ]
      using  $\alpha wpd$   $xs$ -neq  $x1$ -in  $x2$ -in  $x1$ -semantics  $x2$ -semantics by blast
  } moreover {
    assume  $ys$ -neq:  $y1 \neq y2$ 
    have  $y1$ -semantics: semantics-mltl-ext  $\pi$   $y1$ 
      using Suc(8) unfolding  $\psi1$ -is semantics-mltl-ext-def by simp
    have  $y2$ -semantics: semantics-mltl-ext  $\pi$   $y2$ 
      using Suc(9) unfolding  $\psi2$ -is semantics-mltl-ext-def by simp
    have  $?thesis$ 
      using Suc(1)[OF  $\beta iwd$   $\beta nnf$   $\beta$ -is-comp-allones  $\beta wpd$ , of set  $?Dy$   $y1$   $y2$ ]
      using  $\beta wpd$   $ys$ -neq  $y1$ -in  $y2$ -in  $y1$ -semantics  $y2$ -semantics by blast
  }
}

```

```

ultimately show ?thesis
  using eo by argo
next
case (Or-mltl-ext  $\alpha$   $\beta$ )
let ?Dx = LP-mltl-aux (convert-nnf-ext  $\alpha$ ) k
let ?Dy = LP-mltl-aux (convert-nnf-ext  $\beta$ ) k
have D-is: D = set ( And-mltl-list ?Dx ?Dy @
  And-mltl-list [Notc  $\alpha$ ] ?Dy @
  And-mltl-list ?Dx [Notc  $\beta$ ])
  using Suc(6) unfolding Or-mltl-ext LP-mltl-aux.simps
  by metis
then have  $\psi 1$ -eo: List.member (And-mltl-list ?Dx ?Dy)  $\psi 1$   $\vee$ 
  List.member (And-mltl-list [Notc  $\alpha$ ] ?Dy)  $\psi 1$   $\vee$ 
  List.member (And-mltl-list ?Dx [Notc  $\beta$ ])  $\psi 1$ 
  using Suc(7) by (simp add: member-def)
have  $\psi 2$ -eo: List.member (And-mltl-list ?Dx ?Dy)  $\psi 2$   $\vee$ 
  List.member (And-mltl-list [Notc  $\alpha$ ] ?Dy)  $\psi 2$   $\vee$ 
  List.member (And-mltl-list ?Dx [Notc  $\beta$ ])  $\psi 2$ 
  using D-is Suc(7) by (simp add: member-def)

have  $\alpha$ -iwd: intervals-welldef (to-mltl  $\alpha$ )
  using Suc(2) unfolding Or-mltl-ext by simp
have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
  using Suc(3) unfolding Or-mltl-ext
by (metis convert-nnf-ext.simps(5) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(4))
have  $\alpha$ -is-comp: is-composition-MLTL-allones  $\alpha$ 
  using Suc(4) unfolding Or-mltl-ext by simp
have  $\alpha$ -wpd: wpd-mltl (to-mltl  $\alpha$ )  $\leq$  length  $\pi$ 
  using Suc(5) unfolding Or-mltl-ext by simp
have  $\alpha$ -conv-same: set (LP-mltl-aux (convert-nnf-ext  $\alpha$ ) k) = set (LP-mltl-aux
 $\alpha$  k)
  by (metis  $\alpha$ -nnf convert-nnf-ext-convert-nnf-ext)

have  $\beta$ -iwd: intervals-welldef (to-mltl  $\beta$ )
  using Suc(2) unfolding Or-mltl-ext
  by simp
have  $\beta$ -nnf:  $\exists \varphi$ -init.  $\beta = \text{convert-nnf-ext } \varphi$ -init
  using Suc(3) unfolding Or-mltl-ext
by (metis convert-nnf-ext.simps(5) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(4))
have  $\beta$ -is-comp: is-composition-MLTL-allones  $\beta$ 
  using Suc(4) unfolding Or-mltl-ext
  by simp
have  $\beta$ -wpd: wpd-mltl (to-mltl  $\beta$ )  $\leq$  length  $\pi$ 
  using Suc(5) unfolding Or-mltl-ext by simp
have  $\beta$ -conv-same: set (LP-mltl-aux (convert-nnf-ext  $\beta$ ) k) = set (LP-mltl-aux
 $\beta$  k)
  by (metis  $\beta$ -nnf convert-nnf-ext-convert-nnf-ext)

{

```

```

assume List.member (And-mltl-list ?Dx ?Dy)  $\psi_1$ 
then obtain x1 y1 where  $\psi_1$ -is:  $\psi_1 = \text{And-mltl-ext } x1 \ y1$ 
and  $x1y1$ : ( $x1 \in \text{set } ?Dx \wedge y1 \in \text{set } ?Dy$ )
using And-mltl-list-member
by (metis in-set-member)
have x1-semantics: semantics-mltl-ext  $\pi \ x1$  and
y1-semantics: semantics-mltl-ext  $\pi \ y1$ 
using Suc(8) unfolding semantics-mltl-ext-def  $\psi_1$ -is by simp-all
have  $\alpha$ -semantics: semantics-mltl-ext  $\pi \ \alpha$  using LP-mltl-aux-language-union-converse
by (metis  $\alpha$ -wpd  $\alpha$ -is-comp  $\alpha$ -iwd  $\alpha$ -nnf allones-implies-is-composition-MLTL
convert-nnf-ext-convert-nnf-ext x1-semantics x1y1)
have  $\beta$ -semantics: semantics-mltl-ext  $\pi \ \beta$  using LP-mltl-aux-language-union-converse
by (metis  $\beta$ -wpd  $\beta$ -is-comp  $\beta$ -iwd  $\beta$ -nnf allones-implies-is-composition-MLTL
convert-nnf-ext-convert-nnf-ext x1y1 y1-semantics)

```

```

{
assume List.member (And-mltl-list ?Dx ?Dy)  $\psi_2$ 
then obtain x2 y2 where  $\psi_2$ -is:  $\psi_2 = \text{And-mltl-ext } x2 \ y2$ 
and  $x2y2$ : ( $x2 \in \text{set } ?Dx \wedge y2 \in \text{set } ?Dy$ )
using And-mltl-list-member
by (metis in-set-member)
have x2-semantics: semantics-mltl-ext  $\pi \ x2$  and
y2-semantics: semantics-mltl-ext  $\pi \ y2$ 
using Suc(9) unfolding semantics-mltl-ext-def  $\psi_2$ -is by simp-all
have xs-ys-eo:  $x1 \neq x2 \vee y1 \neq y2$ 
using  $x1y1 \ x2y2$  Suc(7)  $\psi_1$ -is  $\psi_2$ -is by blast
have xs-neq:  $x1 \neq x2 \implies \text{False}$ 
using Suc(1)[OF  $\alpha$ -iwd  $\alpha$ -nnf  $\alpha$ -is-comp  $\alpha$ -wpd  $\alpha$ -conv-same, of  $x1 \ x2$ ]
using  $x1y1 \ x2y2 \ x1$ -semantics  $x2$ -semantics by blast
have ys-neq:  $y1 \neq y2 \implies \text{False}$ 
using Suc(1)[OF  $\beta$ -iwd  $\beta$ -nnf  $\beta$ -is-comp  $\beta$ -wpd  $\beta$ -conv-same, of  $y1 \ y2$ ]
using  $x1y1 \ x2y2 \ y1$ -semantics  $y2$ -semantics by blast
have ?thesis
using xs-neq ys-neq xs-ys-eo by blast
} moreover {
assume List.member (And-mltl-list [Notc  $\alpha$ ] ?Dy)  $\psi_2$ 
then obtain x2 y2 where  $\psi_2$ -is:  $\psi_2 = \text{And-mltl-ext } x2 \ y2$ 
and  $x2y2$ : ( $x2 = \text{Not}_c \ \alpha \wedge y2 \in \text{set } ?Dy$ )
using And-mltl-list-member
by (metis member-def member-rec(1) member-rec(2))
have x2-is:  $x2 = \text{Not}_c \ \alpha$ 
using  $x2y2$  by auto
have x2-semantics: semantics-mltl-ext  $\pi \ x2$  and
y2-semantics: semantics-mltl-ext  $\pi \ y2$ 
using Suc(9) unfolding semantics-mltl-ext-def  $\psi_2$ -is by simp-all
have xs-ys-eo:  $x1 \neq x2 \vee y1 \neq y2$ 
using  $x1y1 \ x2y2 \ Suc(7)$   $\psi_1$ -is  $\psi_2$ -is by blast
have ?thesis
using  $\alpha$ -semantics  $x2$ -semantics unfolding  $x2$ -is semantics-mltl-ext-def

```

```

    by simp
  } moreover {
    assume List.member (And-mltl-list ?Dx [Notc β]) ψ2
    then obtain x2 y2 where ψ2-is: ψ2 = And-mltl-ext x2 y2
      and x2y2: (x2 ∈ set ?Dx ∧ y2 = Notc β)
      using And-mltl-list-member
      by (metis member-def member-rec(1) member-rec(2))
    have y2-is: y2 = Notc β
      using x2y2 by auto
    have x2-antics: semantics-mltl-ext π x2 and
      y2-antics: semantics-mltl-ext π y2
      using Suc(9) unfolding semantics-mltl-ext-def ψ2-is by simp-all
    have xs-ys-eo: x1 ≠ x2 ∨ y1 ≠ y2
      using x1y1 x2y2 Suc(7) ψ1-is ψ2-is by blast
    have ?thesis
      using β-antics y2-antics unfolding y2-is semantics-mltl-ext-def
      by simp
  }
  ultimately have ?thesis
    using ψ2-eo by argo
} moreover {
  assume List.member (And-mltl-list [Notc α] ?Dy) ψ1
  then obtain x1 y1 where ψ1-is: ψ1 = And-mltl-ext x1 y1
    and x1y1: (x1 = Notc α ∧ y1 ∈ set ?Dy)
    using And-mltl-list-member
    by (metis member-def member-rec(1) member-rec(2))
  have x1-antics: semantics-mltl-ext π x1 and
    y1-antics: semantics-mltl-ext π y1
    using Suc(8) unfolding semantics-mltl-ext-def ψ1-is by simp-all
  have x1-is: x1 = Notc α
    using x1y1 by auto
  have not-α-antics: ¬semantics-mltl-ext π α
    using x1y1 x1-antics unfolding semantics-mltl-ext-def by auto
  have β-antics: semantics-mltl-ext π β using LP-mltl-aux-language-union-converse
  by (metis β-wpd β-is-comp β-iwd β-nnf allones-implies-is-composition-MLTL
  convert-nnf-ext-convert-nnf-ext x1y1 y1-antics)

  {
    assume List.member (And-mltl-list ?Dx ?Dy) ψ2
    then obtain x2 y2 where ψ2-is: ψ2 = And-mltl-ext x2 y2
      and x2y2: (x2 ∈ set ?Dx ∧ y2 ∈ set ?Dy)
      using And-mltl-list-member
      by (metis in-set-member)
    have x1-antics: semantics-mltl-ext π x2
      using Suc(9) unfolding ψ2-is semantics-mltl-ext-def to-mltl.simps by
simp
    have semantics-mltl-ext π α
      using LP-mltl-aux-language-union-converse
    by (metis α-wpd α-is-comp α-iwd α-nnf allones-implies-is-composition-MLTL

```

```

convert-nnf-ext-convert-nnf-ext x1-semantics x2y2)
  then have ?thesis using not- $\alpha$ -semantics by blast
} moreover {
  assume List.member (And-mltl-list [Notc  $\alpha$ ] ?Dy)  $\psi$ 2
  then obtain x2 y2 where  $\psi$ 2-is:  $\psi$ 2 = And-mltl-ext x2 y2
    and x2y2: (x2 = Notc  $\alpha$   $\wedge$  y2  $\in$  set ?Dy)
    using And-mltl-list-member
    by (metis member-def member-rec(1) member-rec(2))

  have y2-semantics: semantics-mltl-ext  $\pi$  y2
    using Suc(9) unfolding  $\psi$ 2-is semantics-mltl-ext-def to-mltl.simps by
simp
  have ys-neg: y1  $\neq$  y2
    using x1y1 x2y2 Suc(7)  $\psi$ 1-is  $\psi$ 2-is by blast
  then have ?thesis
    using Suc(1)
    using  $\beta$ -wpd  $\beta$ -conv-same  $\beta$ -is-comp  $\beta$ -iwd  $\beta$ -nnf x1y1 x2y2 y1-semantics
y2-semantics by blast
} moreover {
  assume List.member (And-mltl-list ?Dx [Notc  $\beta$ ])  $\psi$ 2
  then obtain x2 y2 where  $\psi$ 2-is:  $\psi$ 2 = And-mltl-ext x2 y2
    and x2y2: (x2  $\in$  set ?Dx  $\wedge$  y2 = Notc  $\beta$ )
    using And-mltl-list-member
    by (metis member-def member-rec(1) member-rec(2))
  have x2-semantics: semantics-mltl-ext  $\pi$  x2
    using Suc(9) unfolding  $\psi$ 2-is semantics-mltl-ext-def to-mltl.simps by
simp
  have ?thesis
    by (metis LP-mltl-aux-language-union-converse  $\alpha$ -wpd  $\alpha$ -is-comp
 $\alpha$ -iwd  $\alpha$ -nnf allones-implies-is-composition-MLTL convert-nnf-ext-convert-nnf-ext
not- $\alpha$ -semantics x2-semantics x2y2)
}
ultimately have ?thesis
  using  $\psi$ 2-eo by argo
} moreover {
  assume List.member (And-mltl-list ?Dx [Notc  $\beta$ ])  $\psi$ 1
  then obtain x1 y1 where  $\psi$ 1-is:  $\psi$ 1 = And-mltl-ext x1 y1
    and x1y1: (x1  $\in$  set ?Dx  $\wedge$  y1 = Notc  $\beta$ )
    using And-mltl-list-member
    by (metis member-def member-rec(1) member-rec(2))
  have x1-semantics: semantics-mltl-ext  $\pi$  x1 and
    y1-semantics: semantics-mltl-ext  $\pi$  y1
    using Suc(8) unfolding semantics-mltl-ext-def  $\psi$ 1-is by simp-all
  have x1-is: y1 = Notc  $\beta$ 
    using x1y1 by auto
  have not- $\beta$ -semantics:  $\neg$ semantics-mltl-ext  $\pi$   $\beta$ 
    using x1y1 y1-semantics unfolding semantics-mltl-ext-def by auto
  have  $\alpha$ -semantics: semantics-mltl-ext  $\pi$   $\alpha$  using LP-mltl-aux-language-union-converse
  by (metis  $\alpha$ -wpd  $\alpha$ -is-comp  $\alpha$ -iwd  $\alpha$ -nnf allones-implies-is-composition-MLTL

```

```

convert-nnf-ext-convert-nnf-ext x1-semantics x1y1)

{
  assume List.member (And-mltl-list ?Dx ?Dy)  $\psi_2$ 
  then obtain x2 y2 where  $\psi_2$ -is:  $\psi_2 = \text{And-mltl-ext } x2 \ y2$ 
    and x2y2:  $(x2 \in \text{set } ?Dx \wedge y2 \in \text{set } ?Dy)$ 
    using And-mltl-list-member
    by (metis in-set-member)
  have semantics-mltl-ext  $\pi \ y2$ 
    using Suc(9) unfolding  $\psi_2$ -is semantics-mltl-ext-def to-mltl.simps by
auto
    then have  $\beta$ -semantics: semantics-mltl-ext  $\pi \ \beta$ 
      using LP-mltl-aux-language-union-converse
    by (metis  $\beta$ -wpd  $\beta$ -is-comp  $\beta$ -iwd  $\beta$ -nnf allones-implies-is-composition-MLTL
convert-nnf-ext-convert-nnf-ext x2y2)
    then have ?thesis
      by (simp add: not- $\beta$ -semantics)
} moreover {
  assume List.member (And-mltl-list [Notc  $\alpha$ ] ?Dy)  $\psi_2$ 
  then obtain x2 y2 where  $\psi_2$ -is:  $\psi_2 = \text{And-mltl-ext } x2 \ y2$ 
    and x2y2:  $(x2 = \text{Not}_c \ \alpha \wedge y2 \in \text{set } ?Dy)$ 
    using And-mltl-list-member
    by (metis member-def member-rec(1) member-rec(2))
  have semantics-mltl-ext  $\pi \ y2$ 
    using Suc(9) unfolding  $\psi_2$ -is semantics-mltl-ext-def to-mltl.simps by
auto
    then have  $\beta$ -semantics: semantics-mltl-ext  $\pi \ \beta$ 
      using LP-mltl-aux-language-union-converse
    by (metis  $\beta$ -wpd  $\beta$ -is-comp  $\beta$ -iwd  $\beta$ -nnf allones-implies-is-composition-MLTL
convert-nnf-ext-convert-nnf-ext x2y2)
    then have ?thesis
      by (simp add: not- $\beta$ -semantics)
} moreover {
  assume List.member (And-mltl-list ?Dx [Notc  $\beta$ ])  $\psi_2$ 
  then obtain x2 y2 where  $\psi_2$ -is:  $\psi_2 = \text{And-mltl-ext } x2 \ y2$ 
    and x2y2:  $(x2 \in \text{set } ?Dx \wedge y2 = \text{Not}_c \ \beta)$ 
    using And-mltl-list-member
    by (metis member-def member-rec(1) member-rec(2))
  have semantics-mltl-ext  $\pi \ x2$ 
    using Suc(9) unfolding  $\psi_2$ -is semantics-mltl-ext-def to-mltl.simps by
auto
    then have ?thesis
      using Suc.IH Suc.prem(6)  $\alpha$ -wpd  $\alpha$ -conv-same  $\alpha$ -is-comp  $\alpha$ -iwd  $\alpha$ -nnf
 $\psi_1$ -is  $\psi_2$ -is x1-semantics x1y1 x2y2 by blast
}
ultimately have ?thesis
  using  $\psi_2$ -eo by argo
}
ultimately show ?thesis

```

```

    using  $\psi1\text{-eo}$  by argo
  next
  case (Future-mltl-ext a b L  $\alpha$ )
  have a-leq-b:  $a \leq b$  and
     $\alpha\text{-welldef}$ : intervals-welldef (to-mltl  $\alpha$ )
  using Suc(2) unfolding intervals-welldef.simps Future-mltl-ext to-mltl.simps
    by simp-all
  have  $\alpha\text{-nnf}$ :  $\exists \varphi\text{-init}. \alpha = \text{convert-nnf-ext } \varphi\text{-init}$ 
    using Suc(3) unfolding Future-mltl-ext
  by (metis convert-nnf-ext.simps(6) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(5))

  have  $\alpha\text{-convert}$ :  $\text{convert-nnf-ext } \alpha = \alpha$ 
    using  $\alpha\text{-nnf}$  convert-nnf-ext-convert-nnf-ext by metis
  have  $\alpha\text{-composition-allones}$ : is-composition-MLTL-allones  $\alpha$  and
    L-composition-allones: is-composition-allones (b-a+1) L
    using Future-mltl-ext Suc.prem(3) by simp-all
  have  $\alpha\text{-composition}$ : is-composition-MLTL  $\alpha$ 
    using Future-mltl-ext Suc.prem(3) allones-implies-is-composition-MLTL
  by auto
  have L-composition: is-composition (b-a+1) L
    using Future-mltl-ext Suc.prem(3) allones-implies-is-composition-MLTL
  is-composition-MLTL.simps(5) by blast
  have  $\alpha\text{-wpd}$ :  $b + \text{wpd-mltl} (\text{to-mltl } \alpha) \leq \text{length } \pi$ 
    using Suc(5) unfolding Future-mltl-ext to-mltl.simps wpd-mltl.simps
    by auto
  let ?D = LP-mltl-aux  $\alpha$  k
  let ?s = interval-times a L
  have length-L:  $1 \leq \text{length } L$ 
    using composition-length-lb[OF L-composition] a-leq-b by linarith
  have length-L-allones:  $\text{length } L = b - a + 1$ 
    using L-composition-allones
  by (simp add: length-is-composition-allones)
  have sfirst:  $?s!0 = a$ 
    using interval-times-first by simp
  have slast:  $?s!(\text{length } L) = b + 1$ 
    using interval-times-last[OF a-leq-b L-composition] by blast
  have length-s:  $\text{length } ?s = \text{length } L + 1$ 
    using interval-times-length by simp
  let ?front = set (Future-mltl-list ?D (?s!0) (?s!1 - 1) [?s!1 - ?s!0])
  let ?back = set (concat (map ( $\lambda i. \text{And-mltl-list}$ 
    [Global-mltl-ext (?s!0) (?s!i - 1) [?s!i - ?s!0] (Notc  $\alpha$ )]
    (Future-mltl-list ?D (?s!i) (?s!(i + 1) - 1) [?s!(i +
  1) - ?s!i]))
    [1.. $\text{length } L$ ]))
  have D-is:  $D = ?\text{front} \cup ?\text{back}$ 
    using Suc(6) unfolding Future-mltl-ext LP-mltl-aux.simps to-mltl.simps
    using  $\alpha\text{-convert}$  list-concat-set-union by metis
  have s1:  $?s!1 = a + 1$ 
    using interval-times-allones[OF a-leq-b L-composition-allones] length-s

```



```

length-L
  by force
  have dropa-wpd: wpd-mltl (to-mltl  $\alpha$ )  $\leq$  length (drop a  $\pi$ )
  using  $\alpha$ -wpd a-leq-b by simp
  {
    assume *:  $\psi 1 \in ?front$ 
    obtain  $x1$  where  $\psi 1$ -is:  $\psi 1 = \text{Future-mltl-ext } a \ a \ [1] \ x1$ 
      and  $x1$ -in:  $x1 \in \text{set } ?D$ 
      using * unfolding  $sfirst \ s1 \ \text{Future-mltl-list.simps}$  by auto
    have  $x1$ -semantics: semantics-mltl-ext (drop a  $\pi$ )  $x1$ 
      using Suc(8) unfolding  $\psi 1$ -is semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps
      by auto
    have  $\alpha$ -semantics: semantics-mltl-ext (drop a  $\pi$ )  $\alpha$ 
      using LP-mltl-aux-language-union-converse[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition dropa-wpd, of ?D k]
      using  $x1$ -semantics  $x1$ -in by blast
    {
      assume **:  $\psi 2 \in ?front$ 
      obtain  $x2$  where  $\psi 2$ -is:  $\psi 2 = \text{Future-mltl-ext } a \ a \ [1] \ x2$ 
        and  $x2$ -in:  $x2 \in \text{set } ?D$ 
        using ** unfolding  $sfirst \ s1 \ \text{Future-mltl-list.simps}$  by auto
      have  $x2$ -semantics: semantics-mltl-ext (drop a  $\pi$ )  $x2$ 
        using Suc(9) unfolding  $\psi 2$ -is semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps
        by auto
      have  $xs$ -neg:  $x1 \neq x2$ 
        using Suc(7) unfolding  $\psi 1$ -is  $\psi 2$ -is by blast
      have ?thesis using dropa-wpd
        using Suc(1)[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition-allones, of drop a  $\pi$  set ?D  $x1 \ x2$ ]
        using  $xs$ -neg  $x1$ -in  $x2$ -in  $x1$ -semantics  $x2$ -semantics by blast
    } moreover {
      assume **:  $\psi 2 \in ?back$ 
      then obtain  $i$  where  $\psi 2$ -is:  $\psi 2 \in \text{set } ((\text{And-mltl-list } [\text{Global-mltl-ext } (?s \ 0) (?s \ i - 1) [?s!i - ?s!0] (\text{Not}_c \ \alpha)] (\text{Future-mltl-list } ?D (?s \ i) (?s \ (i + 1) - 1) [?s \ (i + 1) - ?s \ i]))$ 
        and  $i$ -bound:  $1 \leq i \wedge i < \text{length } L$ 
        by force
      have  $si$ :  $?s!i = a+i$ 
        using interval-times-allones
        using L-composition-allones a-leq-b  $i$ -bound length-s by auto
      have  $si1$ :  $?s!(i+1) = a+i+1$ 
        using interval-times-allones
        using L-composition-allones a-leq-b  $i$ -bound length-s by auto
      obtain  $x2$  where  $\psi 2$ -is:  $\psi 2 = \text{And-mltl-ext } (\text{Global-mltl-ext } a \ (a+i-1) [i] (\text{Not}_c \ \alpha))$ 
        (Future-mltl-ext (a+i) (a+i) [1]  $x2$ )

```

and $x2\text{-in}: x2 \in \text{set } ?D$
using $\psi2\text{-is } si\ si1\ sfirst$ **by** *auto*
then have $?thesis$ **using** $Suc(9)$ **unfolding** $\psi2\text{-is semantics-mltl-ext-def}$
to-mltl.simps semantics-mltl.simps
using $i\text{-bound } \alpha\text{-wpd}$
by (*metis* $\alpha\text{-semantics wpd-geq-one drop-eq-Nil2 dropa-wpd eq-imp-le}$
 $le\text{-neq-implies-less length-0-conv less-nat-zero-code not-one-le-zero semantics-mltl-ext-def}$)

}
ultimately have $?thesis$
using $Suc(7)$ $D\text{-is}$ **by** *blast*
} **moreover** **{**
assume $*$: $\psi1 \in ?back$
then obtain $i1$ **where** $\psi1\text{-is}: \psi1 \in \text{set } ((\text{And-mltl-list}$
 $[\text{Global-mltl-ext } (?s ! 0) (?s ! i1 - 1) [?s!i1 - ?s!0] (\text{Not}_c$
 $\alpha)])$
 $(\text{Future-mltl-list } ?D (?s ! i1) (?s ! (i1 + 1) - 1) [?s ! (i1$
 $+ 1) - ?s ! i1]))))$
and $i1\text{-bound}: 1 \leq i1 \wedge i1 < \text{length } L$
by *force*
have $si1: ?s!i1 = a+i1$
using $interval\text{-times-allones}$
using $L\text{-composition-allones } a\text{-leq-b } i1\text{-bound length-s}$ **by** *auto*
have $si'1: ?s!(i1+1) = a+i1+1$
using $interval\text{-times-allones}$
using $L\text{-composition-allones } a\text{-leq-b } i1\text{-bound length-s}$ **by** *auto*
obtain $x1$ **where** $\psi1\text{-is}: \psi1 = \text{And-mltl-ext } (\text{Global-mltl-ext } a (a+i1-1)$
 $[?s!i1 - ?s!0] (\text{Not}_c \alpha))$
 $(\text{Future-mltl-ext } (a+i1) (a+i1) [1] x1)$
and $x1\text{-in}: x1 \in \text{set } ?D$
using $\psi1\text{-is } si1\ si'1\ sfirst$ **by** *auto*
have $\text{not-}\alpha\text{-semantics}: \neg \text{semantics-mltl-ext } (\text{drop } a\ \pi)\ \alpha$
using $Suc(8)$ **unfolding** $\psi1\text{-is semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps}$
by *auto*
{
assume $**$: $\psi2 \in ?front$
obtain $x2$ **where** $\psi2\text{-is}: \psi2 = \text{Future-mltl-ext } a\ a\ [1]\ x2$
and $x2\text{-in}: x2 \in \text{set } ?D$
using $**$ **unfolding** $sfirst\ s1\ \text{Future-mltl-list.simps}$ **by** *auto*
have $x2\text{-semantics}: \text{semantics-mltl-ext } (\text{drop } a\ \pi)\ x2$
using $Suc(9)$ **unfolding** $\psi2\text{-is semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps}$
by *auto*
have $\alpha\text{-semantics}: \text{semantics-mltl-ext } (\text{drop } a\ \pi)\ \alpha$
using $LP\text{-mltl-aux-language-union-converse}[OF\ \alpha\text{-welldef } \alpha\text{-nnf } \alpha\text{-composition}$
 $\text{dropa-wpd, of } ?D\ k]$
using $x2\text{-semantics } x2\text{-in}$ **by** *blast*
then have $?thesis$ **using** $\text{not-}\alpha\text{-semantics}$ **by** *blast*

```

} moreover {
  assume **:  $\psi 2 \in ?back$ 
  then obtain  $i2$  where  $\psi 2$ -is:  $\psi 2 \in set ((And\text{-mttl}\text{-list}$ 
    [ $Global\text{-mttl}\text{-ext} (?s ! 0) (?s ! i2 - 1) [?s!i2 - ?s!0] (Not_c$ 
 $\alpha)$ ]
    ( $Future\text{-mttl}\text{-list} ?D (?s ! i2) (?s ! (i2 + 1) - 1) [?s ! (i2$ 
 $+ 1) - ?s ! i2]))$ )
    and  $i2$ -bound:  $1 \leq i2 \wedge i2 < length L$ 
    by force
    have  $si2$ :  $?s!i2 = a+i2$ 
    using interval-times-allones
    using L-composition-allones a-leq-b i2-bound length-s by auto
    have  $si'2$ :  $?s!(i2+1) = a+i2+1$ 
    using interval-times-allones
    using L-composition-allones a-leq-b i2-bound length-s by auto
    obtain  $x2$  where  $\psi 2$ -is:  $\psi 2 = And\text{-mttl}\text{-ext} (Global\text{-mttl}\text{-ext} a (a+i2-1)$ 
 $[?s!i2 - ?s!0] (Not_c \alpha)$ 
    ( $Future\text{-mttl}\text{-ext} (a+i2) (a+i2) [1] x2)$ )
    and  $x2$ -in:  $x2 \in set ?D$ 
    using  $\psi 2$ -is  $si2$   $si'2$  sfirst by auto
    have  $x1$ -semantics: semantics-mttl-ext ( $drop (a+i1) \pi$ )  $x1$ 
    using Suc(8) unfolding  $\psi 1$ -is semantics-mttl-ext-def to-mttl.simps
semantics-mttl.simps
    using  $i1$ -bound  $\alpha$ -wpd by auto
    have  $wpd$ -mttl ( $to\text{-mttl} \alpha$ )  $\leq length (drop (a + i1) \pi)$ 
    using  $i1$ -bound unfolding length-L-allones
    using a-leq-b  $\alpha$ -wpd by auto
    then have  $\alpha$ -semantics: semantics-mttl-ext ( $drop (a+i1) \pi$ )  $\alpha$ 
    using LP-mttl-aux-language-union-converse[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition,
of drop (a+i1)  $\pi$  ?D k]
    using  $x1$ -semantics  $x1$ -in by blast
    have  $x2$ -semantics: semantics-mttl-ext ( $drop (a+i2) \pi$ )  $x2$ 
    using Suc(9) unfolding  $\psi 2$ -is semantics-mttl-ext-def to-mttl.simps
semantics-mttl.simps
    using  $i2$ -bound  $\alpha$ -wpd by auto
    have  $wpd$ -mttl ( $to\text{-mttl} \alpha$ )  $\leq length (drop (a + i2) \pi)$ 
    using  $i2$ -bound unfolding length-L-allones
    using a-leq-b  $\alpha$ -wpd by auto
    then have  $\alpha$ -semantics2: semantics-mttl-ext ( $drop (a+i2) \pi$ )  $\alpha$ 
    using LP-mttl-aux-language-union-converse[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition,
of drop (a+i2)  $\pi$  ?D k]
    using  $x2$ -semantics  $x2$ -in by blast
    {
    assume  $i1$ -eq- $i2$ :  $i1 = i2$ 
    have  $wpd$ :  $wpd\text{-mttl} (to\text{-mttl} \alpha) \leq length (drop (a + i1) \pi)$ 
    using  $i1$ -bound  $\alpha$ -wpd a-leq-b unfolding length-L-allones by auto
    have  $x1 \neq x2$ 
    using  $i1$ -eq- $i2$   $\psi 1$ -is  $\psi 2$ -is Suc(7) by blast
    then have ?thesis

```

```

      using Suc(1)[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition-allones, of drop (a+i1)
 $\pi$  set ?D x1 x2]
      using x1-in x1-antics x2-in x2-antics wpd i1-eq-i2 by blast
    } moreover {
      assume i1-le-i2: i1 < i2
      then have a ≤ a+i1 ∧ a+i1 ≤ a + i2 - 1
        by simp
      then have x1-antics: ¬semantics-mltl-ext (drop (a+i1)  $\pi$ )  $\alpha$ 
        using Suc(9) unfolding  $\psi$ 2-is semantics-mltl-ext-def to-mltl.simps
semantics-mltl.simps
      using i2-bound  $\alpha$ -wpd a-leq-b by auto
      then have ?thesis using  $\alpha$ -semantics by blast
    } moreover {
      assume i1-ge-i2: i1 > i2
      then have a ≤ a+i2 ∧ a+i2 ≤ a + i1 - 1
        by simp
      then have x2-antics: ¬semantics-mltl-ext (drop (a+i2)  $\pi$ )  $\alpha$ 
        using Suc(8) unfolding  $\psi$ 1-is semantics-mltl-ext-def to-mltl.simps
semantics-mltl.simps
      using i1-bound  $\alpha$ -wpd a-leq-b by auto
      then have ?thesis using  $\alpha$ -semantics2 by blast
    }
  } ultimately have ?thesis by linarith
}
ultimately have ?thesis
  using Suc(7) D-is by blast
}
ultimately show ?thesis
  using Suc(7) D-is by blast
next
case (Global-mltl-ext a b L  $\alpha$ )
have a-leq-b: a ≤ b and
   $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ )
using Suc(2) unfolding intervals-welldef.simps Global-mltl-ext to-mltl.simps
  by simp-all
have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha$  = convert-nnf-ext  $\varphi$ -init
  using Suc(3) unfolding Global-mltl-ext
by (metis convert-nnf-ext.simps(7) convert-nnf-ext-convert-nnf-ext mtl-ext.inject(6))

have  $\alpha$ -convert: convert-nnf-ext  $\alpha$  =  $\alpha$ 
  using  $\alpha$ -nnf convert-nnf-ext-convert-nnf-ext by metis
have  $\alpha$ -composition-allones: is-composition-MLTL-allones  $\alpha$ 
  using Global-mltl-ext Suc.prem(3) by simp-all
have  $\alpha$ -composition: is-composition-MLTL  $\alpha$ 
  using Global-mltl-ext Suc.prem(3) allones-implies-is-composition-MLTL by
auto
have  $\alpha$ -wpd: b + wpd-mltl (to-mltl  $\alpha$ ) ≤ length  $\pi$ 
  using Suc(5) unfolding Global-mltl-ext to-mltl.simps wpd-mltl.simps
  by auto

```

```

let ?D = LP-mltl-aux α k
{
  assume *: length ?D ≤ 1
  then have D-is: D = {Global-mltl-ext a b L α}
    using Suc(6) unfolding Global-mltl-ext LP-mltl-aux.simps
    using α-convert by auto
  then have ?thesis
    using Suc(7) by blast
} moreover {
  assume *: length ?D > 1
  then have D-is: D = set (Global-mltl-decomp ?D a (b - a) L)
    using Suc(6) unfolding Global-mltl-ext LP-mltl-aux.simps
    using α-convert by auto
  obtain X1 where ψ1-is: ψ1 = Ands-mltl-ext X1
    and X1-fact: ∀ i < length X1. ∃ y ∈ set (LP-mltl-aux α k).
      X1 ! i = Global-mltl-ext (a + i) (a + i) [1] y
    and length-X1: length X1 = Suc (b - a)
    using in-Global-mltl-decomp-exact-forward[OF *]
    using Suc(7) D-is by blast
  obtain X2 where ψ2-is: ψ2 = Ands-mltl-ext X2
    and X2-fact: ∀ i < length X2. ∃ y ∈ set (LP-mltl-aux α k).
      X2 ! i = Global-mltl-ext (a + i) (a + i) [1] y
    and length-X2: length X2 = Suc (b - a)
    using in-Global-mltl-decomp-exact-forward[OF *]
    using Suc(7) D-is by blast
  have X1-neq-X2: X1 ≠ X2
    using Suc(7) ψ1-is ψ2-is by blast
  then have ∃ i < b - a + 1. X1 ! i ≠ X2 ! i
    using length-X1 length-X2
    by (metis add.commute nth-equalityI plus-1-eq-Suc)
  then obtain i where i-bound: i < b - a + 1
    and X1i-neq-X2i: X1 ! i ≠ X2 ! i by blast
  obtain y1 where X1i-is: X1 ! i = Global-mltl-ext (a + i) (a + i) [1] y1
    and y1-in: y1 ∈ set ?D
    using X1-fact i-bound length-X1 by auto
  obtain y2 where X2i-is: X2 ! i = Global-mltl-ext (a + i) (a + i) [1] y2
    and y2-in: y2 ∈ set ?D
    using X2-fact i-bound length-X2 by auto
  have y1-neq-y2: y1 ≠ y2
    using X1i-is X2i-is X1i-neq-X2i by simp
  have semantics-mltl-ext π (X1 ! i)
    using Ands-mltl-semantics[of X1 π] Suc(8) unfolding ψ1-is
    by (metis Suc-eq-plus1 i-bound le-add2 length-X1 nth-mem)
  then have y1-semantics: semantics-mltl-ext (drop (a + i) π) y1
    unfolding X1i-is semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps
    using i-bound α-wpd a-leq-b
    by (metis Nat.add-diff-assoc Nat.le-diff-conv2 add-leD1 wpd-geq-one
      diff-add-inverse diff-add-inverse2 less-eq-iff-succ-less not-add-less1 order-refl)

```

```

have semantics-mltl-ext  $\pi$  (X2!i)
  using Ands-mltl-semanticsof X2  $\pi$  Suc(9) unfolding  $\psi 2$ -is
  by (metis Suc-eq-plus1 i-bound le-add2 length-X2 nth-mem)
then have y2-semantics: semantics-mltl-ext (drop (a+i)  $\pi$ ) y2
unfolding X2i-is semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps
  using i-bound  $\alpha$ -wpd a-leq-b
  by (metis Nat.add-diff-assoc Nat.le-diff-conv2 add-leD1 wpd-geq-one
diff-add-inverse diff-add-inverse2 less-eq-iff-succ-less not-add-less1 order-refl)

  have wpd: wpd-mltl (to-mltl  $\alpha$ )  $\leq$  length (drop (a+i)  $\pi$ )
    using  $\alpha$ -wpd i-bound a-leq-b by auto
  have ?thesis
    using Suc(1)[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition-allones wpd, of set ?D
y1 y2]
    using y1-in y2-in y1-semantics y2-semantics y1-neq-y2 by simp
  }
ultimately show ?thesis by linarith
next
case (Until-mltl-ext  $\alpha$  a b L  $\beta$ )
have a-leq-b:  $a \leq b$  and
   $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ ) and
   $\beta$ -welldef: intervals-welldef (to-mltl  $\beta$ )
using Suc(2) unfolding intervals-welldef.simps Until-mltl-ext to-mltl.simps
by simp-all
have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
  using Suc(3) unfolding Until-mltl-ext
by (metis convert-nnf-ext.simps(8) convert-nnf-ext-convert-nnf-ext mtl-ext.inject(7))

have  $\alpha$ -convert: convert-nnf-ext  $\alpha = \alpha$ 
  using  $\alpha$ -nnf convert-nnf-ext-convert-nnf-ext by metis
have  $\beta$ -nnf:  $\exists \varphi$ -init.  $\beta = \text{convert-nnf-ext } \varphi$ -init
  using Suc(3) unfolding Until-mltl-ext
by (metis convert-nnf-ext.simps(8) convert-nnf-ext-convert-nnf-ext mtl-ext.inject(7))

have  $\beta$ -convert: convert-nnf-ext  $\beta = \beta$ 
  using  $\beta$ -nnf convert-nnf-ext-convert-nnf-ext by metis
have  $\alpha$ -composition-allones: is-composition-MLTL-allones  $\alpha$  and
   $\beta$ -composition-allones: is-composition-MLTL-allones  $\beta$  and
  L-composition-allones: is-composition-allones (b-a+1) L
  using Until-mltl-ext Suc.premis(3) by simp-all
have  $\alpha$ -composition: is-composition-MLTL  $\alpha$ 
  using Until-mltl-ext Suc.premis(3) allones-implies-is-composition-MLTL by
auto
have  $\beta$ -composition: is-composition-MLTL  $\beta$ 
  using Until-mltl-ext Suc.premis(3) allones-implies-is-composition-MLTL
is-composition-MLTL.simps(5)
by force
have L-composition: is-composition (b-a+1) L
  using L-composition-allones allones-implies-is-composition by auto

```

```

have  $\alpha$ -wpd:  $b + \text{wpd-mltl } (\text{to-mltl } \alpha) - 1 \leq \text{length } \pi$  and
   $\beta$ -wpd:  $b + \text{wpd-mltl } (\text{to-mltl } \beta) \leq \text{length } \pi$ 
  using Suc(5) unfolding Until-mltl-ext to-mltl.simps wpd-mltl.simps
  by auto
let ?D = LP-mltl-aux  $\beta$   $k$ 
let ?s = interval-times  $a$   $L$ 
have length-L:  $1 \leq \text{length } L$ 
  using composition-length-lb[OF L-composition] a-leq-b by linarith
have length-L-allones:  $\text{length } L = b - a + 1$ 
  using L-composition-allones
  by (simp add: length-is-composition-allones)
have sfirst: ?s!0 =  $a$ 
  using interval-times-first by simp
have slast: ?s!( $\text{length } L$ ) =  $b + 1$ 
  using interval-times-last[OF a-leq-b L-composition]
  by blast
have length-s:  $\text{length } ?s = \text{length } L + 1$ 
  using interval-times-length by simp
have s1: ?s!1 =  $a + 1$ 
  using interval-times-allones
  by (metis L-composition-allones a-leq-b length-L length-s less-eq-iff-succ-less)
let ?front = set (Until-mltl-list  $\alpha$  ?D (?s!0) (?s!1 - 1) [?s!1 - ?s!0])
let ?back = set (concat (map ( $\lambda i$ . And-mltl-list
  [Global-mltl-ext
    (?s!0) (?s!i - 1) [?s!i - ?s!0] (And-mltl-ext  $\alpha$  (Not $c$ 
 $\beta$ )))
    (Until-mltl-list  $\alpha$  ?D (?s!i) (?s!(i + 1) - 1)
    [?s!(i + 1) - ?s!i])) [1..<length L]))
have split:  $D = ?\text{front} \cup ?\text{back}$ 
  using Suc(6) unfolding Until-mltl-ext LP-mltl-aux.simps
  using  $\alpha$ -convert  $\beta$ -convert list-concat-set-union
  by metis
{
  assume *:  $\psi 1 \in ?\text{front}$ 
  then obtain  $x1$  where  $\psi 1$ -is:  $\psi 1 = \text{Until-mltl-ext } \alpha$   $a$   $a$  [1]  $x1$ 
    and  $x1$ -in:  $x1 \in \text{set } ?D$ 
    unfolding sfirst s1 by auto
  have  $x1$ -semantics: semantics-mltl ( $\text{drop } a$   $\pi$ ) (to-mltl  $x1$ )
    using Suc(8) unfolding  $\psi 1$ -is semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps
    by auto
  have wpd-drop $a$ : wpd-mltl (to-mltl  $\beta$ )  $\leq \text{length}$  ( $\text{drop } a$   $\pi$ )
    using  $\beta$ -wpd a-leq-b by simp
  then have  $\beta$ -semantics: semantics-mltl-ext ( $\text{drop } a$   $\pi$ )  $\beta$ 
    unfolding semantics-mltl-ext-def
  using LP-mltl-aux-language-union-converse[OF  $\beta$ -welldef  $\beta$ -nnf  $\beta$ -composition,
of drop a  $\pi$  ?D  $k$ ]
  using  $x1$ -semantics  $x1$ -in unfolding semantics-mltl-ext-def by blast
{

```

```

assume **:  $\psi 2 \in ?front$ 
then obtain  $x 2$  where  $\psi 2\text{-is}$ :  $\psi 2 = \text{Until-mltl-ext } \alpha \ a \ a \ [1] \ x 2$ 
and  $x 2\text{-in}$ :  $x 2 \in \text{set } ?D$ 
unfolding  $sfirst \ s 1$  by auto
have  $x 2\text{-semantics}$ : semantics-mltl (drop a  $\pi$ ) (to-mltl  $x 2$ )
using Suc(9) unfolding  $\psi 2\text{-is}$  semantics-mltl-ext-def to-mltl.simps
semantics-mltl.simps
by auto
have  $x 1\text{-neq-}x 2$ :  $x 1 \neq x 2$ 
using Suc(7)  $\psi 1\text{-is}$   $\psi 2\text{-is}$  by simp
have  $?thesis$ 
using Suc(1)[OF  $\beta\text{-welldef } \beta\text{-nnf } \beta\text{-composition-allones}$ , of drop a  $\pi$  set
 $?D \ x 1 \ x 2$ ]
using  $x 1\text{-semantics}$   $x 1\text{-in}$   $x 2\text{-semantics}$   $x 2\text{-in}$   $x 1\text{-neq-}x 2$ 
using semantics-mltl-ext-def wpd-dropa by blast
} moreover {
assume **:  $\psi 2 \in ?back$ 
then obtain  $i \ y 2$  where
 $\psi 2\text{-is}$ :  $\psi 2 = \text{And-mltl-ext } (\text{Global-mltl-ext } (?s!0) (?s!i-1) [?s!i - ?s!0])$ 
(And-mltl-ext  $\alpha$  (Notc  $\beta$ ))
 $(\text{Until-mltl-ext } \alpha \ (?s!i) \ (?s!(i+1)-1) \ [(?s!(i+1)) - (?s!i)] \ y 2)$ 
and  $i\text{-bound}$ :  $1 \leq i \wedge i < \text{length } L$ 
and  $y 2\text{-in}$ :  $y 2 \in \text{set } ?D$ 
by auto
have  $p$ :  $\neg \text{semantics-mltl-ext } (\text{drop a } \pi) \ \beta$ 
using Suc(9) unfolding  $\psi 2\text{-is}$  semantics-mltl-ext-def to-mltl.simps
semantics-mltl.simps
using  $i\text{-bound}$   $\text{length-}L\text{-allones}$ 
by (metis wpd-dropa wpd-geq-one drop-all eq-imp-le le-neq-implies-less
 $\text{length-}0\text{-conv less-nat-zero-code not-one-le-zero sfirst}$ )
have  $?thesis$  using  $\beta\text{-semantics}$   $p$ 
by metis
}
ultimately have  $?thesis$  using Suc(7) split by blast
} moreover {
assume *:  $\psi 1 \in ?back$ 
then obtain  $i 1 \ y 1$  where
 $\psi 1\text{-is}$ :  $\psi 1 = \text{And-mltl-ext } (\text{Global-mltl-ext } (?s!0) (?s!i 1-1) [?s!i 1 - ?s!0])$ 
(And-mltl-ext  $\alpha$  (Notc  $\beta$ ))
 $(\text{Until-mltl-ext } \alpha \ (?s!i 1) \ (?s!(i 1+1)-1) \ [(?s!(i 1+1)) - (?s!i 1)] \ y 1)$ 
and  $i 1\text{-bound}$ :  $1 \leq i 1 \wedge i 1 < \text{length } L$ 
and  $y 1\text{-in}$ :  $y 1 \in \text{set } ?D$ 
by auto
have  $s i 1$ :  $?s!i 1 = a + i 1$ 
using  $\text{interval-times-allones}$ 
using  $L\text{-composition-allones}$   $a\text{-leq-}b$   $i 1\text{-bound}$   $\text{length-}s$  by auto
have  $s i 1'$ :  $?s!(i 1+1) = a + i 1 + 1$ 
using  $\text{interval-times-allones}$ 
using  $L\text{-composition-allones}$   $a\text{-leq-}b$   $i 1\text{-bound}$   $\text{length-}s$  by auto

```



```

have  $\psi 1$ -is:  $\psi 1 = \text{And-mltl-ext } (\text{Global-mltl-ext } a (a+i1-1) [i1]) (\text{And-mltl-ext } \alpha (\text{Not}_c \beta))$ 
  ( $\text{Until-mltl-ext } \alpha (a+i1) (a+i1) [1] y1$ )
  using  $si1$   $si1'$   $sfirst$   $\psi 1$ -is by auto
have  $y1$ -semantics:  $\text{semantics-mltl-ext } (\text{drop } (a+i1) \pi) y1$ 
  using  $\text{Suc}(8)$  unfolding  $\psi 1$ -is  $\text{semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps}$ 
  by auto
have  $\text{wpd-mltl } (\text{to-mltl } \beta) \leq \text{length } (\text{drop } (a + i1) \pi)$ 
  using  $\beta$ -wpd  $i1$ -bound  $\text{length-L-allones}$  by auto
then have  $\beta$ -semantics1:  $\text{semantics-mltl-ext } (\text{drop } (a+i1) \pi) \beta$ 
using  $\text{LP-mltl-aux-language-union-converse}[OF \beta\text{-welldef } \beta\text{-nnf } \beta\text{-composition, of drop } (a+i1) \pi ?D k]$ 
  using  $y1$ -semantics  $y1$ -in by blast
{
  assume **:  $\psi 2 \in ?front$ 
  then obtain  $x2$  where  $\psi 2$ -is:  $\psi 2 = \text{Until-mltl-ext } \alpha a a [1] x2$ 
    and  $x2$ -in:  $x2 \in \text{set } ?D$ 
    unfolding  $sfirst$   $s1$  by auto
    have  $x2$ -semantics:  $\text{semantics-mltl } (\text{drop } a \pi) (\text{to-mltl } x2)$ 
      using  $\text{Suc}(9)$  unfolding  $\psi 2$ -is  $\text{semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps}$ 
      by auto
    have  $\text{wpd-mltl } (\text{to-mltl } \beta) \leq \text{length } (\text{drop } a \pi)$ 
      using  $\beta$ -wpd  $a$ -leq- $b$  by auto
    then have  $\beta$ -semantics2:  $\text{semantics-mltl } (\text{drop } a \pi) (\text{to-mltl } \beta)$ 
using  $\text{LP-mltl-aux-language-union-converse}[OF \beta\text{-welldef } \beta\text{-nnf } \beta\text{-composition, of drop } a \pi ?D k]$ 
      using  $x2$ -semantics  $x2$ -in unfolding  $\text{semantics-mltl-ext-def}$ 
      by blast
    then have  $?thesis$ 
      using  $\text{Suc}(8)$  unfolding  $\psi 1$ -is  $\text{semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps}$ 
      by auto
  } moreover {
    assume **:  $\psi 2 \in ?back$ 
    then obtain  $i2$   $y2$  where
       $\psi 2$ -is:  $\psi 2 = \text{And-mltl-ext } (\text{Global-mltl-ext } (?s!0) (?s!i2-1) [?s!i2 - ?s!0]) (\text{And-mltl-ext } \alpha (\text{Not}_c \beta))$ 
      ( $\text{Until-mltl-ext } \alpha (?s!i2) (?s!(i2+1)-1) [(?s!(i2+1)) - (?s!i2)] y2$ )
      and  $i2$ -bound:  $1 \leq i2 \wedge i2 < \text{length } L$ 
      and  $y2$ -in:  $y2 \in \text{set } ?D$ 
      by auto
    have  $si2$ :  $?s!i2 = a + i2$ 
      using  $\text{interval-times-allones}$ 
      using  $L$ -composition-allones  $a$ -leq- $b$   $i2$ -bound  $\text{length-s}$  by auto
    have  $si2'$ :  $?s!(i2+1) = a + i2 + 1$ 
      using  $\text{interval-times-allones}$ 
      using  $L$ -composition-allones  $a$ -leq- $b$   $i2$ -bound  $\text{length-s}$  by auto
  }

```

```

have  $\psi 2$ -is:  $\psi 2 = \text{And-mltl-ext } (\text{Global-mltl-ext } a \ (a+i2-1) \ [i2]) \ (\text{And-mltl-ext } \alpha \ (\text{Not}_c \ \beta))$ 
  (Until-mltl-ext  $\alpha \ (a+i2) \ (a+i2) \ [1] \ y2$ )
  using si2 si2' sfirst  $\psi 2$ -is by auto
  have y2-semantics: semantics-mltl-ext (drop  $(a+i2) \ \pi$ ) y2
    using Suc(9) unfolding  $\psi 2$ -is semantics-mltl-ext-def to-mltl.simps
semantics-mltl.simps
  by auto
  have wpd-dropi2: wpd-mltl (to-mltl  $\beta$ )  $\leq \text{length } (\text{drop } (a + i2) \ \pi)$ 
    using  $\beta$ -wpd i2-bound length-L-allones by auto
  then have  $\beta$ -semantics2: semantics-mltl-ext (drop  $(a+i2) \ \pi$ )  $\beta$ 
  using LP-mltl-aux-language-union-converse[OF  $\beta$ -welldef  $\beta$ -nnf  $\beta$ -composition,
of drop  $(a+i2) \ \pi$  ?D k]
    using y2-semantics y2-in by blast
  {
    assume i1-eq-i2:  $i1 = i2$ 
    then have y1-neq-y2:  $y1 \neq y2$ 
      using  $\psi 1$ -is  $\psi 2$ -is Suc(7) by blast
    then have ?thesis
    using Suc(1)[OF  $\beta$ -welldef  $\beta$ -nnf  $\beta$ -composition-allones, of drop  $(a+i1)$ 
 $\pi$  set ?D y1 y2]
      using wpd-dropi2 i1-eq-i2 y1-semantics y1-in y2-semantics y2-in
      by blast
  } moreover {
    assume i1-le-i2:  $i1 < i2$ 
    then have  $\neg$ semantics-mltl-ext (drop  $(a + i1) \ \pi$ )  $\beta$ 
      using Suc(9) unfolding  $\psi 2$ -is semantics-mltl-ext-def to-mltl.simps
semantics-mltl.simps
      using add.assoc add-le-imp-le-diff by force
    then have ?thesis
      using  $\beta$ -semantics1 by blast
  } moreover {
    assume i1-ge-i2:  $i1 > i2$ 
    then have  $\neg$ semantics-mltl-ext (drop  $(a + i2) \ \pi$ )  $\beta$ 
      using Suc(8) unfolding  $\psi 1$ -is semantics-mltl-ext-def to-mltl.simps
semantics-mltl.simps
      using add.assoc add-le-imp-le-diff by force
    then have ?thesis
      using  $\beta$ -semantics2 by blast
  }
  ultimately have ?thesis by linarith
}
ultimately have ?thesis
  using split Suc(7) by blast
}
ultimately show ?thesis
  using split Suc(7) by blast
next
case (Release-mltl-ext  $\alpha \ a \ b \ L \ \beta$ )

```

have $a\text{-leq-}b$: $a \leq b$ **and**
 $\alpha\text{-welldef}$: *intervals-welldef* (*to-mltl* α) **and**
 $\beta\text{-welldef}$: *intervals-welldef* (*to-mltl* β)
using *Suc(2)* **unfolding** *intervals-welldef.simps* *Release-mltl-ext to-mltl.simps*
by *simp-all*
have $\alpha\text{-nnf}$: $\exists \varphi\text{-init. } \alpha = \text{convert-nnf-ext } \varphi\text{-init}$
using *Suc(3)* **unfolding** *Release-mltl-ext*
by (*metis convert-nnf-ext.simps(9)* *convert-nnf-ext-convert-nnf-ext mltl-ext.inject(8)*)

have $\alpha\text{-convert}$: *convert-nnf-ext* $\alpha = \alpha$
using $\alpha\text{-nnf}$ *convert-nnf-ext-convert-nnf-ext* **by** *metis*
have $\beta\text{-nnf}$: $\exists \varphi\text{-init. } \beta = \text{convert-nnf-ext } \varphi\text{-init}$
using *Suc(3)* **unfolding** *Release-mltl-ext*
by (*metis convert-nnf-ext.simps(9)* *convert-nnf-ext-convert-nnf-ext mltl-ext.inject(8)*)

have $\beta\text{-convert}$: *convert-nnf-ext* $\beta = \beta$
using $\beta\text{-nnf}$ *convert-nnf-ext-convert-nnf-ext* **by** *metis*
have $\alpha\text{-composition-allones}$: *is-composition-MLTL-allones* α **and**
 $\beta\text{-composition-allones}$: *is-composition-MLTL-allones* β **and**
 $L\text{-composition-allones}$: *is-composition-allones* ($b-a+1$) L
using *Release-mltl-ext Suc.prem(3)* **by** *simp-all*
have $\alpha\text{-composition}$: *is-composition-MLTL* α
using *Release-mltl-ext Suc.prem(3)* *allones-implies-is-composition-MLTL*
by *auto*
have $\beta\text{-composition}$: *is-composition-MLTL* β
using *Release-mltl-ext Suc.prem(3)* *allones-implies-is-composition-MLTL*
is-composition-MLTL.simps(5)
by *force*
have $L\text{-composition}$: *is-composition* ($b-a+1$) L
using $L\text{-composition-allones}$ *allones-implies-is-composition* **by** *auto*
have $\alpha\text{-wpd}$: $b + \text{wpd-mltl} (\text{to-mltl } \alpha) \leq \text{length } \pi$ **and**
 $\beta\text{-wpd}$: $b + \text{wpd-mltl} (\text{to-mltl } \beta) \leq \text{length } \pi$
using *Suc(5)* **unfolding** *Release-mltl-ext to-mltl.simps wpd-mltl.simps*
by *auto*
let $?D = LP\text{-mltl-aux } \alpha k$
let $?s = \text{interval-times } a L$
have $\text{length-}L$: $1 \leq \text{length } L$
using *composition-length-lb[OF L-composition]* $a\text{-leq-}b$ **by** *linarith*
have $\text{length-}L\text{-allones}$: $\text{length } L = b-a+1$
using $L\text{-composition-allones}$
by (*simp add: length-is-composition-allones*)
have $s\text{first}$: $?s!0 = a$
using *interval-times-first* **by** *simp*
have $s\text{last}$: $?s!(\text{length } L) = b+1$
using *interval-times-last[OF a-leq-b L-composition]*
by *blast*
have $\text{length-}s$: $\text{length } ?s = \text{length } L + 1$
using *interval-times-length* **by** *simp*
have $\text{length-}L$: $\text{length } L = b-a+1$

```

using length-is-composition-allones[OF L-composition-allones]
by blast
have s1: ?s ! 1 = a+1
using interval-times-allones
using L-composition L-composition-allones a-leq-b add-gr-0 composition-length-lb
length-s by auto
have length- $\pi$ -ge-b: length  $\pi > b$ 
using  $\alpha$ -wpd wpd-geq-one
by (metis One-nat-def Suc-n-not-le-n add-diff-cancel-left' add-leD1 diff-is-0-eq'
le-neq-implies-less)
let ?front = set [Global-mltl-ext a b L (And-mltl-ext (Notc  $\alpha$ )  $\beta$ )]
let ?middle = set (Mighty-Release-mltl-list ?D  $\beta$  (?s ! 0) (?s ! 1 - 1)
[?s ! 1 - ?s ! 0])
let ?back = set (concat (map ( $\lambda$ i. And-mltl-list
[Global-mltl-ext
(?s ! 0) (?s ! i - 1) [?s!i - ?s!0] (And-mltl-ext (Notc
 $\alpha$ )  $\beta$ ])
(Mighty-Release-mltl-list ?D  $\beta$  (?s ! i)
(?s ! (i + 1) - 1) [?s ! (i + 1) - ?s ! i]))
[1.. $\text{length } L$ ]))
have D-is: D = ?front  $\cup$  ?middle  $\cup$  ?back
using Suc(6) unfolding Release-mltl-ext LP-mltl-aux.simps
using  $\alpha$ -convert list-concat-set-union
by (metis append-assoc)
{
assume *:  $\psi 1 \in ?front$ 
then have  $\psi 1$ :  $\psi 1 = \text{Global-mltl-ext } a \ b \ L \ (\text{And-mltl-ext } (\text{Not}_c \ \alpha) \ \beta)$ 
by auto
{
assume **:  $\psi 2 \in ?front$ 
have ?thesis using * ** Suc(7) by auto
} moreover {
assume **:  $\psi 2 \in ?middle$ 
then obtain x where  $\psi 2$ :  $\psi 2 = \text{Mighty-Release-mltl-ext } x \ \beta$ 
a (?s ! 1 - 1) [?s ! 1 - a]
and x-in: x  $\in$  set ?D
using sfirst by auto
have  $\psi 2$ :  $\psi 2 = \text{Mighty-Release-mltl-ext } x \ \beta \ a \ a \ [1]$ 
using s1  $\psi 2$  by simp
have x-semantics: semantics-mltl (drop a  $\pi$ ) (to-mltl x)
using Suc(9) unfolding  $\psi 1 \ \psi 2$  semantics-mltl-ext-def to-mltl.simps
Mighty-Release-mltl-ext.simps semantics-mltl.simps
by force
have wpd-mltl (to-mltl  $\alpha$ )  $\leq$  length (drop a  $\pi$ )
using  $\alpha$ -wpd a-leq-b by auto
then have semantics-mltl (drop a  $\pi$ ) (to-mltl  $\alpha$ )
using LP-mltl-aux-language-union-converse[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition,
of drop a  $\pi$  ?D k]
using x-semantics x-in unfolding semantics-mltl-ext-def by blast

```

```

then have ?thesis
  using Suc(8) unfolding  $\psi 1$  semantics-mltl-ext-def to-mltl.simps
  Mighty-Release-mltl-ext.simps semantics-mltl.simps
  using length- $\pi$ -ge-b by auto
} moreover {
  assume **:  $\psi 2 \in ?back$ 
  then obtain  $i 2$  where  $\psi 2$ -in:  $\psi 2 \in set (And\text{-mltl}\text{-list}$ 
    [Global-mltl-ext
      (interval-times  $a L ! 0$ )
      (interval-times  $a L ! i 2 - 1$ ) [ $?s!i 2 - ?s!0$ ] (And-mltl-ext
        ( $Not_c \alpha$ )  $\beta$ )]
      (Mighty-Release-mltl-list (LP-mltl-aux  $\alpha k$ )  $\beta$ )
      (interval-times  $a L ! i 2$ )
      (interval-times  $a L ! (i 2 + 1) - 1$ )
      [interval-times  $a L ! (i 2 + 1) -$ 
        interval-times  $a L ! i 2$ ])
      and  $i 2$ -bound:  $1 \leq i 2 \wedge i 2 < length L$ 
    by force
  have  $si 2$ :  $?s!i 2 = a + i 2$ 
    using interval-times-allones[OF  $a$ -leq-b  $L$ -composition-allones, of  $i 2$ ]
    using  $i 2$ -bound length- $L$  length- $s$  by auto
  have  $si 2'$ :  $?s!(i 2 + 1) = a + i 2 + 1$ 
    using interval-times-allones[OF  $a$ -leq-b  $L$ -composition-allones, of  $i 2 + 1$ ]
    using  $i 2$ -bound length- $L$  length- $s$  by auto
  obtain  $x 2$  where  $\psi 2$ :  $\psi 2 = And\text{-mltl}\text{-ext}$ 
    (Global-mltl-ext  $a (a + i 2 - 1) [i 2]$  (And-mltl-ext ( $Not_c \alpha$ )  $\beta$ ))
    (Mighty-Release-mltl-ext  $x 2 \beta (a + i 2) (a + i 2) [1]$ )
    and  $x 2$ -in:  $x 2 \in set ?D$ 
    using  $\psi 2$ -in sfirst  $si 2 si 2'$  by auto
  have  $x 2$ -semantics: semantics-mltl (drop ( $a + i 2$ )  $\pi$ ) (to-mltl  $x 2$ )
    using Suc(9) unfolding  $\psi 2$  semantics-mltl-ext-def to-mltl.simps
  Mighty-Release-mltl-ext.simps semantics-mltl.simps
  by force
  have wpd-mltl (to-mltl  $\alpha$ )  $\leq length$  (drop ( $a + i 2$ )  $\pi$ )
    using  $\alpha$ -wpd  $a$ -leq-b  $i 2$ -bound length- $L$  by auto
  then have semantics-mltl (drop ( $a + i 2$ )  $\pi$ ) (to-mltl  $\alpha$ )
    using LP-mltl-aux-language-union-converse[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition,
  of drop ( $a + i 2$ )  $\pi$  ?D  $k$ ]
    using  $x 2$ -semantics  $x 2$ -in unfolding semantics-mltl-ext-def by blast
  then have ?thesis
    using Suc(8) unfolding  $\psi 1$  semantics-mltl-ext-def to-mltl.simps
  Mighty-Release-mltl-ext.simps semantics-mltl.simps
    using length- $\pi$ -ge-b  $i 2$ -bound length- $L$  by auto
  }
ultimately have ?thesis using Suc(7)  $D$ -is by blast
} moreover {
  assume *:  $\psi 1 \in ?middle$ 
  then obtain  $x 1$  where  $\psi 1$ :  $\psi 1 = Mighty\text{-Release}\text{-mltl}\text{-ext}$   $x 1 \beta$ 
     $a (?s ! 1 - 1) [?s ! 1 - a]$ 

```

```

      and x1-in: x1 ∈ set ?D
    using sfirst by auto
  have ψ1: ψ1 = Mighty-Release-mltl-ext x1 β a a [1]
    using s1 ψ1 by simp
  have x1-antics: semantics-mltl (drop a π) (to-mltl x1)
  using Suc(8) unfolding ψ1 semantics-mltl-ext-def to-mltl.simps Mighty-Release-mltl-ext.simps
  semantics-mltl.simps
    by force
  have wpd-mltl (to-mltl α) ≤ length (drop a π)
    using α-wpd a-leq-b by auto
  then have α-antics: semantics-mltl (drop a π) (to-mltl α)
  using LP-mltl-aux-language-union-converse[OF α-welldef α-nnf α-composition,
  of drop a π ?D k]
    using x1-antics x1-in unfolding semantics-mltl-ext-def by blast
  {
    assume **: ψ2 ∈ ?front
    then have ψ2: ψ2 = Global-mltl-ext a b L (And-mltl-ext (Notc α) β)
      by auto
    have ?thesis
      using α-antics using Suc(9) unfolding ψ2 semantics-mltl-ext-def
  to-mltl.simps semantics-mltl.simps
        using a-leq-b length-π-ge-b by simp
  } moreover {
    assume **: ψ2 ∈ ?middle
    then obtain x2 where ψ2: ψ2 = Mighty-Release-mltl-ext x2 β
      a (?s ! 1 - 1) [?s ! 1 - a]
      and x2-in: x2 ∈ set ?D
      using sfirst by auto
    have ψ2: ψ2 = Mighty-Release-mltl-ext x2 β a a [1]
      using s1 ψ2 by simp
    have x2-antics: semantics-mltl (drop a π) (to-mltl x2)
      using Suc(9) unfolding ψ2 semantics-mltl-ext-def to-mltl.simps
  Mighty-Release-mltl-ext.simps semantics-mltl.simps
        by force
    have x1-neq-x2: x1 ≠ x2
      using Suc(7) ψ1 ψ2 by blast
    have wpd-mltl (to-mltl α) ≤ length (drop a π)
      using α-wpd a-leq-b by simp
    then have ?thesis
      using Suc(1)[OF α-welldef α-nnf α-composition-allones, of drop a π
  set ?D x1 x2]
        using x1-neq-x2 x1-antics x2-antics x1-in x2-in
        unfolding semantics-mltl-ext-def by blast
  } moreover {
    assume **: ψ2 ∈ ?back
    then obtain i2 where ψ2-in: ψ2 ∈ set (And-mltl-list
      [Global-mltl-ext
      (interval-times a L ! 0)
      (interval-times a L ! i2 - 1) [?s!i2 - ?s!0] (And-mltl-ext

```

```

(Notc α) β]]
      (Mighty-Release-mltl-list (LP-mltl-aux α k) β
        (interval-times a L ! i2)
        (interval-times a L ! (i2 + 1) - 1)
        [interval-times a L ! (i2 + 1) -
          interval-times a L ! i2]))
      and i2-bound: 1 ≤ i2 ∧ i2 < length L
    by force
  have si2: ?s!i2 = a+i2
    using interval-times-allones[OF a-leq-b L-composition-allones, of i2]
    using i2-bound length-L length-s by auto
  have si2': ?s!(i2+1) = a+i2+1
    using interval-times-allones[OF a-leq-b L-composition-allones, of i2+1]
    using i2-bound length-L length-s by auto
  obtain x2 where ψ2: ψ2 = And-mltl-ext
    (Global-mltl-ext a (a + i2 - 1) [i2] (And-mltl-ext (Notc α) β))
    (Mighty-Release-mltl-ext x2 β (a + i2) (a + i2) [1])
    and x2-in: x2 ∈ set ?D
    using ψ2-in sfirst si2 si2' by auto
  have x2-semantics: semantics-mltl (drop (a + i2) π) (to-mltl x2)
    using Suc(9) unfolding ψ2 semantics-mltl-ext-def to-mltl.simps
Mighty-Release-mltl-ext.simps semantics-mltl.simps
    by force
  have wpd-mltl (to-mltl α) ≤ length (drop (a + i2) π)
    using α-wpd a-leq-b i2-bound length-L by auto
  then have semantics-mltl (drop (a + i2) π) (to-mltl α)
  using LP-mltl-aux-language-union-converse[OF α-welldef α-nnf α-composition,
of drop (a + i2) π ?D k]
    using x2-semantics x2-in unfolding semantics-mltl-ext-def by blast
  have ?thesis using α-semantics
    using Suc(9) unfolding ψ2 Mighty-Release-mltl-ext.simps seman-
tics-mltl-ext-def to-mltl.simps semantics-mltl.simps
    by auto
}
ultimately have ?thesis using Suc(7) D-is by blast
} moreover {
  assume *: ψ1 ∈ ?back
  then obtain i1 where ψ1-in: ψ1 ∈ set (And-mltl-list
    [Global-mltl-ext
      (interval-times a L ! 0)
      (interval-times a L ! i1 - 1) [?s!i1 - ?s!0] (And-mltl-ext
(Notc α) β)]
    (Mighty-Release-mltl-list (LP-mltl-aux α k) β
      (interval-times a L ! i1)
      (interval-times a L ! (i1 + 1) - 1)
      [interval-times a L ! (i1 + 1) -
        interval-times a L ! i1]))
    and i1-bound: 1 ≤ i1 ∧ i1 < length L
  by force

```

```

have  $si1$ :  $?s!i1 = a+i1$ 
  using interval-times-allones[OF a-leq-b L-composition-allones, of i1]
  using i1-bound length-L length-s by auto
have  $si1'$ :  $?s!(i1+1) = a+i1+1$ 
  using interval-times-allones[OF a-leq-b L-composition-allones, of i1+1]
  using i1-bound length-L length-s by auto
obtain  $x1$  where  $\psi1$ :  $\psi1 = \text{And-mltl-ext}$ 
  (Global-mltl-ext  $a$  ( $a + i1 - 1$ ) [ $i1$ ] (And-mltl-ext (Notc  $\alpha$ )  $\beta$ ))
  (Mighty-Release-mltl-ext  $x1$   $\beta$  ( $a + i1$ ) ( $a + i1$ ) [ $1$ ])
  and  $x1\text{-in}$ :  $x1 \in \text{set } ?D$ 
  using  $\psi1\text{-in}$  sfirst  $si1$   $si1'$  by auto
have  $x1\text{-semantics}$ : semantics-mltl (drop ( $a + i1$ )  $\pi$ ) (to-mltl  $x1$ )
using Suc(8) unfolding  $\psi1$  semantics-mltl-ext-def to-mltl.simps Mighty-Release-mltl-ext.simps
semantics-mltl.simps
  by force
have  $complen1$ : wpd-mltl (to-mltl  $\alpha$ )  $\leq$  length (drop ( $a + i1$ )  $\pi$ )
  using  $\alpha\text{-wpd}$  a-leq-b i1-bound length-L by auto
then have  $\alpha\text{-semantics1}$ : semantics-mltl (drop ( $a + i1$ )  $\pi$ ) (to-mltl  $\alpha$ )
using LP-mltl-aux-language-union-converse[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition,
of drop ( $a + i1$ )  $\pi$   $?D$   $k$ ]
  using  $x1\text{-semantics}$   $x1\text{-in}$  unfolding semantics-mltl-ext-def by blast
  {
    assume  $*$ :  $\psi2 \in ?front$ 
    then have  $\psi2$ :  $\psi2 = \text{Global-mltl-ext}$   $a$   $b$   $L$  (And-mltl-ext (Notc  $\alpha$ )  $\beta$ )
    by auto
    have  $?thesis$ 
      using Suc(9) unfolding  $\psi2$  semantics-mltl-ext-def to-mltl.simps
Mighty-Release-mltl-ext.simps semantics-mltl.simps
      using length- $\pi$ -ge-b i1-bound length-L
      by (smt (verit, best)  $\langle$ semantics-mltl (drop ( $a + i1$ )  $\pi$ ) (to-mltl  $\alpha$ ) $\rangle$ 
diff-add-inverse diff-le-mono le-antisym le-trans less-eq-iff-succ-less less-irrefl-nat
less-or-eq-imp-le nat-le-iff-add nat-le-linear)
    } moreover {
      assume  $*$ :  $\psi2 \in ?middle$ 
      then obtain  $x2$  where  $\psi2$ :  $\psi2 = \text{Mighty-Release-mltl-ext}$   $x2$   $\beta$ 
         $a$  ( $?s!1 - 1$ ) [ $?s!1 - a$ ]
        and  $x2\text{-in}$ :  $x2 \in \text{set } ?D$ 
        using sfirst by auto
        have  $\psi2$ :  $\psi2 = \text{Mighty-Release-mltl-ext}$   $x2$   $\beta$   $a$   $a$  [ $1$ ]
        using  $s1$   $\psi2$  by simp
        have  $x2\text{-semantics}$ : semantics-mltl (drop  $a$   $\pi$ ) (to-mltl  $x2$ )
        using Suc(9) unfolding  $\psi2$  semantics-mltl-ext-def to-mltl.simps
Mighty-Release-mltl-ext.simps semantics-mltl.simps
        by force
        have wpd-mltl (to-mltl  $\alpha$ )  $\leq$  length (drop  $a$   $\pi$ )
        using  $\alpha\text{-wpd}$  a-leq-b by auto
        then have  $\alpha\text{-semantics}$ : semantics-mltl (drop  $a$   $\pi$ ) (to-mltl  $\alpha$ )
        using LP-mltl-aux-language-union-converse[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition,
of drop  $a$   $\pi$   $?D$   $k$ ]

```



```

    using x2-semantic x2-in unfolding semantic-mltl-ext-def by blast
    have ?thesis
      using Suc(8) unfolding  $\psi_1$  Mighty-Release-mltl-ext.simps semantic-mltl-ext-def to-mltl.simps semantic-mltl.simps
      using  $\alpha$ -semantic by auto
  } moreover {
    assume *:  $\psi_2 \in ?back$ 
    then obtain i2 where  $\psi_2$ -in:  $\psi_2 \in set (And-mltl-list$ 
      [Global-mltl-ext
        (interval-times a L ! 0)
        (interval-times a L ! i2 - 1) [?s!i2 - ?s!0] (And-mltl-ext
          (Notc  $\alpha$ )  $\beta$ )]
          (Mighty-Release-mltl-list (LP-mltl-aux  $\alpha$  k))  $\beta$ 
          (interval-times a L ! i2)
          (interval-times a L ! (i2 + 1) - 1)
          [interval-times a L ! (i2 + 1) -
            interval-times a L ! i2])
          and i2-bound: 1  $\leq$  i2  $\wedge$  i2 < length L
      by force
    have si2: ?s!i2 = a+i2
      using interval-times-allones[OF a-leq-b L-composition-allones, of i2]
      using i2-bound length-L length-s by auto
    have si2': ?s!(i2+1) = a+i2+1
      using interval-times-allones[OF a-leq-b L-composition-allones, of i2+1]
      using i2-bound length-L length-s by auto
    obtain x2 where  $\psi_2$ :  $\psi_2 = And-mltl-ext$ 
      (Global-mltl-ext a (a + i2 - 1) [i2]) (And-mltl-ext (Notc  $\alpha$ )  $\beta$ )
      (Mighty-Release-mltl-ext x2  $\beta$  (a + i2) (a + i2) [1])
      and x2-in: x2  $\in$  set ?D
      using  $\psi_2$ -in sfirst si2 si2' by auto
    have x2-semantic: semantic-mltl (drop (a + i2)  $\pi$ ) (to-mltl x2)
      using Suc(9) unfolding  $\psi_2$  semantic-mltl-ext-def to-mltl.simps
      Mighty-Release-mltl-ext.simps semantic-mltl.simps
      by force
    have complen2: wpd-mltl (to-mltl  $\alpha$ )  $\leq$  length (drop (a + i2)  $\pi$ )
      using  $\alpha$ -wpd a-leq-b i2-bound length-L by auto
    then have  $\alpha$ -semantic2: semantic-mltl (drop (a + i2)  $\pi$ ) (to-mltl  $\alpha$ )
      using LP-mltl-aux-language-union-converse[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition,
      of drop (a + i2)  $\pi$  ?D k]
      using x2-semantic x2-in unfolding semantic-mltl-ext-def by blast
    {
      assume eq: i1 = i2
      then have x1-neq-x2: x1  $\neq$  x2
        using Suc(7)  $\psi_1$   $\psi_2$  by blast
      have ?thesis using eq
        using Suc(1)[OF  $\alpha$ -welldef  $\alpha$ -nnf  $\alpha$ -composition-allones complen1,
        of set ?D x1 x2]
        using x1-in x2-in x1-semantic x2-semantic x1-neq-x2 unfolding
        semantic-mltl-ext-def

```

```

      by blast
    } moreover {
      assume le: i1 < i2
      then have ¬semantics-mltl (drop (a + i1) π) (to-mltl α)
      using Suc(9) unfolding ψ2 semantics-mltl-ext-def semantics-mltl.simps
to-mltl.simps
      using length-π-ge-b a-leq-b by simp
      then have ?thesis
      using α-semantics1 by blast
    } moreover {
      assume ge: i1 > i2
      then have ¬semantics-mltl (drop (a + i2) π) (to-mltl α)
      using Suc(8) unfolding ψ1 semantics-mltl-ext-def semantics-mltl.simps
to-mltl.simps
      using length-π-ge-b a-leq-b by simp
      then have ?thesis
      using α-semantics2 by blast
    }
  ultimately have ?thesis by linarith
}
ultimately have ?thesis using Suc(7) D-is by blast
}
ultimately show ?thesis using Suc(7) D-is by blast
qed
qed

```

lemma *LP-mltl-language-disjoint-aux*:

```

fixes φ::'a mltl-ext and ψ1 ψ2::'a mltl-ext and k::nat
assumes intervals-welldef: intervals-welldef (to-mltl φ)
assumes is-nnf: ∃φ-init. φ = convert-nnf-ext φ-init
assumes composition: is-composition-MLTL-allones φ
assumes D-decomp: D = set (LP-mltl-aux φ k)
assumes diff-formulas: (ψ1 ∈ D) ∧ (ψ2 ∈ D) ∧ ψ1 ≠ ψ2
assumes r-wpd: r ≥ wpd-mltl (to-mltl φ)
shows (language-mltl-r (to-mltl ψ1) r)
  ∩ (language-mltl-r (to-mltl ψ2) r) = {}

```

proof–

```

{
  assume contra: (language-mltl-r (to-mltl ψ1) r)
    ∩ (language-mltl-r (to-mltl ψ2) r) ≠ {}
  then have ∃π. π ∈ (language-mltl-r (to-mltl ψ1) r) ∧
    π ∈ (language-mltl-r (to-mltl ψ2) r)
    by auto
  then obtain π where in1: π ∈ (language-mltl-r (to-mltl ψ1) r)
    and in2: π ∈ (language-mltl-r (to-mltl ψ2) r)
    by blast
  have sem1: semantics-mltl-ext π ψ1 and
    sem2: semantics-mltl-ext π ψ2 and
    len: length π ≥ wpd-mltl (to-mltl φ)

```

```

    using in1 in2 assms(6)
    unfolding language-mltl-r-def semantics-mltl-ext-def
    by simp-all
  have False
    using LP-mltl-language-disjoint-aux-helper[OF assms(1-3) len assms(4, 5)
sem1 sem2]
    by simp
}
then show ?thesis by blast
qed

```

theorem *LP-mltl-language-disjoint*:

```

  fixes  $\varphi::'a$  mltl-ext and  $\psi1 \ \psi2::'a$  mltl and  $k::nat$ 
  assumes intervals-welldef: intervals-welldef (to-mltl  $\varphi$ )
  assumes composition: is-composition-MLTL-allones  $\varphi$ 
  assumes D-decomp:  $D = set (LP-mltl \ \varphi \ k)$ 
  assumes diff-formulas:  $(\psi1 \in D) \wedge (\psi2 \in D) \wedge \psi1 \neq \psi2$ 
  assumes r-wpd:  $r \geq wpd\text{-mltl} (to\text{-mltl} \ \varphi)$ 
  shows  $(language\text{-mltl}\text{-}r \ \psi1 \ r) \cap (language\text{-mltl}\text{-}r \ \psi2 \ r) = \{\}$ 
proof -
  let ?D = LP-mltl-aux (convert-nnf-ext  $\varphi$ ) k
  let ? $\varphi$  = convert-nnf-ext  $\varphi$ 
  have cond1: intervals-welldef (to-mltl (convert-nnf-ext  $\varphi$ ))
    using intervals-welldef
    by (metis convert-nnf-ext-to-mltl-commute nnf-intervals-welldef)
  have cond2:  $\exists \varphi\text{-init}. \text{convert-nnf-ext} \ \varphi = \text{convert-nnf-ext} \ \varphi\text{-init}$ 
    by blast
  have cond3: is-composition-MLTL-allones (convert-nnf-ext  $\varphi$ )
    using composition
    by (simp add: intervals-welldef is-composition-allones-convert-nnf-ext)
  have cond4:  $set (LP\text{-mltl}\text{-aux} (convert\text{-nnf}\text{-ext} \ \varphi) \ k) =$ 
     $set (LP\text{-mltl}\text{-aux} (convert\text{-nnf}\text{-ext} \ \varphi) \ k)$ 
    by blast
  obtain  $\psi1' \ \psi2'$  where  $\psi1: \psi1 = to\text{-mltl} (convert\text{-nnf}\text{-ext} \ \psi1')$ 
    and  $\psi1'\text{-in}: \psi1' \in set \ ?D$ 
    and  $\psi2: \psi2 = to\text{-mltl} (convert\text{-nnf}\text{-ext} \ \psi2')$ 
    and  $\psi2'\text{-in}: \psi2' \in set \ ?D$ 
    using D-decomp unfolding LP-mltl.simps
    using diff-formulas by auto
  have  $\psi1'\text{-neq}: \psi1' \neq \psi2'$ 
    using diff-formulas  $\psi1 \ \psi2$  by blast
  have  $\psi1\text{-welldef}: intervals\text{-welldef} \ \psi1$ 
    using assms(4) D-decomp unfolding LP-mltl.simps
    using LP-mltl-aux-intervals-welldef
    by (metis  $\psi1 \ \psi1'\text{-in}$  allones-implies-is-composition-MLTL composition convert-nnf-ext-to-mltl-commute intervals-welldef nnf-intervals-welldef)
  then have  $\psi1'\text{-welldef}: intervals\text{-welldef} (to\text{-mltl} \ \psi1')$ 
    using  $\psi1$ 

```

```

using LP-mltl-aux-intervals-welldef  $\psi 1'$ -in allones-implies-is-composition-MLTL
composition intervals-welldef by auto
have  $\psi 2$ -welldef: intervals-welldef  $\psi 2$ 
using assms(4) D-decomp unfolding LP-mltl.simps
using LP-mltl-aux-intervals-welldef
by (metis  $\psi 2$   $\psi 2'$ -in allones-implies-is-composition-MLTL composition con-
vert-nnf-ext-to-mltl-commute intervals-welldef nnf-intervals-welldef)
then have  $\psi 2'$ -welldef: intervals-welldef (to-mltl  $\psi 2'$ )
using  $\psi 2$ 
using LP-mltl-aux-intervals-welldef  $\psi 2'$ -in allones-implies-is-composition-MLTL
composition intervals-welldef by auto
have intersect: language-mltl-r (to-mltl  $\psi 1'$ )  $r \cap$ 
language-mltl-r (to-mltl  $\psi 2'$ )  $r = \{\}$ 
using LP-mltl-language-disjoint-aux[OF cond1 cond2 cond3 cond4, of  $\psi 1'$   $\psi 2'$ 
 $r$ ]
using  $\psi 1'$ -in  $\psi 2'$ -in  $\psi$ 's-neq  $r$ -wpd
by (metis convert-nnf-ext-preserves-wpd)
have semantics-mltl  $\pi$  (to-mltl (convert-nnf-ext  $\varphi$ )) =
semantics-mltl  $\pi$  (to-mltl  $\varphi$ )
if intervals-welldef (to-mltl  $\varphi$ )
for  $\varphi::'a$  mltl-ext and  $\pi$ 
using that unfolding semantic-equiv-ext-def
by (metis convert-nnf-ext-to-mltl-commute convert-nnf-preserves-semantics)
then show ?thesis using intersect
unfolding language-mltl-r-def  $\psi 1$   $\psi 2$ 
using  $\psi 1'$ -welldef  $\psi 2'$ -welldef
by auto
qed

```

8.4 Disjointedness Theorem (special case of $k=1$)

```

lemma LP-mltl-language-disjoint-aux-helper-k1:
fixes  $\varphi$   $\psi 1$   $\psi 2::'a$  mltl-ext and  $\pi::'a$  set list
assumes intervals-welldef: intervals-welldef (to-mltl  $\varphi$ )
assumes is-nnf:  $\exists \varphi$ -init.  $\varphi =$  convert-nnf-ext  $\varphi$ -init
assumes composition: is-composition-MLTL  $\varphi$ 
assumes tracelen: length  $\pi \geq$  wpd-mltl (to-mltl  $\varphi$ )
assumes D-decomp:  $D =$  set (LP-mltl-aux  $\varphi$  (Suc 0))
assumes diff-formulas:  $(\psi 1 \in D) \wedge (\psi 2 \in D) \wedge \psi 1 \neq \psi 2$ 
assumes sat1: semantics-mltl-ext  $\pi$   $\psi 1$ 
assumes sat2: semantics-mltl-ext  $\pi$   $\psi 2$ 
shows False
proof(cases  $\varphi$ )
case True-mltl-ext
then show ?thesis using assms
unfolding True-mltl-ext LP-mltl.simps LP-mltl-aux.simps
by auto
next
case False-mltl-ext

```

```

then show ?thesis using assms
  unfolding False-mltl-ext LP-mltl.simps LP-mltl-aux.simps
  by auto
next
case (Prop-mltl-ext p)
then show ?thesis using assms
  unfolding Prop-mltl-ext LP-mltl.simps LP-mltl-aux.simps
  by auto
next
case (Not-mltl-ext q)
then have  $\exists p. q = \text{Prop-mltl-ext } p$ 
  using convert-nnf-form-Not-Implies-Prop assms
  by (metis convert-nnf-ext-to-mltl-commute to-mltl.simps(4) to-mltl-prop-bijective)

then obtain  $p$  where  $q = \text{Prop-mltl-ext } p$  by blast
then show ?thesis
  using assms unfolding Not-mltl-ext LP-mltl.simps LP-mltl-aux.simps
  by auto
next
case (And-mltl-ext  $\alpha \beta$ )
show ?thesis
  using assms(5) unfolding And-mltl-ext LP-mltl-aux.simps
  using assms(6) by auto
next
case (Or-mltl-ext  $\alpha \beta$ )
let ?Dx = [convert-nnf-ext  $\alpha$ ]
let ?Dy = [convert-nnf-ext  $\beta$ ]
have D-is:  $D = \text{set } ( \text{And-mltl-list } ?Dx ?Dy @
  \text{And-mltl-list } [\text{Not}_c \alpha] ?Dy @
  \text{And-mltl-list } ?Dx [\text{Not}_c \beta] )$ 
  using assms(5) unfolding Or-mltl-ext LP-mltl-aux.simps
  by metis
then have  $\psi 1$ -eo:  $\text{List.member } (\text{And-mltl-list } ?Dx ?Dy) \psi 1 \vee
  \text{List.member } (\text{And-mltl-list } [\text{Not}_c \alpha] ?Dy) \psi 1 \vee
  \text{List.member } (\text{And-mltl-list } ?Dx [\text{Not}_c \beta]) \psi 1$ 
  using assms(6) by (simp add: member-def)
have  $\psi 2$ -eo:  $\text{List.member } (\text{And-mltl-list } ?Dx ?Dy) \psi 2 \vee
  \text{List.member } (\text{And-mltl-list } [\text{Not}_c \alpha] ?Dy) \psi 2 \vee
  \text{List.member } (\text{And-mltl-list } ?Dx [\text{Not}_c \beta]) \psi 2$ 
  using D-is assms(6) by (simp add: member-def)

have  $\alpha$ -iwd: intervals-welldef (to-mltl  $\alpha$ )
  using assms(1) unfolding Or-mltl-ext by simp
have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
  using assms(2) unfolding Or-mltl-ext
  by (metis convert-nnf-ext.simps(5) convert-nnf-ext-convert-nnf-ext mtl-ext.inject(4))
have  $\alpha$ -is-comp: is-composition-MLTL  $\alpha$ 
  using assms unfolding Or-mltl-ext by simp
have  $\alpha$ -wpd: wpd-mltl (to-mltl  $\alpha$ )  $\leq$  length  $\pi$ 

```

```

using assms unfolding Or-mltl-ext by simp
have  $\alpha$ -conv-same: set (LP-mltl-aux (convert-nnf-ext  $\alpha$ ) 1) = set (LP-mltl-aux
 $\alpha$  1)
  by (metis  $\alpha$ -nnf convert-nnf-ext-convert-nnf-ext)

have  $\beta$ -iwd: intervals-welldef (to-mltl  $\beta$ )
  using assms unfolding Or-mltl-ext
  by simp
have  $\beta$ -nnf:  $\exists \varphi$ -init.  $\beta$  = convert-nnf-ext  $\varphi$ -init
  using assms unfolding Or-mltl-ext
by (metis convert-nnf-ext.simps(5) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(4))
have  $\beta$ -is-comp: is-composition-MLTL  $\beta$ 
  using assms unfolding Or-mltl-ext
  by simp
have  $\beta$ -wpd: wpd-mltl (to-mltl  $\beta$ )  $\leq$  length  $\pi$ 
  using assms unfolding Or-mltl-ext by simp
have  $\beta$ -conv-same: set (LP-mltl-aux (convert-nnf-ext  $\beta$ )  $k$ ) = set (LP-mltl-aux
 $\beta$   $k$ )
  by (metis  $\beta$ -nnf convert-nnf-ext-convert-nnf-ext)

{
  assume List.member (And-mltl-list ?Dx ?Dy)  $\psi$ 1
  then have  $\psi$ 1-is:  $\psi$ 1 = And-mltl-ext  $\alpha$   $\beta$ 
    unfolding List.member-def
    using  $\alpha$ -nnf  $\beta$ -nnf convert-nnf-ext-convert-nnf-ext
    by (metis And-mltl-list-member  $\langle$ List.member (And-mltl-list [convert-nnf-ext
 $\alpha$ ] [convert-nnf-ext  $\beta$ ])  $\psi$ 1 $\rangle$  member-rec(1) member-rec(2))
  have  $x$ 1-semantics: semantics-mltl-ext  $\pi$   $\alpha$  and
     $y$ 1-semantics: semantics-mltl-ext  $\pi$   $\beta$ 
    using assms(7) unfolding  $\psi$ 1-is semantics-mltl-ext-def by simp-all
  {
    assume List.member (And-mltl-list ?Dx ?Dy)  $\psi$ 2
    then have  $\psi$ 2-is:  $\psi$ 2 = And-mltl-ext  $\alpha$   $\beta$ 
      unfolding List.member-def
      using  $\alpha$ -nnf  $\beta$ -nnf convert-nnf-ext-convert-nnf-ext
      by (metis And-mltl-list-member-forward  $\langle$ List.member (And-mltl-list
[convert-nnf-ext  $\alpha$ ] [convert-nnf-ext  $\beta$ ])  $\psi$ 2 $\rangle$  member-rec(1) member-rec(2))
    then have ?thesis
      using  $\psi$ 1-is assms by blast
  } moreover {
    assume List.member (And-mltl-list [Not $c$   $\alpha$ ] ?Dy)  $\psi$ 2
    then have  $\psi$ 2-is:  $\psi$ 2 = And-mltl-ext (Not $c$   $\alpha$ )  $\beta$ 
      unfolding List.member-def
      using  $\alpha$ -nnf  $\beta$ -nnf convert-nnf-ext-convert-nnf-ext
      by (metis And-mltl-list-member  $\langle$ List.member (And-mltl-list [Not $c$   $\alpha$ ]
[convert-nnf-ext  $\beta$ ])  $\psi$ 2 $\rangle$  member-rec(1) member-rec(2))
    have  $x$ 2-semantics: semantics-mltl-ext  $\pi$  (Not $c$   $\alpha$ ) and
       $y$ 2-semantics: semantics-mltl-ext  $\pi$   $\beta$ 
      using assms unfolding semantics-mltl-ext-def  $\psi$ 2-is by simp-all
  }
}

```

```

then have ?thesis
  using x1-antics unfolding antics-mltl-ext-def by simp
} moreover {
  assume List.member (And-mltl-list ?Dx [Notc β]) ψ2
  then have ψ2-is: ψ2 = And-mltl-ext α (Notc β)
    unfolding List.member-def
    using α-nnf β-nnf convert-nnf-ext-convert-nnf-ext
  by (metis And-mltl-list-member ⟨List.member (And-mltl-list [convert-nnf-ext
α] [Notc β]) ψ2⟩ member-rec(1) member-rec(2))
  have x2-antics: antics-mltl-ext π α and
    y2-antics: antics-mltl-ext π (Notc β)
    using assms unfolding antics-mltl-ext-def ψ2-is by simp-all
  then have ?thesis
    using y1-antics unfolding antics-mltl-ext-def by simp
}
ultimately have ?thesis
  using ψ2-eo by argo
} moreover {
  assume List.member (And-mltl-list [Notc α] ?Dy) ψ1
  then have ψ1-is: ψ1 = And-mltl-ext (Notc α) (β)
    unfolding List.member-def
    using α-nnf β-nnf convert-nnf-ext-convert-nnf-ext
  by (metis And-mltl-list-member ⟨List.member (And-mltl-list [Notc α] [convert-nnf-ext
β]) ψ1⟩ member-rec(1) member-rec(2))
  have x1-antics: antics-mltl-ext π (Notc α) and
    y1-antics: antics-mltl-ext π (β)
    using assms unfolding antics-mltl-ext-def ψ1-is by simp-all

{
  assume List.member (And-mltl-list ?Dx ?Dy) ψ2
  then have ψ2-is: ψ2 = And-mltl-ext α β
    unfolding List.member-def
    using α-nnf β-nnf convert-nnf-ext-convert-nnf-ext
  by (metis And-mltl-list-member ⟨List.member (And-mltl-list [convert-nnf-ext
α] [convert-nnf-ext β]) ψ2⟩ member-rec(1) member-rec(2))
  have ?thesis
    using assms(7,8) unfolding ψ1-is ψ2-is antics-mltl-ext-def by auto
} moreover {
  assume List.member (And-mltl-list [Notc α] ?Dy) ψ2
  then have ψ2-is: ψ2 = And-mltl-ext (Notc α) β
    unfolding List.member-def
    using α-nnf β-nnf convert-nnf-ext-convert-nnf-ext
    by (metis And-mltl-list-member ⟨List.member (And-mltl-list [Notc α]
[convert-nnf-ext β]) ψ2⟩ member-rec(1) member-rec(2))
  have x2-antics: antics-mltl-ext π (Notc α) and
    y2-antics: antics-mltl-ext π β
    using assms unfolding antics-mltl-ext-def ψ2-is by simp-all
  then have ?thesis
    using ψ1-is ψ2-is assms by blast

```

```

} moreover {
  assume List.member (And-mltl-list ?Dx [Notc β]) ψ2
  then have ψ2-is: ψ2 = And-mltl-ext α (Notc β)
    unfolding List.member-def
    using α-nnf β-nnf convert-nnf-ext-convert-nnf-ext
  by (metis And-mltl-list-member ‹List.member (And-mltl-list [convert-nnf-ext
α] [Notc β]) ψ2› member-rec(1) member-rec(2))
  have x2-semantics: semantics-mltl-ext π α and
    y2-semantics: semantics-mltl-ext π (Notc β)
    using assms unfolding semantics-mltl-ext-def ψ2-is by simp-all
  then have ?thesis
    using y1-semantics unfolding semantics-mltl-ext-def by simp
}
ultimately have ?thesis
  using ψ2-eo by argo
} moreover {
  assume List.member (And-mltl-list ?Dx [Notc β]) ψ1
  then have ψ1-is: ψ1 = And-mltl-ext α (Notc β)
    unfolding List.member-def
    using α-nnf β-nnf convert-nnf-ext-convert-nnf-ext
  by (metis And-mltl-list-member ‹List.member (And-mltl-list [convert-nnf-ext
α] [Notc β]) ψ1› member-rec(1) member-rec(2))
  have x1-semantics: semantics-mltl-ext π α and
    y1-semantics: semantics-mltl-ext π (Notc β)
    using assms unfolding semantics-mltl-ext-def ψ1-is by simp-all

{
  assume List.member (And-mltl-list ?Dx ?Dy) ψ2
  then have ψ2-is: ψ2 = And-mltl-ext α β
    unfolding List.member-def
    using α-nnf β-nnf convert-nnf-ext-convert-nnf-ext
  by (metis And-mltl-list-member-forward ‹List.member (And-mltl-list [convert-nnf-ext
α] [convert-nnf-ext β]) ψ2› member-rec(1) member-rec(2))
  have ?thesis
    using assms(7,8) unfolding ψ1-is ψ2-is semantics-mltl-ext-def by auto
} moreover {
  assume List.member (And-mltl-list [Notc α] ?Dy) ψ2
  then have ψ2-is: ψ2 = And-mltl-ext (Notc α) β
    unfolding List.member-def
    using α-nnf β-nnf convert-nnf-ext-convert-nnf-ext
  by (metis And-mltl-list-member ‹List.member (And-mltl-list [Notc α]
[convert-nnf-ext β]) ψ2› member-rec(1) member-rec(2))
  have x2-semantics: semantics-mltl-ext π (Notc α) and
    y2-semantics: semantics-mltl-ext π β
    using assms unfolding semantics-mltl-ext-def ψ2-is by simp-all
  then have ?thesis
    using x1-semantics x2-semantics unfolding semantics-mltl-ext-def by
auto
} moreover {

```



```

assume List.member (And-mltl-list ?Dx [Notc β]) ψ2
then have ψ2-is: ψ2 = And-mltl-ext α (Notc β)
  unfolding List.member-def
  using α-nnf β-nnf convert-nnf-ext-convert-nnf-ext
by (metis And-mltl-list-member ⟨List.member (And-mltl-list [convert-nnf-ext
α] [Notc β]) ψ2⟩ member-rec(1) member-rec(2))
  have x2-semantics: semantics-mltl-ext π α and
    y2-semantics: semantics-mltl-ext π (Notc β)
  using assms unfolding semantics-mltl-ext-def ψ2-is by simp-all
then have ?thesis
  using ψ1-is ψ2-is assms by blast
}
ultimately have ?thesis
  using ψ2-eo by argo
}
ultimately show ?thesis
  using ψ1-eo by argo
next
case (Future-mltl-ext a b L α)
have a-leq-b: a ≤ b and
  α-welldef: intervals-welldef (to-mltl α)
  using assms unfolding intervals-welldef.simps Future-mltl-ext to-mltl.simps
  by simp-all
have α-nnf: ∃φ-init. α = convert-nnf-ext φ-init
  using assms unfolding Future-mltl-ext
by (metis convert-nnf-ext.simps(6) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(5))

have α-convert: convert-nnf-ext α = α
  using α-nnf convert-nnf-ext-convert-nnf-ext by metis
have α-composition: is-composition-MLTL α and
  L-composition: is-composition (b-a+1) L
  using Future-mltl-ext assms by simp-all
have α-wpd: b + wpd-mltl (to-mltl α) ≤ length π
  using assms unfolding Future-mltl-ext to-mltl.simps wpd-mltl.simps
  by auto
let ?D = [α]
let ?s = interval-times a L
have length-L: 1 ≤ length L
  using composition-length-lb[OF L-composition] a-leq-b by linarith
have sfirst: ?s!0 = a
  using interval-times-first by simp
have slast: ?s!(length L) = b+1
  using interval-times-last[OF a-leq-b L-composition] by blast
have length-s: length ?s = length L + 1
  using interval-times-length by simp
let ?front = set [Future-mltl-ext (?s! 0) (?s! 1 - 1) [?s! 1 - ?s! 0] α]
let ?back = set (concat (map (λi. And-mltl-list
  [Global-mltl-ext (?s! 0) (?s! i - 1) [?s!i - ?s!0] (Notc α)]
  [Future-mltl-ext (?s! i) (?s! (i + 1) - 1) [?s! (i + 1) -
```

```

?s ! i]  $\alpha$ )
      [1.. $\text{length } L$ ]))
  have front-eq: set (Future-mltl-list ?D (?s ! 0) (?s ! 1 - 1) [?s ! 1 - ?s ! 0])
= ?front
  by simp
  have back-eq: ?back = set (concat
    (map ( $\lambda i$ . And-mltl-list
      [Global-mltl-ext (?s ! 0) (?s ! i - 1) [?s!i - ?s!0] (Notc  $\alpha$ )]
      (Future-mltl-list ?D (?s ! i) (?s ! (i + 1) - 1)
        [?s ! (i + 1) - ?s ! i]))
      [1.. $\text{length } L$ ]))
  by auto
  have D-is: D = ?front  $\cup$  ?back
  using assms(5) unfolding Future-mltl-ext LP-mltl-aux.simps to-mltl.simps
  using list-concat-set-union unfolding  $\alpha$ -convert
  using front-eq back-eq
  by (metis (no-types, lifting))
  have dropa-wpd: wpd-mltl (to-mltl  $\alpha$ )  $\leq$  length (drop a  $\pi$ )
  using  $\alpha$ -wpd a-leq-b by simp
  {
    assume *:  $\psi_1 \in ?front$ 
    then have  $\psi_1$ :  $\psi_1 = \text{Future-mltl-ext } (?s!0) (?s!1-1) [?s!1 - ?s!0] \alpha$ 
      by auto
    obtain j1 where  $\alpha$ -semantics1: semantics-mltl-ext (drop j1  $\pi$ )  $\alpha$ 
      and j1-bound:  $a \leq j1 \wedge j1 \leq ?s!1-1$ 
    using assms(7) unfolding sfirst  $\psi_1$  semantics-mltl-ext-def semantics-mltl.simps
to-mltl.simps
      by blast
    {
      assume **:  $\psi_2 \in ?front$ 
      then have  $\psi_2$ :  $\psi_2 = \text{Future-mltl-ext } (?s!0) (?s!1-1) [?s!1 - ?s!0] \alpha$ 
        by auto
      obtain j2 where  $\alpha$ -semantics-2: semantics-mltl-ext (drop j2  $\pi$ )  $\alpha$ 
        and j2-bound:  $a \leq j2 \wedge j2 \leq ?s!1-1$ 
      using assms(8) unfolding sfirst  $\psi_2$  semantics-mltl-ext-def seman-
tics-mltl.simps to-mltl.simps
        by blast
      have ?thesis
        using assms(6)  $\psi_1$   $\psi_2$  by blast
    } moreover {
      assume **:  $\psi_2 \in ?back$ 
      then obtain i2 where  $\psi_2$ :  $\psi_2 = (\text{And-mltl-ext }
        (\text{Global-mltl-ext } (?s ! 0) (?s ! i2 - 1) [?s!i2 - ?s!0] (\text{Not}_c \alpha))
        (\text{Future-mltl-ext } (?s ! i2) (?s ! (i2 + 1) - 1) [?s ! (i2 + 1)
- ?s ! i2] \alpha))
        and i2-bound:  $1 \leq i2 \wedge i2 < \text{length } L$ 
      by force
      obtain j2 where  $\alpha$ -semantics2: semantics-mltl-ext (drop j2  $\pi$ )  $\alpha$ 
        and j2-bound: ?s!i2  $\leq$  j2  $\wedge$  j2  $\leq$  ?s!(i2+1)-1$ 
```

```

and global-before2:  $\forall i. a \leq i \wedge i \leq ?s ! i2 - 1 \longrightarrow$ 
 $\neg \text{semantics-mltl } (\text{drop } i \ \pi) \ (\text{to-mltl } \alpha)$ 
using assms(8) unfolding  $\psi2$  semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps
unfolding sfirst using  $\alpha$ -wpd a-leq-b by auto
have bound1: interval-times a L ! 1 ≤ interval-times a L ! i2
using interval-times-diff-ge-general[OF a-leq-b L-composition, of i2 1 ?s]
using i2-bound by force
have ?thesis using bound1
using  $\alpha$ -semantics1 global-before2 j1-bound unfolding semantics-mltl-ext-def
by auto
}
ultimately have ?thesis
using assms(6) D-is by blast
} moreover {
assume *:  $\psi1 \in ?back$ 
then obtain i1 where  $\psi1: \psi1 = (\text{And-mltl-ext}$ 
 $(\text{Global-mltl-ext } (?s ! 0) (?s ! i1 - 1) [?s!i1 - ?s!0] (\text{Not}_c \alpha))$ 
 $(\text{Future-mltl-ext } (?s ! i1) (?s ! (i1 + 1) - 1) [?s ! (i1 + 1)$ 
-  $?s ! i1] \alpha))$ 
and i1-bound:  $1 \leq i1 \wedge i1 < \text{length } L$ 
by force
have lb1:  $a \leq ?s!i1$ 
using interval-times-diff-ge-general[OF a-leq-b L-composition, of i1 0 ?s]
unfolding sfirst using i1-bound by simp
have welldef1:  $?s!i1 < ?s!(i1+1)$ 
using interval-times-diff-ge[OF a-leq-b L-composition, of i1 ?s]
using i1-bound by blast
have ub1:  $?s!(i1+1)-1 \leq b$ 
using interval-times-diff-ge-general[OF a-leq-b L-composition, of length L
i1+1 ?s]
using slast i1-bound
by (metis le-diff-conv le-eq-less-or-eq less-iff-succ-less-eq)
obtain j1 where  $\alpha$ -semantics1: semantics-mltl-ext (drop j1 π) α
and j1-bound:  $?s!i1 \leq j1 \wedge j1 \leq ?s!(i1+1)-1$ 
and global-before1:  $\forall i. a \leq i \wedge i \leq ?s ! i1 - 1 \longrightarrow$ 
 $\neg \text{semantics-mltl } (\text{drop } i \ \pi) \ (\text{to-mltl } \alpha)$ 
using assms(7) unfolding  $\psi1$  semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps
unfolding sfirst using  $\alpha$ -wpd a-leq-b by auto
have bound1: interval-times a L ! 1 ≤ interval-times a L ! i1
using interval-times-diff-ge-general[OF a-leq-b L-composition, of i1 1 ?s]
using i1-bound by force
{
assume **:  $\psi2 \in ?front$ 
then have  $\psi2: \psi2 = \text{Future-mltl-ext } (?s!0) (?s!1-1) [?s!1 - ?s!0] \alpha$ 
by auto
obtain j2 where  $\alpha$ -semantics2: semantics-mltl-ext (drop j2 π) α
and j2-bound:  $a \leq j2 \wedge j2 \leq ?s!1-1$ 

```

```

      using assms(8) unfolding sfirst  $\psi_2$  semantics-mltl-ext-def semantics-mltl.simps to-mltl.simps
      by blast
    then have ?thesis
      using global-before1  $\alpha$ -semantics2 bound1
      unfolding semantics-mltl-ext-def by auto
  } moreover {
    assume **:  $\psi_2 \in ?back$ 
    then obtain i2 where  $\psi_2: \psi_2 = (And\text{-mltl}\text{-ext}$ 
      (Global-mltl-ext ( $?s!0$ ) ( $?s!i_2 - 1$ ) [ $?s!i_2 - ?s!0$ ] (Notc  $\alpha$ ))
      (Future-mltl-ext ( $?s!i_2$ ) ( $?s!(i_2 + 1) - 1$ ) [ $?s!(i_2 + 1)$ 
-  $?s!i_2$ ]  $\alpha$ ))
      and i2-bound:  $1 \leq i_2 \wedge i_2 < \text{length } L$ 
      by force
    obtain j2 where  $\alpha$ -semantics2: semantics-mltl-ext (drop j2  $\pi$ )  $\alpha$ 
      and j2-bound:  $?s!i_2 \leq j_2 \wedge j_2 \leq ?s!(i_2+1)-1$ 
      and global-before2:  $\forall i. a \leq i \wedge i \leq ?s!i_2 - 1 \longrightarrow$ 
         $\neg \text{semantics-mltl}(\text{drop } i \pi)(\text{to-mltl } \alpha)$ 
    using assms(8) unfolding  $\psi_2$  semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps
    unfolding sfirst using  $\alpha$ -wpd a-leq-b by auto
    have lb2:  $a \leq ?s!i_2$ 
      using interval-times-diff-ge-general[OF a-leq-b L-composition, of i2 0 ?s]
    unfolding sfirst using i2-bound by simp
    have welldef2:  $?s!i_2 < ?s!(i_2+1)$ 
      using interval-times-diff-ge[OF a-leq-b L-composition, of i2 ?s]
    using i2-bound by blast
    have ub2:  $?s!(i_2+1)-1 \leq b$ 
      using interval-times-diff-ge-general[OF a-leq-b L-composition, of length L
i2+1 ?s]
    using slast i2-bound
    by (metis le-diff-conv le-eq-less-or-eq less-iff-succ-less-eq)
  {
    assume i1-eq-i2:  $i_1 = i_2$ 
    then have ?thesis
      using assms(6)  $\psi_1$   $\psi_2$  by blast
  } moreover {
    assume i1-le-i2:  $i_1 < i_2$ 
    then have  $?s!(i_1 + 1) \leq ?s!i_2$ 
      using interval-times-diff-ge-general[OF a-leq-b L-composition, of i2 i1+1
?s]
    using i1-bound i2-bound
    by (metis le-eq-less-or-eq less-iff-succ-less-eq)
    then have  $j_1 \leq \text{interval-times } a \ L!i_2 - 1$ 
      using j1-bound by auto
    then have ?thesis
      using  $\alpha$ -semantics1 global-before2 j1-bound lb1
      unfolding semantics-mltl-ext-def by simp
  } moreover {

```

```

assume  $i1\text{-ge-}i2: i1 > i2$ 
then have  $?s ! (i2 + 1) \leq ?s ! i1$ 
  using interval-times-diff-ge-general[OF a-leq-b L-composition, of i1 i2+1
?S]
    using  $i2\text{-bound } i1\text{-bound}$ 
    by (metis le-eq-less-or-eq less-iff-succ-less-eq)
    then have  $j2 \leq \text{interval-times } a \ L ! i1 - 1$ 
    using  $j2\text{-bound}$  by auto
    then have  $?thesis$ 
    using  $\alpha\text{-semantics2 } global\text{-before1 } j2\text{-bound } lb2$ 
    unfolding semantics-mltl-ext-def by simp
  }
  ultimately have  $?thesis$  by linarith
}
ultimately have  $?thesis$ 
  using assms(6) D-is by blast
}
ultimately show  $?thesis$ 
  using assms(6) D-is by blast
next
case (Global-mltl-ext a b L  $\alpha$ )
have  $a\text{-leq-}b: a \leq b$  and
   $\alpha\text{-welldef: intervals-welldef (to-mltl } \alpha)$ 
  using assms unfolding intervals-welldef.simps Global-mltl-ext to-mltl.simps
  by simp-all
have  $\alpha\text{-nnf: } \exists \varphi\text{-init. } \alpha = \text{convert-}nnf\text{-ext } \varphi\text{-init}$ 
  using assms unfolding Global-mltl-ext
by (metis convert-}nnf\text{-ext.simps(7) convert-}nnf\text{-ext-convert-}nnf\text{-ext mltl-ext.inject(6))

have  $\alpha\text{-convert: convert-}nnf\text{-ext } \alpha = \alpha$ 
  using  $\alpha\text{-nnf convert-}nnf\text{-ext-convert-}nnf\text{-ext}$  by metis
have  $\alpha\text{-composition: is-composition-MLTL } \alpha$ 
  using Global-mltl-ext assms by simp-all
have  $\alpha\text{-wpd: } b + \text{wpd-mltl (to-mltl } \alpha) \leq \text{length } \pi$ 
  using assms unfolding Global-mltl-ext to-mltl.simps wpd-mltl.simps
  by auto
have  $D\text{-is: } D = \{Global\text{-mltl-ext } a \ b \ L \ \alpha\}$ 
  using assms(5) unfolding Global-mltl-ext LP-mltl-aux.simps  $\alpha\text{-convert}$ 
  by auto
then show  $?thesis$ 
  using assms by blast
next
case (Until-mltl-ext  $\alpha$  a b L  $\beta$ )
have  $a\text{-leq-}b: a \leq b$  and
   $\alpha\text{-welldef: intervals-welldef (to-mltl } \alpha)$  and
   $\beta\text{-welldef: intervals-welldef (to-mltl } \beta)$ 
  using assms unfolding intervals-welldef.simps Until-mltl-ext to-mltl.simps
  by simp-all
have  $\alpha\text{-nnf: } \exists \varphi\text{-init. } \alpha = \text{convert-}nnf\text{-ext } \varphi\text{-init}$ 

```

```

using assms unfolding Until-mltl-ext
by (metis convert-nnf-ext.simps(8) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(7))

have  $\alpha$ -convert: convert-nnf-ext  $\alpha = \alpha$ 
using  $\alpha$ -nnf convert-nnf-ext-convert-nnf-ext by metis
have  $\beta$ -nnf:  $\exists \varphi$ -init.  $\beta = \text{convert-nnf-ext } \varphi$ -init
using assms unfolding Until-mltl-ext
by (metis convert-nnf-ext.simps(8) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(7))

have  $\beta$ -convert: convert-nnf-ext  $\beta = \beta$ 
using  $\beta$ -nnf convert-nnf-ext-convert-nnf-ext by metis
have  $\alpha$ -composition: is-composition-MLTL  $\alpha$  and
   $\beta$ -composition: is-composition-MLTL  $\beta$  and
   $L$ -composition: is-composition  $(b-a+1)$   $L$ 
using Until-mltl-ext assms by simp-all
have  $\alpha$ -wpd:  $b + \text{wpd-mltl } (\text{to-mltl } \alpha) \leq \text{length } \pi$  and
   $\beta$ -wpd:  $b + \text{wpd-mltl } (\text{to-mltl } \beta) \leq \text{length } \pi$ 
using assms unfolding Until-mltl-ext to-mltl.simps wpd-mltl.simps
by auto
let  $?s = \text{interval-times } a$   $L$ 
have  $\text{length-}L$ :  $1 \leq \text{length } L$ 
using composition-length-lb[OF L-composition] a-leq-b by linarith
have  $s$ -first:  $?s!0 = a$ 
using interval-times-first by simp
have  $s$ -last:  $?s!(\text{length } L) = b+1$ 
using interval-times-last[OF a-leq-b L-composition]
by blast
have  $\text{length-}s$ :  $\text{length } ?s = \text{length } L + 1$ 
using interval-times-length by simp
let  $?D = [\beta]$ 
let  $?front = \{ \text{Until-mltl-ext } \alpha (?s!0) (?s!1 - 1) [?s!1 - ?s!0] \beta \}$ 
let  $?back = \text{set } (\text{map } (\lambda i. \text{And-mltl-ext}$ 
  (Global-mltl-ext
     $(?s!0) (?s!i - 1) [?s!i - ?s!0] (\text{And-mltl-ext } \alpha (\text{Not}_c$ 
       $\beta)))$ 
    (Until-mltl-ext  $\alpha (?s!i) (?s!(i+1) - 1)$ 
       $[?s!(i+1) - ?s!i] \beta)) [1..<\text{length } L])$ 
have  $\text{front-eq}$ :  $?front = \text{set } (\text{Until-mltl-list } \alpha ?D (?s!0) (?s!1 - 1) [?s!1 - ?s!0])$ 
by simp
have  $\text{back-eq}$ :  $?back = \text{set } (\text{concat}$ 
  (map  $(\lambda i. \text{And-mltl-list}$ 
    [Global-mltl-ext
       $(?s!0) (?s!i - 1) [?s!i - ?s!0] (\text{And-mltl-ext } \alpha (\text{Not}_c \beta))$ 
      (Until-mltl-list  $\alpha ?D (?s!i) (?s!(i+1) - 1)$ 
         $[?s!(i+1) - ?s!i]))$ 
     $[1..<\text{length } L]))$ 
by simp
have  $D$ -is:  $D = ?front \cup ?back$ 

```

```

using assms(5) unfolding Until-mltl-ext LP-mltl-aux.simps
using  $\alpha$ -convert  $\beta$ -convert list-concat-set-union using front-eq back-eq
by (smt (verit) map-eq-conv)
{
  assume *:  $\psi 1 \in ?front$ 
  then have  $\psi 1$ :  $\psi 1 = \text{Until-mltl-ext } \alpha \text{ } (?s ! 0) \text{ } (?s ! 1 - 1) \text{ } [?s ! 1 - ?s ! 0]$ 
}
 $\beta$ 
  by blast
  obtain  $j1$  where  $j1$ -bound:  $?s!0 \leq j1 \wedge j1 \leq ?s!1-1$ 
    and  $\beta$ -semantics1: semantics-mltl-ext (drop  $j1$   $\pi$ )  $\beta$ 
    and  $\alpha$ -semantics1:  $\forall j. (?s!0 \leq j \wedge j < j1) \longrightarrow (\text{semantics-mltl-ext}$ 
(drop  $j$   $\pi$ )  $\alpha$ )
    using assms(7) unfolding  $\psi 1$  semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps
    by blast
    {
      assume **:  $\psi 2 \in ?front$ 
      then have  $\psi 2$ :  $\psi 2 = \text{Until-mltl-ext } \alpha \text{ } (?s ! 0) \text{ } (?s ! 1 - 1) \text{ } [?s ! 1 - ?s ! 0]$ 
    }
     $\beta$ 
    by blast
    obtain  $j2$  where  $j2$ -bound:  $?s!0 \leq j2 \wedge j2 \leq ?s!1-1$ 
      and  $\beta$ -semantics2: semantics-mltl-ext (drop  $j2$   $\pi$ )  $\beta$ 
      and  $\alpha$ -semantics2:  $\forall j. (?s!0 \leq j \wedge j < j2) \longrightarrow (\text{semantics-mltl-ext}$ 
(drop  $j2$   $\pi$ )  $\alpha$ )
      using assms(8) unfolding  $\psi 2$  semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps
      using  $\psi 1$   $\psi 2$  diff-formulas by blast
      have ?thesis
      using  $\psi 1$   $\psi 2$  diff-formulas by blast
    } moreover {
      assume **:  $\psi 2 \in ?back$ 
      then obtain  $i2$  where  $\psi 2$ :  $\psi 2 = \text{And-mltl-ext}$ 
      (Global-mltl-ext ( $?s ! 0$ ) ( $?s ! i2 - 1$ ) [ $?s!i2 - ?s!0$ ] (And-mltl-ext
 $\alpha$  (Notc  $\beta$ )))
      (Until-mltl-ext  $\alpha$  ( $?s ! i2$ ) ( $?s ! (i2 + 1) - 1$ ) [ $?s ! (i2 + 1) -$ 
 $?s ! i2$ ]  $\beta$ )
      and  $i2$ -bound:  $1 \leq i2 \wedge i2 < \text{length } L$ 
      by auto
      obtain  $j2$  where  $j2$ -bound:  $(?s ! i2) \leq j2 \wedge j2 \leq (?s ! (i2 + 1) - 1)$ 
      and  $\beta$ -semantics2: semantics-mltl (drop  $j2$   $\pi$ ) (to-mltl  $\beta$ )
      and  $\alpha$ -semantics2:  $(\forall j. \text{interval-times } a \text{ } L ! i2 \leq j \wedge j < j2 \longrightarrow$ 
semantics-mltl (drop  $j$   $\pi$ ) (to-mltl  $\alpha$ ))
      and global-before2:  $\forall i. ?s ! 0 \leq i \wedge i \leq ?s ! i2 - 1 \longrightarrow$ 
semantics-mltl (drop  $i$   $\pi$ ) (to-mltl  $\alpha$ )  $\wedge$ 
 $\neg$  semantics-mltl (drop  $i$   $\pi$ ) (to-mltl  $\beta$ )
      using assms(8) unfolding  $\psi 2$  semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps
      using  $\alpha$ -wpd by auto
      have bound1:  $?s ! 1 \leq ?s ! i2$ 

```

```

    using interval-times-diff-ge-general[OF a-leq-b L-composition, of i2 1 ?s]
    using i2-bound by force
  then have ?thesis
    using  $\beta$ -semantics1 global-before2 j1-bound unfolding sfirst
    unfolding semantics-mltl-ext-def by auto
  }
  ultimately have ?thesis using D-is assms by blast
} moreover {
  assume *:  $\psi 1 \in ?back$ 
  then obtain i1 where  $\psi 1: \psi 1 = \text{And-mltl-ext}$ 
    (Global-mltl-ext (?s ! 0) (?s ! i1 - 1) [?s!i1 - ?s!0] (And-mltl-ext
 $\alpha$  (Notc  $\beta$ )))
    (Until-mltl-ext  $\alpha$  (?s ! i1) (?s ! (i1 + 1) - 1) [?s ! (i1 + 1) -
?s ! i1]  $\beta$ )
    and i1-bound:  $1 \leq i1 \wedge i1 < \text{length } L$ 
  by auto
  have lb1:  $a \leq ?s!i1$ 
    using interval-times-diff-ge-general[OF a-leq-b L-composition, of i1 0 ?s]
    unfolding sfirst using i1-bound by simp
  have welldef1:  $?s!i1 < ?s!(i1+1)$ 
    using interval-times-diff-ge[OF a-leq-b L-composition, of i1 ?s]
    using i1-bound by blast
  have ub1:  $?s!(i1+1)-1 \leq b$ 
    using interval-times-diff-ge-general[OF a-leq-b L-composition, of length L
i1+1 ?s]
    using slast i1-bound
    by (metis le-diff-conv le-eq-less-or-eq less-iff-succ-less-eq)
  obtain j1 where j1-bound:  $(?s ! i1) \leq j1 \wedge j1 \leq (?s ! (i1 + 1) - 1)$ 
    and  $\beta$ -semantics1:  $\text{semantics-mltl (drop } j1 \ \pi) (to\text{-mltl } \beta)$ 
    and  $\alpha$ -semantics1:  $(\forall j. \text{interval-times } a \ L ! i1 \leq j \wedge j < j1 \longrightarrow$ 
       $\text{semantics-mltl (drop } j \ \pi) (to\text{-mltl } \alpha))$ 
    and global-before1:  $\forall i. ?s ! 0 \leq i \wedge i \leq ?s ! i1 - 1 \longrightarrow$ 
       $\text{semantics-mltl (drop } i \ \pi) (to\text{-mltl } \alpha) \wedge$ 
       $\neg \text{semantics-mltl (drop } i \ \pi) (to\text{-mltl } \beta)$ 
    using assms(7) unfolding  $\psi 1$  semantics-mltl-ext-def to-mltl.simps seman-
tics-mltl.simps
    using  $\alpha$ -wpd by auto
  have bound1:  $?s ! 1 \leq ?s ! i1$ 
    using interval-times-diff-ge-general[OF a-leq-b L-composition, of i1 1 ?s]
    using i1-bound by force
  {
    assume **:  $\psi 2 \in ?front$ 
    then have  $\psi 2: \psi 2 = \text{Until-mltl-ext } \alpha$  (?s ! 0) (?s ! 1 - 1) [?s ! 1 - ?s !
0]  $\beta$ 
      by blast
    have ?thesis
      using assms(8) unfolding  $\psi 2$  semantics-mltl-ext-def to-mltl.simps seman-
tics-mltl.simps
      unfolding sfirst

```



```

by (smt (verit, ccfv-SIG) bound1 diff-is-0-eq' global-before1 interval-times-first
le0 le-trans nat-le-linear ordered-cancel-comm-monoid-diff-class.le-diff-conv2)
} moreover {
  assume **:  $\psi_2 \in ?back$ 
  then obtain  $i_2$  where  $\psi_2: \psi_2 = \text{And-mltl-ext}$ 
    ( $\text{Global-mltl-ext } (?s ! 0) (?s ! i_2 - 1) [?s!i_2 - ?s!0] (\text{And-mltl-ext}$ 
 $\alpha (\text{Not}_c \beta))$ )
    ( $\text{Until-mltl-ext } \alpha (?s ! i_2) (?s ! (i_2 + 1) - 1) [?s ! (i_2 + 1) -$ 
 $?s ! i_2] \beta$ )
    and  $i_2\text{-bound}: 1 \leq i_2 \wedge i_2 < \text{length } L$ 
  by auto
  have  $lb_2: a \leq ?s!i_2$ 
  using interval-times-diff-ge-general[OF a-leq-b L-composition, of  $i_2$  0  $?s$ ]
  unfolding sfirst using  $i_2\text{-bound}$  by simp
  have  $welldef_2: ?s!i_2 < ?s!(i_2+1)$ 
  using interval-times-diff-ge[OF a-leq-b L-composition, of  $i_2$   $?s$ ]
  using  $i_2\text{-bound}$  by blast
  have  $ub_2: ?s!(i_2+1) - 1 \leq b$ 
  using interval-times-diff-ge-general[OF a-leq-b L-composition, of length  $L$ 
 $i_2+1$   $?s$ ]
  using slast  $i_2\text{-bound}$ 
  by (metis le-diff-conv le-eq-less-or-eq less-iff-succ-less-eq)
  obtain  $j_2$  where  $j_2\text{-bound}: (?s ! i_2) \leq j_2 \wedge j_2 \leq (?s ! (i_2 + 1) - 1)$ 
    and  $\beta\text{-semantics}_2: \text{semantics-mltl } (\text{drop } j_2 \pi) (\text{to-mltl } \beta)$ 
    and  $\alpha\text{-semantics}_2: (\forall j. \text{interval-times } a \ L \ ! \ i_2 \leq j \wedge j < j_2 \longrightarrow$ 
 $\text{semantics-mltl } (\text{drop } j \pi) (\text{to-mltl } \alpha))$ 
    and  $\text{global-before}_2: \forall i. ?s ! 0 \leq i \wedge i \leq ?s ! i_2 - 1 \longrightarrow$ 
 $\text{semantics-mltl } (\text{drop } i \pi) (\text{to-mltl } \alpha) \wedge$ 
 $\neg \text{semantics-mltl } (\text{drop } i \pi) (\text{to-mltl } \beta)$ 
  using  $\text{assms}(8)$  unfolding  $\psi_2$   $\text{semantics-mltl-ext-def}$   $\text{to-mltl.simps}$   $\text{semantics-mltl.simps}$ 
  using  $\alpha\text{-wpd}$  by auto
  {
    assume  $i_1\text{-eq-}i_2: i_1 = i_2$ 
    then have  $?thesis$ 
    using  $\text{assms}(6)$   $\psi_1$   $\psi_2$  by blast
  } moreover {
    assume  $i_1\text{-le-}i_2: i_1 < i_2$ 
    then have  $?s ! (i_1 + 1) \leq ?s ! i_2$ 
    using interval-times-diff-ge-general[OF a-leq-b L-composition, of  $i_2$   $i_1+1$ 
 $?s$ ]
    using  $i_1\text{-bound}$   $i_2\text{-bound}$ 
    by (metis le-eq-less-or-eq less-iff-succ-less-eq)
    then have  $?thesis$ 
    using  $\beta\text{-semantics}_1$   $\text{global-before}_2$   $j_1\text{-bound}$  unfolding sfirst
    using  $lb_1$  by auto
  } moreover {
    assume  $i_1\text{-ge-}i_2: i_1 > i_2$ 
    then have  $?s ! (i_2 + 1) \leq ?s ! i_1$ 

```

```

    using interval-times-diff-ge-general[OF a-leq-b L-composition, of i1 i2+1
?s]
    using i1-bound i2-bound
    by (metis le-eq-less-or-eq less-iff-succ-less-eq)
  then have ?thesis
    using  $\beta$ -semantics2 global-before1 j2-bound unfolding sfirst
    using lb2 by auto
  }
  ultimately have ?thesis by linarith
}
ultimately have ?thesis
  using D-is assms by blast
}
ultimately show ?thesis
  using D-is assms by blast
next
case (Release-mltl-ext  $\alpha$  a b L  $\beta$ )
have a-leq-b:  $a \leq b$  and
   $\alpha$ -welldef: intervals-welldef (to-mltl  $\alpha$ ) and
   $\beta$ -welldef: intervals-welldef (to-mltl  $\beta$ )
  using assms unfolding intervals-welldef.simps Release-mltl-ext to-mltl.simps
  by simp-all
have  $\alpha$ -nnf:  $\exists \varphi$ -init.  $\alpha = \text{convert-nnf-ext } \varphi$ -init
  using assms unfolding Release-mltl-ext
  by (metis convert-nnf-ext.simps(9) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(8))

have  $\alpha$ -convert:  $\text{convert-nnf-ext } \alpha = \alpha$ 
  using  $\alpha$ -nnf convert-nnf-ext-convert-nnf-ext by metis
have  $\beta$ -nnf:  $\exists \varphi$ -init.  $\beta = \text{convert-nnf-ext } \varphi$ -init
  using assms unfolding Release-mltl-ext
  by (metis convert-nnf-ext.simps(9) convert-nnf-ext-convert-nnf-ext mltl-ext.inject(8))

have  $\beta$ -convert:  $\text{convert-nnf-ext } \beta = \beta$ 
  using  $\beta$ -nnf convert-nnf-ext-convert-nnf-ext by metis
have  $\alpha$ -composition: is-composition-MLTL  $\alpha$  and
   $\beta$ -composition: is-composition-MLTL  $\beta$  and
  L-composition: is-composition (b-a+1) L
  using Release-mltl-ext assms by simp-all
have  $\alpha$ -wpd:  $b + \text{wpd-mltl (to-mltl } \alpha) \leq \text{length } \pi$  and
   $\beta$ -wpd:  $b + \text{wpd-mltl (to-mltl } \beta) \leq \text{length } \pi$ 
  using assms unfolding Release-mltl-ext to-mltl.simps wpd-mltl.simps
  by auto
let ?s = interval-times a L
have length-L:  $1 \leq \text{length } L$ 
  using composition-length-lb[OF L-composition] a-leq-b by linarith
have sfirst: ?s!0 = a
  using interval-times-first by simp
have slast: ?s!(length L) = b+1
  using interval-times-last[OF a-leq-b L-composition]

```

```

    by blast
  have length-s: length ?s = length L + 1
    using interval-times-length by simp
  let ?D = [α]
  let ?front = {Global-mltl-ext a b L (And-mltl-ext (Notc α) β)}
  let ?middle = {Mighty-Release-mltl-ext α β (?s ! 0) (?s ! 1 - 1)
    [?s ! 1 - ?s ! 0]}
  let ?back = set (map (λi. And-mltl-ext
    (Global-mltl-ext
      (?s ! 0) (?s ! i - 1) [?s!i - ?s!0] (And-mltl-ext (Notc α)
β))
    (Mighty-Release-mltl-ext α β (?s ! i)
      (?s ! (i + 1) - 1) [?s ! (i + 1) - ?s ! i]))
    [1..c α) β)]
      (Mighty-Release-mltl-list ?D β (?s ! i)
        (?s ! (i + 1) - 1) [?s ! (i + 1) - ?s ! i]))
      [1..c α) β)
      by auto
    have global1: (∀ i. a ≤ i ∧ i ≤ b →
      ¬ semantics-mltl (drop i π) (to-mltl α) ∧
      semantics-mltl (drop i π) (to-mltl β))
      using assms(7) unfolding ψ1 semantics-mltl-ext-def to-mltl.simps seman-
tics-mltl.simps
      using α-wpd a-leq-b
      by (metis add-diff-cancel-left' cancel-comm-monoid-add-class.diff-cancel
dual-order.trans le-add1 not-one-le-zero order-antisym-conv wpd-geq-one)
    {
      assume **: ψ2 ∈ ?front
      then have ψ2: ψ2 = Global-mltl-ext a b L (And-mltl-ext (Notc α) β)
        by auto
      have global2: (∀ i. a ≤ i ∧ i ≤ b →
        ¬ semantics-mltl (drop i π) (to-mltl α) ∧

```

```

    semantics-mltl (drop i  $\pi$ ) (to-mltl  $\beta$ )
  using assms(8) unfolding  $\psi 2$  semantics-mltl-ext-def to-mltl.simps seman-
tics-mltl.simps
  using  $\alpha$ -wpd a-leq-b
  by (metis add-diff-cancel-left' cancel-comm-monoid-add-class.diff-cancel
dual-order.trans le-add1 not-one-le-zero order-antisym-conv wpd-geq-one)
  have ?thesis using * ** assms by auto
} moreover {
  assume **:  $\psi 2 \in ?middle$ 
  then have  $\psi 2$ :  $\psi 2 = \text{Mighty-Release-mltl-ext } \alpha \beta$  ( $?s ! 0$ )
    ( $?s ! 1 - 1$ ) [ $?s ! 1 - ?s ! 0$ ]
  by blast
  obtain  $j 2$  where  $j 2$ -bound: ( $?s ! 0 \leq j 2 \wedge j 2 \leq ?s ! 1 - 1$ )
    and  $\alpha$ -semantics2: semantics-mltl (drop  $j 2$   $\pi$ ) (to-mltl  $\alpha$ )
  using assms(8) unfolding  $\psi 2$  Mighty-Release-mltl-ext.simps seman-
tics-mltl-ext-def to-mltl.simps semantics-mltl.simps
  by blast
  have bound1: interval-times a L ! 1 - 1  $\leq$  b
  using interval-times-diff-ge-general[OF a-leq-b L-composition, of length L
1 ?s]
  using slast length-L by force
  then have ?thesis using  $\alpha$ -semantics2 global1  $j 2$ -bound unfolding sfirst
  by simp
} moreover {
  assume **:  $\psi 2 \in ?back$ 
  then obtain  $i 2$  where  $\psi 2$ :  $\psi 2 = \text{And-mltl-ext}$ 
    (Global-mltl-ext
    (interval-times a L ! 0) (interval-times a L !  $i 2 - 1$ ) [ $?s ! i 2 -$ 
? $s ! 0$ ] (And-mltl-ext (Notc  $\alpha$ )  $\beta$ ))
    (Mighty-Release-mltl-ext  $\alpha \beta$  (interval-times a L !  $i 2$ )
    (interval-times a L ! ( $i 2 + 1$ ) - 1)
    [interval-times a L ! ( $i 2 + 1$ ) - interval-times a L !  $i 2$ ])
    and  $i 2$ -bound:  $1 \leq i 2 \wedge i 2 < \text{length } L$ 
  by auto
  obtain  $j 2$  where  $j 2$ -bound: (( $?s ! i 2$ )  $\leq j 2 \wedge j 2 \leq ?s ! (i 2 + 1) - 1$ )
    and  $\alpha$ -semantics2: semantics-mltl (drop  $j 2$   $\pi$ ) (to-mltl  $\alpha$ )
  using assms(8) unfolding  $\psi 2$  Mighty-Release-mltl-ext.simps seman-
tics-mltl-ext-def to-mltl.simps semantics-mltl.simps
  by blast
  have lb2: a  $\leq ?s ! i 2$ 
  using interval-times-diff-ge-general[OF a-leq-b L-composition, of  $i 2$  0 ?s]
  unfolding sfirst using  $i 2$ -bound by simp
  have welldef2:  $?s ! i 2 < ?s ! (i 2 + 1)$ 
  using interval-times-diff-ge[OF a-leq-b L-composition, of  $i 2$  ?s]
  using  $i 2$ -bound by blast
  have ub2: interval-times a L ! ( $i 2 + 1$ ) - 1  $\leq$  b
  using interval-times-diff-ge-general[OF a-leq-b L-composition, of length L
 $i 2 + 1$  ?s]
  using slast  $i 2$ -bound

```

```

    by (metis add commute diff-diff-left diff-is-0-eq le-neq-implies-less less-iff-succ-less-eq
less-or-eq-imp-le)
    have ?thesis using  $\alpha$ -semantics2 global1 j2-bound
    unfolding sfirst using lb2 ub2 by simp
  }
  ultimately have ?thesis using assms D-is by blast
} moreover {
  assume *:  $\psi_1 \in ?middle$ 
  then have  $\psi_1: \psi_1 = \text{Mighty-Release-mltl-ext } \alpha \beta (?s ! 0)$ 
    ( $?s ! 1 - 1$ ) [ $?s ! 1 - ?s ! 0$ ]
    by blast
  obtain j1 where j1-bound: ( $?s ! 0 \leq j1 \wedge j1 \leq ?s ! 1 - 1$ )
    and  $\alpha$ -semantics1: semantics-mltl (drop j1  $\pi$ ) (to-mltl  $\alpha$ )
  using assms(7) unfolding  $\psi_1$  Mighty-Release-mltl-ext.simps semantics-mltl-ext-def
to-mltl.simps semantics-mltl.simps
  by blast
  have bound1: interval-times a L ! 1 - 1  $\leq$  b
  using interval-times-diff-ge-general[OF a-leq-b L-composition, of length L 1
?s]
  using slast length-L by force
  {
    assume **:  $\psi_2 \in ?front$ 
    then have  $\psi_2: \psi_2 = \text{Global-mltl-ext } a b L (\text{And-mltl-ext } (\text{Not}_c \alpha) \beta)$ 
      by auto
    have global2: ( $\forall i. a \leq i \wedge i \leq b \longrightarrow$ 
 $\neg \text{ semantics-mltl } (\text{drop } i \pi) (\text{to-mltl } \alpha) \wedge$ 
 $\text{ semantics-mltl } (\text{drop } i \pi) (\text{to-mltl } \beta))$ 
    using assms(8) unfolding  $\psi_2$  semantics-mltl-ext-def to-mltl.simps seman-
tics-mltl.simps
    using  $\alpha$ -wpd a-leq-b
    by (metis add-diff-cancel-left' cancel-comm-monoid-add-class.diff-cancel
dual-order.trans le-add1 not-one-le-zero order-antisym-conv wpd-geq-one)
    have ?thesis
    using global2  $\alpha$ -semantics1 j1-bound unfolding sfirst using bound1 by
simp
  } moreover {
    assume **:  $\psi_2 \in ?middle$ 
    then have  $\psi_2: \psi_2 = \text{Mighty-Release-mltl-ext } \alpha \beta (?s ! 0)$ 
      ( $?s ! 1 - 1$ ) [ $?s ! 1 - ?s ! 0$ ]
      by blast
    then have ?thesis using  $\psi_1$  assms by blast
  } moreover {
    assume **:  $\psi_2 \in ?back$ 
    then obtain i2 where  $\psi_2: \psi_2 = \text{And-mltl-ext}$ 
      (Global-mltl-ext
      (interval-times a L ! 0) (interval-times a L ! i2 - 1) [ $?s!i2 -$ 
?s!0] ( $\text{And-mltl-ext } (\text{Not}_c \alpha) \beta$ ))
      (Mighty-Release-mltl-ext  $\alpha \beta$  (interval-times a L ! i2)
      (interval-times a L ! (i2 + 1) - 1))
  }

```

$[interval-times\ a\ L!\ (i2 + 1) - interval-times\ a\ L!\ i2]$
and $i2-bound: 1 \leq i2 \wedge i2 < length\ L$

by auto

obtain $j2$ **where** $j2-bound: ((?s!\ i2) \leq j2 \wedge j2 \leq ?s!\ (i2 + 1) - 1)$
and $\alpha-semantics2: semantics-mltl\ (drop\ j2\ \pi)\ (to-mltl\ \alpha)$
and $global-before2: \forall i. interval-times\ a\ L!\ 0 \leq i \wedge i \leq interval-times$
 $a\ L!\ i2 - 1 \longrightarrow$

$\neg semantics-mltl\ (drop\ i\ \pi)\ (to-mltl\ \alpha) \wedge$
 $semantics-mltl\ (drop\ i\ \pi)\ (to-mltl\ \beta)$
using $assms(8)$ **unfolding** $\psi2$ *Mighty-Release-mltl-ext.simps semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps*
unfolding $sfirst$ **using** $\alpha-wpd$ **by auto**
have $lb2: a < ?s!\ i2$
using $interval-times-diff-ge-general[OF\ a-leq-b\ L-composition, of\ i2\ 0\ ?s]$
unfolding $sfirst$ **using** $i2-bound$ **by simp**
have $welldef2: ?s!\ i2 < ?s!\ (i2+1)$
using $interval-times-diff-ge[OF\ a-leq-b\ L-composition, of\ i2\ ?s]$
using $i2-bound$ **by blast**
have $ub2: interval-times\ a\ L!\ (i2 + 1) - 1 \leq b$
using $interval-times-diff-ge-general[OF\ a-leq-b\ L-composition, of\ length\ L$
 $i2+1\ ?s]$
using $slast\ i2-bound$
by (*metis add.commute diff-diff-left diff-is-0-eq le-neq-implies-less less-iff-succ-less-eq less-or-eq-imp-le*)
have $bound1: interval-times\ a\ L!\ 1 \leq interval-times\ a\ L!\ i2$
using $interval-times-diff-ge-general[OF\ a-leq-b\ L-composition, of\ i2\ 1\ ?s]$
using $i2-bound$ **by force**
have $?thesis$ **using** $global-before2\ \alpha-semantics1\ bound1$
using $j1-bound$ **unfolding** $sfirst$ **by auto**

}

ultimately have $?thesis$ **using** $assms\ D-is$ **by blast**

} **moreover** **{**

assume $*$: $\psi1 \in ?back$

then obtain $i1$ **where** $\psi1: \psi1 = And-mltl-ext$
(Global-mltl-ext
 $(interval-times\ a\ L!\ 0)\ (interval-times\ a\ L!\ i1 - 1)\ [?s!\ i1 -$
 $?s!\ 0]\ (And-mltl-ext\ (Not_c\ \alpha)\ \beta))$
 $(Mighty-Release-mltl-ext\ \alpha\ \beta\ (interval-times\ a\ L!\ i1)$
 $(interval-times\ a\ L!\ (i1 + 1) - 1)$
 $[interval-times\ a\ L!\ (i1 + 1) - interval-times\ a\ L!\ i1])$
and $i1-bound: 1 \leq i1 \wedge i1 < length\ L$

by auto

obtain $j1$ **where** $j1-bound: ((?s!\ i1) \leq j1 \wedge j1 \leq ?s!\ (i1 + 1) - 1)$
and $\alpha-semantics1: semantics-mltl\ (drop\ j1\ \pi)\ (to-mltl\ \alpha)$
and $global-before1: \forall i. interval-times\ a\ L!\ 0 \leq i \wedge i \leq interval-times$
 $a\ L!\ i1 - 1 \longrightarrow$

$\neg semantics-mltl\ (drop\ i\ \pi)\ (to-mltl\ \alpha) \wedge$
 $semantics-mltl\ (drop\ i\ \pi)\ (to-mltl\ \beta)$
using $assms(7)$ **unfolding** $\psi1$ *Mighty-Release-mltl-ext.simps semantics-mltl-ext-def*

```

to-mltl.simps semantics-mltl.simps
  unfolding sfirst using  $\alpha$ -wpd by auto
  have lb1:  $a \leq ?s!i1$ 
    using interval-times-diff-ge-general[OF a-leq-b L-composition, of i1 0 ?s]
    unfolding sfirst using i1-bound by simp
  have welldef1:  $?s!i1 < ?s!(i1+1)$ 
    using interval-times-diff-ge[OF a-leq-b L-composition, of i1 ?s]
    using i1-bound by blast
  have ub1:  $interval\text{-times } a \ L \ ! \ (i1 + 1) - 1 \leq b$ 
    using interval-times-diff-ge-general[OF a-leq-b L-composition, of length L
i1+1 ?s]
    using slast i1-bound
  by (metis add.commute diff-diff-left diff-is-0-eq le-neq-implies-less less-iff-succ-less-eq
less-or-eq-imp-le)
  have bound1:  $interval\text{-times } a \ L \ ! \ 1 \leq interval\text{-times } a \ L \ ! \ i1$ 
    using interval-times-diff-ge-general[OF a-leq-b L-composition, of i1 1 ?s]
    using i1-bound by force
  {
  assume *:  $\psi2 \in ?front$ 
  then have  $\psi2: \psi2 = Global\text{-mltl-ext } a \ b \ L \ (And\text{-mltl-ext } (Not_c \ \alpha) \ \beta)$ 
    by auto
  have global2:  $(\forall i. a \leq i \wedge i \leq b \longrightarrow$ 
 $\neg semantics\text{-mltl } (drop \ i \ \pi) \ (to\text{-mltl } \ \alpha) \wedge$ 
 $semantics\text{-mltl } (drop \ i \ \pi) \ (to\text{-mltl } \ \beta))$ 
    using assms(8) unfolding  $\psi2$  semantics-mltl-ext-def to-mltl.simps seman-
tics-mltl.simps
    using  $\alpha$ -wpd a-leq-b
    by (metis add-diff-cancel-left' cancel-comm-monoid-add-class.diff-cancel
dual-order.trans le-add1 not-one-le-zero order-antisym-conv wpd-geq-one)
  have ?thesis using  $\alpha$ -semantics1 global2 j1-bound
    unfolding sfirst using lb1 ub1 by simp
  } moreover {
  assume *:  $\psi2 \in ?middle$ 
  then have  $\psi2: \psi2 = Mighty\text{-Release-mltl-ext } \alpha \ \beta \ ( ?s \ ! \ 0)$ 
 $( ?s \ ! \ 1 - 1) \ [ ?s \ ! \ 1 - ?s \ ! \ 0]$ 
    by blast
  obtain j2 where j2-bound:  $( ?s \ ! \ 0 \leq j2 \wedge j2 \leq ?s \ ! \ 1 - 1)$ 
    and  $\alpha$ -semantics2:  $semantics\text{-mltl } (drop \ j2 \ \pi) \ (to\text{-mltl } \ \alpha)$ 
    using assms(8) unfolding  $\psi2$  Mighty-Release-mltl-ext.simps seman-
tics-mltl-ext-def to-mltl.simps semantics-mltl.simps
    by blast
  have bound1:  $interval\text{-times } a \ L \ ! \ 1 \leq interval\text{-times } a \ L \ ! \ i1$ 
    using interval-times-diff-ge-general[OF a-leq-b L-composition, of i1 1 ?s]
    using i1-bound by force
  then have ?thesis
    using  $\alpha$ -semantics2 global-before1
    using j2-bound unfolding sfirst by auto
  } moreover {
  assume *:  $\psi2 \in ?back$ 

```

```

then obtain  $i2$  where  $\psi2$ :  $\psi2 = \text{And-mltl-ext}$ 
  ( $\text{Global-mltl-ext}$ 
    ( $\text{interval-times } a \ L \ ! \ 0$ ) ( $\text{interval-times } a \ L \ ! \ i2 - 1$ ) [ $?s!i2 -$ 
 $?s!0$ ] ( $\text{And-mltl-ext}$  ( $\text{Not}_c \ \alpha$ )  $\beta$ ))
  ( $\text{Mighty-Release-mltl-ext } \alpha \ \beta$  ( $\text{interval-times } a \ L \ ! \ i2$ )
    ( $\text{interval-times } a \ L \ ! \ (i2 + 1) - 1$ )
    [ $\text{interval-times } a \ L \ ! \ (i2 + 1) - \text{interval-times } a \ L \ ! \ i2$ ])
    and  $i2\text{-bound}$ :  $1 \leq i2 \wedge i2 < \text{length } L$ 

  by auto
obtain  $j2$  where  $j2\text{-bound}$ :  $((?s \ ! \ i2) \leq j2 \wedge j2 \leq ?s \ ! \ (i2 + 1) - 1)$ 
  and  $\alpha\text{-semantics2}$ :  $\text{semantics-mltl } (\text{drop } j2 \ \pi) \ (\text{to-mltl } \alpha)$ 
  and  $\text{global-before2}$ :  $\forall i. \text{interval-times } a \ L \ ! \ 0 \leq i \wedge i \leq \text{interval-times}$ 
 $a \ L \ ! \ i2 - 1 \longrightarrow$ 
 $\neg \text{semantics-mltl } (\text{drop } i \ \pi) \ (\text{to-mltl } \alpha) \wedge$ 
 $\text{semantics-mltl } (\text{drop } i \ \pi) \ (\text{to-mltl } \beta)$ 
  using  $\text{assms}(8)$  unfolding  $\psi2$   $\text{Mighty-Release-mltl-ext.simps semantics-mltl-ext-def to-mltl.simps semantics-mltl.simps}$ 
  unfolding  $\text{sfirst}$  using  $\alpha\text{-wpd}$  by auto
  have  $lb2$ :  $a \leq ?s!i2$ 
  using  $\text{interval-times-diff-ge-general}[OF \ a\text{-leq-b } L\text{-composition, of } i2 \ 0 \ ?s]$ 
  unfolding  $\text{sfirst}$  using  $i2\text{-bound}$  by simp
  have  $\text{welldef2}$ :  $?s!i2 < ?s!(i2+1)$ 
  using  $\text{interval-times-diff-ge}[OF \ a\text{-leq-b } L\text{-composition, of } i2 \ ?s]$ 
  using  $i2\text{-bound}$  by blast
  have  $ub2$ :  $\text{interval-times } a \ L \ ! \ (i2 + 1) - 1 \leq b$ 
  using  $\text{interval-times-diff-ge-general}[OF \ a\text{-leq-b } L\text{-composition, of length } L$ 
 $i2+1 \ ?s]$ 
  using  $\text{slast } i2\text{-bound}$ 
by ( $\text{metis add.commute diff-diff-left diff-is-0-eq le-neq-implies-less less-iff-succ-less-eq}$ 
 $\text{less-or-eq-imp-le}$ )
  {
    assume  $\text{eq}$ :  $i1 = i2$ 
    then have  $?thesis$ 
    using  $\text{assms}(6)$   $\psi1 \ \psi2$  by blast
  } moreover {
    assume  $\text{le}$ :  $i1 < i2$ 
    then have  $\text{interval-times } a \ L \ ! \ (i1 + 1) \leq \text{interval-times } a \ L \ ! \ (i2)$ 
    using  $\text{interval-times-diff-ge-general}[OF \ a\text{-leq-b } L\text{-composition, of } i2 \ i1+1$ 
 $?s]$ 
    using  $i1\text{-bound } i2\text{-bound}$ 
    by ( $\text{metis le-eq-less-or-eq less-iff-succ-less-eq}$ )
    then have  $?thesis$ 
    using  $\alpha\text{-semantics1 global-before2 } j1\text{-bound}$ 
    using  $lb1$  unfolding  $\text{sfirst}$  by auto
  } moreover {
    assume  $\text{ge}$ :  $i1 > i2$ 
    then have  $\text{interval-times } a \ L \ ! \ (i2 + 1) \leq \text{interval-times } a \ L \ ! \ (i1)$ 
    using  $\text{interval-times-diff-ge-general}[OF \ a\text{-leq-b } L\text{-composition, of } i1 \ i2+1$ 
 $?s]$ 

```



```

    using i1-bound i2-bound
    by (metis le-eq-less-or-eq less-iff-succ-less-eq)
  then have ?thesis
    using  $\alpha$ -semantics2 global-before1 j2-bound
    using lb2 unfolding sfirst by auto
  }
  ultimately have ?thesis by linarith
}
ultimately have ?thesis using assms D-is by blast
}
ultimately show ?thesis using assms D-is by blast
qed

```

lemma *LP-mltl-language-disjoint-aux-k1*:

```

fixes  $\varphi::'a$  mltl-ext and  $\psi1 \ \psi2::'a$  mltl-ext and  $k::nat$ 
assumes intervals-welldef: intervals-welldef (to-mltl  $\varphi$ )
assumes is-nnf:  $\exists \varphi$ -init.  $\varphi = \text{convert-nnf-ext } \varphi$ -init
assumes composition: is-composition-MLTL  $\varphi$ 
assumes D-decomp:  $D = \text{set } (LP\text{-mltl-aux } \varphi \ 1)$ 
assumes diff-formulas:  $(\psi1 \in D) \wedge (\psi2 \in D) \wedge \psi1 \neq \psi2$ 
assumes r-wpd:  $r \geq \text{wpd-mltl } (to\text{-mltl } \varphi)$ 
shows (language-mltl-r (to-mltl  $\psi1$ )  $r$ )
       $\cap$  (language-mltl-r (to-mltl  $\psi2$ )  $r$ ) = {}
proof-
{
  assume contra: (language-mltl-r (to-mltl  $\psi1$ )  $r$ )
     $\cap$  (language-mltl-r (to-mltl  $\psi2$ )  $r$ )  $\neq \{\}$ 
  then have  $\exists \pi. \pi \in (\text{language-mltl-r } (to\text{-mltl } \psi1) \ r) \wedge$ 
     $\pi \in (\text{language-mltl-r } (to\text{-mltl } \psi2) \ r)$ 
    by auto
  then obtain  $\pi$  where in1:  $\pi \in (\text{language-mltl-r } (to\text{-mltl } \psi1) \ r)$ 
    and in2:  $\pi \in (\text{language-mltl-r } (to\text{-mltl } \psi2) \ r)$ 
    by blast
  have sem1: semantics-mltl-ext  $\pi \ \psi1$  and
    sem2: semantics-mltl-ext  $\pi \ \psi2$  and
    len: length  $\pi \geq \text{wpd-mltl } (to\text{-mltl } \varphi)$ 
    using in1 in2 assms(6)
    unfolding language-mltl-r-def semantics-mltl-ext-def
    by simp-all
  have False
    by (metis D-decomp LP-mltl-language-disjoint-aux-helper-k1 One-nat-def com-
      position diff-formulas intervals-welldef is-nnf len sem1 sem2)
}
then show ?thesis by blast
qed

```

theorem *LP-mltl-language-disjoint-k1*:

```

fixes  $\varphi::'a$  mttl-ext and  $\psi1$   $\psi2::'a$  mttl and  $k::nat$ 
assumes intervals-welldef: intervals-welldef (to-mttl  $\varphi$ )
assumes composition: is-composition-MLTL  $\varphi$ 
assumes D-decomp:  $D = set$  (LP-mttl  $\varphi$  1)
assumes diff-formulas:  $(\psi1 \in D) \wedge (\psi2 \in D) \wedge \psi1 \neq \psi2$ 
assumes r-wpd:  $r \geq wpd\text{-mttl}$  (to-mttl  $\varphi$ )
shows (language-mttl-r  $\psi1$   $r$ )  $\cap$  (language-mttl-r  $\psi2$   $r$ ) = {}
proof –
let ?D = LP-mttl-aux (convert-nnf-ext  $\varphi$ ) 1
let ? $\varphi$  = convert-nnf-ext  $\varphi$ 
have cond1: intervals-welldef (to-mttl (convert-nnf-ext  $\varphi$ ))
  using intervals-welldef
  by (metis convert-nnf-ext-to-mttl-commute nnf-intervals-welldef)
have cond2:  $\exists \varphi\text{-init.}$  convert-nnf-ext  $\varphi = \text{convert-nnf-ext } \varphi\text{-init}$ 
  by blast
have cond3: is-composition-MLTL (convert-nnf-ext  $\varphi$ )
  using composition
  by (simp add: intervals-welldef is-composition-convert-nnf-ext)
have cond4:  $set$  (LP-mttl-aux (convert-nnf-ext  $\varphi$ ) 1) =
   $set$  (LP-mttl-aux (convert-nnf-ext  $\varphi$ ) 1)
  by blast
obtain  $\psi1'$   $\psi2'$  where  $\psi1$ :  $\psi1 = to\text{-mttl}$  (convert-nnf-ext  $\psi1'$ )
  and  $\psi1'\text{-in}$ :  $\psi1' \in set$  ?D
  and  $\psi2$ :  $\psi2 = to\text{-mttl}$  (convert-nnf-ext  $\psi2'$ )
  and  $\psi2'\text{-in}$ :  $\psi2' \in set$  ?D
  using D-decomp unfolding LP-mttl.simps
  using diff-formulas by auto
have  $\psi's\text{-neq}$ :  $\psi1' \neq \psi2'$ 
  using diff-formulas  $\psi1$   $\psi2$  by blast
have  $\psi1\text{-welldef}$ : intervals-welldef  $\psi1$ 
  using assms(4) D-decomp unfolding LP-mttl.simps
  using LP-mttl-aux-intervals-welldef
  by (metis  $\psi1$   $\psi1'\text{-in}$  composition convert-nnf-ext-to-mttl-commute intervals-welldef
nnf-intervals-welldef)
  then have  $\psi1'\text{-welldef}$ : intervals-welldef (to-mttl  $\psi1'$ )
    using  $\psi1$ 
    using LP-mttl-aux-intervals-welldef  $\psi1'\text{-in}$  allones-implies-is-composition-MLTL
composition intervals-welldef by auto
  have  $\psi2\text{-welldef}$ : intervals-welldef  $\psi2$ 
    using assms(4) D-decomp unfolding LP-mttl.simps
    using LP-mttl-aux-intervals-welldef
    by (metis  $\psi2$   $\psi2'\text{-in}$  composition convert-nnf-ext-to-mttl-commute intervals-welldef
nnf-intervals-welldef)
  then have  $\psi2'\text{-welldef}$ : intervals-welldef (to-mttl  $\psi2'$ )
    using  $\psi2$ 
    using LP-mttl-aux-intervals-welldef  $\psi2'\text{-in}$  allones-implies-is-composition-MLTL
composition intervals-welldef by auto
  have intersect: language-mttl-r (to-mttl  $\psi1'$ )  $r$   $\cap$ 
    language-mttl-r (to-mttl  $\psi2'$ )  $r = \{\}$ 

```

```

    using LP-mltl-language-disjoint-aux-k1[OF cond1 cond2 cond3 cond4, of  $\psi 1'$ 
 $\psi 2' r$ ]
    using  $\psi 1'$ -in  $\psi 2'$ -in  $\psi'$ 's-neq r-wpd
    by (metis convert-nnf-ext-preserves-wpd)
    have semantics-mltl  $\pi$  (to-mltl (convert-nnf-ext  $\varphi$ )) =
      semantics-mltl  $\pi$  (to-mltl  $\varphi$ )
    if intervals-welldef (to-mltl  $\varphi$ )
    for  $\varphi::'a$  mltl-ext and  $\pi$ 
    using that unfolding semantic-equiv-ext-def
    by (metis convert-nnf-ext-to-mltl-commute convert-nnf-preserves-semantics)
    then show ?thesis using intersect
    unfolding language-mltl-r-def  $\psi 1$   $\psi 2$ 
    using  $\psi 1'$ -welldef  $\psi 2'$ -welldef
    by auto
qed

end
theory MTLT-Language-Partition-Codegen

imports MTLT-Language-Partition-Algorithm Show.Shows-Literal

begin

```

9 Pretty Parsing

```

fun nat-to-string:: nat  $\Rightarrow$  string where
  nat-to-string n = String.explode (Shows-Literal.showl n)

fun mltl-to-literal-aux:: nat mltl  $\Rightarrow$  string where
  mltl-to-literal-aux Truem = "true"
| mltl-to-literal-aux Falsem = "false"
| mltl-to-literal-aux (Propm (p)) = "p"@(nat-to-string p)
| mltl-to-literal-aux (Notm  $\varphi$ ) = "!"@(mltl-to-literal-aux  $\varphi$ )@"'"
| mltl-to-literal-aux ( $\varphi$  Andm  $\psi$ ) = "(" @ (mltl-to-literal-aux  $\varphi$ ) @ "&" @ (mltl-to-literal-aux
 $\psi$ ) @ ")"
| mltl-to-literal-aux ( $\varphi$  Orm  $\psi$ ) = "(" @ (mltl-to-literal-aux  $\varphi$ ) @ "|" @ (mltl-to-literal-aux
 $\psi$ ) @ ")"
| mltl-to-literal-aux (Gm [a,b]  $\varphi$ ) = "(G[" @ (nat-to-string a) @ "," @ (nat-to-string
b) @ "]" @ (mltl-to-literal-aux  $\varphi$ ) @ ")"
| mltl-to-literal-aux (Fm [a,b]  $\varphi$ ) = "(F[" @ (nat-to-string a) @ "," @ (nat-to-string
b) @ "]" @ (mltl-to-literal-aux  $\varphi$ ) @ ")"
| mltl-to-literal-aux ( $\varphi$  Rm [a,b]  $\psi$ ) = "(" @ (mltl-to-literal-aux  $\varphi$ ) @ " R[" @
(nat-to-string a) @ "," @ (nat-to-string b) @ "]" @ (mltl-to-literal-aux  $\psi$ ) @ ")"
| mltl-to-literal-aux ( $\varphi$  Um [a,b]  $\psi$ ) = "(" @ (mltl-to-literal-aux  $\varphi$ ) @ " U[" @
(nat-to-string a) @ "," @ (nat-to-string b) @ "]" @ (mltl-to-literal-aux  $\psi$ ) @ ")"

fun mltl-to-literal:: nat mltl  $\Rightarrow$  String.literal
  where mltl-to-literal  $\varphi$  = String.implode (mltl-to-literal-aux  $\varphi$ )

```

```
value mttl-to-literal ((Propm (?) Andm Truem) Um[?,?] Falsem) =  
  STR "((p? & true) U[?,?] false)"
```

10 Code Export

```
export-code LP-mttl mttl-to-literal in Haskell module-name LP-mttl  
end
```

References

- [1] K. Kosaian, Z. Wang, and E. Sloan. Mission-time linear temporal logic. *Archive of Formal Proofs*, January 2025. https://isa-afp.org/entries/Mission_Time_LTL.html, Formal proof development.