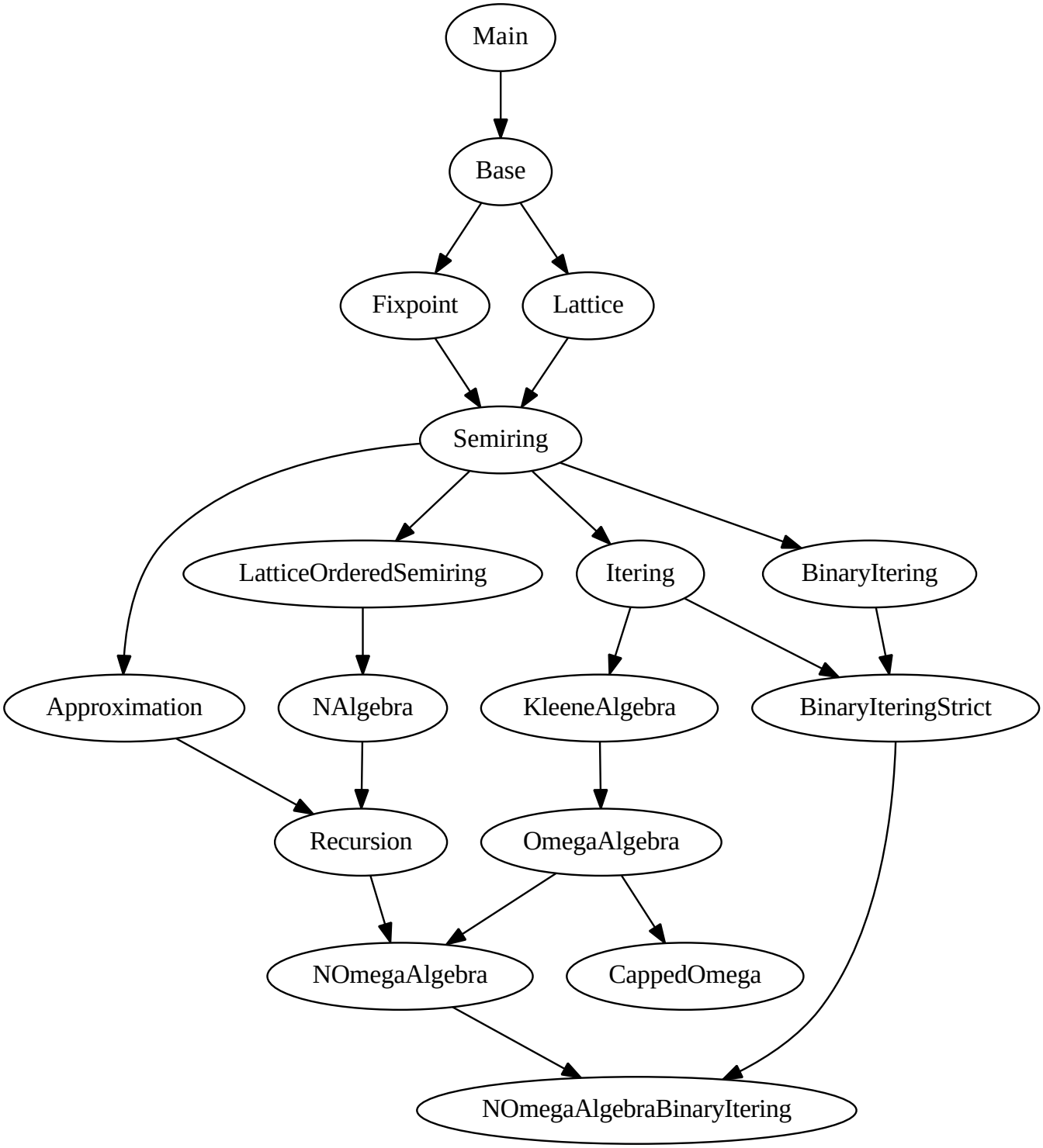
Isabelle/HOL Theories of An Algebraic Approach to Computations with Progress

Walter Guttmann

Department of Computer Science and Software Engineering
University of Canterbury
2015

This document is a reference for the paper *An Algebraic Approach to Computations with Progress*. Proofs of some results in this paper can be found in Sections 12–16. A comment of the form '— AACP Theorem n' precedes results in the theory files which contribute to Theorem n of the paper. The theories have been developed with Isabelle2014. The dependences of the theory files are shown on the following page.

Isabelle/HOL Theories


Main
Base
Fixpoint
Lattice
Semiring
LatticeOrderedSemiring
Itering
BinaryItering
Approximation
NAlgebra
KleeneAlgebra
BinaryIteringStrict
Recursion
OmegaAlgebra
NOmegaAlgebra
CappedOmega
NOmegaAlgebraBinaryItering

1 Base

theory *Base*

imports *Main*

begin

— This theory and the subsequent theories have been developed with Isabelle2014.

nitpick-params [*timeout* = *600*]

declare [[*smt-timeout* = *600*]]

class *mult* = *times*

begin

notation
times (**infixl** · *70*) **and**
times (**infixl** ; *70*)

end

class *neg* = *uminus*

begin

no-notation
uminus (− - [*81*] *80*)

notation
uminus (− - [*80*] *80*)

end

class *while* =
fixes *while* :: *'a* ⇒ *'a* ⇒ *'a* (**infixr** ⋆ *59*)

class *L* =
fixes *L* :: *'a*

class *n* =
fixes *n* :: *'a* ⇒ *'a*

class *d* =
fixes *d* :: *'a* ⇒ *'a*

class *diamond* =
fixes *diamond* :: *'a* ⇒ *'a* ⇒ *'a* (| - > - [*50*,*90*] *95*)

class *box* =
fixes *box* :: *'a* ⇒ *'a* ⇒ *'a* (| -] - [*50*,*90*] *95*)

context *ord*

begin

definition *isotone* :: (*'a* ⇒ *'a*) ⇒ *bool*
where *isotone f* ⟷ (∀ *x y* . *x* ≤ *y* ⟶ *f*(*x*) ≤ *f*(*y*))

definition *lifted-less-eq* :: (*'a* ⇒ *'a*) ⇒ (*'a* ⇒ *'a*) ⇒ *bool* ((- ≤≤ -) [*51*, *51*] *50*)
where *f* ≤≤ *g* ⟷ (∀ *x* . *f*(*x*) ≤ *g*(*x*))

definition *galois* :: (*'a* ⇒ *'a*) ⇒ (*'a* ⇒ *'a*) ⇒ *bool*
where *galois l u* ⟷ (∀ *x y* . *l*(*x*) ≤ *y* ⟷ *x* ≤ *u*(*y*))

definition *ascending-chain* :: (*nat* ⇒ *'a*) ⇒ *bool*
where *ascending-chain f* ⟷ (∀ *n* . *f n* ≤ *f* (*Suc n*))

definition *descending-chain* :: $(nat \Rightarrow 'a) \Rightarrow bool$
where $descending\text{-}chain\ f \longleftrightarrow (\forall n\ .\ f\ (Suc\ n) \leq f\ n)$

definition *directed* :: $'a\ set \Rightarrow bool$
where $directed\ X \longleftrightarrow X \neq \{\} \wedge (\forall x{\in}X\ .\ \forall y{\in}X\ .\ \exists z{\in}X\ .\ x \leq z \wedge y \leq z)$

definition *codirected* :: $'a\ set \Rightarrow bool$
where $codirected\ X \longleftrightarrow X \neq \{\} \wedge (\forall x{\in}X\ .\ \forall y{\in}X\ .\ \exists z{\in}X\ .\ z \leq x \wedge z \leq y)$

definition *chain* :: $'a\ set \Rightarrow bool$
where $chain\ X \longleftrightarrow (\forall x{\in}X\ .\ \forall y{\in}X\ .\ x \leq y \vee y \leq x)$

end

context *order*

begin

lemma *lifted-reflexive*: $f = g \longrightarrow f \leq\leq g$
by (*metis lifted-less-eq-def order-refl*)

lemma *lifted-transitive*: $f \leq\leq g \wedge g \leq\leq h \longrightarrow f \leq\leq h$
by (*smt lifted-less-eq-def order-trans*)

lemma *lifted-antisymmetric*: $f \leq\leq g \wedge g \leq\leq f \longrightarrow f = g$
by (*metis* (*full-types*) *antisym ext lifted-less-eq-def*)

lemma *galois-char*: $galois\ l\ u \longleftrightarrow (\forall x\ .\ x \leq u(l(x))) \wedge (\forall x\ .\ l(u(x)) \leq x) \wedge isotone\ l \wedge isotone\ u$
apply (*rule iffI*)
apply (*metis* (*full-types*) *galois-def isotone-def order-refl order-trans*)
apply (*metis galois-def isotone-def order-trans*)
done

lemma *galois-closure*: $galois\ l\ u \longrightarrow l(x) = l(u(l(x))) \wedge u(x) = u(l(u(x)))$
by (*smt antisym galois-char isotone-def*)

lemma *ascending-chain-k*: $ascending\text{-}chain\ f \longrightarrow f\ m \leq f\ (m + k)$
apply (*induct k*)
apply *simp*
apply (*metis add-Suc-right ascending-chain-def order-trans*)
done

lemma *ascending-chain-isotone*: $ascending\text{-}chain\ f \wedge m \leq k \longrightarrow f\ m \leq f\ k$
by (*metis ascending-chain-k le-iff-add*)

lemma *ascending-chain-comparable*: $ascending\text{-}chain\ f \longrightarrow f\ k \leq f\ m \vee f\ m \leq f\ k$
by (*metis nat-le-linear ascending-chain-isotone*)

lemma *ascending-chain-chain*: $ascending\text{-}chain\ f \longrightarrow chain\ (range\ f)$
by (*smt ascending-chain-comparable chain-def image-iff*)

lemma *chain-directed*: $X \neq \{\} \wedge chain\ X \longrightarrow directed\ X$
by (*metis chain-def directed-def*)

lemma *ascending-chain-directed*: $ascending\text{-}chain\ f \longrightarrow directed\ (range\ f)$
by (*metis UNIV-not-empty ascending-chain-chain chain-directed empty-is-image*)

lemma *descending-chain-k*: $descending\text{-}chain\ f \longrightarrow f\ (m + k) \leq f\ m$
apply (*induct k*)
apply *simp*
apply (*metis add-Suc-right descending-chain-def order-trans*)
done

lemma *descending-chain-antitone*: $descending\text{-}chain\ f \wedge m \leq k \longrightarrow f\ k \leq f\ m$
by (*metis descending-chain-k le-iff-add*)

lemma *descending-chain-comparable*: $descending\text{-}chain\ f \longrightarrow f\ k \leq f\ m \vee f\ m \leq f\ k$
by (*metis nat-le-linear descending-chain-antitone*)

lemma *descending-chain-chain*: *descending-chain f* ⟶ *chain* (*range f*)
unfolding *chain-def*
apply *simp*
apply (*smt descending-chain-comparable*)
done

lemma *chain-codirected*: *X* ≠ {} ∧ *chain X* ⟶ *codirected X*
by (*metis chain-def codirected-def*)

lemma *descending-chain-codirected*: *descending-chain f* ⟶ *codirected* (*range f*)
by (*metis UNIV-not-empty descending-chain-chain chain-codirected empty-is-image*)

end

context *complete-lattice*

begin

lemma *sup-Sup*: **assumes** *nonempty*: *A* ≠ {}
shows *sup x* (*Sup A*) = *Sup* ((*sup x*) ‘ *A*)
apply (*rule antisym*)
apply (*metis Sup-mono Sup-upper2 assms ex-in-conv imageI le-supI sup-ge1 sup-ge2*)
apply (*smt Sup-least Sup-upper image-iff le-iff-sup sup.commute sup-ge1 sup-left-commute*)
done

lemma *sup-SUP*: *Y* ≠ {} ⟶ *sup x* (*SUP y*:*Y* . *f y*) = (*SUP y*:*Y*. *sup x* (*f y*))
unfolding *SUP-def*
apply *rule*
apply (*subst sup-Sup*)
apply (*smt empty-is-image*)
apply (*metis image-image*)
done

lemma *inf-Inf*: **assumes** *nonempty*: *A* ≠ {}
shows *inf x* (*Inf A*) = *Inf* ((*inf x*) ‘ *A*)
apply (*rule antisym*)
apply (*smt Inf-greatest Inf-lower image-iff le-iff-inf inf.commute inf-le1 inf-left-commute*)
apply (*metis Inf-mono Inf-lower2 assms ex-in-conv imageI le-infI inf-le1 inf-le2*)
done

lemma *inf-INF*: *Y* ≠ {} ⟶ *inf x* (*INF y*:*Y* . *f y*) = (*INF y*:*Y*. *inf x* (*f y*))
unfolding *INF-def*
apply *rule*
apply (*subst inf-Inf*)
apply (*smt empty-is-image*)
apply (*metis image-image*)
done

lemma *SUP-image-id*[*simp*]: (*SUP x*:*f*‘*A* . *x*) = (*SUP x*:*A* . *f x*)
by *simp*

lemma *INF-image-id*[*simp*]: (*INF x*:*f*‘*A* . *x*) = (*INF x*:*A* . *f x*)
by *simp*

end

lemma *image-Collect-2*: *f* ‘ { *g x* | *x* . *P x* } = { *f* (*g x*) | *x* . *P x* }
by *auto*

— The following instantiation and four lemmas are from Jose Divason Mallagaray.

instantiation *fun* :: (*type*, *type*) *power*

begin

definition *one-fun* :: *'a* ⇒ *'a*
where *one-fun-def*: *one-fun* ≡ *id*

definition *times-fun* :: (*'a* ⇒ *'a*) ⇒ (*'a* ⇒ *'a*) ⇒ (*'a* ⇒ *'a*)

where *times-fun-def*: *times-fun* ≡ *comp*

instance
by *intro-classes*

end

lemma *id-power*: *id^m* = *id*
apply (*induct m*)
apply (*metis one-fun-def power-0*)
apply (*simp add*: *times-fun-def*)
done

lemma *power-zero-id*: *f^0* = *id*
by (*metis one-fun-def power-0*)

lemma *power-succ-unfold*: *f^Suc m* = *f* ∘ *f^m*
by (*metis power-Suc times-fun-def*)

lemma *power-succ-unfold-ext*: (*f^Suc m*) *x* = *f* ((*f^m*) *x*)
by (*metis o-apply power-succ-unfold*)

end

2 Fixpoint

theory *Fixpoint*

imports *Base*

begin

context *order*

begin

definition *is-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a \Rightarrow bool$ **where** *is-fixpoint* $f\ x \longleftrightarrow f(x) = x$
definition *is-prefixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a \Rightarrow bool$ **where** *is-prefixpoint* $f\ x \longleftrightarrow f(x) \leq x$
definition *is-postfixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a \Rightarrow bool$ **where** *is-postfixpoint* $f\ x \longleftrightarrow f(x) \geq x$
definition *is-least-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a \Rightarrow bool$ **where** *is-least-fixpoint* $f\ x \longleftrightarrow f(x) = x \wedge (\forall y\ .\ f(y) = y \longrightarrow x \leq y)$
definition *is-greatest-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a \Rightarrow bool$ **where** *is-greatest-fixpoint* $f\ x \longleftrightarrow f(x) = x \wedge (\forall y\ .\ f(y) = y \longrightarrow x \geq y)$
definition *is-least-prefixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a \Rightarrow bool$ **where** *is-least-prefixpoint* $f\ x \longleftrightarrow f(x) \leq x \wedge (\forall y\ .\ f(y) \leq y \longrightarrow x \leq y)$
definition *is-greatest-postfixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a \Rightarrow bool$ **where** *is-greatest-postfixpoint* $f\ x \longleftrightarrow f(x) \geq x \wedge (\forall y\ .\ f(y) \geq y \longrightarrow x \geq y)$
definition *has-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow bool$ **where** *has-fixpoint* $f \longleftrightarrow (\exists x\ .\ is\text{-}fixpoint\ f\ x)$
definition *has-prefixpoint* :: $('a \Rightarrow 'a) \Rightarrow bool$ **where** *has-prefixpoint* $f \longleftrightarrow (\exists x\ .\ is\text{-}prefixpoint\ f\ x)$
definition *has-postfixpoint* :: $('a \Rightarrow 'a) \Rightarrow bool$ **where** *has-postfixpoint* $f \longleftrightarrow (\exists x\ .\ is\text{-}postfixpoint\ f\ x)$
definition *has-least-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow bool$ **where** *has-least-fixpoint* $f \longleftrightarrow (\exists x\ .\ is\text{-}least\text{-}fixpoint\ f\ x)$
definition *has-greatest-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow bool$ **where** *has-greatest-fixpoint* $f \longleftrightarrow (\exists x\ .\ is\text{-}greatest\text{-}fixpoint\ f\ x)$
definition *has-least-prefixpoint* :: $('a \Rightarrow 'a) \Rightarrow bool$ **where** *has-least-prefixpoint* $f \longleftrightarrow (\exists x\ .\ is\text{-}least\text{-}prefixpoint\ f\ x)$
definition *has-greatest-postfixpoint* :: $('a \Rightarrow 'a) \Rightarrow bool$ **where** *has-greatest-postfixpoint* $f \longleftrightarrow (\exists x\ .\ is\text{-}greatest\text{-}postfixpoint\ f\ x)$
definition *the-least-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a\ (\mu\ -\ [201]\ 200)$ **where** $\mu\ \ f = (THE\ x\ .\ is\text{-}least\text{-}fixpoint\ f\ x)$
definition *the-greatest-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a\ (\nu\ -\ [201]\ 200)$ **where** $\nu\ \ f = (THE\ x\ .\ is\text{-}greatest\text{-}fixpoint\ f\ x)$
definition *the-least-prefixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a\ (p\mu\ -\ [201]\ 200)$ **where** $p\mu\ f = (THE\ x\ .\ is\text{-}least\text{-}prefixpoint\ f\ x)$
definition *the-greatest-postfixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a\ (p\nu\ -\ [201]\ 200)$ **where** $p\nu\ f = (THE\ x\ .\ is\text{-}greatest\text{-}postfixpoint\ f\ x)$

lemma *least-fixpoint-unique*: $has\text{-}least\text{-}fixpoint\ f \longrightarrow (\exists!x\ .\ is\text{-}least\text{-}fixpoint\ f\ x)$
by (*smt antisym has-least-fixpoint-def is-least-fixpoint-def*)

lemma *greatest-fixpoint-unique*: $has\text{-}greatest\text{-}fixpoint\ f \longrightarrow (\exists!x\ .\ is\text{-}greatest\text{-}fixpoint\ f\ x)$
by (*smt antisym has-greatest-fixpoint-def is-greatest-fixpoint-def*)

lemma *least-prefixpoint-unique*: $has\text{-}least\text{-}prefixpoint\ f \longrightarrow (\exists!x\ .\ is\text{-}least\text{-}prefixpoint\ f\ x)$
by (*smt antisym has-least-prefixpoint-def is-least-prefixpoint-def*)

lemma *greatest-postfixpoint-unique*: $has\text{-}greatest\text{-}postfixpoint\ f \longrightarrow (\exists!x\ .\ is\text{-}greatest\text{-}postfixpoint\ f\ x)$
by (*smt antisym has-greatest-postfixpoint-def is-greatest-postfixpoint-def*)

lemma *least-fixpoint*: $has\text{-}least\text{-}fixpoint\ f \longrightarrow is\text{-}least\text{-}fixpoint\ f\ (\mu\ f)$
proof
assume *has-least-fixpoint f*
hence $is\text{-}least\text{-}fixpoint\ f\ (THE\ x\ .\ is\text{-}least\text{-}fixpoint\ f\ x)$
by (*smt least-fixpoint-unique theI'*)
thus $is\text{-}least\text{-}fixpoint\ f\ (\mu\ f)$
by (*simp add*: *is-least-fixpoint-def the-least-fixpoint-def*)
qed

lemma *greatest-fixpoint*: $has\text{-}greatest\text{-}fixpoint\ f \longrightarrow is\text{-}greatest\text{-}fixpoint\ f\ (\nu\ f)$
proof
assume *has-greatest-fixpoint f*
hence $is\text{-}greatest\text{-}fixpoint\ f\ (THE\ x\ .\ is\text{-}greatest\text{-}fixpoint\ f\ x)$
by (*smt greatest-fixpoint-unique theI'*)
thus $is\text{-}greatest\text{-}fixpoint\ f\ (\nu\ f)$
by (*simp add*: *is-greatest-fixpoint-def the-greatest-fixpoint-def*)
qed

lemma *least-prefixpoint*: $has\text{-}least\text{-}prefixpoint\ f \longrightarrow is\text{-}least\text{-}prefixpoint\ f\ (p\mu\ f)$
proof
assume *has-least-prefixpoint f*

hence *is-least-prefixpoint f* (*THE x* . *is-least-prefixpoint f x*)
by (*smt least-prefixpoint-unique theI'*)
thus *is-least-prefixpoint f* (*pμ f*)
by (*simp add*: *is-least-prefixpoint-def the-least-prefixpoint-def*)
qed

lemma *greatest-postfixpoint*: *has-greatest-postfixpoint f* ⟶ *is-greatest-postfixpoint f* (*pν f*)
proof
assume *has-greatest-postfixpoint f*
hence *is-greatest-postfixpoint f* (*THE x* . *is-greatest-postfixpoint f x*)
by (*smt greatest-postfixpoint-unique theI'*)
thus *is-greatest-postfixpoint f* (*pν f*)
by (*simp add*: *is-greatest-postfixpoint-def the-greatest-postfixpoint-def*)
qed

lemma *least-fixpoint-same*: *is-least-fixpoint f x* ⟶ *x* = *μ f*
by (*metis least-fixpoint least-fixpoint-unique has-least-fixpoint-def*)

lemma *greatest-fixpoint-same*: *is-greatest-fixpoint f x* ⟶ *x* = *ν f*
by (*metis greatest-fixpoint greatest-fixpoint-unique has-greatest-fixpoint-def*)

lemma *least-prefixpoint-same*: *is-least-prefixpoint f x* ⟶ *x* = *pμ f*
by (*metis least-prefixpoint least-prefixpoint-unique has-least-prefixpoint-def*)

lemma *greatest-postfixpoint-same*: *is-greatest-postfixpoint f x* ⟶ *x* = *pν f*
by (*metis greatest-postfixpoint greatest-postfixpoint-unique has-greatest-postfixpoint-def*)

lemma *least-fixpoint-char*: *is-least-fixpoint f x* ⟷ *has-least-fixpoint f* ∧ *x* = *μ f*
by (*metis least-fixpoint-same has-least-fixpoint-def*)

lemma *least-prefixpoint-char*: *is-least-prefixpoint f x* ⟷ *has-least-prefixpoint f* ∧ *x* = *pμ f*
by (*metis least-prefixpoint-same has-least-prefixpoint-def*)

lemma *greatest-fixpoint-char*: *is-greatest-fixpoint f x* ⟷ *has-greatest-fixpoint f* ∧ *x* = *ν f*
by (*metis greatest-fixpoint-same has-greatest-fixpoint-def*)

lemma *greatest-postfixpoint-char*: *is-greatest-postfixpoint f x* ⟷ *has-greatest-postfixpoint f* ∧ *x* = *pν f*
by (*metis greatest-postfixpoint-same has-greatest-postfixpoint-def*)

lemma *mu-unfold*: *has-least-fixpoint f* ⟶ *f* (*μ f*) = *μ f*
by (*metis is-least-fixpoint-def least-fixpoint*)

lemma *pmu-unfold*: *has-least-prefixpoint f* ⟶ *f* (*pμ f*) ≤ *pμ f*
by (*metis is-least-prefixpoint-def least-prefixpoint*)

lemma *nu-unfold*: *has-greatest-fixpoint f* ⟶ *ν f* = *f* (*ν f*)
by (*metis is-greatest-fixpoint-def greatest-fixpoint*)

lemma *pnu-unfold*: *has-greatest-postfixpoint f* ⟶ *pν f* ≤ *f* (*pν f*)
by (*metis is-greatest-postfixpoint-def greatest-postfixpoint*)

lemma *least-prefixpoint-fixpoint*: *has-least-prefixpoint f* ∧ *isotone f* ⟶ *is-least-fixpoint f* (*pμ f*)
by (*smt eq-iff is-least-fixpoint-def is-least-prefixpoint-def isotone-def least-prefixpoint*)

lemma *pmu-mu*: *has-least-prefixpoint f* ∧ *isotone f* ⟶ *pμ f* = *μ f*
by (*smt has-least-fixpoint-def is-least-fixpoint-def least-fixpoint-unique least-prefixpoint-fixpoint least-fixpoint*)

lemma *greatest-postfixpoint-fixpoint*: *has-greatest-postfixpoint f* ∧ *isotone f* ⟶ *is-greatest-fixpoint f* (*pν f*)
by (*smt eq-iff is-greatest-fixpoint-def is-greatest-postfixpoint-def isotone-def greatest-postfixpoint*)

lemma *pnu-nu*: *has-greatest-postfixpoint f* ∧ *isotone f* ⟶ *pν f* = *ν f*
by (*smt has-greatest-fixpoint-def is-greatest-fixpoint-def greatest-fixpoint-unique greatest-postfixpoint-fixpoint greatest-fixpoint*)

lemma *pmu-isotone*: *has-least-prefixpoint f* ∧ *has-least-prefixpoint g* ∧ *f* ≤≤ *g* ⟶ *pμ f* ≤ *pμ g*
by (*smt is-least-prefixpoint-def least-prefixpoint lifted-less-eq-def order-trans*)

lemma *mu-isotone*: *has-least-prefixpoint f* ∧ *has-least-prefixpoint g* ∧ *isotone f* ∧ *isotone g* ∧ *f* ≤≤ *g* ⟶ *μ f* ≤ *μ g*
by (*metis pmu-isotone pmu-mu*)

lemma *pnu-isotone*: *has-greatest-postfixpoint f* ∧ *has-greatest-postfixpoint g* ∧ *f* ≤≤ *g* ⟶ *pν f* ≤ *pν g*
by (*smt is-greatest-postfixpoint-def lifted-less-eq-def order-trans greatest-postfixpoint*)

lemma *nu-isotone*: *has-greatest-postfixpoint f* ∧ *has-greatest-postfixpoint g* ∧ *isotone f* ∧ *isotone g* ∧ *f* ≤≤ *g* ⟶ *ν f* ≤ *ν g*
by (*metis pnu-isotone pnu-nu*)

lemma *mu-square*: *isotone f* ∧ *has-least-fixpoint f* ∧ *has-least-fixpoint* (*f* ∘ *f*) ⟶ *μ f* = *μ* (*f* ∘ *f*)
by (*metis* (*no-types*, *hide-lams*) *antisym is-least-fixpoint-def ord.isotone-def least-fixpoint-char least-fixpoint-unique o-apply*)

lemma *nu-square*: *isotone f* ∧ *has-greatest-fixpoint f* ∧ *has-greatest-fixpoint* (*f* ∘ *f*) ⟶ *ν f* = *ν* (*f* ∘ *f*)
by (*metis* (*no-types*, *hide-lams*) *antisym is-greatest-fixpoint-def ord.isotone-def greatest-fixpoint-char greatest-fixpoint-unique o-apply*)

lemma *mu-roll*: *isotone g* ∧ *has-least-fixpoint* (*f* ∘ *g*) ∧ *has-least-fixpoint* (*g* ∘ *f*) ⟶ *μ* (*g* ∘ *f*) = *g*(*μ* (*f* ∘ *g*))
apply (*rule impI*)
apply (*rule antisym*)
apply (*smt is-least-fixpoint-def least-fixpoint o-apply*)
apply (*smt is-least-fixpoint-def isotone-def least-fixpoint o-apply*)
done

lemma *nu-roll*: *isotone g* ∧ *has-greatest-fixpoint* (*f* ∘ *g*) ∧ *has-greatest-fixpoint* (*g* ∘ *f*) ⟶ *ν* (*g* ∘ *f*) = *g*(*ν* (*f* ∘ *g*))
apply (*rule impI*)
apply (*rule antisym*)
apply (*smt is-greatest-fixpoint-def greatest-fixpoint isotone-def o-apply*)
apply (*smt is-greatest-fixpoint-def greatest-fixpoint o-apply*)
done

lemma *mu-below-nu*: *has-least-fixpoint f* ∧ *has-greatest-fixpoint f* ⟶ *μ f* ≤ *ν f*
by (*metis is-greatest-fixpoint-def is-least-fixpoint-def least-fixpoint greatest-fixpoint*)

lemma *pmu-below-pnu-fix*: *has-fixpoint f* ∧ *has-least-prefixpoint f* ∧ *has-greatest-postfixpoint f* ⟶ *pμ f* ≤ *pν f*
by (*smt has-fixpoint-def is-fixpoint-def is-greatest-postfixpoint-def is-least-prefixpoint-def le-less order-trans least-prefixpoint greatest-postfixpoint*)

lemma *pmu-below-pnu-iso*: *isotone f* ∧ *has-least-prefixpoint f* ∧ *has-greatest-postfixpoint f* ⟶ *pμ f* ≤ *pν f*
by (*metis has-fixpoint-def is-fixpoint-def is-least-fixpoint-def least-prefixpoint-fixpoint pmu-below-pnu-fix*)

lemma *mu-fusion-1*: *galois l u* ∧ *isotone h* ∧ *has-least-prefixpoint g* ∧ *has-least-fixpoint h* ∧ *l*(*g*(*u*(*μ h*))) ≤ *h*(*l*(*u*(*μ h*))) ⟶ *l*(*pμ g*) ≤ *μ h*
proof
assume *1*: *galois l u* ∧ *isotone h* ∧ *has-least-prefixpoint g* ∧ *has-least-fixpoint h* ∧ *l*(*g*(*u*(*μ h*))) ≤ *h*(*l*(*u*(*μ h*)))
hence *l*(*g*(*u*(*μ h*))) ≤ *μ h*
by (*metis galois-char least-fixpoint-same least-fixpoint-unique is-least-fixpoint-def isotone-def order-trans*)
thus *l*(*pμ g*) ≤ *μ h* **using** *1*
by (*metis galois-def least-prefixpoint is-least-prefixpoint-def least-fixpoint-same least-fixpoint-unique*)
qed

lemma *mu-fusion-2*: *galois l u* ∧ *isotone h* ∧ *has-least-prefixpoint g* ∧ *has-least-fixpoint h* ∧ *l* ∘ *g* ≤≤ *h* ∘ *l* ⟶ *l*(*pμ g*) ≤ *μ h*
by (*metis lifted-less-eq-def mu-fusion-1 o-apply*)

lemma *mu-fusion-equal-1*: *galois l u* ∧ *isotone g* ∧ *isotone h* ∧ *has-least-prefixpoint g* ∧ *has-least-fixpoint h* ∧ *l*(*g*(*u*(*μ h*))) ≤ *h*(*l*(*u*(*μ h*))) ∧ *l*(*g*(*pμ g*)) = *h*(*l*(*pμ g*)) ⟶ *μ h* = *l*(*pμ g*) ∧ *μ h* = *l*(*μ g*)
by (*metis antisym least-fixpoint least-prefixpoint-fixpoint is-least-fixpoint-def mu-fusion-1 pmu-mu*)

lemma *mu-fusion-equal-2*: *galois l u* ∧ *isotone h* ∧ *has-least-prefixpoint g* ∧ *has-least-prefixpoint h* ∧ *l*(*g*(*u*(*μ h*))) ≤ *h*(*l*(*u*(*μ h*))) ∧ *l*(*g*(*pμ g*)) = *h*(*l*(*pμ g*)) ⟶ *pμ h* = *l*(*pμ g*) ∧ *μ h* = *l*(*pμ g*)
by (*smt antisym galois-char least-fixpoint-char least-prefixpoint least-prefixpoint-fixpoint is-least-prefixpoint-def isotone-def mu-fusion-1*)

lemma *mu-fusion-equal-3*: *galois l u* ∧ *isotone g* ∧ *isotone h* ∧ *has-least-prefixpoint g* ∧ *has-least-fixpoint h* ∧ *l* ∘ *g* = *h* ∘ *l* ⟶ *μ h* = *l*(*pμ g*) ∧ *μ h* = *l*(*μ g*)
by (*metis mu-fusion-equal-1 o-apply order-refl*)

lemma *mu-fusion-equal-4*: *galois l u* ∧ *isotone h* ∧ *has-least-prefixpoint g* ∧ *has-least-prefixpoint h* ∧ *l* ∘ *g* = *h* ∘ *l* ⟶ *pμ h* = *l*(*pμ g*) ∧ *μ h* = *l*(*pμ g*)
by (*metis mu-fusion-equal-2 o-apply order-refl*)

lemma *nu-fusion-1*: *galois l u* ∧ *isotone h* ∧ *has-greatest-postfixpoint g* ∧ *has-greatest-fixpoint h* ∧ *h*(*u*(*l*(*ν h*))) ≤ *u*(*g*(*l*(*ν h*))) ⟶ *ν h* ≤ *u*(*pν g*)

proof
assume *1*: *galois l u* ∧ *isotone h* ∧ *has-greatest-postfixpoint g* ∧ *has-greatest-fixpoint h* ∧ $h(u(l(\nu\ h))) \leq u(g(l(\nu\ h)))$
hence $\nu\ h \leq u(g(l(\nu\ h)))$ **using** *1*
by (*metis galois-char greatest-fixpoint-same greatest-fixpoint-unique is-greatest-fixpoint-def isotone-def order-trans*)
thus $\nu\ h \leq u(p\nu\ g)$ **using** *1*
by (*smt galois-def greatest-postfixpoint is-greatest-postfixpoint-def greatest-fixpoint-same greatest-fixpoint-unique*)
qed

lemma *nu-fusion-2*: *galois l u* ∧ *isotone h* ∧ *has-greatest-postfixpoint g* ∧ *has-greatest-fixpoint h* ∧ $h \circ u \leq\leq u \circ g \longrightarrow \nu\ h \leq u(p\nu\ g)$
by (*metis lifted-less-eq-def nu-fusion-1 o-apply*)

lemma *nu-fusion-equal-1*: *galois l u* ∧ *isotone g* ∧ *isotone h* ∧ *has-greatest-postfixpoint g* ∧ *has-greatest-fixpoint h* ∧ $h(u(l(\nu\ h))) \leq u(g(l(\nu\ h)))$ ∧ $h(u(p\nu\ g)) = u(g(p\nu\ g)) \longrightarrow \nu\ h = u(p\nu\ g) \wedge \nu\ h = u(\nu\ g)$
by (*metis antisym greatest-fixpoint greatest-postfixpoint-fixpoint is-greatest-fixpoint-def nu-fusion-1 pnu-nu*)

lemma *nu-fusion-equal-2*: *galois l u* ∧ *isotone h* ∧ *has-greatest-postfixpoint g* ∧ *has-greatest-postfixpoint h* ∧ $h(u(l(\nu\ h))) \leq u(g(l(\nu\ h)))$ ∧ $h(u(p\nu\ g)) = u(g(p\nu\ g)) \longrightarrow p\nu\ h = u(p\nu\ g) \wedge \nu\ h = u(p\nu\ g)$
by (*smt antisym galois-char greatest-fixpoint-char greatest-postfixpoint greatest-postfixpoint-fixpoint is-greatest-postfixpoint-def isotone-def nu-fusion-1*)

lemma *nu-fusion-equal-3*: *galois l u* ∧ *isotone g* ∧ *isotone h* ∧ *has-greatest-postfixpoint g* ∧ *has-greatest-fixpoint h* ∧ $h \circ u = u \circ g \longrightarrow \nu\ h = u(p\nu\ g) \wedge \nu\ h = u(\nu\ g)$
by (*metis nu-fusion-equal-1 o-apply order-refl*)

lemma *nu-fusion-equal-4*: *galois l u* ∧ *isotone h* ∧ *has-greatest-postfixpoint g* ∧ *has-greatest-postfixpoint h* ∧ $h \circ u = u \circ g \longrightarrow p\nu\ h = u(p\nu\ g) \wedge \nu\ h = u(p\nu\ g)$
by (*metis nu-fusion-equal-2 o-apply order-refl*)

lemma *mu-exchange-1*: *galois l u* ∧ *isotone g* ∧ *isotone h* ∧ *has-least-prefixpoint* $(l \circ h)$ ∧ *has-least-prefixpoint* $(h \circ g)$ ∧ *has-least-fixpoint* $(g \circ h)$ ∧ $l \circ h \circ g \leq\leq g \circ h \circ l \longrightarrow \mu(l \circ h) \leq \mu(g \circ h)$
proof
assume *1*: *galois l u* ∧ *isotone g* ∧ *isotone h* ∧ *has-least-prefixpoint* $(l \circ h)$ ∧ *has-least-prefixpoint* $(h \circ g)$ ∧ *has-least-fixpoint* $(g \circ h)$ ∧ $l \circ h \circ g \leq\leq g \circ h \circ l$
hence $l \circ (h \circ g) \leq\leq (g \circ h) \circ l$
by (*metis o-assoc*)
thus $\mu(l \circ h) \leq \mu(g \circ h)$ **using** *1*
by (*smt galois-char is-least-prefixpoint-def isotone-def least-fixpoint-char least-prefixpoint least-prefixpoint-fixpoint mu-fusion-2 mu-roll o-apply*)
qed

lemma *mu-exchange-2*: *galois l u* ∧ *isotone g* ∧ *isotone h* ∧ *has-least-prefixpoint* $(l \circ h)$ ∧ *has-least-prefixpoint* $(h \circ l)$ ∧ *has-least-prefixpoint* $(h \circ g)$ ∧ *has-least-fixpoint* $(g \circ h)$ ∧ *has-least-fixpoint* $(h \circ g)$ ∧ $l \circ h \circ g \leq\leq g \circ h \circ l \longrightarrow \mu(h \circ l) \leq \mu(h \circ g)$
by (*smt galois-char isotone-def least-fixpoint-char least-prefixpoint-fixpoint mu-exchange-1 mu-roll o-apply*)

lemma *mu-exchange-equal*: *galois l u* ∧ *galois k t* ∧ *isotone h* ∧ *has-least-prefixpoint* $(l \circ h)$ ∧ *has-least-prefixpoint* $(h \circ l)$ ∧ *has-least-prefixpoint* $(k \circ h)$ ∧ *has-least-prefixpoint* $(h \circ k)$ ∧ $l \circ h \circ k = k \circ h \circ l \longrightarrow \mu(l \circ h) = \mu(k \circ h) \wedge \mu(h \circ l) = \mu(h \circ k)$
by (*smt antisym galois-char isotone-def least-fixpoint-char least-prefixpoint-fixpoint lifted-reflexive mu-exchange-1 mu-exchange-2 o-apply*)

lemma *nu-exchange-1*: *galois l u* ∧ *isotone g* ∧ *isotone h* ∧ *has-greatest-postfixpoint* $(u \circ h)$ ∧ *has-greatest-postfixpoint* $(h \circ g)$ ∧ *has-greatest-fixpoint* $(g \circ h)$ ∧ $g \circ h \circ u \leq\leq u \circ h \circ g \longrightarrow \nu(g \circ h) \leq \nu(u \circ h)$
proof
assume *1*: *galois l u* ∧ *isotone g* ∧ *isotone h* ∧ *has-greatest-postfixpoint* $(u \circ h)$ ∧ *has-greatest-postfixpoint* $(h \circ g)$ ∧ *has-greatest-fixpoint* $(g \circ h)$ ∧ $g \circ h \circ u \leq\leq u \circ h \circ g$
hence $(g \circ h) \circ u \leq\leq u \circ (h \circ g)$
by (*metis o-assoc*)
thus $\nu(g \circ h) \leq \nu(u \circ h)$ **using** *1*
by (*smt galois-char is-greatest-postfixpoint-def isotone-def greatest-fixpoint-char greatest-postfixpoint greatest-postfixpoint-fixpoint nu-fusion-2 nu-roll o-apply*)
qed

lemma *nu-exchange-2*: *galois l u* ∧ *isotone g* ∧ *isotone h* ∧ *has-greatest-postfixpoint* $(u \circ h)$ ∧ *has-greatest-postfixpoint* $(h \circ u)$ ∧ *has-greatest-postfixpoint* $(h \circ g)$ ∧ *has-greatest-fixpoint* $(g \circ h)$ ∧ *has-greatest-fixpoint* $(h \circ g)$ ∧ $g \circ h \circ u \leq\leq u \circ h \circ g \longrightarrow \nu(h \circ g) \leq \nu(h \circ u)$
by (*smt galois-char isotone-def greatest-fixpoint-char greatest-postfixpoint-fixpoint nu-exchange-1 nu-roll o-apply*)

lemma *nu-exchange-equal*: *galois l u* ∧ *galois k t* ∧ *isotone h* ∧ *has-greatest-postfixpoint* (*u* ∘ *h*) ∧ *has-greatest-postfixpoint* (*h* ∘ *u*) ∧ *has-greatest-postfixpoint* (*t* ∘ *h*) ∧ *has-greatest-postfixpoint* (*h* ∘ *t*) ∧ *u* ∘ *h* ∘ *t* = *t* ∘ *h* ∘ *u* ⟶ *ν*(*u* ∘ *h*) = *ν*(*t* ∘ *h*) ∧ *ν*(*h* ∘ *u*) = *ν*(*h* ∘ *t*)
by (*smt antisym galois-char isotone-def greatest-fixpoint-char greatest-postfixpoint-fixpoint lifted-reflexive nu-exchange-1 nu-exchange-2 o-apply*)

lemma *mu-commute-fixpoint-1*: *isotone f* ∧ *has-least-fixpoint* (*f* ∘ *g*) ∧ *f* ∘ *g* = *g* ∘ *f* ⟶ *is-fixpoint f* (*μ*(*f* ∘ *g*))
by (*metis is-fixpoint-def mu-roll*)

lemma *mu-commute-fixpoint-2*: *isotone g* ∧ *has-least-fixpoint* (*f* ∘ *g*) ∧ *f* ∘ *g* = *g* ∘ *f* ⟶ *is-fixpoint g* (*μ*(*f* ∘ *g*))
by (*metis is-fixpoint-def mu-roll*)

lemma *mu-commute-least-fixpoint*: *isotone f* ∧ *isotone g* ∧ *has-least-fixpoint f* ∧ *has-least-fixpoint g* ∧ *has-least-fixpoint* (*f* ∘ *g*) ∧ *f* ∘ *g* = *g* ∘ *f* ⟶ (*μ*(*f* ∘ *g*) = *μ f* ⟶ *μ g* ≤ *μ f*)
by (*metis is-least-fixpoint-def least-fixpoint-same least-fixpoint-unique mu-roll*)

lemma *nu-commute-fixpoint-1*: *isotone f* ∧ *has-greatest-fixpoint* (*f* ∘ *g*) ∧ *f* ∘ *g* = *g* ∘ *f* ⟶ *is-fixpoint f* (*ν*(*f* ∘ *g*))
by (*metis is-fixpoint-def nu-roll*)

lemma *nu-commute-fixpoint-2*: *isotone g* ∧ *has-greatest-fixpoint* (*f* ∘ *g*) ∧ *f* ∘ *g* = *g* ∘ *f* ⟶ *is-fixpoint g* (*ν*(*f* ∘ *g*))
by (*metis is-fixpoint-def nu-roll*)

lemma *nu-commute-greatest-fixpoint*: *isotone f* ∧ *isotone g* ∧ *has-greatest-fixpoint f* ∧ *has-greatest-fixpoint g* ∧ *has-greatest-fixpoint* (*f* ∘ *g*) ∧ *f* ∘ *g* = *g* ∘ *f* ⟶ (*ν*(*f* ∘ *g*) = *ν f* ⟶ *ν f* ≤ *ν g*)
by (*smt is-greatest-fixpoint-def greatest-fixpoint-same greatest-fixpoint-unique nu-roll*)

lemma *mu-diagonal-1*: *isotone* (*λx* . *f x x*) ∧ (∀ *x* . *isotone* (*λy* . *f x y*)) ∧ *isotone* (*λx* . *μ*(*λy* . *f x y*)) ∧ (∀ *x* . *has-least-fixpoint* (*λy* . *f x y*)) ∧ *has-least-prefixpoint* (*λx* . *μ*(*λy* . *f x y*)) ⟶ *μ*(*λx* . *f x x*) = *μ*(*λx* . *μ*(*λy* . *f x y*))
by (*smt is-least-fixpoint-def is-least-prefixpoint-def least-fixpoint-same least-fixpoint-unique least-prefixpoint least-prefixpoint-fixpoint*)

lemma *mu-diagonal-2*: (∀ *x* . *isotone* (*λy* . *f x y*) ∧ *isotone* (*λy* . *f y x*) ∧ *has-least-prefixpoint* (*λy* . *f x y*)) ∧ *has-least-prefixpoint* (*λx* . *μ*(*λy* . *f x y*)) ⟶ *μ*(*λx* . *f x x*) = *μ*(*λx* . *μ*(*λy* . *f x y*))
proof
assume *1*: (∀ *x* . *isotone* (*λy* . *f x y*) ∧ *isotone* (*λy* . *f y x*) ∧ *has-least-prefixpoint* (*λy* . *f x y*)) ∧ *has-least-prefixpoint* (*λx* . *μ*(*λy* . *f x y*))
hence *isotone* (*λx* . *μ*(*λy* . *f x y*))
by (*smt isotone-def lifted-less-eq-def mu-isotone*)
thus *μ*(*λx* . *f x x*) = *μ*(*λx* . *μ*(*λy* . *f x y*)) **using** *1*
by (*smt is-least-fixpoint-def is-least-prefixpoint-def least-fixpoint-same least-prefixpoint least-prefixpoint-fixpoint*)
qed

lemma *nu-diagonal-1*: *isotone* (*λx* . *f x x*) ∧ (∀ *x* . *isotone* (*λy* . *f x y*)) ∧ *isotone* (*λx* . *ν*(*λy* . *f x y*)) ∧ (∀ *x* . *has-greatest-fixpoint* (*λy* . *f x y*)) ∧ *has-greatest-postfixpoint* (*λx* . *ν*(*λy* . *f x y*)) ⟶ *ν*(*λx* . *f x x*) = *ν*(*λx* . *ν*(*λy* . *f x y*))
by (*smt is-greatest-fixpoint-def is-greatest-postfixpoint-def greatest-fixpoint-same greatest-fixpoint-unique greatest-postfixpoint greatest-postfixpoint-fixpoint*)

lemma *nu-diagonal-2*: (∀ *x* . *isotone* (*λy* . *f x y*) ∧ *isotone* (*λy* . *f y x*) ∧ *has-greatest-postfixpoint* (*λy* . *f x y*)) ∧ *has-greatest-postfixpoint* (*λx* . *ν*(*λy* . *f x y*)) ⟶ *ν*(*λx* . *f x x*) = *ν*(*λx* . *ν*(*λy* . *f x y*))
proof
assume *1*: (∀ *x* . *isotone* (*λy* . *f x y*) ∧ *isotone* (*λy* . *f y x*) ∧ *has-greatest-postfixpoint* (*λy* . *f x y*)) ∧ *has-greatest-postfixpoint* (*λx* . *ν*(*λy* . *f x y*))
hence *isotone* (*λx* . *ν*(*λy* . *f x y*))
by (*smt isotone-def lifted-less-eq-def nu-isotone*)
thus *ν*(*λx* . *f x x*) = *ν*(*λx* . *ν*(*λy* . *f x y*)) **using** *1*
by (*smt greatest-fixpoint-same greatest-postfixpoint greatest-postfixpoint-fixpoint is-greatest-fixpoint-def is-greatest-postfixpoint-def*)
qed

end

end

3 Lattice

theory *Lattice*

imports *Base*

begin

class *join-semilattice* = *plus* + *ord* +
  assumes *add-associative*: $(x + y) + z = x + (y + z)$
  assumes *add-commutative*: $x + y = y + x$
  assumes *add-idempotent*: $x + x = x$
  assumes *less-eq-def*: $x \leq y \longleftrightarrow x + y = y$
  assumes *less-def*: $x < y \longleftrightarrow x \leq y \wedge \neg\,(y \leq x)$

begin

subclass *order*
  apply *unfold-locales*
  apply (*metis less-def*)
  apply (*metis add-idempotent less-eq-def*)
  apply (*metis add-associative less-eq-def*)
  apply (*metis add-commutative less-eq-def*)
  done

lemma *add-left-isotone*: $x \leq y \longrightarrow x + z \leq y + z$
  by (*smt add-associative add-commutative add-idempotent less-eq-def*)

lemma *add-right-isotone*: $x \leq y \longrightarrow z + x \leq z + y$
  by (*metis add-commutative add-left-isotone*)

lemma *add-isotone*: $w \leq y \wedge x \leq z \longrightarrow w + x \leq y + z$
  by (*smt add-associative add-commutative less-eq-def*)

lemma *add-left-upper-bound*: $x \leq x + y$
  by (*metis add-associative add-idempotent less-eq-def*)

lemma *add-right-upper-bound*: $y \leq x + y$
  by (*metis add-commutative add-left-upper-bound*)

lemma *add-least-upper-bound*: $x \leq z \wedge y \leq z \longleftrightarrow x + y \leq z$
  by (*smt add-associative add-commutative add-left-upper-bound less-eq-def*)

lemma *add-left-divisibility*: $x \leq y \longleftrightarrow (\exists\, z\ .\ x + z = y)$
  by (*metis add-left-upper-bound less-eq-def*)

lemma *add-right-divisibility*: $x \leq y \longleftrightarrow (\exists\, z\ .\ z + x = y)$
  by (*metis add-commutative add-left-divisibility*)

lemma *add-same-context*:$x \leq y + z \wedge y \leq x + z \longrightarrow x + z = y + z$
  by (*smt add-associative add-commutative less-eq-def*)

lemma *add-relative-same-increasing*: $x \leq y \wedge x + z = x + w \longrightarrow y + z = y + w$
  by (*smt add-associative add-right-divisibility*)

lemma *ascending-chain-left-add*: *ascending-chain f* $\longrightarrow$ *ascending-chain* $(\lambda n\ .\ x + f\ n)$
  by (*metis ascending-chain-def add-right-isotone*)

lemma *ascending-chain-right-add*: *ascending-chain f* $\longrightarrow$ *ascending-chain* $(\lambda n\ .\ f\ n + x)$
  by (*metis ascending-chain-def add-left-isotone*)

lemma *descending-chain-left-add*: *descending-chain f* $\longrightarrow$ *descending-chain* $(\lambda n\ .\ x + f\ n)$
  by (*metis descending-chain-def add-right-isotone*)

lemma *descending-chain-right-add*: *descending-chain f* $\longrightarrow$ *descending-chain* $(\lambda n\ .\ f\ n + x)$
  by (*metis descending-chain-def add-left-isotone*)

primrec *pSum0* :: $(nat \Rightarrow 'a) \Rightarrow nat \Rightarrow 'a$
  where *pSum0 f 0* = *f 0*

| *pSum0 f (Suc m) = pSum0 f m + f m*

lemma *pSum0-below*: $(\forall\, i\ .\ f\ i \leq x) \longrightarrow pSum0\ f\ m \leq x$
apply (*induct m*)
apply (*metis pSum0.simps(1)*)
by (*metis add-least-upper-bound pSum0.simps(2)*)

end

class *bounded-join-semilattice = join-semilattice + zero +*
assumes *add-left-zero*: $0 + x = x$

begin

lemma *add-right-zero*: $x + 0 = x$
by (*metis add-commutative add-left-zero*)

lemma *zero-least*: $0 \leq x$
by (*metis add-left-upper-bound add-left-zero*)

end

class *meet =*
fixes *meet* :: $'a \Rightarrow 'a \Rightarrow 'a$ (**infixl** $\frown$ *65*)

class *meet-semilattice = meet + ord +*
assumes *meet-associative*: $(x \frown y) \frown z = x \frown (y \frown z)$
assumes *meet-commutative*: $x \frown y = y \frown x$
assumes *meet-idempotent*: $x \frown x = x$
assumes *meet-less-eq-def*: $x \leq y \longleftrightarrow x \frown y = x$
assumes *meet-less-def*: $x < y \longleftrightarrow x \leq y \wedge \neg\, (y \leq x)$

sublocale *meet-semilattice < meet*!: *join-semilattice* **where** *plus = meet* **and** *less-eq* $= (\lambda x\ y\ .\ y \leq x)$ **and** *less* $= (\lambda x\ y\ .\ y < x)$
apply *unfold-locales*
apply (*rule meet-associative*)
apply (*rule meet-commutative*)
apply (*rule meet-idempotent*)
apply (*metis meet-commutative meet-less-eq-def*)
apply (*metis meet-less-def*)
done

class *T =*
fixes *T* :: $'a$ ($\top$)

class *bounded-meet-semilattice = meet-semilattice + T +*
assumes *meet-left-top*: $T \frown x = x$

sublocale *bounded-meet-semilattice < meet*!: *bounded-join-semilattice* **where** *plus = meet* **and** *less-eq* $= (\lambda x\ y\ .\ y \leq x)$ **and** *less* $= (\lambda x\ y\ .\ y < x)$ **and** *zero = T*
apply *unfold-locales*
apply (*rule meet-left-top*)
done

class *bounded-distributive-lattice = bounded-join-semilattice + bounded-meet-semilattice +*
assumes *meet-left-dist-add*: $x \frown (y + z) = (x \frown y) + (x \frown z)$
assumes *add-left-dist-meet*: $x + (y \frown z) = (x + y) \frown (x + z)$
assumes *meet-absorb*: $x \frown (x + y) = x$
assumes *add-absorb*: $x + (x \frown y) = x$

begin

lemma *meet-left-zero*: $0 \frown x = 0$
by (*metis add-absorb add-left-zero*)

lemma *meet-right-zero*: $x \frown 0 = 0$
by (*metis meet-commutative meet-left-zero*)

lemma *add-left-top-1*: $T + x = T$

by (*metis add-absorb meet-left-top*)

lemma *add-right-top-1*: $x + T = T$
by (*metis add-commutative add-left-top-1*)

lemma *meet-same-context*:$x \leq y \frown z \wedge y \leq x \frown z \longrightarrow x \frown z = y \frown z$
by (*metis eq-iff meet.add-least-upper-bound*)

lemma *relative-equality*: $x + z = y + z \wedge x \frown z = y \frown z \longrightarrow x = y$
by (*metis add-absorb add-commutative add-left-dist-meet*)

end

end

4 Semiring

theory *Semiring*

imports *Fixpoint Lattice*

begin

class *monoid* = *mult* + *one* +
  assumes *mult-associative*: $(x\ ;\ y)\ ;\ z = x\ ;\ (y\ ;\ z)$
  assumes *mult-left-one-1*: $1\ ;\ x = x$
  assumes *mult-right-one*: $x\ ;\ 1 = x$

class *non-associative-left-semiring* = *bounded-join-semilattice* + *mult* + *one* +
  assumes *mult-left-sub-dist-add*: $x\ ;\ y + x\ ;\ z \leq x\ ;\ (y + z)$
  assumes *mult-right-dist-add*: $(x + y)\ ;\ z = x\ ;\ z + y\ ;\ z$
  assumes *mult-left-zero*: $0\ ;\ x = 0$
  assumes *mult-left-one*: $1\ ;\ x = x$
  assumes *mult-sub-right-one*: $x \leq x\ ;\ 1$

begin

lemma *mult-left-isotone*: $x \leq y \longrightarrow x\ ;\ z \leq y\ ;\ z$
  by (*metis less-eq-def mult-right-dist-add*)

lemma *mult-right-isotone*: $x \leq y \longrightarrow z\ ;\ x \leq z\ ;\ y$
  by (*metis add-least-upper-bound less-eq-def mult-left-sub-dist-add*)

lemma *mult-isotone*: $w \leq y \wedge x \leq z \longrightarrow w\ ;\ x \leq y\ ;\ z$
  by (*smt mult-left-isotone mult-right-isotone order-trans*)

lemma *affine-isotone*: *isotone* $(\lambda x\ .\ y\ ;\ x + z)$
  by (*smt add-commutative add-right-isotone isotone-def mult-right-isotone*)

lemma *mult-left-sub-dist-add-left*: $x\ ;\ y \leq x\ ;\ (y + z)$
  by (*metis add-left-upper-bound mult-right-isotone*)

lemma *mult-left-sub-dist-add-right*: $x\ ;\ z \leq x\ ;\ (y + z)$
  by (*metis add-right-upper-bound mult-right-isotone*)

lemma *mult-right-sub-dist-add-left*: $x\ ;\ z \leq (x + y)\ ;\ z$
  by (*metis add-left-upper-bound mult-right-dist-add*)

lemma *mult-right-sub-dist-add-right*: $y\ ;\ z \leq (x + y)\ ;\ z$
  by (*metis add-right-upper-bound mult-right-dist-add*)

lemma *case-split-left*: $1 \leq w + z \wedge w\ ;\ x \leq y \wedge z\ ;\ x \leq y \longrightarrow x \leq y$
  by (*smt add-associative add-commutative less-eq-def mult-left-one mult-right-dist-add order-refl*)

lemma *case-split-left-equal*: $w + z = 1 \wedge w\ ;\ x = w\ ;\ y \wedge z\ ;\ x = z\ ;\ y \longrightarrow x = y$
  by (*metis mult-left-one mult-right-dist-add*)

lemma *ascending-chain-left-mult*: *ascending-chain* $f \longrightarrow$ *ascending-chain* $(\lambda n\ .\ x\ ;\ f\ n)$
  by (*metis ascending-chain-def mult-right-isotone*)

lemma *ascending-chain-right-mult*: *ascending-chain* $f \longrightarrow$ *ascending-chain* $(\lambda n\ .\ f\ n\ ;\ x)$
  by (*metis ascending-chain-def mult-left-isotone*)

lemma *descending-chain-left-mult*: *descending-chain* $f \longrightarrow$ *descending-chain* $(\lambda n\ .\ x\ ;\ f\ n)$
  by (*metis descending-chain-def mult-right-isotone*)

lemma *descending-chain-right-mult*: *descending-chain* $f \longrightarrow$ *descending-chain* $(\lambda n\ .\ f\ n\ ;\ x)$
  by (*metis descending-chain-def mult-left-isotone*)

— Some results about transitive closures in this class and the next two classes are taken from a joint paper with Rudolf Berghammer.

abbreviation $Lf :: 'a \Rightarrow ('a \Rightarrow 'a)$ **where** $Lf\ y \equiv (\lambda\ x\ .\ 1 + x\ ;\ y)$
abbreviation $Rf :: 'a \Rightarrow ('a \Rightarrow 'a)$ **where** $Rf\ y \equiv (\lambda\ x\ .\ 1 + y\ ;\ x)$

abbreviation $Sf :: 'a \Rightarrow ('a \Rightarrow 'a)$ **where** $Sf\ y \equiv (\lambda\ x\ .\ 1 + y + x\ ;\ x)$

abbreviation $lstar :: 'a \Rightarrow 'a$ **where** $lstar\ y \equiv p\mu\ (Lf\ y)$
abbreviation $rstar :: 'a \Rightarrow 'a$ **where** $rstar\ y \equiv p\mu\ (Rf\ y)$
abbreviation $sstar :: 'a \Rightarrow 'a$ **where** $sstar\ y \equiv p\mu\ (Sf\ y)$

lemma *lstar-rec-isotone*: $isotone\ (Lf\ y)$
by (*smt2 add-isotone add-right-divisibility isotone-def mult-right-sub-dist-add-right order.refl*)

lemma *rstar-rec-isotone*: $isotone\ (Rf\ y)$
by (*metis add-isotone isotone-def less-eq-def mult-left-sub-dist-add-left order-refl*)

lemma *sstar-rec-isotone*: $isotone\ (Sf\ y)$
by (*simp add*: *add-right-isotone isotone-def mult-isotone*)

lemma *lstar-fixpoint*: $has\text{-}least\text{-}prefixpoint\ (Lf\ y) \longrightarrow lstar\ y = \mu\ (Lf\ y)$
by (*metis pmu-mu lstar-rec-isotone*)

lemma *rstar-fixpoint*: $has\text{-}least\text{-}prefixpoint\ (Rf\ y) \longrightarrow rstar\ y = \mu\ (Rf\ y)$
by (*metis pmu-mu rstar-rec-isotone*)

lemma *sstar-fixpoint*: $has\text{-}least\text{-}prefixpoint\ (Sf\ y) \longrightarrow sstar\ y = \mu\ (Sf\ y)$
by (*metis pmu-mu sstar-rec-isotone*)

lemma *sstar-increasing*: $has\text{-}least\text{-}prefixpoint\ (Sf\ y) \longrightarrow y \leq sstar\ y$
by (*metis add-least-upper-bound add-left-upper-bound order.trans pmu-unfold*)

lemma *rstar-below-sstar*: $has\text{-}least\text{-}prefixpoint\ (Rf\ y) \wedge has\text{-}least\text{-}prefixpoint\ (Sf\ y) \longrightarrow rstar\ y \leq sstar\ y$
proof
assume *1*: $has\text{-}least\text{-}prefixpoint\ (Rf\ y) \wedge has\text{-}least\text{-}prefixpoint\ (Sf\ y)$
hence $Rf\ y\ (sstar\ y) \leq Sf\ y\ (sstar\ y)$
by (*smt2 add-isotone add-left-upper-bound add-right-upper-bound dual-order.trans mult-left-isotone pmu-unfold*)
also have $... \leq sstar\ y$ **using** *1*
by (*metis* (*erased*, *lifting*) *pmu-unfold*)
finally show $rstar\ y \leq sstar\ y$ **using** *1*
by (*metis* (*erased*, *lifting*) *is-least-prefixpoint-def least-prefixpoint*)
qed

end

class *pre-left-semiring* = *non-associative-left-semiring* +
assumes *mult-semi-associative*: $(x\ ;\ y)\ ;\ z \leq x\ ;\ (y\ ;\ z)$

begin

lemma *mult-one-associative*: $x\ ;\ 1\ ;\ y = x\ ;\ y$
by (*metis dual-order.antisym mult-left-isotone mult-left-one mult-semi-associative mult-sub-right-one*)

lemma *mult-sup-associative-one*: $(x\ ;\ (y\ ;\ 1))\ ;\ z \leq x\ ;\ (y\ ;\ z)$
by (*metis mult-semi-associative mult-one-associative*)

lemma *rstar-increasing*: $has\text{-}least\text{-}prefixpoint\ (Rf\ y) \longrightarrow y \leq rstar\ y$
proof
assume $has\text{-}least\text{-}prefixpoint\ (Rf\ y)$
hence $Rf\ y\ (rstar\ y) \leq rstar\ y$
by (*metis pmu-unfold*)
thus $y \leq rstar\ y$
by (*metis add-least-upper-bound mult-right-isotone mult-sub-right-one order.trans*)
qed

end

class *residuated-pre-left-semiring* = *pre-left-semiring* + *inverse* +
assumes *lres-galois*: $x\ ;\ y \leq z \longleftrightarrow x \leq z\ /\ y$

begin

— Theorem 32.2

lemma *lres-left-isotone*: $x \leq y \longrightarrow x \mathbin{/} z \leq y \mathbin{/} z$
by (*metis lres-galois order.refl order.trans*)

— Theorem 32.2

lemma *lres-right-antitone*: $x \leq y \longrightarrow z \mathbin{/} y \leq z \mathbin{/} x$
by (*metis lres-galois mult-right-isotone order.refl order-trans*)

lemma *lres-inverse*: $(x \mathbin{/} y) \cdot y \leq x$
by (*metis lres-galois order-refl*)

lemma *lres-one*: $x \mathbin{/} 1 \leq x$
by (*metis dual-order.trans mult-sub-right-one lres-inverse*)

lemma *lres-mult-sub-lres-lres*: $x \mathbin{/} (z \cdot y) \leq (x \mathbin{/} y) \mathbin{/} z$
by (*metis lres-galois lres-inverse mult-semi-associative order.trans*)

— Theorem 32.4

lemma *mult-lres-sub-assoc*: $x \cdot (y \mathbin{/} z) \leq (x \cdot y) \mathbin{/} z$
by (*metis* (*full-types*) *lres-galois lres-inverse mult-right-isotone mult-semi-associative order-trans*)

lemma *lstar-below-rstar*: *has-least-prefixpoint* $(Lf\ y) \wedge$ *has-least-prefixpoint* $(Rf\ y) \longrightarrow lstar\ y \leq rstar\ y$
proof
assume *1*: *has-least-prefixpoint* $(Lf\ y) \wedge$ *has-least-prefixpoint* $(Rf\ y)$
have $y \cdot (rstar\ y \mathbin{/} y) \cdot y \leq y \cdot rstar\ y$
by (*metis mult-right-isotone mult-semi-associative order-trans lres-inverse*)
also have $\ldots \leq rstar\ y$ **using** *1*
by (*metis add-least-upper-bound pmu-unfold*)
finally have $y \cdot (rstar\ y \mathbin{/} y) \leq rstar\ y \mathbin{/} y$
by (*metis lres-galois*)
hence $Rf\ y\ (rstar\ y \mathbin{/} y) \leq rstar\ y \mathbin{/} y$ **using** *1*
by (*metis add-least-upper-bound lres-galois mult-left-one rstar-increasing*)
hence $rstar\ y \leq rstar\ y \mathbin{/} y$ **using** *1*
by (*metis is-least-prefixpoint-def least-prefixpoint*)
hence $Lf\ y\ (rstar\ y) \leq rstar\ y$ **using** *1*
by (*metis add-least-upper-bound lres-galois pmu-unfold*)
thus $lstar\ y \leq rstar\ y$ **using** *1*
by (*metis* (*erased*, *lifting*) *is-least-prefixpoint-def least-prefixpoint*)
qed

— The next proof follows an argument of Rudolf Berghammer; see Satz 10.1.5 in his 2012 book.

lemma *rstar-sstar*: *has-least-prefixpoint* $(Rf\ y) \wedge$ *has-least-prefixpoint* $(Sf\ y) \longrightarrow rstar\ y = sstar\ y$
proof
assume *1*: *has-least-prefixpoint* $(Rf\ y) \wedge$ *has-least-prefixpoint* $(Sf\ y)$
have $Rf\ y\ (rstar\ y \mathbin{/} rstar\ y) \cdot rstar\ y \leq rstar\ y + y \cdot ((rstar\ y \mathbin{/} rstar\ y) \cdot rstar\ y)$
by (*metis add-isotone mult-left-one mult-right-dist-add mult-semi-associative*)
also have $\ldots \leq rstar\ y + y \cdot rstar\ y$
by (*metis add-right-isotone mult-right-isotone lres-inverse*)
also have $\ldots \leq rstar\ y$ **using** *1*
by (*metis* (*full-types*) *add-least-upper-bound order-refl pmu-unfold*)
finally have $Rf\ y\ (rstar\ y \mathbin{/} rstar\ y) \leq rstar\ y \mathbin{/} rstar\ y$
by (*metis lres-galois*)
hence $rstar\ y \cdot rstar\ y \leq rstar\ y$ **using** *1*
by (*metis* (*erased*, *lifting*) *is-least-prefixpoint-def least-prefixpoint lres-galois*)
hence $y + rstar\ y \cdot rstar\ y \leq rstar\ y$ **using** *1*
by (*metis add-least-upper-bound rstar-increasing*)
hence $Sf\ y\ (rstar\ y) \leq rstar\ y$ **using** *1*
by (*metis* (*full-types*) *add-least-upper-bound pmu-unfold*)
hence $sstar\ y \leq rstar\ y$ **using** *1*
by (*metis* (*erased*, *lifting*) *is-least-prefixpoint-def least-prefixpoint*)
thus $rstar\ y = sstar\ y$ **using** *1*
by (*metis antisym rstar-below-sstar*)
qed

end

class *idempotent-left-semiring* = *non-associative-left-semiring* + *monoid*

begin

subclass *pre-left-semiring*
apply *unfold-locales*
apply (*metis mult-associative order-refl*)
done

lemma *zero-right-mult-decreasing*: $x\ ;\ 0 \leq x$
by (*metis add-right-zero mult-left-sub-dist-add-right mult-right-one*)

lemma *test-preserves-equation*: $p \leq p\ ;\ p \wedge p \leq 1 \longrightarrow (p\ ;\ x \leq x\ ;\ p \longleftrightarrow p\ ;\ x = p\ ;\ x\ ;\ p)$
apply *rule*
apply *rule*
apply (*rule antisym*)
apply (*smt antisym mult-associative mult-right-isotone mult-right-one*)
apply (*metis mult-right-isotone mult-right-one*)
apply (*metis mult-left-isotone mult-left-one*)
done

end

class *idempotent-left-zero-semiring* = *idempotent-left-semiring* +
assumes *mult-left-dist-add*: $x\ ;\ (y + z) = x\ ;\ y + x\ ;\ z$

begin

lemma *case-split-right*: $1 \leq w + z \wedge x\ ;\ w \leq y \wedge x\ ;\ z \leq y \longrightarrow x \leq y$
by (*smt add-associative add-commutative less-eq-def mult-left-dist-add mult-right-one*)

lemma *case-split-right-equal*: $w + z = 1 \wedge x\ ;\ w = y\ ;\ w \wedge x\ ;\ z = y\ ;\ z \longrightarrow x = y$
by (*metis mult-left-dist-add mult-right-one*)

end

class *idempotent-semiring* = *idempotent-left-zero-semiring* +
assumes *mult-right-zero*: $x\ ;\ 0 = 0$

class *bounded-pre-left-semiring* = *pre-left-semiring* + T +
assumes *add-right-top*: $x + T = T$

begin

lemma *add-left-top*: $T + x = T$
by (*metis add-commutative add-right-top*)

lemma *top-greatest*: $x \leq T$
by (*metis add-left-top add-right-upper-bound*)

lemma *top-left-mult-increasing*: $x \leq T\ ;\ x$
by (*metis mult-left-isotone mult-left-one top-greatest*)

lemma *top-right-mult-increasing*: $x \leq x\ ;\ T$
by (*metis order-trans mult-right-isotone mult-sub-right-one top-greatest*)

lemma *top-mult-top*: $T\ ;\ T = T$
by (*metis add-right-divisibility add-right-top top-right-mult-increasing*)

definition *vector* :: $'a \Rightarrow bool$
where *vector* $x \longleftrightarrow x = x\ ;\ T$

lemma *vector-zero*: *vector 0*
by (*metis mult-left-zero vector-def*)

lemma *vector-top*: *vector T*
by (*metis top-mult-top vector-def*)

lemma *vector-add-closed*: *vector* $x \wedge$ *vector* $y \longrightarrow$ *vector* $(x + y)$
by (*metis mult-right-dist-add vector-def*)

lemma *vector-left-mult-closed*: *vector y* ⟶ *vector* (*x* ; *y*)
by (*metis antisym mult-semi-associative top-right-mult-increasing vector-def*)

end

class *bounded-residuated-pre-left-semiring* = *residuated-pre-left-semiring* + *bounded-pre-left-semiring*

begin

— Theorem 32.8

lemma *lres-top-decreasing*: *x* / *T* ≤ *x*
by (*metis lres-one lres-right-antitone order-trans top-greatest*)

— Theorem 32.9

lemma *top-lres-absorb*: *T* / *x* = *T*
by (*metis eq-iff lres-galois top-greatest*)

end

class *bounded-idempotent-left-semiring* = *bounded-pre-left-semiring* + *idempotent-left-semiring*

class *bounded-idempotent-left-zero-semiring* = *bounded-idempotent-left-semiring* + *idempotent-left-zero-semiring*

class *bounded-idempotent-semiring* = *bounded-idempotent-left-zero-semiring* + *idempotent-semiring*

end

5 Itering

theory *Itering*

imports *Semiring*

begin

class *circ* =
  fixes *circ* :: $'a \Rightarrow 'a$ ($-^\circ$ [*100*] *100*)

class *left-conway-semiring* = *idempotent-left-semiring* + *circ* +
  assumes *circ-left-unfold*: *1* + x ; $x^\circ = x^\circ$
  assumes *circ-left-slide*: $(x\ ;\ y)^\circ\ ;\ x \leq x\ ;\ (y\ ;\ x)^\circ$
  assumes *circ-add-1*: $(x + y)^\circ = x^\circ\ ;\ (y\ ;\ x^\circ)^\circ$

begin

— Theorem 14.19

lemma *circ-mult-sub*: *1* + x ; $(y\ ;\ x)^\circ\ ;\ y \leq (x\ ;\ y)^\circ$
  by (*metis add-right-isotone circ-left-slide circ-left-unfold mult-associative mult-right-isotone*)

— Theorem 14.8

lemma *circ-right-unfold-sub*: *1* + $x^\circ\ ;\ x \leq x^\circ$
  by (*metis circ-mult-sub mult-left-one mult-right-one*)

— Theorem 1.1 and Theorem 14.1

lemma *circ-zero*: $0^\circ = 1$
  by (*metis add-right-zero circ-left-unfold mult-left-zero*)

— Theorem 1.4 and Theorem 1.13 and Theorem 14.4 and Theorem 14.13

lemma *circ-increasing*: $x \leq x^\circ$
  by (*metis add-right-upper-bound circ-left-unfold circ-right-unfold-sub mult-left-one mult-right-sub-dist-add-left order-trans*)

— Theorem 1.3 and Theorem 14.3

lemma *circ-reflexive*: *1* $\leq x^\circ$
  by (*metis add-left-divisibility circ-left-unfold*)

— Theorem 1.5

lemma *circ-mult-increasing*: $x \leq x\ ;\ x^\circ$
  by (*metis circ-reflexive mult-right-isotone mult-right-one*)

— Theorem 14.5

lemma *circ-mult-increasing-2*: $x \leq x^\circ\ ;\ x$
  by (*metis circ-reflexive mult-left-isotone mult-left-one-1*)

— Theorem 1.11 and Theorem 14.11

lemma *circ-transitive-equal*: $x^\circ\ ;\ x^\circ = x^\circ$
  by (*metis add-idempotent circ-add-1 circ-left-unfold mult-associative*)

— Theorem 1.17 and Theorem 14.17

lemma *circ-circ-circ*: $x^{\circ\circ\circ} = x^{\circ\circ}$
  by (*metis add-idempotent circ-add-1 circ-increasing circ-transitive-equal less-eq-def*)

— Theorem 1.18 and Theorem 14.18

lemma *circ-one*: $1^\circ = 1^{\circ\circ}$
  by (*metis circ-circ-circ circ-zero*)

— Theorem 14.21

lemma *circ-add-sub*: $(x^\circ \ ; \ y)^\circ \ ; \ x^\circ \leq (x + y)^\circ$
by (*metis circ-add-1 circ-left-slide*)

lemma *circ-plus-one*: $x^\circ = 1 + x^\circ$
by (*metis less-eq-def circ-reflexive*)

— Theorem 1.12 and Theorem 14.12

lemma *circ-rtc-2*: $1 + x + x^\circ \ ; \ x^\circ = x^\circ$
by (*metis add-associative circ-increasing circ-plus-one circ-transitive-equal less-eq-def*)

— Theorem 1.2 and Theorem 14.2

lemma *mult-zero-circ*: $(x \ ; \ 0)^\circ = 1 + x \ ; \ 0$
by (*metis circ-left-unfold mult-associative mult-left-zero*)

lemma *mult-zero-add-circ*: $(x + y \ ; \ 0)^\circ = x^\circ \ ; \ (y \ ; \ 0)^\circ$
by (*metis circ-add-1 mult-associative mult-left-zero*)

— Theorem 14.6

lemma *circ-plus-sub*: $x^\circ \ ; \ x \leq x \ ; \ x^\circ$
by (*metis circ-left-slide mult-left-one mult-right-one*)

lemma *circ-loop-fixpoint*: $y \ ; \ (y^\circ \ ; \ z) + z = y^\circ \ ; \ z$
by (*metis add-commutative circ-left-unfold mult-associative mult-left-one mult-right-dist-add*)

— Theorem 1.6 and Theorem 14.7

lemma *left-plus-below-circ*: $x \ ; \ x^\circ \leq x^\circ$
by (*metis add-right-upper-bound circ-left-unfold*)

lemma *right-plus-below-circ*: $x^\circ \ ; \ x \leq x^\circ$
by (*metis add-least-upper-bound circ-right-unfold-sub*)

lemma *circ-add-upper-bound*: $x \leq z^\circ \wedge y \leq z^\circ \longrightarrow x + y \leq z^\circ$
by (*metis add-least-upper-bound*)

lemma *circ-mult-upper-bound*: $x \leq z^\circ \wedge y \leq z^\circ \longrightarrow x \ ; \ y \leq z^\circ$
by (*metis mult-isotone circ-transitive-equal*)

lemma *circ-sub-dist*: $x^\circ \leq (x + y)^\circ$
by (*metis circ-add-sub circ-plus-one mult-left-one mult-right-sub-dist-add-left order-trans*)

lemma *circ-sub-dist-1*: $x \leq (x + y)^\circ$
by (*metis add-least-upper-bound circ-increasing*)

lemma *circ-sub-dist-2*: $x \ ; \ y \leq (x + y)^\circ$
by (*metis add-commutative circ-mult-upper-bound circ-sub-dist-1*)

— Theorem 1.20 and Theorem 14.23

lemma *circ-sub-dist-3*: $x^\circ \ ; \ y^\circ \leq (x + y)^\circ$
by (*metis add-commutative circ-mult-upper-bound circ-sub-dist*)

— Theorem 1 and Theorem 14

lemma *circ-isotone*: $x \leq y \longrightarrow x^\circ \leq y^\circ$
by (*metis circ-sub-dist less-eq-def*)

— Theorem 1.21 and Theorem 14.24

lemma *circ-add-2*: $(x + y)^\circ \leq (x^\circ \ ; \ y^\circ)^\circ$
by (*metis add-least-upper-bound circ-increasing circ-isotone circ-reflexive mult-isotone mult-left-one mult-right-one*)

lemma *circ-sup-one-left-unfold*: $1 \leq x \longrightarrow x \ ; \ x^\circ = x^\circ$
by (*metis antisym less-eq-def mult-left-one mult-right-sub-dist-add-left left-plus-below-circ*)

lemma *circ-sup-one-right-unfold*: $1 \leq x \longrightarrow x^\circ \ ; \ x = x^\circ$

by (*metis antisym less-eq-def mult-left-sub-dist-add-left mult-right-one right-plus-below-circ*)

— Theorem 1.23 and Theorem 14.26

lemma *circ-decompose-4*: $(x^\circ \ ; \ y^\circ)^\circ = x^\circ \ ; \ (y^\circ \ ; \ x^\circ)^\circ$
by (*metis add-associative add-commutative circ-add-1 circ-loop-fixpoint circ-plus-one circ-rtc-2 circ-transitive-equal mult-associative*)

— Theorem 1.22 and Theorem 14.25

lemma *circ-decompose-5*: $(x^\circ \ ; \ y^\circ)^\circ = (y^\circ \ ; \ x^\circ)^\circ$
by (*smt add-associative add-commutative add-left-zero circ-add-1 circ-decompose-4 mult-left-zero mult-right-one*)

lemma *circ-decompose-6*: $x^\circ \ ; \ (y \ ; \ x^\circ)^\circ = y^\circ \ ; \ (x \ ; \ y^\circ)^\circ$
by (*metis add-commutative circ-add-1*)

lemma *circ-decompose-7*: $(x + y)^\circ = x^\circ \ ; \ y^\circ \ ; \ (x + y)^\circ$
by (*metis circ-add-1 circ-decompose-6 circ-transitive-equal mult-associative*)

lemma *circ-decompose-8*: $(x + y)^\circ = (x + y)^\circ \ ; \ x^\circ \ ; \ y^\circ$
by (*metis antisym eq-refl mult-associative mult-isotone mult-right-one circ-mult-upper-bound circ-reflexive circ-sub-dist-3*)

lemma *circ-decompose-9*: $(x^\circ \ ; \ y^\circ)^\circ = x^\circ \ ; \ y^\circ \ ; \ (x^\circ \ ; \ y^\circ)^\circ$
by (*metis circ-decompose-4 mult-associative*)

lemma *circ-decompose-10*: $(x^\circ \ ; \ y^\circ)^\circ = (x^\circ \ ; \ y^\circ)^\circ \ ; \ x^\circ \ ; \ y^\circ$
by (*metis add-right-upper-bound circ-loop-fixpoint circ-reflexive circ-sup-one-right-unfold mult-associative order-trans*)

lemma *circ-back-loop-prefixpoint*: $(z \ ; \ y^\circ) \ ; \ y + z \leq z \ ; \ y^\circ$
by (*metis add-least-upper-bound circ-left-unfold mult-associative mult-left-sub-dist-add-left mult-right-isotone mult-right-one right-plus-below-circ*)

— Theorem 1 and Theorem 14

lemma *circ-loop-is-fixpoint*: *is-fixpoint* $(\lambda x \ . \ y \ ; \ x + z) \ (y^\circ \ ; \ z)$
by (*metis circ-loop-fixpoint is-fixpoint-def*)

— Theorem 14

lemma *circ-back-loop-is-prefixpoint*: *is-prefixpoint* $(\lambda x \ . \ x \ ; \ y + z) \ (z \ ; \ y^\circ)$
by (*metis circ-back-loop-prefixpoint is-prefixpoint-def*)

— Theorem 1.16 and Theorem 14.16

lemma *circ-circ-add*: $(1 + x)^\circ = x^{\circ\circ}$
by (*metis add-commutative circ-add-1 circ-decompose-4 circ-zero mult-right-one*)

— Theorem 14.14

lemma *circ-circ-mult-sub*: $x^\circ \ ; \ 1^\circ \leq x^{\circ\circ}$
by (*metis circ-increasing circ-isotone circ-mult-upper-bound circ-reflexive*)

— Theorem 14.9

lemma *left-plus-circ*: $(x \ ; \ x^\circ)^\circ = x^\circ$
by (*smt add-idempotent circ-add-1 circ-loop-fixpoint circ-transitive-equal mult-right-dist-add*)

— Theorem 1.10 and Theorem 14.10

lemma *right-plus-circ*: $(x^\circ \ ; \ x)^\circ = x^\circ$
by (*metis add-commutative circ-isotone circ-loop-fixpoint circ-plus-sub circ-sub-dist eq-iff left-plus-circ*)

lemma *circ-square*: $(x \ ; \ x)^\circ \leq x^\circ$
by (*metis circ-increasing circ-isotone left-plus-circ mult-right-isotone*)

— Theorem 1.19

lemma *circ-mult-sub-add*: $(x \ ; \ y)^\circ \leq (x + y)^\circ$
by (*metis add-left-upper-bound add-right-upper-bound circ-isotone circ-square mult-isotone order-trans*)

lemma *circ-add-mult-zero*: $x^\circ \ ; \ y = (x + y \ ; \ 0)^\circ \ ; \ y$
proof −
have $(x + y \ ; \ 0)^\circ \ ; \ y = x^\circ \ ; \ (1 + y \ ; \ 0) \ ; \ y$
by (*metis mult-zero-add-circ mult-zero-circ*)
also have ... $= x^\circ \ ; \ (y + y \ ; \ 0)$
by (*metis mult-associative mult-left-one mult-left-zero mult-right-dist-add*)
also have ... $= x^\circ \ ; \ y$
by (*metis add-commutative less-eq-def zero-right-mult-decreasing*)
finally show *?thesis*
by *metis*
qed

— Theorem 14.22

lemma *troeger-1*: $(x + y)^\circ = x^\circ \ ; \ (1 + y \ ; \ (x + y)^\circ)$
by (*metis circ-add-1 circ-left-unfold mult-associative*)

lemma *troeger-2*: $(x + y)^\circ \ ; \ z = x^\circ \ ; \ (y \ ; \ (x + y)^\circ \ ; \ z + z)$
by (*metis circ-add-1 circ-loop-fixpoint mult-associative*)

lemma *troeger-3*: $(x + y \ ; \ 0)^\circ = x^\circ \ ; \ (1 + y \ ; \ 0)$
by (*metis mult-zero-add-circ mult-zero-circ*)

lemma *circ-add-sub-add-one-1*: $x + y \leq x^\circ \ ; \ (1 + y)$
by (*smt add-associative add-commutative add-idempotent circ-increasing circ-loop-fixpoint less-eq-def mult-left-sub-dist-add-left mult-right-one*)

lemma *circ-add-sub-add-one-2*: $x^\circ \ ; \ (x + y) \leq x^\circ \ ; \ (1 + y)$
by (*metis circ-add-sub-add-one-1 circ-transitive-equal mult-associative mult-right-isotone*)

lemma *circ-add-sub-add-one*: $x \ ; \ x^\circ \ ; \ (x + y) \leq x \ ; \ x^\circ \ ; \ (1 + y)$
by (*metis circ-add-sub-add-one-2 mult-associative mult-right-isotone*)

lemma *circ-square-2*: $(x \ ; \ x)^\circ \ ; \ (x + 1) \leq x^\circ$
by (*metis add-least-upper-bound circ-increasing circ-mult-upper-bound circ-reflexive circ-square*)

— Theorem 1.25 and Theorem 14.28

lemma *circ-extra-circ*: $(y \ ; \ x^\circ)^\circ = (y \ ; \ y^\circ \ ; \ x^\circ)^\circ$
by (*smt add-commutative add-idempotent circ-add-1 circ-left-unfold mult-associative*)

— Theorem 14.15

lemma *circ-circ-sub-mult*: $1^\circ \ ; \ x^\circ \leq x^{\circ\circ}$
by (*metis circ-increasing circ-isotone circ-mult-upper-bound circ-reflexive*)

— Theorem 14.27

lemma *circ-decompose-11*: $(x^\circ \ ; \ y^\circ)^\circ = (x^\circ \ ; \ y^\circ)^\circ \ ; \ x^\circ$
by (*smt circ-add-1 circ-circ-add circ-decompose-4 circ-decompose-8 circ-rtc-2 circ-transitive-equal mult-associative*)

— Theorem 14.20

lemma *circ-mult-below-circ-circ*: $(x \ ; \ y)^\circ \leq (x^\circ \ ; \ y)^\circ \ ; \ x^\circ$
by (*metis circ-increasing circ-isotone circ-reflexive dual-order.trans mult-left-isotone mult-right-isotone mult-right-one*)

— Theorem 14 counterexamples

lemma *circ-right-unfold*: $1 + x^\circ \ ; \ x = x^\circ$ **nitpick** [*expect=genuine*] **oops**
lemma *circ-mult*: $1 + x \ ; \ (y \ ; \ x)^\circ \ ; \ y = (x \ ; \ y)^\circ$ **nitpick** [*expect=genuine*] **oops**
lemma *circ-slide*: $(x \ ; \ y)^\circ \ ; \ x = x \ ; \ (y \ ; \ x)^\circ$ **nitpick** [*expect=genuine*] **oops**
lemma *circ-plus-same*: $x^\circ \ ; \ x = x \ ; \ x^\circ$ **nitpick** [*expect=genuine*] **oops**
lemma $1^\circ \ ; \ x^\circ \leq x^\circ \ ; \ 1^\circ$ **nitpick** [*expect=genuine,card=7*] **oops**
lemma *circ-circ-mult-1*: $x^\circ \ ; \ 1^\circ = x^{\circ\circ}$ **nitpick** [*expect=genuine,card=7*] **oops**
lemma $x^\circ \ ; \ 1^\circ \leq 1^\circ \ ; \ x^\circ$ **nitpick** [*expect=genuine,card=7*] **oops**
lemma *circ-circ-mult*: $1^\circ \ ; \ x^\circ = x^{\circ\circ}$ **nitpick** [*expect=genuine,card=7*] **oops**
lemma *circ-add*: $(x^\circ \ ; \ y)^\circ \ ; \ x^\circ = (x + y)^\circ$ **nitpick** [*expect=genuine,card=8*] **oops**
lemma *circ-unfold-sum*: $(x + y)^\circ = x^\circ + x^\circ \ ; \ y \ ; \ (x + y)^\circ$ **nitpick** [*expect=genuine,card=7*] **oops**

lemma *mult-zero-add-circ-2*: $(x + y\ ;\ 0)^\circ = x^\circ + x^\circ\ ;\ y\ ;\ 0$ **nitpick** [*expect=genuine,card=7*] **oops**
lemma *sub-mult-one-circ*: $x\ ;\ 1^\circ \leq 1^\circ\ ;\ x$ **nitpick** [*expect=genuine*] **oops**
lemma *circ-back-loop-fixpoint*: $(z\ ;\ y^\circ)\ ;\ y + z = z\ ;\ y^\circ$ **nitpick** [*expect=genuine*] **oops**
lemma *circ-back-loop-is-fixpoint*: *is-fixpoint* $(\lambda x\ .\ x\ ;\ y + z)\ (z\ ;\ y^\circ)$ **nitpick** [*expect=genuine*] **oops**
lemma $x^\circ\ ;\ y^\circ \leq (x^\circ\ ;\ y)^\circ\ ;\ x^\circ$ **nitpick** [*expect=genuine,card=7*] **oops**

end

class *bounded-left-conway-semiring* = *bounded-idempotent-left-semiring* + *left-conway-semiring*

begin

lemma *circ-top-1*: $T^\circ = T$
by (*metis add-right-top antisym circ-left-unfold mult-left-sub-dist-add-left mult-right-one top-greatest*)

lemma *circ-right-top-1*: $x^\circ\ ;\ T = T$
by (*metis add-right-top circ-loop-fixpoint*)

lemma *circ-left-top-1*: $T\ ;\ x^\circ = T$
by (*metis antisym circ-plus-one mult-left-sub-dist-add-left mult-right-one top-greatest*)

lemma *mult-top-circ-1*: $(x\ ;\ T)^\circ = 1 + x\ ;\ T$
by (*metis circ-left-top-1 circ-left-unfold mult-associative*)

end

class *left-zero-conway-semiring* = *idempotent-left-zero-semiring* + *left-conway-semiring*

begin

lemma *mult-zero-add-circ-2*: $(x + y\ ;\ 0)^\circ = x^\circ + x^\circ\ ;\ y\ ;\ 0$
by (*metis mult-associative mult-left-dist-add mult-right-one troeger-3*)

lemma *circ-unfold-sum*: $(x + y)^\circ = x^\circ + x^\circ\ ;\ y\ ;\ (x + y)^\circ$
by (*metis mult-associative mult-left-dist-add mult-right-one troeger-1*)

end

class *left-conway-semiring-1* = *left-conway-semiring* +
assumes *circ-right-slide*: $x\ ;\ (y\ ;\ x)^\circ \leq (x\ ;\ y)^\circ\ ;\ x$

begin

— Theorem 1.26

lemma *circ-slide-1*: $x\ ;\ (y\ ;\ x)^\circ = (x\ ;\ y)^\circ\ ;\ x$
by (*metis antisym circ-left-slide circ-right-slide*)

— Theorem 1.9

lemma *circ-right-unfold-1*: $1 + x^\circ\ ;\ x = x^\circ$
by (*metis circ-left-unfold circ-slide-1 mult-left-one mult-right-one*)

lemma *circ-mult-1*: $(x\ ;\ y)^\circ = 1 + x\ ;\ (y\ ;\ x)^\circ\ ;\ y$
by (*metis circ-left-unfold circ-slide-1 mult-associative*)

lemma *circ-add-9*: $(x + y)^\circ = (x^\circ\ ;\ y)^\circ\ ;\ x^\circ$
by (*metis circ-add-1 circ-slide-1*)

— Theorem 1.7

lemma *circ-plus-same*: $x^\circ\ ;\ x = x\ ;\ x^\circ$
by (*metis circ-slide-1 mult-left-one mult-right-one*)

lemma *circ-decompose-12*: $x^\circ\ ;\ y^\circ \leq (x^\circ\ ;\ y)^\circ\ ;\ x^\circ$
by (*metis circ-add-9 circ-sub-dist-3*)

end

class *left-zero-conway-semiring-1* = *left-zero-conway-semiring* + *left-conway-semiring-1*

begin

lemma *circ-back-loop-fixpoint*: $(z\ ;\ y^\circ)\ ;\ y + z = z\ ;\ y^\circ$
by (*metis add-commutative circ-left-unfold circ-plus-same mult-associative mult-left-dist-add mult-right-one*)

— Theorem 1

lemma *circ-back-loop-is-fixpoint*: *is-fixpoint* $(\lambda x\ .\ x\ ;\ y + z)\ (z\ ;\ y^\circ)$
by (*metis circ-back-loop-fixpoint is-fixpoint-def*)

lemma *circ-elimination*: $x\ ;\ y = 0 \longrightarrow x\ ;\ y^\circ \leq x$
by (*metis add-left-zero circ-back-loop-fixpoint circ-plus-same mult-associative mult-left-zero order-refl*)

end

class *itering-1* = *left-conway-semiring-1* +
assumes *circ-simulate*: $z\ ;\ x \leq y\ ;\ z \longrightarrow z\ ;\ x^\circ \leq y^\circ\ ;\ z$

begin

— Theorem 1.15

lemma *circ-circ-mult*: $1^\circ\ ;\ x^\circ = x^{\circ\circ}$
by (*metis antisym circ-circ-add circ-reflexive circ-simulate circ-sub-dist-3 circ-sup-one-left-unfold circ-transitive-equal mult-left-one order-refl*)

lemma *sub-mult-one-circ*: $x\ ;\ 1^\circ \leq 1^\circ\ ;\ x$
by (*metis circ-simulate mult-left-one mult-right-one order-refl*)

lemma *circ-import*: $p \leq p\ ;\ p \wedge p \leq 1 \wedge p\ ;\ x \leq x\ ;\ p \longrightarrow p\ ;\ x^\circ = p\ ;\ (p\ ;\ x)^\circ$
apply *rule*
apply (*rule antisym*)
apply (*smt antisym circ-simulate circ-slide-1 mult-associative mult-right-isotone mult-right-one order-refl test-preserves-equation*)
apply (*metis circ-isotone mult-left-isotone mult-left-one mult-right-isotone*)
done

end

class *itering-2* = *left-conway-semiring-1* +
assumes *circ-simulate-right*: $z\ ;\ x \leq y\ ;\ z + w \longrightarrow z\ ;\ x^\circ \leq y^\circ\ ;\ (z + w\ ;\ x^\circ)$
assumes *circ-simulate-left*: $x\ ;\ z \leq z\ ;\ y + w \longrightarrow x^\circ\ ;\ z \leq (z + x^\circ\ ;\ w)\ ;\ y^\circ$

begin

subclass *itering-1*
apply *unfold-locales*
apply (*metis add-right-zero circ-simulate-right mult-left-zero*)
done

lemma *circ-simulate-left-1*: $x\ ;\ z \leq z\ ;\ y \longrightarrow x^\circ\ ;\ z \leq z\ ;\ y^\circ + x^\circ\ ;\ 0$
by (*metis add-right-zero circ-simulate-left mult-associative mult-left-zero mult-right-dist-add*)

lemma *circ-separate-1*: $y\ ;\ x \leq x\ ;\ y \longrightarrow (x + y)^\circ = x^\circ\ ;\ y^\circ$
proof −
have $y\ ;\ x \leq x\ ;\ y \longrightarrow y^\circ\ ;\ x\ ;\ y^\circ \leq x\ ;\ y^\circ + y^\circ\ ;\ 0$
by (*smt circ-simulate-left-1 circ-transitive-equal mult-associative mult-left-isotone mult-left-zero mult-right-dist-add*)
thus *?thesis*
by (*smt add-commutative circ-add-1 circ-simulate-right circ-sub-dist-3 less-eq-def mult-associative mult-left-zero zero-right-mult-decreasing*)
qed

— Theorem 1.14

lemma *circ-circ-mult-1*: $x^\circ\ ;\ 1^\circ = x^{\circ\circ}$
by (*metis add-commutative circ-circ-add circ-separate-1 mult-left-one mult-right-one order-refl*)

end

class *itering-3* = *itering-2* + *left-zero-conway-semiring-1*

begin

lemma *circ-simulate-1*: $y\ ;\ x \leq x\ ;\ y \longrightarrow y^\circ\ ;\ x^\circ \leq x^\circ\ ;\ y^\circ$
by (*smt add-associative add-right-zero circ-loop-fixpoint circ-simulate circ-simulate-left-1 mult-associative mult-left-zero mult-zero-add-circ-2*)

— Theorem 4

lemma *atomicity-refinement*: $s = s\ ;\ q \wedge x = q\ ;\ x \wedge q\ ;\ b = 0 \wedge r\ ;\ b \leq b\ ;\ r \wedge r\ ;\ l \leq l\ ;\ r \wedge x\ ;\ l \leq l\ ;\ x \wedge b\ ;\ l \leq l\ ;\ b \wedge q\ ;\ l \leq l\ ;\ q \wedge r^\circ\ ;\ q \leq q\ ;\ r^\circ \wedge q \leq 1 \longrightarrow s\ ;\ (x + b + r + l)^\circ\ ;\ q \leq s\ ;\ (x\ ;\ b^\circ\ ;\ q + r + l)^\circ$
proof
assume *1*: $s = s\ ;\ q \wedge x = q\ ;\ x \wedge q\ ;\ b = 0 \wedge r\ ;\ b \leq b\ ;\ r \wedge r\ ;\ l \leq l\ ;\ r \wedge x\ ;\ l \leq l\ ;\ x \wedge b\ ;\ l \leq l\ ;\ b \wedge q\ ;\ l \leq l\ ;\ q \wedge r^\circ\ ;\ q \leq q\ ;\ r^\circ \wedge q \leq 1$
hence $s\ ;\ (x + b + r + l)^\circ\ ;\ q = s\ ;\ l^\circ\ ;\ (x + b + r)^\circ\ ;\ q$
by (*smt add-commutative add-least-upper-bound circ-separate-1 mult-associative mult-left-sub-dist-add-right mult-right-dist-add order-trans*)
also have ... $= s\ ;\ l^\circ\ ;\ b^\circ\ ;\ r^\circ\ ;\ q\ ;\ (x\ ;\ b^\circ\ ;\ r^\circ\ ;\ q)^\circ$ **using** *1*
by (*smt add-associative add-commutative circ-add-1 circ-separate-1 circ-slide-1 mult-associative*)
also have ... $\leq s\ ;\ l^\circ\ ;\ b^\circ\ ;\ r^\circ\ ;\ q\ ;\ (x\ ;\ b^\circ\ ;\ q\ ;\ r^\circ)^\circ$ **using** *1*
by (*metis circ-isotone mult-associative mult-right-isotone*)
also have ... $\leq s\ ;\ q\ ;\ l^\circ\ ;\ b^\circ\ ;\ r^\circ\ ;\ (x\ ;\ b^\circ\ ;\ q\ ;\ r^\circ)^\circ$ **using** *1*
by (*metis mult-left-isotone mult-right-isotone mult-right-one*)
also have ... $\leq s\ ;\ l^\circ\ ;\ q\ ;\ b^\circ\ ;\ r^\circ\ ;\ (x\ ;\ b^\circ\ ;\ q\ ;\ r^\circ)^\circ$ **using** *1*
by (*metis circ-simulate mult-associative mult-left-isotone mult-right-isotone*)
also have ... $\leq s\ ;\ l^\circ\ ;\ r^\circ\ ;\ (x\ ;\ b^\circ\ ;\ q\ ;\ r^\circ)^\circ$ **using** *1*
by (*metis add-left-zero circ-back-loop-fixpoint circ-plus-same mult-associative mult-left-zero mult-left-isotone mult-right-isotone mult-right-one*)
also have ... $\leq s\ ;\ (x\ ;\ b^\circ\ ;\ q + r + l)^\circ$ **using** *1*
by (*metis add-commutative circ-add-1 circ-sub-dist-3 mult-associative mult-right-isotone*)
finally show $s\ ;\ (x + b + r + l)^\circ\ ;\ q \leq s\ ;\ (x\ ;\ b^\circ\ ;\ q + r + l)^\circ$ **.**
qed

end

class *itering* = *idempotent-left-zero-semiring* + *circ* +
assumes *circ-add*: $(x + y)^\circ = (x^\circ\ ;\ y)^\circ\ ;\ x^\circ$
assumes *circ-mult*: $(x\ ;\ y)^\circ = 1 + x\ ;\ (y\ ;\ x)^\circ\ ;\ y$
assumes *circ-simulate-right-plus*: $z\ ;\ x \leq y\ ;\ y^\circ\ ;\ z + w \longrightarrow z\ ;\ x^\circ \leq y^\circ\ ;\ (z + w\ ;\ x^\circ)$
assumes *circ-simulate-left-plus*: $x\ ;\ z \leq z\ ;\ y^\circ + w \longrightarrow x^\circ\ ;\ z \leq (z + x^\circ\ ;\ w)\ ;\ y^\circ$

begin

lemma *circ-right-unfold*: $1 + x^\circ\ ;\ x = x^\circ$
by (*metis circ-mult mult-left-one mult-right-one*)

lemma *circ-slide*: $x\ ;\ (y\ ;\ x)^\circ = (x\ ;\ y)^\circ\ ;\ x$
by (*smt2 circ-mult mult-associative mult-left-dist-add mult-left-one mult-right-dist-add mult-right-one order-refl*)

— Theorem 50.6

subclass *itering-3*
apply *unfold-locales*
apply (*metis circ-mult mult-left-one mult-right-one*) — Theorem 1.8
apply (*metis circ-slide order-refl*)
apply (*metis circ-add circ-slide*)
apply (*metis circ-slide order-refl*)
apply (*metis add-left-isotone circ-right-unfold mult-left-isotone mult-left-sub-dist-add-left mult-right-one order-trans circ-simulate-right-plus*)
apply (*metis add-commutative add-left-upper-bound add-right-isotone circ-mult mult-right-isotone mult-right-one order-trans circ-simulate-left-plus*)
done

lemma *circ-simulate-right-plus-1*: $z\ ;\ x \leq y\ ;\ y^\circ\ ;\ z \longrightarrow z\ ;\ x^\circ \leq y^\circ\ ;\ z$
by (*metis add-right-zero circ-simulate-right-plus mult-left-zero*)

lemma *circ-simulate-left-plus-1*: $x \ ; \ z \leq z \ ; \ y^\circ \longrightarrow x^\circ \ ; \ z \leq z \ ; \ y^\circ + x^\circ \ ; \ 0$
by (*smt add-right-zero circ-simulate-left-plus mult-associative mult-left-zero mult-right-dist-add*)

lemma *circ-simulate-2*: $y \ ; \ x^\circ \leq x^\circ \ ; \ y^\circ \longleftrightarrow y^\circ \ ; \ x^\circ \leq x^\circ \ ; \ y^\circ$
apply (*rule iffI*)
apply (*smt add-associative add-right-zero circ-loop-fixpoint circ-simulate-left-plus-1 mult-associative mult-left-zero mult-zero-add-circ-2*)
apply (*metis circ-increasing mult-left-isotone order-trans*)
done

lemma *circ-simulate-absorb*: $y \ ; \ x \leq x \longrightarrow y^\circ \ ; \ x \leq x + y^\circ \ ; \ 0$
by (*metis circ-simulate-left-plus-1 circ-zero mult-right-one*)

lemma *circ-simulate-3*: $y \ ; \ x^\circ \leq x^\circ \longrightarrow y^\circ \ ; \ x^\circ \leq x^\circ \ ; \ y^\circ$
by (*metis add-least-upper-bound circ-reflexive circ-simulate-2 less-eq-def mult-right-isotone mult-right-one*)

lemma *circ-separate-mult-1*: $y \ ; \ x \leq x \ ; \ y \longrightarrow (x \ ; \ y)^\circ \leq x^\circ \ ; \ y^\circ$
by (*metis circ-mult-sub-add circ-separate-1*)

— Theorem 1.24

lemma *circ-separate-unfold*: $(y \ ; \ x^\circ)^\circ = y^\circ + y^\circ \ ; \ y \ ; \ x \ ; \ x^\circ \ ; \ (y \ ; \ x^\circ)^\circ$
by (*smt add-commutative circ-add circ-left-unfold circ-loop-fixpoint mult-associative mult-left-dist-add mult-right-one*)

— Theorem 3

lemma *separation*: $y \ ; \ x \leq x \ ; \ y^\circ \longrightarrow (x + y)^\circ = x^\circ \ ; \ y^\circ$
proof −
have $y \ ; \ x \leq x \ ; \ y^\circ \longrightarrow y^\circ \ ; \ x \ ; \ y^\circ \leq x \ ; \ y^\circ + y^\circ \ ; \ 0$
by (*smt circ-simulate-left-plus-1 circ-transitive-equal mult-associative mult-left-isotone mult-left-zero mult-right-dist-add*)
thus *?thesis*
by (*smt add-commutative circ-add-1 circ-simulate-right circ-sub-dist-3 less-eq-def mult-associative mult-left-zero zero-right-mult-decreasing*)
qed

— Theorem 3

lemma *simulation*: $y \ ; \ x \leq x \ ; \ y^\circ \longrightarrow y^\circ \ ; \ x^\circ \leq x^\circ \ ; \ y^\circ$
by (*metis add-right-upper-bound circ-isotone circ-mult-upper-bound circ-sub-dist separation*)

— Theorem 3

lemma *circ-simulate-4*: $y \ ; \ x \leq x \ ; \ x^\circ \ ; \ (1 + y) \longrightarrow y^\circ \ ; \ x^\circ \leq x^\circ \ ; \ y^\circ$
proof
assume $y \ ; \ x \leq x \ ; \ x^\circ \ ; \ (1 + y)$
hence $(1 + y) \ ; \ x \leq x \ ; \ x^\circ \ ; \ (1 + y)$
by (*smt add-associative add-commutative add-left-upper-bound circ-back-loop-fixpoint less-eq-def mult-left-dist-add mult-left-one mult-right-dist-add mult-right-one*)
hence $y \ ; \ x^\circ \leq x^\circ \ ; \ y^\circ$
by (*metis circ-add-upper-bound circ-increasing circ-reflexive circ-simulate-right-plus-1 mult-right-isotone mult-right-sub-dist-add-right order-trans*)
thus $y^\circ \ ; \ x^\circ \leq x^\circ \ ; \ y^\circ$
by (*metis circ-simulate-2*)
qed

lemma *circ-simulate-5*: $y \ ; \ x \leq x \ ; \ x^\circ \ ; \ (x + y) \longrightarrow y^\circ \ ; \ x^\circ \leq x^\circ \ ; \ y^\circ$
by (*metis circ-add-sub-add-one circ-simulate-4 order-trans*)

lemma *circ-simulate-6*: $y \ ; \ x \leq x \ ; \ (x + y) \longrightarrow y^\circ \ ; \ x^\circ \leq x^\circ \ ; \ y^\circ$
by (*metis add-commutative circ-back-loop-fixpoint circ-simulate-5 mult-right-sub-dist-add-left order-trans*)

— Theorem 3

lemma *circ-separate-4*: $y \ ; \ x \leq x \ ; \ x^\circ \ ; \ (1 + y) \longrightarrow (x + y)^\circ = x^\circ \ ; \ y^\circ$
proof
assume *1*: $y \ ; \ x \leq x \ ; \ x^\circ \ ; \ (1 + y)$
hence $y \ ; \ x \ ; \ x^\circ \leq x \ ; \ x^\circ + x \ ; \ x^\circ \ ; \ y \ ; \ x^\circ$
by (*smt circ-transitive-equal less-eq-def mult-associative mult-left-dist-add mult-right-dist-add mult-right-one*)
also have $\ldots \leq x \ ; \ x^\circ + x \ ; \ x^\circ \ ; \ x^\circ \ ; \ y^\circ$ **using** *1*

by (*metis add-right-isotone circ-simulate-2 circ-simulate-4 mult-associative mult-right-isotone*)
finally have $y\ ;\ x\ ;\ x^\circ \leq x\ ;\ x^\circ\ ;\ y^\circ$
by (*metis circ-reflexive circ-transitive-equal less-eq-def mult-associative mult-right-isotone mult-right-one*)
hence $y^\circ\ ;\ (y^\circ\ ;\ x)^\circ \leq x^\circ\ ;\ (y^\circ + y^\circ\ ;\ 0\ ;\ (y^\circ\ ;\ x)^\circ)$
by (*smt add-right-upper-bound circ-back-loop-fixpoint circ-simulate-left-plus-1 circ-simulate-right-plus circ-transitive-equal mult-associative order-trans*)
thus $(x + y)^\circ = x^\circ\ ;\ y^\circ$
by (*smt add-commutative antisym circ-add-1 circ-slide circ-sub-dist-3 circ-transitive-equal less-eq-def mult-associative mult-left-zero mult-right-sub-dist-add-right zero-right-mult-decreasing*)
qed

lemma *circ-separate-5*: $y\ ;\ x \leq x\ ;\ x^\circ\ ;\ (x + y) \longrightarrow (x + y)^\circ = x^\circ\ ;\ y^\circ$
by (*metis circ-add-sub-add-one circ-separate-4 order-trans*)

lemma *circ-separate-6*: $y\ ;\ x \leq x\ ;\ (x + y) \longrightarrow (x + y)^\circ = x^\circ\ ;\ y^\circ$
by (*metis add-commutative circ-back-loop-fixpoint circ-separate-5 mult-right-sub-dist-add-left order-trans*)

end

class *bounded-itering* = *bounded-idempotent-left-zero-semiring* + *itering*

begin

— Theorem 1

lemma *circ-top*: $T^\circ = T$
by (*metis add-right-top antisym circ-left-unfold mult-left-sub-dist-add-left mult-right-one top-greatest*)

lemma *circ-right-top*: $x^\circ\ ;\ T = T$
by (*metis add-right-top circ-loop-fixpoint*)

lemma *circ-left-top*: $T\ ;\ x^\circ = T$
by (*metis add-right-top circ-add circ-right-top circ-top*)

lemma *mult-top-circ*: $(x\ ;\ T)^\circ = 1 + x\ ;\ T$
by (*metis circ-left-top circ-mult mult-associative*)

— Theorem 1 counterexamples

lemma $1 = x^\circ$ **nitpick** [*expect=genuine*] **oops**
lemma $x = x^\circ$ **nitpick** [*expect=genuine*] **oops**
lemma $x = x\ ;\ x^\circ$ **nitpick** [*expect=genuine*] **oops**
lemma $x\ ;\ x^\circ = x^\circ$ **nitpick** [*expect=genuine*] **oops**
lemma $x^\circ = x^{\circ\circ}$ **nitpick** [*expect=genuine*] **oops**
lemma $(x\ ;\ y)^\circ = (x + y)^\circ$ **nitpick** [*expect=genuine*] **oops**
lemma $x^\circ\ ;\ y^\circ = (x + y)^\circ$ **nitpick** [*expect=genuine,card=6*] **oops**
lemma $(x + y)^\circ = (x^\circ\ ;\ y^\circ)^\circ$ **nitpick** [*expect=genuine*] **oops**
lemma $1 = 1^\circ$ **nitpick** [*expect=genuine*] **oops**

lemma $1 = (x\ ;\ 0)^\circ$ **nitpick** [*expect=genuine*] **oops**
lemma $1 + x\ ;\ 0 = x^\circ$ **nitpick** [*expect=genuine*] **oops**
lemma $x^\circ = x^\circ\ ;\ 1^\circ$ **nitpick** [*expect=genuine*] **oops**
lemma $z + y\ ;\ x = x \longrightarrow y^\circ\ ;\ z \leq x$ **nitpick** [*expect=genuine*] **oops**
lemma $y\ ;\ x = x \longrightarrow y^\circ\ ;\ x \leq x$ **nitpick** [*expect=genuine*] **oops**
lemma $z + x\ ;\ y = x \longrightarrow z\ ;\ y^\circ \leq x$ **nitpick** [*expect=genuine*] **oops**
lemma $x\ ;\ y = x \longrightarrow x\ ;\ y^\circ \leq x$ **nitpick** [*expect=genuine*] **oops**
lemma $x = z + y\ ;\ x \longrightarrow x \leq y^\circ\ ;\ z$ **nitpick** [*expect=genuine*] **oops**
lemma $x = y\ ;\ x \longrightarrow x \leq y^\circ$ **nitpick** [*expect=genuine*] **oops**
lemma $x\ ;\ z = z\ ;\ y \longrightarrow x^\circ\ ;\ z \leq z\ ;\ y^\circ$ **nitpick** [*expect=genuine*] **oops**

lemma $x^\circ = (x\ ;\ x)^\circ\ ;\ (x + 1)$ **oops**
lemma $y^\circ\ ;\ x^\circ \leq x^\circ\ ;\ y^\circ \longrightarrow (x + y)^\circ = x^\circ\ ;\ y^\circ$ **oops**
lemma $y\ ;\ x \leq (1 + x)\ ;\ y^\circ \longrightarrow (x + y)^\circ = x^\circ\ ;\ y^\circ$ **oops**
lemma $y\ ;\ x \leq x \longrightarrow y^\circ\ ;\ x \leq 1^\circ\ ;\ x$ **oops**

end

class *left-conway-semiring-L* = *left-conway-semiring* + *L* +
assumes *one-circ-mult-split*: $1^\circ\ ;\ x = L + x$

assumes *L-split-add*: $x \ ; \ (y + L) \leq x \ ; \ y + L$

begin

lemma *L-def*: $L = 1^\circ \ ; \ 0$
by (*metis add-right-zero one-circ-mult-split*)

lemma *one-circ-split*: $1^\circ = L + 1$
by (*metis mult-right-one one-circ-mult-split*)

lemma *one-circ-circ-split*: $1^{\circ\circ} = L + 1$
by (*metis circ-one one-circ-split*)

lemma *sub-mult-one-circ*: $x \ ; \ 1^\circ \leq 1^\circ \ ; \ x$
by (*metis L-split-add add-commutative mult-right-one one-circ-mult-split*)

lemma *one-circ-mult-split-2*: $1^\circ \ ; \ x = x \ ; \ 1^\circ + L$
by (*smt add-associative add-commutative add-least-upper-bound circ-back-loop-prefixpoint less-eq-def one-circ-mult-split sub-mult-one-circ*)

lemma *sub-mult-one-circ-split*: $x \ ; \ 1^\circ \leq x + L$
by (*metis add-commutative one-circ-mult-split sub-mult-one-circ*)

lemma *sub-mult-one-circ-split-2*: $x \ ; \ 1^\circ \leq x + 1^\circ$
by (*metis L-def add-right-isotone order-trans sub-mult-one-circ-split zero-right-mult-decreasing*)

lemma *L-split*: $x \ ; \ L \leq x \ ; \ 0 + L$
by (*metis L-split-add add-left-zero*)

lemma *L-left-zero*: $L \ ; \ x = L$
by (*metis L-def mult-associative mult-left-zero*)

lemma *one-circ-L*: $1^\circ \ ; \ L = L$
by (*metis L-def circ-transitive-equal mult-associative*)

lemma *mult-L-circ*: $(x \ ; \ L)^\circ = 1 + x \ ; \ L$
by (*metis L-left-zero circ-left-unfold mult-associative*)

lemma *mult-L-circ-mult*: $(x \ ; \ L)^\circ \ ; \ y = y + x \ ; \ L$
by (*metis L-left-zero mult-L-circ mult-associative mult-left-one mult-right-dist-add*)

lemma *circ-L*: $L^\circ = L + 1$
by (*metis L-left-zero add-commutative circ-left-unfold*)

lemma *L-below-one-circ*: $L \leq 1^\circ$
by (*metis L-def zero-right-mult-decreasing*)

lemma *circ-circ-mult-1*: $x^\circ \ ; \ 1^\circ = x^{\circ\circ}$
by (*metis L-left-zero add-commutative circ-add-1 circ-circ-add mult-zero-circ one-circ-split*)

lemma *circ-circ-mult*: $1^\circ \ ; \ x^\circ = x^{\circ\circ}$
by (*metis antisym circ-circ-mult-1 circ-circ-sub-mult sub-mult-one-circ*)

lemma *circ-circ-split*: $x^{\circ\circ} = L + x^\circ$
by (*metis circ-circ-mult one-circ-mult-split*)

lemma *circ-add-6*: $L + (x + y)^\circ = (x^\circ \ ; \ y^\circ)^\circ$
by (*metis add-associative add-commutative circ-add-1 circ-circ-add circ-circ-split circ-decompose-4*)

end

class *itering-L* = *itering* + *L* +
assumes *L-def*: $L = 1^\circ \ ; \ 0$

begin

lemma *one-circ-split*: $1^\circ = L + 1$
by (*metis L-def add-commutative antisym circ-add-upper-bound circ-reflexive circ-simulate-absorb mult-right-one order-refl zero-right-mult-decreasing*)

lemma *one-circ-mult-split*: $1^\circ \; ; \; x = L + x$
by (*metis L-def add-commutative circ-loop-fixpoint mult-associative mult-left-zero mult-zero-circ one-circ-split*)

lemma *sub-mult-one-circ-split*: $x \; ; \; 1^\circ \leq x + L$
by (*metis add-commutative one-circ-mult-split sub-mult-one-circ*)

lemma *sub-mult-one-circ-split-2*: $x \; ; \; 1^\circ \leq x + 1^\circ$
by (*metis L-def add-right-isotone order-trans sub-mult-one-circ-split zero-right-mult-decreasing*)

lemma *L-split*: $x \; ; \; L \leq x \; ; \; 0 + L$
by (*smt L-def mult-associative mult-left-isotone mult-right-dist-add sub-mult-one-circ-split-2*)

subclass *left-conway-semiring-L*
apply *unfold-locales*
apply (*metis L-def add-commutative circ-loop-fixpoint mult-associative mult-left-zero mult-zero-circ one-circ-split*)
apply (*metis add-commutative mult-associative mult-left-isotone one-circ-mult-split sub-mult-one-circ*)
done

lemma *circ-left-induct-mult-L*: $L \leq x \longrightarrow x \; ; \; y \leq x \longrightarrow x \; ; \; y^\circ \leq x$
by (*metis circ-one circ-simulate less-eq-def one-circ-mult-split*)

lemma *circ-left-induct-mult-iff-L*: $L \leq x \longrightarrow x \; ; \; y \leq x \longleftrightarrow x \; ; \; y^\circ \leq x$
by (*smt add-least-upper-bound circ-back-loop-fixpoint circ-left-induct-mult-L less-eq-def*)

lemma *circ-left-induct-L*: $L \leq x \longrightarrow x \; ; \; y + z \leq x \longrightarrow z \; ; \; y^\circ \leq x$
by (*metis add-least-upper-bound circ-left-induct-mult-L less-eq-def mult-right-dist-add*)

end

end

6 KleeneAlgebra

theory *KleeneAlgebra*

imports *Itering*

begin

class *star* =
fixes *star* :: $'a \Rightarrow 'a$ ($-^\star$ [100] 100)

class *left-kleene-algebra* = *idempotent-left-semiring* + *star* +
assumes *star-left-unfold* : 1 + y ; $y^\star \leq y^\star$
assumes *star-left-induct* : z + y ; $x \leq x \longrightarrow y^\star$; $z \leq x$

begin

lemma *star-left-unfold-equal*: 1 + x ; $x^\star = x^\star$
by (*smt add-right-isotone antisym mult-right-isotone mult-right-one star-left-induct star-left-unfold*)

lemma *star-left-slide*: (x ; y)* ; $x \leq x$; (y ; x)*
by (*metis mult-associative mult-left-sub-dist-add mult-right-one star-left-induct star-left-unfold-equal*)

lemma *star-isotone*: $x \leq y \longrightarrow x^\star \leq y^\star$
by (*metis add-right-isotone mult-left-isotone order-trans star-left-unfold mult-right-one star-left-induct*)

lemma *star-add-1-sub*: $x^\star$; (y ; $x^\star$)$^\star \leq (x + y)^\star$
proof −
have $x^\star$; (y ; $x^\star$)$^\star \leq x^\star$; (y ; $(x + y)^\star$)*
by (*smt add-left-upper-bound mult-right-isotone star-isotone*)
also have ... $\leq x^\star$; ((x + y) ; $(x + y)^\star$)*
by (*smt add-right-upper-bound mult-left-isotone mult-right-isotone star-isotone*)
also have ... $\leq x^\star$; $(x + y)^{\star\star}$
by (*smt add-least-upper-bound mult-right-isotone star-isotone star-left-unfold*)
also have ... $\leq (x + y)^\star$; $(x + y)^{\star\star}$
by (*smt add-left-upper-bound mult-left-isotone star-isotone*)
also have ... $\leq (x + y)^\star$
by (*smt add-least-upper-bound mult-right-one star-left-induct star-left-unfold*)
finally show $x^\star$; (y ; $x^\star$)$^\star \leq (x + y)^\star$
by *smt*
qed

lemma *star-add-1*: $(x + y)^\star = x^\star$; (y ; $x^\star$)*
apply (*rule antisym*)
apply (*smt add-least-upper-bound add-left-upper-bound add-right-upper-bound mult-associative mult-left-one mult-right-dist-add mult-right-one star-left-induct star-left-unfold-equal*)
apply (*smt star-add-1-sub*)
done

end

— Theorem 50.1

sublocale *left-kleene-algebra* < *star*!: *left-conway-semiring* **where** *circ* = *star*
apply *unfold-locales*
apply (*metis star-left-unfold-equal*)
apply (*metis star-left-slide*)
apply (*metis star-add-1*)
done

context *left-kleene-algebra*

begin

— Many lemmas in this class are taken from Georg Struth's Isabelle theories.

lemma *star-sub-one*: $x \leq 1 \longrightarrow x^\star = 1$
by (*metis add-right-isotone eq-iff less-eq-def mult-right-one star.circ-plus-one star-left-induct*)

lemma *star-one*: $1^\star = 1$
by (*metis eq-iff star-sub-one*)

lemma *star-left-induct-mult*: $x\ ;\ y \leq y \longrightarrow x^\star\ ;\ y \leq y$
by (*metis add-commutative less-eq-def order-refl star-left-induct*)

lemma *star-left-induct-mult-iff*: $x\ ;\ y \leq y \longleftrightarrow x^\star\ ;\ y \leq y$
by (*metis mult-associative mult-left-isotone mult-left-one mult-right-isotone order-trans star-left-induct-mult star.circ-reflexive star.left-plus-below-circ*)

lemma *star-involutive*: $x^\star = x^{\star\star}$
by (*smt antisym less-eq-def mult-left-sub-dist-add-left mult-right-one star-left-induct star.circ-plus-one star.left-plus-below-circ star.circ-transitive-equal*)

lemma *star-sup-one*: $(1 + x)^\star = x^\star$
by (*metis star.circ-circ-add star-involutive*)

lemma *star-left-induct-equal*: $z + x\ ;\ y = y \longrightarrow x^\star\ ;\ z \leq y$
by (*metis order-refl star-left-induct*)

lemma *star-left-induct-mult-equal*: $x\ ;\ y = y \longrightarrow x^\star\ ;\ y \leq y$
by (*metis order-refl star-left-induct-mult*)

lemma *star-star-upper-bound*: $x^\star \leq z^\star \longrightarrow x^{\star\star} \leq z^\star$
by (*metis star-involutive*)

lemma *star-simulation-left*: $x\ ;\ z \leq z\ ;\ y \longrightarrow x^\star\ ;\ z \leq z\ ;\ y^\star$
by (*smt add-commutative add-least-upper-bound mult-right-dist-add less-eq-def mult-associative mult-right-one star.left-plus-below-circ star.circ-increasing star-left-induct star-involutive star.circ-isotone star.circ-reflexive mult-left-sub-dist-add-left*)

lemma *quasicomm-1*: $y\ ;\ x \leq x\ ;\ (x + y)^\star \longleftrightarrow y^\star\ ;\ x \leq x\ ;\ (x + y)^\star$
by (*smt mult-isotone order-refl order-trans star.circ-increasing star-involutive star-simulation-left*)

lemma *star-rtc-3*: $1 + x + y\ ;\ y = y \longrightarrow x^\star \leq y$
by (*metis add-least-upper-bound less-eq-def mult-left-sub-dist-add-left mult-right-one star-left-induct-mult-iff star.circ-sub-dist*)

lemma *star-decompose-1*: $(x + y)^\star = (x^\star\ ;\ y^\star)^\star$
apply (*rule antisym*)
apply (*smt add-least-upper-bound mult-isotone mult-left-one mult-right-one star.circ-increasing star.circ-isotone star.circ-reflexive*)
apply (*smt star.circ-isotone star.circ-sub-dist-3 star-involutive*)
done

lemma *star-sum*: $(x + y)^\star = (x^\star + y^\star)^\star$
by (*metis star-decompose-1 star-involutive*)

lemma *star-decompose-3*: $(x^\star\ ;\ y^\star)^\star = x^\star\ ;\ (y\ ;\ x^\star)^\star$
by (*metis star-decompose-1 star.circ-add-1*)

lemma *star-loop-least-fixpoint*: $y\ ;\ x + z = x \longrightarrow y^\star\ ;\ z \leq x$
by (*metis add-commutative star-left-induct-equal*)

lemma *star-loop-is-least-fixpoint*: *is-least-fixpoint* $(\lambda x\ .\ y\ ;\ x + z)\ (y^\star\ ;\ z)$
by (*smt is-least-fixpoint-def star.circ-loop-fixpoint star-loop-least-fixpoint*)

lemma *star-loop-mu*: $\mu\ (\lambda x\ .\ y\ ;\ x + z) = y^\star\ ;\ z$
by (*metis least-fixpoint-same star-loop-is-least-fixpoint*)

lemma *affine-has-least-fixpoint*: *has-least-fixpoint* $(\lambda x\ .\ y\ ;\ x + z)$
by (*metis has-least-fixpoint-def star-loop-is-least-fixpoint*)

lemma *circ-add*: $(x^\star\ ;\ y)^\star\ ;\ x^\star = (x + y)^\star$ **nitpick** [*expect=genuine,card=7*] **oops**
lemma *circ-mult*: $1 + x\ ;\ (y\ ;\ x)^\star\ ;\ y = (x\ ;\ y)^\star$ **nitpick** [*expect=genuine*] **oops**
lemma *circ-plus-same*: $x^\star\ ;\ x = x\ ;\ x^\star$ **nitpick** [*expect=genuine*] **oops**
lemma *circ-unfold-sum*: $(x + y)^\star = x^\star + x^\star\ ;\ y\ ;\ (x + y)^\star$ **nitpick** [*expect=genuine,card=8*] **oops**
lemma *mult-zero-add-circ-2*: $(x + y\ ;\ 0)^\star = x^\star + x^\star\ ;\ y\ ;\ 0$ **nitpick** [*expect=genuine,card=7*] **oops**
lemma *circ-simulate-left*: $x\ ;\ z \leq z\ ;\ y + w \longrightarrow x^\star\ ;\ z \leq (z + x^\star\ ;\ w)\ ;\ y^\star$ **nitpick** [*expect=genuine*] **oops**
lemma *circ-simulate-1*: $y\ ;\ x \leq x\ ;\ y \longrightarrow y^\star\ ;\ x^\star \leq x^\star\ ;\ y^\star$ **nitpick** [*expect=genuine,card=7*] **oops**

lemma *circ-separate-1*: $y \ ; \ x \leq x \ ; \ y \longrightarrow (x + y)^\star = x^\star \ ; \ y^\star$ **nitpick** [*expect=genuine,card=7*] **oops**
lemma *atomicity-refinement*: $s = s \ ; \ q \wedge x = q \ ; \ x \wedge q \ ; \ b = 0 \wedge r \ ; \ b \leq b \ ; \ r \wedge r \ ; \ l \leq l \ ; \ r \wedge x \ ; \ l \leq l \ ; \ x \wedge b \ ; \ l \leq l \ ; \ b \wedge q \ ; \ l \leq l \ ; \ q \wedge r^\star \ ; \ q \leq q \ ; \ r^\star \wedge q \leq 1 \longrightarrow s \ ; \ (x + b + r + l)^\star \ ; \ q \leq s \ ; \ (x \ ; \ b^\star \ ; \ q + r + l)^\star$ **nitpick** [*expect=genuine*] **oops**
lemma *circ-simulate-left-plus*: $x \ ; \ z \leq z \ ; \ y^\star + w \longrightarrow x^\star \ ; \ z \leq (z + x^\star \ ; \ w) \ ; \ y^\star$ **nitpick** [*expect=genuine*] **oops**
lemma *circ-separate-unfold*: $(y \ ; \ x^\star)^\star = y^\star + y^\star \ ; \ y \ ; \ x \ ; \ x^\star \ ; \ (y \ ; \ x^\star)^\star$ **nitpick** [*expect=genuine*] **oops**
lemma *separation*: $y \ ; \ x \leq x \ ; \ y^\star \longrightarrow (x + y)^\star = x^\star \ ; \ y^\star$ **nitpick** [*expect=genuine,card=7*] **oops**
lemma *circ-simulate-4*: $y \ ; \ x \leq x \ ; \ x^\star \ ; \ (1 + y) \longrightarrow y^\star \ ; \ x^\star \leq x^\star \ ; \ y^\star$ **nitpick** [*expect=genuine,card=7*] **oops**
lemma *circ-simulate-5*: $y \ ; \ x \leq x \ ; \ x^\star \ ; \ (x + y) \longrightarrow y^\star \ ; \ x^\star \leq x^\star \ ; \ y^\star$ **nitpick** [*expect=genuine,card=7*] **oops**
lemma *circ-simulate-6*: $y \ ; \ x \leq x \ ; \ (x + y) \longrightarrow y^\star \ ; \ x^\star \leq x^\star \ ; \ y^\star$ **nitpick** [*expect=genuine,card=7*] **oops**
lemma *circ-separate-4*: $y \ ; \ x \leq x \ ; \ x^\star \ ; \ (1 + y) \longrightarrow (x + y)^\star = x^\star \ ; \ y^\star$ **nitpick** [*expect=genuine,card=7*] **oops**
lemma *circ-separate-5*: $y \ ; \ x \leq x \ ; \ x^\star \ ; \ (x + y) \longrightarrow (x + y)^\star = x^\star \ ; \ y^\star$ **nitpick** [*expect=genuine,card=7*] **oops**
lemma *circ-separate-6*: $y \ ; \ x \leq x \ ; \ (x + y) \longrightarrow (x + y)^\star = x^\star \ ; \ y^\star$ **nitpick** [*expect=genuine,card=7*] **oops**

end

class *strong-left-kleene-algebra* = *left-kleene-algebra* +
assumes *star-right-induct*: $z + x \ ; \ y \leq x \longrightarrow z \ ; \ y^\star \leq x$

begin

lemma *star-plus*: $y^\star \ ; \ y = y \ ; \ y^\star$
by (*smt add-least-upper-bound antisym less-eq-def mult-left-one mult-right-dist-add star.circ-plus-sub star-left-unfold star-right-induct*)

lemma *star-slide*: $(x \ ; \ y)^\star \ ; \ x = x \ ; \ (y \ ; \ x)^\star$
by (*smt add-least-upper-bound antisym mult-associative mult-left-isotone mult-left-one mult-right-one order-trans star-left-slide star-left-unfold star-right-induct*)

lemma *star-simulation-right*: $z \ ; \ x \leq y \ ; \ z \longrightarrow z \ ; \ x^\star \leq y^\star \ ; \ z$
by (*smt add-commutative add-least-upper-bound add-left-upper-bound mult-associative order-trans star.circ-loop-fixpoint star-left-induct star-plus star-right-induct*)

end

sublocale *strong-left-kleene-algebra* < *star*!: *itering-1* **where** *circ* = *star*
apply *unfold-locales*
apply (*metis star-slide order-refl*)
apply (*metis star-simulation-right*)
done

context *strong-left-kleene-algebra*

begin

lemma *star-right-induct-mult*: $y \ ; \ x \leq y \longrightarrow y \ ; \ x^\star \leq y$
by (*metis add-least-upper-bound eq-refl star-right-induct*)

lemma *star-right-induct-mult-iff*: $y \ ; \ x \leq y \longleftrightarrow y \ ; \ x^\star \leq y$
by (*metis mult-right-isotone order-trans star.circ-increasing star-right-induct-mult*)

lemma *star-simulation-right-equal*: $z \ ; \ x = y \ ; \ z \longrightarrow z \ ; \ x^\star = y^\star \ ; \ z$
by (*metis eq-iff star-simulation-left star-simulation-right*)

lemma *star-simulation-star*: $x \ ; \ y \leq y \ ; \ x \longrightarrow x^\star \ ; \ y^\star \leq y^\star \ ; \ x^\star$
by (*metis star-simulation-left star-simulation-right*)

lemma *star-right-induct-equal*: $z + y \ ; \ x = y \longrightarrow z \ ; \ x^\star \leq y$
by (*metis order-refl star-right-induct*)

lemma *star-right-induct-mult-equal*: $y \ ; \ x = y \longrightarrow y \ ; \ x^\star \leq y$
by (*metis order-refl star-right-induct-mult*)

lemma *star-back-loop-least-fixpoint*: $x \ ; \ y + z = x \longrightarrow z \ ; \ y^\star \leq x$
by (*metis add-commutative star-right-induct-equal*)

lemma *star-back-loop-is-least-fixpoint*: *is-least-fixpoint* $(\lambda x \ . \ x \ ; \ y + z) \ (z \ ; \ y^\star)$
by (*smt add-commutative add-right-isotone antisym is-least-fixpoint-def mult-left-isotone star.circ-back-loop-prefixpoint star-back-loop-least-fixpoint star-right-induct*)

lemma *star-back-loop-mu*: $\mu\ (\lambda x\ .\ x\ ;\ y + z) = z\ ;\ y^\star$
by (*metis least-fixpoint-same star-back-loop-is-least-fixpoint*)

lemma *star-square*: $x^\star = (1 + x)\ ;\ (x\ ;\ x)^\star$
proof −
let *?f* = $\lambda y\ .\ y\ ;\ x + 1$
have *1*: *isotone ?f*
by (*smt add-left-isotone isotone-def mult-left-isotone*)
have *2*: $?f \circ ?f = (\lambda y\ .\ y\ ;\ (x\ ;\ x) + (1 + x))$
by (*simp add*: *add-associative add-commutative mult-associative mult-left-one mult-right-dist-add o-def*)
thus *?thesis* **using** *1*
by (*metis mu-square mult-left-one star-back-loop-mu has-least-fixpoint-def star-back-loop-is-least-fixpoint*)
qed

lemma *star-square-2*: $x^\star = (x\ ;\ x)^\star\ ;\ (x + 1)$
by (*smt add-commutative antisym mult-left-one mult-left-sub-dist-add mult-right-dist-add mult-right-one star.circ-square-2 star-slide star-square*)

lemma *star-circ-simulate-right-plus*: $z\ ;\ x \leq y\ ;\ y^\star\ ;\ z + w \longrightarrow z\ ;\ x^\star \leq y^\star\ ;\ (z + w\ ;\ x^\star)$
proof
assume *1*: $z\ ;\ x \leq y\ ;\ y^\star\ ;\ z + w$
have $(z + w\ ;\ x^\star)\ ;\ x \leq z\ ;\ x + w\ ;\ x^\star$
by (*metis add-right-isotone mult-associative mult-right-dist-add mult-right-isotone star.circ-increasing star.circ-transitive-equal*)
also have $... \leq y\ ;\ y^\star\ ;\ z + w + w\ ;\ x^\star$ **using** *1*
by (*metis add-left-isotone*)
also have $... \leq y\ ;\ y^\star\ ;\ z + w\ ;\ x^\star$
by (*metis add-least-upper-bound add-right-isotone add-right-upper-bound star.circ-back-loop-prefixpoint*)
also have $... \leq y^\star\ ;\ (z + w\ ;\ x^\star)$
by (*metis add-least-upper-bound mult-isotone mult-left-isotone mult-left-one mult-left-sub-dist-add-left star.circ-reflexive star.left-plus-below-circ*)
finally have $y^\star\ ;\ (z + w\ ;\ x^\star)\ ;\ x \leq y^\star\ ;\ (z + w\ ;\ x^\star)$
by (*metis mult-associative mult-right-isotone star.circ-transitive-equal*)
thus $z\ ;\ x^\star \leq y^\star\ ;\ (z + w\ ;\ x^\star)$
by (*metis add-least-upper-bound star-right-induct mult-left-sub-dist-add-left star.circ-loop-fixpoint*)
qed

lemma *star-circ-simulate-left-plus*: $x\ ;\ z \leq z\ ;\ y^\star + w \longrightarrow x^\star\ ;\ z \leq (z + x^\star\ ;\ w)\ ;\ y^\star$ **nitpick** [*expect=genuine,card=7*] **oops**

end

class *left-zero-kleene-algebra* = *idempotent-left-zero-semiring* + *strong-left-kleene-algebra*

begin

lemma *star-star-absorb*: $y^\star\ ;\ (y^\star\ ;\ x)^\star\ ;\ y^\star = (y^\star\ ;\ x)^\star\ ;\ y^\star$
by (*metis add-commutative mult-associative star.circ-decompose-4 star.circ-slide-1 star-decompose-1 star-decompose-3*)

lemma *star-circ-simulate-left-plus*: $x\ ;\ z \leq z\ ;\ y^\star + w \longrightarrow x^\star\ ;\ z \leq (z + x^\star\ ;\ w)\ ;\ y^\star$
proof
assume *1*: $x\ ;\ z \leq z\ ;\ y^\star + w$
have $x\ ;\ ((z + x^\star\ ;\ w)\ ;\ y^\star) \leq x\ ;\ z\ ;\ y^\star + x^\star\ ;\ w\ ;\ y^\star$
by (*smt add-right-isotone mult-associative mult-left-dist-add mult-right-dist-add mult-right-sub-dist-add-left star.circ-loop-fixpoint*)
also have $... \leq (z + w + x^\star\ ;\ w)\ ;\ y^\star$ **using** *1*
by (*smt add-left-divisibility add-left-isotone mult-associative mult-right-dist-add star.circ-transitive-equal*)
also have $... = (z + x^\star\ ;\ w)\ ;\ y^\star$
by (*metis add-associative add-right-upper-bound less-eq-def star.circ-loop-fixpoint*)
finally show $x^\star\ ;\ z \leq (z + x^\star\ ;\ w)\ ;\ y^\star$
by (*metis add-least-upper-bound mult-left-sub-dist-add-left mult-right-one star.circ-right-unfold-1 star-left-induct*)
qed

end

— Theorem 2.1

sublocale *left-zero-kleene-algebra* < *star*!: *itering* **where** *circ* = *star*
apply *unfold-locales*
apply (*metis star.circ-add-9*)

apply (*metis star.circ-mult-1*)
apply (*rule star-circ-simulate-right-plus*)
apply (*rule star-circ-simulate-left-plus*)
done

class *kleene-algebra* = *left-zero-kleene-algebra* + *idempotent-semiring*

class *left-kleene-conway-semiring* = *left-kleene-algebra* + *left-conway-semiring*

begin

lemma *star-below-circ*: $x^\star \leq x^\circ$
by (*metis circ-left-unfold mult-right-one order-refl star-left-induct*)

lemma *star-zero-below-circ-mult*: $x^\star \ ; \ 0 \leq x^\circ \ ; \ y$
by (*metis mult-isotone star-below-circ zero-least*)

lemma *star-mult-circ*: $x^\star \ ; \ x^\circ = x^\circ$
by (*metis add-right-divisibility antisym circ-left-unfold star-left-induct-mult star.circ-loop-fixpoint*)

lemma *circ-mult-star*: $x^\circ \ ; \ x^\star = x^\circ$
by (*metis add-associative add-least-upper-bound circ-left-unfold circ-rtc-2 eq-iff left-plus-circ star.circ-add-sub star.circ-back-loop-prefixpoint star.circ-increasing star-below-circ star-mult-circ star-sup-one*)

lemma *circ-star*: $x^{\circ\star} = x^\circ$
by (*metis circ-left-unfold left-plus-circ less-def less-le star.circ-increasing star-below-circ star-sup-one*)

lemma *star-circ*: $x^{\star\circ} = x^{\circ\circ}$
by (*metis antisym circ-circ-add circ-sub-dist less-eq-def star.circ-rtc-2 star-below-circ*)

lemma *circ-add-3*: $(x^\circ \ ; \ y^\circ)^\star \leq (x + y)^\circ$
by (*metis circ-add-1 circ-isotone circ-left-unfold circ-star mult-left-sub-dist-add-left mult-right-isotone mult-right-one star.circ-isotone*)

end

class *left-zero-kleene-conway-semiring* = *left-zero-kleene-algebra* + *itering*

begin

subclass *left-kleene-conway-semiring* **..**

lemma *circ-isolate*: $x^\circ = x^\circ \ ; \ 0 + x^\star$
by (*metis add-commutative antisym circ-add-upper-bound circ-mult-star circ-simulate-absorb star.left-plus-below-circ star-below-circ zero-right-mult-decreasing*)

lemma *circ-isolate-mult*: $x^\circ \ ; \ y = x^\circ \ ; \ 0 + x^\star \ ; \ y$
by (*metis circ-isolate mult-associative mult-left-zero mult-right-dist-add*)

lemma *circ-isolate-mult-sub*: $x^\circ \ ; \ y \leq x^\circ + x^\star \ ; \ y$
by (*metis add-left-isotone circ-isolate-mult zero-right-mult-decreasing*)

lemma *circ-sub-decompose*: $(x^\circ \ ; \ y)^\circ \leq (x^\star \ ; \ y)^\circ \ ; \ x^\circ$
by (*smt add-commutative add-least-upper-bound add-right-upper-bound circ-back-loop-fixpoint circ-isolate-mult mult-zero-add-circ-2 zero-right-mult-decreasing*)

lemma *circ-add-4*: $(x + y)^\circ = (x^\star \ ; \ y)^\circ \ ; \ x^\circ$
apply (*rule antisym*)
apply (*smt circ-add circ-sub-decompose circ-transitive-equal mult-associative mult-left-isotone*)
apply (*smt circ-add circ-isotone mult-left-isotone star-below-circ*)
done

lemma *circ-add-5*: $(x^\circ \ ; \ y)^\circ \ ; \ x^\circ = (x^\star \ ; \ y)^\circ \ ; \ x^\circ$
by (*metis circ-add circ-add-4*)

lemma *plus-circ*: $(x^\star \ ; \ x)^\circ = x^\circ$
by (*smt add-idempotent circ-add-4 circ-decompose-7 circ-star star.circ-decompose-5 star.right-plus-circ*)

lemma $(x^\star \ ; \ y \ ; \ x^\star)^\circ = (x^\star \ ; \ y)^\circ$ **nitpick** [*expect=genuine*] **oops**

end

class *bounded-left-kleene-algebra* = *bounded-idempotent-left-semiring* + *left-kleene-algebra*

sublocale *bounded-left-kleene-algebra* < *star*!: *bounded-left-conway-semiring* **where** *circ* = *star* **..**

class *bounded-left-zero-kleene-algebra* = *bounded-idempotent-left-semiring* + *left-zero-kleene-algebra*

sublocale *bounded-left-zero-kleene-algebra* < *star*!: *bounded-itering* **where** *circ* = *star* **..**

class *bounded-kleene-algebra* = *bounded-idempotent-semiring* + *kleene-algebra*

sublocale *bounded-kleene-algebra* < *star*!: *bounded-itering* **where** *circ* = *star* **..**

end

7 OmegaAlgebra

theory *OmegaAlgebra*

imports *KleeneAlgebra*

begin

class *omega* =
fixes *omega* :: $'a \Rightarrow 'a$ (-^{ω} [100] 100)

class *left-omega-algebra* = *left-kleene-algebra* + *omega* +
assumes *omega-unfold*: $y^{\omega} = y \ ; \ y^{\omega}$
assumes *omega-induct*: $x \leq z + y \ ; \ x \longrightarrow x \leq y^{\omega} + y^{\star} \ ; \ z$

begin

— Many lemmas in this class are taken from Georg Struth's Isabelle theories.

lemma *star-zero-below-omega*: $x^{\star} \ ; \ 0 \leq x^{\omega}$
by (*metis add-left-zero omega-unfold star-left-induct-equal*)

lemma *star-zero-below-omega-zero*: $x^{\star} \ ; \ 0 \leq x^{\omega} \ ; \ 0$
by (*metis add-left-zero mult-associative omega-unfold star-left-induct-equal*)

lemma *omega-induct-mult*: $y \leq x \ ; \ y \longrightarrow y \leq x^{\omega}$
by (*metis add-commutative add-left-zero less-eq-def omega-induct star-zero-below-omega*)

lemma *omega-sub-dist*: $x^{\omega} \leq (x+y)^{\omega}$
by (*metis mult-right-sub-dist-add-left omega-induct-mult omega-unfold*)

lemma *omega-isotone*: $x \leq y \longrightarrow x^{\omega} \leq y^{\omega}$
by (*metis less-eq-def omega-sub-dist*)

lemma *omega-induct-equal*: $y = z + x \ ; \ y \longrightarrow y \leq x^{\omega} + x^{\star} \ ; \ z$
by (*metis omega-induct order-refl*)

lemma *omega-zero*: $0^{\omega} = 0$
by (*metis mult-left-zero omega-unfold*)

lemma *omega-one-greatest*: $x \leq 1^{\omega}$
by (*metis mult-left-one omega-induct-mult order-refl*)

lemma *star-mult-omega*: $x^{\star} \ ; \ x^{\omega} = x^{\omega}$
by (*metis antisym-conv mult-isotone omega-unfold star.circ-increasing star-left-induct-mult-equal star-left-induct-mult-iff*)

lemma *omega-sub-vector*: $x^{\omega} \ ; \ y \leq x^{\omega}$
by (*metis mult-associative omega-induct-mult omega-unfold order-refl*)

lemma *omega-simulation*: $z \ ; \ x \leq y \ ; \ z \longrightarrow z \ ; \ x^{\omega} \leq y^{\omega}$
by (*smt less-eq-def mult-associative mult-right-sub-dist-add-left omega-induct-mult omega-unfold*)

lemma *omega-omega*: $x^{\omega\omega} \leq x^{\omega}$
by (*metis omega-sub-vector omega-unfold*)

lemma *left-plus-omega*: $(x \ ; \ x^{\star})^{\omega} = x^{\omega}$
by (*metis antisym mult-associative omega-induct-mult omega-unfold order-refl star.left-plus-circ star-mult-omega*)

lemma *omega-slide*: $x \ ; \ (y \ ; \ x)^{\omega} = (x \ ; \ y)^{\omega}$
by (*metis antisym mult-associative mult-right-isotone omega-simulation omega-unfold order-refl*)

lemma *omega-simulation-2*: $y \ ; \ x \leq x \ ; \ y \longrightarrow (x \ ; \ y)^{\omega} \leq x^{\omega}$
by (*metis less-eq-def mult-right-isotone omega-induct-mult omega-slide omega-sub-dist*)

lemma *wagner*: $(x + y)^{\omega} = x \ ; \ (x + y)^{\omega} + z \longrightarrow (x + y)^{\omega} = x^{\omega} + x^{\star} \ ; \ z$
by (*metis add-commutative add-least-upper-bound eq-iff omega-induct omega-sub-dist star-left-induct*)

lemma *right-plus-omega*: $(x^{\star} \ ; \ x)^{\omega} = x^{\omega}$
by (*metis left-plus-omega omega-slide star-mult-omega*)

lemma *omega-sub-dist-1*: $(x\ ;\ y^\star)^\omega \leq (x + y)^\omega$
by (*metis add-least-upper-bound left-plus-omega mult-isotone mult-left-one mult-right-dist-add omega-isotone order-refl star-decompose-1 star.circ-increasing star.circ-plus-one*)

lemma *omega-sub-dist-2*: $(x^\star\ ;\ y)^\omega \leq (x + y)^\omega$
by (*metis add-commutative mult-isotone omega-slide omega-sub-dist-1 star-mult-omega star.circ-sub-dist*)

lemma *omega-star*: $(x^\omega)^\star = 1 + x^\omega$
by (*metis eq-iff mult-left-sub-dist-add-left mult-right-one omega-sub-vector star.circ-left-unfold*)

lemma *omega-mult-omega-star*: $x^\omega\ ;\ x^{\omega\star} = x^\omega$
by (*metis add-least-upper-bound antisym omega-sub-vector star.circ-back-loop-prefixpoint*)

lemma *omega-sum-unfold-1*: $(x + y)^\omega = x^\omega + x^\star\ ;\ y\ ;\ (x + y)^\omega$
by (*metis mult-associative mult-right-dist-add omega-unfold wagner*)

lemma *omega-sum-unfold-2*: $(x + y)^\omega \leq (x^\star\ ;\ y)^\omega + (x^\star\ ;\ y)^\star\ ;\ x^\omega$
by (*metis omega-induct-equal omega-sum-unfold-1*)

lemma *omega-sum-unfold-3*: $(x^\star\ ;\ y)^\star\ ;\ x^\omega \leq (x + y)^\omega$
by (*metis omega-sum-unfold-1 star-left-induct-equal*)

lemma *omega-decompose*: $(x + y)^\omega = (x^\star\ ;\ y)^\omega + (x^\star\ ;\ y)^\star\ ;\ x^\omega$
by (*metis add-least-upper-bound antisym omega-sub-dist-2 omega-sum-unfold-2 omega-sum-unfold-3*)

lemma *omega-loop-fixpoint*: $y\ ;\ (y^\omega + y^\star\ ;\ z) + z = y^\omega + y^\star\ ;\ z$
apply (*rule antisym*)
apply (*smt add-commutative add-least-upper-bound add-right-isotone add-right-upper-bound mult-left-sub-dist-add-left mult-right-isotone omega-induct omega-unfold order-trans star.circ-loop-fixpoint*)
apply (*smt add-associative add-left-isotone mult-left-sub-dist-add omega-unfold star.circ-loop-fixpoint*)
done

lemma *omega-loop-greatest-fixpoint*: $y\ ;\ x + z = x \longrightarrow x \leq y^\omega + y^\star\ ;\ z$
by (*metis add-commutative omega-induct-equal*)

lemma *omega-square*: $x^\omega = (x\ ;\ x)^\omega$
by (*metis antisym mult-associative omega-induct-mult omega-mult-omega-star omega-slide omega-sub-vector omega-unfold*)

lemma *mult-zero-omega*: $(x\ ;\ 0)^\omega = x\ ;\ 0$
by (*metis mult-left-zero omega-slide*)

lemma *mult-zero-add-omega*: $(x + y\ ;\ 0)^\omega = x^\omega + x^\star\ ;\ y\ ;\ 0$
by (*smt add-associative add-commutative add-idempotent mult-associative mult-left-one mult-left-zero mult-right-dist-add mult-zero-omega star.mult-zero-circ omega-decompose*)

lemma *omega-mult-star*: $x^\omega\ ;\ x^\star = x^\omega$
by (*metis antisym mult-left-sub-dist-add-left mult-right-one omega-sub-vector star.circ-plus-one*)

lemma *omega-loop-is-greatest-fixpoint*: *is-greatest-fixpoint* $(\lambda x\ .\ y\ ;\ x + z)$ $(y^\omega + y^\star\ ;\ z)$
by (*smt is-greatest-fixpoint-def omega-loop-fixpoint omega-loop-greatest-fixpoint*)

lemma *omega-loop-nu*: $\nu\ (\lambda x\ .\ y\ ;\ x + z) = y^\omega + y^\star\ ;\ z$
by (*metis greatest-fixpoint-same omega-loop-is-greatest-fixpoint*)

lemma *omega-loop-zero-is-greatest-fixpoint*: *is-greatest-fixpoint* $(\lambda x\ .\ y\ ;\ x)$ (y^ω)
by (*metis is-greatest-fixpoint-def omega-induct-mult omega-unfold order-refl*)

lemma *omega-loop-zero-nu*: $\nu\ (\lambda x\ .\ y\ ;\ x) = y^\omega$
by (*metis greatest-fixpoint-same omega-loop-zero-is-greatest-fixpoint*)

lemma *affine-has-greatest-fixpoint*: *has-greatest-fixpoint* $(\lambda x\ .\ y\ ;\ x + z)$
by (*metis has-greatest-fixpoint-def omega-loop-is-greatest-fixpoint*)

lemma *omega-separate-unfold*: $(x^\star\ ;\ y)^\omega = y^\omega + y^\star\ ;\ x\ ;\ (x^\star\ ;\ y)^\omega$
by (*metis add-commutative mult-associative omega-slide omega-sum-unfold-1 star.circ-loop-fixpoint*)

lemma *omega-zero-left-slide*: $(x\ ;\ y)^\star\ ;\ ((x\ ;\ y)^\omega\ ;\ 0 + 1)\ ;\ x \leq x\ ;\ (y\ ;\ x)^\star\ ;\ ((y\ ;\ x)^\omega\ ;\ 0 + 1)$
proof −
have $x + x\ ;\ (y\ ;\ x)\ ;\ (y\ ;\ x)^\star\ ;\ ((y\ ;\ x)^\omega\ ;\ 0 + 1) \leq x\ ;\ (y\ ;\ x)^\star\ ;\ ((y\ ;\ x)^\omega\ ;\ 0 + 1)$

by (*smt add-commutative add-least-upper-bound mult-associative mult-left-isotone mult-right-isotone star.circ-back-loop-prefixpoint star.left-plus-below-circ star.mult-zero-add-circ star.mult-zero-circ*)
hence $((x\ ;\ y)^\omega\ ;\ 0\ +\ 1)\ ;\ x\ +\ x\ ;\ y\ ;\ (x\ ;\ (y\ ;\ x)^\star\ ;\ ((y\ ;\ x)^\omega\ ;\ 0\ +\ 1)) \leq x\ ;\ (y\ ;\ x)^\star\ ;\ ((y\ ;\ x)^\omega\ ;\ 0\ +\ 1)$
by (*smt add-associative less-eq-def mult-associative mult-left-one mult-left-sub-dist-add-left mult-left-zero mult-right-dist-add omega-slide star-mult-omega*)
thus *?thesis*
by (*metis mult-associative star-left-induct*)
qed

lemma *omega-zero-add-1*: $(x\ +\ y)^\star\ ;\ ((x\ +\ y)^\omega\ ;\ 0\ +\ 1) = x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1)\ ;\ (y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\star\ ;\ ((y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\omega\ ;\ 0\ +\ 1)$
proof (*rule antisym*)
have *1*: $(x\ +\ y)\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1)\ ;\ (y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\star\ ;\ ((y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\omega\ ;\ 0\ +\ 1) \leq x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1)\ ;\ (y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\star\ ;\ ((y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\omega\ ;\ 0\ +\ 1)$
by (*smt add-associative add-commutative less-eq-def mult-associative mult-left-isotone mult-right-dist-add star.circ-add-1 star.left-plus-below-circ star.mult-zero-add-circ star.mult-zero-circ star-decompose-1*)
have *2*: $1 \leq x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1)\ ;\ (y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\star\ ;\ ((y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\omega\ ;\ 0\ +\ 1)$
by (*smt add-commutative mult-associative star.circ-add-1 star.circ-reflexive star.mult-zero-add-circ star.mult-zero-circ*)
have $(y\ ;\ x^\star)^\omega\ ;\ 0 \leq (y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\omega\ ;\ 0$
by (*smt mult-left-isotone mult-left-sub-dist-add-right mult-right-one omega-isotone*)
also have *3*: $\ldots \leq (x^\omega\ ;\ 0\ +\ 1)\ ;\ (y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\star\ ;\ ((y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\omega\ ;\ 0\ +\ 1)$
by (*smt add-commutative mult-associative mult-left-one mult-right-sub-dist-add-left order-trans star.circ-sub-dist-1 star.mult-zero-add-circ star.mult-zero-circ*)
finally have *4*: $(x^\star\ ;\ y)^\omega\ ;\ 0 \leq x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1)\ ;\ (y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\star\ ;\ ((y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\omega\ ;\ 0\ +\ 1)$
by (*smt mult-associative mult-right-isotone omega-slide*)
have $y\ ;\ (x^\star\ ;\ y)^\star\ ;\ x^\omega\ ;\ 0 \leq y\ ;\ (x^\star\ ;\ (x^\omega\ ;\ 0\ +\ y))^\star\ ;\ x^\star\ ;\ x^\omega\ ;\ 0\ ;\ (y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\omega\ ;\ 0$
by (*metis mult-left-isotone mult-left-sub-dist-add-right mult-right-isotone star.circ-isotone mult-associative mult-left-zero star-mult-omega*)
also have $\ldots \leq y\ ;\ (x^\star\ ;\ (x^\omega\ ;\ 0\ +\ y))^\star\ ;\ (x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1)\ ;\ y)^\omega\ ;\ 0$
by (*smt mult-associative mult-left-isotone mult-left-sub-dist-add-left omega-slide*)
also have $\ldots = y\ ;\ (x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1)\ ;\ y)^\omega\ ;\ 0$
by (*smt mult-associative mult-left-one mult-left-zero mult-right-dist-add star-mult-omega*)
finally have $x^\star\ ;\ y\ ;\ (x^\star\ ;\ y)^\star\ ;\ x^\omega\ ;\ 0 \leq x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1)\ ;\ (y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\star\ ;\ ((y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\omega\ ;\ 0\ +\ 1)$ **using** *3*
by (*smt mult-associative mult-right-isotone omega-slide order-trans*)
hence $(x^\star\ ;\ y)^\star\ ;\ x^\omega\ ;\ 0 \leq x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1)\ ;\ (y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\star\ ;\ ((y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\omega\ ;\ 0\ +\ 1)$
by (*smt add-associative add-commutative less-eq-def mult-associative mult-isotone mult-left-one mult-right-one mult-right-sub-dist-add-left order-trans star.circ-loop-fixpoint star.circ-reflexive star.mult-zero-circ*)
hence $(x\ +\ y)^\omega\ ;\ 0 \leq x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1)\ ;\ (y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\star\ ;\ ((y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\omega\ ;\ 0\ +\ 1)$ **using** *4*
by (*metis add-least-upper-bound mult-right-dist-add omega-decompose*)
thus $(x\ +\ y)^\star\ ;\ ((x\ +\ y)^\omega\ ;\ 0\ +\ 1) \leq x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1)\ ;\ (y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\star\ ;\ ((y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\omega\ ;\ 0\ +\ 1)$ **using** *1 2*
by (*smt add-least-upper-bound mult-associative star-left-induct*)
next
have *5*: $x^\omega\ ;\ 0 \leq (x\ +\ y)^\star\ ;\ ((x\ +\ y)^\omega\ ;\ 0\ +\ 1)$
by (*metis add-commutative add-left-zero mult-associative mult-left-isotone mult-left-one mult-right-dist-add omega-sub-dist order-trans star-mult-omega zero-right-mult-decreasing*)
have *6*: $(y\ ;\ x^\star)^\omega\ ;\ 0 \leq (x\ +\ y)^\star\ ;\ ((x\ +\ y)^\omega\ ;\ 0\ +\ 1)$
by (*metis add-commutative mult-left-isotone omega-sub-dist-1 mult-associative mult-left-sub-dist-add-left order-trans star-mult-omega*)
have *7*: $(y\ ;\ x^\star)^\star \leq (x\ +\ y)^\star$
by (*metis mult-left-one mult-right-sub-dist-add-left star.circ-add-1 star.circ-plus-one*)
hence $(y\ ;\ x^\star)^\star\ ;\ x^\omega\ ;\ 0 \leq (x\ +\ y)^\star\ ;\ ((x\ +\ y)^\omega\ ;\ 0\ +\ 1)$
by (*smt add-associative less-eq-def mult-associative mult-isotone mult-right-dist-add omega-sub-dist*)
hence $(x^\omega\ ;\ 0\ +\ y\ ;\ x^\star)^\omega\ ;\ 0 \leq (x\ +\ y)^\star\ ;\ ((x\ +\ y)^\omega\ ;\ 0\ +\ 1)$ **using** *6*
by (*smt add-commutative add-least-upper-bound mult-associative mult-right-dist-add mult-zero-add-omega omega-unfold omega-zero*)
hence $(y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\omega\ ;\ 0 \leq y\ ;\ x^\star\ ;\ (x\ +\ y)^\star\ ;\ ((x\ +\ y)^\omega\ ;\ 0\ +\ 1)$
by (*smt mult-associative mult-left-one mult-left-zero mult-right-dist-add mult-right-isotone omega-slide*)
also have $\ldots \leq (x\ +\ y)^\star\ ;\ ((x\ +\ y)^\omega\ ;\ 0\ +\ 1)$ **using** *7*
by (*metis mult-left-isotone order-refl star.circ-mult-upper-bound star-left-induct-mult-iff*)
finally have $(y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\star\ ;\ ((y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\omega\ ;\ 0\ +\ 1) \leq (x\ +\ y)^\star\ ;\ ((x\ +\ y)^\omega\ ;\ 0\ +\ 1)$ **using** *5*
by (*smt add-commutative add-least-upper-bound mult-associative order-refl star.circ-mult-upper-bound star.circ-reflexive star.circ-sub-dist-1 star.mult-zero-add-circ star.mult-zero-circ star-left-induct*)
hence $(x^\omega\ ;\ 0\ +\ 1)\ ;\ (y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\star\ ;\ ((y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\omega\ ;\ 0\ +\ 1) \leq (x\ +\ y)^\star\ ;\ ((x\ +\ y)^\omega\ ;\ 0\ +\ 1)$ **using** *5*
by (*metis add-commutative mult-associative star.circ-isotone star.circ-mult-upper-bound star.mult-zero-add-circ star.mult-zero-circ star-involutive*)
thus $x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1)\ ;\ (y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\star\ ;\ ((y\ ;\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))^\omega\ ;\ 0\ +\ 1) \leq (x\ +\ y)^\star\ ;\ ((x\ +\ y)^\omega\ ;\ 0\ +\ 1)$
by (*smt add-associative add-commutative mult-associative star.circ-mult-upper-bound star.circ-sub-dist*

star.mult-zero-add-circ star.mult-zero-circ)
qed

lemma *star-omega-greatest*: $x^{\star\omega} = 1^\omega$
by (*metis add-commutative less-eq-def omega-one-greatest omega-sub-dist star.circ-plus-one*)

lemma *omega-vector-greatest*: $x^\omega \ ; \ 1^\omega = x^\omega$
by (*metis antisym mult-isotone omega-mult-omega-star omega-one-greatest omega-sub-vector*)

lemma *mult-greatest-omega*: $(x \ ; \ 1^\omega)^\omega \leq x \ ; \ 1^\omega$
by (*metis mult-right-isotone omega-slide omega-sub-vector*)

lemma *omega-mult-star-2*: $x^\omega \ ; \ y^\star = x^\omega$
by (*metis mult-associative omega-mult-star omega-vector-greatest star-involutive star-omega-greatest*)

lemma *omega-import*: $p \leq p \ ; \ p \wedge p \ ; \ x \leq x \ ; \ p \longrightarrow p \ ; \ x^\omega = p \ ; \ (p \ ; \ x)^\omega$
proof
assume *1*: $p \leq p \ ; \ p \wedge p \ ; \ x \leq x \ ; \ p$
hence $p \ ; \ x^\omega \leq p \ ; \ (p \ ; \ x) \ ; \ x^\omega$
by (*metis mult-associative mult-left-isotone omega-unfold*)
also have $... \leq p \ ; \ x \ ; \ p \ ; \ x^\omega$ **using** *1*
by (*metis mult-associative mult-left-isotone mult-right-isotone*)
finally have $p \ ; \ x^\omega \leq (p \ ; \ x)^\omega$
by (*metis mult-associative omega-induct-mult*)
hence $p \ ; \ x^\omega \leq p \ ; \ (p \ ; \ x)^\omega$ **using** *1*
by (*metis mult-associative mult-left-isotone mult-right-isotone order-trans*)
thus $p \ ; \ x^\omega = p \ ; \ (p \ ; \ x)^\omega$ **using** *1*
by (*metis add-left-divisibility antisym mult-right-isotone omega-induct-mult omega-slide omega-sub-dist*)
qed

lemma *omega-circ-simulate-right-plus*: $z \ ; \ x \leq y \ ; \ (y^\omega \ ; \ 0 + y^\star) \ ; \ z + w \longrightarrow z \ ; \ (x^\omega \ ; \ 0 + x^\star) \leq (y^\omega \ ; \ 0 + y^\star) \ ; \ (z + w \ ; \ (x^\omega \ ; \ 0 + x^\star))$ **nitpick** [*expect=genuine*] **oops**
lemma *omega-circ-simulate-left-plus*: $x \ ; \ z \leq z \ ; \ (y^\omega \ ; \ 0 + y^\star) + w \longrightarrow (x^\omega \ ; \ 0 + x^\star) \ ; \ z \leq (z + (x^\omega \ ; \ 0 + x^\star) \ ; \ w) \ ; \ (y^\omega \ ; \ 0 + y^\star)$ **nitpick** [*expect=genuine*] **oops**

end

— Theorem 50.2

sublocale *left-omega-algebra* < *comb0*!: *left-conway-semiring* **where** *circ* = $(\lambda x \ . \ x^\star \ ; \ (x^\omega \ ; \ 0 + 1))$
apply *unfold-locales*
apply (*smt add-associative add-commutative less-eq-def mult-associative mult-left-sub-dist-add-left omega-unfold star.circ-loop-fixpoint star-mult-omega*)
apply (*smt mult-associative omega-zero-left-slide*)
apply (*smt mult-associative omega-zero-add-1*)
done

class *left-zero-omega-algebra* = *left-zero-kleene-algebra* + *left-omega-algebra*

begin

lemma *star-omega-absorb*: $y^\star \ ; \ (y^\star \ ; \ x)^\star \ ; \ y^\omega = (y^\star \ ; \ x)^\star \ ; \ y^\omega$
proof −
have $y^\star \ ; \ (y^\star \ ; \ x)^\star \ ; \ y^\omega = y^\star \ ; \ y^\star \ ; \ x \ ; \ (y^\star \ ; \ x)^\star \ ; \ y^\omega + y^\star \ ; \ y^\omega$
by (*metis add-commutative mult-associative mult-right-dist-add star.circ-back-loop-fixpoint star.circ-plus-same*)
thus *?thesis*
by (*metis mult-associative star.circ-loop-fixpoint star.circ-transitive-equal star-mult-omega*)
qed

lemma *omega-circ-simulate-right-plus*: $z \ ; \ x \leq y \ ; \ (y^\omega \ ; \ 0 + y^\star) \ ; \ z + w \longrightarrow z \ ; \ (x^\omega \ ; \ 0 + x^\star) \leq (y^\omega \ ; \ 0 + y^\star) \ ; \ (z + w \ ; \ (x^\omega \ ; \ 0 + x^\star))$
proof
assume $z \ ; \ x \leq y \ ; \ (y^\omega \ ; \ 0 + y^\star) \ ; \ z + w$
hence *1*: $z \ ; \ x \leq y^\omega \ ; \ 0 + y \ ; \ y^\star \ ; \ z + w$
by (*metis mult-associative mult-left-dist-add mult-left-zero mult-right-dist-add omega-unfold*)
hence $(y^\omega \ ; \ 0 + y^\star \ ; \ z + y^\star \ ; \ w \ ; \ x^\omega \ ; \ 0 + y^\star \ ; \ w \ ; \ x^\star) \ ; \ x \leq y^\omega \ ; \ 0 + y^\star \ ; \ (y^\omega \ ; \ 0 + y \ ; \ y^\star \ ; \ z + w) + y^\star \ ; \ w \ ; \ x^\omega \ ; \ 0 + y^\star \ ; \ w \ ; \ x^\star$
by (*smt add-associative add-left-upper-bound add-right-upper-bound less-eq-def mult-associative mult-left-dist-add mult-left-zero mult-right-dist-add star.circ-back-loop-fixpoint*)

also have ... = y^ω ; $0 + y^\star$; y ; $y^\star$; $z + y^\star$; w ; x^ω ; $0 + y^\star$; w ; $x^\star$
by (*smt add-associative add-right-upper-bound less-eq-def mult-associative mult-left-dist-add star.circ-back-loop-fixpoint star-mult-omega*)
also have ... ≤ y^ω ; $0 + y^\star$; $z + y^\star$; w ; x^ω ; $0 + y^\star$; w ; $x^\star$
by (*smt add-commutative add-left-isotone mult-left-isotone star.circ-increasing star.circ-plus-same star.circ-transitive-equal*)
finally have $z + (y^\omega$; $0 + y^\star$; $z + y^\star$; w ; x^ω ; $0 + y^\star$; w ; $x^\star)$; $x \leq y^\omega$; $0 + y^\star$; $z + y^\star$; w ; x^ω ; $0 + y^\star$; w ; $x^\star$
by (*smt add-least-upper-bound add-left-upper-bound star.circ-loop-fixpoint*)
hence *2*: z ; $x^\star \leq y^\omega$; $0 + y^\star$; $z + y^\star$; w ; x^ω ; $0 + y^\star$; w ; $x^\star$
by (*metis star-right-induct*)
have z ; x^ω ; $0 \leq (y^\omega$; $0 + y$; $y^\star$; $z + w)$; x^ω ; 0 **using** *1*
by (*smt add-left-divisibility mult-associative mult-right-sub-dist-add-left omega-unfold*)
hence z ; x^ω ; $0 \leq y^\omega + y^\star$; $(y^\omega$; $0 + w$; x^ω ; $0)$
by (*smt add-associative add-commutative left-plus-omega mult-associative mult-left-zero mult-right-dist-add omega-induct star.left-plus-circ*)
thus z ; $(x^\omega$; $0 + x^\star) \leq (y^\omega$; $0 + y^\star)$; $(z + w$; $(x^\omega$; $0 + x^\star))$ **using** *2*
by (*smt add-associative add-commutative less-eq-def mult-associative mult-left-dist-add mult-left-zero mult-right-dist-add omega-unfold omega-zero star-mult-omega zero-right-mult-decreasing*)
qed

lemma *omega-circ-simulate-left-plus*: x ; $z \leq z$; $(y^\omega$; $0 + y^\star) + w \longrightarrow (x^\omega$; $0 + x^\star)$; $z \leq (z + (x^\omega$; $0 + x^\star)$; $w)$; $(y^\omega$; $0 + y^\star)$
proof
assume *1*: x ; $z \leq z$; $(y^\omega$; $0 + y^\star) + w$
have x ; $(z$; y^ω ; $0 + z$; $y^\star + x^\omega$; $0 + x^\star$; w ; y^ω ; $0 + x^\star$; w ; $y^\star) = x$; z ; y^ω ; $0 + x$; z ; $y^\star + x^\omega$; $0 + x$; $x^\star$; w ; y^ω ; $0 + x$; $x^\star$; w ; $y^\star$
by (*smt mult-associative mult-left-dist-add omega-unfold*)
also have ... ≤ x ; z ; y^ω ; $0 + x$; z ; $y^\star + x^\omega$; $0 + x^\star$; w ; y^ω ; $0 + x^\star$; w ; $y^\star$
by (*metis add-isotone add-right-isotone mult-left-isotone star.left-plus-below-circ*)
also have ... ≤ $(z$; y^ω ; $0 + z$; $y^\star + w)$; y^ω ; $0 + (z$; y^ω ; $0 + z$; $y^\star + w)$; $y^\star + x^\omega$; $0 + x^\star$; w ; y^ω ; $0 + x^\star$; w ; $y^\star$ **using** *1*
by (*metis add-left-isotone mult-associative mult-left-dist-add mult-left-isotone*)
also have ... = z ; y^ω ; $0 + z$; $y^\star$; y^ω ; $0 + w$; y^ω ; $0 + z$; y^ω ; $0 + z$; $y^\star$; $y^\star + w$; $y^\star + x^\omega$; $0 + x^\star$; w ; y^ω ; $0 + x^\star$; w ; $y^\star$
by (*smt add-associative mult-associative mult-left-zero mult-right-dist-add*)
also have ... = z ; y^ω ; $0 + z$; $y^\star + x^\omega$; $0 + x^\star$; w ; y^ω ; $0 + x^\star$; w ; $y^\star$
by (*smt add-associative add-commutative add-idempotent mult-associative mult-right-dist-add star.circ-loop-fixpoint star.circ-transitive-equal star-mult-omega*)
finally have $(x^\omega$; $0 + x^\star)$; $z \leq z$; y^ω ; $0 + z$; $y^\star + x^\omega$; $0 + x^\star$; w ; y^ω ; $0 + x^\star$; w ; $y^\star$
by (*smt add-least-upper-bound add-left-upper-bound mult-associative mult-left-zero mult-right-dist-add star.circ-back-loop-fixpoint star-left-induct*)
thus $(x^\omega$; $0 + x^\star)$; $z \leq (z + (x^\omega$; $0 + x^\star)$; $w)$; $(y^\omega$; $0 + y^\star)$
by (*smt add-associative mult-associative mult-left-dist-add mult-left-zero mult-right-dist-add*)
qed

lemma *omega-translate*: $x^\star$; $(x^\omega$; $0 + 1) = x^\omega$; $0 + x^\star$
by (*metis mult-associative mult-left-dist-add mult-right-one star-mult-omega*)

lemma *omega-circ-simulate-right*: z ; $x \leq y$; $z + w \longrightarrow z$; $(x^\omega$; $0 + x^\star) \leq (y^\omega$; $0 + y^\star)$; $(z + w$; $(x^\omega$; $0 + x^\star))$
proof
assume z ; $x \leq y$; $z + w$
also have ... ≤ y ; $(y^\omega$; $0 + y^\star)$; $z + w$
by (*metis add-left-isotone comb0.circ-reflexive mult-left-isotone mult-right-isotone mult-right-one omega-translate*)
finally show z ; $(x^\omega$; $0 + x^\star) \leq (y^\omega$; $0 + y^\star)$; $(z + w$; $(x^\omega$; $0 + x^\star))$
by (*metis omega-circ-simulate-right-plus*)
qed

end

sublocale *left-zero-omega-algebra* < *comb1*!: *left-conway-semiring-1* **where** *circ* = $(\lambda x$. $x^\star$; $(x^\omega$; $0 + 1))$
apply *unfold-locales*
apply (*smt eq-iff mult-associative mult-left-dist-add mult-left-zero mult-right-dist-add mult-right-one omega-slide star-slide*)
done

sublocale *left-zero-omega-algebra* < *comb0*!: *itering* **where** *circ* = $(\lambda x$. $x^\star$; $(x^\omega$; $0 + 1))$
apply *unfold-locales*
apply (*metis comb1.circ-add-9*)
apply (*metis comb1.circ-mult-1*)
apply (*metis omega-circ-simulate-right-plus omega-translate*)
apply (*metis omega-circ-simulate-left-plus omega-translate*)

done

— Theorem 2.2

sublocale *left-zero-omega-algebra* < *comb2*!: *itering* **where** *circ* = $(\lambda x\ .\ x^\omega\ ;\ 0\ +\ x^\star)$
apply *unfold-locales*
apply (*metis comb1.circ-add-9 omega-translate*)
apply (*metis comb1.circ-mult-1 omega-translate*)
apply (*metis omega-circ-simulate-right-plus*)
apply (*metis omega-circ-simulate-left-plus*)
done

class *omega-algebra* = *kleene-algebra* + *left-zero-omega-algebra*

class *left-omega-conway-semiring* = *left-omega-algebra* + *left-conway-semiring*

begin

subclass *left-kleene-conway-semiring* **..**

lemma *circ-below-omega-star*: $x^\circ \leq x^\omega + x^\star$
by (*metis circ-left-unfold mult-right-one omega-induct order-refl*)

lemma *omega-mult-circ*: $x^\omega\ ;\ x^\circ = x^\omega$
by (*metis circ-star mult-associative omega-mult-star omega-vector-greatest star-omega-greatest*)

lemma *circ-mult-omega*: $x^\circ\ ;\ x^\omega = x^\omega$
by (*metis antisym add-right-divisibility circ-loop-fixpoint circ-plus-sub omega-simulation*)

lemma *circ-omega-greatest*: $x^{\circ\omega} = 1^\omega$
by (*metis circ-star star-omega-greatest*)

lemma *omega-circ*: $x^{\omega\circ} = 1 + x^\omega$
by (*metis antisym circ-left-unfold mult-left-sub-dist-add-left mult-right-one omega-sub-vector*)

end

class *bounded-left-omega-algebra* = *bounded-left-kleene-algebra* + *left-omega-algebra*

begin

lemma *omega-one*: $1^\omega = T$
by (*smt add-left-top less-eq-def omega-one-greatest*)

lemma *star-omega-top*: $x^{\star\omega} = T$
by (*metis add-left-top less-eq-def omega-one omega-sub-dist star.circ-plus-one*)

lemma *omega-vector*: $x^\omega\ ;\ T = x^\omega$
by (*metis add-commutative less-eq-def omega-sub-vector top-right-mult-increasing*)

lemma *mult-top-omega*: $(x\ ;\ T)^\omega \leq x\ ;\ T$
by (*metis mult-right-isotone omega-slide top-greatest*)

end

sublocale *bounded-left-omega-algebra* < *comb0*!: *bounded-left-conway-semiring* **where** *circ* = $(\lambda x\ .\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))$ **..**

class *bounded-left-zero-omega-algebra* = *bounded-left-zero-kleene-algebra* + *left-zero-omega-algebra*

begin

subclass *bounded-left-omega-algebra* **..**

end

sublocale *bounded-left-zero-omega-algebra* < *comb0*!: *bounded-itering* **where** *circ* = $(\lambda x\ .\ x^\star\ ;\ (x^\omega\ ;\ 0\ +\ 1))$ **..**

class *bounded-omega-algebra* = *bounded-kleene-algebra* + *omega-algebra*

begin

subclass *bounded-left-zero-omega-algebra* **..**

end

class *bounded-left-omega-conway-semiring* = *bounded-left-omega-algebra* + *left-omega-conway-semiring*

begin

subclass *left-kleene-conway-semiring* **..**

subclass *bounded-left-conway-semiring* **..**

lemma *circ-omega*: $x^{\circ\omega} = T$
by (*metis circ-star star-omega-top*)

end

class *top-left-omega-algebra* = *bounded-left-omega-algebra* +
assumes *top-left-zero*: $T\ ;\ x = T$

begin

lemma *omega-translate-3*: $x^\star\ ;\ (x^\omega\ ;\ 0 + 1) = x^\star\ ;\ (x^\omega + 1)$
by (*metis mult-associative omega-mult-star-2 star.circ-top-1 top-left-zero*)

end

— Theorem 50.2

sublocale *top-left-omega-algebra* < *comb4*!: *left-conway-semiring* **where** *circ* = $(\lambda x\ .\ x^\star\ ;\ (x^\omega + 1))$
apply *unfold-locales*
apply (*metis comb0.circ-left-unfold omega-translate-3*)
apply (*metis comb0.circ-left-slide omega-translate-3*)
apply (*metis comb0.circ-add-1 omega-translate-3*)
done

class *top-left-zero-omega-algebra* = *bounded-left-zero-omega-algebra* +
assumes *top-left-zero*: $T\ ;\ x = T$

begin

lemma *omega-translate-2*: $x^\omega\ ;\ 0 + x^\star = x^\omega + x^\star$
by (*metis mult-associative omega-mult-star-2 star.circ-top top-left-zero*)

end

— Theorem 2.3

sublocale *top-left-zero-omega-algebra* < *comb3*!: *itering* **where** *circ* = $(\lambda x\ .\ x^\omega + x^\star)$
apply *unfold-locales*
apply (*metis comb2.circ-add-9 omega-translate-2*)
apply (*metis comb2.circ-mult-1 omega-translate-2*)
apply (*metis omega-circ-simulate-right-plus omega-translate-2*)
apply (*metis omega-circ-simulate-left-plus omega-translate-2*)
done

class *Omega* =
fixes *Omega* :: $'a \Rightarrow 'a$ $(\text{-}^\Omega\ [100]\ 100)$

end

8 BinaryItering

theory *BinaryItering*

imports *Semiring*

begin

class *binary-itering* = *idempotent-left-zero-semiring* + *while* +
assumes *while-productstar*: $(x\ ;\ y) \star z = z + x\ ;\ ((y\ ;\ x) \star (y\ ;\ z))$
assumes *while-sumstar*: $(x + y) \star z = (x \star y) \star (x \star z)$
assumes *while-left-dist-add*: $x \star (y + z) = (x \star y) + (x \star z)$
assumes *while-sub-associative*: $(x \star y)\ ;\ z \leq x \star (y\ ;\ z)$
assumes *while-simulate-left-plus*: $x\ ;\ z \leq z\ ;\ (y \star 1) + w \longrightarrow x \star (z\ ;\ v) \leq z\ ;\ (y \star v) + (x \star (w\ ;\ (y \star v)))$
assumes *while-simulate-right-plus*: $z\ ;\ x \leq y\ ;\ (y \star z) + w \longrightarrow z\ ;\ (x \star v) \leq y \star (z\ ;\ v + w\ ;\ (x \star v))$

begin

— Theorem 9.1

lemma *while-zero*: $0 \star x = x$
by (*metis add-right-zero mult-left-zero while-productstar*)

— Theorem 9.4

lemma *while-mult-increasing*: $x\ ;\ y \leq x \star y$
by (*metis add-least-upper-bound mult-left-one order-refl while-productstar*)

— Theorem 9.2

lemma *while-one-increasing*: $x \leq x \star 1$
by (*metis mult-right-one while-mult-increasing*)

— Theorem 9.3

lemma *while-increasing*: $y \leq x \star y$
by (*metis add-left-divisibility mult-left-one while-productstar*)

— Theorem 9.42

lemma *while-right-isotone*: $y \leq z \longrightarrow x \star y \leq x \star z$
by (*metis less-eq-def while-left-dist-add*)

— Theorem 9.41

lemma *while-left-isotone*: $x \leq y \longrightarrow x \star z \leq y \star z$
by (*metis less-eq-def while-increasing while-sumstar*)

lemma *while-isotone*: $w \leq x \wedge y \leq z \longrightarrow w \star y \leq x \star z$
by (*smt order-trans while-left-isotone while-right-isotone*)

— Theorem 9.17

lemma *while-left-unfold*: $x \star y = y + x\ ;\ (x \star y)$
by (*metis mult-left-one mult-right-one while-productstar*)

lemma *while-simulate-left-plus-1*: $x\ ;\ z \leq z\ ;\ (y \star 1) \longrightarrow x \star (z\ ;\ w) \leq z\ ;\ (y \star w) + (x \star 0)$
by (*metis add-right-zero mult-left-zero while-simulate-left-plus*)

— Theorem 11.1

lemma *while-simulate-absorb*: $y\ ;\ x \leq x \longrightarrow y \star x \leq x + (y \star 0)$
by (*metis while-simulate-left-plus-1 while-zero mult-right-one*)

— Theorem 9.10

lemma *while-transitive*: $x \star (x \star y) = x \star y$
by (*metis add-right-upper-bound add-right-zero antisym while-increasing while-left-dist-add while-left-unfold while-simulate-absorb*)

— Theorem 9.25

lemma *while-slide*: $(x\ ;\ y) \star (x\ ;\ z) = x\ ;\ ((y\ ;\ x) \star z)$
by (*metis mult-associative mult-left-dist-add while-left-unfold while-productstar*)

— Theorem 9.21

lemma *while-zero-2*: $(x\ ;\ 0) \star y = x\ ;\ 0 + y$
by (*metis add-commutative mult-associative mult-left-zero while-left-unfold*)

— Theorem 9.5

lemma *while-mult-star-exchange*: $x\ ;\ (x \star y) = x \star (x\ ;\ y)$
by (*metis mult-left-one while-slide*)

— Theorem 9.18

lemma *while-right-unfold*: $x \star y = y + (x \star (x\ ;\ y))$
by (*metis while-left-unfold while-mult-star-exchange*)

— Theorem 9.7

lemma *while-one-mult-below*: $(x \star 1)\ ;\ y \leq x \star y$
by (*metis mult-left-one while-sub-associative*)

lemma *while-plus-one*: $x \star y = y + (x \star y)$
by (*metis less-eq-def while-increasing*)

— Theorem 9.19

lemma *while-rtc-2*: $y + x\ ;\ y + (x \star (x \star y)) = x \star y$
by (*metis add-associative less-eq-def while-mult-increasing while-plus-one while-transitive*)

— Theorem 9.6

lemma *while-left-plus-below*: $x\ ;\ (x \star y) \leq x \star y$
by (*metis add-right-divisibility while-left-unfold*)

lemma *while-right-plus-below*: $x \star (x\ ;\ y) \leq x \star y$
by (*metis while-left-plus-below while-mult-star-exchange*)

lemma *while-right-plus-below-2*: $(x \star x)\ ;\ y \leq x \star y$
by (*smt order-trans while-right-plus-below while-sub-associative*)

— Theorem 9.47

lemma *while-mult-transitive*: $x \leq z \star y \wedge y \leq z \star w \longrightarrow x \leq z \star w$
by (*smt order-trans while-right-isotone while-transitive*)

— Theorem 9.48

lemma *while-mult-upper-bound*: $x \leq z \star 1 \wedge y \leq z \star w \longrightarrow x\ ;\ y \leq z \star w$
by (*metis less-eq-def mult-right-sub-dist-add-left order-trans while-mult-transitive while-one-mult-below*)

lemma *while-one-mult-while-below*: $(y \star 1)\ ;\ (y \star v) \leq y \star v$
by (*metis order-refl while-mult-upper-bound*)

— Theorem 9.34

lemma *while-sub-dist*: $x \star z \leq (x + y) \star z$
by (*metis add-left-upper-bound while-left-isotone*)

lemma *while-sub-dist-1*: $x\ ;\ z \leq (x + y) \star z$
by (*metis order-trans while-mult-increasing while-sub-dist*)

lemma *while-sub-dist-2*: $x\ ;\ y\ ;\ z \leq (x + y) \star z$
by (*metis add-commutative mult-associative while-mult-transitive while-sub-dist-1*)

— Theorem 9.36

lemma *while-sub-dist-3*: $x \star (y \star z) \leq (x + y) \star z$
by (*metis add-right-upper-bound while-left-isotone while-mult-transitive while-sub-dist*)

— Theorem 9.44

lemma *while-absorb-2*: $x \leq y \longrightarrow y \star (x \star z) = y \star z$
by (*metis add-commutative less-eq-def while-left-dist-add while-plus-one while-sub-dist-3*)

lemma *while-simulate-right-plus-1*: $z \ ; x \leq y \ ; (y \star z) \longrightarrow z \ ; (x \star w) \leq y \star (z \ ; w)$
by (*metis add-right-zero mult-left-zero while-simulate-right-plus*)

— Theorem 9.39

lemma *while-sumstar-1-below*: $x \star ((y \ ; (x \star 1)) \star z) \leq ((x \star 1) \ ; y) \star (x \star z)$
proof −
have *1*: $x \ ; (((x \star 1) \ ; y) \star (x \star z)) \leq ((x \star 1) \ ; y) \star (x \star z)$
by (*smt add-isotone add-right-upper-bound mult-associative mult-left-dist-add mult-right-sub-dist-add-right while-left-unfold*)
have $x \star ((y \ ; (x \star 1)) \star z) \leq (x \star z) + (x \star (y \ ; (((x \star 1) \ ; y) \star ((x \star 1) \ ; z))))$
by (*metis eq-refl while-left-dist-add while-productstar*)
also have ... $\leq (x \star z) + (x \star ((x \star 1) \ ; y \ ; (((x \star 1) \ ; y) \star ((x \star 1) \ ; z))))$
by (*metis add-right-isotone mult-associative mult-left-one mult-right-sub-dist-add-left while-left-unfold while-right-isotone*)
also have ... $\leq (x \star z) + (x \star (((x \star 1) \ ; y) \star ((x \star 1) \ ; z)))$
by (*metis add-right-isotone add-right-upper-bound while-left-unfold while-right-isotone*)
also have ... $\leq x \star (((x \star 1) \ ; y) \star (x \star z))$
by (*smt add-associative add-left-upper-bound less-eq-def mult-left-one while-left-dist-add while-left-unfold while-sub-associative*)
also have ... $\leq (((x \star 1) \ ; y) \star (x \star z)) + (x \star 0)$ **using** *1*
by (*metis while-simulate-absorb*)
also have ... $= ((x \star 1) \ ; y) \star (x \star z)$
by (*smt add-associative add-commutative add-left-zero while-left-dist-add while-left-unfold*)
finally show *?thesis*
.
qed

lemma *while-sumstar-2-below*: $((x \star 1) \ ; y) \star (x \star z) \leq (x \star y) \star (x \star z)$
by (*metis mult-left-one while-left-isotone while-sub-associative*)

— Theorem 9.38

lemma *while-add-1-below*: $x \star ((y \ ; (x \star 1)) \star z) \leq (x + y) \star z$
proof −
have $((x \star 1) \ ; y) \star ((x \star 1) \ ; z) \leq (x + y) \star z$
by (*metis while-isotone while-one-mult-below while-sumstar*)
hence $(y \ ; (x \star 1)) \star z \leq z + y \ ; ((x + y) \star z)$
by (*metis add-right-isotone mult-right-isotone while-productstar*)
also have ... $\leq (x + y) \star z$
by (*metis add-right-isotone add-right-upper-bound mult-left-isotone while-left-unfold*)
finally show *?thesis*
by (*metis add-commutative add-right-upper-bound while-isotone while-transitive*)
qed

— Theorem 9.16

lemma *while-while-while*: $((x \star 1) \star 1) \star y = (x \star 1) \star y$
by (*smt add-commutative less-eq-def mult-right-one while-left-plus-below while-mult-star-exchange while-plus-one while-sumstar while-transitive*)

lemma *while-one*: $(1 \star 1) \star y = 1 \star y$
by (*metis while-while-while while-zero*)

— Theorem 9.22

lemma *while-add-below*: $x + y \leq x \star (y \star 1)$
by (*smt add-commutative add-isotone case-split-right order-trans while-increasing while-left-plus-below while-mult-increasing while-plus-one*)

— Theorem 9.32

lemma *while-add-2*: $(x + y) \star z \leq (x \star (y \star 1)) \star z$

by (*metis while-add-below while-left-isotone*)

— Theorem 9.45

lemma *while-sup-one-left-unfold*: $1 \leq x \longrightarrow x \,;\, (x \star y) = x \star y$
by (*metis less-eq-def mult-left-one mult-right-dist-add while-mult-star-exchange while-right-unfold while-transitive*)

lemma *while-sup-one-right-unfold*: $1 \leq x \longrightarrow x \star (x \,;\, y) = x \star y$
by (*metis while-mult-star-exchange while-sup-one-left-unfold*)

— Theorem 9.30

lemma *while-decompose-7*: $(x + y) \star z = x \star (y \star ((x + y) \star z))$
by (*metis eq-iff order-trans while-increasing while-sub-dist-3 while-transitive*)

— Theorem 9.31

lemma *while-decompose-8*: $(x + y) \star z = (x + y) \star (x \star (y \star z))$
by (*metis add-commutative while-sumstar while-transitive*)

— Theorem 9.27

lemma *while-decompose-9*: $(x \star (y \star 1)) \star z = x \star (y \star ((x \star (y \star 1)) \star z))$
by (*smt add-commutative less-eq-def order-trans while-add-below while-increasing while-sub-dist-3*)

lemma *while-decompose-10*: $(x \star (y \star 1)) \star z = (x \star (y \star 1)) \star (x \star (y \star z))$
proof −
have *1*: $(x \star (y \star 1)) \star z \leq (x \star (y \star 1)) \star (x \star (y \star z))$
by (*metis add-associative less-eq-def while-left-dist-add while-plus-one*)
have $x + (y \star 1) \leq x \star (y \star 1)$
by (*metis add-least-upper-bound while-add-below while-increasing*)
hence $(x \star (y \star 1)) \star (x \star (y \star z)) \leq (x \star (y \star 1)) \star z$
by (*smt add-least-upper-bound eq-refl order-trans while-absorb-2 while-one-increasing*)
thus *?thesis* **using** *1*
by (*metis antisym*)
qed

lemma *while-back-loop-fixpoint*: $z \,;\, (y \star (y \,;\, x)) + z \,;\, x = z \,;\, (y \star x)$
by (*metis add-commutative mult-left-dist-add while-right-unfold*)

lemma *while-back-loop-prefixpoint*: $z \,;\, (y \star 1) \,;\, y + z \leq z \,;\, (y \star 1)$
by (*metis add-least-upper-bound mult-associative mult-right-isotone mult-right-one order-refl while-increasing while-mult-upper-bound while-one-increasing*)

— Theorem 9

lemma *while-loop-is-fixpoint*: *is-fixpoint* $(\lambda x \,.\, y \,;\, x + z)\ (y \star z)$
by (*smt add-commutative is-fixpoint-def while-left-unfold*)

— Theorem 9

lemma *while-back-loop-is-prefixpoint*: *is-prefixpoint* $(\lambda x \,.\, x \,;\, y + z)\ (z \,;\, (y \star 1))$
by (*metis is-prefixpoint-def while-back-loop-prefixpoint*)

— Theorem 9.20

lemma *while-while-add*: $(1 + x) \star y = (x \star 1) \star y$
by (*metis add-commutative while-decompose-10 while-sumstar while-zero*)

lemma *while-while-mult-sub*: $x \star (1 \star y) \leq (x \star 1) \star y$
by (*metis add-commutative while-sub-dist-3 while-while-add*)

— Theorem 9.11

lemma *while-right-plus*: $(x \star x) \star y = x \star y$
by (*metis add-idempotent while-plus-one while-sumstar while-transitive*)

— Theorem 9.12

lemma *while-left-plus*: $(x\ ;\ (x \star 1)) \star y = x \star y$
by (*metis mult-right-one while-mult-star-exchange while-right-plus*)

— Theorem 9.9

lemma *while-below-while-one*: $x \star x \leq x \star 1$
by (*metis while-one-increasing while-right-plus*)

lemma *while-below-while-one-mult*: $x\ ;\ (x \star x) \leq x\ ;\ (x \star 1)$
by (*metis mult-right-isotone while-below-while-one*)

— Theorem 9.23

lemma *while-add-sub-add-one*: $x \star (x + y) \leq x \star (1 + y)$
by (*metis add-left-isotone while-below-while-one while-left-dist-add*)

lemma *while-add-sub-add-one-mult*: $x\ ;\ (x \star (x + y)) \leq x\ ;\ (x \star (1 + y))$
by (*metis mult-right-isotone while-add-sub-add-one*)

lemma *while-elimination*: $x\ ;\ y = 0 \longrightarrow x\ ;\ (y \star z) = x\ ;\ z$
by (*metis add-right-zero mult-associative mult-left-dist-add mult-left-zero while-left-unfold*)

— Theorem 9.8

lemma *while-square*: $(x\ ;\ x) \star y \leq x \star y$
by (*metis while-left-isotone while-mult-increasing while-right-plus*)

— Theorem 9.35

lemma *while-mult-sub-add*: $(x\ ;\ y) \star z \leq (x + y) \star z$
by (*metis while-increasing while-isotone while-mult-increasing while-sumstar*)

— Theorem 9.43

lemma *while-absorb-1*: $x \leq y \longrightarrow x \star (y \star z) = y \star z$
by (*metis antisym less-eq-def while-increasing while-sub-dist-3*)

lemma *while-absorb-3*: $x \leq y \longrightarrow x \star (y \star z) = y \star (x \star z)$
by (*metis while-absorb-1 while-absorb-2*)

— Theorem 9.24

lemma *while-square-2*: $(x\ ;\ x) \star ((x + 1)\ ;\ y) \leq x \star y$
by (*metis add-least-upper-bound while-increasing while-mult-transitive while-mult-upper-bound while-one-increasing while-square*)

lemma *while-separate-unfold-below*: $(y\ ;\ (x \star 1)) \star z \leq (y \star z) + (y \star (y\ ;\ x\ ;\ (x \star ((y\ ;\ (x \star 1)) \star z))))$
proof −
have $(y\ ;\ (x \star 1)) \star z = (y \star (y\ ;\ x\ ;\ (x \star 1))) \star (y \star z)$
by (*metis mult-associative mult-left-dist-add mult-right-one while-left-unfold while-sumstar*)
hence $(y\ ;\ (x \star 1)) \star z = (y \star z) + (y \star (y\ ;\ x\ ;\ (x \star 1)))\ ;\ ((y\ ;\ (x \star 1)) \star z)$
by (*metis while-left-unfold*)
also have $\ldots \leq (y \star z) + (y \star (y\ ;\ x\ ;\ (x \star 1))\ ;\ ((y\ ;\ (x \star 1)) \star z))$
by (*metis add-right-isotone while-sub-associative*)
also have $\ldots \leq (y \star z) + (y \star (y\ ;\ x\ ;\ (x \star ((y\ ;\ (x \star 1)) \star z))))$
by (*smt add-right-isotone mult-associative mult-right-isotone while-one-mult-below while-right-isotone*)
finally show *?thesis*
.
qed

— Theorem 9.33

lemma *while-mult-zero-add*: $(x + y\ ;\ 0) \star z = x \star ((y\ ;\ 0) \star z)$
proof −
have $(x + y\ ;\ 0) \star z = (x \star (y\ ;\ 0)) \star (x \star z)$
by (*metis while-sumstar*)
also have $\ldots = (x \star z) + (x \star (y\ ;\ 0))\ ;\ ((x \star (y\ ;\ 0)) \star (x \star z))$
by (*metis while-left-unfold*)
also have $\ldots \leq (x \star z) + (x \star (y\ ;\ 0))$

by (*metis add-right-isotone mult-associative mult-left-zero while-sub-associative*)
also have ... = $x \star ((y\ ;\ 0) \star z)$
by (*metis add-commutative while-left-dist-add while-zero-2*)
finally show *?thesis*
by (*metis le-neq-trans less-def while-sub-dist-3*)
qed

lemma *while-add-mult-zero*: $(x + y\ ;\ 0) \star y = x \star y$
by (*metis less-eq-def while-mult-zero-add while-zero-2 zero-right-mult-decreasing*)

lemma *while-mult-zero-add-2*: $(x + y\ ;\ 0) \star z = (x \star z) + (x \star (y\ ;\ 0))$
by (*metis add-commutative while-left-dist-add while-mult-zero-add while-zero-2*)

lemma *while-add-zero-star*: $(x + y\ ;\ 0) \star z = x \star (y\ ;\ 0 + z)$
by (*metis while-mult-zero-add while-zero-2*)

lemma *while-unfold-sum*: $(x + y) \star z = (x \star z) + (x \star (y\ ;\ ((x + y) \star z)))$
apply (*rule antisym*)
apply (*smt add-associative less-eq-def while-absorb-1 while-increasing while-mult-star-exchange while-right-unfold while-sub-associative while-sumstar*)
apply (*metis add-least-upper-bound while-decompose-7 while-mult-increasing while-right-isotone while-sub-dist*)
done

lemma *while-simulate-left*: $x\ ;\ z \leq z\ ;\ y + w \longrightarrow x \star (z\ ;\ v) \leq z\ ;\ (y \star v) + (x \star (w\ ;\ (y \star v)))$
by (*metis add-left-isotone mult-right-isotone order-trans while-one-increasing while-simulate-left-plus*)

lemma *while-simulate-right*: $z\ ;\ x \leq y\ ;\ z + w \longrightarrow z\ ;\ (x \star v) \leq y \star (z\ ;\ v + w\ ;\ (x \star v))$
proof −
have $y\ ;\ z + w \leq y\ ;\ (y \star z) + w$
by (*metis add-left-isotone mult-right-isotone while-increasing*)
thus *?thesis*
by (*smt order-trans while-simulate-right-plus*)
qed

lemma *while-simulate*: $z\ ;\ x \leq y\ ;\ z \longrightarrow z\ ;\ (x \star v) \leq y \star (z\ ;\ v)$
by (*metis add-right-zero mult-left-zero while-simulate-right*)

— Theorem 9.14

lemma *while-while-mult*: $1 \star (x \star y) = (x \star 1) \star y$
proof −
have $(x \star 1) \star y \leq (x \star 1)\ ;\ ((x \star 1) \star y)$
by (*metis order-refl while-increasing while-sup-one-left-unfold*)
also have ... $\leq 1 \star ((x \star 1)\ ;\ y)$
by (*metis mult-left-one order-refl while-mult-upper-bound while-simulate*)
also have ... $\leq 1 \star (x \star y)$
by (*metis while-one-mult-below while-right-isotone*)
finally show *?thesis*
by (*metis antisym while-sub-dist-3 while-while-add*)
qed

lemma *while-simulate-left-1*: $x\ ;\ z \leq z\ ;\ y \longrightarrow x \star (z\ ;\ v) \leq z\ ;\ (y \star v) + (x \star 0)$
by (*metis add-right-zero mult-left-zero while-simulate-left*)

— Theorem 9.46

lemma *while-associative-1*: $1 \leq z \longrightarrow x \star (y\ ;\ z) = (x \star y)\ ;\ z$
proof
assume *1*: $1 \leq z$
have $x \star (y\ ;\ z) \leq x \star ((x \star y)\ ;\ z)$
by (*metis less-eq-def mult-right-dist-add while-plus-one while-right-isotone*)
also have ... $\leq (x \star y)\ ;\ (0 \star z) + (x \star 0)$
by (*metis mult-associative mult-right-sub-dist-add-right while-left-unfold while-simulate-absorb while-zero*)
also have ... $\leq (x \star y)\ ;\ z + (x \star 0)$ **using** *1*
by (*metis add-least-upper-bound add-left-upper-bound add-right-upper-bound case-split-right while-plus-one while-zero*)
also have ... = $(x \star y)\ ;\ z$
by (*metis add-right-zero mult-right-dist-add while-left-dist-add*)
finally show $x \star (y\ ;\ z) = (x \star y)\ ;\ z$
by (*metis antisym while-sub-associative*)

qed

— Theorem 9.29

lemma *while-associative-while-1*: $x \star (y \ ; (z \star 1)) = (x \star y) \ ; (z \star 1)$
by (*metis while-associative-1 while-increasing*)

— Theorem 9.13

lemma *while-one-while*: $(x \star 1) \ ; (y \star 1) = x \star (y \star 1)$
by (*metis mult-left-one while-associative-while-1*)

lemma *while-decompose-5-below*: $(x \star (y \star 1)) \star z \leq (y \star (x \star 1)) \star z$
by (*smt add-commutative mult-left-dist-add mult-right-one while-increasing while-left-unfold while-mult-star-exchange while-one-while while-plus-one while-sumstar*)

— Theorem 9.26

lemma *while-decompose-5*: $(x \star (y \star 1)) \star z = (y \star (x \star 1)) \star z$
by (*metis antisym while-decompose-5-below*)

lemma *while-decompose-4*: $(x \star (y \star 1)) \star z = x \star ((y \star (x \star 1)) \star z)$
by (*metis while-decompose-5 while-decompose-9 while-transitive*)

— Theorem 11.7

lemma *while-simulate-2*: $y \ ; (x \star 1) \leq x \star (y \star 1) \longleftrightarrow y \star (x \star 1) \leq x \star (y \star 1)$
proof (*rule iffI*)
assume $y \ ; (x \star 1) \leq x \star (y \star 1)$
hence $y \ ; (x \star 1) \leq (x \star 1) \ ; (y \star 1)$
by (*metis while-one-while*)
hence $y \star ((x \star 1) \ ; 1) \leq (x \star 1) \ ; (y \star 1) + (y \star 0)$
by (*metis while-simulate-left-plus-1*)
hence $y \star (x \star 1) \leq (x \star (y \star 1)) + (y \star 0)$
by (*metis mult-right-one while-one-while*)
also have $... = x \star (y \star 1)$
by (*metis add-commutative less-eq-def order-trans while-increasing while-right-isotone zero-least*)
finally show $y \star (x \star 1) \leq x \star (y \star 1)$
.
next
assume $y \star (x \star 1) \leq x \star (y \star 1)$
thus $y \ ; (x \star 1) \leq x \star (y \star 1)$
by (*metis order-trans while-mult-increasing*)
qed

lemma *while-simulate-1*: $y \ ; x \leq x \ ; y \longrightarrow y \star (x \star 1) \leq x \star (y \star 1)$
by (*metis order-trans while-mult-increasing while-right-isotone while-simulate while-simulate-2*)

lemma *while-simulate-3*: $y \ ; (x \star 1) \leq x \star 1 \longrightarrow y \star (x \star 1) \leq x \star (y \star 1)$
by (*metis add-idempotent case-split-right while-increasing while-mult-upper-bound while-simulate-2*)

— Theorem 9.28

lemma *while-extra-while*: $(y \ ; (x \star 1)) \star z = (y \ ; (y \star (x \star 1))) \star z$
proof −
have $y \ ; (y \star (x \star 1)) \leq y \ ; (x \star 1) \ ; (y \ ; (x \star 1) \star 1)$
by (*smt add-commutative add-left-upper-bound mult-right-one order-trans while-back-loop-prefixpoint while-left-isotone while-mult-star-exchange*)
hence *1*: $(y \ ; (y \star (x \star 1))) \star z \leq (y \ ; (x \star 1)) \star z$
by (*metis while-simulate-right-plus-1 mult-left-one*)
have $(y \ ; (x \star 1)) \star z \leq (y \ ; (y \star (x \star 1))) \star z$
by (*metis while-increasing while-left-isotone while-mult-star-exchange*)
thus *?thesis* **using** *1*
by (*metis antisym*)
qed

— Theorem 11.6

lemma *while-separate-4*: $y \ ; x \leq x \ ; (x \star (1 + y)) \longrightarrow (x + y) \star z = x \star (y \star z)$

proof
assume *1*: $y \ ; \ x \leq x \ ; \ (x \star (1 + y))$
hence $(1 + y) \ ; \ x \leq x \ ; \ (x \star (1 + y))$
by (*smt add-associative add-least-upper-bound mult-left-one mult-left-sub-dist-add-left mult-right-dist-add mult-right-one while-left-unfold*)
hence *2*: $(1 + y) \ ; \ (x \star 1) \leq x \star (1 + y)$
by (*metis mult-right-one while-simulate-right-plus-1*)
have $y \ ; \ x \ ; \ (x \star 1) \leq x \ ; \ (x \star ((1 + y) \ ; \ (x \star 1)))$ **using** *1*
by (*smt less-eq-def mult-associative mult-right-dist-add while-associative-1 while-increasing*)
also have ... $\leq x \ ; \ (x \star (1 + y))$ **using** *2*
by (*metis mult-right-isotone order-refl while-mult-transitive*)
also have ... $\leq x \ ; \ (x \star 1) \ ; \ (y \star 1)$
by (*metis add-least-upper-bound mult-associative mult-right-isotone while-increasing while-one-increasing while-one-while while-right-isotone*)
finally have $y \star (x \ ; \ (x \star 1)) \leq x \ ; \ (x \star 1) \ ; \ (y \star 1) + (y \star 0)$
by (*metis mult-associative mult-right-one while-simulate-left-plus-1*)
hence $(y \star 1) \ ; \ (y \star x) \leq x \ ; \ (x \star y \star 1) + (y \star 0)$
by (*smt less-eq-def mult-associative mult-right-one order-refl order-trans while-absorb-2 while-left-dist-add while-mult-star-exchange while-one-mult-below while-one-while while-plus-one*)
hence $(y \star 1) \ ; \ ((y \star x) \star (y \star z)) \leq x \star ((y \star 1) \ ; \ (y \star z) + (y \star 0) \ ; \ ((y \star x) \star (y \star z)))$
by (*metis while-simulate-right-plus*)
also have ... $\leq x \star ((y \star z) + (y \star 0))$
by (*metis add-isotone mult-left-zero order-refl while-absorb-2 while-one-mult-below while-right-isotone while-sub-associative*)
also have ... $= x \star y \star z$
by (*metis add-right-zero while-left-dist-add*)
finally show $(x + y) \star z = x \star (y \star z)$
by (*smt add-commutative less-eq-def mult-left-one mult-right-dist-add while-plus-one while-sub-associative while-sumstar*)
qed

lemma *while-separate-5*: $y \ ; \ x \leq x \ ; \ (x \star (x + y)) \longrightarrow (x + y) \star z = x \star (y \star z)$
by (*smt order-trans while-add-sub-add-one-mult while-separate-4*)

lemma *while-separate-6*: $y \ ; \ x \leq x \ ; \ (x + y) \longrightarrow (x + y) \star z = x \star (y \star z)$
by (*smt order-trans while-increasing while-mult-star-exchange while-separate-5*)

— Theorem 11.4

lemma *while-separate-1*: $y \ ; \ x \leq x \ ; \ y \longrightarrow (x + y) \star z = x \star (y \star z)$
by (*metis add-least-upper-bound less-eq-def mult-left-sub-dist-add-right while-separate-6*)

— Theorem 11.2

lemma *while-separate-mult-1*: $y \ ; \ x \leq x \ ; \ y \longrightarrow (x \ ; \ y) \star z \leq x \star (y \star z)$
by (*metis while-mult-sub-add while-separate-1*)

— Theorem 11.5

lemma *separation*: $y \ ; \ x \leq x \ ; \ (y \star 1) \longrightarrow (x + y) \star z = x \star (y \star z)$
proof
assume $y \ ; \ x \leq x \ ; \ (y \star 1)$
hence $y \star x \leq x \ ; \ (y \star 1) + (y \star 0)$
by (*metis mult-right-one while-simulate-left-plus-1*)
also have ... $\leq x \ ; \ (x \star y \star 1) + (y \star 0)$
by (*metis add-left-isotone while-increasing while-mult-star-exchange*)
finally have $(y \star 1) \ ; \ (y \star x) \leq x \ ; \ (x \star y \star 1) + (y \star 0)$
by (*metis order-refl order-trans while-absorb-2 while-one-mult-below*)
hence $(y \star 1) \ ; \ ((y \star x) \star (y \star z)) \leq x \star ((y \star 1) \ ; \ (y \star z) + (y \star 0) \ ; \ ((y \star x) \star (y \star z)))$
by (*metis while-simulate-right-plus*)
also have ... $\leq x \star ((y \star z) + (y \star 0))$
by (*metis add-isotone mult-left-zero order-refl while-absorb-2 while-one-mult-below while-right-isotone while-sub-associative*)
also have ... $= x \star y \star z$
by (*metis add-right-zero while-left-dist-add*)
finally show $(x + y) \star z = x \star (y \star z)$
by (*smt add-commutative less-eq-def mult-left-one mult-right-dist-add while-plus-one while-sub-associative while-sumstar*)
qed

— Theorem 11.5

lemma *while-separate-left*: $y \ ; \ x \leq x \ ; \ (y \star 1) \longrightarrow y \star (x \star z) \leq x \star (y \star z)$

by (*metis add-commutative separation while-sub-dist-3*)

— Theorem 11.6

lemma *while-simulate-4*: $y\ ;\ x \leq x\ ;\ (x \star (1 + y)) \longrightarrow y \star (x \star z) \leq x \star (y \star z)$
by (*metis add-commutative while-separate-4 while-sub-dist-3*)

lemma *while-simulate-5*: $y\ ;\ x \leq x\ ;\ (x \star (x + y)) \longrightarrow y \star (x \star z) \leq x \star (y \star z)$
by (*smt order-trans while-add-sub-add-one-mult while-simulate-4*)

lemma *while-simulate-6*: $y\ ;\ x \leq x\ ;\ (x + y) \longrightarrow y \star (x \star z) \leq x \star (y \star z)$
by (*smt order-trans while-increasing while-mult-star-exchange while-simulate-5*)

— Theorem 11.3

lemma *while-simulate-7*: $y\ ;\ x \leq x\ ;\ y \longrightarrow y \star (x \star z) \leq x \star (y \star z)$
by (*metis add-commutative mult-left-sub-dist-add-left order-trans while-simulate-6*)

lemma *while-while-mult-1*: $x \star (1 \star y) = 1 \star (x \star y)$
by (*metis add-commutative mult-left-one mult-right-one order-refl while-separate-1*)

— Theorem 9.15

lemma *while-while-mult-2*: $x \star (1 \star y) = (x \star 1) \star y$
by (*metis while-while-mult while-while-mult-1*)

— Theorem 11.8

lemma *while-import*: $p \leq p\ ;\ p \wedge p \leq 1 \wedge p\ ;\ x \leq x\ ;\ p \longrightarrow p\ ;\ (x \star y) = p\ ;\ ((p\ ;\ x) \star y)$
proof
assume *1*: $p \leq p\ ;\ p \wedge p \leq 1 \wedge p\ ;\ x \leq x\ ;\ p$
hence $p\ ;\ (x \star y) \leq (p\ ;\ x) \star (p\ ;\ y)$
by (*smt add-commutative less-eq-def mult-associative mult-left-dist-add mult-right-one while-simulate*)
also have ... $\leq (p\ ;\ x) \star y$ **using** *1*
by (*metis less-eq-def mult-left-one mult-right-dist-add while-right-isotone*)
finally have *2*: $p\ ;\ (x \star y) \leq p\ ;\ ((p\ ;\ x) \star y)$ **using** *1*
by (*smt add-commutative less-eq-def mult-associative mult-left-dist-add mult-right-one*)
have $p\ ;\ ((p\ ;\ x) \star y) \leq p\ ;\ (x \star y)$ **using** *1*
by (*metis mult-left-isotone mult-left-one mult-right-isotone while-left-isotone*)
thus $p\ ;\ (x \star y) = p\ ;\ ((p\ ;\ x) \star y)$ **using** *2*
by (*metis antisym*)
qed

— Theorem 11.8

lemma *while-preserve*: $p \leq p\ ;\ p \wedge p \leq 1 \wedge p\ ;\ x \leq x\ ;\ p \longrightarrow p\ ;\ (x \star y) = p\ ;\ (x \star (p\ ;\ y))$
apply *rule*
apply (*rule antisym*)
apply (*metis mult-associative mult-left-isotone mult-right-isotone order-trans while-simulate*)
apply (*metis mult-left-isotone mult-left-one mult-right-isotone while-right-isotone*)
done

lemma *while-plus-below-while*: $(x \star 1)\ ;\ x \leq x \star 1$
by (*metis order-refl while-mult-upper-bound while-one-increasing*)

— Theorem 9.40

lemma *while-01*: $(w\ ;\ (x \star 1)) \star (y\ ;\ z) \leq (x \star w) \star ((x \star y)\ ;\ z)$
proof −
have $(w\ ;\ (x \star 1)) \star (y\ ;\ z) = y\ ;\ z + w\ ;\ (((x \star 1)\ ;\ w) \star ((x \star 1)\ ;\ y\ ;\ z))$
by (*metis mult-associative while-productstar*)
also have ... $\leq y\ ;\ z + w\ ;\ ((x \star w) \star ((x \star y)\ ;\ z))$
by (*metis add-right-isotone mult-left-isotone mult-right-isotone while-isotone while-one-mult-below*)
also have ... $\leq (x \star y)\ ;\ z + (x \star w)\ ;\ ((x \star w) \star ((x \star y)\ ;\ z))$
by (*metis add-isotone mult-right-sub-dist-add-left while-left-unfold*)
finally show *?thesis*
by (*metis while-left-unfold*)
qed

— Theorem 9.37

lemma *while-while-sub-associative*: $x \star (y \star z) \leq ((x \star y) \star z) + (x \star z)$
proof −
have *1*: $x \ ; (x \star 1) \leq (x \star 1) \ ; ((x \star y) \star 1)$
by (*metis add-least-upper-bound order-trans while-back-loop-prefixpoint while-left-plus-below*)
have $x \star (y \star z) \leq x \star ((x \star 1) \ ; (y \star z))$
by (*metis mult-left-isotone mult-left-one while-increasing while-right-isotone*)
also have ... $\leq (x \star 1) \ ; ((x \star y) \star (y \star z)) + (x \star 0)$ **using** *1*
by (*metis while-simulate-left-plus-1*)
also have ... $\leq (x \star 1) \ ; ((x \star y) \star z) + (x \star z)$
by (*metis add-isotone order-refl while-absorb-2 while-increasing while-right-isotone zero-least*)
also have ... $= (x \star 1) \ ; z + (x \star 1) \ ; (x \star y) \ ; ((x \star y) \star z) + (x \star z)$
by (*metis mult-associative mult-left-dist-add while-left-unfold*)
also have ... $= (x \star y) \ ; ((x \star y) \star z) + (x \star z)$
by (*smt add-associative add-commutative less-eq-def mult-left-one mult-right-dist-add order-refl while-absorb-1 while-plus-one while-sub-associative*)
also have ... $\leq ((x \star y) \star z) + (x \star z)$
by (*metis add-left-isotone while-left-plus-below*)
finally show *?thesis*
.
qed

lemma *while-induct*: $x \ ; z \leq z \wedge y \leq z \wedge x \star 1 \leq z \longrightarrow x \star y \leq z$
by (*metis add-commutative add-least-upper-bound add-left-zero less-eq-def while-right-isotone while-simulate-absorb*)

lemma *while-sumstar-4-below*: $(x \star y) \star ((x \star 1) \ ; z) \leq x \star ((y \ ; (x \star 1)) \star z)$ **oops**
lemma *while-sumstar-2*: $(x + y) \star z = x \star ((y \ ; (x \star 1)) \star z)$ **oops**
lemma *while-sumstar-3*: $(x + y) \star z = ((x \star 1) \ ; y) \star (x \star z)$ **oops**
lemma *while-decompose-6*: $x \star ((y \ ; (x \star 1)) \star z) = y \star ((x \ ; (y \star 1)) \star z)$ **oops**
lemma *while-finite-associative*: $x \star 0 = 0 \longrightarrow (x \star y) \ ; z = x \star (y \ ; z)$ **oops**
lemma *atomicity-refinement*: $s = s \ ; q \wedge x = q \ ; x \wedge q \ ; b = 0 \wedge r \ ; b \leq b \ ; r \wedge r \ ; l \leq l \ ; r \wedge x \ ; l \leq l \ ; x \wedge b \ ; l \leq l \ ; b \wedge q \ ; l \leq l \ ; q \wedge r \star q \leq q \ ; (r \star 1) \wedge q \leq 1 \longrightarrow s \ ; ((x + b + r + l) \star (q \ ; z)) \leq s \ ; ((x \ ; (b \star q) + r + l) \star z)$ **oops**

lemma *while-separate-right-plus*: $y \ ; x \leq x \ ; (x \star (1 + y)) + 1 \longrightarrow y \star (x \star z) \leq x \star (y \star z)$ **oops**
lemma *while-square-1*: $x \star 1 = (x \ ; x) \star (x + 1)$ **oops**
lemma *while-absorb-below-one*: $y \ ; x \leq x \longrightarrow y \star x \leq 1 \star x$ **oops**
lemma $y \star (x \star 1) \leq x \star (y \star 1) \longrightarrow (x + y) \star 1 = x \star (y \star 1)$ **oops**
lemma $y \ ; x \leq (1 + x) \ ; (y \star 1) \longrightarrow (x + y) \star 1 = x \star (y \star 1)$ **oops**

end

class *bounded-binary-itering* = *bounded-idempotent-left-zero-semiring* + *binary-itering*

begin

— Theorem 9

lemma *while-right-top*: $x \star T = T$
by (*metis add-left-top while-left-unfold*)

— Theorem 9

lemma *while-left-top*: $T \ ; (x \star 1) = T$
by (*metis add-right-top antisym top-greatest while-back-loop-prefixpoint*)

— Theorem 10.10 counterexamples

lemma *while-sum-below-one*: $y \ ; ((x + y) \star z) \leq (y \ ; (x \star 1)) \star z$ **nitpick** [*expect=genuine*] **oops**
lemma *while-denest-1*: $w \ ; (x \star (y \ ; z)) \leq (w \ ; (x \star y)) \star z$ **nitpick** [*expect=genuine*] **oops**
lemma *while-denest-2*: $w \ ; ((x \star (y \ ; w)) \star z) = w \ ; (((x \star y) \ ; w) \star z)$ **nitpick** [*expect=genuine*] **oops**
lemma *while-denest-4*: $(x \star w) \star (x \star (y \ ; z)) = (x \star w) \star ((x \star y) \ ; z)$ **nitpick** [*expect=genuine*] **oops**
lemma *while-denest-6*: $(w \ ; (x \star y)) \star z = z + w \ ; ((x + y \ ; w) \star (y \ ; z))$ **nitpick** [*expect=genuine*] **oops**
lemma *while-decompose-6*: $x \star ((y \ ; (x \star 1)) \star z) = y \star ((x \ ; (y \star 1)) \star z)$ **oops**
lemma *while-finite-associative*: $x \star 0 = 0 \longrightarrow (x \star y) \ ; z = x \star (y \ ; z)$ **oops**
lemma *while-sumstar-2*: $(x + y) \star z = x \star ((y \ ; (x \star 1)) \star z)$ **oops**
lemma *while-sumstar-3*: $(x + y) \star z = ((x \star 1) \ ; y) \star (x \star z)$ **oops**
lemma *while-sumstar-1*: $(x + y) \star z = (x \star y) \star ((x \star 1) \ ; z)$ **nitpick** [*expect=genuine*] **oops**
lemma *while-mult-zero-zero*: $(x \ ; (y \star 0)) \star 0 = x \ ; (y \star 0)$ **nitpick** [*expect=genuine*] **oops**

lemma *while-denest-3*: $(x \star w) \star (x \star 0) = (x \star w) \star 0$ **nitpick** [*expect=genuine*] **oops**
lemma *while-denest-5*: $w \ ; ((x \star (y \ ; w)) \star (x \star (y \ ; z))) = w \ ; (((x \star y) \ ; w) \star ((x \star y) \ ; z))$ **nitpick** [*expect=genuine*] **oops**
lemma *while-separate-unfold*: $(y \ ; (x \star 1)) \star z = (y \star z) + (y \star (y \ ; x \ ; (x \star ((y \ ; (x \star 1)) \star z))))$ **nitpick** [*expect=genuine*] **oops**
lemma *while-02*: $x \star ((x \star w) \star ((x \star y) \ ; z)) = (x \star w) \star ((x \star y) \ ; z)$ **nitpick** [*expect=genuine*] **oops**

lemma *while-denest-0*: $w \ ; (x \star (y \ ; z)) \leq (w \ ; (x \star y)) \star (w \ ; (x \star y) \ ; z)$ **nitpick** [*expect=genuine*] **oops**
lemma *while-mult-sub-while-while*: $x \star (y \ ; z) \leq (x \star y) \star z$ **nitpick** [*expect=genuine*] **oops**
lemma *while-zero-zero*: $(x \star 0) \star 0 = x \star 0$ **nitpick** [*expect=genuine*] **oops**
lemma *while-sumstar-3-below*: $(x \star y) \star (x \star z) \leq (x \star y) \star ((x \star 1) \ ; z)$ **nitpick** [*expect=genuine*] **oops**

end

class *extended-binary-itering* = *binary-itering* +
assumes *while-denest-0*: $w \ ; (x \star (y \ ; z)) \leq (w \ ; (x \star y)) \star (w \ ; (x \star y) \ ; z)$

begin

— Theorem 10.2

lemma *while-denest-1*: $w \ ; (x \star (y \ ; z)) \leq (w \ ; (x \star y)) \star z$
by (*metis order-trans while-denest-0 while-right-plus-below*)

lemma *while-mult-sub-while-while*: $x \star (y \ ; z) \leq (x \star y) \star z$
by (*metis mult-left-one while-denest-1*)

lemma *while-zero-zero*: $(x \star 0) \star 0 = x \star 0$
by (*smt less-eq-def mult-left-zero while-left-dist-add while-mult-star-exchange while-mult-sub-while-while while-mult-zero-add-2 while-plus-one while-sumstar*)

— Theorem 10.11

lemma *while-mult-zero-zero*: $(x \ ; (y \star 0)) \star 0 = x \ ; (y \star 0)$
apply (*rule antisym*)
apply (*metis add-least-upper-bound add-right-zero mult-left-zero mult-right-isotone while-left-dist-add while-slide while-sub-associative*)
apply (*metis mult-left-zero while-denest-1*)
done

— Theorem 10.3

lemma *while-denest-2*: $w \ ; ((x \star (y \ ; w)) \star z) = w \ ; (((x \star y) \ ; w) \star z)$
apply (*rule antisym*)
apply (*metis mult-associative while-denest-0 while-simulate-right-plus-1 while-slide*)
apply (*metis mult-right-isotone while-left-isotone while-sub-associative*)
done

— Theorem 10.12

lemma *while-denest-3*: $(x \star w) \star (x \star 0) = (x \star w) \star 0$
by (*metis while-absorb-2 while-right-isotone while-zero-zero zero-least*)

— Theorem 10.15

lemma *while-02*: $x \star ((x \star w) \star ((x \star y) \ ; z)) = (x \star w) \star ((x \star y) \ ; z)$
proof −
have $x \ ; ((x \star w) \star ((x \star y) \ ; z)) = x \ ; (x \star y) \ ; z + x \ ; (x \star w) \ ; ((x \star w) \star ((x \star y) \ ; z))$
by (*metis mult-associative mult-left-dist-add while-left-unfold*)
also have $\ldots \leq (x \star w) \star ((x \star y) \ ; z)$
by (*metis add-isotone mult-right-sub-dist-add-right while-left-unfold*)
finally have $x \star ((x \star w) \star ((x \star y) \ ; z)) \leq ((x \star w) \star ((x \star y) \ ; z)) + (x \star 0)$
by (*metis while-simulate-absorb*)
also have $\ldots = (x \star w) \star ((x \star y) \ ; z)$
by (*metis add-commutative less-eq-def order-trans while-mult-sub-while-while while-right-isotone zero-least*)
finally show *?thesis*
by (*metis antisym while-increasing*)
qed

lemma *while-sumstar-3-below*: $(x \star y) \star (x \star z) \leq (x \star y) \star ((x \star 1) \ ; z)$

proof −
have $(x \star y) \star (x \star z) = (x \star z) + ((x \star y) \star ((x \star y) \; ; (x \star z)))$
by (*metis while-right-unfold*)
also have ... $\leq (x \star z) + ((x \star y) \star (x \star (y \; ; (x \star z))))$
by (*metis add-right-isotone while-right-isotone while-sub-associative*)
also have ... $\leq (x \star z) + ((x \star y) \star (x \star ((x \star y) \star (x \star z))))$
by (*smt add-right-isotone order-trans while-increasing while-mult-upper-bound while-one-increasing while-right-isotone*)
also have ... $\leq (x \star z) + ((x \star y) \star (x \star ((x \star y) \star ((x \star 1) \; ; z))))$
by (*metis add-right-isotone mult-left-isotone mult-left-one order-trans while-increasing while-right-isotone while-sumstar while-transitive*)
also have ... $= (x \star z) + ((x \star y) \star ((x \star 1) \; ; z))$
by (*metis while-02 while-transitive*)
also have ... $= (x \star y) \star ((x \star 1) \; ; z)$
by (*smt add-associative mult-left-one mult-right-dist-add while-02 while-left-dist-add while-plus-one*)
finally show *?thesis*
.
qed

lemma *while-sumstar-4-below*: $(x \star y) \star ((x \star 1) \; ; z) \leq x \star ((y \; ; (x \star 1)) \star z)$
proof −
have $(x \star y) \star ((x \star 1) \; ; z) = (x \star 1) \; ; z + (x \star y) \; ; ((x \star y) \star ((x \star 1) \; ; z))$
by (*metis while-left-unfold*)
also have ... $\leq (x \star z) + (x \star (y \; ; ((x \star y) \star ((x \star 1) \; ; z))))$
by (*metis add-isotone while-one-mult-below while-sub-associative*)
also have ... $= (x \star z) + (x \star (y \; ; (((x \star 1) \; ; y) \star ((x \star 1) \; ; z))))$
by (*metis mult-left-one while-denest-2*)
also have ... $= x \star ((y \; ; (x \star 1)) \star z)$
by (*metis while-left-dist-add while-productstar*)
finally show *?thesis*
.
qed

— Theorem 10.10

lemma *while-sumstar-1*: $(x + y) \star z = (x \star y) \star ((x \star 1) \; ; z)$
by (*smt eq-iff order-trans while-add-1-below while-sumstar while-sumstar-3-below while-sumstar-4-below*)

— Theorem 10.8

lemma *while-sumstar-2*: $(x + y) \star z = x \star ((y \; ; (x \star 1)) \star z)$
by (*metis eq-iff while-add-1-below while-sumstar-1 while-sumstar-4-below*)

— Theorem 10.9

lemma *while-sumstar-3*: $(x + y) \star z = ((x \star 1) \; ; y) \star (x \star z)$
by (*metis eq-iff while-sumstar while-sumstar-1-below while-sumstar-2 while-sumstar-2-below*)

— Theorem 10.6

lemma *while-decompose-6*: $x \star ((y \; ; (x \star 1)) \star z) = y \star ((x \; ; (y \star 1)) \star z)$
by (*metis add-commutative while-sumstar-2*)

— Theorem 10.4

lemma *while-denest-4*: $(x \star w) \star (x \star (y \; ; z)) = (x \star w) \star ((x \star y) \; ; z)$
proof −
have $(x \star w) \star (x \star (y \; ; z)) = x \star ((w \; ; (x \star 1)) \star (y \; ; z))$
by (*metis while-sumstar while-sumstar-2*)
also have ... $\leq (x \star w) \star ((x \star y) \; ; z)$
by (*smt antisym while-01 while-02 while-increasing while-right-isotone*)
finally show *?thesis*
by (*metis antisym while-right-isotone while-sub-associative*)
qed

— Theorem 10.13

lemma *while-denest-5*: $w \; ; ((x \star (y \; ; w)) \star (x \star (y \; ; z))) = w \; ; (((x \star y) \; ; w) \star ((x \star y) \; ; z))$
by (*metis while-denest-2 while-denest-4*)

— Theorem 10.5

lemma *while-denest-6*: $(w\ ;\ (x \star y)) \star z = z + w\ ;\ ((x + y\ ;\ w) \star (y\ ;\ z))$
by (*metis while-denest-5 while-productstar while-sumstar*)

— Theorem 10.1

lemma *while-sum-below-one*: $y\ ;\ ((x + y) \star z) \leq (y\ ;\ (x \star 1)) \star z$
by (*metis add-right-divisibility mult-left-one while-denest-6*)

— Theorem 10.14

lemma *while-separate-unfold*: $(y\ ;\ (x \star 1)) \star z = (y \star z) + (y \star (y\ ;\ x\ ;\ (x \star ((y\ ;\ (x \star 1)) \star z))))$
proof −
have $y \star (y\ ;\ x\ ;\ (x \star ((y\ ;\ (x \star 1)) \star z))) \leq y \star (y\ ;\ ((x + y) \star z))$
by (*metis mult-associative mult-right-isotone while-sumstar-2 while-left-plus-below while-right-isotone*)
also have ... $\leq (y\ ;\ (x \star 1)) \star z$
by (*metis add-commutative add-left-upper-bound while-absorb-1 while-mult-star-exchange while-sum-below-one*)
finally have $(y \star z) + (y \star (y\ ;\ x\ ;\ (x \star ((y\ ;\ (x \star 1)) \star z)))) \leq (y\ ;\ (x \star 1)) \star z$
by (*metis add-least-upper-bound mult-left-sub-dist-add-left mult-right-one while-left-isotone while-left-unfold*)
thus *?thesis*
by (*metis antisym while-separate-unfold-below*)
qed

— Theorem 10.7

lemma *while-finite-associative*: $x \star 0 = 0 \longrightarrow (x \star y)\ ;\ z = x \star (y\ ;\ z)$
by (*metis while-denest-4 while-zero*)

— Theorem 12

lemma *atomicity-refinement*: $s = s\ ;\ q \wedge x = q\ ;\ x \wedge q\ ;\ b = 0 \wedge r\ ;\ b \leq b\ ;\ r \wedge r\ ;\ l \leq l\ ;\ r \wedge x\ ;\ l \leq l\ ;\ x \wedge b\ ;\ l \leq l\ ;\ b \wedge q\ ;\ l \leq l\ ;\ q \wedge r \star q \leq q\ ;\ (r \star 1) \wedge q \leq 1 \longrightarrow s\ ;\ ((x + b + r + l) \star (q\ ;\ z)) \leq s\ ;\ ((x\ ;\ (b \star q) + r + l) \star z)$
proof
assume *1*: $s = s\ ;\ q \wedge x = q\ ;\ x \wedge q\ ;\ b = 0 \wedge r\ ;\ b \leq b\ ;\ r \wedge r\ ;\ l \leq l\ ;\ r \wedge x\ ;\ l \leq l\ ;\ x \wedge b\ ;\ l \leq l\ ;\ b \wedge q\ ;\ l \leq l\ ;\ q \wedge r \star q \leq q\ ;\ (r \star 1) \wedge q \leq 1$
hence *2*: $(x + b + r)\ ;\ l \leq l\ ;\ (x + b + r)$
by (*smt add-commutative add-least-upper-bound mult-left-sub-dist-add-right mult-right-dist-add order-trans*)
have $q\ ;\ ((x\ ;\ (b \star r \star 1)\ ;\ q) \star z) \leq (x\ ;\ (b \star r \star 1)\ ;\ q) \star z$ **using** *1*
by (*smt eq-refl order-trans while-increasing while-mult-upper-bound*)
also have ... $\leq (x\ ;\ (b \star ((r \star 1)\ ;\ q))) \star z$
by (*metis mult-associative mult-right-isotone while-left-isotone while-sub-associative*)
also have ... $\leq (x\ ;\ (b \star r \star q)) \star z$
by (*metis mult-right-isotone while-left-isotone while-one-mult-below while-right-isotone*)
also have ... $\leq (x\ ;\ (b \star (q\ ;\ (r \star 1)))) \star z$ **using** *1*
by (*metis mult-right-isotone while-left-isotone while-right-isotone*)
finally have *3*: $q\ ;\ ((x\ ;\ (b \star r \star 1)\ ;\ q) \star z) \leq (x\ ;\ (b \star q)\ ;\ (r \star 1)) \star z$
by (*metis mult-associative while-associative-while-1*)
have $s\ ;\ ((x + b + r + l) \star (q\ ;\ z)) = s\ ;\ (l \star (x + b + r) \star (q\ ;\ z))$ **using** *2*
by (*metis add-commutative while-separate-1*)
also have ... $= s\ ;\ q\ ;\ (l \star b \star r \star (q\ ;\ x\ ;\ (b \star r \star 1)) \star (q\ ;\ z))$ **using** *1*
by (*smt add-associative add-commutative while-sumstar-2 while-separate-1*)
also have ... $= s\ ;\ q\ ;\ (l \star b \star r \star (q\ ;\ ((x\ ;\ (b \star r \star 1)\ ;\ q) \star z)))$
by (*smt mult-associative while-slide*)
also have ... $\leq s\ ;\ q\ ;\ (l \star b \star r \star (x\ ;\ (b \star q)\ ;\ (r \star 1)) \star z)$ **using** *3*
by (*metis mult-right-isotone while-right-isotone*)
also have ... $\leq s\ ;\ (l \star q\ ;\ (b \star r \star (x\ ;\ (b \star q)\ ;\ (r \star 1)) \star z))$ **using** *1*
by (*smt mult-associative mult-right-isotone while-simulate*)
also have ... $= s\ ;\ (l \star q\ ;\ (r \star (x\ ;\ (b \star q)\ ;\ (r \star 1)) \star z))$ **using** *1*
by (*metis while-elimination*)
also have ... $\leq s\ ;\ (l \star r \star (x\ ;\ (b \star q)\ ;\ (r \star 1)) \star z)$ **using** *1*
by (*metis add-left-divisibility mult-left-one mult-right-dist-add mult-right-isotone while-right-isotone*)
also have ... $= s\ ;\ (l \star (r + x\ ;\ (b \star q)) \star z)$
by (*metis while-sumstar-2*)
also have ... $\leq s\ ;\ ((x\ ;\ (b \star q) + r + l) \star z)$
by (*metis add-commutative mult-right-isotone while-sub-dist-3*)
finally show $s\ ;\ ((x + b + r + l) \star (q\ ;\ z)) \leq s\ ;\ ((x\ ;\ (b \star q) + r + l) \star z)$
.
qed

end

class *bounded-extended-binary-itering* = *bounded-binary-itering* + *extended-binary-itering*

begin

lemma *while-unfold-below*: $x = z + y \ ; \ x \longrightarrow x \leq y \star z$ **nitpick** [*expect=genuine*] **oops**
lemma *while-unfold-below-1*: $x = y \ ; \ x \longrightarrow x \leq y \star 1$ **nitpick** [*expect=genuine*] **oops**
lemma *while-loop-is-greatest-postfixpoint*: *is-greatest-postfixpoint* $(\lambda x \ . \ y \ ; \ x + z) \ (y \star z)$ **nitpick** [*expect=genuine*] **oops**
lemma *while-loop-is-greatest-fixpoint*: *is-greatest-fixpoint* $(\lambda x \ . \ y \ ; \ x + z) \ (y \star z)$ **nitpick** [*expect=genuine*] **oops**
lemma *while-sub-mult-one*: $x \ ; \ (1 \star y) \leq 1 \star x$ **nitpick** [*expect=genuine*] **oops**
lemma *while-sub-while-zero*: $x \star z \leq (x \star y) \star z$ **nitpick** [*expect=genuine*] **oops**
lemma *while-while-sub-associative*: $x \star (y \star z) \leq (x \star y) \star z$ **nitpick** [*expect=genuine*] **oops**
lemma *while-top*: $T \star x = T$ **nitpick** [*expect=genuine*] **oops**
lemma *while-one-top*: $1 \star x = T$ **nitpick** [*expect=genuine*] **oops**
lemma *while-mult-top*: $(x \ ; \ T) \star z = z + x \ ; \ T$ **nitpick** [*expect=genuine*] **oops**
lemma *tarski*: $x \leq x \ ; \ T \ ; \ x \ ; \ T$ **nitpick** [*expect=genuine*] **oops**
lemma *tarski-mult-top-idempotent*: $x \ ; \ T = x \ ; \ T \ ; \ x \ ; \ T$ **nitpick** [*expect=genuine*] **oops**
lemma *tarski-top-omega-below*: $x \ ; \ T \leq (x \ ; \ T) \star 0$ **nitpick** [*expect=genuine*] **oops**
lemma *tarski-top-omega*: $x \ ; \ T = (x \ ; \ T) \star 0$ **nitpick** [*expect=genuine*] **oops**
lemma *tarski-below-top-omega*: $x \leq (x \ ; \ T) \star 0$ **nitpick** [*expect=genuine*] **oops**
lemma *tarski*: $x = 0 \vee T \ ; \ x \ ; \ T = T$ **nitpick** [*expect=genuine*] **oops**
lemma $(x + y) \star z = ((x \star 1) \ ; \ y) \star ((x \star 1) \ ; \ z)$ **nitpick** [*expect=genuine*] **oops**
lemma $1 = (x \ ; \ 0) \star 1$ **nitpick** [*expect=genuine*] **oops**
lemma *while-top-2*: $T \star z = T \ ; \ z$ **nitpick** [*expect=genuine*] **oops**
lemma *while-mult-top-2*: $(x \ ; \ T) \star z = z + x \ ; \ T \ ; \ z$ **nitpick** [*expect=genuine*] **oops**
lemma *while-one-mult*: $(x \star 1) \ ; \ x = x \star x$ **nitpick** [*expect=genuine*] **oops**
lemma $(x \star 1) \ ; \ y = x \star y$ **nitpick** [*expect=genuine*] **oops**
lemma *while-associative*: $(x \star y) \ ; \ z = x \star (y \ ; \ z)$ **nitpick** [*expect=genuine*] **oops**
lemma *while-back-loop-is-fixpoint*: *is-fixpoint* $(\lambda x \ . \ x \ ; \ y + z) \ (z \ ; \ (y \star 1))$ **nitpick** [*expect=genuine*] **oops**
lemma $1 + x \ ; \ 0 = x \star 1$ **nitpick** [*expect=genuine*] **oops**
lemma $x = x \ ; \ (x \star 1)$ **nitpick** [*expect=genuine*] **oops**
lemma $x \ ; \ (x \star 1) = x \star 1$ **nitpick** [*expect=genuine*] **oops**
lemma $x \star 1 = x \star (1 \star 1)$ **nitpick** [*expect=genuine*] **oops**
lemma $(x + y) \star 1 = (x \star (y \star 1)) \star 1$ **nitpick** [*expect=genuine*] **oops**
lemma $z + y \ ; \ x = x \longrightarrow y \star z \leq x$ **nitpick** [*expect=genuine*] **oops**
lemma $y \ ; \ x = x \longrightarrow y \star x \leq x$ **nitpick** [*expect=genuine*] **oops**
lemma $z + x \ ; \ y = x \longrightarrow z \ ; \ (y \star 1) \leq x$ **nitpick** [*expect=genuine*] **oops**
lemma $x \ ; \ y = x \longrightarrow x \ ; \ (y \star 1) \leq x$ **nitpick** [*expect=genuine*] **oops**
lemma $x \ ; \ z = z \ ; \ y \longrightarrow x \star z \leq z \ ; \ (y \star 1)$ **nitpick** [*expect=genuine*] **oops**
lemma $x \ ; \ ((y \ ; \ x) \star y) \leq x \ ; \ ((y \ ; \ x) \star 1) \ ; \ y$ **nitpick** [*expect=genuine*] **oops**

end

end

9 BinaryIteringStrict

theory *BinaryIteringStrict*

imports *BinaryItering Itering*

begin

class *strict-itering* = *itering* + *while* +
  assumes *while-def*: $x \star y = x^\circ \ ; \ y$

begin

— Theorem 8.1

subclass *extended-binary-itering*
  apply *unfold-locales*
  apply (*metis add-commutative circ-loop-fixpoint circ-slide mult-associative while-def*)
  apply (*metis circ-add mult-associative while-def*)
  apply (*metis mult-left-dist-add while-def*)
  apply (*metis mult-associative order-refl while-def*)
  apply (*metis circ-simulate-left-plus mult-associative mult-left-isotone mult-right-dist-add mult-right-one while-def*)
  apply (*metis circ-simulate-right-plus mult-associative mult-left-isotone mult-right-dist-add while-def*)
  apply (*metis add-right-divisibility circ-loop-fixpoint mult-associative while-def*)
  done

— Theorem 13.1

lemma *while-associative*: $(x \star y) \ ; \ z = x \star (y \ ; \ z)$
  by (*metis mult-associative while-def*)

— Theorem 13.3

lemma *while-one-mult*: $(x \star 1) \ ; \ x = x \star x$
  by (*metis mult-right-one while-def*)

lemma *while-back-loop-is-fixpoint*: *is-fixpoint* $(\lambda x \ . \ x \ ; \ y + z) \ (z \ ; \ (y \star 1))$
  by (*metis circ-back-loop-is-fixpoint mult-right-one while-def*)

— Theorem 13.4

lemma $(x + y) \star z = ((x \star 1) \ ; \ y) \star ((x \star 1) \ ; \ z)$
  by (*metis mult-right-one while-def while-sumstar*)

— Theorem 13.2

lemma $(x \star 1) \ ; \ y = x \star y$
  by (*metis mult-left-one while-associative*)

lemma $y \star (x \star 1) \leq x \star (y \star 1) \longrightarrow (x + y) \star 1 = x \star (y \star 1)$ **oops**
lemma $y \ ; \ x \leq (1 + x) \ ; \ (y \star 1) \longrightarrow (x + y) \star 1 = x \star (y \star 1)$ **oops**
lemma *while-square-1*: $x \star 1 = (x \ ; \ x) \star (x + 1)$ **oops**
lemma *while-absorb-below-one*: $y \ ; \ x \leq x \longrightarrow y \star x \leq 1 \star x$ **oops**

end

class *bounded-strict-itering* = *bounded-itering* + *strict-itering*

begin

subclass *bounded-extended-binary-itering* **..**

— Theorem 13.6

lemma *while-top-2*: $T \star z = T \ ; \ z$
  by (*metis circ-top while-def*)

— Theorem 13.5

lemma *while-mult-top-2*: $(x\ ;\ T) \star z = z + x\ ;\ T\ ;\ z$
by (*metis circ-left-top mult-associative while-def while-left-unfold*)

— Theorem 13 counterexamples

lemma *while-one-top*: $1 \star x = T$ **nitpick** [*expect=genuine*] **oops**
lemma *while-top*: $T \star x = T$ **nitpick** [*expect=genuine*] **oops**
lemma *while-sub-mult-one*: $x\ ;\ (1 \star y) \leq 1 \star x$ **nitpick** [*expect=genuine*] **oops**
lemma *while-unfold-below-1*: $x = y\ ;\ x \longrightarrow x \leq y \star 1$ **nitpick** [*expect=genuine*] **oops**
lemma *while-unfold-below*: $x = z + y\ ;\ x \longrightarrow x \leq y \star z$ **nitpick** [*expect=genuine*] **oops**
lemma *while-unfold-below*: $x \leq z + y\ ;\ x \longrightarrow x \leq y \star z$ **nitpick** [*expect=genuine*] **oops**
lemma *while-mult-top*: $(x\ ;\ T) \star z = z + x\ ;\ T$ **nitpick** [*expect=genuine*] **oops**
lemma *tarski-mult-top-idempotent*: $x\ ;\ T = x\ ;\ T\ ;\ x\ ;\ T$ **nitpick** [*expect=genuine*] **oops**

lemma *while-loop-is-greatest-postfixpoint*: *is-greatest-postfixpoint* $(\lambda x\ .\ y\ ;\ x + z)\ (y \star z)$ **nitpick** [*expect=genuine*] **oops**
lemma *while-loop-is-greatest-fixpoint*: *is-greatest-fixpoint* $(\lambda x\ .\ y\ ;\ x + z)\ (y \star z)$ **nitpick** [*expect=genuine*] **oops**
lemma *while-sub-while-zero*: $x \star z \leq (x \star y) \star z$ **nitpick** [*expect=genuine*] **oops**
lemma *while-while-sub-associative*: $x \star (y \star z) \leq (x \star y) \star z$ **nitpick** [*expect=genuine*] **oops**
lemma *tarski*: $x \leq x\ ;\ T\ ;\ x\ ;\ T$ **nitpick** [*expect=genuine*] **oops**
lemma *tarski-top-omega-below*: $x\ ;\ T \leq (x\ ;\ T) \star 0$ **nitpick** [*expect=genuine*] **oops**
lemma *tarski-top-omega*: $x\ ;\ T = (x\ ;\ T) \star 0$ **nitpick** [*expect=genuine*] **oops**
lemma *tarski-below-top-omega*: $x \leq (x\ ;\ T) \star 0$ **nitpick** [*expect=genuine*] **oops**
lemma *tarski*: $x = 0 \vee T\ ;\ x\ ;\ T = T$ **nitpick** [*expect=genuine*] **oops**
lemma $1 = (x\ ;\ 0) \star 1$ **nitpick** [*expect=genuine*] **oops**
lemma $1 + x\ ;\ 0 = x \star 1$ **nitpick** [*expect=genuine*] **oops**
lemma $x = x\ ;\ (x \star 1)$ **nitpick** [*expect=genuine*] **oops**
lemma $x\ ;\ (x \star 1) = x \star 1$ **nitpick** [*expect=genuine*] **oops**
lemma $x \star 1 = x \star (1 \star 1)$ **nitpick** [*expect=genuine*] **oops**
lemma $(x + y) \star 1 = (x \star (y \star 1)) \star 1$ **nitpick** [*expect=genuine*] **oops**
lemma $z + y\ ;\ x = x \longrightarrow y \star z \leq x$ **nitpick** [*expect=genuine*] **oops**
lemma $y\ ;\ x = x \longrightarrow y \star x \leq x$ **nitpick** [*expect=genuine*] **oops**
lemma $z + x\ ;\ y = x \longrightarrow z\ ;\ (y \star 1) \leq x$ **nitpick** [*expect=genuine*] **oops**
lemma $x\ ;\ y = x \longrightarrow x\ ;\ (y \star 1) \leq x$ **nitpick** [*expect=genuine*] **oops**
lemma $x\ ;\ z = z\ ;\ y \longrightarrow x \star z \leq z\ ;\ (y \star 1)$ **nitpick** [*expect=genuine*] **oops**

end

class *binary-itering-unary* = *extended-binary-itering* + *circ* +
assumes *circ-def*: $x^\circ = x \star 1$

begin

— Theorem 50.7

subclass *left-conway-semiring*
apply *unfold-locales*
apply (*metis circ-def while-left-unfold*)
apply (*metis circ-def mult-right-one while-one-mult-below while-slide*)
apply (*metis circ-def while-one-while while-sumstar-2*)
done

end

class *strict-binary-itering* = *binary-itering* + *circ* +
assumes *while-associative*: $(x \star y)\ ;\ z = x \star (y\ ;\ z)$
assumes *circ-def*: $x^\circ = x \star 1$

begin

— Theorem 2.8

subclass *itering*
apply *unfold-locales*
apply (*metis circ-def mult-left-one-1 while-associative while-sumstar*)
apply (*metis circ-def mult-right-one while-associative while-productstar while-slide*)
apply (*metis circ-def mult-right-one while-associative mult-left-one-1 while-slide while-simulate-right-plus*)
apply (*metis circ-def mult-right-one while-associative mult-left-one-1 while-simulate-left-plus mult-right-dist-add*)
done

— Theorem 8.5

subclass *extended-binary-itering*
apply *unfold-locales*
apply (*metis mult-associative while-associative while-increasing*)
done

end

end

10 Approximation

theory *Approximation*

imports *Semiring*

begin

class *apx* =
  fixes *apx* :: $'a \Rightarrow 'a \Rightarrow bool$ (**infix** $\sqsubseteq$ *50*)

class *apx-order* = *apx* +
  assumes *apx-reflexive*: $x \sqsubseteq x$
  assumes *apx-antisymmetric*: $x \sqsubseteq y \wedge y \sqsubseteq x \longrightarrow x = y$
  assumes *apx-transitive*: $x \sqsubseteq y \wedge y \sqsubseteq z \longrightarrow x \sqsubseteq z$

sublocale *apx-order* < *apx*!: *order* **where** *less-eq* = *apx* **and** *less* = $\lambda x\ y\ .\ x \sqsubseteq y \wedge \neg\ y \sqsubseteq x$
  apply *unfold-locales*
  apply *rule*
  apply (*rule apx-reflexive*)
  apply (*metis apx-transitive*)
  apply (*metis apx-antisymmetric*)
  done

context *apx-order*

begin

abbreviation *the-apx-least-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a$ (κ - [*201*] *200*) **where** $\kappa\ f \equiv apx.the\text{-}least\text{-}fixpoint\ f$
abbreviation *the-apx-least-prefixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a$ ($p\kappa$ - [*201*] *200*) **where** $p\kappa\ f \equiv apx.the\text{-}least\text{-}prefixpoint\ f$

definition *is-apx-meet* :: $'a \Rightarrow 'a \Rightarrow 'a \Rightarrow bool$ **where** $is\text{-}apx\text{-}meet\ x\ y\ z \longleftrightarrow z \sqsubseteq x \wedge z \sqsubseteq y \wedge (\forall w\ .\ w \sqsubseteq x \wedge w \sqsubseteq y \longrightarrow w \sqsubseteq z)$
definition *has-apx-meet* :: $'a \Rightarrow 'a \Rightarrow bool$ **where** $has\text{-}apx\text{-}meet\ x\ y \longleftrightarrow (\exists z\ .\ is\text{-}apx\text{-}meet\ x\ y\ z)$
definition *the-apx-meet* :: $'a \Rightarrow 'a \Rightarrow 'a$ (**infixl** $\triangle$ *66*) **where** $x \triangle y = (THE\ z\ .\ is\text{-}apx\text{-}meet\ x\ y\ z)$

lemma *apx-meet-unique*: $has\text{-}apx\text{-}meet\ x\ y \longrightarrow (\exists!z\ .\ is\text{-}apx\text{-}meet\ x\ y\ z)$
  by (*smt apx-antisymmetric has-apx-meet-def is-apx-meet-def*)

lemma *apx-meet*: $has\text{-}apx\text{-}meet\ x\ y \longrightarrow is\text{-}apx\text{-}meet\ x\ y\ (x \triangle y)$
proof
  assume *has-apx-meet x y*
  hence *is-apx-meet x y* (*THE z . is-apx-meet x y z*)
    by (*smt apx-meet-unique theI'*)
  thus *is-apx-meet x y* ($x \triangle y$)
    by (*simp add*: *is-apx-meet-def the-apx-meet-def*)
qed

lemma *apx-greatest-lower-bound*: $has\text{-}apx\text{-}meet\ x\ y \longrightarrow (w \sqsubseteq x \wedge w \sqsubseteq y \longleftrightarrow w \sqsubseteq x \triangle y)$
  by (*smt apx-meet apx-transitive is-apx-meet-def*)

lemma *apx-meet-same*: $is\text{-}apx\text{-}meet\ x\ y\ z \longrightarrow z = x \triangle y$
  by (*metis apx-meet apx-meet-unique has-apx-meet-def*)

lemma *apx-meet-char*: $is\text{-}apx\text{-}meet\ x\ y\ z \longleftrightarrow has\text{-}apx\text{-}meet\ x\ y \wedge z = x \triangle y$
  by (*metis apx-meet-same has-apx-meet-def*)

end

class *apx-biorder* = *apx-order* + *order*

begin

lemma *mu-below-kappa*: $has\text{-}least\text{-}fixpoint\ f \wedge apx.has\text{-}least\text{-}fixpoint\ f \longrightarrow \mu\ f \leq \kappa\ f$
  by (*metis apx.least-fixpoint apx.is-least-fixpoint-def is-least-fixpoint-def least-fixpoint*)

lemma *kappa-below-nu*: $has\text{-}greatest\text{-}fixpoint\ f \wedge apx.has\text{-}least\text{-}fixpoint\ f \longrightarrow \kappa\ f \leq \nu\ f$
  by (*metis apx.least-fixpoint greatest-fixpoint apx.is-least-fixpoint-def is-greatest-fixpoint-def*)

lemma *kappa-apx-below-mu*: *has-least-fixpoint f* ∧ *apx.has-least-fixpoint f* ⟶ κ *f* ⊑ μ *f*
by (*metis apx.least-fixpoint apx.is-least-fixpoint-def is-least-fixpoint-def least-fixpoint*)

lemma *kappa-apx-below-nu*: *has-greatest-fixpoint f* ∧ *apx.has-least-fixpoint f* ⟶ κ *f* ⊑ ν *f*
by (*metis apx.least-fixpoint greatest-fixpoint apx.is-least-fixpoint-def is-greatest-fixpoint-def*)

end

class *apx-semiring* = *apx-biorder* + *idempotent-left-semiring* + *L* +
assumes *apx-L-least*: *L* ⊑ *x*
assumes *add-apx-left-isotone*: *x* ⊑ *y* ⟶ *x* + *z* ⊑ *y* + *z*
assumes *mult-apx-left-isotone*: *x* ⊑ *y* ⟶ *x* ; *z* ⊑ *y* ; *z*
assumes *mult-apx-right-isotone*: *x* ⊑ *y* ⟶ *z* ; *x* ⊑ *z* ; *y*

begin

lemma *add-apx-right-isotone*: *x* ⊑ *y* ⟶ *z* + *x* ⊑ *z* + *y*
by (*metis add-apx-left-isotone add-commutative*)

lemma *add-apx-isotone*: *w* ⊑ *y* ∧ *x* ⊑ *z* ⟶ *w* + *x* ⊑ *y* + *z*
by (*metis add-apx-left-isotone add-apx-right-isotone apx-transitive*)

lemma *mult-apx-isotone*: *w* ⊑ *y* ∧ *x* ⊑ *z* ⟶ *w* ; *x* ⊑ *y* ; *z*
by (*metis apx-transitive mult-apx-left-isotone mult-apx-right-isotone*)

lemma *affine-apx-isotone*: *apx.isotone* (λ*x* . *y* ; *x* + *z*)
by (*smt add-apx-left-isotone apx.isotone-def mult-apx-right-isotone*)

end

end

11 LatticeOrderedSemiring

theory *LatticeOrderedSemiring*

imports *Semiring*

begin

— Many results in this theory are taken from a joint paper with Rudolf Berghammer.

— M0-algebra

class *lattice-ordered-pre-left-semiring* = *pre-left-semiring* + *bounded-distributive-lattice*

begin

subclass *bounded-pre-left-semiring*
apply *unfold-locales*
apply (*metis add-right-top-1*)
done

lemma *top-mult-right-one*: $x \ ; \ T = x \ ; \ T \ ; \ 1$
by (*metis add-commutative add-left-top less-eq-def mult-semi-associative mult-sub-right-one*)

lemma *mult-left-sub-dist-meet-left*: $x \ ; \ (y \frown z) \leq x \ ; \ y$
by (*metis meet.add-left-upper-bound mult-right-isotone*)

lemma *mult-left-sub-dist-meet-right*: $x \ ; \ (y \frown z) \leq x \ ; \ z$
by (*metis meet-commutative mult-left-sub-dist-meet-left*)

lemma *mult-right-sub-dist-meet-left*: $(x \frown y) \ ; \ z \leq x \ ; \ z$
by (*metis meet.add-left-upper-bound mult-left-isotone*)

lemma *mult-right-sub-dist-meet-right*: $(x \frown y) \ ; \ z \leq y \ ; \ z$
by (*metis meet.add-right-upper-bound mult-left-isotone*)

lemma *mult-right-sub-dist-meet*: $(x \frown y) \ ; \ z \leq x \ ; \ z \frown y \ ; \ z$
by (*metis meet.add-least-upper-bound mult-right-sub-dist-meet-left mult-right-sub-dist-meet-right*)

— Figure 1: fundamental properties

definition *total* :: $'a \Rightarrow$ *bool* **where** *total* $x \longleftrightarrow x \ ; \ T = T$
definition *co-total* :: $'a \Rightarrow$ *bool* **where** *co-total* $x \longleftrightarrow x \ ; \ 0 = 0$
definition *transitive* :: $'a \Rightarrow$ *bool* **where** *transitive* $x \longleftrightarrow x \ ; \ x \leq x$
definition *dense* :: $'a \Rightarrow$ *bool* **where** *dense* $x \longleftrightarrow x \leq x \ ; \ x$
definition *reflexive* :: $'a \Rightarrow$ *bool* **where** *reflexive* $x \longleftrightarrow 1 \leq x$
definition *co-reflexive* :: $'a \Rightarrow$ *bool* **where** *co-reflexive* $x \longleftrightarrow x \leq 1$
definition *idempotent* :: $'a \Rightarrow$ *bool* **where** *idempotent* $x \longleftrightarrow x \ ; \ x = x$
definition *up-closed* :: $'a \Rightarrow$ *bool* **where** *up-closed* $x \longleftrightarrow x \ ; \ 1 = x$
definition *add-distributive* :: $'a \Rightarrow$ *bool* **where** *add-distributive* $x \longleftrightarrow (\forall y \ z \ . \ x \ ; \ (y + z) = x \ ; \ y + x \ ; \ z)$
definition *meet-distributive* :: $'a \Rightarrow$ *bool* **where** *meet-distributive* $x \longleftrightarrow (\forall y \ z \ . \ x \ ; \ (y \frown z) = x \ ; \ y \frown x \ ; \ z)$
definition *contact* :: $'a \Rightarrow$ *bool* **where** *contact* $x \longleftrightarrow x \ ; \ x + 1 = x$
definition *kernel* :: $'a \Rightarrow$ *bool* **where** *kernel* $x \longleftrightarrow x \ ; \ x \frown 1 = x \ ; \ 1$
definition *add-dist-contact* :: $'a \Rightarrow$ *bool* **where** *add-dist-contact* $x \longleftrightarrow$ *add-distributive* $x \wedge$ *contact* x
definition *meet-dist-kernel* :: $'a \Rightarrow$ *bool* **where** *meet-dist-kernel* $x \longleftrightarrow$ *meet-distributive* $x \wedge$ *kernel* x
definition *test* :: $'a \Rightarrow$ *bool* **where** *test* $x \longleftrightarrow x \ ; \ T \frown 1 = x$
definition *co-test* :: $'a \Rightarrow$ *bool* **where** *co-test* $x \longleftrightarrow x \ ; \ 0 + 1 = x$
definition *co-vector* :: $'a \Rightarrow$ *bool* **where** *co-vector* $x \longleftrightarrow x \ ; \ 0 = x$

— CPCP Theorem 5 / Figure 2: relations between properties

lemma *reflexive-total*: *reflexive* $x \longrightarrow$ *total* x
by (*metis eq-iff mult-isotone mult-left-one meet.zero-least reflexive-def total-def*)

lemma *reflexive-dense*: *reflexive* $x \longrightarrow$ *dense* x
by (*metis mult-left-isotone mult-left-one reflexive-def dense-def*)

lemma *reflexive-transitive-up-closed*: *reflexive* $x \wedge$ *transitive* $x \longrightarrow$ *up-closed* x
by (*metis antisym-conv mult-isotone mult-sub-right-one reflexive-def reflexive-dense transitive-def dense-def up-closed-def*)

lemma *co-reflexive-co-total*: *co-reflexive x* ⟶ *co-total x*
by (*metis co-reflexive-def co-total-def eq-iff mult-left-isotone mult-left-one zero-least*)

lemma *co-reflexive-transitive*: *co-reflexive x* ⟶ *transitive x*
by (*metis co-reflexive-def mult-left-isotone mult-left-one transitive-def*)

lemma *idempotent-transitive-dense*: *idempotent x* ⟷ *transitive x* ∧ *dense x*
by (*metis eq-iff transitive-def dense-def idempotent-def*)

lemma *contact-reflexive*: *contact x* ⟶ *reflexive x*
by (*metis contact-def add-right-upper-bound reflexive-def*)

lemma *contact-transitive*: *contact x* ⟶ *transitive x*
by (*metis contact-def add-left-upper-bound transitive-def*)

lemma *contact-dense*: *contact x* ⟶ *dense x*
by (*metis contact-reflexive reflexive-dense*)

lemma *contact-idempotent*: *contact x* ⟶ *idempotent x*
by (*metis contact-transitive contact-dense idempotent-transitive-dense*)

lemma *contact-up-closed*: *contact x* ⟶ *up-closed x*
by (*metis contact-def contact-idempotent dual-order.antisym mult-left-sub-dist-add-right mult-sub-right-one idempotent-def up-closed-def*)

lemma *contact-reflexive-idempotent-up-closed*: *contact x* ⟷ *reflexive x* ∧ *idempotent x* ∧ *up-closed x*
by (*metis contact-def contact-idempotent contact-up-closed add-commutative less-eq-def reflexive-def idempotent-def*)

lemma *kernel-co-reflexive*: *kernel x* ⟶ *co-reflexive x*
by (*metis co-reflexive-def kernel-def meet.add-least-upper-bound mult-sub-right-one*)

lemma *kernel-transitive*: *kernel x* ⟶ *transitive x*
by (*metis co-reflexive-transitive kernel-co-reflexive*)

lemma *kernel-dense*: *kernel x* ⟶ *dense x*
by (*metis kernel-def meet.add-least-upper-bound mult-sub-right-one dense-def*)

lemma *kernel-idempotent*: *kernel x* ⟶ *idempotent x*
by (*metis kernel-transitive kernel-dense idempotent-transitive-dense*)

lemma *kernel-up-closed*: *kernel x* ⟶ *up-closed x*
by (*metis co-reflexive-def kernel-co-reflexive kernel-def kernel-idempotent meet-less-eq-def idempotent-def up-closed-def*)

lemma *kernel-co-reflexive-idempotent-up-closed*: *kernel x* ⟷ *co-reflexive x* ∧ *idempotent x* ∧ *up-closed x*
by (*metis co-reflexive-def kernel-def kernel-idempotent kernel-up-closed meet.less-eq-def meet-commutative idempotent-def up-closed-def*)

lemma *test-co-reflexive*: *test x* ⟶ *co-reflexive x*
by (*metis co-reflexive-def meet.add-right-upper-bound test-def*)

lemma *test-up-closed*: *test x* ⟶ *up-closed x*
by (*metis eq-iff mult-left-one mult-sub-right-one mult-right-sub-dist-meet test-def top-mult-right-one up-closed-def*)

lemma *co-test-reflexive*: *co-test x* ⟶ *reflexive x*
by (*metis co-test-def add-right-upper-bound reflexive-def*)

lemma *co-test-transitive*: *co-test x* ⟶ *transitive x*
by (*smt2 co-test-def add-associative less-eq-def mult-left-one mult-left-zero mult-right-dist-add mult-semi-associative transitive-def*)

lemma *co-test-idempotent*: *co-test x* ⟶ *idempotent x*
by (*metis co-test-reflexive co-test-transitive reflexive-dense idempotent-transitive-dense*)

lemma *co-test-up-closed*: *co-test x* ⟶ *up-closed x*
by (*metis co-test-reflexive co-test-idempotent contact-def contact-up-closed add-commutative less-eq-def reflexive-def idempotent-def*)

lemma *co-test-contact*: *co-test x* ⟶ *contact x*
by (*metis co-test-reflexive co-test-idempotent co-test-up-closed contact-reflexive-idempotent-up-closed*)

lemma *vector-transitive*: *vector x* ⟶ *transitive x*
by (*metis mult-right-isotone meet.zero-least vector-def transitive-def*)

lemma *vector-up-closed*: *vector x* ⟶ *up-closed x*
by (*metis vector-def top-mult-right-one up-closed-def*)

— CPCP Theorem 8 / Figure 3: closure properties

— total

lemma *one-total*: *total 1*
by (*metis mult-left-one total-def*)

lemma *top-total*: *total T*
by (*metis top-mult-top total-def*)

lemma *add-total*: *total x* ∧ *total y* ⟶ *total* (*x* + *y*)
by (*metis add-left-top mult-right-dist-add total-def*)

— co-total

lemma *zero-co-total*: *co-total 0*
by (*metis co-total-def mult-left-zero*)

lemma *one-co-total*: *co-total 1*
by (*metis co-total-def mult-left-one*)

lemma *add-co-total*: *co-total x* ∧ *co-total y* ⟶ *co-total* (*x* + *y*)
by (*metis co-total-def add-right-zero mult-right-dist-add*)

lemma *meet-co-total*: *co-total x* ∧ *co-total y* ⟶ *co-total* (*x* ⌢ *y*)
by (*metis co-total-def add-left-zero antisym-conv less-eq-def mult-right-sub-dist-meet-left*)

lemma *comp-co-total*: *co-total x* ∧ *co-total y* ⟶ *co-total* (*x* ; *y*)
by (*metis co-total-def eq-iff mult-semi-associative zero-least*)

— sub-transitive

lemma *zero-transitive*: *transitive 0*
by (*metis mult-left-zero zero-least transitive-def*)

lemma *one-transitive*: *transitive 1*
by (*metis mult-left-one order-refl transitive-def*)

lemma *top-transitive*: *transitive T*
by (*metis meet.zero-least transitive-def*)

lemma *meet-transitive*: *transitive x* ∧ *transitive y* ⟶ *transitive* (*x* ⌢ *y*)
by (*smt2 meet.less-eq-def meet-associative meet-commutative mult-left-sub-dist-meet-left mult-right-sub-dist-meet-left transitive-def*)

— dense

lemma *zero-dense*: *dense 0*
by (*metis zero-least dense-def*)

lemma *one-dense*: *dense 1*
by (*metis mult-sub-right-one dense-def*)

lemma *top-dense*: *dense T*
by (*metis top-left-mult-increasing dense-def*)

lemma *add-dense*: *dense x* ∧ *dense y* ⟶ *dense* (*x* + *y*)
proof
assume *dense x* ∧ *dense y*
hence *x* ≤ *x* ; *x* ∧ *y* ≤ *y* ; *y*
by (*metis dense-def*)
hence *x* ≤ (*x* + *y*) ; (*x* + *y*) ∧ *y* ≤ (*x* + *y*) ; (*x* + *y*)
by (*metis add-left-upper-bound dual-order.trans mult-isotone add-right-upper-bound*)

hence $x + y \leq (x + y) \; ; \; (x + y)$
by (*metis add-least-upper-bound*)
thus *dense* $(x + y)$
by (*metis dense-def*)
qed

— reflexive

lemma *one-reflexive*: *reflexive 1*
by (*metis order-refl reflexive-def*)

lemma *top-reflexive*: *reflexive T*
by (*metis meet.zero-least reflexive-def*)

lemma *add-reflexive*: *reflexive* $x \wedge$ *reflexive* $y \longrightarrow$ *reflexive* $(x + y)$
by (*metis add-associative less-eq-def reflexive-def*)

lemma *meet-reflexive*: *reflexive* $x \wedge$ *reflexive* $y \longrightarrow$ *reflexive* $(x \frown y)$
by (*metis meet.add-least-upper-bound reflexive-def*)

lemma *comp-reflexive*: *reflexive* $x \wedge$ *reflexive* $y \longrightarrow$ *reflexive* $(x \; ; \; y)$
by (*metis mult-left-isotone mult-left-one order-trans reflexive-def*)

— co-reflexive

lemma *zero-co-reflexive*: *co-reflexive 0*
by (*metis co-reflexive-def zero-least*)

lemma *one-co-reflexive*: *co-reflexive 1*
by (*metis co-reflexive-def order-refl*)

lemma *add-co-reflexive*: *co-reflexive* $x \wedge$ *co-reflexive* $y \longrightarrow$ *co-reflexive* $(x + y)$
by (*metis co-reflexive-def add-least-upper-bound*)

lemma *meet-co-reflexive*: *co-reflexive* $x \wedge$ *co-reflexive* $y \longrightarrow$ *co-reflexive* $(x \frown y)$
by (*metis co-reflexive-def meet.less-eq-def meet-associative*)

lemma *comp-co-reflexive*: *co-reflexive* $x \wedge$ *co-reflexive* $y \longrightarrow$ *co-reflexive* $(x \; ; \; y)$
by (*metis co-reflexive-def mult-isotone mult-left-one*)

— idempotent

lemma *zero-idempotent*: *idempotent 0*
by (*metis mult-left-zero idempotent-def*)

lemma *one-idempotent*: *idempotent 1*
by (*metis mult-left-one idempotent-def*)

lemma *top-idempotent*: *idempotent T*
by (*metis top-mult-top idempotent-def*)

— up-closed

lemma *zero-up-closed*: *up-closed 0*
by (*metis mult-left-zero up-closed-def*)

lemma *one-up-closed*: *up-closed 1*
by (*metis mult-left-one up-closed-def*)

lemma *top-up-closed*: *up-closed T*
by (*metis top-mult-top vector-def vector-up-closed*)

lemma *add-up-closed*: *up-closed* $x \wedge$ *up-closed* $y \longrightarrow$ *up-closed* $(x + y)$
by (*metis mult-right-dist-add up-closed-def*)

lemma *meet-up-closed*: *up-closed* $x \wedge$ *up-closed* $y \longrightarrow$ *up-closed* $(x \frown y)$
by (*metis dual-order.antisym mult-sub-right-one mult-right-sub-dist-meet up-closed-def*)

lemma *comp-up-closed*: *up-closed* $x \wedge$ *up-closed* $y \longrightarrow$ *up-closed* $(x \; ; \; y)$

by (*metis dual-order.antisym mult-semi-associative mult-sub-right-one up-closed-def*)

— add-distributive

lemma *zero-add-distributive*: *add-distributive 0*
by (*metis add-distributive-def add-idempotent mult-left-zero*)

lemma *one-add-distributive*: *add-distributive 1*
by (*metis add-distributive-def mult-left-one*)

lemma *add-add-distributive*: *add-distributive x* $\wedge$ *add-distributive y* $\longrightarrow$ *add-distributive* $(x + y)$
by (*smt2 add-distributive-def add-associative add-commutative mult-right-dist-add*)

— meet-distributive

lemma *zero-meet-distributive*: *meet-distributive 0*
by (*metis meet-left-zero mult-left-zero meet-distributive-def*)

lemma *one-meet-distributive*: *meet-distributive 1*
by (*metis mult-left-one meet-distributive-def*)

— contact

lemma *one-contact*: *contact 1*
by (*metis contact-def add-idempotent mult-left-one*)

lemma *top-contact*: *contact T*
by (*metis contact-def add-left-top top-mult-top*)

lemma *meet-contact*: *contact x* $\wedge$ *contact y* $\longrightarrow$ *contact* $(x \frown y)$
by (*smt2 contact-def contact-reflexive contact-transitive contact-up-closed meet.less-eq-def meet-commutative meet-left-dist-add mult-left-sub-dist-add-right meet-transitive meet-up-closed reflexive-def transitive-def up-closed-def*)

— kernel

lemma *zero-kernel*: *kernel 0*
by (*metis kernel-co-reflexive-idempotent-up-closed zero-co-reflexive zero-idempotent zero-up-closed*)

lemma *one-kernel*: *kernel 1*
by (*metis kernel-def meet-idempotent mult-left-one*)

lemma *add-kernel*: *kernel x* $\wedge$ *kernel y* $\longrightarrow$ *kernel* $(x + y)$
by (*metis add-co-reflexive add-dense add-up-closed co-reflexive-transitive kernel-co-reflexive-idempotent-up-closed idempotent-transitive-dense*)

— add-distributive contact

lemma *one-add-dist-contact*: *add-dist-contact 1*
by (*metis add-dist-contact-def one-add-distributive one-contact*)

— meet-distributive kernel

lemma *zero-meet-dist-kernel*: *meet-dist-kernel 0*
by (*metis meet-dist-kernel-def zero-kernel zero-meet-distributive*)

lemma *one-meet-dist-kernel*: *meet-dist-kernel 1*
by (*metis meet-dist-kernel-def one-kernel one-meet-distributive*)

— test

lemma *zero-test*: *test 0*
by (*metis meet-commutative meet-right-zero mult-left-zero test-def*)

lemma *one-test*: *test 1*
by (*metis meet-left-top mult-left-one test-def*)

lemma *add-test*: *test x* $\wedge$ *test y* $\longrightarrow$ *test* $(x + y)$
by (*metis* (*no-types*, *lifting*) *meet-commutative meet-left-dist-add mult-right-dist-add test-def*)

lemma *meet-test*: *test x* ∧ *test y* ⟶ *test* (*x* ⌢ *y*)
by (*smt2 test-def meet-commutative meet.add-least-upper-bound meet.add-right-isotone mult-right-sub-dist-meet-left meet.add-left-upper-bound top-right-mult-increasing antisym*)

— co-test

lemma *one-co-test*: *co-test 1*
by (*metis co-test-def co-total-def add-left-zero one-co-total*)

lemma *add-co-test*: *co-test x* ∧ *co-test y* ⟶ *co-test* (*x* + *y*)
by (*smt2 co-test-contact co-test-def contact-def add-associative add-commutative add-left-zero mult-left-one mult-right-dist-add*)

— vector

lemma *zero-vector*: *vector 0*
by (*metis mult-left-zero vector-def*)

lemma *top-vector*: *vector T*
by (*metis top-mult-top vector-def*)

lemma *add-vector*: *vector x* ∧ *vector y* ⟶ *vector* (*x* + *y*)
by (*metis mult-right-dist-add vector-def*)

lemma *meet-vector*: *vector x* ∧ *vector y* ⟶ *vector* (*x* ⌢ *y*)
by (*metis antisym meet.add-least-upper-bound mult-right-sub-dist-meet-left mult-right-sub-dist-meet-right top-right-mult-increasing vector-def*)

lemma *comp-vector*: *vector y* ⟶ *vector* (*x* ; *y*)
by (*metis antisym-conv mult-semi-associative top-right-mult-increasing vector-def*)

end

class *lattice-ordered-pre-left-semiring-1* = *non-associative-left-semiring* + *bounded-distributive-lattice* +
assumes *mult-associative-one*: *x* ; (*y* ; *z*) = (*x* ; (*y* ; *1*)) ; *z*
assumes *mult-right-dist-meet-one*: (*x* ; *1* ⌢ *y* ; *1*) ; *z* = *x* ; *z* ⌢ *y* ; *z*

begin

subclass *pre-left-semiring*
apply *unfold-locales*
apply (*metis mult-associative-one mult-left-isotone mult-right-isotone mult-sub-right-one*)
done

subclass *lattice-ordered-pre-left-semiring*
..

lemma *mult-zero-associative*: *x* ; *0* ; *y* = *x* ; *0*
by (*smt mult-left-zero mult-associative-one*)

lemma *mult-zero-add-one-dist*: (*x* ; *0* + *1*) ; *z* = *x* ; *0* + *z*
by (*metis mult-left-one mult-right-dist-add mult-zero-associative*)

lemma *mult-zero-add-dist*: (*x* ; *0* + *y*) ; *z* = *x* ; *0* + *y* ; *z*
by (*metis mult-right-dist-add mult-zero-associative*)

lemma *vector-zero-meet-one-comp*: (*x* ; *0* ⌢ *1*) ; *y* = *x* ; *0* ⌢ *y*
by (*metis mult-left-one mult-right-dist-meet-one mult-zero-associative*)

— CPCP Theorem 5 / Figure 2: relations between properties

lemma *co-test-meet-distributive*: *co-test x* ⟶ *meet-distributive x*
by (*metis add-left-dist-meet co-test-def meet-distributive-def mult-zero-add-one-dist*)

lemma *co-test-add-distributive*: *co-test x* ⟶ *add-distributive x*
by (*smt2 add-associative add-commutative add-distributive-def add-left-upper-bound co-test-def less-eq-def mult-zero-add-one-dist*)

lemma *co-test-add-dist-contact*: *co-test x* ⟶ *add-dist-contact x*
by (*metis co-test-add-distributive add-dist-contact-def co-test-contact*)

— CPCP Theorem 8 / Figure 3: closure properties

— co-test

lemma *meet-co-test*: *co-test* $x \wedge$ *co-test* $y \longrightarrow$ *co-test* $(x \frown y)$
by (*smt2 add-commutative add-left-dist-meet co-test-def co-test-up-closed up-closed-def mult-right-dist-meet-one*)

lemma *comp-co-test*: *co-test* $x \wedge$ *co-test* $y \longrightarrow$ *co-test* $(x \mathbin{;} y)$
by (*metis add-associative co-test-def mult-zero-add-dist mult-zero-add-one-dist*)

end

class *lattice-ordered-pre-left-semiring-2* = *lattice-ordered-pre-left-semiring* +
assumes *mult-sub-associative-one*: $x \mathbin{;} (y \mathbin{;} z) \leq (x \mathbin{;} (y \mathbin{;} 1)) \mathbin{;} z$
assumes *mult-right-dist-meet-one-sub*: $x \mathbin{;} z \frown y \mathbin{;} z \leq (x \mathbin{;} 1 \frown y \mathbin{;} 1) \mathbin{;} z$

begin

subclass *lattice-ordered-pre-left-semiring-1*
apply *unfold-locales*
apply (*metis meet.eq-iff mult-sub-associative-one mult-sup-associative-one*)
apply (*metis meet.antisym-conv mult-one-associative mult-right-dist-meet-one-sub mult-right-sub-dist-meet*)
done

end

class *multirelation-algebra-1* = *lattice-ordered-pre-left-semiring* +
assumes *mult-left-top*: $T \mathbin{;} x = T$

begin

— CPCP Theorem 8 / Figure 3: closure properties

lemma *top-add-distributive*: *add-distributive* T
by (*metis add-distributive-def add-left-top mult-left-top*)

lemma *top-meet-distributive*: *meet-distributive* T
by (*metis meet-idempotent meet-distributive-def mult-left-top*)

lemma *top-add-dist-contact*: *add-dist-contact* T
by (*metis add-dist-contact-def top-add-distributive top-contact*)

lemma *top-co-test*: *co-test* T
by (*metis co-test-def add-left-top mult-left-top*)

end

— M1-algebra

class *multirelation-algebra-2* = *multirelation-algebra-1* + *lattice-ordered-pre-left-semiring-2*

begin

lemma *mult-top-associative*: $x \mathbin{;} T \mathbin{;} y = x \mathbin{;} T$
by (*metis mult-left-top mult-associative-one*)

lemma *vector-meet-one-comp*: $(x \mathbin{;} T \frown 1) \mathbin{;} y = x \mathbin{;} T \frown y$
by (*metis mult-left-one mult-left-top mult-associative-one mult-right-dist-meet-one*)

lemma *vector-left-annihilator*: *vector* $x \longrightarrow x \mathbin{;} y = x$
by (*metis mult-left-top vector-def mult-associative-one*)

— properties

lemma *test-comp-meet*: *test* $x \wedge$ *test* $y \longrightarrow x \mathbin{;} y = x \frown y$
by (*smt2 meet-associative meet-commutative meet-idempotent test-def vector-meet-one-comp*)

— CPCP Theorem 5 / Figure 2: relations between properties

lemma *test-add-distributive*: *test x* $\longrightarrow$ *add-distributive x*
by (*metis add-distributive-def meet-left-dist-add test-def vector-meet-one-comp*)

lemma *test-meet-distributive*: *test x* $\longrightarrow$ *meet-distributive x*
by (*smt2 meet.less-eq-def meet-associative meet-commutative meet-distributive-def meet.add-right-upper-bound mult-left-one test-def vector-meet-one-comp*)

lemma *test-meet-dist-kernel*: *test x* $\longrightarrow$ *meet-dist-kernel x*
by (*metis kernel-co-reflexive-idempotent-up-closed meet-associative meet-dist-kernel-def meet-idempotent test-co-reflexive test-def test-up-closed idempotent-def vector-meet-one-comp test-meet-distributive*)

lemma *vector-idempotent*: *vector x* $\longrightarrow$ *idempotent x*
by (*metis idempotent-def vector-left-annihilator*)

lemma *vector-add-distributive*: *vector x* $\longrightarrow$ *add-distributive x*
by (*metis add-distributive-def add-idempotent vector-left-annihilator*)

lemma *vector-meet-distributive*: *vector x* $\longrightarrow$ *meet-distributive x*
by (*metis meet-distributive-def meet-idempotent vector-left-annihilator*)

lemma *vector-co-vector*: *vector x* $\longleftrightarrow$ *co-vector x*
by (*metis co-vector-def vector-def mult-zero-associative vector-left-annihilator*)

— CPCP Theorem 8 / Figure 3: closure properties

— test

lemma *comp-test*: *test x* $\wedge$ *test y* $\longrightarrow$ *test* (*x* ; *y*)
by (*metis meet-associative meet-distributive-def meet.add-right-zero test-def test-up-closed up-closed-def mult-associative-one test-meet-distributive*)

end

class *dual* =
fixes *dual* :: *'a* $\Rightarrow$ *'a* (-^{d} [*100*] *100*)

class *multirelation-algebra-3* = *lattice-ordered-pre-left-semiring* + *dual* +
assumes *dual-involutive*: $x^{dd} = x$
assumes *dual-dist-add*: $(x + y)^d = x^d \frown y^d$
assumes *dual-one*: $1^d = 1$

begin

lemma *dual-dist-meet*: $(x \frown y)^d = x^d + y^d$
by (*metis dual-dist-add dual-involutive*)

lemma *dual-antitone*: $x \leq y \longrightarrow y^d \leq x^d$
by (*metis dual-dist-meet add-left-divisibility meet.add-left-divisibility*)

lemma *dual-zero*: $0^d = T$
by (*metis dual-dist-meet add-right-top dual-involutive meet-left-zero*)

lemma *dual-top*: $T^d = 0$
by (*metis dual-zero dual-involutive*)

— CPCP Theorem 8 / Figure 3: closure properties

lemma *reflexive-co-reflexive-dual*: *reflexive x* $\longleftrightarrow$ *co-reflexive* (x^d)
by (*metis co-reflexive-def dual-antitone dual-involutive dual-one reflexive-def*)

end

class *multirelation-algebra-4* = *multirelation-algebra-3* +
assumes *dual-sub-dist-comp*: $(x \ ; \ y)^d \leq x^d \ ; \ y^d$

begin

subclass *multirelation-algebra-1*
apply *unfold-locales*

apply (*metis dual-zero dual-sub-dist-comp dual-involutive meet.less-eq-def meet-commutative meet-left-top mult-left-zero*)
done

lemma *dual-sub-dist-comp-one*: $(x\ ;\ y)^d \leq (x\ ;\ 1)^d\ ;\ y^d$
by (*metis dual-sub-dist-comp mult-one-associative*)

— CPCP Theorem 8 / Figure 3: closure properties

lemma *co-total-total-dual*: *co-total* $x \longrightarrow$ *total* (x^d)
by (*metis co-total-def dual-sub-dist-comp dual-zero meet.less-eq-def meet-commutative meet-left-top total-def*)

lemma *transitive-dense-dual*: *transitive* $x \longrightarrow$ *dense* (x^d)
by (*metis dual-antitone dual-sub-dist-comp order-trans transitive-def dense-def*)

end

— M2-algebra

class *multirelation-algebra-5* = *multirelation-algebra-3* +
assumes *dual-dist-comp-one*: $(x\ ;\ y)^d = (x\ ;\ 1)^d\ ;\ y^d$

begin

subclass *multirelation-algebra-4*
apply *unfold-locales*
apply (*metis dual-antitone mult-sub-right-one mult-left-isotone dual-dist-comp-one*)
done

lemma *strong-up-closed*: $x\ ;\ 1 \leq x \longrightarrow x^d\ ;\ y^d \leq (x\ ;\ y)^d$
by (*metis dual-dist-comp-one eq-iff mult-sub-right-one*)

lemma *strong-up-closed-2*: *up-closed* $x \longrightarrow (x\ ;\ y)^d = x^d\ ;\ y^d$
by (*metis dual-sub-dist-comp eq-iff strong-up-closed up-closed-def*)

subclass *lattice-ordered-pre-left-semiring-2*
apply *unfold-locales*
apply (*smt2 comp-up-closed dual-antitone dual-dist-comp-one dual-involutive dual-one mult-left-one mult-one-associative mult-semi-associative up-closed-def strong-up-closed-2*)
apply (*smt2 dual-dist-comp-one dual-dist-meet dual-involutive eq-refl mult-one-associative mult-right-dist-add*)
done

— CPCP Theorem 6

subclass *multirelation-algebra-2*
..

— CPCP Theorem 8 / Figure 3: closure properties

— up-closed

lemma *up-closed-dual*: *up-closed* $x \longleftrightarrow$ *up-closed* (x^d)
by (*metis dual-involutive dual-one up-closed-def strong-up-closed-2*)

— contact

lemma *contact-kernel-dual*: *contact* $x \longleftrightarrow$ *kernel* (x^d)
by (*metis contact-def contact-up-closed dual-dist-add dual-involutive dual-one kernel-def kernel-up-closed up-closed-def strong-up-closed-2*)

— add-distributive contact

lemma *add-dist-contact-meet-dist-kernel-dual*: *add-dist-contact* $x \longleftrightarrow$ *meet-dist-kernel* (x^d)
proof
assume *1*: *add-dist-contact x*
hence *2*: *up-closed x*
by (*metis add-dist-contact-def contact-up-closed*)
have *add-distributive x* **using** *1*
by (*metis add-dist-contact-def*)
hence *meet-distributive* (x^d) **using** *2*

by (*smt2 meet-distributive-def add-distributive-def dual-dist-add dual-involutive strong-up-closed-2*)
thus *meet-dist-kernel* (x^d) **using** *1*
by (*metis contact-kernel-dual add-dist-contact-def meet-dist-kernel-def*)
next
assume *3*: *meet-dist-kernel* (x^d)
hence *2*: *up-closed* (x^d)
by (*metis kernel-up-closed meet-dist-kernel-def*)
have *meet-distributive* (x^d) **using** *3*
by (*metis meet-dist-kernel-def*)
hence *add-distributive* (x^{dd}) **using** *2*
by (*smt2 meet-distributive-def add-distributive-def dual-dist-add dual-involutive strong-up-closed-2*)
thus *add-dist-contact x* **using** *3*
by (*metis contact-kernel-dual add-dist-contact-def meet-dist-kernel-def dual-involutive*)
qed

— test

lemma *test-co-test-dual*: *test x* $\longleftrightarrow$ *co-test* (x^d)
by (*smt2 co-test-def co-test-up-closed dual-dist-meet dual-involutive dual-one dual-top test-def test-up-closed strong-up-closed-2*)

— vector

lemma *vector-dual*: *vector x* $\longleftrightarrow$ *vector* (x^d)
by (*metis dual-dist-comp-one comp-vector dual-involutive dual-top vector-def zero-vector*)

end

class *multirelation-algebra-6* = *multirelation-algebra-4* +
assumes *dual-sub-dist-comp-one*: $(x\ ;\ 1)^d\ ;\ y^d \leq (x\ ;\ y)^d$

begin

subclass *multirelation-algebra-5*
apply *unfold-locales*
apply (*metis dual-sub-dist-comp dual-sub-dist-comp-one meet.eq-iff mult-one-associative*)
done

lemma *dense x* $\wedge$ *co-reflexive x* $\longrightarrow$ *up-closed x* **nitpick** [*expect=genuine*] **oops**
lemma $x\ ;\ T \frown y\ ;\ z \leq (x\ ;\ T \frown y)\ ;\ z$ **nitpick** [*expect=genuine,card=8*] **oops**

end

— M3-algebra

class *up-closed-multirelation-algebra* = *multirelation-algebra-3* +
assumes *dual-dist-comp*: $(x\ ;\ y)^d = x^d\ ;\ y^d$

begin

lemma *mult-right-dist-meet*: $(x \frown y)\ ;\ z = x\ ;\ z \frown y\ ;\ z$
by (*metis dual-dist-add dual-dist-comp dual-involutive mult-right-dist-add*)

— CPCP Theorem 7

subclass *idempotent-left-semiring*
apply *unfold-locales*
apply (*metis antisym dual-antitone dual-dist-comp dual-involutive mult-semi-associative*)
apply (*metis mult-left-one*)
apply (*metis dual-dist-add dual-dist-comp dual-involutive dual-one less-eq-def meet-absorb mult-sub-right-one*)
done

subclass *multirelation-algebra-6*
apply *unfold-locales*
apply (*metis dual-dist-comp eq-iff*)
apply (*metis dual-dist-comp eq-iff mult-right-one*)
done

lemma *vector-meet-comp*: $(x\ ;\ T \frown y)\ ;\ z = x\ ;\ T \frown y\ ;\ z$
by (*metis mult-associative mult-left-top mult-right-dist-meet*)

lemma *vector-zero-meet-comp*: $(x\ ;\ 0 \frown y)\ ;\ z = x\ ;\ 0 \frown y\ ;\ z$
by (*metis mult-associative mult-left-zero mult-right-dist-meet*)

— CPCP Theorem 8 / Figure 3: closure properties

— total

lemma *meet-total*: $total\ x \wedge total\ y \longrightarrow total\ (x \frown y)$
by (*metis meet-left-top total-def mult-right-dist-meet*)

lemma *comp-total*: $total\ x \wedge total\ y \longrightarrow total\ (x\ ;\ y)$
by (*metis mult-associative total-def*)

lemma *total-co-total-dual*: $total\ x \longleftrightarrow co\text{-}total\ (x^d)$
by (*metis co-total-def dual-dist-comp dual-involutive dual-top total-def*)

— dense

lemma *transitive-iff-dense-dual*: $transitive\ x \longleftrightarrow dense\ (x^d)$
by (*metis dense-def dual-antitone dual-dist-comp dual-involutive transitive-def*)

— idempotent

lemma *idempotent-dual*: $idempotent\ x \longleftrightarrow idempotent\ (x^d)$
by (*metis dual-involutive idempotent-transitive-dense transitive-iff-dense-dual*)

— add-distributive

lemma *comp-add-distributive*: $add\text{-}distributive\ x \wedge add\text{-}distributive\ y \longrightarrow add\text{-}distributive\ (x\ ;\ y)$
by (*metis add-distributive-def mult-associative*)

lemma *add-meet-distributive-dual*: $add\text{-}distributive\ x \longleftrightarrow meet\text{-}distributive\ (x^d)$
by (*metis* (*no-types*, *hide-lams*) *add-distributive-def dual-dist-add dual-dist-comp dual-involutive meet-distributive-def*)

— meet-distributive

lemma *meet-meet-distributive*: $meet\text{-}distributive\ x \wedge meet\text{-}distributive\ y \longrightarrow meet\text{-}distributive\ (x \frown y)$
by (*smt2 meet-distributive-def meet-associative meet-commutative mult-right-dist-meet*)

lemma *comp-meet-distributive*: $meet\text{-}distributive\ x \wedge meet\text{-}distributive\ y \longrightarrow meet\text{-}distributive\ (x\ ;\ y)$
by (*metis meet-distributive-def mult-associative*)

lemma $co\text{-}total\ x \wedge transitive\ x \wedge up\text{-}closed\ x \longrightarrow co\text{-}reflexive\ x$ **nitpick** [*expect*=*genuine*] **oops**
lemma $total\ x \wedge dense\ x \wedge up\text{-}closed\ x \longrightarrow reflexive\ x$ **nitpick** [*expect*=*genuine*] **oops**
lemma $x\ ;\ T \frown x^d\ ;\ 0 = 0$ **nitpick** [*expect*=*genuine*] **oops**

end

class *multirelation-algebra-7* = *multirelation-algebra-4* +
assumes *vector-meet-comp*: $(x\ ;\ T \frown y)\ ;\ z = x\ ;\ T \frown y\ ;\ z$

begin

lemma *vector-zero-meet-comp*: $(x\ ;\ 0 \frown y)\ ;\ z = x\ ;\ 0 \frown y\ ;\ z$
by (*metis vector-def comp-vector vector-meet-comp zero-vector*)

lemma *test-add-distributive*: $test\ x \longrightarrow add\text{-}distributive\ x$
by (*metis add-distributive-def meet-left-dist-add mult-left-one test-def vector-meet-comp*)

lemma *test-meet-distributive*: $test\ x \longrightarrow meet\text{-}distributive\ x$
by (*smt2 meet.less-eq-def meet-associative meet-commutative meet-distributive-def meet.add-right-upper-bound mult-left-one test-def vector-meet-comp*)

lemma *test-meet-dist-kernel*: $test\ x \longrightarrow meet\text{-}dist\text{-}kernel\ x$
by (*metis kernel-co-reflexive-idempotent-up-closed meet-associative meet-dist-kernel-def meet-idempotent mult-left-one test-co-reflexive test-def test-up-closed idempotent-def vector-meet-comp test-meet-distributive*)

lemma *co-test-meet-distributive*: $co\text{-}test\ x \longrightarrow meet\text{-}distributive\ x$
proof

assume *co-test x*
hence $x = x \ ; \ 0 + 1$
by (*metis co-test-def*)
hence $\forall\, y\ z\ .\ x \ ; \ y \frown x \ ; \ z = x \ ; \ (y \frown z)$
by (*metis mult-left-one mult-left-top mult-right-dist-add meet.add-right-zero vector-zero-meet-comp add-left-dist-meet*)
thus *meet-distributive x*
by (*metis meet-distributive-def*)
qed

lemma *co-test-add-distributive*: *co-test x* $\longrightarrow$ *add-distributive x*
proof
assume *co-test x*
hence *1*: $x = x \ ; \ 0 + 1$
by (*metis co-test-def*)
hence $\forall\, y\ z\ .\ x \ ; \ (y + z) = x \ ; \ y + x \ ; \ z$
by (*metis add-associative add-commutative add-idempotent mult-left-one mult-left-top mult-right-dist-add meet.add-right-zero vector-zero-meet-comp*)
thus *add-distributive x*
by (*metis add-distributive-def*)
qed

lemma *co-test-add-dist-contact*: *co-test x* $\longrightarrow$ *add-dist-contact x*
by (*metis co-test-add-distributive add-dist-contact-def co-test-contact*)

end

end

12 NAlgebra

theory *NAlgebra*

imports *LatticeOrderedSemiring*

begin

class *C-left-n-algebra* = *bounded-idempotent-left-semiring* + *bounded-distributive-lattice* + *n* + *L*

begin

abbreviation $C :: 'a \Rightarrow 'a$ **where** $C\ x \equiv n(L)\ ;\ T \frown x$

— AACP Theorem 3.38

lemma *C-isotone*: $x \leq y \longrightarrow C\ x \leq C\ y$
by (*metis meet.add-right-isotone*)

— AACP Theorem 3.40

lemma *C-decreasing*: $C\ x \leq x$
by (*metis meet.add-right-upper-bound*)

end

class *left-n-algebra* = *C-left-n-algebra* +
assumes *n-dist-n-add* : $n(x) + n(y) = n(n(x)\ ;\ T + y)$
assumes *n-export* : $n(x)\ ;\ n(y) = n(n(x)\ ;\ y)$
assumes *n-left-upper-bound* : $n(x) \leq n(x + y)$
assumes *n-nL-meet-L-nL0* : $n(L)\ ;\ x = (x \frown L) + n(L\ ;\ 0)\ ;\ x$
assumes *n-n-L-split-n-n-L-L* : $x\ ;\ n(y)\ ;\ L = x\ ;\ 0 + n(x\ ;\ n(y)\ ;\ L)\ ;\ L$
assumes *n-sub-nL* : $n(x) \leq n(L)$
assumes *n-L-decreasing* : $n(x)\ ;\ L \leq x$
assumes *n-L-T-meet-mult-combined*: $C\ (x\ ;\ y)\ ;\ z \leq C\ x\ ;\ y\ ;\ C\ z$
assumes *n-n-top-split-n-top* : $x\ ;\ n(y)\ ;\ T \leq x\ ;\ 0 + n(x\ ;\ y)\ ;\ T$
assumes *n-top-meet-L-below-L* : $x\ ;\ T\ ;\ y \frown L \leq x\ ;\ L\ ;\ y$

begin

subclass *lattice-ordered-pre-left-semiring* **..**

lemma *n-L-T-meet-mult-below*: $C\ (x\ ;\ y) \leq C\ x\ ;\ y$
proof −
have $C\ (x\ ;\ y) \leq C\ x\ ;\ y\ ;\ C\ 1$
by (*metis mult-right-one mult-associative n-L-T-meet-mult-combined*)
also have ... $\leq C\ x\ ;\ y$
by (*metis mult-right-one C-decreasing mult-right-isotone mult-associative*)
finally show *?thesis*
.
qed

— AACP Theorem 3.41

lemma *n-L-T-meet-mult-propagate*: $C\ x\ ;\ y \leq x\ ;\ C\ y$
proof −
have $C\ x\ ;\ y \leq C\ x\ ;\ 1\ ;\ C\ y$
by (*metis mult-right-one mult-associative n-L-T-meet-mult-combined mult-right-one*)
also have ... $\leq x\ ;\ C\ y$
by (*metis mult-right-one mult-associative mult-right-one C-decreasing mult-left-isotone*)
finally show *?thesis*
.
qed

— AACP Theorem 3.43

lemma *C-n-mult-closed*: $C\ (n(x)\ ;\ y) = n(x)\ ;\ y$
by (*metis meet.less-eq-def mult-isotone n-sub-nL top-greatest*)

— AACP Theorem 3.40

lemma *meet-L-below-C*: $x \frown L \leq C\ x$
by (*metis add-left-upper-bound meet.add-left-isotone meet-commutative meet-left-top n-nL-meet-L-nL0*)

— AACP Theorem 3.42

lemma *n-L-T-meet-mult*: $C\ (x\ ;\ y) = C\ x\ ;\ y$
apply (*rule antisym*)
apply (*rule n-L-T-meet-mult-below*)
apply (*metis meet.add-least-upper-bound meet.add-left-upper-bound mult-associative mult-isotone mult-right-sub-dist-meet-right top-greatest top-mult-top*)
done

— AACP Theorem 3.42

lemma *C-mult-propagate*: $C\ x\ ;\ y = C\ x\ ;\ C\ y$
by (*smt2 C-n-mult-closed meet.less-eq-def meet-associative meet-less-eq-def mult-left-sub-dist-meet-right n-L-T-meet-mult-propagate*)

— AACP Theorem 3.32

lemma *meet-L-below-n-L*: $x \frown L \leq n(L)\ ;\ x$
by (*metis add-left-divisibility n-nL-meet-L-nL0*)

— AACP Theorem 3.27

lemma *n-vector-meet-L*: $x\ ;\ T \frown L \leq x\ ;\ L$
by (*metis mult-right-one n-top-meet-L-below-L*)

lemma *n-right-upper-bound*: $n(x) \leq n(y + x)$
by (*metis add-commutative n-left-upper-bound*)

— AACP Theorem 3.1

lemma *n-isotone*: $x \leq y \longrightarrow n(x) \leq n(y)$
by (*metis less-eq-def n-left-upper-bound*)

lemma *n-add-left-zero*: $n(0) + n(x) = n(x)$
by (*metis less-eq-def n-isotone zero-least*)

— AACP Theorem 3.13

lemma *n-mult-right-zero-L*: $n(x)\ ;\ 0 \leq L$
by (*metis dual-order.trans mult-isotone n-L-decreasing n-sub-nL zero-least*)

lemma *n-add-left-top*: $n(T) + n(x) = n(T)$
by (*metis add-commutative add-right-top-1 n-dist-n-add*)

— AACP Theorem 3.18

lemma *n-n-L*: $n(n(x)\ ;\ L) = n(x)$
by (*metis add-commutative add-right-zero less-eq-def n-dist-n-add n-export n-sub-nL n-add-left-top n-add-left-zero*)

lemma *n-mult-transitive* : $n(x)\ ;\ n(x) \leq n(x)$
by (*metis mult-right-isotone n-export n-sub-nL n-n-L*)

lemma *n-mult-left-absorb-add-sub*: $n(x)\ ;\ (n(x) + n(y)) \leq n(x)$
by (*metis add-associative less-eq-def mult-left-sub-dist-add-left n-export n-sub-nL n-n-L*)

— AACP Theorem 3.21

lemma *n-mult-left-lower-bound*: $n(x)\ ;\ n(y) \leq n(x)$
by (*metis mult-right-isotone n-export n-sub-nL n-n-L*)

— AACP Theorem 3.20

lemma *n-mult-left-zero*: $n(0)\ ;\ n(x) = n(0)$
by (*metis add-commutative less-eq-def n-export n-add-left-zero n-mult-left-lower-bound*)

lemma *n-mult-right-one*: $n(x) \ ; \ n(T) = n(x)$
by (*metis add-left-upper-bound add-right-zero meet.eq-iff n-dist-n-add n-export n-mult-left-lower-bound*)

lemma *n-L-increasing*: $n(x) \leq n(n(x) \ ; \ L)$
by (*metis order-refl n-n-L*)

— AACP Theorem 3.2

lemma *n-galois*: $n(x) \leq n(y) \longleftrightarrow n(x) \ ; \ L \leq y$
by (*metis mult-left-isotone n-L-decreasing n-L-increasing n-isotone order-trans*)

lemma *n-add-n-top*: $n(x + n(x) \ ; \ T) = n(x)$
by (*metis add-commutative add-idempotent n-dist-n-add*)

— AACP Theorem 3.6

lemma *n-L-below-nL-top*: $L \leq n(L) \ ; \ T$
by (*metis meet.add-right-zero meet-L-below-C meet-left-top*)

— AACP Theorem 3.4

lemma *n-less-eq-char-n*: $x \leq y \longleftrightarrow x \leq y + L \wedge C \, x \leq y + n(y) \ ; \ T$
proof
assume $x \leq y$
thus $x \leq y + L \wedge C \, x \leq y + n(y) \ ; \ T$
by (*metis add-left-upper-bound dual-order.trans meet.add-right-upper-bound*)
next
assume *1*: $x \leq y + L \wedge C \, x \leq y + n(y) \ ; \ T$
hence $x \leq y + (x \frown L)$
by (*metis meet-less-eq-def add-left-dist-meet add-right-upper-bound meet.add-left-isotone*)
also have ... $\leq y + C \, x$
by (*metis add-right-isotone less-eq-def meet-left-dist-add n-L-below-nL-top meet-commutative*)
also have ... $\leq y + n(y) \ ; \ T$ **using** *1*
by (*metis add-least-upper-bound add-left-upper-bound*)
finally have $x \leq y + (L \frown n(y) \ ; \ T)$ **using** *1*
by (*metis meet.add-least-upper-bound add-left-dist-meet*)
thus $x \leq y$
by (*metis add-idempotent add-least-upper-bound n-vector-meet-L less-eq-def meet-commutative n-L-decreasing*)
qed

— AACP Theorem 3.31

lemma *n-L-decreasing-meet-L*: $n(x) \ ; \ L \leq x \frown L$
by (*metis meet.add-least-upper-bound n-L-decreasing n-sub-nL n-galois*)

— AACP Theorem 3.5

lemma *n-zero-L-zero*: $n(0) \ ; \ L = 0$
by (*metis antisym n-L-decreasing zero-least*)

lemma *n-L-top-below-L*: $L \ ; \ T \leq L$
proof −
have $n(L \ ; \ 0) \ ; \ L \ ; \ T \leq L \ ; \ 0$
by (*metis mult-associative mult-left-isotone n-L-decreasing vector-def zero-vector*)
hence $n(L \ ; \ 0) \ ; \ L \ ; \ T \leq L$
by (*metis order-trans zero-right-mult-decreasing*)
hence $n(L) \ ; \ L \ ; \ T \leq L$
by (*metis add-least-upper-bound meet.add-right-upper-bound mult-associative n-nL-meet-L-nL0*)
thus $L \ ; \ T \leq L$
by (*metis add-least-upper-bound eq-iff meet-idempotent n-nL-meet-L-nL0 top-right-mult-increasing*)
qed

— AACP Theorem 3.9

lemma *n-L-top-L*: $L \ ; \ T = L$
by (*metis antisym n-L-top-below-L top-right-mult-increasing*)

— AACP Theorem 3.10

lemma *n-L-below-L*: $L\ ;\ x \leq L$
by (*metis add-right-top n-L-top-below-L mult-left-sub-dist-add-left order-trans*)

— AACP Theorem 3.7

lemma *n-nL-nT*: $n(L) = n(T)$
by (*metis antisym n-isotone n-sub-nL top-greatest*)

— AACP Theorem 3.8

lemma *n-L-L*: $n(L)\ ;\ L = L$
by (*metis meet.less-eq-def meet-absorb meet-idempotent n-L-decreasing n-nL-meet-L-nL0*)

lemma *n-top-L*: $n(T)\ ;\ L = L$
by (*metis n-L-L n-nL-nT*)

— AACP Theorem 3.23

lemma *n-n-L-split-n-L*: $x\ ;\ n(y)\ ;\ L \leq x\ ;\ 0 + n(x\ ;\ y)\ ;\ L$
by (*metis n-n-L-split-n-n-L-L n-L-decreasing mult-associative mult-left-isotone mult-right-isotone n-isotone add-right-isotone*)

— AACP Theorem 3.12

lemma *n-L-split-n-L-L*: $x\ ;\ L = x\ ;\ 0 + n(x\ ;\ L)\ ;\ L$
apply (*rule antisym*)
apply (*metis mult-associative n-n-L-split-n-L n-L-L*)
apply (*metis add-least-upper-bound mult-right-isotone n-L-decreasing zero-least*)
done

— AACP Theorem 3.11

lemma *n-L-split-L*: $x\ ;\ L \leq x\ ;\ 0 + L$
by (*metis add-commutative add-left-isotone meet.add-least-upper-bound n-L-decreasing-meet-L n-L-split-n-L-L*)

— AACP Theorem 3.24

lemma *n-split-top*: $x\ ;\ n(y)\ ;\ T \leq x\ ;\ y + n(x\ ;\ y)\ ;\ T$
proof −
have $x\ ;\ 0 + n(x\ ;\ y)\ ;\ T \leq x\ ;\ y + n(x\ ;\ y)\ ;\ T$
by (*metis add-left-isotone mult-right-isotone zero-least*)
thus *?thesis*
by (*smt n-n-top-split-n-top order-trans*)
qed

— AACP Theorem 3.9

lemma *n-L-L-L*: $L\ ;\ L = L$
by (*metis antisym n-vector-meet-L n-L-below-L meet-idempotent n-L-top-L*)

— AACP Theorem 3.9

lemma *n-L-top-L-L*: $L\ ;\ T\ ;\ L = L$
by (*metis n-L-L-L n-L-top-L*)

— AACP Theorem 3.19

lemma *n-n-nL*: $n(x) = n(x)\ ;\ n(L)$
by (*metis n-export n-n-L*)

lemma *n-L-mult-idempotent*: $n(L)\ ;\ n(L) = n(L)$
by (*metis n-n-nL*)

— AACP Theorem 3.22

lemma *n-n-L-n*: $n(x\ ;\ n(y)\ ;\ L) \leq n(x\ ;\ y)$
by (*metis mult-associative mult-right-isotone n-L-decreasing n-isotone*)

— AACP Theorem 3.3

lemma *n-less-eq-char*: $x \leq y \longleftrightarrow x \leq y + L \wedge x \leq y + n(y)\ ;\ T$
by (*smt add-absorb add-associative add-idempotent n-less-eq-char-n meet-less-eq-def meet-left-dist-add n-add-n-top*)

— AACP Theorem 3.28

lemma *n-top-meet-L-split-L*: $x\ ;\ T\ ;\ y \frown L \leq x\ ;\ 0 + L\ ;\ y$
proof −
have $x\ ;\ T\ ;\ y \frown L \leq x\ ;\ 0 + n(x\ ;\ L)\ ;\ L\ ;\ y$
by (*smt n-top-meet-L-below-L mult-associative n-L-L-L n-L-split-n-L-L mult-right-dist-add mult-left-zero*)
also have $\ldots \leq x\ ;\ 0 + x\ ;\ L\ ;\ y$
by (*metis add-right-isotone mult-right-sub-dist-add-right n-L-split-n-L-L*)
also have $\ldots \leq x\ ;\ 0 + (x\ ;\ 0 + L)\ ;\ y$
by (*metis add-right-isotone mult-left-isotone n-L-split-L*)
also have $\ldots = x\ ;\ 0 + x\ ;\ 0\ ;\ y + L\ ;\ y$
by (*metis add-associative mult-right-dist-add*)
also have $\ldots = x\ ;\ 0 + L\ ;\ y$
by (*metis add-idempotent mult-associative mult-left-zero*)
finally show *?thesis*
.
qed

— AACP Theorem 3.29

lemma *n-top-meet-L-L-meet-L*: $x\ ;\ T\ ;\ y \frown L = x\ ;\ L\ ;\ y \frown L$
apply (*rule antisym*)
apply (*metis meet.add-least-upper-bound meet.add-right-upper-bound n-top-meet-L-below-L*)
apply (*metis meet.add-left-isotone mult-left-isotone mult-right-isotone top-greatest*)
done

lemma *n-n-top-below-n-L*: $n(x\ ;\ T) \leq n(x\ ;\ L)$
by (*metis n-vector-meet-L n-L-decreasing-meet-L n-galois order-trans*)

— AACP Theorem 3.14

lemma *n-n-top-n-L*: $n(x\ ;\ T) = n(x\ ;\ L)$
by (*metis antisym mult-right-isotone n-isotone n-n-top-below-n-L top-greatest*)

— AACP Theorem 3.30

lemma *n-meet-L-0-below-0-meet-L*: $(x \frown L)\ ;\ 0 \leq x\ ;\ 0 \frown L$
by (*metis meet.add-least-upper-bound mult-isotone order.refl zero-right-mult-decreasing*)

— AACP Theorem 3.15

lemma *n-n-L-below-L*: $n(x)\ ;\ L \leq x\ ;\ L$
by (*metis mult-associative mult-left-isotone n-L-L-L n-L-decreasing*)

lemma *n-n-L-below-n-L-L*: $n(x)\ ;\ L \leq n(x\ ;\ L)\ ;\ L$
by (*metis n-n-L-below-L mult-left-isotone n-galois*)

— AACP Theorem 3.16

lemma *n-below-n-L*: $n(x) \leq n(x\ ;\ L)$
by (*metis n-n-L-below-L n-galois*)

— AACP Theorem 3.17

lemma *n-below-n-L-mult*: $n(x) \leq n(L)\ ;\ n(x)$
by (*metis n-export order-trans meet-L-below-n-L n-L-decreasing-meet-L n-isotone n-n-L*)

— AACP Theorem 3.33

lemma *n-meet-L-below*: $n(x) \frown L \leq x$
by (*metis meet.add-left-isotone n-L-decreasing n-vector-meet-L order-trans top-right-mult-increasing*)

— AACP Theorem 3.35

lemma *n-meet-L-top-below-n-L*: $(n(x) \frown L)\ ;\ T \leq n(x)\ ;\ L$
proof −

have $(n(x) \frown L)\ ;\ T \leq n(x)\ ;\ T \frown L\ ;\ T$
by (*metis meet.add-least-upper-bound meet.add-left-upper-bound meet-commutative mult-left-isotone*)
thus *?thesis*
by (*metis n-L-top-L n-vector-meet-L order-trans*)
qed

— AACP Theorem 3.34

lemma *n-meet-L-top-below*: $(n(x) \frown L)\ ;\ T \leq x$
by (*metis n-L-decreasing n-meet-L-top-below-n-L order-trans*)

— AACP Theorem 3.36

lemma *n-n-meet-L*: $n(x) = n(x \frown L)$
by (*metis add-absorb add-commutative less-eq-def n-L-decreasing-meet-L n-n-L n-right-upper-bound*)

lemma *n-T-below-n-meet*: $n(x)\ ;\ T = n(C\ x)\ ;\ T$
by (*metis C-n-mult-closed meet-associative meet-commutative n-L-L n-n-meet-L*)

— AACP Theorem 3.44

lemma *n-C*: $n(C\ x) = n(x)$
by (*metis n-T-below-n-meet n-export n-mult-right-one*)

— AACP Theorem 3.37

lemma *n-T-meet-L*: $n(x)\ ;\ T \frown L = n(x)\ ;\ L$
by (*metis antisym-conv n-L-decreasing-meet-L n-n-L n-n-top-n-L n-vector-meet-L*)

— AACP Theorem 3.39

lemma *n-L-top-meet-L*: $C\ L = L$
by (*metis meet-less-eq-def n-L-below-nL-top meet-commutative*)

end

class *n-algebra* = *left-n-algebra* + *idempotent-left-zero-semiring*

begin

— AACP Theorem 3.25

lemma *n-top-split-0*: $n(x)\ ;\ T\ ;\ y \leq x\ ;\ y + n(x\ ;\ 0)\ ;\ T$
proof −
have *1*: $n(x)\ ;\ T\ ;\ y \frown L \leq x\ ;\ y$
by (*metis mult-left-isotone n-L-decreasing n-top-meet-L-below-L order-trans*)
have $n(x)\ ;\ T\ ;\ y = n(x)\ ;\ n(L)\ ;\ T\ ;\ y$
by (*metis n-export n-n-L*)
also have ... $= n(x)\ ;\ ((T\ ;\ y \frown L) + n(L\ ;\ 0)\ ;\ T\ ;\ y)$
by (*metis mult-associative n-nL-meet-L-nL0*)
also have ... $\leq n(x)\ ;\ (T\ ;\ y \frown L) + n(x)\ ;\ n(L\ ;\ 0)\ ;\ T$
by (*metis add-right-isotone mult-associative mult-left-dist-add mult-right-isotone top-greatest*)
also have ... $\leq (n(x)\ ;\ T\ ;\ y \frown L) + n(n(x)\ ;\ L\ ;\ 0)\ ;\ T$
by (*metis add-left-isotone dual-order.trans meet.add-least-upper-bound mult-associative mult-left-sub-dist-meet-left mult-left-sub-dist-meet-right n-export n-galois n-sub-nL*)
also have ... $\leq x\ ;\ y + n(n(x)\ ;\ L\ ;\ 0)\ ;\ T$ **using** *1*
by (*metis add-left-isotone*)
also have ... $\leq x\ ;\ y + n(x\ ;\ 0)\ ;\ T$
by (*metis add-right-isotone mult-left-isotone n-L-decreasing n-isotone*)
finally show *?thesis*
.
qed

— AACP Theorem 3.26

lemma *n-top-split*: $n(x)\ ;\ T\ ;\ y \leq x\ ;\ y + n(x\ ;\ y)\ ;\ T$
by (*metis add-right-isotone add-right-zero meet.order-trans mult-left-isotone mult-left-sub-dist-add-right n-isotone*

n-top-split-0)

lemma *n-n-meet-L-n-zero*: $n(x) = (n(x) \frown L) + n(x\ ;\ 0)$ **oops**

lemma *n-below-n-zero*: $n(x) \leq x + n(x\ ;\ 0)$ **oops**

lemma *n-n-top-split-n-L-n-zero-top*: $n(x)\ ;\ T = n(x)\ ;\ L + n(x\ ;\ 0)\ ;\ T$ **oops**

lemma *n-zero*: $n(0) = 0$ **nitpick** [*expect=genuine*] **oops**
lemma *n-one*: $n(1) = 0$ **nitpick** [*expect=genuine*] **oops**
lemma *n-nL-one*: $n(L) = 1$ **nitpick** [*expect=genuine*] **oops**
lemma *n-nT-one*: $n(T) = 1$ **nitpick** [*expect=genuine*] **oops**
lemma *n-n-zero*: $n(x) = n(x\ ;\ 0)$ **nitpick** [*expect=genuine*] **oops**
lemma *n-dist-add*: $n(x) + n(y) = n(x + y)$ **nitpick** [*expect=genuine*] **oops**
lemma *n-L-split*: $x\ ;\ n(y)\ ;\ L = x\ ;\ 0 + n(x\ ;\ y)\ ;\ L$ **nitpick** [*expect=genuine*] **oops**
lemma *n-split*: $x \leq x\ ;\ 0 + n(x\ ;\ L)\ ;\ T$ **nitpick** [*expect=genuine*] **oops**
lemma *n-mult-top-1*: $n(x\ ;\ y) \leq n(x\ ;\ n(y)\ ;\ T)$ **nitpick** [*expect=genuine*] **oops**
lemma *l91-1*: $n(L)\ ;\ x \leq n(x\ ;\ T)\ ;\ T$ **nitpick** [*expect=genuine*] **oops**
lemma *meet-domain-top*: $x \frown n(y)\ ;\ T = n(y)\ ;\ x$ **nitpick** [*expect=genuine*] **oops**
lemma *meet-domain-2*: $x \frown n(y)\ ;\ T \leq n(L)\ ;\ x$ **nitpick** [*expect=genuine*] **oops**
lemma *n-nL-top-n-top-meet-L-top-2*: $n(L)\ ;\ x\ ;\ T \leq n(x\ ;\ T \frown L)\ ;\ T$ **nitpick** [*expect=genuine*] **oops**
lemma *n-nL-top-n-top-meet-L-top-1*: $n(x\ ;\ T \frown L)\ ;\ T \leq n(L)\ ;\ x\ ;\ T$ **nitpick** [*expect=genuine*] **oops**
lemma *l9*: $x\ ;\ 0 \frown L \leq n(x\ ;\ L)\ ;\ L$ **nitpick** [*expect=genuine*] **oops**
lemma *l18-2*: $n(x\ ;\ L)\ ;\ L \leq n(x)\ ;\ L$ **nitpick** [*expect=genuine*] **oops**
lemma *l51-1*: $n(x)\ ;\ L \leq (x \frown L)\ ;\ 0$ **nitpick** [*expect=genuine*] **oops**
lemma *l51-2*: $(x \frown L)\ ;\ 0 \leq n(x)\ ;\ L$ **nitpick** [*expect=genuine*] **oops**

lemma *n-split-equal*: $x + n(x\ ;\ L)\ ;\ T = x\ ;\ 0 + n(x\ ;\ L)\ ;\ T$ **nitpick** [*expect=genuine*] **oops**
lemma *n-split-top*: $x\ ;\ T \leq x\ ;\ 0 + n(x\ ;\ L)\ ;\ T$ **nitpick** [*expect=genuine*] **oops**
lemma *n-mult*: $n(x\ ;\ n(y)\ ;\ L) = n(x\ ;\ y)$ **nitpick** [*expect=genuine*] **oops**
lemma *n-mult-1*: $n(x\ ;\ y) \leq n(x\ ;\ n(y)\ ;\ L)$ **nitpick** [*expect=genuine*] **oops**
lemma *n-mult-top*: $n(x\ ;\ n(y)\ ;\ T) = n(x\ ;\ y)$ **nitpick** [*expect=genuine*] **oops**
lemma *n-mult-right-upper-bound*: $n(x\ ;\ y) \leq n(z) \longleftrightarrow n(x) \leq n(z) \wedge x\ ;\ n(y)\ ;\ L \leq x\ ;\ 0 + n(z)\ ;\ L$ **nitpick** [*expect=genuine*] **oops**
lemma *meet-domain*: $x \frown n(y)\ ;\ z = n(y)\ ;\ (x \frown z)$ **nitpick** [*expect=genuine*] **oops**
lemma *meet-domain-1*: $x \frown n(y)\ ;\ z \leq n(y)\ ;\ x$ **nitpick** [*expect=genuine*] **oops**
lemma *meet-domain-top-3*: $x \frown n(y)\ ;\ T \leq n(y)\ ;\ x$ **nitpick** [*expect=genuine*] **oops**
lemma *n-n-top-n-top-split-n-n-top-top*: $n(x)\ ;\ T + x\ ;\ n(y)\ ;\ T = x\ ;\ 0 + n(x\ ;\ n(y)\ ;\ T)\ ;\ T$ **nitpick** [*expect=genuine*] **oops**
lemma *n-n-top-n-top-split-n-n-top-top-1*: $x\ ;\ 0 + n(x\ ;\ n(y)\ ;\ T)\ ;\ T \leq n(x)\ ;\ T + x\ ;\ n(y)\ ;\ T$ **nitpick** [*expect=genuine*] **oops**
lemma *n-n-top-n-top-split-n-n-top-top-2*: $n(x)\ ;\ T + x\ ;\ n(y)\ ;\ T \leq x\ ;\ 0 + n(x\ ;\ n(y)\ ;\ T)\ ;\ T$ **nitpick** [*expect=genuine*] **oops**
lemma *n-nL-top-n-top-meet-L-top*: $n(L)\ ;\ x\ ;\ T = n(x\ ;\ T \frown L)\ ;\ T$ **nitpick** [*expect=genuine*] **oops**
lemma *l18*: $n(x)\ ;\ L = n(x\ ;\ L)\ ;\ L$ **nitpick** [*expect=genuine*] **oops**
lemma *l22*: $x\ ;\ 0 \frown L = n(x)\ ;\ L$ **nitpick** [*expect=genuine*] **oops**
lemma *l22-1*: $x\ ;\ 0 \frown L = n(x\ ;\ L)\ ;\ L$ **nitpick** [*expect=genuine*] **oops**
lemma *l22-2*: $x \frown L = n(x)\ ;\ L$ **nitpick** [*expect=genuine*] **oops**
lemma *l22-3*: $x \frown L = n(x\ ;\ L)\ ;\ L$ **nitpick** [*expect=genuine*] **oops**
lemma *l22-4*: $x \frown L \leq n(x)\ ;\ L$ **nitpick** [*expect=genuine*] **oops**
lemma *l22-5*: $x\ ;\ 0 \frown L \leq n(x)\ ;\ L$ **nitpick** [*expect=genuine*] **oops**
lemma *l23*: $x\ ;\ T \frown L = n(x)\ ;\ L$ **nitpick** [*expect=genuine*] **oops**
lemma *l51*: $n(x)\ ;\ L = (x \frown L)\ ;\ 0$ **nitpick** [*expect=genuine*] **oops**
lemma *l91*: $x = x\ ;\ T \longrightarrow n(L)\ ;\ x \leq n(x)\ ;\ T$ **nitpick** [*expect=genuine*] **oops**
lemma *l92*: $x = x\ ;\ T \longrightarrow n(L)\ ;\ x \leq n(x \frown L)\ ;\ T$ **nitpick** [*expect=genuine*] **oops**
lemma $x \frown L \leq n(x)\ ;\ T$ **nitpick** [*expect=genuine*] **oops**
lemma *n-meet-comp*: $n(x) \frown n(y) \leq n(x)\ ;\ n(y)$ **nitpick** [*expect=genuine*] **oops**

lemma *n-meet-L-0-0-meet-L*: $(x \frown L)\ ;\ 0 = x\ ;\ 0 \frown L$ **oops**

end

end

13 Recursion

theory *Recursion*

imports *Approximation NAlgebra*

begin

class *n-algebra-apx* = *n-algebra* + *apx* +
  assumes *apx-def*: $x \sqsubseteq y \longleftrightarrow x \leq y + L \wedge C\ y \leq x + n(x)\ ;\ T$

begin

lemma *apx-transitive-2*: $x \sqsubseteq y \wedge y \sqsubseteq z \longrightarrow x \sqsubseteq z$
proof
  assume *1*: $x \sqsubseteq y \wedge y \sqsubseteq z$
  hence $C\ z \leq C\ (y + n(y)\ ;\ T)$
    by (*metis apx-def meet.add-least-upper-bound meet.add-left-upper-bound*)
  also have ... $= C\ y + n(y)\ ;\ T$
    by (*metis add-commutative add-left-dist-meet mult-right-dist-add n-add-left-top n-nL-nT*)
  also have ... $\leq x + n(x)\ ;\ T + n(y)\ ;\ T$ **using** *1*
    by (*metis add-left-isotone apx-def*)
  also have ... $= x + n(x)\ ;\ T + n(C\ y)\ ;\ T$
    by (*metis n-T-below-n-meet*)
  also have ... $\leq x + n(x)\ ;\ T$ **using** *1*
    by (*metis add-associative add-idempotent add-right-isotone apx-def mult-left-isotone n-add-n-top n-isotone*)
  finally show $x \sqsubseteq z$ **using** *1*
    by (*smt add-associative add-commutative apx-def less-eq-def*)
qed

lemma *apx-meet-L*: $y \sqsubseteq x \longrightarrow x \frown L \leq y \frown L$
proof
  assume *1*: $y \sqsubseteq x$
  have $x \frown L = C\ x \frown L$
    by (*metis meet-associative meet-commutative n-L-top-meet-L*)
  also have ... $\leq (y + n(y)\ ;\ T) \frown L$ **using** *1*
    by (*metis apx-def meet.add-left-isotone*)
  also have ... $= (y \frown L) + (n(y)\ ;\ T \frown L)$ **using** *1*
    by (*metis meet-commutative meet-left-dist-add*)
  also have ... $\leq (y \frown L) + n(y \frown L)\ ;\ T$
    by (*metis add-least-upper-bound n-vector-meet-L meet.add-least-upper-bound n-L-decreasing order-refl order-trans*)
  finally show $x \frown L \leq y \frown L$
    by (*metis less-eq-def meet.add-right-upper-bound n-less-eq-char*)
qed

— AACP Theorem 4.1

subclass *apx-biorder*
  apply *unfold-locales*
  apply (*metis add-absorb add-least-upper-bound add-left-upper-bound apx-def meet-commutative*)
  apply (*metis add-same-context antisym apx-def apx-meet-L relative-equality*)
  apply (*metis apx-transitive-2*)
  done

lemma *add-apx-left-isotone-2*: $x \sqsubseteq y \longrightarrow x + z \sqsubseteq y + z$
proof
  assume *1*: $x \sqsubseteq y$
  hence *2*: $x + z \leq y + z + L$
    by (*smt add-associative add-commutative add-left-isotone apx-def*)
  have $C\ (y + z) \leq x + n(x)\ ;\ T + C\ z$ **using** *1*
    by (*metis add-left-isotone apx-def meet-left-dist-add meet-commutative*)
  also have ... $\leq x + z + n(x)\ ;\ T$
    by (*smt2 add-associative add-commutative add-right-isotone meet.add-right-upper-bound*)
  also have ... $\leq x + z + n(x + z)\ ;\ T$
    by (*metis add-commutative add-right-isotone mult-left-isotone n-right-upper-bound*)
  finally show $x + z \sqsubseteq y + z$ **using** *2*
    by (*metis apx-def*)
qed

lemma *mult-apx-left-isotone-2*: $x \sqsubseteq y \longrightarrow x \ ; z \sqsubseteq y \ ; z$
proof
assume *1*: $x \sqsubseteq y$
hence $x \ ; z \leq y \ ; z + L \ ; z$
by (*metis apx-def mult-left-isotone mult-right-dist-add*)
hence *2*: $x \ ; z \leq y \ ; z + L$
by (*metis add-commutative add-left-isotone n-L-below-L order-trans*)
have $C \ (y \ ; z) = C \ y \ ; z$
by (*metis meet-commutative n-L-T-meet-mult*)
also have ... $\leq x \ ; z + n(x) \ ; T \ ; z$ **using** *1*
by (*metis apx-def mult-left-isotone mult-right-dist-add*)
also have ... $\leq x \ ; z + n(x \ ; z) \ ; T$
by (*metis add-least-upper-bound add-left-upper-bound n-top-split*)
finally show $x \ ; z \sqsubseteq y \ ; z$ **using** *2*
by (*metis apx-def*)
qed

lemma *mult-apx-right-isotone-2*: $x \sqsubseteq y \longrightarrow z \ ; x \sqsubseteq z \ ; y$
proof
assume *1*: $x \sqsubseteq y$
hence $z \ ; x \leq z \ ; y + z \ ; L$
by (*metis apx-def mult-left-dist-add mult-right-isotone*)
also have ... $\leq z \ ; y + z \ ; 0 + L$
by (*metis add-associative add-right-isotone n-L-split-L*)
finally have *2*: $z \ ; x \leq z \ ; y + L$
by (*metis add-right-zero mult-left-dist-add*)
have $C \ (z \ ; y) \leq z \ ; C \ y$
by (*metis meet-commutative n-L-T-meet-mult n-L-T-meet-mult-propagate*)
also have ... $\leq z \ ; (x + n(x) \ ; T)$ **using** *1*
by (*metis apx-def mult-right-isotone*)
also have ... $= z \ ; x + z \ ; n(x) \ ; T$
by (*metis mult-associative mult-left-dist-add*)
also have ... $\leq z \ ; x + n(z \ ; x) \ ; T$
by (*metis add-least-upper-bound add-left-upper-bound n-split-top*)
finally show $z \ ; x \sqsubseteq z \ ; y$ **using** *2*
by (*metis apx-def*)
qed

— AACP Theorem 4.1 and Theorem 4.2

subclass *apx-semiring*
apply *unfold-locales*
apply (*metis add-commutative add-left-upper-bound apx-def less-eq-def meet.add-left-upper-bound n-L-below-nL-top*)
apply (*rule add-apx-left-isotone-2*)
apply (*rule mult-apx-left-isotone-2*)
apply (*rule mult-apx-right-isotone-2*)
done

— AACP Theorem 4.2

lemma *meet-L-apx-isotone*: $x \sqsubseteq y \longrightarrow x \frown L \sqsubseteq y \frown L$
by (*smt add-commutative add-idempotent add-left-dist-meet apx-def apx-meet-L n-less-eq-char-n meet-commutative meet.add-right-isotone*)

— AACP Theorem 4.2

lemma *n-L-apx-isotone*: $x \sqsubseteq y \longrightarrow n(x) \ ; L \sqsubseteq n(y) \ ; L$
proof
assume *1*: $x \sqsubseteq y$
have $C \ (n(y) \ ; L) \leq n(C \ y) \ ; L$
by (*metis meet.add-left-upper-bound n-T-below-n-meet n-n-L n-n-top-n-L meet-commutative*)
also have ... $\leq n(x) \ ; L + n(n(x) \ ; L) \ ; T$ **using** *1*
by (*metis add-left-upper-bound apx-def mult-left-isotone n-add-n-top n-export n-isotone order-trans*)
finally show $n(x) \ ; L \sqsubseteq n(y) \ ; L$
by (*metis apx-def less-eq-def meet.add-least-upper-bound n-L-decreasing-meet-L*)
qed

definition *kappa-apx-meet* :: $('a \Rightarrow 'a) \Rightarrow bool$
where *kappa-apx-meet f* $\longleftrightarrow$ *apx.has-least-fixpoint f* $\wedge$ *has-apx-meet* $(\mu \ f) \ (\nu \ f) \wedge \kappa \ f = \mu \ f \ \triangle \ \nu \ f$

definition *kappa-mu-nu* :: $('a \Rightarrow 'a) \Rightarrow bool$
where *kappa-mu-nu f* $\longleftrightarrow$ *apx.has-least-fixpoint f* $\wedge$ $\kappa\ f = \mu\ f + (\nu\ f \frown L)$

definition *nu-below-mu-nu* :: $('a \Rightarrow 'a) \Rightarrow bool$
where *nu-below-mu-nu f* $\longleftrightarrow$ $C\ (\nu\ f) \leq \mu\ f + (\nu\ f \frown L) + n(\nu\ f)\ ;\ T$

definition *nu-below-mu-nu-2* :: $('a \Rightarrow 'a) \Rightarrow bool$
where *nu-below-mu-nu-2 f* $\longleftrightarrow$ $C\ (\nu\ f) \leq \mu\ f + (\nu\ f \frown L) + n(\mu\ f + (\nu\ f \frown L))\ ;\ T$

definition *mu-nu-apx-nu* :: $('a \Rightarrow 'a) \Rightarrow bool$
where *mu-nu-apx-nu f* $\longleftrightarrow$ $\mu\ f + (\nu\ f \frown L) \sqsubseteq \nu\ f$

definition *mu-nu-apx-meet* :: $('a \Rightarrow 'a) \Rightarrow bool$
where *mu-nu-apx-meet f* $\longleftrightarrow$ *has-apx-meet* $(\mu\ f)\ (\nu\ f) \wedge \mu\ f \mathbin{\triangle} \nu\ f = \mu\ f + (\nu\ f \frown L)$

definition *apx-meet-below-nu* :: $('a \Rightarrow 'a) \Rightarrow bool$
where *apx-meet-below-nu f* $\longleftrightarrow$ *has-apx-meet* $(\mu\ f)\ (\nu\ f) \wedge \mu\ f \mathbin{\triangle} \nu\ f \leq \nu\ f$

lemma *mu-below-l*: $\mu\ f \leq \mu\ f + (\nu\ f \frown L)$
by (*metis add-left-upper-bound*)

lemma *l-below-nu*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\longrightarrow$ $\mu\ f + (\nu\ f \frown L) \leq \nu\ f$
by (*metis add-least-upper-bound meet.add-left-upper-bound mu-below-nu*)

lemma *n-l-nu*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\longrightarrow$ $(\mu\ f + (\nu\ f \frown L)) \frown L = \nu\ f \frown L$
by (*smt add-commutative add-left-dist-meet less-eq-def meet-absorb meet-associative meet-commutative mu-below-nu*)

lemma *l-apx-mu*: $\mu\ f + (\nu\ f \frown L) \sqsubseteq \mu\ f$
proof −
have *1*: $\mu\ f + (\nu\ f \frown L) \leq \mu\ f + L$
by (*metis add-right-isotone meet.add-right-upper-bound*)
have $C\ (\mu\ f) \leq \mu\ f + (\nu\ f \frown L) + n(\mu\ f + (\nu\ f \frown L))\ ;\ T$
by (*metis add-left-upper-bound n-less-eq-char-n meet-commutative*)
thus *?thesis* **using** *1*
by (*metis apx-def*)
qed

— AACP Theorem 4.8 implies Theorem 4.9

lemma *nu-below-mu-nu-nu-below-mu-nu-2*: *nu-below-mu-nu f* $\longrightarrow$ *nu-below-mu-nu-2 f*
proof
assume *1*: *nu-below-mu-nu f*
have $C\ (\nu\ f) = C\ (C\ (\nu\ f))$
by (*metis meet-associative meet-idempotent*)
also have ... $\leq C\ (\mu\ f + (\nu\ f \frown L) + n(\nu\ f)\ ;\ T)$ **using** *1*
by (*metis calculation meet.add-least-upper-bound meet.add-left-upper-bound nu-below-mu-nu-def*)
also have ... $= C\ (\mu\ f + (\nu\ f \frown L)) + C\ (n(\nu\ f)\ ;\ T)$
by (*metis meet-left-dist-add*)
also have ... $= C\ (\mu\ f + (\nu\ f \frown L)) + n(\nu\ f)\ ;\ T$
by (*metis C-n-mult-closed*)
also have ... $\leq \mu\ f + (\nu\ f \frown L) + n(\nu\ f)\ ;\ T$
by (*metis add-left-isotone meet.add-right-upper-bound*)
also have ... $= \mu\ f + (\nu\ f \frown L) + n(\nu\ f \frown L)\ ;\ T$
by (*metis n-n-meet-L*)
also have ... $\leq \mu\ f + (\nu\ f \frown L) + n(\mu\ f + (\nu\ f \frown L))\ ;\ T$
by (*metis add-right-isotone mult-left-isotone n-right-upper-bound*)
finally show *nu-below-mu-nu-2 f*
by (*metis nu-below-mu-nu-2-def*)
qed

— AACP Theorem 4.9 implies Theorem 4.8

lemma *nu-below-mu-nu-2-nu-below-mu-nu*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\wedge$ *nu-below-mu-nu-2 f* $\longrightarrow$ *nu-below-mu-nu f*
proof
assume *1*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\wedge$ *nu-below-mu-nu-2 f*
hence $C\ (\nu\ f) \leq \mu\ f + (\nu\ f \frown L) + n(\mu\ f + (\nu\ f \frown L))\ ;\ T$
by (*metis nu-below-mu-nu-2-def*)
also have ... $\leq \mu\ f + (\nu\ f \frown L) + n(\nu\ f)\ ;\ T$ **using** *1*

by (*metis add-right-isotone l-below-nu mult-left-isotone n-isotone*)
finally show *nu-below-mu-nu f*
by (*metis nu-below-mu-nu-def*)
qed

lemma *nu-below-mu-nu-equivalent*: *has-least-fixpoint f* ∧ *has-greatest-fixpoint f* ⟶ (*nu-below-mu-nu f* ⟷ *nu-below-mu-nu-2 f*)
by (*metis nu-below-mu-nu-2-nu-below-mu-nu nu-below-mu-nu-nu-below-mu-nu-2*)

— AACP Theorem 4.9 implies Theorem 4.10

lemma *nu-below-mu-nu-2-mu-nu-apx-nu*: *has-least-fixpoint f* ∧ *has-greatest-fixpoint f* ∧ *nu-below-mu-nu-2 f* ⟶ *mu-nu-apx-nu f*
proof
assume *1*: *has-least-fixpoint f* ∧ *has-greatest-fixpoint f* ∧ *nu-below-mu-nu-2 f*
hence *μ f* + (*ν f* ⌢ *L*) ≤ *ν f* + *L*
by (*metis add-commutative add-right-upper-bound l-below-nu order-trans*)
thus *mu-nu-apx-nu f* **using** *1*
by (*metis apx-def mu-nu-apx-nu-def nu-below-mu-nu-2-def*)
qed

— AACP Theorem 4.10 implies Theorem 4.11

lemma *mu-nu-apx-nu-mu-nu-apx-meet*: *has-least-fixpoint f* ∧ *has-greatest-fixpoint f* ∧ *mu-nu-apx-nu f* ⟶ *mu-nu-apx-meet f*
proof
let *?l* = *μ f* + (*ν f* ⌢ *L*)
assume *has-least-fixpoint f* ∧ *has-greatest-fixpoint f* ∧ *mu-nu-apx-nu f*
hence *is-apx-meet* (*μ f*) (*ν f*) *?l*
by (*smt add-apx-left-isotone add-commutative apx-meet-L is-apx-meet-def l-apx-mu less-eq-def meet.add-least-upper-bound mu-nu-apx-nu-def*)
thus *mu-nu-apx-meet f*
by (*smt apx-meet-char mu-nu-apx-meet-def*)
qed

— AACP Theorem 4.11 implies Theorem 4.12

lemma *mu-nu-apx-meet-apx-meet-below-nu*: *has-least-fixpoint f* ∧ *has-greatest-fixpoint f* ∧ *mu-nu-apx-meet f* ⟶ *apx-meet-below-nu f*
by (*metis apx-meet-below-nu-def l-below-nu mu-nu-apx-meet-def*)

— AACP Theorem 4.12 implies Theorem 4.9

lemma *apx-meet-below-nu-nu-below-mu-nu-2*: *apx-meet-below-nu f* ⟶ *nu-below-mu-nu-2 f*
proof −
let *?l* = *μ f* + (*ν f* ⌢ *L*)
have ∀ *m* . *m* ⊑ *μ f* ∧ *m* ⊑ *ν f* ∧ *m* ≤ *ν f* ⟶ *C* (*ν f*) ≤ *?l* + *n*(*?l*) ; *T*
proof
fix *m*
show *m* ⊑ *μ f* ∧ *m* ⊑ *ν f* ∧ *m* ≤ *ν f* ⟶ *C* (*ν f*) ≤ *?l* + *n*(*?l*) ; *T*
proof
assume *1*: *m* ⊑ *μ f* ∧ *m* ⊑ *ν f* ∧ *m* ≤ *ν f*
hence *m* ≤ *?l*
by (*smt add-commutative add-left-dist-meet add-left-upper-bound apx-def meet-less-eq-def meet.add-least-upper-bound*)
hence *m* + *n*(*m*) ; *T* ≤ *?l* + *n*(*?l*) ; *T*
by (*metis add-isotone mult-left-isotone n-isotone*)
thus *C* (*ν f*) ≤ *?l* + *n*(*?l*) ; *T* **using** *1*
by (*smt apx-def order-trans*)
qed
qed
thus *?thesis*
by (*smt apx-meet-below-nu-def apx-meet-same apx-meet-unique is-apx-meet-def nu-below-mu-nu-2-def*)
qed

— AACP Theorem 4.5 implies Theorem 4.6

lemma *has-apx-least-fixpoint-kappa-apx-meet*: *has-least-fixpoint f* ∧ *has-greatest-fixpoint f* ∧ *apx.has-least-fixpoint f* ⟶ *kappa-apx-meet f*
proof
assume *1*: *has-least-fixpoint f* ∧ *has-greatest-fixpoint f* ∧ *apx.has-least-fixpoint f*

hence *2*: $\forall\, w\ .\ w \sqsubseteq \mu\ f \wedge w \sqsubseteq \nu\ f \longrightarrow C\ (\kappa\ f) \leq w + n(w)\ ;\ T$
by (*metis apx-def meet.add-right-isotone order-trans kappa-below-nu*)
have $\forall\, w\ .\ w \sqsubseteq \mu\ f \wedge w \sqsubseteq \nu\ f \longrightarrow w \leq \kappa\ f + L$ **using** *1*
by (*metis add-left-isotone apx-def mu-below-kappa order-trans*)
hence $\forall\, w\ .\ w \sqsubseteq \mu\ f \wedge w \sqsubseteq \nu\ f \longrightarrow w \sqsubseteq \kappa\ f$ **using** *2*
by (*metis apx-def*)
hence *is-apx-meet* $(\mu\ f)\ (\nu\ f)\ (\kappa\ f)$ **using** *1*
by (*smt apx-meet-char is-apx-meet-def kappa-apx-below-mu kappa-apx-below-nu kappa-apx-meet-def*)
thus *kappa-apx-meet f* **using** *1*
by (*metis apx-meet-char kappa-apx-meet-def*)
qed

— AACP Theorem 4.6 implies Theorem 4.12

lemma *kappa-apx-meet-apx-meet-below-nu*: *has-greatest-fixpoint f* $\wedge$ *kappa-apx-meet f* $\longrightarrow$ *apx-meet-below-nu f*
by (*metis apx-meet-below-nu-def kappa-apx-meet-def kappa-below-nu*)

— AACP Theorem 4.12 implies Theorem 4.7

lemma *apx-meet-below-nu-kappa-mu-nu*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\wedge$ *isotone f* $\wedge$ *apx.isotone f* $\wedge$ *apx-meet-below-nu f* $\longrightarrow$ *kappa-mu-nu f*
proof
let *?l* $= \mu\ f + (\nu\ f \frown L)$
let *?m* $= \mu\ f \mathbin{\triangle} \nu\ f$
assume *1*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\wedge$ *isotone f* $\wedge$ *apx.isotone f* $\wedge$ *apx-meet-below-nu f*
hence *2*: *?m* $=$ *?l*
by (*metis apx-meet-below-nu-nu-below-mu-nu-2 mu-nu-apx-meet-def mu-nu-apx-nu-mu-nu-apx-meet nu-below-mu-nu-2-mu-nu-apx-nu*)
have *3*: $?l \leq f(?l) + L$
proof −
have $?l \leq \mu\ f + L$
by (*metis add-right-isotone meet.add-right-upper-bound*)
also have $\ldots = f(\mu\ f) + L$ **using** *1*
by (*metis is-least-fixpoint-def least-fixpoint*)
also have $\ldots \leq f(?l) + L$ **using** *1*
by (*metis add-left-isotone add-left-upper-bound isotone-def*)
finally show $?l \leq f(?l) + L$
by *metis*
qed
have $C\ (f(?l)) \leq ?l + n(?l)\ ;\ T$
proof −
have $C\ (f(?l)) \leq C\ (f(\nu\ f))$ **using** *1 2*
by (*metis apx-meet-below-nu-def isotone-def meet.add-right-isotone*)
also have $\ldots = C\ (\nu\ f)$ **using** *1*
by (*metis greatest-fixpoint is-greatest-fixpoint-def*)
also have $\ldots \leq ?l + n(?l)\ ;\ T$ **using** *1*
by (*metis apx-meet-below-nu-nu-below-mu-nu-2 nu-below-mu-nu-2-def*)
finally show $C\ (f(?l)) \leq ?l + n(?l)\ ;\ T$
by *metis*
qed
hence *4*: $?l \sqsubseteq f(?l)$ **using** *3*
by (*metis apx-def*)
have *5*: $f(?l) \sqsubseteq \mu\ f$
proof −
have $?l \sqsubseteq \mu\ f$
by (*metis l-apx-mu*)
thus $f(?l) \sqsubseteq \mu\ f$ **using** *1*
by (*metis apx.isotone-def is-least-fixpoint-def least-fixpoint*)
qed
have *6*: $f(?l) \sqsubseteq \nu\ f$
proof −
have $?l \sqsubseteq \nu\ f$ **using** *1 2*
by (*metis apx-greatest-lower-bound apx-meet-below-nu-def apx-reflexive*)
thus $f(?l) \sqsubseteq \nu\ f$ **using** *1*
by (*metis apx.isotone-def greatest-fixpoint is-greatest-fixpoint-def*)
qed
hence $f(?l) \sqsubseteq ?l$ **using** *1 2 5*
by (*metis apx-greatest-lower-bound apx-meet-below-nu-def*)
hence *7*: $f(?l) = ?l$ **using** *4*

by (*metis apx-antisymmetric*)
have $\forall\, y\ .\ f(y) = y \longrightarrow ?l \sqsubseteq y$
proof
fix y
show $f(y) = y \longrightarrow ?l \sqsubseteq y$
proof
assume *8*: $f(y) = y$
hence *9*: $?l \leq y + L$ **using** *1*
by (*metis add-isotone is-least-fixpoint-def least-fixpoint meet.add-right-upper-bound*)
have $y \leq \nu\ f$ **using** *1 8*
by (*metis greatest-fixpoint is-greatest-fixpoint-def*)
hence $C\ y \leq ?l + n(?l)\ ;\ T$ **using** *1 4 6*
by (*smt apx-meet-below-nu-nu-below-mu-nu-2 meet.add-right-isotone nu-below-mu-nu-2-def order-trans*)
thus $?l \sqsubseteq y$ **using** *9*
by (*metis apx-def*)
qed
qed
thus *kappa-mu-nu f* **using** *1 2 7*
by (*smt apx.least-fixpoint-same apx.has-least-fixpoint-def apx.is-least-fixpoint-def kappa-mu-nu-def*)
qed

— AACP Theorem 4.7 implies Theorem 4.5

lemma *kappa-mu-nu-has-apx-least-fixpoint*: *kappa-mu-nu f* $\longrightarrow$ *apx.has-least-fixpoint f*
by (*metis kappa-mu-nu-def*)

— AACP Theorem 4.8 implies Theorem 4.7

lemma *nu-below-mu-nu-kappa-mu-nu*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\wedge$ *isotone f* $\wedge$ *apx.isotone f* $\wedge$ *nu-below-mu-nu f* $\longrightarrow$ *kappa-mu-nu f*
by (*metis apx-meet-below-nu-kappa-mu-nu mu-nu-apx-meet-apx-meet-below-nu mu-nu-apx-nu-mu-nu-apx-meet nu-below-mu-nu-nu-below-mu-nu-2 nu-below-mu-nu-2-mu-nu-apx-nu*)

— AACP Theorem 4.7 implies Theorem 4.8

lemma *kappa-mu-nu-nu-below-mu-nu*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\wedge$ *kappa-mu-nu f* $\longrightarrow$ *nu-below-mu-nu f*
by (*metis apx-meet-below-nu-nu-below-mu-nu-2 has-apx-least-fixpoint-kappa-apx-meet nu-below-mu-nu-2-nu-below-mu-nu kappa-apx-meet-apx-meet-below-nu kappa-mu-nu-has-apx-least-fixpoint*)

definition *kappa-mu-nu-L* :: $('a \Rightarrow 'a) \Rightarrow$ *bool*
where *kappa-mu-nu-L f* $\longleftrightarrow$ *apx.has-least-fixpoint f* $\wedge\ \kappa\ f = \mu\ f + n(\nu\ f)\ ;\ L$

definition *nu-below-mu-nu-L* :: $('a \Rightarrow 'a) \Rightarrow$ *bool*
where *nu-below-mu-nu-L f* $\longleftrightarrow\ C\ (\nu\ f) \leq \mu\ f + n(\nu\ f)\ ;\ T$

definition *mu-nu-apx-nu-L* :: $('a \Rightarrow 'a) \Rightarrow$ *bool*
where *mu-nu-apx-nu-L f* $\longleftrightarrow\ \mu\ f + n(\nu\ f)\ ;\ L \sqsubseteq \nu\ f$

definition *mu-nu-apx-meet-L* :: $('a \Rightarrow 'a) \Rightarrow$ *bool*
where *mu-nu-apx-meet-L f* $\longleftrightarrow$ *has-apx-meet* $(\mu\ f)\ (\nu\ f) \wedge \mu\ f \mathbin{\triangle} \nu\ f = \mu\ f + n(\nu\ f)\ ;\ L$

lemma *n-below-l*: $x + n(y)\ ;\ L \leq x + (y \frown L)$
by (*metis add-right-isotone n-L-decreasing-meet-L*)

lemma *n-equal-l*: *nu-below-mu-nu-L f* $\longrightarrow \mu\ f + n(\nu\ f)\ ;\ L = \mu\ f + (\nu\ f \frown L)$
proof
assume *nu-below-mu-nu-L f*
hence $\nu\ f \frown L \leq (\mu\ f + n(\nu\ f)\ ;\ T) \frown L$
by (*metis meet-L-below-C meet.add-least-upper-bound meet.add-right-upper-bound nu-below-mu-nu-L-def order-trans*)
also have $\ldots \leq \mu\ f + (n(\nu\ f)\ ;\ T \frown L)$
by (*metis add-left-dist-meet add-right-upper-bound meet.add-right-isotone*)
also have $\ldots \leq \mu\ f + n(\nu\ f)\ ;\ L$
by (*metis add-right-isotone n-vector-meet-L*)
finally have $\mu\ f + (\nu\ f \frown L) \leq \mu\ f + n(\nu\ f)\ ;\ L$
by (*metis add-least-upper-bound add-left-upper-bound*)
thus $\mu\ f + n(\nu\ f)\ ;\ L = \mu\ f + (\nu\ f \frown L)$
by (*metis antisym n-below-l*)
qed

— AACP Theorem 4.14 implies Theorem 4.8

lemma *nu-below-mu-nu-L-nu-below-mu-nu*: *nu-below-mu-nu-L f* ⟶ *nu-below-mu-nu f*
by (*metis add-associative add-right-top mult-left-dist-add n-equal-l nu-below-mu-nu-L-def nu-below-mu-nu-def*)

— AACP Theorem 4.14 implies Theorem 4.13

lemma *nu-below-mu-nu-L-kappa-mu-nu-L*: *has-least-fixpoint f* ∧ *has-greatest-fixpoint f* ∧ *isotone f* ∧ *apx.isotone f* ∧ *nu-below-mu-nu-L f* ⟶ *kappa-mu-nu-L f*
by (*metis n-equal-l nu-below-mu-nu-L-nu-below-mu-nu nu-below-mu-nu-kappa-mu-nu kappa-mu-nu-L-def kappa-mu-nu-def*)

— AACP Theorem 4.14 implies Theorem 4.15

lemma *nu-below-mu-nu-L-mu-nu-apx-nu-L*: *has-least-fixpoint f* ∧ *has-greatest-fixpoint f* ∧ *nu-below-mu-nu-L f* ⟶ *mu-nu-apx-nu-L f*
by (*metis mu-nu-apx-nu-L-def mu-nu-apx-nu-def n-equal-l nu-below-mu-nu-2-mu-nu-apx-nu nu-below-mu-nu-L-nu-below-mu-nu nu-below-mu-nu-nu-below-mu-nu-2*)

— AACP Theorem 4.14 implies Theorem 4.16

lemma *nu-below-mu-nu-L-mu-nu-apx-meet-L*: *has-least-fixpoint f* ∧ *has-greatest-fixpoint f* ∧ *nu-below-mu-nu-L f* ⟶ *mu-nu-apx-meet-L f*
by (*metis mu-nu-apx-meet-L-def mu-nu-apx-meet-def mu-nu-apx-nu-mu-nu-apx-meet n-equal-l nu-below-mu-nu-2-mu-nu-apx-nu nu-below-mu-nu-L-nu-below-mu-nu nu-below-mu-nu-nu-below-mu-nu-2*)

— AACP Theorem 4.15 implies Theorem 4.14

lemma *mu-nu-apx-nu-L-nu-below-mu-nu-L*: *has-least-fixpoint f* ∧ *has-greatest-fixpoint f* ∧ *mu-nu-apx-nu-L f* ⟶ *nu-below-mu-nu-L f*
proof
let *?n* = *μ f* + *n*(*ν f*) ; *L*
let *?l* = *μ f* + (*ν f* ⌢ *L*)
assume *1*: *has-least-fixpoint f* ∧ *has-greatest-fixpoint f* ∧ *mu-nu-apx-nu-L f*
hence *C* (*ν f*) ≤ *?n* + *n*(*?n*) ; *T*
by (*metis apx-def mu-nu-apx-nu-L-def*)
also have ... ≤ *?n* + *n*(*?l*) ; *T*
by (*metis add-right-isotone n-isotone mult-left-isotone n-below-l*)
also have ... ≤ *?n* + *n*(*ν f*) ; *T* **using** *1*
by (*metis add-right-isotone n-isotone l-below-nu mult-left-isotone*)
finally show *nu-below-mu-nu-L f*
by (*metis add-associative add-right-top mult-left-dist-add nu-below-mu-nu-L-def*)
qed

— AACP Theorem 4.13 implies Theorem 4.15

lemma *kappa-mu-nu-L-mu-nu-apx-nu-L*: *has-greatest-fixpoint f* ∧ *kappa-mu-nu-L f* ⟶ *mu-nu-apx-nu-L f*
by (*metis mu-nu-apx-nu-L-def kappa-apx-below-nu kappa-mu-nu-L-def*)

— AACP Theorem 4.16 implies Theorem 4.15

lemma *mu-nu-apx-meet-L-mu-nu-apx-nu-L*: *mu-nu-apx-meet-L f* ⟶ *mu-nu-apx-nu-L f*
by (*smt apx-meet-same has-apx-meet-def is-apx-meet-def mu-nu-apx-meet-L-def mu-nu-apx-nu-L-def*)

— AACP Theorem 4.13 implies Theorem 4.14

lemma *kappa-mu-nu-L-nu-below-mu-nu-L*: *has-least-fixpoint f* ∧ *has-greatest-fixpoint f* ∧ *kappa-mu-nu-L f* ⟶ *nu-below-mu-nu-L f*
by (*metis mu-nu-apx-nu-L-nu-below-mu-nu-L kappa-mu-nu-L-mu-nu-apx-nu-L*)

lemma *nu-below-mu-nu-nu-below-mu-nu-L*: *nu-below-mu-nu f* ⟶ *nu-below-mu-nu-L f* **nitpick** [*expect=genuine*] **oops**

lemma *unfold-fold-1*: *isotone f* ∧ *has-least-prefixpoint f* ∧ *apx.has-least-fixpoint f* ∧ *f*(*x*) ≤ *x* ⟶ *κ f* ≤ *x* + *L*
by (*metis add-left-isotone apx-def has-least-fixpoint-def is-least-prefixpoint-def least-prefixpoint-char least-prefixpoint-fixpoint order-trans pmu-mu kappa-apx-below-mu*)

lemma *unfold-fold-2*: *isotone f* ∧ *apx.isotone f* ∧ *has-least-prefixpoint f* ∧ *has-greatest-fixpoint f* ∧ *apx.has-least-fixpoint f* ∧ *f*(*x*) ≤ *x* ∧ *κ f* ⌢ *L* ≤ *x* ⌢ *L* ⟶ *κ f* ≤ *x*
proof
assume *1*: *isotone f* ∧ *apx.isotone f* ∧ *has-least-prefixpoint f* ∧ *has-greatest-fixpoint f* ∧ *apx.has-least-fixpoint f* ∧ *f*(*x*) ≤ *x*

$\wedge\ \kappa\ f \frown L \leq x \frown L$
hence $\kappa\ f \frown L = \nu\ f \frown L$
by (*metis apx-meet-L meet.add-left-isotone meet.antisym kappa-apx-below-nu kappa-below-nu*)
hence $\kappa\ f = (\kappa\ f \frown L) + \mu\ f$ **using** *1*
by (*metis apx-meet-below-nu-kappa-mu-nu has-apx-least-fixpoint-kappa-apx-meet add-commutative least-fixpoint-char least-prefixpoint-fixpoint kappa-apx-meet-apx-meet-below-nu kappa-mu-nu-def*)
thus $\kappa\ f \leq x$ **using** *1*
by (*metis add-least-upper-bound is-least-prefixpoint-def least-prefixpoint meet.add-least-upper-bound pmu-mu*)
qed

end

class *n-algebra-apx-2* = *n-algebra* + *apx* +
assumes *apx-def*: $x \sqsubseteq y \longleftrightarrow x \leq y + L \wedge y \leq x + n(x)\ ;\ T$

begin

lemma *apx-transitive-2*: $x \sqsubseteq y \wedge y \sqsubseteq z \longrightarrow x \sqsubseteq z$
proof
assume *1*: $x \sqsubseteq y \wedge y \sqsubseteq z$
hence $z \leq y + n(y)\ ;\ T$
by (*metis apx-def*)
also have $... \leq x + n(x)\ ;\ T + n(y)\ ;\ T$ **using** *1*
by (*metis add-left-isotone apx-def*)
also have $... \leq x + n(x)\ ;\ T$ **using** *1*
by (*metis add-associative add-idempotent add-right-isotone apx-def mult-left-isotone n-add-n-top n-isotone*)
finally show $x \sqsubseteq z$ **using** *1*
by (*smt add-associative add-commutative apx-def less-eq-def*)
qed

lemma *apx-meet-L*: $y \sqsubseteq x \longrightarrow x \frown L \leq y \frown L$
proof
assume *1*: $y \sqsubseteq x$
have $x \frown L \leq (y \frown L) + (n(y)\ ;\ T \frown L)$ **using** *1*
by (*metis apx-def meet-commutative meet-left-dist-add meet.add-left-isotone*)
also have $... \leq (y \frown L) + n(y \frown L)\ ;\ T$
by (*metis add-least-upper-bound n-vector-meet-L meet.add-least-upper-bound n-L-decreasing order-refl order-trans*)
finally show $x \frown L \leq y \frown L$
by (*metis less-eq-def meet.add-right-upper-bound n-less-eq-char*)
qed

— AACP Theorem 4.1

subclass *apx-biorder*
apply *unfold-locales*
apply (*metis add-left-upper-bound apx-def*)
apply (*metis add-same-context antisym apx-def apx-meet-L relative-equality*)
apply (*metis apx-transitive-2*)
done

lemma *add-apx-left-isotone-2*: $x \sqsubseteq y \longrightarrow x + z \sqsubseteq y + z$
proof
assume *1*: $x \sqsubseteq y$
hence *2*: $x + z \leq y + z + L$
by (*smt add-associative add-commutative add-left-isotone apx-def*)
have $y + z \leq x + n(x)\ ;\ T + z$ **using** *1*
by (*metis add-left-isotone apx-def*)
also have $... \leq x + z + n(x + z)\ ;\ T$
by (*metis add-associative add-commutative add-right-isotone mult-left-isotone n-right-upper-bound*)
finally show $x + z \sqsubseteq y + z$ **using** *2*
by (*metis apx-def*)
qed

lemma *mult-apx-left-isotone-2*: $x \sqsubseteq y \longrightarrow x\ ;\ z \sqsubseteq y\ ;\ z$
proof
assume *1*: $x \sqsubseteq y$
hence $x\ ;\ z \leq y\ ;\ z + L\ ;\ z$
by (*metis apx-def mult-left-isotone mult-right-dist-add*)
hence *2*: $x\ ;\ z \leq y\ ;\ z + L$

by (*metis add-commutative add-left-isotone n-L-below-L order-trans*)
have $y\ ;\ z \leq x\ ;\ z + n(x)\ ;\ T\ ;\ z$ **using** *1*
by (*metis apx-def mult-left-isotone mult-right-dist-add*)
also have $... \leq x\ ;\ z + n(x\ ;\ z)\ ;\ T$
by (*metis add-least-upper-bound add-left-upper-bound n-top-split*)
finally show $x\ ;\ z \sqsubseteq y\ ;\ z$ **using** *2*
by (*metis apx-def*)
qed

lemma *mult-apx-right-isotone-2*: $x \sqsubseteq y \longrightarrow z\ ;\ x \sqsubseteq z\ ;\ y$
proof
assume *1*: $x \sqsubseteq y$
hence $z\ ;\ x \leq z\ ;\ y + z\ ;\ L$
by (*metis apx-def mult-left-dist-add mult-right-isotone*)
also have $... \leq z\ ;\ y + z\ ;\ 0 + L$
by (*metis add-associative add-right-isotone n-L-split-L*)
finally have *2*: $z\ ;\ x \leq z\ ;\ y + L$
by (*metis add-right-zero mult-left-dist-add*)
have $z\ ;\ y \leq z\ ;\ (x + n(x)\ ;\ T)$ **using** *1*
by (*metis apx-def mult-right-isotone*)
also have $... = z\ ;\ x + z\ ;\ n(x)\ ;\ T$
by (*metis mult-associative mult-left-dist-add*)
also have $... \leq z\ ;\ x + n(z\ ;\ x)\ ;\ T$
by (*metis add-least-upper-bound add-left-upper-bound n-split-top*)
finally show $z\ ;\ x \sqsubseteq z\ ;\ y$ **using** *2*
by (*metis apx-def*)
qed

end

end

14 NOmegaAlgebra

theory *NOmegaAlgebra*

imports *OmegaAlgebra Recursion*

begin

class *itering-apx* = *bounded-itering* + *n-algebra-apx*

begin

lemma *circ-L*: $L^\circ = L + 1$
by (*metis add-commutative mult-top-circ n-L-top-L*)

lemma *C-circ-import*: $C\ (x^\circ) \leq (C\ x)^\circ$
proof −
have *1*: $C\ x\ ;\ x^\circ \leq (C\ x)^\circ\ ;\ C\ x$
by (*metis C-mult-propagate circ-simulate eq-refl*)
have $C\ (x^\circ) = C\ (1 + x\ ;\ x^\circ)$
by (*metis circ-left-unfold*)
also have ... $= C\ 1 + C\ (x\ ;\ x^\circ)$
by (*metis meet-left-dist-add*)
also have ... $\leq 1 + C\ (x\ ;\ x^\circ)$
by (*metis add-left-isotone meet.add-right-upper-bound*)
also have ... $= 1 + C\ x\ ;\ x^\circ$
by (*metis n-L-T-meet-mult*)
also have ... $\leq (C\ x)^\circ$ **using** *1*
by (*metis add-right-isotone circ-left-unfold circ-plus-same*)
finally show *?thesis*
.
qed

— AACP Theorem 4.3 and Theorem 4.4

lemma *circ-apx-isotone*: $x \sqsubseteq y \longrightarrow x^\circ \sqsubseteq y^\circ$
proof
assume $x \sqsubseteq y$
hence *1*: $x \leq y + L \wedge C\ y \leq x + n(x)\ ;\ T$
by (*metis apx-def*)
have $C\ (y^\circ) \leq (C\ y)^\circ$
by (*metis C-circ-import*)
also have ... $\leq x^\circ + x^\circ\ ;\ n(x)\ ;\ T$ **using** *1*
by (*metis circ-isotone circ-left-top circ-unfold-sum mult-associative*)
also have ... $\leq x^\circ + (x^\circ\ ;\ 0 + n(x^\circ\ ;\ x)\ ;\ T)$
by (*smt add-right-isotone n-n-top-split-n-top*)
also have ... $\leq x^\circ + (x^\circ\ ;\ 0 + n(x^\circ)\ ;\ T)$
by (*metis add-right-isotone mult-left-isotone n-isotone right-plus-below-circ*)
also have ... $= x^\circ + n(x^\circ)\ ;\ T$
by (*smt add-associative add-commutative less-eq-def zero-right-mult-decreasing*)
finally have *2*: $C\ (y^\circ) \leq x^\circ + n(x^\circ)\ ;\ T$
by *metis*
have $x^\circ \leq y^\circ\ ;\ L^\circ$ **using** *1*
by (*metis circ-add-1 circ-back-loop-fixpoint circ-isotone n-L-below-L less-eq-def mult-associative*)
also have ... $= y^\circ + y^\circ\ ;\ L$
by (*metis add-commutative circ-L mult-left-dist-add mult-right-one*)
also have ... $\leq y^\circ + y^\circ\ ;\ 0 + L$
by (*metis add-associative add-right-isotone n-L-split-L*)
finally have $x^\circ \leq y^\circ + L$
by (*metis add-commutative less-eq-def zero-right-mult-decreasing*)
thus $x^\circ \sqsubseteq y^\circ$ **using** *2*
by (*metis apx-def*)
qed

end

class *n-omega-algebra-1* = *bounded-left-zero-omega-algebra* + *n-algebra-apx* + *Omega* +
assumes *Omega-def*: $x^\Omega = n(x^\omega)\ ;\ L + x^\star$

begin

— AACP Theorem 8.13

lemma *C-omega-export*: $C\ (x^\omega) = (C\ x)^\omega$
proof −
have $C\ (x^\omega) = C\ x\ ;\ C\ (x^\omega)$
by (*metis C-mult-propagate n-L-T-meet-mult omega-unfold*)
hence *1*: $C\ (x^\omega) \leq (C\ x)^\omega$
by (*metis omega-induct-mult order.refl*)
have $(C\ x)^\omega = C\ (x\ ;\ (C\ x)^\omega)$
by (*metis n-L-T-meet-mult omega-unfold*)
also have $... \leq C\ (x^\omega)$
by (*metis calculation meet.add-least-upper-bound meet.add-left-upper-bound meet-commutative omega-isotone*)
finally show *?thesis* **using** *1*
by (*metis antisym*)
qed

— AACP Theorem 8.2

lemma *L-mult-star*: $L\ ;\ x^\star = L$
by (*metis less-eq-def mult-associative n-L-below-L star.circ-back-loop-fixpoint*)

— AACP Theorem 8.3

lemma *mult-L-star*: $(x\ ;\ L)^\star = 1 + x\ ;\ L$
by (*smt L-mult-star mult-associative star.circ-mult*)

lemma *mult-L-omega-below*: $(x\ ;\ L)^\omega \leq x\ ;\ L$
by (*metis mult-right-isotone n-L-below-L omega-slide*)

— AACP Theorem 8.5

lemma *mult-L-add-star*: $(x\ ;\ L + y)^\star = y^\star + y^\star\ ;\ x\ ;\ L$
by (*metis L-mult-star mult-associative star.circ-add-1 star.circ-decompose-6 star.circ-unfold-sum*)

lemma *mult-L-add-omega-below*: $(x\ ;\ L + y)^\omega \leq y^\omega + y^\star\ ;\ x\ ;\ L$
proof −
have $(x\ ;\ L + y)^\omega \leq y^\star\ ;\ x\ ;\ L + (y^\star\ ;\ x\ ;\ L)^\star\ ;\ y^\omega$
by (*metis add-commutative mult-associative omega-decompose add-left-isotone mult-L-omega-below*)
also have $... \leq y^\omega + y^\star\ ;\ x\ ;\ L$
by (*smt add-associative add-commutative less-eq-def mult-L-star mult-associative mult-left-dist-add mult-left-one mult-right-dist-add n-L-below-L order-refl*)
finally show *?thesis*
by *metis*
qed

lemma *n-Omega-isotone*: $x \leq y \longrightarrow x^\Omega \leq y^\Omega$
by (*metis Omega-def add-isotone mult-left-isotone n-isotone omega-isotone star-isotone*)

lemma *n-star-below-Omega*: $x^\star \leq x^\Omega$
by (*metis add-right-upper-bound Omega-def*)

lemma *mult-L-star-mult-below*: $(x\ ;\ L)^\star\ ;\ y \leq y + x\ ;\ L$
by (*metis add-right-isotone mult-associative mult-right-isotone n-L-below-L star-left-induct*)

end

sublocale *n-omega-algebra-1* < *star*!: *itering-apx* **where** *circ* = *star* **..**

class *n-omega-algebra* = *n-omega-algebra-1* + *n-algebra-apx* +
assumes *n-split-omega-mult*: $C\ (x^\omega) \leq x^\star\ ;\ n(x^\omega)\ ;\ T$
assumes *tarski*: $x\ ;\ L \leq x\ ;\ L\ ;\ x\ ;\ L$

begin

— AACP Theorem 8.4

lemma *mult-L-omega*: $(x\ ;\ L)^\omega = x\ ;\ L$

apply (*rule antisym*)
apply (*rule mult-L-omega-below*)
apply (*metis mult-associative omega-induct-mult tarski*)
done

— AACP Theorem 8.6

lemma *mult-L-add-omega*: $(x\ ;\ L + y)^\omega = y^\omega + y^\star\ ;\ x\ ;\ L$
apply (*rule antisym*)
apply (*rule mult-L-add-omega-below*)
apply (*metis add-right-isotone add-right-upper-bound less-eq-def mult-L-omega mult-associative mult-isotone omega-sub-dist star.circ-sub-dist star-mult-omega*)
done

— AACP Theorem 8.1

lemma *tarski-mult-top-idempotent*: $x\ ;\ L = x\ ;\ L\ ;\ x\ ;\ L$
by (*metis mult-L-omega mult-associative omega-unfold*)

— AACP Theorem 8.7

lemma *n-below-n-omega*: $n(x) \leq n(x^\omega)$
proof −
have $n(x)\ ;\ L \leq n(x)\ ;\ L\ ;\ n(x)\ ;\ L$
by (*metis tarski*)
also have $... \leq x\ ;\ n(x)\ ;\ L$
by (*metis mult-left-isotone n-L-decreasing*)
finally have $n(x)\ ;\ L \leq x^\omega$
by (*metis mult-associative omega-induct-mult*)
thus *?thesis*
by (*metis n-galois*)
qed

— AACP Theorem 8.14

lemma *n-split-omega-add-zero*: $C\ (x^\omega) \leq x^\star\ ;\ 0 + n(x^\omega)\ ;\ T$
proof −
have $n(x^\omega)\ ;\ T + x\ ;\ (x^\star\ ;\ 0 + n(x^\omega)\ ;\ T) = n(x^\omega)\ ;\ T + x\ ;\ x^\star\ ;\ 0 + x\ ;\ n(x^\omega)\ ;\ T$
by (*metis add-associative mult-associative mult-left-dist-add*)
also have $... \leq n(x^\omega)\ ;\ T + x\ ;\ x^\star\ ;\ 0 + x\ ;\ 0 + n(x^\omega)\ ;\ T$
by (*metis add-associative add-right-isotone n-n-top-split-n-top omega-unfold*)
also have $... = x\ ;\ x^\star\ ;\ 0 + n(x^\omega)\ ;\ T$
by (*smt add-associative add-commutative add-left-top add-right-zero mult-associative mult-left-dist-add*)
also have $... \leq x^\star\ ;\ 0 + n(x^\omega)\ ;\ T$
by (*metis add-left-isotone mult-left-isotone star.left-plus-below-circ*)
finally have $x^\star\ ;\ n(x^\omega)\ ;\ T \leq x^\star\ ;\ 0 + n(x^\omega)\ ;\ T$
by (*metis mult-associative star-left-induct*)
thus *?thesis*
by (*metis n-split-omega-mult order-trans*)
qed

lemma *n-split-omega-add*: $C\ (x^\omega) \leq x^\star + n(x^\omega)\ ;\ T$
by (*metis add-left-isotone n-split-omega-add-zero order-trans zero-right-mult-decreasing*)

— AACP Theorem 8.12

lemma *n-dist-omega-star*: $n(y^\omega + y^\star\ ;\ z) = n(y^\omega) + n(y^\star\ ;\ z)$
proof −
have $n(y^\omega + y^\star\ ;\ z) = n(C\ (y^\omega) + C\ (y^\star\ ;\ z))$
by (*metis meet-left-dist-add n-C*)
also have $... \leq n(C\ (y^\omega) + y^\star\ ;\ z)$
by (*smt2 add-least-upper-bound add-left-upper-bound add-right-upper-bound meet.add-left-upper-bound meet-commutative n-isotone order-trans*)
also have $... \leq n(y^\star\ ;\ 0 + n(y^\omega)\ ;\ T + y^\star\ ;\ z)$
by (*metis add-commutative add-right-isotone n-isotone n-split-omega-add-zero*)
also have $... = n(y^\omega) + n(y^\star\ ;\ z)$
by (*smt add-associative add-commutative add-right-zero mult-left-dist-add n-dist-n-add*)
finally show *?thesis*
by (*metis add-least-upper-bound n-left-upper-bound n-right-upper-bound antisym*)

qed

lemma *mult-L-add-circ-below*: $(x\ ;\ L + y)^{\Omega} \leq n(y^{\omega})\ ;\ L + y^{\star}\ ;\ y^{\star}\ ;\ x\ ;\ L$
proof −
have $(x\ ;\ L + y)^{\Omega} \leq n(y^{\omega} + y^{\star}\ ;\ x\ ;\ L)\ ;\ L + (x\ ;\ L + y)^{\star}$
by (*metis add-left-isotone mult-L-add-omega-below mult-left-isotone n-isotone Omega-def*)
also have $\ldots = n(y^{\omega})\ ;\ L + n(y^{\star}\ ;\ x\ ;\ L)\ ;\ L + (x\ ;\ L + y)^{\star}$
by (*metis mult-associative mult-right-dist-add n-dist-omega-star*)
also have $\ldots \leq n(y^{\omega})\ ;\ L + y^{\star} + y^{\star}\ ;\ x\ ;\ L$
by (*smt add-associative add-commutative add-idempotent add-right-isotone mult-L-add-star n-L-decreasing*)
finally show *?thesis*
by *metis*
qed

lemma *n-mult-omega-L-below-zero*: $n(y\ ;\ x^{\omega})\ ;\ L \leq y\ ;\ x^{\star}\ ;\ 0 + y\ ;\ n(x^{\omega})\ ;\ L$
proof −
have $n(y\ ;\ x^{\omega})\ ;\ L \leq C\ (y\ ;\ x^{\omega}) \frown L$
by (*metis n-C n-L-increasing n-galois n-n-L n-n-meet-L*)
also have $\ldots \leq y\ ;\ C\ (x^{\omega}) \frown L$
by (*metis meet.add-left-isotone n-L-T-meet-mult n-L-T-meet-mult-propagate*)
also have $\ldots \leq y\ ;\ (x^{\star}\ ;\ 0 + n(x^{\omega})\ ;\ T) \frown L$
by (*metis meet-commutative meet.add-right-isotone mult-right-isotone n-split-omega-add-zero*)
also have $\ldots = (y\ ;\ x^{\star}\ ;\ 0 \frown L) + (y\ ;\ n(x^{\omega})\ ;\ T \frown L)$
by (*metis meet-commutative meet-left-dist-add mult-associative mult-left-dist-add*)
also have $\ldots \leq (y\ ;\ x^{\star}\ ;\ 0 \frown L) + y\ ;\ n(x^{\omega})\ ;\ L$
by (*metis add-right-isotone n-vector-meet-L*)
also have $\ldots \leq y\ ;\ x^{\star}\ ;\ 0 + y\ ;\ n(x^{\omega})\ ;\ L$
by (*metis add-left-isotone meet.add-left-upper-bound*)
finally show *?thesis*
by *metis*
qed

— AACP Theorem 8.10

lemma *n-mult-omega-L-star-zero*: $y\ ;\ x^{\star}\ ;\ 0 + n(y\ ;\ x^{\omega})\ ;\ L = y\ ;\ x^{\star}\ ;\ 0 + y\ ;\ n(x^{\omega})\ ;\ L$
apply (*rule antisym*)
apply (*metis add-least-upper-bound mult-associative mult-left-dist-add mult-left-sub-dist-add-left n-mult-omega-L-below-zero*)
apply (*smt add-associative add-commutative add-left-zero add-right-isotone mult-associative mult-left-dist-add n-n-L-split-n-L*)
done

— AACP Theorem 8.11

lemma *n-mult-omega-L-star*: $y\ ;\ x^{\star} + n(y\ ;\ x^{\omega})\ ;\ L = y\ ;\ x^{\star} + y\ ;\ n(x^{\omega})\ ;\ L$
by (*metis zero-right-mult-decreasing n-mult-omega-L-star-zero add-relative-same-increasing*)

lemma *n-mult-omega-L-below*: $n(y\ ;\ x^{\omega})\ ;\ L \leq y\ ;\ x^{\star} + y\ ;\ n(x^{\omega})\ ;\ L$
by (*metis add-right-upper-bound n-mult-omega-L-star*)

lemma *n-omega-L-below-zero*: $n(x^{\omega})\ ;\ L \leq x\ ;\ x^{\star}\ ;\ 0 + x\ ;\ n(x^{\omega})\ ;\ L$
by (*smt omega-unfold n-mult-omega-L-below-zero add-left-isotone star.left-plus-below-circ order-trans*)

lemma *n-omega-L-below*: $n(x^{\omega})\ ;\ L \leq x^{\star} + x\ ;\ n(x^{\omega})\ ;\ L$
by (*metis omega-unfold n-mult-omega-L-below add-left-isotone star.left-plus-below-circ order-trans*)

lemma *n-omega-L-star-zero*: $x\ ;\ x^{\star}\ ;\ 0 + n(x^{\omega})\ ;\ L = x\ ;\ x^{\star}\ ;\ 0 + x\ ;\ n(x^{\omega})\ ;\ L$
by (*metis n-mult-omega-L-star-zero omega-unfold*)

— AACP Theorem 8.8

lemma *n-omega-L-star*: $x^{\star} + n(x^{\omega})\ ;\ L = x^{\star} + x\ ;\ n(x^{\omega})\ ;\ L$
by (*metis star.circ-mult-upper-bound star.left-plus-below-circ zero-least n-omega-L-star-zero add-relative-same-increasing*)

— AACP Theorem 8.9

lemma *n-omega-L-star-zero-star*: $x^{\star}\ ;\ 0 + n(x^{\omega})\ ;\ L = x^{\star}\ ;\ 0 + x^{\star}\ ;\ n(x^{\omega})\ ;\ L$
by (*metis n-mult-omega-L-star-zero star-mult-omega mult-associative star.circ-transitive-equal*)

— AACP Theorem 8.8

lemma *n-omega-L-star-star*: $x^\star + n(x^\omega) \mathbin{;} L = x^\star + x^\star \mathbin{;} n(x^\omega) \mathbin{;} L$
by (*metis zero-right-mult-decreasing n-omega-L-star-zero-star add-relative-same-increasing*)

lemma *n-Omega-left-unfold*: $1 + x \mathbin{;} x^\Omega = x^\Omega$
by (*smt Omega-def add-associative add-commutative mult-associative mult-left-dist-add n-omega-L-star star.circ-left-unfold*)

lemma *n-Omega-left-slide*: $(x \mathbin{;} y)^\Omega \mathbin{;} x \leq x \mathbin{;} (y \mathbin{;} x)^\Omega$
proof −
have $(x \mathbin{;} y)^\Omega \mathbin{;} x \leq x \mathbin{;} y \mathbin{;} n((x \mathbin{;} y)^\omega) \mathbin{;} L + (x \mathbin{;} y)^\star \mathbin{;} x$
by (*smt Omega-def add-commutative add-left-isotone mult-associative mult-right-dist-add mult-right-isotone n-L-below-L n-omega-L-star*)
also have $\ldots \leq x \mathbin{;} (y \mathbin{;} 0 + n(y \mathbin{;} (x \mathbin{;} y)^\omega) \mathbin{;} L) + (x \mathbin{;} y)^\star \mathbin{;} x$
by (*smt add-associative add-commutative less-eq-def mult-associative mult-left-dist-add mult-left-sub-dist-add-left n-n-L-split-n-L star.circ-slide*)
also have $\ldots = x \mathbin{;} (y \mathbin{;} x)^\Omega$
by (*smt Omega-def add-associative add-commutative less-eq-def mult-associative mult-isotone mult-left-dist-add omega-slide star.circ-increasing star.circ-slide zero-least*)
finally show *?thesis*
by *metis*
qed

lemma *n-Omega-add-1*: $(x + y)^\Omega = x^\Omega \mathbin{;} (y \mathbin{;} x^\Omega)^\Omega$
proof −
have *1*: $(x + y)^\Omega = n((x^\star \mathbin{;} y)^\omega) \mathbin{;} L + n((x^\star \mathbin{;} y)^\star \mathbin{;} x^\omega) \mathbin{;} L + (x^\star \mathbin{;} y)^\star \mathbin{;} x^\star$
by (*smt Omega-def mult-right-dist-add n-dist-omega-star omega-decompose star.circ-add*)
have $n((x^\star \mathbin{;} y)^\omega) \mathbin{;} L \leq (x^\star \mathbin{;} y)^\star + x^\star \mathbin{;} (y \mathbin{;} n((x^\star \mathbin{;} y)^\omega) \mathbin{;} L)$
by (*metis n-omega-L-below mult-associative*)
also have $\ldots \leq (x^\star \mathbin{;} y)^\star + x^\star \mathbin{;} y \mathbin{;} 0 + x^\star \mathbin{;} n((y \mathbin{;} x^\star)^\omega) \mathbin{;} L$
by (*smt add-associative add-right-isotone mult-associative mult-left-dist-add mult-right-isotone n-n-L-split-n-L omega-slide*)
also have $\ldots = (x^\star \mathbin{;} y)^\star + x^\star \mathbin{;} n((y \mathbin{;} x^\star)^\omega) \mathbin{;} L$
by (*metis add-commutative less-eq-def star.circ-sub-dist-1 zero-right-mult-decreasing*)
also have $\ldots \leq x^\star \mathbin{;} (y \mathbin{;} x^\star)^\star + x^\star \mathbin{;} n((y \mathbin{;} x^\star)^\omega) \mathbin{;} L$
by (*metis add-left-isotone mult-right-isotone star.circ-increasing star.circ-isotone star-decompose-3*)
also have $\ldots \leq x^\star \mathbin{;} (y \mathbin{;} x^\Omega)^\Omega$
by (*metis Omega-def add-commutative mult-associative mult-left-dist-add mult-right-isotone n-Omega-isotone n-star-below-Omega*)
also have $\ldots \leq x^\Omega \mathbin{;} (y \mathbin{;} x^\Omega)^\Omega$
by (*metis n-star-below-Omega mult-left-isotone*)
finally have *2*: $n((x^\star \mathbin{;} y)^\omega) \mathbin{;} L \leq x^\Omega \mathbin{;} (y \mathbin{;} x^\Omega)^\Omega$
by *metis*
have $n((x^\star \mathbin{;} y)^\star \mathbin{;} x^\omega) \mathbin{;} L \leq n(x^\omega) \mathbin{;} L + x^\star \mathbin{;} (y \mathbin{;} x^\star)^\star + x^\star \mathbin{;} (y \mathbin{;} x^\star)^\star \mathbin{;} y \mathbin{;} n(x^\omega) \mathbin{;} L$
by (*smt add-associative add-commutative mult-left-one mult-right-dist-add n-mult-omega-L-below star.circ-mult star.circ-slide*)
also have $\ldots = n(x^\omega) \mathbin{;} L \mathbin{;} (y \mathbin{;} x^\Omega)^\star + x^\star \mathbin{;} (y \mathbin{;} x^\Omega)^\star$
by (*smt Omega-def add-associative mult-L-add-star mult-associative mult-left-dist-add L-mult-star*)
also have $\ldots \leq x^\Omega \mathbin{;} (y \mathbin{;} x^\Omega)^\Omega$
by (*metis mult-right-dist-add Omega-def n-star-below-Omega mult-right-isotone*)
finally have *3*: $n((x^\star \mathbin{;} y)^\star \mathbin{;} x^\omega) \mathbin{;} L \leq x^\Omega \mathbin{;} (y \mathbin{;} x^\Omega)^\Omega$
by *metis*
have $(x^\star \mathbin{;} y)^\star \mathbin{;} x^\star \leq x^\Omega \mathbin{;} (y \mathbin{;} x^\Omega)^\Omega$
by (*metis star-slide mult-isotone mult-right-isotone n-star-below-Omega order-trans star-isotone*)
hence *4*: $(x + y)^\Omega \leq x^\Omega \mathbin{;} (y \mathbin{;} x^\Omega)^\Omega$ **using** *1 2 3*
by (*metis add-least-upper-bound*)
have *5*: $x^\Omega \mathbin{;} (y \mathbin{;} x^\Omega)^\Omega \leq n(x^\omega) \mathbin{;} L + x^\star \mathbin{;} n((y \mathbin{;} x^\Omega)^\omega) \mathbin{;} L + x^\star \mathbin{;} (y \mathbin{;} x^\Omega)^\star$
by (*smt Omega-def add-associative add-left-isotone mult-associative mult-left-dist-add mult-right-dist-add mult-right-isotone n-L-below-L*)
have $n(x^\omega) \mathbin{;} L \leq n((x^\star \mathbin{;} y)^\star \mathbin{;} x^\omega) \mathbin{;} L$
by (*metis add-commutative add-left-upper-bound mult-left-isotone n-isotone star.circ-loop-fixpoint*)
hence *6*: $n(x^\omega) \mathbin{;} L \leq (x + y)^\Omega$ **using** *1*
by (*metis Omega-def add-left-upper-bound n-Omega-isotone order-trans*)
have $x^\star \mathbin{;} n((y \mathbin{;} x^\Omega)^\omega) \mathbin{;} L \leq x^\star \mathbin{;} n((y \mathbin{;} x^\star)^\omega + (y \mathbin{;} x^\star)^\star \mathbin{;} y \mathbin{;} n(x^\omega) \mathbin{;} L) \mathbin{;} L$
by (*metis Omega-def mult-L-add-omega-below mult-associative mult-left-dist-add mult-left-isotone mult-right-isotone n-isotone*)
also have $\ldots \leq x^\star \mathbin{;} 0 + n(x^\star \mathbin{;} ((y \mathbin{;} x^\star)^\omega + (y \mathbin{;} x^\star)^\star \mathbin{;} y \mathbin{;} n(x^\omega) \mathbin{;} L)) \mathbin{;} L$
by (*metis n-n-L-split-n-L*)
also have $\ldots \leq x^\star + n((x^\star \mathbin{;} y)^\omega + x^\star \mathbin{;} (y \mathbin{;} x^\star)^\star \mathbin{;} y \mathbin{;} n(x^\omega) \mathbin{;} L) \mathbin{;} L$
by (*smt add-left-isotone mult-associative mult-left-dist-add omega-slide zero-right-mult-decreasing*)
also have $\ldots \leq x^\star + n((x^\star \mathbin{;} y)^\omega + (x^\star \mathbin{;} y)^\star \mathbin{;} n(x^\omega) \mathbin{;} L) \mathbin{;} L$
by (*smt add-right-divisibility add-right-isotone mult-left-isotone n-isotone star.circ-mult*)

also have ... $\leq x^\star + n((x + y)^\omega)\ ;\ L$
by (*metis add-right-isotone mult-associative mult-left-isotone mult-right-isotone n-L-decreasing n-isotone omega-decompose*)
also have ... $\leq (x + y)^\Omega$
by (*metis add-left-isotone star.circ-sub-dist Omega-def add-commutative*)
finally have *7*: $x^\star\ ;\ n((y\ ;\ x^\Omega)^\omega)\ ;\ L \leq (x + y)^\Omega$
by *metis*
have $x^\star\ ;\ (y\ ;\ x^\Omega)^\star \leq (x^\star\ ;\ y)^\star\ ;\ x^\star + (x^\star\ ;\ y)^\star\ ;\ n(x^\omega)\ ;\ L$
by (*smt Omega-def add-right-isotone mult-L-add-star mult-associative mult-left-dist-add mult-left-isotone star.left-plus-below-circ star-slide*)
also have ... $\leq (x^\star\ ;\ y)^\star\ ;\ x^\star + n((x^\star\ ;\ y)^\star\ ;\ x^\omega)\ ;\ L$
by (*metis add-associative add-right-isotone add-right-zero mult-left-dist-add n-n-L-split-n-L*)
also have ... $\leq (x + y)^\Omega$
by (*smt Omega-def add-commutative add-right-isotone mult-left-isotone n-right-upper-bound omega-decompose star.circ-add*)
finally have $n(x^\omega)\ ;\ L + x^\star\ ;\ n((y\ ;\ x^\Omega)^\omega)\ ;\ L + x^\star\ ;\ (y\ ;\ x^\Omega)^\star \leq (x + y)^\Omega$ **using** *6 7*
by (*metis add-least-upper-bound*)
hence $x^\Omega\ ;\ (y\ ;\ x^\Omega)^\Omega \leq (x + y)^\Omega$ **using** *5*
by (*smt order-trans*)
thus *?thesis* **using** *4*
by (*metis antisym*)
qed

end

sublocale *n-omega-algebra* < *nL-omega*!: *left-zero-conway-semiring* **where** *circ* = *Omega*
apply *unfold-locales*
apply (*metis n-Omega-left-unfold*)
apply (*metis n-Omega-left-slide*)
apply (*metis n-Omega-add-1*)
done

context *n-omega-algebra*

begin

— AACP Theorem 8.16

lemma *omega-apx-isotone*: $x \sqsubseteq y \longrightarrow x^\omega \sqsubseteq y^\omega$
proof
assume $x \sqsubseteq y$
hence *1*: $x \leq y + L \wedge C\ y \leq x + n(x)\ ;\ T$
by (*metis apx-def*)
have $n(x)\ ;\ T + x\ ;\ (x^\omega + n(x^\omega)\ ;\ T) \leq n(x)\ ;\ T + x^\omega + n(x^\omega)\ ;\ T$
by (*smt add-associative mult-associative mult-left-dist-add add-right-isotone n-n-top-split-n-top add-right-zero omega-unfold*)
also have ... $\leq x^\omega + n(x^\omega)\ ;\ T$
by (*metis add-commutative add-right-isotone mult-left-isotone n-below-n-omega add-associative add-idempotent*)
finally have *2*: $x^\star\ ;\ n(x)\ ;\ T \leq x^\omega + n(x^\omega)\ ;\ T$
by (*metis mult-associative star-left-induct*)
have $C\ (y^\omega) = (C\ y)^\omega$
by (*metis C-omega-export*)
also have ... $\leq (x + n(x)\ ;\ T)^\omega$ **using** *1*
by (*metis omega-isotone*)
also have ... $= (x^\star\ ;\ n(x)\ ;\ T)^\omega + (x^\star\ ;\ n(x)\ ;\ T)^\star\ ;\ x^\omega$
by (*metis mult-associative omega-decompose*)
also have ... $\leq x^\star\ ;\ n(x)\ ;\ T + (x^\star\ ;\ n(x)\ ;\ T)^\star\ ;\ x^\omega$
by (*metis add-left-isotone mult-top-omega*)
also have ... $= x^\star\ ;\ n(x)\ ;\ T + (1 + x^\star\ ;\ n(x)\ ;\ T\ ;\ (x^\star\ ;\ n(x)\ ;\ T)^\star)\ ;\ x^\omega$
by (*metis mult-associative star.circ-left-top star.mult-top-circ*)
also have ... $\leq x^\omega + x^\star\ ;\ n(x)\ ;\ T$
by (*smt add-isotone add-least-upper-bound mult-associative mult-left-one mult-right-dist-add mult-right-isotone order-refl top-greatest*)
also have ... $\leq x^\omega + n(x^\omega)\ ;\ T$ **using** *2*
by (*metis add-least-upper-bound add-left-upper-bound*)
finally have *3*: $C\ (y^\omega) \leq x^\omega + n(x^\omega)\ ;\ T$
by *metis*
have $x^\omega \leq (y + L)^\omega$ **using** *1*
by (*metis omega-isotone*)
also have ... $= (y^\star\ ;\ L)^\omega + (y^\star\ ;\ L)^\star\ ;\ y^\omega$

by (*metis omega-decompose*)
also have ... = $y^\star\ ;\ L\ ;\ (y^\star\ ;\ L)^\omega + (y^\star\ ;\ L)^\star\ ;\ y^\omega$
by (*metis omega-unfold*)
also have ... ≤ $y^\star\ ;\ L + (y^\star\ ;\ L)^\star\ ;\ y^\omega$
by (*metis add-left-isotone n-L-below-L mult-associative mult-right-isotone*)
also have ... = $y^\star\ ;\ L + (1 + y^\star\ ;\ L\ ;\ (y^\star\ ;\ L)^\star)\ ;\ y^\omega$
by (*metis star.circ-left-unfold*)
also have ... ≤ $y^\star\ ;\ L + y^\omega$
by (*metis add-commutative add-least-upper-bound add-right-upper-bound mult-L-star-mult-below mult-associative star.circ-mult star.circ-slide*)
also have ... ≤ $y^\star\ ;\ 0 + L + y^\omega$
by (*metis add-left-isotone n-L-split-L*)
finally have $x^\omega \leq y^\omega + L$
by (*metis add-associative add-commutative less-eq-def star-zero-below-omega*)
thus $x^\omega \sqsubseteq y^\omega$ **using** *3*
by (*metis apx-def*)
qed

lemma *combined-apx-left-isotone*: $x \sqsubseteq y \longrightarrow n(x^\omega)\ ;\ L + x^\star\ ;\ z \sqsubseteq n(y^\omega)\ ;\ L + y^\star\ ;\ z$
by (*metis add-apx-isotone mult-apx-left-isotone omega-apx-isotone star.circ-apx-isotone n-L-apx-isotone*)

lemma *combined-apx-left-isotone-2*: $x \sqsubseteq y \longrightarrow (x^\omega \frown L) + x^\star\ ;\ z \sqsubseteq (y^\omega \frown L) + y^\star\ ;\ z$
by (*metis add-apx-isotone mult-apx-left-isotone omega-apx-isotone star.circ-apx-isotone meet-L-apx-isotone*)

lemma *combined-apx-right-isotone*: $y \sqsubseteq z \longrightarrow n(x^\omega)\ ;\ L + x^\star\ ;\ y \sqsubseteq n(x^\omega)\ ;\ L + x^\star\ ;\ z$
by (*metis add-apx-right-isotone mult-apx-right-isotone*)

lemma *combined-apx-right-isotone-2*: $y \sqsubseteq z \longrightarrow (x^\omega \frown L) + x^\star\ ;\ y \sqsubseteq (x^\omega \frown L) + x^\star\ ;\ z$
by (*metis add-apx-right-isotone mult-apx-right-isotone*)

lemma *combined-apx-isotone*: $x \sqsubseteq y \wedge w \sqsubseteq z \longrightarrow n(x^\omega)\ ;\ L + x^\star\ ;\ w \sqsubseteq n(y^\omega)\ ;\ L + y^\star\ ;\ z$
by (*metis add-apx-isotone mult-apx-isotone omega-apx-isotone star.circ-apx-isotone n-L-apx-isotone*)

lemma *combined-apx-isotone-2*: $x \sqsubseteq y \wedge w \sqsubseteq z \longrightarrow (x^\omega \frown L) + x^\star\ ;\ w \sqsubseteq (y^\omega \frown L) + y^\star\ ;\ z$
by (*metis add-apx-isotone mult-apx-isotone omega-apx-isotone star.circ-apx-isotone meet-L-apx-isotone*)

lemma *n-split-nu-mu*: $C\ (y^\omega + y^\star\ ;\ z) \leq y^\star\ ;\ z + n(y^\omega + y^\star\ ;\ z)\ ;\ T$
proof −
have $C\ (y^\omega + y^\star\ ;\ z) \leq C\ (y^\omega) + y^\star\ ;\ z$
by (*metis add-right-isotone meet.add-right-upper-bound meet-left-dist-add*)
also have ... ≤ $y^\star\ ;\ 0 + n(y^\omega)\ ;\ T + y^\star\ ;\ z$
by (*metis add-commutative add-right-isotone n-split-omega-add-zero*)
also have ... ≤ $y^\star\ ;\ z + n(y^\omega + y^\star\ ;\ z)\ ;\ T$
by (*smt add-associative add-commutative add-right-isotone add-right-zero mult-left-dist-add mult-left-isotone n-left-upper-bound*)
finally show *?thesis*
by *metis*
qed

lemma *n-split-nu-mu-2*: $C\ (y^\omega + y^\star\ ;\ z) \leq y^\star\ ;\ z + ((y^\omega + y^\star\ ;\ z) \frown L) + n(y^\omega + y^\star\ ;\ z)\ ;\ T$
proof −
have $C\ (y^\omega + y^\star\ ;\ z) \leq C\ (y^\omega) + y^\star\ ;\ z$
by (*metis add-right-isotone meet.add-right-upper-bound meet-left-dist-add*)
also have ... ≤ $y^\star\ ;\ 0 + n(y^\omega)\ ;\ T + y^\star\ ;\ z$
by (*metis add-commutative add-right-isotone n-split-omega-add-zero*)
also have ... ≤ $y^\star\ ;\ z + n(y^\omega + y^\star\ ;\ z)\ ;\ T$
by (*smt add-associative add-commutative add-right-isotone add-right-zero mult-left-dist-add mult-left-isotone n-left-upper-bound*)
finally show *?thesis*
by (*smt2 add-associative add-commutative meet-left-dist-add meet-commutative add-right-upper-bound order-trans*)
qed

lemma *loop-exists*: $C\ (\nu\ (\lambda x\ .\ y\ ;\ x + z)) \leq \mu\ (\lambda x\ .\ y\ ;\ x + z) + n(\nu\ (\lambda x\ .\ y\ ;\ x + z))\ ;\ T$
by (*metis n-split-nu-mu omega-loop-nu star-loop-mu*)

lemma *loop-exists-2*: $C\ (\nu\ (\lambda x\ .\ y\ ;\ x + z)) \leq \mu\ (\lambda x\ .\ y\ ;\ x + z) + (\nu\ (\lambda x\ .\ y\ ;\ x + z) \frown L) + n(\nu\ (\lambda x\ .\ y\ ;\ x + z))\ ;\ T$
by (*metis n-split-nu-mu-2 omega-loop-nu star-loop-mu*)

lemma *loop-apx-least-fixpoint*: *apx.is-least-fixpoint* $(\lambda x\ .\ y\ ;\ x + z)\ (\mu\ (\lambda x\ .\ y\ ;\ x + z) + n(\nu\ (\lambda x\ .\ y\ ;\ x + z))\ ;\ L)$

proof −
have *kappa-mu-nu-L* $(\lambda x\ .\ y\ ;\ x + z)$
by (*metis affine-apx-isotone loop-exists affine-has-greatest-fixpoint affine-has-least-fixpoint affine-isotone nu-below-mu-nu-L-def nu-below-mu-nu-L-kappa-mu-nu-L*)
thus *?thesis*
by (*smt apx.least-fixpoint-char kappa-mu-nu-L-def*)
qed

lemma *loop-apx-least-fixpoint-2*: *apx.is-least-fixpoint* $(\lambda x\ .\ y\ ;\ x + z)$ $(\mu\ (\lambda x\ .\ y\ ;\ x + z) + (\nu\ (\lambda x\ .\ y\ ;\ x + z) \frown L))$
proof −
have *kappa-mu-nu* $(\lambda x\ .\ y\ ;\ x + z)$
by (*metis affine-apx-isotone affine-has-greatest-fixpoint affine-has-least-fixpoint affine-isotone loop-exists-2 nu-below-mu-nu-def nu-below-mu-nu-kappa-mu-nu*)
thus *?thesis*
by (*smt apx.least-fixpoint-char kappa-mu-nu-def*)
qed

lemma *loop-has-apx-least-fixpoint*: *apx.has-least-fixpoint* $(\lambda x\ .\ y\ ;\ x + z)$
by (*metis apx.has-least-fixpoint-def loop-apx-least-fixpoint-2*)

lemma *loop-semantics*: $\kappa\ (\lambda x\ .\ y\ ;\ x + z) = \mu\ (\lambda x\ .\ y\ ;\ x + z) + n(\nu\ (\lambda x\ .\ y\ ;\ x + z))\ ;\ L$
by (*metis apx.least-fixpoint-char loop-apx-least-fixpoint*)

lemma *loop-semantics-2*: $\kappa\ (\lambda x\ .\ y\ ;\ x + z) = \mu\ (\lambda x\ .\ y\ ;\ x + z) + (\nu\ (\lambda x\ .\ y\ ;\ x + z) \frown L)$
by (*metis apx.least-fixpoint-char loop-apx-least-fixpoint-2*)

— AACP Theorem 8.15

lemma *loop-semantics-kappa-mu-nu*: $\kappa\ (\lambda x\ .\ y\ ;\ x + z) = n(y^\omega)\ ;\ L + y^\star\ ;\ z$
proof −
have $\kappa\ (\lambda x\ .\ y\ ;\ x + z) = y^\star\ ;\ z + n(y^\omega + y^\star\ ;\ z)\ ;\ L$
by (*metis loop-semantics omega-loop-nu star-loop-mu*)
thus *?thesis*
by (*smt n-dist-omega-star add-associative mult-right-dist-add add-commutative less-eq-def n-L-decreasing*)
qed

— AACP Theorem 8.15

lemma *loop-semantics-kappa-mu-nu-2*: $\kappa\ (\lambda x\ .\ y\ ;\ x + z) = (y^\omega \frown L) + y^\star\ ;\ z$
proof −
have $\kappa\ (\lambda x\ .\ y\ ;\ x + z) = y^\star\ ;\ z + ((y^\omega + y^\star\ ;\ z) \frown L)$
by (*metis loop-semantics-2 omega-loop-nu star-loop-mu*)
thus *?thesis*
by (*smt add-absorb add-associative add-commutative add-left-dist-meet*)
qed

— AACP Theorem 8.16

lemma *loop-semantics-apx-left-isotone*: $w \sqsubseteq y \longrightarrow \kappa\ (\lambda x\ .\ w\ ;\ x + z) \sqsubseteq \kappa\ (\lambda x\ .\ y\ ;\ x + z)$
by (*metis loop-semantics-kappa-mu-nu-2 combined-apx-left-isotone-2*)

— AACP Theorem 8.16

lemma *loop-semantics-apx-right-isotone*: $w \sqsubseteq z \longrightarrow \kappa\ (\lambda x\ .\ y\ ;\ x + w) \sqsubseteq \kappa\ (\lambda x\ .\ y\ ;\ x + z)$
by (*metis loop-semantics-kappa-mu-nu-2 combined-apx-right-isotone-2*)

lemma *loop-semantics-apx-isotone*: $v \sqsubseteq y \wedge w \sqsubseteq z \longrightarrow \kappa\ (\lambda x\ .\ v\ ;\ x + w) \sqsubseteq \kappa\ (\lambda x\ .\ y\ ;\ x + z)$
by (*metis loop-semantics-kappa-mu-nu-2 combined-apx-isotone-2*)

end

end

15 NOmegaAlgebraBinaryItering

theory *NOmegaAlgebraBinaryItering*

imports *NOmegaAlgebra BinaryIteringStrict*

begin

sublocale *extended-binary-itering* < *left-zero-conway-semiring* **where** *circ* = $(\lambda x\ .\ x \star 1)$
apply *unfold-locales*
apply (*metis while-left-unfold*)
apply (*metis mult-right-one while-one-mult-below while-slide*)
apply (*metis while-one-while while-sumstar-2*)
done

class *binary-itering-apx* = *bounded-binary-itering* + *n-algebra-apx*

begin

lemma *C-while-import*: $C\ (x \star z) = C\ (C\ x \star z)$
proof −
have *1*: $C\ x\ ;\ (x \star z) \leq C\ x \star (C\ x\ ;\ z)$
by (*metis C-mult-propagate eq-refl while-simulate*)
have $C\ (x \star z) = C\ z + C\ x\ ;\ (x \star z)$
by (*metis meet-left-dist-add n-L-T-meet-mult while-left-unfold*)
also have ... $\leq C\ x \star z$ **using** *1*
by (*metis add-isotone meet.add-right-upper-bound while-right-unfold*)
finally have $C\ (x \star z) \leq C\ (C\ x \star z)$
by (*metis meet.add-least-upper-bound meet.add-left-upper-bound*)
thus *?thesis*
by (*smt2 meet.add-left-upper-bound meet.less-eq-def meet-associative meet-commutative while-left-isotone*)
qed

lemma *C-while-preserve*: $C\ (x \star z) = C\ (x \star C\ z)$
proof −
have $C\ x\ ;\ (x \star z) \leq C\ x \star (C\ x\ ;\ z)$
by (*metis C-mult-propagate eq-refl while-simulate*)
also have ... $\leq x \star (x\ ;\ C\ z)$
by (*metis C-decreasing n-L-T-meet-mult-propagate while-isotone*)
finally have *1*: $C\ x\ ;\ (x \star z) \leq x \star (x\ ;\ C\ z)$
by *metis*
have $C\ (x \star z) = C\ z + C\ x\ ;\ (x \star z)$
by (*metis meet-left-dist-add n-L-T-meet-mult while-left-unfold*)
also have ... $\leq x \star C\ z$ **using** *1*
by (*metis add-least-upper-bound while-increasing while-mult-increasing while-mult-transitive*)
finally have $C\ (x \star z) \leq C\ (x \star C\ z)$
by (*metis meet.add-least-upper-bound meet.add-left-upper-bound*)
thus *?thesis*
by (*smt2 meet.add-left-upper-bound meet.less-eq-def meet-associative meet-commutative while-right-isotone*)
qed

lemma *C-while-import-preserve*: $C\ (x \star z) = C\ (C\ x \star C\ z)$
by (*metis C-while-import C-while-preserve*)

lemma *while-L-L*: $L \star L = L$
by (*metis n-L-top-L while-mult-star-exchange while-right-top*)

lemma *while-L-below-add*: $L \star x \leq x + L$
by (*metis while-left-unfold add-right-isotone n-L-below-L*)

lemma *while-L-split*: $x \star L \leq (x \star y) + L$
proof −
have $x \star L \leq (x \star 0) + L$
by (*metis add-commutative add-left-zero mult-right-one n-L-split-L while-right-unfold while-simulate-left-plus while-zero*)
thus *?thesis*
by (*metis add-commutative add-right-isotone order-trans while-right-isotone zero-least*)
qed

lemma *while-n-while-top-split*: $x \star (n(x \star y)\ ;\ T) \leq (x \star 0) + n(x \star y)\ ;\ T$

proof −
have $x \ ; \ n(x \star y) \ ; \ T \leq x \ ; \ 0 + n(x \ ; \ (x \star y)) \ ; \ T$
by (*metis n-n-top-split-n-top*)
also have ... $\leq n(x \star y) \ ; \ T + x \ ; \ 0$
by (*metis add-commutative add-right-isotone mult-left-isotone n-isotone while-left-plus-below*)
finally have $x \star (n(x \star y) \ ; \ T) \leq n(x \star y) \ ; \ T + (x \star (x \ ; \ 0))$
by (*metis mult-associative mult-right-one while-simulate-left mult-left-zero while-left-top*)
also have ... $\leq (x \star 0) + n(x \star y) \ ; \ T$
by (*metis add-least-upper-bound add-left-isotone while-right-plus-below*)
finally show *?thesis*
by *metis*
qed

lemma *circ-apx-right-isotone*: $x \sqsubseteq y \longrightarrow z \star x \sqsubseteq z \star y$
proof
assume $x \sqsubseteq y$
hence *1*: $x \leq y + L \wedge C \ y \leq x + n(x) \ ; \ T$
by (*metis apx-def*)
hence $z \star x \leq (z \star y) + (z \star L)$
by (*metis while-left-dist-add while-right-isotone*)
hence *2*: $z \star x \leq (z \star y) + L$
by (*smt add-least-upper-bound add-left-upper-bound while-L-split order-trans*)
have $z \star (n(z \star x) \ ; \ T) \leq (z \star 0) + n(z \star x) \ ; \ T$
by (*metis while-n-while-top-split*)
also have ... $\leq (z \star x) + n(z \star x) \ ; \ T$
by (*metis add-left-isotone while-right-isotone zero-least*)
finally have *3*: $z \star (n(x) \ ; \ T) \leq (z \star x) + n(z \star x) \ ; \ T$
by (*metis mult-left-isotone n-isotone order-trans while-increasing while-right-isotone*)
have $C \ (z \star y) \leq z \star C \ y$
by (*metis C-while-preserve meet.add-right-divisibility*)
also have ... $\leq (z \star x) + (z \star (n(x) \ ; \ T))$ **using** *1*
by (*metis while-left-dist-add while-right-isotone*)
also have ... $\leq (z \star x) + n(z \star x) \ ; \ T$ **using** *3*
by (*metis add-least-upper-bound add-left-upper-bound*)
finally show $z \star x \sqsubseteq z \star y$ **using** *2*
by (*metis apx-def*)
qed

end

class *extended-binary-itering-apx = binary-itering-apx + bounded-extended-binary-itering +*
assumes *n-below-while-zero*: $n(x) \leq n(x \star 0)$

begin

lemma *circ-apx-right-isotone*: $x \sqsubseteq y \longrightarrow x \star z \sqsubseteq y \star z$
proof
assume $x \sqsubseteq y$
hence *1*: $x \leq y + L \wedge C \ y \leq x + n(x) \ ; \ T$
by (*metis apx-def*)
hence $x \star z \leq ((y \star 1) \ ; \ L) \star (y \star z)$
by (*metis while-left-isotone while-sumstar-3*)
also have ... $\leq (y \star z) + (y \star 1) \ ; \ L$
by (*metis while-productstar add-right-isotone mult-right-isotone n-L-below-L while-slide*)
also have ... $\leq (y \star z) + L$
by (*metis add-commutative add-least-upper-bound add-right-upper-bound order-trans while-L-split while-one-mult-below*)
finally have *2*: $x \star z \leq (y \star z) + L$
by *metis*
have $C \ (y \star z) \leq C \ y \star z$
by (*metis C-while-import meet.add-right-divisibility*)
also have ... $\leq ((x \star 1) \ ; \ n(x) \ ; \ T) \star (x \star z)$ **using** *1*
by (*metis while-left-isotone mult-associative while-sumstar-3*)
also have ... $\leq (x \star z) + (x \star 1) \ ; \ n(x) \ ; \ T$
by (*metis while-productstar add-left-top add-right-isotone mult-associative mult-left-sub-dist-add-right while-slide*)
also have ... $\leq (x \star z) + (x \star (n(x) \ ; \ T))$
by (*metis add-right-isotone mult-associative while-one-mult-below*)
also have ... $\leq (x \star z) + (x \star (n(x \star z) \ ; \ T))$
by (*metis n-below-while-zero zero-least while-right-isotone n-isotone mult-left-isotone add-right-isotone order-trans*)
also have ... $\leq (x \star z) + n(x \star z) \ ; \ T$

by (*smt add-associative add-right-isotone while-n-while-top-split add-right-zero while-left-dist-add*)
finally show $x \star z \sqsubseteq y \star z$ **using** *2*
by (*metis apx-def*)
qed

lemma *while-top*: $T \star x = L + T \ ; \ x$ **nitpick** [*expect=genuine*] **oops**
lemma *while-one-top*: $1 \star x = L + x$ **nitpick** [*expect=genuine*] **oops**
lemma *while-unfold-below-1*: $x = y \ ; \ x \longrightarrow x \leq y \star 1$ **nitpick** [*expect=genuine*] **oops**

lemma *while-square-1*: $x \star 1 = (x \ ; \ x) \star (x + 1)$ **oops**
lemma *while-absorb-below-one*: $y \ ; \ x \leq x \longrightarrow y \star x \leq 1 \star x$ **oops**
lemma *while-mult-L*: $(x \ ; \ L) \star z = z + x \ ; \ L$ **oops**
lemma *tarski-top-omega-below-2*: $x \ ; \ L \leq (x \ ; \ L) \star 0$ **oops**
lemma *tarski-top-omega-2*: $x \ ; \ L = (x \ ; \ L) \star 0$ **oops**
lemma *while-separate-right-plus*: $y \ ; \ x \leq x \ ; \ (x \star (1 + y)) + 1 \longrightarrow y \star (x \star z) \leq x \star (y \star z)$ **oops**
lemma $y \star (x \star 1) \leq x \star (y \star 1) \longrightarrow (x + y) \star 1 = x \star (y \star 1)$ **oops**
lemma $y \ ; \ x \leq (1 + x) \ ; \ (y \star 1) \longrightarrow (x + y) \star 1 = x \star (y \star 1)$ **oops**

end

class *n-omega-algebra-binary* = *n-omega-algebra* + *while* +
assumes *while-def*: $x \star y = n(x^\omega) \ ; \ L + x^\star \ ; \ y$

begin

lemma *while-omega-meet-L-star*: $x \star y = (x^\omega \frown L) + x^\star \ ; \ y$
by (*metis loop-semantics-kappa-mu-nu loop-semantics-kappa-mu-nu-2 while-def*)

lemma *while-one-mult-while-below-1*: $(y \star 1) \ ; \ (y \star v) \leq y \star v$
proof −
have $(y \star 1) \ ; \ (y \star v) \leq y \star (y \star v)$
by (*smt add-left-isotone mult-associative mult-right-dist-add mult-right-isotone n-L-below-L while-def mult-left-one*)
also have ... $= n(y^\omega) \ ; \ L + y^\star \ ; \ n(y^\omega) \ ; \ L + y^\star \ ; \ y^\star \ ; \ v$
by (*metis while-def mult-left-dist-add add-associative mult-associative*)
also have ... $= n(y^\omega) \ ; \ L + n(y^\star \ ; \ y^\omega) \ ; \ L + y^\star \ ; \ y^\star \ ; \ v$
by (*smt n-mult-omega-L-star-zero add-relative-same-increasing add-associative add-left-zero mult-left-sub-dist-add-left add-commutative*)
finally show *?thesis*
by (*metis add-idempotent star.circ-transitive-equal star-mult-omega while-def*)
qed

lemma *star-below-while*: $x^\star \ ; \ y \leq x \star y$
by (*metis add-right-upper-bound while-def*)

subclass *bounded-binary-itering*
proof *unfold-locales*
fix *x y z*
have $z + x \ ; \ ((y \ ; \ x) \star (y \ ; \ z)) = x \ ; \ (y \ ; \ x)^\star \ ; \ y \ ; \ z + x \ ; \ n((y \ ; \ x)^\omega) \ ; \ L + z$
by (*smt add-associative add-commutative mult-associative mult-left-dist-add while-def*)
also have ... $= x \ ; \ (y \ ; \ x)^\star \ ; \ y \ ; \ z + n(x \ ; \ (y \ ; \ x)^\omega) \ ; \ L + z$
by (*metis mult-associative mult-right-isotone zero-least n-mult-omega-L-star-zero add-relative-same-increasing*)
also have ... $= (x \ ; \ y)^\star \ ; \ z + n(x \ ; \ (y \ ; \ x)^\omega) \ ; \ L$
by (*smt add-associative add-commutative mult-associative star.circ-loop-fixpoint star-slide*)
also have ... $= (x \ ; \ y) \star z$
by (*smt omega-slide while-def add-commutative*)
finally show $(x \ ; \ y) \star z = z + x \ ; \ ((y \ ; \ x) \star (y \ ; \ z))$
by *metis*
next
fix *x y z*
have $(x \star y) \star (x \star z) = n((n(x^\omega) \ ; \ L + x^\star \ ; \ y)^\omega) \ ; \ L + (n(x^\omega) \ ; \ L + x^\star \ ; \ y)^\star \ ; \ (x \star z)$
by (*metis while-def*)
also have ... $= n((x^\star \ ; \ y)^\omega + (x^\star \ ; \ y)^\star \ ; \ n(x^\omega) \ ; \ L) \ ; \ L + ((x^\star \ ; \ y)^\star + (x^\star \ ; \ y)^\star \ ; \ n(x^\omega) \ ; \ L) \ ; \ (x \star z)$
by (*metis mult-L-add-star mult-L-add-omega*)
also have ... $= n((x^\star \ ; \ y)^\omega) \ ; \ L + n((x^\star \ ; \ y)^\star \ ; \ n(x^\omega) \ ; \ L) \ ; \ L + (x^\star \ ; \ y)^\star \ ; \ (x \star z) + (x^\star \ ; \ y)^\star \ ; \ n(x^\omega) \ ; \ L \ ; \ (x \star z)$
by (*metis mult-associative n-dist-omega-star mult-right-dist-add add-associative*)
also have ... $= n((x^\star \ ; \ y)^\omega) \ ; \ L + n((x^\star \ ; \ y)^\star \ ; \ n(x^\omega) \ ; \ L) \ ; \ L + (x^\star \ ; \ y)^\star \ ; \ 0 + (x^\star \ ; \ y)^\star \ ; \ (x \star z) + (x^\star \ ; \ y)^\star \ ; \ n(x^\omega) \ ; \ L \ ; \ (x \star z)$
by (*smt add-associative add-left-zero mult-left-dist-add*)
also have ... $= n((x^\star \ ; \ y)^\omega) \ ; \ L + ((x^\star \ ; \ y)^\star \ ; \ n(x^\omega) \ ; \ L \ ; \ (x \star z) + (x^\star \ ; \ y)^\star \ ; \ n(x^\omega) \ ; \ L + (x^\star \ ; \ y)^\star \ ; \ (x \star z))$

by (*smt n-n-L-split-n-n-L-L add-commutative add-associative*)
also have ... = $n((x^\star\ ;\ y)^\omega)\ ;\ L + ((x^\star\ ;\ y)^\star\ ;\ n(x^\omega)\ ;\ L + (x^\star\ ;\ y)^\star\ ;\ (x \star z))$
by (*smt mult-L-omega omega-sub-vector less-eq-def*)
also have ... = $n((x^\star\ ;\ y)^\omega)\ ;\ L + (x^\star\ ;\ y)^\star\ ;\ (x \star z)$
by (*metis add-left-divisibility mult-associative mult-right-isotone while-def less-eq-def*)
also have ... = $(x^\star\ ;\ y)^\star\ ;\ x^\star\ ;\ z + (x^\star\ ;\ y)^\star\ ;\ n(x^\omega)\ ;\ L + n((x^\star\ ;\ y)^\omega)\ ;\ L$
by (*metis add-commutative mult-associative mult-left-dist-add while-def*)
also have ... = $(x^\star\ ;\ y)^\star\ ;\ x^\star\ ;\ z + n((x^\star\ ;\ y)^\star\ ;\ x^\omega)\ ;\ L + n((x^\star\ ;\ y)^\omega)\ ;\ L$
by (*metis add-right-zero mult-left-dist-add add-associative n-mult-omega-L-star-zero*)
also have ... = $(x + y) \star z$
by (*metis add-associative add-commutative omega-decompose star.circ-add while-def mult-right-dist-add n-dist-omega-star*)
finally show $(x + y) \star z = (x \star y) \star (x \star z)$
by *metis*
next
fix *x y z*
show $x \star (y + z) = (x \star y) + (x \star z)$
by (*smt add-associative add-commutative add-left-upper-bound less-eq-def mult-left-dist-add while-def*)
next
fix *x y z*
show $(x \star y)\ ;\ z \leq x \star (y\ ;\ z)$
by (*smt add-left-isotone mult-associative mult-right-dist-add mult-right-isotone n-L-below-L while-def*)
next
fix *v w x y z*
show $x\ ;\ z \leq z\ ;\ (y \star 1) + w \longrightarrow x \star (z\ ;\ v) \leq z\ ;\ (y \star v) + (x \star (w\ ;\ (y \star v)))$
proof
assume *1*: $x\ ;\ z \leq z\ ;\ (y \star 1) + w$
have $z\ ;\ v + x\ ;\ (z\ ;\ (y \star v) + x^\star\ ;\ (w\ ;\ (y \star v))) \leq z\ ;\ v + x\ ;\ z\ ;\ (y \star v) + x^\star\ ;\ (w\ ;\ (y \star v))$
by (*metis add-associative add-right-isotone mult-associative mult-left-dist-add mult-left-isotone star.left-plus-below-circ*)
also have ... $\leq z\ ;\ v + z\ ;\ (y \star 1)\ ;\ (y \star v) + w\ ;\ (y \star v) + x^\star\ ;\ (w\ ;\ (y \star v))$ **using** *1*
by (*metis add-associative add-left-isotone add-right-isotone mult-left-isotone mult-right-dist-add*)
also have ... $\leq z\ ;\ v + z\ ;\ (y \star v) + x^\star\ ;\ (w\ ;\ (y \star v))$
by (*smt add-least-upper-bound add-right-upper-bound less-eq-def mult-associative mult-left-dist-add star.circ-loop-fixpoint while-one-mult-while-below-1*)
also have ... = $z\ ;\ (y \star v) + x^\star\ ;\ (w\ ;\ (y \star v))$
by (*metis less-eq-def mult-left-dist-add mult-left-one mult-right-sub-dist-add-left order-trans star.circ-plus-one star-below-while*)
finally have $x^\star\ ;\ z\ ;\ v \leq z\ ;\ (y \star v) + x^\star\ ;\ (w\ ;\ (y \star v))$
by (*metis mult-associative star-left-induct*)
thus $x \star (z\ ;\ v) \leq z\ ;\ (y \star v) + (x \star (w\ ;\ (y \star v)))$
by (*smt add-associative add-commutative add-right-isotone mult-associative while-def*)
qed
next
fix *v w x y z*
show $z\ ;\ x \leq y\ ;\ (y \star z) + w \longrightarrow z\ ;\ (x \star v) \leq y \star (z\ ;\ v + w\ ;\ (x \star v))$
proof
assume $z\ ;\ x \leq y\ ;\ (y \star z) + w$
hence *1*: $z\ ;\ x \leq y\ ;\ y^\star\ ;\ z + (y\ ;\ n(y^\omega)\ ;\ L + w)$
by (*smt add-associative add-commutative mult-associative mult-left-dist-add while-def*)
hence $z\ ;\ x^\star \leq y^\star\ ;\ (z + (y\ ;\ n(y^\omega)\ ;\ L + w)\ ;\ x^\star)$
by (*metis star.circ-simulate-right-plus*)
also have ... = $y^\star\ ;\ z + y^\star\ ;\ y\ ;\ n(y^\omega)\ ;\ L + y^\star\ ;\ w\ ;\ x^\star$
by (*smt add-associative mult-associative mult-left-dist-add mult-right-dist-add L-mult-star*)
also have ... = $y^\star\ ;\ z + n(y^\star\ ;\ y\ ;\ y^\omega)\ ;\ L + y^\star\ ;\ w\ ;\ x^\star$
by (*metis add-relative-same-increasing mult-isotone n-mult-omega-L-star-zero star.left-plus-below-circ star.right-plus-circ zero-least*)
also have ... = $n(y^\omega)\ ;\ L + y^\star\ ;\ z + y^\star\ ;\ w\ ;\ x^\star$
by (*metis add-commutative omega-unfold right-plus-omega*)
finally have $z\ ;\ x^\star\ ;\ v \leq n(y^\omega)\ ;\ L\ ;\ v + y^\star\ ;\ z\ ;\ v + y^\star\ ;\ w\ ;\ x^\star\ ;\ v$
by (*smt less-eq-def mult-right-dist-add*)
also have ... $\leq n(y^\omega)\ ;\ L + y^\star\ ;\ (z\ ;\ v + w\ ;\ x^\star\ ;\ v)$
by (*metis n-L-below-L mult-associative mult-right-isotone add-left-isotone mult-left-dist-add add-associative*)
also have ... $\leq n(y^\omega)\ ;\ L + y^\star\ ;\ (z\ ;\ v + w\ ;\ (x \star v))$
by (*metis add-commutative add-right-isotone mult-associative mult-left-sub-dist-add-left mult-right-isotone while-def*)
finally have *2*: $z\ ;\ x^\star\ ;\ v \leq y \star (z\ ;\ v + w\ ;\ (x \star v))$
by (*metis while-def*)
have *3*: $y^\star\ ;\ y\ ;\ y^\star\ ;\ 0 \leq y^\star\ ;\ w\ ;\ x^\omega$
by (*metis add-commutative add-left-zero mult-associative mult-left-sub-dist-add-left star.circ-loop-fixpoint star.circ-transitive-equal*)
have $z\ ;\ x^\omega \leq y\ ;\ y^\star\ ;\ z\ ;\ x^\omega + (y\ ;\ n(y^\omega)\ ;\ L + w)\ ;\ x^\omega$ **using** *1*

by (*metis mult-associative mult-left-isotone mult-right-dist-add omega-unfold*)
hence $z\ ;\ x^\omega \leq y^\omega + y^\star\ ;\ y\ ;\ n(y^\omega)\ ;\ L\ ;\ x^\omega + y^\star\ ;\ w\ ;\ x^\omega$
by (*smt add-associative add-commutative left-plus-omega mult-associative mult-left-dist-add mult-right-dist-add omega-induct star.left-plus-circ*)
also have $\ldots \leq y^\omega + y^\star\ ;\ y\ ;\ n(y^\omega)\ ;\ L + y^\star\ ;\ w\ ;\ x^\omega$
by (*metis add-left-isotone add-right-isotone mult-associative mult-right-isotone n-L-below-L*)
also have $\ldots = y^\omega + n(y^\star\ ;\ y\ ;\ y^\omega)\ ;\ L + y^\star\ ;\ w\ ;\ x^\omega$ **using** *3*
by (*smt add-associative add-commutative add-relative-same-increasing n-mult-omega-L-star-zero*)
also have $\ldots = y^\omega + y^\star\ ;\ w\ ;\ x^\omega$
by (*metis mult-associative omega-unfold star-mult-omega add-commutative less-eq-def n-L-decreasing*)
finally have $n(z\ ;\ x^\omega)\ ;\ L \leq n(y^\omega)\ ;\ L + n(y^\star\ ;\ w\ ;\ x^\omega)\ ;\ L$
by (*metis mult-associative mult-left-isotone mult-right-dist-add n-dist-omega-star n-isotone*)
also have $\ldots \leq n(y^\omega)\ ;\ L + y^\star\ ;\ (w\ ;\ (n(x^\omega)\ ;\ L + x^\star\ ;\ 0))$
by (*smt add-commutative add-right-isotone mult-associative mult-left-dist-add n-mult-omega-L-below-zero*)
also have $\ldots \leq n(y^\omega)\ ;\ L + y^\star\ ;\ (w\ ;\ (n(x^\omega)\ ;\ L + x^\star\ ;\ v))$
by (*metis add-right-isotone mult-right-isotone zero-least*)
also have $\ldots \leq n(y^\omega)\ ;\ L + y^\star\ ;\ (z\ ;\ v + w\ ;\ (n(x^\omega)\ ;\ L + x^\star\ ;\ v))$
by (*metis add-right-isotone mult-left-sub-dist-add-right*)
finally have *4*: $n(z\ ;\ x^\omega)\ ;\ L \leq y \star (z\ ;\ v + w\ ;\ (x \star v))$
by (*metis while-def*)
have $z\ ;\ (x \star v) = z\ ;\ n(x^\omega)\ ;\ L + z\ ;\ x^\star\ ;\ v$
by (*metis while-def mult-left-dist-add mult-associative*)
also have $\ldots = n(z\ ;\ x^\omega)\ ;\ L + z\ ;\ x^\star\ ;\ v$
by (*metis add-commutative add-relative-same-increasing mult-right-isotone n-mult-omega-L-star-zero zero-least*)
finally show $z\ ;\ (x \star v) \leq y \star (z\ ;\ v + w\ ;\ (x \star v))$ **using** *2 4*
by (*metis add-least-upper-bound*)
qed
qed

lemma *while-top*: $T \star x = L + T\ ;\ x$
by (*metis n-top-L star.circ-top star-omega-top while-def*)

lemma *while-one-top*: $1 \star x = L + x$
by (*smt mult-left-one n-top-L omega-one star-one while-def*)

lemma *while-finite-associative*: $x^\omega = 0 \longrightarrow (x \star y)\ ;\ z = x \star (y\ ;\ z)$
by (*metis add-left-zero mult-associative n-zero-L-zero while-def*)

lemma *while-while-one*: $y \star (x \star 1) = n(y^\omega)\ ;\ L + y^\star\ ;\ n(x^\omega)\ ;\ L + y^\star\ ;\ x^\star$
by (*metis add-associative mult-left-dist-add mult-right-one while-def mult-associative*)

— AACP Theorem 8.17

subclass *bounded-extended-binary-itering*
proof *unfold-locales*
fix $w\ x\ y\ z$
have $w\ ;\ (x \star y\ ;\ z) = n(w\ ;\ n(x^\omega)\ ;\ L)\ ;\ L + w\ ;\ x^\star\ ;\ y\ ;\ z$
by (*smt add-associative add-commutative add-left-zero mult-associative mult-left-dist-add n-n-L-split-n-n-L-L while-def*)
also have $\ldots \leq n((w\ ;\ n(x^\omega)\ ;\ L)^\omega)\ ;\ L + w\ ;\ x^\star\ ;\ y\ ;\ z$
by (*metis eq-refl mult-L-omega*)
also have $\ldots \leq n((w\ ;\ (x \star y))^\omega)\ ;\ L + w\ ;\ x^\star\ ;\ y\ ;\ z$
by (*smt add-left-isotone add-left-upper-bound mult-associative mult-left-isotone mult-right-isotone n-isotone omega-isotone while-def*)
also have $\ldots \leq n((w\ ;\ (x \star y))^\omega)\ ;\ L + w\ ;\ (x \star y)\ ;\ z$
by (*metis star-below-while mult-associative mult-left-isotone mult-right-isotone add-right-isotone*)
also have $\ldots \leq n((w\ ;\ (x \star y))^\omega)\ ;\ L + (w\ ;\ (x \star y))^\star\ ;\ (w\ ;\ (x \star y)\ ;\ z)$
by (*metis add-right-isotone add-right-upper-bound star.circ-loop-fixpoint*)
finally show $w\ ;\ (x \star y\ ;\ z) \leq (w\ ;\ (x \star y)) \star (w\ ;\ (x \star y)\ ;\ z)$
by (*metis while-def*)
qed

subclass *extended-binary-itering-apx*
apply *unfold-locales*
apply (*metis n-below-n-omega n-left-upper-bound n-n-L order-trans while-def*)
done

lemma *while-simulate-4-plus*: $y\ ;\ x \leq x\ ;\ (x \star (1 + y)) \longrightarrow y\ ;\ x\ ;\ x^\star \leq x\ ;\ (x \star (1 + y))$
proof
assume *1*: $y\ ;\ x \leq x\ ;\ (x \star (1 + y))$

have $x\ ;\ (x \star (1 + y)) = x\ ;\ n(x^\omega)\ ;\ L + x\ ;\ x^\star\ ;\ (1 + y)$
by (*metis mult-associative mult-left-dist-add while-def*)
also have ... $= n(x\ ;\ x^\omega)\ ;\ L + x\ ;\ x^\star\ ;\ (1 + y)$
by (*smt n-mult-omega-L-star-zero add-relative-same-increasing add-commutative add-right-zero mult-left-sub-dist-add-right*)
finally have *2*: $x\ ;\ (x \star (1 + y)) = n(x^\omega)\ ;\ L + x\ ;\ x^\star + x\ ;\ x^\star\ ;\ y$
by (*metis add-associative mult-left-dist-add mult-right-one omega-unfold*)
hence $x\ ;\ x^\star\ ;\ y\ ;\ x \leq x\ ;\ x^\star\ ;\ n(x^\omega)\ ;\ L + x\ ;\ x^\star\ ;\ x^\star\ ;\ x + x\ ;\ x^\star\ ;\ x\ ;\ x^\star\ ;\ y$ **using** *1*
by (*metis mult-associative mult-right-isotone mult-left-dist-add star-plus*)
also have ... $= n(x\ ;\ x^\star\ ;\ x^\omega)\ ;\ L + x\ ;\ x^\star\ ;\ x^\star\ ;\ x + x\ ;\ x^\star\ ;\ x\ ;\ x^\star\ ;\ y$
by (*smt n-mult-omega-L-star-zero add-relative-same-increasing add-commutative add-right-zero mult-left-sub-dist-add-right*)
also have ... $= n(x^\omega)\ ;\ L + x\ ;\ x^\star\ ;\ x + x\ ;\ x\ ;\ x^\star\ ;\ y$
by (*metis mult-associative omega-unfold star.circ-plus-same star.circ-transitive-equal star-mult-omega*)
also have ... $\leq n(x^\omega)\ ;\ L + x\ ;\ x^\star + x\ ;\ x^\star\ ;\ y$
by (*smt add-associative add-right-upper-bound less-eq-def mult-associative mult-right-dist-add star.circ-increasing star.circ-plus-same star.circ-transitive-equal*)
finally have *3*: $x\ ;\ x^\star\ ;\ y\ ;\ x \leq n(x^\omega)\ ;\ L + x\ ;\ x^\star + x\ ;\ x^\star\ ;\ y$
by *metis*
have $(n(x^\omega)\ ;\ L + x\ ;\ x^\star + x\ ;\ x^\star\ ;\ y)\ ;\ x \leq n(x^\omega)\ ;\ L + x\ ;\ x^\star\ ;\ x + x\ ;\ x^\star\ ;\ y\ ;\ x$
by (*metis mult-right-dist-add n-L-below-L mult-associative mult-right-isotone add-left-isotone*)
also have ... $\leq n(x^\omega)\ ;\ L + x\ ;\ x^\star + x\ ;\ x^\star\ ;\ y\ ;\ x$
by (*smt add-commutative add-left-isotone mult-associative mult-right-isotone star.left-plus-below-circ star-plus*)
also have ... $\leq n(x^\omega)\ ;\ L + x\ ;\ x^\star + x\ ;\ x^\star\ ;\ y$ **using** *3*
by (*metis add-least-upper-bound add-left-upper-bound*)
finally show $y\ ;\ x\ ;\ x^\star \leq x\ ;\ (x \star (1 + y))$ **using** *1 2*
by (*metis add-least-upper-bound star-right-induct*)
qed

lemma *while-simulate-4-omega*: $y\ ;\ x \leq x\ ;\ (x \star (1 + y)) \longrightarrow y\ ;\ x^\omega \leq x^\omega$
proof
assume *1*: $y\ ;\ x \leq x\ ;\ (x \star (1 + y))$
have $x\ ;\ (x \star (1 + y)) = x\ ;\ n(x^\omega)\ ;\ L + x\ ;\ x^\star\ ;\ (1 + y)$
by (*metis mult-associative mult-left-dist-add while-def*)
also have ... $= n(x\ ;\ x^\omega)\ ;\ L + x\ ;\ x^\star\ ;\ (1 + y)$
by (*smt n-mult-omega-L-star-zero add-relative-same-increasing add-commutative add-right-zero mult-left-sub-dist-add-right*)
finally have $x\ ;\ (x \star (1 + y)) = n(x^\omega)\ ;\ L + x\ ;\ x^\star + x\ ;\ x^\star\ ;\ y$
by (*metis add-associative mult-left-dist-add mult-right-one omega-unfold*)
hence $y\ ;\ x^\omega \leq n(x^\omega)\ ;\ L\ ;\ x^\omega + x\ ;\ x^\star\ ;\ x^\omega + x\ ;\ x^\star\ ;\ y\ ;\ x^\omega$ **using** *1*
by (*smt less-eq-def mult-associative mult-right-dist-add omega-unfold*)
also have ... $\leq x\ ;\ x^\star\ ;\ (y\ ;\ x^\omega) + x^\omega$
by (*metis add-left-isotone mult-L-omega omega-sub-vector mult-associative omega-unfold star-mult-omega n-L-decreasing less-eq-def add-commutative*)
finally have $y\ ;\ x^\omega \leq (x\ ;\ x^\star)^\omega + (x\ ;\ x^\star)^\star\ ;\ x^\omega$
by (*metis add-commutative omega-induct*)
thus $y\ ;\ x^\omega \leq x^\omega$
by (*metis add-idempotent left-plus-omega star-mult-omega*)
qed

lemma *while-square-1*: $x \star 1 = (x\ ;\ x) \star (x + 1)$
by (*metis mult-right-one omega-square star-square-2 while-def*)

lemma *while-absorb-below-one*: $y\ ;\ x \leq x \longrightarrow y \star x \leq 1 \star x$
by (*metis star-left-induct-mult add-isotone n-galois n-sub-nL while-def while-one-top*)

lemma *while-mult-L*: $(x\ ;\ L) \star z = z + x\ ;\ L$
by (*metis add-right-zero mult-left-zero while-denest-5 while-one-top while-productstar while-sumstar*)

lemma *tarski-top-omega-below-2*: $x\ ;\ L \leq (x\ ;\ L) \star 0$
by (*metis add-right-divisibility while-mult-L*)

lemma *tarski-top-omega-2*: $x\ ;\ L = (x\ ;\ L) \star 0$
by (*metis add-left-zero while-mult-L*)

lemma *while-sub-mult-one*: $x\ ;\ (1 \star y) \leq 1 \star x$ **nitpick** [*expect=genuine*] **oops**
lemma *while-unfold-below*: $x = z + y\ ;\ x \longrightarrow x \leq y \star z$ **nitpick** [*expect=genuine*] **oops**
lemma *while-loop-is-greatest-postfixpoint*: *is-greatest-postfixpoint* $(\lambda x\ .\ y\ ;\ x + z)\ (y \star z)$ **nitpick** [*expect=genuine*] **oops**
lemma *while-loop-is-greatest-fixpoint*: *is-greatest-fixpoint* $(\lambda x\ .\ y\ ;\ x + z)\ (y \star z)$ **nitpick** [*expect=genuine*] **oops**
lemma *while-denest-3*: $(x \star w) \star x^\omega = (x \star w)^\omega$ **nitpick** [*expect=genuine*] **oops**
lemma *while-mult-top*: $(x\ ;\ T) \star z = z + x\ ;\ T$ **nitpick** [*expect=genuine*] **oops**
lemma *tarski-below-top-omega*: $x \leq (x\ ;\ L)^\omega$ **nitpick** [*expect=genuine*] **oops**

lemma *tarski-mult-omega-omega*: $(x\ ;\ y^\omega)^\omega = x\ ;\ y^\omega$ **nitpick** [*expect=genuine*] **oops**
lemma *tarski-below-top-omega-2*: $x \leq (x\ ;\ L) \star 0$ **nitpick** [*expect=genuine*] **oops**
lemma $1 = (x\ ;\ 0) \star 1$ **nitpick** [*expect=genuine*] **oops**
lemma *tarski*: $x = 0 \vee T\ ;\ x\ ;\ T = T$ **nitpick** [*expect=genuine*] **oops**
lemma $(x + y) \star z = ((x \star 1)\ ;\ y) \star ((x \star 1)\ ;\ z)$ **nitpick** [*expect=genuine*] **oops**
lemma *while-top-2*: $T \star z = T\ ;\ z$ **nitpick** [*expect=genuine*] **oops**
lemma *while-mult-top-2*: $(x\ ;\ T) \star z = z + x\ ;\ T\ ;\ z$ **nitpick** [*expect=genuine*] **oops**
lemma *while-one-mult*: $(x \star 1)\ ;\ x = x \star x$ **nitpick** [*expect=genuine*] **oops**
lemma $(x \star 1)\ ;\ y = x \star y$ **nitpick** [*expect=genuine*] **oops**
lemma *while-associative*: $(x \star y)\ ;\ z = x \star (y\ ;\ z)$ **nitpick** [*expect=genuine*] **oops**
lemma *while-back-loop-is-fixpoint*: *is-fixpoint* $(\lambda x\ .\ x\ ;\ y + z)\ (z\ ;\ (y \star 1))$ **nitpick** [*expect=genuine*] **oops**
lemma $1 + x\ ;\ 0 = x \star 1$ **nitpick** [*expect=genuine*] **oops**
lemma $x = x\ ;\ (x \star 1)$ **nitpick** [*expect=genuine*] **oops**
lemma $x\ ;\ (x \star 1) = x \star 1$ **nitpick** [*expect=genuine*] **oops**
lemma $x \star 1 = x \star (1 \star 1)$ **nitpick** [*expect=genuine*] **oops**
lemma $(x + y) \star 1 = (x \star (y \star 1)) \star 1$ **nitpick** [*expect=genuine*] **oops**
lemma $z + y\ ;\ x = x \longrightarrow y \star z \leq x$ **nitpick** [*expect=genuine*] **oops**
lemma $y\ ;\ x = x \longrightarrow y \star x \leq x$ **nitpick** [*expect=genuine*] **oops**
lemma $z + x\ ;\ y = x \longrightarrow z\ ;\ (y \star 1) \leq x$ **nitpick** [*expect=genuine*] **oops**
lemma $x\ ;\ y = x \longrightarrow x\ ;\ (y \star 1) \leq x$ **nitpick** [*expect=genuine*] **oops**
lemma $x\ ;\ z = z\ ;\ y \longrightarrow x \star z \leq z\ ;\ (y \star 1)$ **nitpick** [*expect=genuine*] **oops**

lemma *while-unfold-below-1*: $x = y\ ;\ x \longrightarrow x \leq y \star 1$ **nitpick** [*expect=genuine*] **oops**
lemma $x^\omega \leq x^\omega\ ;\ x^\omega$ **oops**
lemma *tarski-omega-idempotent*: $x^{\omega\omega} = x^\omega$ **oops**

end

class *n-omega-algebra-binary-strict* = *n-omega-algebra-binary* + *circ* +
  assumes *L-left-zero*: $L\ ;\ x = L$
  assumes *circ-def*: $x^\circ = n(x^\omega)\ ;\ L + x^\star$

begin

subclass *strict-binary-itering*
  apply *unfold-locales*
  apply (*metis while-def mult-associative L-left-zero mult-right-dist-add*)
  apply (*metis circ-def while-def mult-right-one*)
  done

end

end

16 CappedOmega

theory *CappedOmega*

imports *OmegaAlgebra*

begin

class *capped-omega* =
fixes *capped-omega* :: $'a \Rightarrow 'a \Rightarrow 'a$ ($-^{\omega}-$ [100,100] 100)

class *capped-omega-algebra* = *bounded-left-zero-kleene-algebra* + *bounded-distributive-lattice* + *capped-omega* +
assumes *capped-omega-unfold*: $y^{\omega}{}_{v} = y \ ; \ y^{\omega}{}_{v} \frown v$
assumes *capped-omega-induct*: $x \leq (y \ ; \ x + z) \frown v \longrightarrow x \leq y^{\omega}{}_{v} + y^{\star} \ ; \ z$

— AACP Theorem 6.1

sublocale *capped-omega-algebra* < *capped*!: *bounded-left-zero-omega-algebra* **where** *omega* = $(\lambda y \ . \ y^{\omega}{}_{T})$
apply *unfold-locales*
apply (*metis capped-omega-unfold meet.add-right-zero*)
apply (*metis add-commutative capped-omega-induct meet.add-right-zero*)
done

context *capped-omega-algebra*

begin

— AACP Theorem 6.2

lemma *capped-omega-below-omega*: $y^{\omega}{}_{v} \leq y^{\omega}{}_{T}$
by (*metis capped.omega-induct-mult capped-omega-unfold meet.add-left-upper-bound*)

— AACP Theorem 6.3

lemma *capped-omega-below*: $y^{\omega}{}_{v} \leq v$
by (*metis capped-omega-unfold meet.add-left-divisibility meet-commutative*)

— AACP Theorem 6.4

lemma *capped-omega-one*: $1^{\omega}{}_{v} = v$
proof −
have $v \leq (1 \ ; \ v + 0) \frown v$
by (*metis add-right-zero meet-idempotent mult-left-one order.refl*)
hence $v \leq 1^{\omega}{}_{v} + 1^{\star} \ ; \ 0$
by (*metis capped-omega-induct*)
also have ... = $1^{\omega}{}_{v}$
by (*metis add-right-zero mult-left-one star-one*)
finally show *?thesis*
by (*metis capped-omega-below antisym*)
qed

— AACP Theorem 6.5

lemma *capped-omega-zero*: $0^{\omega}{}_{v} = 0$
by (*metis capped-omega-unfold meet-commutative meet-right-zero mult-left-zero*)

lemma *star-below-cap*: $y \leq u \wedge z \leq v \wedge u \ ; \ v \leq v \longrightarrow y^{\star} \ ; \ z \leq v$
by (*metis add-least-upper-bound dual-order.trans mult-left-isotone star-left-induct*)

lemma *capped-fix*: $y \leq u \wedge z \leq v \wedge u \ ; \ v \leq v \longrightarrow (y \ ; \ (y^{\omega}{}_{v} + y^{\star} \ ; \ z) + z) \frown v = y^{\omega}{}_{v} + y^{\star} \ ; \ z$
proof
assume *1*: $y \leq u \wedge z \leq v \wedge u \ ; \ v \leq v$
have $(y \ ; \ (y^{\omega}{}_{v} + y^{\star} \ ; \ z) + z) \frown v = (y \ ; \ y^{\omega}{}_{v} + y^{\star} \ ; \ z) \frown v$
by (*metis add-associative mult-left-dist-add star.circ-loop-fixpoint*)
also have ... = $(y \ ; \ y^{\omega}{}_{v} \frown v) + (y^{\star} \ ; \ z \frown v)$
by (*metis meet-commutative meet-left-dist-add*)
also have ... = $y^{\omega}{}_{v} + y^{\star} \ ; \ z$ **using** *1*
by (*metis capped-omega-unfold meet-less-eq-def star-below-cap*)
finally show $(y \ ; \ (y^{\omega}{}_{v} + y^{\star} \ ; \ z) + z) \frown v = y^{\omega}{}_{v} + y^{\star} \ ; \ z$

.
qed

lemma *capped-fixpoint*: $y \leq u \wedge z \leq v \wedge u \ ; \ v \leq v \longrightarrow$ *is-fixpoint* $(\lambda x \ . \ (y \ ; \ x + z) \frown v) \ (y^{\omega}{}_{v} + y^{\star} \ ; \ z)$
by (*metis capped-fix meet.is-fixpoint-def*)

lemma *capped-greatest-fixpoint*: $y \leq u \wedge z \leq v \wedge u \ ; \ v \leq v \longrightarrow$ *is-greatest-fixpoint* $(\lambda x \ . \ (y \ ; \ x + z) \frown v) \ (y^{\omega}{}_{v} + y^{\star} \ ; \ z)$
by (*smt2 capped-fix order-refl capped-omega-induct is-greatest-fixpoint-def*)

lemma *capped-postfixpoint*: $y \leq u \wedge z \leq v \wedge u \ ; \ v \leq v \longrightarrow$ *is-postfixpoint* $(\lambda x \ . \ (y \ ; \ x + z) \frown v) \ (y^{\omega}{}_{v} + y^{\star} \ ; \ z)$
by (*metis capped-fix eq-refl is-postfixpoint-def*)

lemma *capped-greatest-postfixpoint*: $y \leq u \wedge z \leq v \wedge u \ ; \ v \leq v \longrightarrow$ *is-greatest-postfixpoint* $(\lambda x \ . \ (y \ ; \ x + z) \frown v) \ (y^{\omega}{}_{v} + y^{\star} \ ; \ z)$
by (*smt2 capped-fix order-refl capped-omega-induct is-greatest-postfixpoint-def*)

— AACP Theorem 6.6

lemma *capped-nu*: $y \leq u \wedge z \leq v \wedge u \ ; \ v \leq v \longrightarrow \nu(\lambda x \ . \ (y \ ; \ x + z) \frown v) = y^{\omega}{}_{v} + y^{\star} \ ; \ z$
by (*metis capped-greatest-fixpoint greatest-fixpoint-same*)

lemma *capped-pnu*: $y \leq u \wedge z \leq v \wedge u \ ; \ v \leq v \longrightarrow p\nu(\lambda x \ . \ (y \ ; \ x + z) \frown v) = y^{\omega}{}_{v} + y^{\star} \ ; \ z$
by (*metis capped-greatest-postfixpoint greatest-postfixpoint-same*)

— AACP Theorem 6.7

lemma *unfold-capped-omega*: $y \leq u \wedge u \ ; \ v \leq v \longrightarrow y \ ; \ y^{\omega}{}_{v} = y^{\omega}{}_{v}$
by (*metis capped-omega-below meet.order-trans mult-isotone capped-omega-unfold meet-less-eq-def*)

— AACP Theorem 6.8

lemma *star-mult-capped-omega*: $y \leq u \wedge u \ ; \ v \leq v \longrightarrow y^{\star} \ ; \ y^{\omega}{}_{v} = y^{\omega}{}_{v}$
proof
assume $y \leq u \wedge u \ ; \ v \leq v$
hence $y \ ; \ y^{\omega}{}_{v} = y^{\omega}{}_{v}$
by (*metis unfold-capped-omega*)
hence $y^{\star} \ ; \ y^{\omega}{}_{v} \leq y^{\omega}{}_{v}$
by (*metis star-left-induct-mult-equal*)
thus $y^{\star} \ ; \ y^{\omega}{}_{v} = y^{\omega}{}_{v}$
by (*metis add-right-upper-bound antisym-conv star.circ-loop-fixpoint*)
qed

— AACP Theorem 6.9

lemma *star-zero-below-capped-omega-zero*: $y \leq u \wedge u \ ; \ v \leq v \longrightarrow y^{\star} \ ; \ 0 \leq y^{\omega}{}_{v} \ ; \ 0$
proof
assume $y \leq u \wedge u \ ; \ v \leq v$
hence $y \ ; \ y^{\omega}{}_{v} \leq v$
by (*metis capped-omega-below meet.order-trans mult-isotone*)
hence $y \ ; \ y^{\omega}{}_{v} = y^{\omega}{}_{v}$
by (*metis capped-omega-unfold meet-less-eq-def*)
thus $y^{\star} \ ; \ 0 \leq y^{\omega}{}_{v} \ ; \ 0$
by (*metis add-commutative add-left-zero mult-associative star-loop-least-fixpoint*)
qed

lemma *star-zero-below-capped-omega*: $y \leq u \wedge u \ ; \ v \leq v \longrightarrow y^{\star} \ ; \ 0 \leq y^{\omega}{}_{v}$
by (*metis order.trans zero-right-mult-decreasing star-zero-below-capped-omega-zero*)

lemma *capped-omega-induct-meet-zero*: $x \leq y \ ; \ x \frown v \longrightarrow x \leq y^{\omega}{}_{v} + y^{\star} \ ; \ 0$
by (*metis add-commutative add-left-zero capped-omega-induct*)

— AACP Theorem 6.10

lemma *capped-omega-induct-meet*: $y \leq u \wedge u \ ; \ v \leq v \longrightarrow x \leq y \ ; \ x \frown v \longrightarrow x \leq y^{\omega}{}_{v}$
by (*metis capped-omega-induct-meet-zero add-commutative less-eq-def star-zero-below-capped-omega*)

lemma *capped-omega-induct-equal*: $x = (y \ ; \ x + z) \frown v \longrightarrow x \leq y^{\omega}{}_{v} + y^{\star} \ ; \ z$
by (*metis capped-omega-induct order-refl*)

— AACP Theorem 6.11

lemma *capped-meet-nu*: $y \leq u \wedge u \ ; \ v \leq v \longrightarrow \nu(\lambda x \ . \ y \ ; \ x \frown v) = y^{\omega}{}_{v}$
proof
assume *1*: $y \leq u \wedge u \ ; \ v \leq v$
hence $y^{\omega}{}_{v} + y^{\star} \ ; \ 0 = y^{\omega}{}_{v}$
by (*smt star-zero-below-capped-omega less-eq-def add-commutative*)
hence $\nu(\lambda x \ . \ (y \ ; \ x + 0) \frown v) = y^{\omega}{}_{v}$ **using** *1*
by (*metis capped-nu zero-least*)
thus $\nu(\lambda x \ . \ y \ ; \ x \frown v) = y^{\omega}{}_{v}$
by (*simp add*: *add-right-zero*)
qed

lemma *capped-meet-pnu*: $y \leq u \wedge u \ ; \ v \leq v \longrightarrow p\nu(\lambda x \ . \ y \ ; \ x \frown v) = y^{\omega}{}_{v}$
proof
assume *1*: $y \leq u \wedge u \ ; \ v \leq v$
hence $y^{\omega}{}_{v} + y^{\star} \ ; \ 0 = y^{\omega}{}_{v}$
by (*smt star-zero-below-capped-omega less-eq-def add-commutative*)
hence $p\nu(\lambda x \ . \ (y \ ; \ x + 0) \frown v) = y^{\omega}{}_{v}$ **using** *1*
by (*metis capped-pnu zero-least*)
thus $p\nu(\lambda x \ . \ y \ ; \ x \frown v) = y^{\omega}{}_{v}$
by (*simp add*: *add-right-zero*)
qed

— AACP Theorem 6.12

lemma *capped-omega-isotone*: $y \leq u \wedge u \ ; \ v \leq v \longrightarrow t \leq y \longrightarrow t^{\omega}{}_{v} \leq y^{\omega}{}_{v}$
by (*metis capped-omega-induct-meet capped-omega-unfold less-eq-def meet.add-left-isotone mult-right-sub-dist-add-left*)

— AACP Theorem 6.13

lemma *capped-omega-simulation*: $y \leq u \wedge s \leq u \wedge u \ ; \ v \leq v \longrightarrow s \ ; \ t \leq y \ ; \ s \longrightarrow s \ ; \ t^{\omega}{}_{v} \leq y^{\omega}{}_{v}$
proof
assume *1*: $y \leq u \wedge s \leq u \wedge u \ ; \ v \leq v$
show $s \ ; \ t \leq y \ ; \ s \longrightarrow s \ ; \ t^{\omega}{}_{v} \leq y^{\omega}{}_{v}$
proof
assume *2*: $s \ ; \ t \leq y \ ; \ s$
have $s \ ; \ t^{\omega}{}_{v} \leq s \ ; \ t \ ; \ t^{\omega}{}_{v} \frown s \ ; \ v$
by (*metis capped-omega-unfold meet.add-least-upper-bound meet.add-right-upper-bound meet-commutative mult-associative mult-right-isotone*)
also have $\ldots \leq s \ ; \ t \ ; \ t^{\omega}{}_{v} \frown v$ **using** *1*
by (*metis meet.add-left-isotone meet-commutative mult-left-isotone order-trans*)
also have $\ldots \leq y \ ; \ s \ ; \ t^{\omega}{}_{v} \frown v$ **using** *2*
by (*metis meet.add-left-isotone mult-left-isotone*)
finally show $s \ ; \ t^{\omega}{}_{v} \leq y^{\omega}{}_{v}$ **using** *1*
by (*metis capped-omega-induct-meet mult-associative*)
qed
qed

lemma *capped-omega-slide-sub*: $s \leq u \wedge y \leq u \wedge u \ ; \ u \leq u \wedge u \ ; \ v \leq v \longrightarrow s \ ; \ (y \ ; \ s)^{\omega}{}_{v} \leq (s \ ; \ y)^{\omega}{}_{v}$
proof
assume *1*: $s \leq u \wedge y \leq u \wedge u \ ; \ u \leq u \wedge u \ ; \ v \leq v$
hence $s \ ; \ y \leq u$
by (*metis meet.order-trans mult-isotone*)
thus $s \ ; \ (y \ ; \ s)^{\omega}{}_{v} \leq (s \ ; \ y)^{\omega}{}_{v}$ **using** *1*
by (*metis mult-associative capped-omega-simulation order-refl*)
qed

— AACP Theorem 6.14

lemma *capped-omega-slide*: $s \leq u \wedge y \leq u \wedge u \ ; \ u \leq u \wedge u \ ; \ v \leq v \longrightarrow s \ ; \ (y \ ; \ s)^{\omega}{}_{v} = (s \ ; \ y)^{\omega}{}_{v}$
by (*smt antisym mult-associative mult-right-isotone capped-omega-unfold capped-omega-slide-sub meet.add-left-upper-bound order-trans*)

lemma *capped-omega-sub-dist*: $s \leq u \wedge y \leq u \wedge u \ ; \ v \leq v \longrightarrow s^{\omega}{}_{v} \leq (s + y)^{\omega}{}_{v}$
by (*metis capped-omega-isotone add-least-upper-bound add-left-upper-bound*)

— AACP Theorem 6.15

lemma *capped-omega-simulation-2*: $s \leq u \wedge y \leq u \wedge u \;;\; u \leq u \wedge u \;;\; v \leq v \longrightarrow y \;;\; s \leq s \;;\; y \longrightarrow (s \;;\; y)^{\omega}{}_{v} \leq s^{\omega}{}_{v}$
proof
assume *1*: $s \leq u \wedge y \leq u \wedge u \;;\; u \leq u \wedge u \;;\; v \leq v$
hence *2*: $s \;;\; y \leq u$
by (*metis mult-isotone order.trans*)
have *3*: $s \;;\; (s \;;\; y)^{\omega}{}_{v} \leq v$ **using** *1*
by (*metis capped-omega-below meet.order-trans mult-isotone*)
show $y \;;\; s \leq s \;;\; y \longrightarrow (s \;;\; y)^{\omega}{}_{v} \leq s^{\omega}{}_{v}$
proof
assume *4*: $y \;;\; s \leq s \;;\; y$
have $(s \;;\; y)^{\omega}{}_{v} = s \;;\; (y \;;\; s)^{\omega}{}_{v}$ **using** *1*
by (*metis capped-omega-slide*)
also have $\ldots \leq s \;;\; (s \;;\; y)^{\omega}{}_{v}$ **using** *1 2 4*
by (*smt less-eq-def mult-right-isotone capped-omega-sub-dist order-trans*)
also have $\ldots = s \;;\; (s \;;\; y)^{\omega}{}_{v} \frown v$ **using** *3*
by (*metis meet.less-eq-def meet-commutative*)
finally show $(s \;;\; y)^{\omega}{}_{v} \leq s^{\omega}{}_{v}$ **using** *1*
by (*metis capped-omega-induct-meet*)
qed
qed

— AACP Theorem 6.16

lemma *left-plus-capped-omega*: $y \leq u \wedge u \;;\; u \leq u \wedge u \;;\; v \leq v \longrightarrow (y \;;\; y^{\star})^{\omega}{}_{v} = y^{\omega}{}_{v}$
proof
assume *1*: $y \leq u \wedge u \;;\; u \leq u \wedge u \;;\; v \leq v$
hence *2*: $y \;;\; y^{\star} \leq u$
by (*metis star-plus star-below-cap*)
hence $y \;;\; y^{\star} \;;\; (y \;;\; y^{\star})^{\omega}{}_{v} \leq v$ **using** *1*
by (*metis capped-omega-below meet.order-trans mult-isotone*)
hence $y \;;\; y^{\star} \;;\; (y \;;\; y^{\star})^{\omega}{}_{v} = (y \;;\; y^{\star})^{\omega}{}_{v}$
by (*metis meet.less-eq-def meet-commutative capped-omega-unfold*)
hence $(y \;;\; y^{\star})^{\omega}{}_{v} \leq y^{\omega}{}_{v}$ **using** *1 2*
by (*smt2 capped-omega-simulation mult-associative mult-semi-associative star.circ-transitive-equal star-simulation-right-equal*)
thus $(y \;;\; y^{\star})^{\omega}{}_{v} = y^{\omega}{}_{v}$ **using** *1 2*
by (*metis antisym capped-omega-isotone star.circ-mult-increasing*)
qed

— AACP Theorem 6.17

lemma *capped-omega-sub-vector*: $z \leq v \wedge y \leq u \wedge u \;;\; v \leq v \longrightarrow y^{\omega}{}_{u} \;;\; z \leq y^{\omega}{}_{v}$
proof
assume *1*: $z \leq v \wedge y \leq u \wedge u \;;\; v \leq v$
have $y^{\omega}{}_{u} \;;\; z \leq y \;;\; y^{\omega}{}_{u} \;;\; z \frown u \;;\; z$
by (*metis capped-omega-unfold meet.add-least-upper-bound meet.add-right-upper-bound meet-commutative mult-left-isotone*)
also have $\ldots \leq y \;;\; y^{\omega}{}_{u} \;;\; z \frown v$ **using** *1*
by (*metis meet.add-left-isotone meet-commutative mult-right-isotone order-trans*)
finally show $y^{\omega}{}_{u} \;;\; z \leq y^{\omega}{}_{v}$ **using** *1*
by (*metis capped-omega-induct-meet mult-associative*)
qed

— AACP Theorem 6.18

lemma *capped-omega-omega*: $y \leq u \wedge u \;;\; v \leq v \longrightarrow (y^{\omega}{}_{u})^{\omega}{}_{v} \leq y^{\omega}{}_{v}$
by (*metis capped-omega-below capped-omega-sub-vector capped-omega-unfold meet.add-left-upper-bound order-trans*)

end

end