Infinite executions of lazy and strict computations

Walter Guttmann, University of Canterbury

2013.05.05

theory *IELSC*

imports *Main*

begin

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

context *ord*

begin

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

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

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

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

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

end

context *order*

begin

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

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

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

lemma *ascending-chain-chain*: $ascending\text{-}chain\ f \rightarrow chain\ (range\ f)$
  unfolding *chain-def*
  apply *simp*
  apply (*smt ascending-chain-comparable*)
  done

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

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

lemma *descending-chain-k*: *descending-chain f* $\rightarrow$ *f* (*n* + *k*) $\leq$ *f n*
apply (*induct k*)
apply *simp*
apply (*metis add-Suc-right descending-chain-def order-trans*)
done

lemma *descending-chain-antitone*: *descending-chain f* $\wedge$ *m* $\leq$ *n* $\rightarrow$ *f n* $\leq$ *f m*
by (*metis descending-chain-k le-iff-add*)

lemma *descending-chain-comparable*: *descending-chain f* $\rightarrow$ *f n* $\leq$ *f m* $\vee$ *f m* $\leq$ *f n*
by (*metis nat-le-linear descending-chain-antitone*)

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

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

lemma *descending-chain-codirected*: *descending-chain f* $\rightarrow$ *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* $\neq$ {}
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* $\neq$ {} $\rightarrow$ *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 SUP-def SUP-image*)
done

lemma *inf-Inf*: **assumes** *nonempty*: *A* $\neq$ {}
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*: $Y \neq \{\} \rightarrow inf\ x\ (INF\ y{:}Y\ .\ f\ y) = (INF\ y{:}Y.\ inf\ x\ (f\ y))$
unfolding *INF-def*
apply *rule*
apply (*subst inf-Inf*)
apply (*smt empty-is-image*)
apply (*metis INF-def INF-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*

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

begin

definition *one-fun* :: $'a \Rightarrow 'a$
where *one-fun-def*: *one-fun* $\equiv$ *id*

definition *times-fun* :: $('a \Rightarrow 'a) \Rightarrow ('a \Rightarrow 'a) \Rightarrow ('a \Rightarrow 'a)$
where *times-fun-def*: *times-fun* $\equiv$ *comp*

instance
by *intro-classes*

end

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

lemma *power-zero-id*: $f\hat{}0 = id$
by (*metis one-fun-def power-0*)

lemma *power-succ-unfold*: $f\hat{}Suc\ n = f \circ f\hat{}n$
by (*metis power-Suc times-fun-def*)

lemma *power-succ-unfold-ext*: $(f\hat{}Suc\ n)\ x = f\ ((f\hat{}n)\ x)$
by (*metis o-apply power-succ-unfold*)

context *order*

begin

definition *isotone* :: $('a \Rightarrow 'a) \Rightarrow bool$
where $isotone\ f \leftrightarrow (\forall x\ y\ .\ x \leq y \rightarrow f(x) \leq f(y))$

definition *is-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a \Rightarrow bool$ **where** $is\text{-}fixpoint\ f\ x \leftrightarrow f(x) = x$
definition *is-prefixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a \Rightarrow bool$ **where** $is\text{-}prefixpoint\ f\ x \leftrightarrow f(x) \leq x$
definition *is-postfixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a \Rightarrow bool$ **where** $is\text{-}postfixpoint\ f\ x \leftrightarrow f(x) \geq x$
definition *is-least-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a \Rightarrow bool$ **where** $is\text{-}least\text{-}fixpoint\ f\ x \leftrightarrow f(x) = x \wedge (\forall y\ .\ f(y) = y \rightarrow x \leq y)$
definition *is-greatest-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a \Rightarrow bool$ **where** $is\text{-}greatest\text{-}fixpoint\ f\ x \leftrightarrow f(x) = x \wedge (\forall y\ .\ f(y) = y \rightarrow x \geq y)$
definition *is-least-prefixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a \Rightarrow bool$ **where** $is\text{-}least\text{-}prefixpoint\ f\ x \leftrightarrow f(x) \leq x \wedge (\forall y\ .\ f(y) \leq y \rightarrow x \leq y)$
definition *is-greatest-postfixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a \Rightarrow bool$ **where** $is\text{-}greatest\text{-}postfixpoint\ f\ x \leftrightarrow f(x) \geq x \wedge (\forall y\ .\ f(y) \geq y \rightarrow x \geq y)$
definition *has-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow bool$ **where** $has\text{-}fixpoint\ f \leftrightarrow (\exists x\ .\ is\text{-}fixpoint\ f\ x)$
definition *has-prefixpoint* :: $('a \Rightarrow 'a) \Rightarrow bool$ **where** $has\text{-}prefixpoint\ f \leftrightarrow (\exists x\ .\ is\text{-}prefixpoint\ f\ x)$
definition *has-postfixpoint* :: $('a \Rightarrow 'a) \Rightarrow bool$ **where** $has\text{-}postfixpoint\ f \leftrightarrow (\exists x\ .\ is\text{-}postfixpoint\ f\ x)$
definition *has-least-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow bool$ **where** $has\text{-}least\text{-}fixpoint\ f \leftrightarrow (\exists x\ .\ is\text{-}least\text{-}fixpoint\ f\ x)$
definition *has-greatest-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow bool$ **where** $has\text{-}greatest\text{-}fixpoint\ f \leftrightarrow (\exists x\ .\ is\text{-}greatest\text{-}fixpoint\ f\ x)$
definition *has-least-prefixpoint* :: $('a \Rightarrow 'a) \Rightarrow bool$ **where** $has\text{-}least\text{-}prefixpoint\ f \leftrightarrow (\exists x\ .\ is\text{-}least\text{-}prefixpoint\ f\ x)$
definition *has-greatest-postfixpoint* :: $('a \Rightarrow 'a) \Rightarrow bool$ **where** $has\text{-}greatest\text{-}postfixpoint\ f \leftrightarrow (\exists x\ .\ is\text{-}greatest\text{-}postfixpoint\ f\ x)$
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*: $has\text{-}least\text{-}fixpoint\ f \rightarrow (\exists!x\ .\ is\text{-}least\text{-}fixpoint\ f\ x)$
by (*smt antisym has-least-fixpoint-def is-least-fixpoint-def*)

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

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

lemma *greatest-postfixpoint-unique*: $has\text{-}greatest\text{-}postfixpoint\ f \rightarrow (\exists!x\ .\ is\text{-}greatest\text{-}postfixpoint\ f\ x)$

by (*smt antisym has-greatest-postfixpoint-def is-greatest-postfixpoint-def*)

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

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

lemma *least-prefixpoint*: *has-least-prefixpoint f* $\rightarrow$ *is-least-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* ($p\mu$ *f*)
by (*simp add*: *is-least-prefixpoint-def the-least-prefixpoint-def*)
qed

lemma *greatest-postfixpoint*: *has-greatest-postfixpoint f* $\rightarrow$ *is-greatest-postfixpoint f* ($p\nu$ *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* ($p\nu$ *f*)
by (*simp add*: *is-greatest-postfixpoint-def the-greatest-postfixpoint-def*)
qed

lemma *least-fixpoint-same*: *is-least-fixpoint f x* $\rightarrow$ $x = \mu$ *f*
by (*metis least-fixpoint least-fixpoint-unique has-least-fixpoint-def*)

lemma *greatest-fixpoint-same*: *is-greatest-fixpoint f x* $\rightarrow$ $x = \nu$ *f*
by (*metis greatest-fixpoint greatest-fixpoint-unique has-greatest-fixpoint-def*)

lemma *least-prefixpoint-same*: *is-least-prefixpoint f x* $\rightarrow$ $x = p\mu$ *f*
by (*metis least-prefixpoint least-prefixpoint-unique has-least-prefixpoint-def*)

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

lemma *least-fixpoint-char*: *is-least-fixpoint f x* $\leftrightarrow$ *has-least-fixpoint f* $\wedge$ *x* = *μ f*
by (*metis least-fixpoint-same has-least-fixpoint-def*)

lemma *least-prefixpoint-char*: *is-least-prefixpoint f x* $\leftrightarrow$ *has-least-prefixpoint f* $\wedge$ *x* = *pμ f*
by (*metis least-prefixpoint-same has-least-prefixpoint-def*)

lemma *greatest-fixpoint-char*: *is-greatest-fixpoint f x* $\leftrightarrow$ *has-greatest-fixpoint f* $\wedge$ *x* = *ν f*
by (*metis greatest-fixpoint-same has-greatest-fixpoint-def*)

lemma *greatest-postfixpoint-char*: *is-greatest-postfixpoint f x* $\leftrightarrow$ *has-greatest-postfixpoint f* $\wedge$ *x* = *pν f*
by (*metis greatest-postfixpoint-same has-greatest-postfixpoint-def*)

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

lemma *pmu-unfold*: *has-least-prefixpoint f* $\rightarrow$ *f* (*pμ f*) $\leq$ *pμ f*
by (*metis is-least-prefixpoint-def least-prefixpoint*)

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

lemma *pnu-unfold*: *has-greatest-postfixpoint f* $\rightarrow$ *pν f* $\leq$ *f* (*pν f*)
by (*metis is-greatest-postfixpoint-def greatest-postfixpoint*)

lemma *least-prefixpoint-fixpoint*: *has-least-prefixpoint f* $\wedge$ *isotone f* $\rightarrow$ *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* $\wedge$ *isotone f* $\rightarrow$ *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* $\wedge$ *isotone f* $\rightarrow$ *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* $\wedge$ *isotone f* $\rightarrow$ *pν f* = *ν f*
by (*smt has-greatest-fixpoint-def is-greatest-fixpoint-def greatest-fixpoint-unique greatest-postfixpoint-fixpoint greatest-fixpoint*)

definition *lifted-less-eq* :: (*'a* $\Rightarrow$ *'a*) $\Rightarrow$ (*'a* $\Rightarrow$ *'a*) $\Rightarrow$ *bool* ((- $\leq\leq$ -) [*51*, *51*] *50*)
where *f* $\leq\leq$ *g* $\leftrightarrow$ ($\forall$ *x* . *f*(*x*) $\leq$ *g*(*x*))

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

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

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

lemma *pmu-isotone*: *has-least-prefixpoint f* $\wedge$ *has-least-prefixpoint g* $\wedge$ $f \leq\leq g \rightarrow p\mu\ f \leq p\mu\ g$
by (*smt is-least-prefixpoint-def least-prefixpoint lifted-less-eq-def order-trans*)

lemma *mu-isotone*: *has-least-prefixpoint f* $\wedge$ *has-least-prefixpoint g* $\wedge$ *isotone f* $\wedge$ *isotone g* $\wedge$ $f \leq\leq g \rightarrow \mu\ f \leq \mu\ g$
by (*metis pmu-isotone pmu-mu*)

lemma *pnu-isotone*: *has-greatest-postfixpoint f* $\wedge$ *has-greatest-postfixpoint g* $\wedge$ $f \leq\leq g \rightarrow p\nu\ f \leq p\nu\ g$
by (*smt is-greatest-postfixpoint-def lifted-less-eq-def order-trans greatest-postfixpoint*)

lemma *nu-isotone*: *has-greatest-postfixpoint f* $\wedge$ *has-greatest-postfixpoint g* $\wedge$ *isotone f* $\wedge$ *isotone g* $\wedge$ $f \leq\leq g \rightarrow \nu\ f \leq \nu\ g$
by (*metis pnu-isotone pnu-nu*)

lemma *mu-square*: *isotone f* $\wedge$ *has-least-fixpoint f* $\wedge$ *has-least-fixpoint* $(f \circ f) \rightarrow \mu\ f = \mu\ (f \circ f)$
by (*metis antisym is-least-fixpoint-def isotone-def least-fixpoint-char least-fixpoint-unique o-apply*)

lemma *nu-square*: *isotone f* $\wedge$ *has-greatest-fixpoint f* $\wedge$ *has-greatest-fixpoint* $(f \circ f) \rightarrow \nu\ f = \nu\ (f \circ f)$
by (*metis antisym is-greatest-fixpoint-def isotone-def greatest-fixpoint-char greatest-fixpoint-unique o-apply*)

lemma *mu-roll*: *isotone g* $\wedge$ *has-least-fixpoint* $(f \circ g)$ $\wedge$ *has-least-fixpoint* $(g \circ f) \rightarrow \mu\ (g \circ f) = g(\mu\ (f \circ g))$
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* $\wedge$ *has-greatest-fixpoint* $(f \circ g)$ $\wedge$ *has-greatest-fixpoint* $(g \circ f) \rightarrow \nu\ (g \circ f) = g(\nu\ (f \circ g))$
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* $\wedge$ *has-greatest-fixpoint f* $\rightarrow \mu\ f \leq \nu\ f$
by (*metis is-greatest-fixpoint-def is-least-fixpoint-def least-fixpoint greatest-fixpoint*)

lemma *pmu-below-pnu-fix*: *has-fixpoint f* $\wedge$ *has-least-prefixpoint f* $\wedge$ *has-greatest-postfixpoint f* $\rightarrow p\mu\ f \leq p\nu\ f$
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* $\wedge$ *has-least-prefixpoint f* $\wedge$ *has-greatest-postfixpoint f* $\rightarrow p\mu\ f \leq p\nu\ f$
by (*metis has-fixpoint-def is-fixpoint-def is-least-fixpoint-def least-prefixpoint-fixpoint pmu-below-pnu-fix*)

definition *galois* :: $('a \Rightarrow 'a) \Rightarrow ('a \Rightarrow 'a) \Rightarrow bool$
where *galois l u* $\leftrightarrow (\forall\, x\ y\ .\ l(x) \leq y \leftrightarrow x \leq u(y))$

lemma *galois-char*: *galois l u* $\leftrightarrow$ $(\forall x\ .\ x \leq u(l(x))) \wedge (\forall x\ .\ l(u(x)) \leq x) \wedge$ *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* $\rightarrow$ $l(x) = l(u(l(x))) \wedge u(x) = u(l(u(x)))$
by (*smt antisym galois-char isotone-def*)

lemma *mu-fusion-1*: *galois l u* $\wedge$ *isotone h* $\wedge$ *has-least-prefixpoint g* $\wedge$ *has-least-fixpoint h* $\wedge$ $l(g(u(\mu\ h))) \leq h(l(u(\mu\ h))) \rightarrow l(p\mu\ g) \leq \mu\ h$
proof
assume *1*: *galois l u* $\wedge$ *isotone h* $\wedge$ *has-least-prefixpoint g* $\wedge$ *has-least-fixpoint h* $\wedge$ $l(g(u(\mu\ h))) \leq h(l(u(\mu\ h)))$
hence $l(g(u(\mu\ h))) \leq \mu\ h$
by (*metis galois-char least-fixpoint-same least-fixpoint-unique is-least-fixpoint-def isotone-def order-trans*)
thus $l(p\mu\ g) \leq \mu\ h$ **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* $\wedge$ *isotone h* $\wedge$ *has-least-prefixpoint g* $\wedge$ *has-least-fixpoint h* $\wedge$ $l \circ g \leq\leq h \circ l \rightarrow l(p\mu\ g) \leq \mu\ h$
by (*metis lifted-less-eq-def mu-fusion-1 o-apply*)

lemma *mu-fusion-equal-1*: *galois l u* $\wedge$ *isotone g* $\wedge$ *isotone h* $\wedge$ *has-least-prefixpoint g* $\wedge$ *has-least-fixpoint h* $\wedge$ $l(g(u(\mu\ h))) \leq h(l(u(\mu\ h))) \wedge l(g(p\mu\ g)) = h(l(p\mu\ g)) \rightarrow \mu\ h = l(p\mu\ g) \wedge \mu\ h = l(\mu\ g)$
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* $\wedge$ *isotone h* $\wedge$ *has-least-prefixpoint g* $\wedge$ *has-least-prefixpoint h* $\wedge$ $l(g(u(\mu\ h))) \leq h(l(u(\mu\ h))) \wedge l(g(p\mu\ g)) = h(l(p\mu\ g)) \rightarrow p\mu\ h = l(p\mu\ g) \wedge \mu\ h = l(p\mu\ g)$
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* $\wedge$ *isotone g* $\wedge$ *isotone h* $\wedge$ *has-least-prefixpoint g* $\wedge$ *has-least-fixpoint h* $\wedge$ $l \circ g = h \circ l \rightarrow \mu\ h = l(p\mu\ g) \wedge \mu\ h = l(\mu\ g)$
by (*metis mu-fusion-equal-1 o-apply order-refl*)

lemma *mu-fusion-equal-4*: *galois l u* $\wedge$ *isotone h* $\wedge$ *has-least-prefixpoint g* $\wedge$ *has-least-prefixpoint h* $\wedge$ $l \circ g = h \circ l \rightarrow p\mu\ h = l(p\mu\ g) \wedge \mu\ h = l(p\mu\ g)$
by (*metis mu-fusion-equal-2 o-apply order-refl*)

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

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

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

lemma *mu-exchange-1*: *galois l u* $\wedge$ *isotone g* $\wedge$ *isotone h* $\wedge$ *has-least-prefixpoint* $(l \circ h)$ $\wedge$ *has-least-prefixpoint* $(h \circ g)$ $\wedge$ *has-least-fixpoint* $(g \circ h)$ $\wedge$ $l \circ h \circ g \leq\leq g \circ h \circ l \rightarrow \mu(l \circ h) \leq \mu(g \circ h)$
proof
assume *1*: *galois l u* $\wedge$ *isotone g* $\wedge$ *isotone h* $\wedge$ *has-least-prefixpoint* $(l \circ h)$ $\wedge$ *has-least-prefixpoint* $(h \circ g)$ $\wedge$ *has-least-fixpoint* $(g \circ h)$ $\wedge$ $l \circ h \circ g \leq\leq g \circ h \circ l$
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* $\wedge$ *isotone g* $\wedge$ *isotone h* $\wedge$ *has-least-prefixpoint* $(l \circ h)$ $\wedge$ *has-least-prefixpoint* $(h \circ l)$ $\wedge$ *has-least-prefixpoint* $(h \circ g)$ $\wedge$ *has-least-fixpoint* $(g \circ h)$ $\wedge$ *has-least-fixpoint* $(h \circ g)$ $\wedge$ $l \circ h \circ g \leq\leq g \circ h \circ l \rightarrow \mu(h \circ l) \leq \mu(h \circ g)$
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* $\wedge$ *galois k t* $\wedge$ *isotone h* $\wedge$ *has-least-prefixpoint* $(l \circ h)$ $\wedge$ *has-least-prefixpoint* $(h \circ l)$ $\wedge$ *has-least-prefixpoint* $(k \circ h)$ $\wedge$ *has-least-prefixpoint* $(h \circ k)$ $\wedge$ $l \circ h \circ k = k \circ h \circ l \rightarrow \mu(l \circ h) = \mu(k \circ h) \wedge \mu(h \circ l) = \mu(h \circ k)$
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* $\wedge$ *isotone g* $\wedge$ *isotone h* $\wedge$ *has-greatest-postfixpoint* $(u \circ h)$ $\wedge$ *has-greatest-postfixpoint* $(h \circ g)$ $\wedge$ *has-greatest-fixpoint* $(g \circ h)$ $\wedge$ $g \circ h \circ u \leq\leq u \circ h \circ g \rightarrow \nu(g \circ h) \leq \nu(u \circ h)$
proof
assume *1*: *galois l u* $\wedge$ *isotone g* $\wedge$ *isotone h* $\wedge$ *has-greatest-postfixpoint* $(u \circ h)$ $\wedge$ *has-greatest-postfixpoint* $(h \circ g)$ $\wedge$ *has-greatest-fixpoint* $(g \circ h)$ $\wedge$ $g \circ h \circ u \leq\leq u \circ h \circ g$
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* $\wedge$ *isotone g* $\wedge$ *isotone h* $\wedge$ *has-greatest-postfixpoint* $(u \circ h)$ $\wedge$ *has-greatest-postfixpoint* $(h \circ u)$ $\wedge$ *has-greatest-postfixpoint* $(h \circ g)$ $\wedge$ *has-greatest-fixpoint* $(g \circ h)$ $\wedge$ *has-greatest-fixpoint* $(h \circ g)$ $\wedge$ $g \circ h \circ u \leq\leq u \circ h \circ g \rightarrow \nu(h \circ g) \leq \nu(h \circ u)$
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* $(\lambda x\ .\ \nu(\lambda y\ .\ f\ x\ y))$
by (*smt isotone-def lifted-less-eq-def nu-isotone*)
thus $\nu(\lambda x\ .\ f\ x\ x) = \nu(\lambda x\ .\ \nu(\lambda y\ .\ f\ x\ y))$ **using** *1*
by (*smt greatest-fixpoint-same greatest-postfixpoint greatest-postfixpoint-fixpoint is-greatest-fixpoint-def is-greatest-postfixpoint-def*)
qed

end

class *il-semiring* = *mult* + *one* + *ord* + *plus* + *zero* +
assumes *add-associative* : $(x + y) + z = x + (y + z)$
assumes *add-commutative* : $x + y = y + x$
assumes *add-idempotent* : $x + x = x$
assumes *add-left-zero* : $0 + x = x$
assumes *mult-associative* : $(x\ ;\ y)\ ;\ z = x\ ;\ (y\ ;\ z)$
assumes *mult-left-one* : $1\ ;\ x = x$
assumes *mult-right-one* : $x\ ;\ 1 = x$
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 *less-eq-def* : $x \leq y \leftrightarrow x + y = y$
assumes *less-def* : $x < y \leftrightarrow x \leq y \wedge \neg\ (y \leq x)$

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

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

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

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

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

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*)

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

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

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

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

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

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

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

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

lemma *test-preserves-equation*: $p \leq p \ ; \ p \wedge p \leq 1 \rightarrow (p \ ; \ x \leq x \ ; \ p \leftrightarrow p \ ; \ x = p \ ; \ x \ ; \ p)$
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

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

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

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

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

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

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

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

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

end

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

class *il-semiring-T* = *il-semiring* + T +
assumes *add-left-top*: $T + x = T$

begin

lemma *add-right-top*: $x + T = T$
  by (*metis add-commutative add-left-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 mult-right-isotone mult-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* $x \leftrightarrow x = x \ ; \ T$

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* $y \rightarrow$ *vector* $(x + y)$
  by (*metis mult-right-dist-add vector-def*)

lemma *vector-left-mult-closed*: *vector* $y \rightarrow$ *vector* $(x \ ; \ y)$
  by (*metis mult-associative vector-def*)

end

class *idl-semiring* = *il-semiring* +
  assumes *mult-left-sup-dist-add*: $x \ ; \ (y + z) \leq x \ ; \ y + x \ ; \ z$

begin

lemma *mult-left-dist-add*: $x \ ; \ y + x \ ; \ z = x \ ; \ (y + z)$
  by (*metis eq-iff mult-left-sub-dist-add mult-left-sup-dist-add*)

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

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

end

class *idl-semiring-T* = *il-semiring-T* + *idl-semiring*

class *circ* =
  fixes *circ* :: *'a* ⇒ *'a* (-° [*100*] *100*)

class *il-conway-semiring* = *il-semiring* + *circ* +
  assumes *circ-left-unfold*: *1* + *x* ; *x*° = *x*°
  assumes *circ-left-slide*: (*x* ; *y*)° ; *x* ≤ *x* ; (*y* ; *x*)°
  assumes *circ-add-1*: (*x* + *y*)° = *x*° ; (*y* ; *x*°)°

begin

lemma *circ-mult-sub*: *1* + *x* ; (*y* ; *x*)° ; *y* ≤ (*x* ; *y*)°
  by (*metis add-right-isotone circ-left-slide circ-left-unfold mult-associative mult-right-isotone*)

lemma *circ-right-unfold-sub*: *1* + *x*° ; *x* ≤ *x*°
  by (*metis circ-mult-sub mult-left-one mult-right-one*)

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

lemma *circ-increasing*: *x* ≤ *x*°
  by (*metis add-right-upper-bound circ-left-unfold circ-right-unfold-sub mult-left-one mult-right-sub-dist-add-left order-trans*)

lemma *circ-reflexive*: *1* ≤ *x*°
  by (*metis add-left-divisibility circ-left-unfold*)

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

lemma *circ-circ-circ*: *x*°°° = *x*°°
  by (*metis add-idempotent circ-add-1 circ-increasing circ-transitive-equal less-eq-def*)

lemma *circ-one*: *1*° = *1*°°
  by (*metis circ-circ-circ circ-zero*)

lemma *circ-add-sub*: (*x*° ; *y*)° ; *x*° ≤ (*x* + *y*)°
  by (*metis circ-add-1 circ-left-slide*)

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

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

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

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

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

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

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

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

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

lemma *circ-mult-upper-bound*: $x \leq z^\circ \wedge y \leq z^\circ \rightarrow x \mathbin{;} y \leq z^\circ$
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*: $x \mathbin{;} y \leq (x + y)^\circ$
by (*metis add-commutative circ-mult-upper-bound circ-sub-dist-1*)

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

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

lemma *circ-add-2*: $(x + y)^\circ \leq (x^\circ \mathbin{;} y^\circ)^\circ$
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*: $1 \leq x \rightarrow x \mathbin{;} x^\circ = x^\circ$
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*: $1 \leq x \rightarrow x^\circ \ ; \ x = x^\circ$
by (*metis antisym less-eq-def mult-left-sub-dist-add-left mult-right-one right-plus-below-circ*)

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*)

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*)

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

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*)

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*)

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*)

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*)

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*)

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

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*)

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*)

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*)

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*)

end

class *il-conway-semiring-T* = *il-semiring-T* + *il-conway-semiring*

begin

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 antisym circ-plus-one mult-left-sub-dist-add-left mult-right-one top-greatest*)

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

end

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

begin

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

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

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

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

lemma *circ-plus-same*: $x^\circ \ ; x = x \ ; x^\circ$
by (*metis circ-slide 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 circ-sub-dist-3*)

end

class *sidl-conway-semiring* = *idl-conway-semiring* + *sil-conway-semiring*

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*)

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

end

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

begin

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*: $p \leq p \ ; p \wedge p \leq 1 \wedge p \ ; x \leq x \ ; p \rightarrow p \ ; x^\circ = p \ ; (p \ ; x)^\circ$
apply *rule*
apply (*rule antisym*)
apply (*smt antisym circ-simulate circ-slide 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* = *itering-1* +
  assumes *circ-simulate-right*: $z\ ;\ x \leq y\ ;\ z + w \rightarrow z\ ;\ x^\circ \leq y^\circ\ ;\ (z + w\ ;\ x^\circ)$
  assumes *circ-simulate-left*: $x\ ;\ z \leq z\ ;\ y + w \rightarrow x^\circ\ ;\ z \leq (z + x^\circ\ ;\ w)\ ;\ y^\circ$

begin

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

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

lemma *circ-separate-1*: $y\ ;\ x \leq x\ ;\ y \rightarrow (x + y)^\circ = x^\circ\ ;\ y^\circ$
proof −
  have $y\ ;\ x \leq x\ ;\ y \rightarrow y^\circ\ ;\ x\ ;\ y^\circ \leq x\ ;\ y^\circ + y^\circ\ ;\ 0$
    using [[*smt-timeout=120*]] **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

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* + *sidl-conway-semiring*

begin

lemma *circ-simulate-1*: $y\ ;\ x \leq x\ ;\ y \rightarrow y^\circ\ ;\ x^\circ \leq x^\circ\ ;\ y^\circ$
  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*)

lemma *atomicity-refinement*: $s = s\ ;\ q \wedge x = q\ ;\ x \wedge q\ ;\ b = 0 \wedge r\ ;\ b \leq b\ ;\ r \wedge r\ ;\ l \leq l\ ;\ r \wedge x\ ;\ l \leq l\ ;\ x \wedge b\ ;\ l \leq l\ ;\ b \wedge q\ ;\ l \leq l\ ;\ q \wedge r^\circ\ ;\ q \leq q\ ;\ r^\circ \wedge q \leq 1 \rightarrow s\ ;\ (x + b + r + l)^\circ\ ;\ q \leq s\ ;\ (x\ ;\ b^\circ\ ;\ q + r + l)^\circ$
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 $s\ ;\ (x + b + r + l)^\circ\ ;\ q = s\ ;\ l^\circ\ ;\ (x + b + r)^\circ\ ;\ q$
    using [[*smt-timeout = 120*]] **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 $\ldots = s\ ;\ l^\circ\ ;\ b^\circ\ ;\ r^\circ\ ;\ q\ ;\ (x\ ;\ b^\circ\ ;\ r^\circ\ ;\ q)^\circ$ **using** *1*
    by (*smt add-associative add-commutative circ-add-1 circ-separate-1 circ-slide mult-associative*)
  also have $\ldots \leq s\ ;\ l^\circ\ ;\ b^\circ\ ;\ r^\circ\ ;\ q\ ;\ (x\ ;\ b^\circ\ ;\ q\ ;\ r^\circ)^\circ$ **using** *1*
    by (*metis circ-isotone mult-associative mult-right-isotone*)
  also have $\ldots \leq s\ ;\ q\ ;\ l^\circ\ ;\ b^\circ\ ;\ r^\circ\ ;\ (x\ ;\ b^\circ\ ;\ q\ ;\ r^\circ)^\circ$ **using** *1*
    by (*metis mult-left-isotone mult-right-isotone mult-right-one*)
  also have $\ldots \leq s\ ;\ l^\circ\ ;\ q\ ;\ b^\circ\ ;\ r^\circ\ ;\ (x\ ;\ b^\circ\ ;\ q\ ;\ r^\circ)^\circ$ **using** *1*
    by (*metis circ-simulate mult-associative mult-left-isotone mult-right-isotone*)
  also have $\ldots \leq s\ ;\ l^\circ\ ;\ r^\circ\ ;\ (x\ ;\ b^\circ\ ;\ q\ ;\ r^\circ)^\circ$ **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-4* = *itering-3* +
assumes *circ-simulate-right-plus*: $z\ ;\ x \leq y\ ;\ y^\circ\ ;\ z + w \rightarrow z\ ;\ x^\circ \leq y^\circ\ ;\ (z + w\ ;\ x^\circ)$
assumes *circ-simulate-left-plus*: $x\ ;\ z \leq z\ ;\ y^\circ + w \rightarrow x^\circ\ ;\ z \leq (z + x^\circ\ ;\ w)\ ;\ y^\circ$

begin

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

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

lemma *circ-simulate-2*: $y\ ;\ x^\circ \leq x^\circ\ ;\ y^\circ \leftrightarrow y^\circ\ ;\ x^\circ \leq x^\circ\ ;\ y^\circ$
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*: $y\ ;\ x \leq x \rightarrow y^\circ\ ;\ x \leq x + y^\circ\ ;\ 0$
by (*metis circ-simulate-left-plus-1 circ-zero mult-right-one*)

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

lemma *circ-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-simulate-right-plus-1 mult-left-one mult-right-isotone mult-right-one*)

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

lemma *circ-extra-circ*: $(y\ ;\ x^\circ)^\circ = (y\ ;\ y^\circ\ ;\ x^\circ)^\circ$
proof −
have $y\ ;\ y^\circ\ ;\ x^\circ \leq y\ ;\ x^\circ\ ;\ (y\ ;\ x^\circ)^\circ$
by (*metis add-commutative circ-add-1 circ-sub-dist-3 mult-associative mult-right-isotone*)
hence $(y\ ;\ y^\circ\ ;\ x^\circ)^\circ \leq (y\ ;\ x^\circ)^\circ$
by (*metis circ-simulate-right-plus-1 mult-left-one mult-right-one*)
thus *?thesis*
by (*metis antisym circ-back-loop-fixpoint circ-isotone mult-right-sub-dist-add-right*)
qed

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*)

lemma *separation*: $y\ ;\ x \leq x\ ;\ y^\circ \rightarrow (x + y)^\circ = x^\circ\ ;\ y^\circ$
proof −
have $y\ ;\ x \leq x\ ;\ y^\circ \rightarrow y^\circ\ ;\ x\ ;\ y^\circ \leq x\ ;\ y^\circ + y^\circ\ ;\ 0$
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

lemma *circ-simulate-4*: $y\ ;\ x \leq x\ ;\ x^\circ\ ;\ (1 + y) \rightarrow y^\circ\ ;\ x^\circ \leq x^\circ\ ;\ y^\circ$
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*: $y\ ;\ x \leq x\ ;\ x^\circ\ ;\ (x + y) \rightarrow y^\circ\ ;\ x^\circ \leq x^\circ\ ;\ y^\circ$
by (*metis circ-add-sub-add-one circ-simulate-4 order-trans*)

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

lemma *circ-separate-4*: $y\ ;\ x \leq x\ ;\ x^\circ\ ;\ (1 + y) \rightarrow (x + y)^\circ = x^\circ\ ;\ y^\circ$
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*: $y\ ;\ x \leq x\ ;\ x^\circ\ ;\ (x + y) \rightarrow (x + y)^\circ = x^\circ\ ;\ y^\circ$
by (*metis circ-add-sub-add-one circ-separate-4 order-trans*)

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

end

class *itering-T* = *idl-semiring-T* + *itering-4*

begin

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*)

end

context *il-semiring*

begin

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

end

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

class *lka* = *il-semiring* + *star* +
assumes *star-left-unfold* : $1 + y \ ; \ y^* \leq y^*$
assumes *star-left-induct* : $z + y \ ; \ x \leq x \rightarrow y^* \ ; \ z \leq x$

begin

lemma *star-left-unfold-equal*: $1 + x \ ; \ x^* = x^*$
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*: $x \leq y \rightarrow x^* \leq y^*$
by (*metis add-right-isotone mult-left-isotone order-trans star-left-unfold mult-right-one star-left-induct*)

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

lemma *star-add-1*: $(x + y)^* = x^* \,;\, (y \,;\, x^*)^*$
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

sublocale *lka* < *star*!: *il-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 *lka*

begin

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

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

lemma *star-left-induct-mult*: $x \,;\, y \leq y \rightarrow x^* \,;\, y \leq y$

by (*metis add-commutative less-eq-def order-refl star-left-induct*)

lemma *star-left-induct-mult-iff*: $x\ ;\ y \leq y \leftrightarrow x^*\ ;\ y \leq y$
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*: $x^* = x^{**}$
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*: $(1 + x)^* = x^*$
by (*metis star.circ-circ-add star-involutive*)

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

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

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

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

lemma *star-rtc-3*: $1 + x + y\ ;\ y = y \rightarrow x^* \leq y$
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*: $(x + y)^* = (x^*\ ;\ y^*)^*$
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*: $(x + y)^* = (x^* + y^*)^*$
by (*metis star-decompose-1 star-involutive*)

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

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

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

lemma *star-loop-mu*: $\mu\ (\lambda x\ .\ y\ ;\ x + z) = y^*\ ;\ z$
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*)

end

class *slka* = *lka* +
assumes *star-right-induct*: $z + x\ ;\ y \leq x \rightarrow z\ ;\ y^* \leq x$

begin

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

lemma *star-plus*: $y^*\ ;\ y = y\ ;\ y^*$
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*: $(x\ ;\ y)^*\ ;\ x = x\ ;\ (y\ ;\ x)^*$
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*: $z\ ;\ x \leq y\ ;\ z \rightarrow z\ ;\ x^* \leq y^*\ ;\ z$
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 *slka* < *star*!: *itering-1* **where** *circ* = *star*
apply *unfold-locales*
apply (*metis star-slide order-refl*)
apply (*metis star-simulation-right*)
done

context *slka*

begin

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

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

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

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

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

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

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

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

lemma *star-square*: $x^* = (1 + x) \mathbin{;} (x \mathbin{;} x)^*$
proof −
let *?f* $= \lambda y \,.\, y \mathbin{;} x + 1$
have *1*: *isotone ?f*
by (*smt add-left-isotone isotone-def mult-left-isotone*)
have *2*: *?f* $\circ$ *?f* $= (\lambda y \,.\, y \mathbin{;} (x \mathbin{;} x) + (1 + x))$
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*: $x^* = (x \mathbin{;} x)^* \mathbin{;} (x + 1)$
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*: $z \mathbin{;} x \leq y \mathbin{;} y^* \mathbin{;} z + w \rightarrow z \mathbin{;} x^* \leq y^* \mathbin{;} (z + w \mathbin{;} x^*)$
proof
assume *1*: $z \mathbin{;} x \leq y \mathbin{;} y^* \mathbin{;} z + w$
have $(z + w \mathbin{;} x^*) \mathbin{;} x \leq z \mathbin{;} x + w \mathbin{;} x^*$
by (*metis add-right-isotone mult-associative mult-right-dist-add mult-right-isotone star.circ-increasing star.circ-transitive-equal*)
also have $\ldots \leq y \mathbin{;} y^* \mathbin{;} z + w + w \mathbin{;} x^*$ **using** *1*
by (*metis add-left-isotone*)
also have $\ldots \leq y \mathbin{;} y^* \mathbin{;} z + w \mathbin{;} x^*$
by (*metis add-least-upper-bound add-right-isotone add-right-upper-bound star.circ-back-loop-prefixpoint*)
also have $\ldots \leq y^* \mathbin{;} (z + w \mathbin{;} x^*)$
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 $y^* \mathbin{;} (z + w \mathbin{;} x^*) \mathbin{;} x \leq y^* \mathbin{;} (z + w \mathbin{;} x^*)$
by (*metis mult-associative mult-right-isotone star.circ-transitive-equal*)
thus $z \mathbin{;} x^* \leq y^* \mathbin{;} (z + w \mathbin{;} x^*)$
by (*metis add-least-upper-bound star-right-induct mult-left-sub-dist-add-left star.circ-loop-fixpoint*)
qed

end

class *sdlka* = *idl-semiring* + *slka*

begin

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

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

end

sublocale *sdlka* < *star*!: *itering-4* **where** *circ* = *star*
apply *unfold-locales*
apply (*metis add-commutative add-right-isotone mult-associative mult-left-sub-dist-add-left star.circ-loop-fixpoint star-circ-simulate-right-plus order-trans*)
apply (*metis add-commutative add-right-isotone mult-right-isotone order-trans star.circ-increasing star-circ-simulate-left-plus*)
apply (*rule star-circ-simulate-right-plus*)
apply (*rule star-circ-simulate-left-plus*)
done

class *lk-conway-semiring* = *lka* + *il-conway-semiring*

begin

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

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

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

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

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

lemma *circ-add-3*: $(x^\circ \ ; \ y^\circ)^* \leq (x + y)^\circ$
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 *sdlk-conway-semiring* = *sdlka* + *itering-4*

begin

subclass *lk-conway-semiring* **..**

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

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

lemma *circ-sub-decompose*: $(x^\circ \ ; \ y)^\circ \leq (x^* \ ; \ y)^\circ \ ; \ x^\circ$
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*: $(x + y)^\circ = (x^* \ ; \ y)^\circ \ ; \ x^\circ$
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*: $(x^\circ \ ; \ y)^\circ \ ; \ x^\circ = (x^* \ ; \ y)^\circ \ ; \ x^\circ$
by (*metis circ-add circ-add-4*)

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

end

class *lka-T* = *il-semiring-T* + *lka*

sublocale *lka-T* < *star*!: *il-conway-semiring-T* **where** *circ* = *star* **..**

class *sdlka-T* = *il-semiring-T* + *sdlka*

sublocale *sdlka-T* < *star*!: *itering-T* **where** *circ* = *star* **..**

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

class *loa* = *lka* + *omega* +
  assumes *omega-unfold*: $y^\omega = y\ ;\ y^\omega$
  assumes *omega-induct*: $x \leq z + y\ ;\ x \rightarrow x \leq y^\omega + y^*\ ;\ z$

begin

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

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

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

lemma *omega-induct-mult*: $y \leq x\ ;\ y \rightarrow y \leq x^\omega$
  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*: $x \leq y \rightarrow x^\omega \leq y^\omega$
  by (*metis less-eq-def omega-sub-dist*)

lemma *omega-induct-equal*: $y = z + x\ ;\ y \rightarrow y \leq x^\omega + x^*\ ;\ z$
  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*: $x^*\ ;\ x^\omega = x^\omega$
  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*: $z \mathbin{;} x \leq y \mathbin{;} z \to z \mathbin{;} x^\omega \leq y^\omega$
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*: $(x \mathbin{;} x^*)^\omega = x^\omega$
by (*metis antisym mult-associative omega-induct-mult omega-unfold order-refl star.left-plus-circ star-mult-omega*)

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

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

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

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

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

lemma *omega-star*: $(x^\omega)^* = 1 + x^\omega$
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*: $x^\omega \mathbin{;} x^{\omega *} = x^\omega$
by (*metis add-least-upper-bound antisym omega-sub-vector star.circ-back-loop-prefixpoint*)

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

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

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

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

lemma *omega-loop-fixpoint*: $y \mathbin{;} (y^\omega + y^* \mathbin{;} z) + z = y^\omega + y^* \mathbin{;} z$

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*: $y\ ;\ x + z = x \rightarrow x \leq y^\omega + y^*\ ;\ z$
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*: $(x + y\ ;\ 0)^\omega = x^\omega + x^*\ ;\ y\ ;\ 0$
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*: $x^\omega\ ;\ x^* = x^\omega$
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* $(\lambda x\ .\ y\ ;\ x + z)$ $(y^\omega + y^*\ ;\ z)$
by (*smt is-greatest-fixpoint-def omega-loop-fixpoint omega-loop-greatest-fixpoint*)

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

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

lemma *omega-import*: $p \leq p\ ;\ p \wedge p\ ;\ x \leq x\ ;\ p \rightarrow p\ ;\ x^\omega = p\ ;\ (p\ ;\ x)^\omega$
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 $\ldots \leq p\ ;\ x\ ;\ p\ ;\ x^\omega$ **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

end

sublocale *loa* < *comb0*!: *il-conway-semiring* **where** *circ* = $(\lambda x\ .\ x^*\ ;\ (x^\omega\ ;\ 0 + 1))$
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 *sdloa* = *sdlka* + *loa*

begin

lemma *star-omega-absorb*: $y^* \mathbin{;} (y^* \mathbin{;} x)^* \mathbin{;} y^\omega = (y^* \mathbin{;} x)^* \mathbin{;} y^\omega$
proof −
have $y^* \mathbin{;} (y^* \mathbin{;} x)^* \mathbin{;} y^\omega = y^* \mathbin{;} y^* \mathbin{;} x \mathbin{;} (y^* \mathbin{;} x)^* \mathbin{;} y^\omega + y^* \mathbin{;} y^\omega$
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*: $z \mathbin{;} x \leq y \mathbin{;} (y^\omega \mathbin{;} 0 + y^*) \mathbin{;} z + w \to z \mathbin{;} (x^\omega \mathbin{;} 0 + x^*) \leq (y^\omega \mathbin{;} 0 + y^*) \mathbin{;} (z + w \mathbin{;} (x^\omega \mathbin{;} 0 + x^*))$
proof
assume $z \mathbin{;} x \leq y \mathbin{;} (y^\omega \mathbin{;} 0 + y^*) \mathbin{;} z + w$
hence *1*: $z \mathbin{;} x \leq y^\omega \mathbin{;} 0 + y \mathbin{;} y^* \mathbin{;} z + w$
by (*metis mult-associative mult-left-dist-add mult-left-zero mult-right-dist-add omega-unfold*)
hence $(y^\omega \mathbin{;} 0 + y^* \mathbin{;} z + y^* \mathbin{;} w \mathbin{;} x^\omega \mathbin{;} 0 + y^* \mathbin{;} w \mathbin{;} x^*) \mathbin{;} x \leq y^\omega \mathbin{;} 0 + y^* \mathbin{;} (y^\omega \mathbin{;} 0 + y \mathbin{;} y^* \mathbin{;} z + w) + y^* \mathbin{;} w \mathbin{;} x^\omega \mathbin{;} 0 + y^* \mathbin{;} w \mathbin{;} x^*$
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 $\ldots = y^\omega \mathbin{;} 0 + y^* \mathbin{;} y \mathbin{;} y^* \mathbin{;} z + y^* \mathbin{;} w \mathbin{;} x^\omega \mathbin{;} 0 + y^* \mathbin{;} w \mathbin{;} x^*$
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 $\ldots \leq y^\omega \mathbin{;} 0 + y^* \mathbin{;} z + y^* \mathbin{;} w \mathbin{;} x^\omega \mathbin{;} 0 + y^* \mathbin{;} w \mathbin{;} x^*$
by (*smt add-commutative add-left-isotone mult-left-isotone star.circ-increasing star.circ-plus-same star.circ-transitive-equal*)
finally have $z + (y^\omega \mathbin{;} 0 + y^* \mathbin{;} z + y^* \mathbin{;} w \mathbin{;} x^\omega \mathbin{;} 0 + y^* \mathbin{;} w \mathbin{;} x^*) \mathbin{;} x \leq y^\omega \mathbin{;} 0 + y^* \mathbin{;} z + y^* \mathbin{;} w \mathbin{;} x^\omega \mathbin{;} 0 + y^* \mathbin{;} w \mathbin{;} x^*$
by (*smt add-least-upper-bound add-left-upper-bound star.circ-loop-fixpoint*)
hence *2*: $z \mathbin{;} x^* \leq y^\omega \mathbin{;} 0 + y^* \mathbin{;} z + y^* \mathbin{;} w \mathbin{;} x^\omega \mathbin{;} 0 + y^* \mathbin{;} w \mathbin{;} x^*$
by (*metis star-right-induct*)
have $z \mathbin{;} x^\omega \mathbin{;} 0 \leq (y^\omega \mathbin{;} 0 + y \mathbin{;} y^* \mathbin{;} z + w) \mathbin{;} x^\omega \mathbin{;} 0$ **using** *1*
by (*smt add-left-divisibility mult-associative mult-right-sub-dist-add-left omega-unfold*)
hence $z \mathbin{;} x^\omega \mathbin{;} 0 \leq y^\omega + y^* \mathbin{;} (y^\omega \mathbin{;} 0 + w \mathbin{;} x^\omega \mathbin{;} 0)$
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 $z \mathbin{;} (x^\omega \mathbin{;} 0 + x^*) \leq (y^\omega \mathbin{;} 0 + y^*) \mathbin{;} (z + w \mathbin{;} (x^\omega \mathbin{;} 0 + x^*))$ **using** *2*
using [[*smt-timeout=60*]] **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*: $x \mathbin{;} z \leq z \mathbin{;} (y^\omega \mathbin{;} 0 + y^*) + w \to (x^\omega \mathbin{;} 0 + x^*) \mathbin{;} z \leq (z + (x^\omega \mathbin{;} 0 + x^*) \mathbin{;} w) \mathbin{;} (y^\omega \mathbin{;} 0 + y^*)$
proof
assume *1*: $x \mathbin{;} z \leq z \mathbin{;} (y^\omega \mathbin{;} 0 + y^*) + w$
have $x \mathbin{;} (z \mathbin{;} y^\omega \mathbin{;} 0 + z \mathbin{;} y^* + x^\omega \mathbin{;} 0 + x^* \mathbin{;} w \mathbin{;} y^\omega \mathbin{;} 0 + x^* \mathbin{;} w \mathbin{;} y^*) = x \mathbin{;} z \mathbin{;} y^\omega \mathbin{;} 0 + x \mathbin{;} z \mathbin{;} y^* + x^\omega \mathbin{;} 0 + x \mathbin{;} x^* \mathbin{;} w \mathbin{;} y^\omega \mathbin{;} 0 + x \mathbin{;} x^* \mathbin{;} w \mathbin{;} y^*$
by (*smt mult-associative mult-left-dist-add omega-unfold*)
also have $\ldots \leq x \mathbin{;} z \mathbin{;} y^\omega \mathbin{;} 0 + x \mathbin{;} z \mathbin{;} y^* + x^\omega \mathbin{;} 0 + x^* \mathbin{;} w \mathbin{;} y^\omega \mathbin{;} 0 + x^* \mathbin{;} w \mathbin{;} y^*$
by (*metis add-isotone add-right-isotone mult-left-isotone star.left-plus-below-circ*)
also have $\ldots \leq (z \mathbin{;} y^\omega \mathbin{;} 0 + z \mathbin{;} y^* + w) \mathbin{;} y^\omega \mathbin{;} 0 + (z \mathbin{;} y^\omega \mathbin{;} 0 + z \mathbin{;} y^* + w) \mathbin{;} y^* + x^\omega \mathbin{;} 0 + x^* \mathbin{;} w \mathbin{;} y^\omega \mathbin{;} 0 + x^* \mathbin{;} w \mathbin{;} y^*$ **using** *1*
by (*metis add-left-isotone mult-associative mult-left-dist-add mult-left-isotone*)
also have $\ldots = z \mathbin{;} y^\omega \mathbin{;} 0 + z \mathbin{;} y^* \mathbin{;} y^\omega \mathbin{;} 0 + w \mathbin{;} y^\omega \mathbin{;} 0 + z \mathbin{;} y^\omega \mathbin{;} 0 + z \mathbin{;} y^* \mathbin{;} y^* + w \mathbin{;} y^* + x^\omega \mathbin{;} 0 + x^* \mathbin{;} w \mathbin{;} y^\omega \mathbin{;} 0 + x^* \mathbin{;} w \mathbin{;} y^*$
by (*smt add-associative mult-associative mult-left-zero mult-right-dist-add*)
also have $\ldots = z \mathbin{;} y^\omega \mathbin{;} 0 + z \mathbin{;} y^* + x^\omega \mathbin{;} 0 + x^* \mathbin{;} w \mathbin{;} y^\omega \mathbin{;} 0 + x^* \mathbin{;} w \mathbin{;} y^*$
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 $(x^\omega \mathbin{;} 0 + x^*) \mathbin{;} z \leq z \mathbin{;} y^\omega \mathbin{;} 0 + z \mathbin{;} y^* + x^\omega \mathbin{;} 0 + x^* \mathbin{;} w \mathbin{;} y^\omega \mathbin{;} 0 + x^* \mathbin{;} w \mathbin{;} y^*$

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 $(x^\omega \ ; \ 0 + x^*) \ ; \ z \leq (z + (x^\omega \ ; \ 0 + x^*) \ ; \ w) \ ; \ (y^\omega \ ; \ 0 + y^*)$
by (*smt add-associative mult-associative mult-left-dist-add mult-left-zero mult-right-dist-add*)
qed

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

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

end

sublocale *sdloa* < *comb0*!: *itering-4* **where** *circ* = $(\lambda x \ . \ x^* \ ; \ (x^\omega \ ; \ 0 + 1))$
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*)
apply (*metis omega-circ-simulate-right mult-left-zero add-right-zero omega-translate*)
apply (*metis omega-circ-simulate-right omega-translate*)
apply (*metis add-left-isotone comb0.circ-increasing mult-right-isotone omega-circ-simulate-left-plus omega-translate order-trans*)
apply (*metis omega-circ-simulate-right-plus omega-translate*)
apply (*metis omega-circ-simulate-left-plus omega-translate*)
done

class *lo-conway-semiring* = *loa* + *il-conway-semiring*

begin

subclass *lk-conway-semiring* **..**

lemma *circ-below-omega-star*: $x^\circ \leq x^\omega + x^*$
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 *loa-T* = *lka-T* + *loa*

begin

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

lemma *star-omega-top*: $x^{*\omega} = T$
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 *loa-T* < *comb0*!: *il-conway-semiring-T* **where** *circ* = $(\lambda x \ . \ x^{*} \ ; \ (x^{\omega} \ ; \ 0 + 1))$ **..**

class *sdloa-T* = *sdlka-T* + *sdloa*

begin

subclass *loa-T* **..**

end

sublocale *sdloa-T* < *comb0*!: *itering-T* **where** *circ* = $(\lambda x \ . \ x^{*} \ ; \ (x^{\omega} \ ; \ 0 + 1))$ **..**

class *lo-conway-semiring-T* = *loa-T* + *lo-conway-semiring*

begin

subclass *lk-conway-semiring* **..**

subclass *il-conway-semiring-T* **..**

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

end

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

class *gra* = *lka* + *Omega* +
  assumes *Omega-unfold* : $y^{\Omega} \leq 1 + y \mathbin{;} y^{\Omega}$
  assumes *Omega-induct* : $x \leq z + y \mathbin{;} x \rightarrow x \leq y^{\Omega} \mathbin{;} z$

begin

lemma *Omega-unfold-equal*: $y^{\Omega} = 1 + y \mathbin{;} y^{\Omega}$
  by (*smt Omega-induct Omega-unfold add-right-isotone antisym mult-right-isotone mult-right-one*)

lemma *Omega-add-1*: $(x + y)^{\Omega} = x^{\Omega} \mathbin{;} (y \mathbin{;} x^{\Omega})^{\Omega}$
  apply (*rule antisym*)
  apply (*smt Omega-induct Omega-unfold-equal add-associative add-commutative add-right-isotone mult-associative mult-right-dist-add mult-right-isotone mult-right-one order-refl*)
  apply (*smt Omega-induct Omega-unfold-equal add-associative add-commutative mult-associative mult-left-one mult-right-dist-add mult-right-one order-refl*)
  done

lemma *Omega-left-slide*: $(x \mathbin{;} y)^{\Omega} \mathbin{;} x \leq x \mathbin{;} (y \mathbin{;} x)^{\Omega}$
proof −
  have $1 + y \mathbin{;} (x \mathbin{;} y)^{\Omega} \mathbin{;} x \leq 1 + y \mathbin{;} x \mathbin{;} (1 + (y \mathbin{;} (x \mathbin{;} y)^{\Omega}) \mathbin{;} x)$
    by (*smt Omega-unfold-equal add-right-isotone mult-associative mult-left-one mult-left-sub-dist-add mult-right-dist-add mult-right-isotone mult-right-one*)
  thus *?thesis*
    by (*smt Omega-induct Omega-unfold-equal add-least-upper-bound mult-associative mult-left-one mult-right-dist-add mult-right-isotone mult-right-one*)
qed

end

sublocale *gra* < *Omega*!: *il-conway-semiring* **where** *circ* = *Omega*
  apply *unfold-locales*
  apply (*metis Omega-unfold-equal*)
  apply (*metis Omega-left-slide*)
  apply (*metis Omega-add-1*)
  done

context *gra*

begin

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

lemma *star-mult-Omega*: $x^{\Omega} = x^{*} \mathbin{;} x^{\Omega}$
  by (*metis Omega.left-plus-below-circ add-commutative add-left-upper-bound eq-iff star.circ-loop-fixpoint star-left-induct-mult-iff*)

lemma *Omega-one-greatest*: $x \leq 1^{\Omega}$

by (*metis Omega-induct add-left-zero mult-left-one order-refl order-trans zero-right-mult-decreasing*)

lemma *greatest-left-zero*: $1^\Omega \ ; \ x = 1^\Omega$
by (*metis antisym Omega-one-greatest Omega-induct add-right-upper-bound mult-left-one*)

end

class *gra-T* = *gra* + *lka-T*

begin

lemma *Omega-one*: $1^\Omega = T$
by (*metis Omega.circ-transitive-equal Omega-induct add-left-top add-right-upper-bound less-eq-def mult-left-one*)

lemma *top-left-zero*: $T \ ; \ x = T$
by (*metis Omega-induct Omega-one add-left-top add-right-upper-bound less-eq-def mult-left-one*)

end

sublocale *gra-T* < *Omega*!: *il-conway-semiring-T* **where** *circ* = *Omega* **..**

class *ldra* = *gra* +
assumes *Omega-isolate*: $y^\Omega \leq y^\Omega \ ; \ 0 + y^*$

begin

lemma *Omega-isolate-equal*: $y^\Omega = y^\Omega \ ; \ 0 + y^*$
by (*metis Omega-isolate add-commutative add-same-context less-eq-def star-below-Omega zero-right-mult-decreasing*)

end

class *ldra-T* = *ldra* + *lka-T*

begin

end

sublocale *ldra-T* < *Omega*!: *il-conway-semiring-T* **where** *circ* = *Omega* **..**

class *dra* = *sdlka* + *ldra*

begin

lemma *Omega-mult*: $(x\ ;\ y)^\Omega = 1 + x\ ;\ (y\ ;\ x)^\Omega\ ;\ y$
by (*smt Omega.circ-left-slide Omega-induct Omega-unfold-equal eq-iff mult-associative mult-left-dist-add mult-right-one*)

lemma *Omega-add*: $(x + y)^\Omega = (x^\Omega\ ;\ y)^\Omega\ ;\ x^\Omega$
by (*smt Omega-add-1 Omega-mult mult-associative mult-left-dist-add mult-left-one mult-right-dist-add mult-right-one*)

lemma *Omega-simulate*: $z\ ;\ x \leq y\ ;\ z \rightarrow z\ ;\ x^\Omega \leq y^\Omega\ ;\ z$
by (*smt Omega-induct Omega-unfold-equal add-right-isotone mult-associative mult-left-dist-add mult-left-isotone mult-right-one*)

end

sublocale *dra* $<$ *Omega*!: *itering-1* **where** *circ* $=$ *Omega*
apply *unfold-locales*
apply (*metis Omega-simulate mult-associative order-refl*)
apply (*metis Omega-simulate*)
done

sublocale *dra* $<$ *Omega*!: *sidl-conway-semiring* **where** *circ* $=$ *Omega* **..**

context *dra*

begin

lemma *Omega-sum-unfold-1*: $(x + y)^\Omega = y^\Omega + y^*\ ;\ x\ ;\ (x + y)^\Omega$
by (*smt Omega.circ-add Omega.circ-loop-fixpoint Omega-isolate-equal add-associative add-commutative mult-associative mult-left-zero mult-right-dist-add*)

lemma *Omega-add-3*: $(x + y)^\Omega = (x^*\ ;\ y)^\Omega\ ;\ x^\Omega$
by (*smt Omega.circ-add Omega.circ-isotone Omega-induct Omega-sum-unfold-1 add-commutative antisym mult-left-isotone order-refl star-below-Omega*)

lemma *star-mult-Omega*: $x^\Omega = x^*\ ;\ x^\Omega$
by (*metis Omega.circ-plus-one Omega.circ-transitive-equal Omega-isolate-equal add-commutative less-eq-def order-refl star.circ-add-mult-zero star.circ-increasing star.circ-plus-same star-involutive star-left-induct star-square*)

lemma *Omega-separate-2*: $y\ ;\ x \leq x\ ;\ (x + y) \rightarrow (x + y)^\Omega = x^\Omega\ ;\ y^\Omega$
by (*smt Omega.circ-sub-dist-3 Omega-induct Omega-sum-unfold-1 add-right-isotone antisym mult-associative mult-left-isotone star-mult-Omega star-simulation-left*)

lemma *Omega-circ-simulate-right-plus*: $z\ ;\ x \leq y\ ;\ y^\Omega\ ;\ z + w \rightarrow z\ ;\ x^\Omega \leq y^\Omega\ ;\ (z + w\ ;\ x^\Omega)$
proof
assume *1*: $z\ ;\ x \leq y\ ;\ y^\Omega\ ;\ z + w$
have $z\ ;\ x^\Omega = z + z\ ;\ x\ ;\ x^\Omega$
by (*metis Omega.circ-back-loop-fixpoint Omega.circ-plus-same add-commutative mult-associative*)
also have $... \leq y\ ;\ y^\Omega\ ;\ z\ ;\ x^\Omega + z + w\ ;\ x^\Omega$ **using** *1*
by (*smt add-associative add-commutative add-right-isotone less-eq-def mult-right-dist-add*)
finally have $z\ ;\ x^\Omega \leq (y\ ;\ y^\Omega)^\Omega\ ;\ (z + w\ ;\ x^\Omega)$
by (*smt Omega-induct add-associative add-commutative mult-associative*)
thus $z\ ;\ x^\Omega \leq y^\Omega\ ;\ (z + w\ ;\ x^\Omega)$
by (*metis Omega.left-plus-circ*)

qed

lemma *Omega-circ-simulate-left-plus*: $x \ ; \ z \leq z \ ; \ y^{\Omega} + w \rightarrow x^{\Omega} \ ; \ z \leq (z + x^{\Omega} \ ; \ w) \ ; \ y^{\Omega}$
proof
assume $x \ ; \ z \leq z \ ; \ y^{\Omega} + w$
hence $x \ ; \ ((z + x^{\Omega} \ ; \ w) \ ; \ y^{\Omega}) \leq (z \ ; \ y^{\Omega} + w + x \ ; \ x^{\Omega} \ ; \ w) \ ; \ y^{\Omega}$
by (*smt mult-associative mult-left-dist-add add-left-isotone mult-left-isotone*)
also have $\ldots \leq z \ ; \ y^{\Omega} \ ; \ y^{\Omega} + w \ ; \ y^{\Omega} + x^{\Omega} \ ; \ w \ ; \ y^{\Omega}$
by (*smt Omega.left-plus-below-circ add-right-isotone mult-left-isotone mult-right-dist-add*)
finally have $x \ ; \ ((z + x^{\Omega} \ ; \ w) \ ; \ y^{\Omega}) \leq (z + x^{\Omega} \ ; \ w) \ ; \ y^{\Omega}$
by (*metis Omega.circ-transitive-equal mult-associative Omega.circ-reflexive add-associative less-eq-def mult-left-one mult-right-dist-add*)
thus $x^{\Omega} \ ; \ z \leq (z + x^{\Omega} \ ; \ w) \ ; \ y^{\Omega}$
by (*smt Omega.circ-back-loop-fixpoint Omega-isolate-equal add-least-upper-bound mult-associative mult-left-zero mult-right-dist-add mult-right-sub-dist-add-left mult-right-sub-dist-add-right star-left-induct*)
qed

lemma *Omega-circ-simulate-right*: $z \ ; \ x \leq y \ ; \ z + w \rightarrow z \ ; \ x^{\Omega} \leq y^{\Omega} \ ; \ (z + w \ ; \ x^{\Omega})$
proof
assume $z \ ; \ x \leq y \ ; \ z + w$
also have $\ldots \leq y \ ; \ y^{\Omega} \ ; \ z + w$
by (*smt Omega.circ-loop-fixpoint add-associative add-commutative add-left-upper-bound mult-associative mult-left-dist-add*)
finally show $z \ ; \ x^{\Omega} \leq y^{\Omega} \ ; \ (z + w \ ; \ x^{\Omega})$
by (*metis Omega-circ-simulate-right-plus*)
qed

end

sublocale *dra* $<$ *Omega*!: *itering-4* **where** *circ* $=$ *Omega*
apply *unfold-locales*
apply (*metis Omega-circ-simulate-right*)
apply (*metis Omega.circ-increasing Omega-circ-simulate-left-plus add-commutative add-right-isotone mult-right-isotone order-trans*)
apply (*metis Omega-circ-simulate-right-plus*)
apply (*metis Omega-circ-simulate-left-plus*)
done

class *dra-T* $=$ *dra* $+$ *sdlka-T*

begin

lemma *Omega-one*: $1^{\Omega} = T$
by (*metis Omega.circ-transitive-equal Omega-induct add-left-top add-right-upper-bound less-eq-def mult-left-one*)

lemma *top-left-zero*: $T \ ; \ x = T$
by (*metis Omega-induct Omega-one add-left-top add-right-upper-bound less-eq-def mult-left-one*)

end

sublocale *dra-T* $<$ *Omega*!: *itering-T* **where** *circ* $=$ *Omega* **..**

class *gra-omega* $=$ *loa* $+$ *Omega* $+$
  assumes *omega-left-zero*: $x^\omega \leq x^\omega \ ; \ y$
  assumes *Omega-def*: $x^\Omega = x^\omega + x^*$

begin

lemma *omega-left-zero-equal*: $x^\omega \ ; \ y = x^\omega$
  by (*metis antisym omega-left-zero omega-sub-vector*)

subclass *ldra*
  apply *unfold-locales*
  apply (*metis Omega-def add-commutative eq-refl mult-right-one omega-loop-fixpoint*)
  apply (*metis Omega-def mult-right-dist-add omega-induct omega-left-zero-equal*)
  apply (*smt Omega-def add-least-upper-bound antisym mult-right-dist-add mult-right-sub-dist-add-left omega-left-zero-equal order-refl star-zero-below-omega*)
  done

end

class *ldra-omega* $=$ *loa-T* $+$ *Omega* $+$
  assumes *top-left-zero*: $T \ ; \ x = T$
  assumes *Omega-def*: $x^\Omega = x^\omega + x^*$

begin

subclass *gra-omega*
  apply *unfold-locales*
  apply (*metis mult-associative omega-vector order-refl top-left-zero*)
  apply (*rule Omega-def*)
  done

end

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

class *il-semiring-lattice* $=$ *il-semiring-T* $+$ *meet* $+$
  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-left-zero* : $0 \frown x = 0$
  assumes *meet-left-top* : $T \frown x = x$
  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 *less-eq-meet*: $x \leq y \leftrightarrow x \frown y = x$
by (*metis add-absorb add-right-upper-bound less-eq-def meet-absorb meet-commutative*)

lemma *meet-left-isotone*: $x \leq y \rightarrow x \frown z \leq y \frown z$
by (*smt less-eq-def meet-commutative meet-left-dist-add*)

lemma *meet-right-isotone*: $x \leq y \rightarrow z \frown x \leq z \frown y$
by (*metis meet-commutative meet-left-isotone*)

lemma *meet-isotone*: $w \leq y \wedge x \leq z \rightarrow w \frown x \leq y \frown z$
by (*smt less-eq-meet meet-associative meet-commutative*)

lemma *meet-left-upper-bound*: $x \frown y \leq x$
by (*metis add-absorb add-right-divisibility*)

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

lemma *meet-least-upper-bound*: $z \leq x \wedge z \leq y \leftrightarrow z \leq x \frown y$
by (*metis less-eq-meet meet-associative meet-left-upper-bound*)

lemma *meet-left-divisibility*: $y \leq x \leftrightarrow (\exists z \,.\, x \frown z = y)$
by (*metis less-eq-meet meet-commutative meet-left-upper-bound*)

lemma *meet-right-divisibility*: $y \leq x \leftrightarrow (\exists z \,.\, z \frown x = y)$
by (*metis meet-commutative meet-left-divisibility*)

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

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

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

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

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

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

lemma *mult-right-top*: $x \frown T = x$
by (*metis meet-commutative meet-left-top*)

lemma *vector-mult-closed*: *vector* $x \wedge$ *vector* $y \rightarrow$ *vector* $(x \frown y)$
by (*metis antisym meet-least-upper-bound mult-right-sub-dist-meet-left mult-right-sub-dist-meet-right top-right-mult-increasing vector-def*)

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

end

class *idl-semiring-lattice* = *idl-semiring-T* + *il-semiring-lattice*

class *nL* =
fixes $n :: 'a \Rightarrow 'a$
fixes $L :: 'a$

class *il-semiring-lattice-nL* = *il-semiring-lattice* + *nL* +
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-isotone-idempotent*: $n(x)\ ;\ n(x + y) = n(x)$
assumes *n-sub-nL-meet-one* : $n(x) \leq n(L) \frown 1$
assumes *n-L-decreasing* : $n(x)\ ;\ L \leq x$
assumes *n-nL-semi-commute* : $n(L)\ ;\ x \leq x\ ;\ n(L)$
assumes *n-nL-meet-L-nL0* : $n(L)\ ;\ x = (x \frown L) + n(L\ ;\ 0)\ ;\ x$
assumes *n-L-split-n-L-L* : $x\ ;\ L = x\ ;\ 0 + n(x\ ;\ L)\ ;\ L$
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

lemma *n-sub-one*: $n(x) \leq 1$
by (*metis meet-least-upper-bound n-sub-nL-meet-one*)

lemma *n-mult-idempotent* : $n(x)\ ;\ n(x) = n(x)$
by (*metis add-idempotent n-isotone-idempotent*)

lemma *n-L-increasing*: $n(x) \leq n(n(x)\ ;\ L)$
by (*smt meet-least-upper-bound mult-right-isotone n-export n-mult-idempotent n-sub-nL-meet-one*)

lemma *meet-L-below-n-L*: $x \frown L \leq n(L)\ ;\ x$
by (*metis add-left-divisibility n-nL-meet-L-nL0*)

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-mult-right-zero*: $n(x) \ ; \ 0 = 0$
by (*metis antisym mult-left-isotone mult-left-one n-sub-one zero-least*)

lemma *n-mult-left-absorb-add*: $n(x) \ ; \ (n(x) + n(y)) = n(x)$
by (*metis add-commutative n-dist-n-add n-isotone-idempotent*)

lemma *n-mult-right-absorb-add*: $(n(x) + n(y)) \ ; \ n(y) = n(y)$
by (*metis less-eq-def mult-left-one mult-right-dist-add n-mult-idempotent n-sub-one*)

lemma *n-add-left-absorb-mult*: $n(x) + n(x) \ ; \ n(y) = n(x)$
by (*metis add-commutative less-eq-def mult-left-sub-dist-add-right n-mult-left-absorb-add*)

lemma *n-add-right-absorb-mult*: $n(x) \ ; \ n(y) + n(y) = n(y)$
by (*metis less-eq-def mult-left-one mult-right-dist-add n-sub-one*)

lemma *n-mult-commutative*: $n(x) \ ; \ n(y) = n(y) \ ; \ n(x)$
by (*metis add-commutative mult-associative n-add-left-absorb-mult n-add-right-absorb-mult n-export n-mult-left-absorb-add n-mult-right-absorb-add*)

lemma *n-add-right-dist-mult*: $n(x) \ ; \ n(y) + n(z) = (n(x) + n(z)) \ ; \ (n(y) + n(z))$
by (*smt add-associative mult-right-dist-add n-dist-n-add n-mult-commutative n-mult-idempotent n-mult-right-absorb-add*)

lemma *n-add-left-dist-mult*: $n(x) + n(y) \ ; \ n(z) = (n(x) + n(y)) \ ; \ (n(x) + n(z))$
by (*metis add-commutative n-add-right-dist-mult*)

lemma *n-order*: $n(x) \leq n(y) \leftrightarrow n(x) \ ; \ n(y) = n(x)$
by (*metis less-eq-def n-add-right-absorb-mult n-mult-left-absorb-add*)

lemma *n-mult-left-lower-bound*: $n(x) \ ; \ n(y) \leq n(x)$
by (*metis add-right-upper-bound n-add-left-absorb-mult*)

lemma *n-mult-right-lower-bound*: $n(x) \ ; \ n(y) \leq n(y)$
by (*metis add-commutative add-right-upper-bound n-add-right-absorb-mult*)

lemma *n-mult-least-upper-bound*: $n(x) \leq n(y) \wedge n(x) \leq n(z) \leftrightarrow n(x) \leq n(y) \ ; \ n(z)$
by (*metis mult-left-isotone n-mult-left-lower-bound n-mult-right-lower-bound n-order order-trans*)

lemma *n-mult-left-divisibility*: $n(x) \leq n(y) \leftrightarrow (\exists \, z \ . \ n(x) = n(y) \ ; \ n(z))$
by (*metis antisym mult-left-isotone n-mult-idempotent n-mult-left-lower-bound n-mult-right-lower-bound*)

lemma *n-mult-right-divisibility*: $n(x) \leq n(y) \leftrightarrow (\exists \, z \ . \ n(x) = n(z) \ ; \ n(y))$
by (*metis n-mult-right-lower-bound n-order*)

lemma *n-left-upper-bound*: $n(x) \leq n(x + y)$

by (*metis n-isotone-idempotent n-order*)

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

lemma *n-isotone*: $x \leq y \rightarrow n(x) \leq n(y)$
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*)

lemma *n-mult-left-zero*: $n(0)\ ;\ n(x) = n(0)$
by (*metis add-left-zero n-isotone-idempotent*)

lemma *n-mult-right-zero-n*: $n(x)\ ;\ n(0) = n(0)$
by (*metis add-commutative n-add-left-zero n-mult-right-absorb-add*)

lemma *n-mult-left-one*: $n(T)\ ;\ n(x) = n(x)$
by (*metis add-commutative add-right-top n-dist-n-add n-mult-right-absorb-add*)

lemma *n-mult-right-one*: $n(x)\ ;\ n(T) = n(x)$
by (*metis add-right-top n-isotone-idempotent*)

lemma *n-add-left-top*: $n(T) + n(x) = n(T)$
by (*metis n-add-left-absorb-mult n-mult-left-one*)

lemma *n-mult-left-dist-add*: $n(x)\ ;\ (n(y) + n(z)) = (n(x)\ ;\ n(y)) + (n(x)\ ;\ n(z))$
by (*metis mult-right-dist-add n-dist-n-add n-mult-commutative*)

lemma *n-n-L*: $n(n(x)\ ;\ L) = n(x)$
by (*metis antisym n-L-decreasing n-L-increasing n-isotone*)

lemma *n-galois*: $n(x) \leq n(y) \leftrightarrow n(x)\ ;\ L \leq y$
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*)

lemma *n-less-eq-char-n*: $x \leq y \leftrightarrow x \leq y + L \wedge n(L)\ ;\ x \leq y + n(y)\ ;\ T$
proof
assume $x \leq y$
thus $x \leq y + L \wedge n(L)\ ;\ x \leq y + n(y)\ ;\ T$
by (*metis less-eq-meet meet-absorb mult-isotone mult-left-one n-sub-one order-trans*)
next
assume *1*: $x \leq y + L \wedge n(L)\ ;\ x \leq y + n(y)\ ;\ T$
hence $x \leq y + (x \frown L)$
by (*metis less-eq-meet add-left-dist-meet add-right-upper-bound meet-left-isotone*)

also have ... $\leq y + n(y)\ ;\ T$ **using** *1*
by (*metis add-least-upper-bound add-left-upper-bound meet-L-below-n-L order-trans*)
finally have $x \leq y + (L \frown n(y)\ ;\ T)$ **using** *1*
by (*metis meet-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

lemma *n-preserves-equation*: $n(y)\ ;\ x \leq x\ ;\ n(y) \leftrightarrow n(y)\ ;\ x = n(y)\ ;\ x\ ;\ n(y)$
by (*metis n-mult-idempotent n-sub-one order-refl test-preserves-equation*)

lemma *n-L-decreasing-meet-L*: $n(x)\ ;\ L \leq x \frown L$
by (*metis add-commutative less-eq-meet meet-absorb meet-commutative meet-least-upper-bound mult-right-sub-dist-add-left n-L-decreasing n-add-left-top n-sub-nL-meet-one*)

lemma *n-sub-nL*: $n(x) \leq n(L)$
by (*metis meet-least-upper-bound n-sub-nL-meet-one*)

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 vector-zero*)
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-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

lemma *n-L-top-L*: $L\ ;\ T = L$
by (*metis antisym n-L-top-below-L top-right-mult-increasing*)

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*)

lemma *n-L-split-L*: $x\ ;\ L \leq x\ ;\ 0 + L$
by (*metis add-commutative add-left-isotone meet-least-upper-bound n-L-decreasing-meet-L n-L-split-n-L-L*)

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

lemma *n-top-split*: $n(x)\ ;\ T\ ;\ y \leq x\ ;\ y + n(x\ ;\ y)\ ;\ T$
proof −
have $n(x)\ ;\ T\ ;\ y \frown L \leq x\ ;\ y$
by (*metis mult-left-isotone n-L-decreasing n-top-meet-L-below-L order-trans*)
thus *?thesis*
by (*smt add-isotone mult-associative mult-isotone n-L-decreasing n-export n-isotone n-mult-commutative n-nL-meet-L-nL0 n-n-L top-greatest zero-least*)
qed

lemma *n-nL-nT*: $n(L) = n(T)$
by (*metis antisym n-isotone n-sub-nL top-greatest*)

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*)

lemma *n-L-top-L-L*: $L\ ;\ T\ ;\ L = L$
by (*metis n-L-L-L n-L-top-L*)

lemma *n-L-below-nL-top*: $L \leq n(L)\ ;\ T$
by (*metis meet-L-below-n-L meet-left-top*)

lemma *n-n-nL*: $n(x) = n(x)\ ;\ n(L)$
by (*metis n-order n-sub-nL*)

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*)

lemma *n-L-nL-L*: $L\ ;\ n(L) = L$
by (*metis antisym n-vector-meet-L n-L-below-L less-eq-meet meet-commutative n-L-below-nL-top n-nL-semi-commute order-trans*)

lemma *n-L-nT-L*: $L\ ;\ n(T) = L$
by (*metis n-L-nL-L n-nL-nT*)

lemma *n-L-L*: $n(L)\ ;\ L = L$
by (*metis add-left-zero mult-left-one n-L-split-n-L-L*)

lemma *n-top-L*: $n(T)\ ;\ L = L$
by (*metis n-L-L n-nL-nT*)

lemma *n-n-L-split-n-L*: $x\ ;\ n(y)\ ;\ L \leq x\ ;\ 0 + n(x\ ;\ y)\ ;\ L$
by (*metis add-right-isotone n-L-top-below-L mult-associative mult-isotone n-L-split-n-L-L n-L-top-L n-mult-right-zero n-n-L-n*)

lemma *n-n-L-split-n-n-L-L*: $x\ ;\ n(y)\ ;\ L = x\ ;\ 0 + n(x\ ;\ n(y)\ ;\ L)\ ;\ L$
by (*metis mult-associative n-L-split-n-L-L n-mult-right-zero*)

lemma *n-nL-split-n-L-top*: $n(L)\ ;\ x \leq x\ ;\ 0 + n(x\ ;\ L)\ ;\ T$

by (*metis n-nL-semi-commute n-n-top-split-n-top order-trans top-right-mult-increasing*)

lemma *n-less-eq-char*: $x \leq y \leftrightarrow x \leq y + L \wedge x \leq y + n(y)\ ;\ T$
by (*smt add-absorb add-associative add-idempotent n-less-eq-char-n less-eq-meet meet-left-dist-add n-add-n-top*)

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*)
thus *?thesis*
by (*metis n-sub-one mult-left-isotone add-right-isotone mult-left-one order-trans*)
qed

lemma *n-top-meet-L-L-meet-L*: $x\ ;\ T\ ;\ y \frown L = x\ ;\ L\ ;\ y \frown L$
apply (*rule antisym*)
apply (*metis meet-least-upper-bound meet-right-upper-bound n-top-meet-L-below-L*)
apply (*metis meet-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*)

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*)

lemma *n-meet-L-0-below-0-meet-L*: $(x \frown L)\ ;\ 0 \leq x\ ;\ 0 \frown L$
by (*metis meet-least-upper-bound mult-right-sub-dist-meet-left zero-right-mult-decreasing*)

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*)

lemma *n-below-n-L*: $n(x) \leq n(x\ ;\ L)$
by (*metis n-n-L-below-L n-galois*)

lemma *n-n-meet-L-n-zero*: $n(x) = n(x) \frown L + n(x\ ;\ 0)$
apply (*rule antisym*)
apply (*smt add-right-isotone mult-associative mult-left-isotone n-L-decreasing n-export n-isotone n-mult-commutative n-nL-meet-L-nL0 n-n-L*)
apply (*metis add-least-upper-bound meet-left-upper-bound n-isotone zero-right-mult-decreasing*)
done

lemma *n-meet-L-below*: $n(x) \frown L \leq x$
by (*metis meet-left-isotone n-L-decreasing n-vector-meet-L order-trans top-right-mult-increasing*)

lemma *n-below-n-zero*: $n(x) \leq x + n(x\ ;\ 0)$

by (*metis add-left-isotone n-meet-L-below n-n-meet-L-n-zero*)

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-least-upper-bound mult-right-sub-dist-meet-left mult-right-sub-dist-meet-right*)
thus *?thesis*
by (*metis n-L-top-L n-vector-meet-L order-trans*)
qed

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*)

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-n-top-split-n-L-n-zero-top*: $n(x)\ ;\ T = n(x)\ ;\ L + n(x\ ;\ 0)\ ;\ T$
apply (*rule antisym*)
apply (*metis add-left-isotone mult-right-dist-add n-meet-L-top-below-n-L n-n-meet-L-n-zero*)
apply (*metis add-least-upper-bound mult-left-isotone mult-right-isotone n-isotone top-greatest zero-right-mult-decreasing*)
done

end

class *idl-semiring-lattice-nL* = *il-semiring-lattice-nL* + *idl-semiring-lattice*

begin

end

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

class *idl-semiring-lattice-apx* = *idl-semiring-lattice-nL* + *apx* +
assumes *apx-def*: $x \sqsubseteq y \leftrightarrow x \leq y + L \wedge n(L)\ ;\ y \leq x + n(x)\ ;\ T$

begin

definition *apx-isotone* :: $('a \Rightarrow 'a) \Rightarrow bool$

where *apx-isotone* $f \leftrightarrow (\forall\, x\ y\ .\ x \sqsubseteq y \rightarrow f(x) \sqsubseteq f(y))$

lemma *apx-reflexive*: $x \sqsubseteq x$
by (*smt add-least-upper-bound add-left-upper-bound apx-def less-eq-def mult-left-one mult-right-dist-add n-sub-one*)

lemma *apx-L-least*: $L \sqsubseteq x$
by (*metis add-least-upper-bound add-right-isotone add-right-upper-bound apx-def mult-right-isotone top-greatest*)

lemma *apx-transitive*: $x \sqsubseteq y \wedge y \sqsubseteq z \rightarrow x \sqsubseteq z$
proof
assume *1*: $x \sqsubseteq y \wedge y \sqsubseteq z$
hence $n(L)\ ;\ z \leq n(L)\ ;\ y + n(L)\ ;\ n(y)\ ;\ T$
by (*metis apx-def mult-associative mult-left-dist-add mult-right-isotone n-mult-idempotent*)
also have $\ldots \leq x + n(x)\ ;\ T + n(n(L)\ ;\ y)\ ;\ T$ **using** *1*
by (*metis add-left-isotone apx-def n-export*)
also have $\ldots \leq x + n(x)\ ;\ T$ **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*: $y \sqsubseteq x \rightarrow x \frown L \leq y \frown L$
proof
assume *1*: $y \sqsubseteq x$
have $n(L)\ ;\ (x \frown L) \leq n(L)\ ;\ x \frown L$
by (*metis eq-iff meet-L-below-n-L meet-least-upper-bound mult-left-sub-dist-meet-right n-L-decreasing*)
also have $\ldots \leq (y \frown L) + (n(y)\ ;\ T \frown L)$ **using** *1*
by (*metis apx-def meet-commutative meet-left-dist-add meet-left-isotone*)
also have $\ldots \leq (y \frown L) + n(y \frown L)\ ;\ T$
by (*metis add-least-upper-bound n-vector-meet-L meet-least-upper-bound n-L-decreasing order-refl order-trans*)
finally show $x \frown L \leq y \frown L$
by (*metis n-less-eq-char-n less-eq-def meet-right-upper-bound*)
qed

lemma *apx-antisymmetric*: $x \sqsubseteq y \wedge y \sqsubseteq x \rightarrow x = y$
by (*metis add-same-context antisym apx-def apx-meet-L relative-equality*)

lemma *add-apx-left-isotone*: $x \sqsubseteq y \rightarrow x + z \sqsubseteq y + z$
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 $n(L)\ ;\ (y + z) = n(L)\ ;\ y + n(L)\ ;\ z$
by (*metis mult-left-dist-add*)
also have $\ldots \leq n(L)\ ;\ y + z$
by (*metis add-commutative add-least-upper-bound add-right-upper-bound n-sub-one mult-left-dist-add mult-left-isotone mult-left-one*)
also have $\ldots \leq x + n(x)\ ;\ T + z$ **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 *add-apx-right-isotone*: $x \sqsubseteq y \rightarrow z + x \sqsubseteq z + y$
by (*metis add-commutative add-apx-left-isotone*)

lemma *add-apx-isotone*: $w \sqsubseteq y \wedge x \sqsubseteq z \rightarrow w + x \sqsubseteq y + z$
by (*metis add-apx-left-isotone add-apx-right-isotone apx-transitive*)

lemma *mult-apx-left-isotone*: $x \sqsubseteq y \rightarrow x\ ;\ z \sqsubseteq y\ ;\ z$
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 $n(L)\ ;\ y\ ;\ z \leq x\ ;\ z + n(x)\ ;\ T\ ;\ z$ **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 mult-associative*)
qed

lemma *mult-apx-right-isotone*: $x \sqsubseteq y \rightarrow z\ ;\ x \sqsubseteq z\ ;\ y$
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 $n(L)\ ;\ z\ ;\ y \leq z\ ;\ n(L)\ ;\ y$
by (*metis n-nL-semi-commute mult-left-isotone*)
also have ... $\leq z\ ;\ (x + n(x)\ ;\ T)$ **using** *1*
by (*metis apx-def mult-associative 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 mult-associative*)

qed

lemma *mult-apx-isotone*: $w \sqsubseteq y \wedge x \sqsubseteq z \rightarrow w \;;\; x \sqsubseteq y \;;\; z$
by (*metis apx-transitive mult-apx-left-isotone mult-apx-right-isotone*)

lemma *meet-L-apx-isotone*: $x \sqsubseteq y \rightarrow x \frown L \sqsubseteq y \frown L$
by (*smt add-commutative add-idempotent add-left-dist-meet apx-def apx-meet-L n-less-eq-char-n meet-commutative meet-right-isotone*)

lemma *n-L-apx-isotone*: $x \sqsubseteq y \rightarrow n(x) \;;\; L \sqsubseteq n(y) \;;\; L$
proof
assume $x \sqsubseteq y$
hence $n(L) \;;\; n(y) \;;\; L \leq n(x) \;;\; L + n(n(x) \;;\; L) \;;\; T$
by (*metis add-left-upper-bound apx-def mult-left-isotone n-add-n-top n-export n-isotone order-trans*)
thus $n(x) \;;\; L \sqsubseteq n(y) \;;\; L$
by (*metis apx-def less-eq-def meet-least-upper-bound mult-associative n-L-decreasing-meet-L*)
qed

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

definition *is-apx-prefixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a \Rightarrow bool$ **where** *is-apx-prefixpoint* $f\ x \leftrightarrow f(x) \sqsubseteq x$
definition *is-apx-least-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a \Rightarrow bool$ **where** *is-apx-least-fixpoint* $f\ x \leftrightarrow f(x) = x \wedge (\forall y \;.\; f(y) = y \rightarrow x \sqsubseteq y)$
definition *is-apx-least-prefixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a \Rightarrow bool$ **where** *is-apx-least-prefixpoint* $f\ x \leftrightarrow f(x) \sqsubseteq x \wedge (\forall y \;.\; f(y) \sqsubseteq y \rightarrow x \sqsubseteq y)$
definition *has-apx-prefixpoint* :: $('a \Rightarrow 'a) \Rightarrow bool$ **where** *has-apx-prefixpoint* $f \leftrightarrow (\exists x \;.\; is\text{-}apx\text{-}prefixpoint\ f\ x)$
definition *has-apx-least-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow bool$ **where** *has-apx-least-fixpoint* $f \leftrightarrow (\exists x \;.\; is\text{-}apx\text{-}least\text{-}fixpoint\ f\ x)$
definition *has-apx-least-prefixpoint* :: $('a \Rightarrow 'a) \Rightarrow bool$ **where** *has-apx-least-prefixpoint* $f \leftrightarrow (\exists x \;.\; is\text{-}apx\text{-}least\text{-}prefixpoint\ f\ x)$
definition *the-apx-least-fixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a$ $(\xi\ \text{-}\ [201]\ 200)$ **where** $\xi\ f = (THE\ x \;.\; is\text{-}apx\text{-}least\text{-}fixpoint\ f\ x)$
definition *the-apx-least-prefixpoint* :: $('a \Rightarrow 'a) \Rightarrow 'a$ $(p\xi\ \text{-}\ [201]\ 200)$ **where** $p\xi\ f = (THE\ x \;.\; is\text{-}apx\text{-}least\text{-}prefixpoint\ f\ x)$

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

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

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

lemma *apx-least-prefixpoint*: *has-apx-least-prefixpoint* $f \rightarrow$ *is-apx-least-prefixpoint* $f\ (p\xi\ f)$
proof
assume *has-apx-least-prefixpoint f*

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

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

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

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

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

lemma *apx-least-prefixpoint-fixpoint*: *has-apx-least-prefixpoint f* ∧ *apx-isotone f* ⟶ *is-apx-least-fixpoint f* (*pξ f*)
by (*smt apx-antisymmetric apx-isotone-def apx-reflexive is-apx-least-fixpoint-def is-apx-least-prefixpoint-def apx-least-prefixpoint*)

lemma *pxi-xi*: *has-apx-least-prefixpoint f* ∧ *apx-isotone f* ⟶ *pξ f* = *ξ f*
by (*smt has-apx-least-fixpoint-def is-apx-least-fixpoint-def apx-least-fixpoint-unique apx-least-prefixpoint-fixpoint apx-least-fixpoint*)

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

lemma *lifted-reflexive*: *f* = *g* ⟶ *f* ⊑⊑ *g*
by (*metis lifted-apx-less-eq-def apx-reflexive*)

lemma *lifted-transitive*: *f* ⊑⊑ *g* ∧ *g* ⊑⊑ *h* ⟶ *f* ⊑⊑ *h*
by (*smt lifted-apx-less-eq-def apx-transitive*)

lemma *lifted-antisymmetric*: *f* ⊑⊑ *g* ∧ *g* ⊑⊑ *f* ⟶ *f* = *g*
by (*metis apx-antisymmetric ext lifted-apx-less-eq-def*)

lemma *pxi-isotone*: *has-apx-least-prefixpoint f* ∧ *has-apx-least-prefixpoint g* ∧ *f* ⊑⊑ *g* ⟶ *pξ f* ⊑ *pξ g*
by (*metis is-apx-least-prefixpoint-def apx-transitive apx-least-prefixpoint lifted-apx-less-eq-def*)

lemma *xi-isotone*: *has-apx-least-prefixpoint f* ∧ *has-apx-least-prefixpoint g* ∧ *apx-isotone f* ∧ *apx-isotone g* ∧ *f* ⊑⊑ *g* ⟶ *ξ f* ⊑ *ξ g*
by (*metis pxi-isotone pxi-xi*)

lemma *xi-square*: *apx-isotone f* ∧ *has-apx-least-fixpoint f* ∧ *has-apx-least-fixpoint* (*f* ∘ *f*) ⟶ *ξ f* = *ξ* (*f* ∘ *f*)
by (*metis apx-antisymmetric is-apx-least-fixpoint-def apx-isotone-def apx-least-fixpoint-char apx-least-fixpoint-unique o-apply*)

lemma *mu-below-xi*: *has-least-fixpoint f* ∧ *has-apx-least-fixpoint f* ⟶ *μ f* ≤ *ξ f*
by (*metis apx-least-fixpoint is-apx-least-fixpoint-def is-least-fixpoint-def least-fixpoint*)

lemma *xi-below-nu*: *has-greatest-fixpoint f* $\wedge$ *has-apx-least-fixpoint f* $\rightarrow$ $\xi\ f \leq \nu\ f$
by (*metis apx-least-fixpoint greatest-fixpoint is-apx-least-fixpoint-def is-greatest-fixpoint-def*)

lemma *xi-apx-below-mu*: *has-least-fixpoint f* $\wedge$ *has-apx-least-fixpoint f* $\rightarrow$ $\xi\ f \sqsubseteq \mu\ f$
by (*metis apx-least-fixpoint is-apx-least-fixpoint-def is-least-fixpoint-def least-fixpoint*)

lemma *xi-apx-below-nu*: *has-greatest-fixpoint f* $\wedge$ *has-apx-least-fixpoint f* $\rightarrow$ $\xi\ f \sqsubseteq \nu\ f$
by (*metis apx-least-fixpoint greatest-fixpoint is-apx-least-fixpoint-def is-greatest-fixpoint-def*)

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

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

lemma *apx-meet*: *has-apx-meet x y* $\rightarrow$ *is-apx-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-apx-meet x y* $\rightarrow$ $(w \sqsubseteq x \wedge w \sqsubseteq y \leftrightarrow w \sqsubseteq x \triangle y)$
by (*smt apx-meet apx-transitive is-apx-meet-def*)

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

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

definition *xi-apx-meet* :: $('a \Rightarrow 'a) \Rightarrow bool$
where *xi-apx-meet f* $\leftrightarrow$ *has-apx-least-fixpoint f* $\wedge$ *has-apx-meet* $(\mu\ f)\ (\nu\ f) \wedge \xi\ f = \mu\ f \triangle \nu\ f$

definition *xi-mu-nu* :: $('a \Rightarrow 'a) \Rightarrow bool$
where *xi-mu-nu f* $\leftrightarrow$ *has-apx-least-fixpoint f* $\wedge$ $\xi\ f = \mu\ f + (\nu\ f \frown L)$

definition *nu-below-mu-nu* :: $('a \Rightarrow 'a) \Rightarrow bool$
where *nu-below-mu-nu f* $\leftrightarrow$ $n(L)\ ;\ \nu\ f \leq \mu\ f + (\nu\ f \frown L) + n(\nu\ f)\ ;\ T$

definition *nu-below-mu-nu-2* :: $('a \Rightarrow 'a) \Rightarrow bool$
where *nu-below-mu-nu-2 f* $\leftrightarrow$ $n(L)\ ;\ \nu\ f \leq \mu\ f + (\nu\ f \frown L) + n(\mu\ f + (\nu\ f \frown L))\ ;\ T$

definition *mu-nu-apx-nu* :: $('a \Rightarrow 'a) \Rightarrow bool$
  where *mu-nu-apx-nu f* $\leftrightarrow \mu\ f + (\nu\ f \frown L) \sqsubseteq \nu\ f$

definition *mu-nu-apx-meet* :: $('a \Rightarrow 'a) \Rightarrow bool$
  where *mu-nu-apx-meet f* $\leftrightarrow$ *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* $\leftrightarrow$ *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* $\rightarrow \mu\ f + (\nu\ f \frown L) \leq \nu\ f$
  by (*metis add-least-upper-bound meet-left-upper-bound mu-below-nu*)

lemma *n-l-nu*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\rightarrow (\mu\ f + (\nu\ f \frown L)) \frown L = \nu\ f \frown L$
  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-right-upper-bound*)
  have *2*: $n(L)\ ;\ \mu\ f \leq \mu\ f + (\nu\ f \frown L) + n(\mu\ f + (\nu\ f \frown L))\ ;\ T$
    by (*metis mult-left-isotone mult-left-one mult-left-sub-dist-add-left n-sub-one order-trans*)
  thus *?thesis* **using** *1*
    by (*metis apx-def*)
qed

lemma *nu-below-mu-nu-nu-below-mu-nu-2*: *nu-below-mu-nu f* $\rightarrow$ *nu-below-mu-nu-2 f*
proof
  assume *1*: *nu-below-mu-nu f*
  have $n(L)\ ;\ \nu\ f = n(L)\ ;\ (n(L)\ ;\ \nu\ f)$
    by (*metis n-mult-idempotent mult-associative*)
  also have ... $\leq n(L)\ ;\ (\mu\ f + (\nu\ f \frown L) + n(\nu\ f)\ ;\ T)$ **using** *1*
    by (*metis mult-right-isotone nu-below-mu-nu-def*)
  also have ... $= n(L)\ ;\ (\mu\ f + (\nu\ f \frown L)) + n(L)\ ;\ n(\nu\ f)\ ;\ T$
    by (*metis mult-associative mult-left-dist-add*)
  also have ... $\leq \mu\ f + (\nu\ f \frown L) + n(L)\ ;\ n(\nu\ f)\ ;\ T$
    by (*metis add-left-isotone mult-left-isotone mult-left-one n-sub-one*)
  also have ... $= \mu\ f + (\nu\ f \frown L) + n(n(\nu\ f)\ ;\ L)\ ;\ T$
    by (*smt n-mult-commutative n-export*)
  also have ... $\leq \mu\ f + (\nu\ f \frown L) + n(\nu\ f \frown L)\ ;\ T$
    by (*metis add-right-isotone mult-left-isotone n-L-decreasing-meet-L n-isotone*)
  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

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* $\rightarrow$ *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 $n(L)$; $\nu\ f \leq \mu\ f + (\nu\ f \frown L) + n(\mu\ f + (\nu\ f \frown L))$; 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* $\wedge$ *has-greatest-fixpoint f* $\rightarrow$ (*nu-below-mu-nu f* $\leftrightarrow$ *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*)

lemma *nu-below-mu-nu-2-mu-nu-apx-nu*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\wedge$ *nu-below-mu-nu-2 f* $\rightarrow$ *mu-nu-apx-nu f*
proof
  assume *1*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\wedge$ *nu-below-mu-nu-2 f*
  hence $\mu\ f + (\nu\ f \frown L) \leq \nu\ f + L$
    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

lemma *mu-nu-apx-nu-mu-nu-apx-meet*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\wedge$ *mu-nu-apx-nu f* $\rightarrow$ *mu-nu-apx-meet f*
proof
  let *?l* $= \mu\ f + (\nu\ f \frown L)$
  assume *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\wedge$ *mu-nu-apx-nu f*
  hence *is-apx-meet* $(\mu\ f)$ $(\nu\ f)$ *?l*
    by (*smt add-apx-left-isotone add-commutative apx-meet-L is-apx-meet-def l-apx-mu less-eq-def meet-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

lemma *mu-nu-apx-meet-apx-meet-below-nu*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\wedge$ *mu-nu-apx-meet f* $\rightarrow$ *apx-meet-below-nu f*
  by (*metis apx-meet-below-nu-def l-below-nu mu-nu-apx-meet-def*)

lemma *apx-meet-below-nu-nu-below-mu-nu-2*: *apx-meet-below-nu f* $\rightarrow$ *nu-below-mu-nu-2 f*
proof −
  let *?l* $= \mu\ f + (\nu\ f \frown L)$
  have $\forall\, m\ .\ m \sqsubseteq \mu\ f \wedge m \sqsubseteq \nu\ f \wedge m \leq \nu\ f \rightarrow n(L)$; $\nu\ f \leq$ *?l* $+ n($*?l*$)$; T
  proof
    fix m
    show $m \sqsubseteq \mu\ f \wedge m \sqsubseteq \nu\ f \wedge m \leq \nu\ f \rightarrow n(L)$; $\nu\ f \leq$ *?l* $+ n($*?l*$)$; T
    proof
      assume *1*: $m \sqsubseteq \mu\ f \wedge m \sqsubseteq \nu\ f \wedge m \leq \nu\ f$

hence $m \leq ?l$
by (*smt add-commutative add-left-dist-meet add-left-upper-bound apx-def less-eq-meet meet-least-upper-bound*)
hence $m + n(m)\ ;\ T \leq ?l + n(?l)\ ;\ T$
by (*metis add-isotone mult-left-isotone n-isotone*)
thus $n(L)\ ;\ \nu\ f \leq ?l + n(?l)\ ;\ T$ **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

lemma *has-apx-least-fixpoint-xi-apx-meet*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\wedge$ *has-apx-least-fixpoint f* $\rightarrow$ *xi-apx-meet f*
proof
assume *1*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\wedge$ *has-apx-least-fixpoint f*
hence *2*: $\forall\, w\ .\ w \sqsubseteq \mu\ f \wedge w \sqsubseteq \nu\ f \rightarrow n(L)\ ;\ \xi\ f \leq w + n(w)\ ;\ T$
by (*metis apx-def mult-right-isotone order-trans xi-below-nu*)
have $\forall\, w\ .\ w \sqsubseteq \mu\ f \wedge w \sqsubseteq \nu\ f \rightarrow w \leq \xi\ f + L$ **using** *1*
by (*metis add-left-isotone apx-def mu-below-xi order-trans*)
hence $\forall\, w\ .\ w \sqsubseteq \mu\ f \wedge w \sqsubseteq \nu\ f \rightarrow w \sqsubseteq \xi\ f$ **using** *2*
by (*metis apx-def*)
hence *is-apx-meet* $(\mu\ f)\ (\nu\ f)\ (\xi\ f)$ **using** *1*
by (*smt apx-meet-char is-apx-meet-def xi-apx-below-mu xi-apx-below-nu xi-apx-meet-def*)
thus *xi-apx-meet f* **using** *1*
by (*metis apx-meet-char xi-apx-meet-def*)
qed

lemma *xi-apx-meet-apx-meet-below-nu*: *has-greatest-fixpoint f* $\wedge$ *xi-apx-meet f* $\rightarrow$ *apx-meet-below-nu f*
by (*metis apx-meet-below-nu-def xi-apx-meet-def xi-below-nu*)

lemma *apx-meet-below-nu-xi-mu-nu*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\wedge$ *isotone f* $\wedge$ *apx-isotone f* $\wedge$ *apx-meet-below-nu f* $\rightarrow$ *xi-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-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 $n(L)\ ;\ f(?l) \leq ?l + n(?l)\ ;\ T$
proof −
  have $n(L)\ ;\ f(?l) \leq n(L)\ ;\ f(\nu\ f)$ **using** *1 2*
    by (*metis apx-meet-below-nu-def isotone-def mult-right-isotone*)
  also have $\ldots = n(L)\ ;\ \nu\ f$ **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 $n(L)\ ;\ f(?l) \leq ?l + n(?l)\ ;\ T$
    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 $\forall\, y\ .\ f(y) = y \rightarrow ?l \sqsubseteq y$
proof
  fix y
  show $f(y) = y \rightarrow ?l \sqsubseteq y$
  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-right-upper-bound*)
    have $y \leq \nu\ f$ **using** *1 8*
      by (*metis greatest-fixpoint is-greatest-fixpoint-def*)
    hence $n(L)\ ;\ y \leq ?l + n(?l)\ ;\ T$ **using** *1 4 6*
      by (*smt apx-meet-below-nu-nu-below-mu-nu-2 mult-right-isotone nu-below-mu-nu-2-def order-trans*)
    thus $?l \sqsubseteq y$ **using** *9*
      by (*metis apx-def*)
  qed

qed
thus *xi-mu-nu f* **using** *1 2 7*
by (*smt apx-least-fixpoint-same has-apx-least-fixpoint-def is-apx-least-fixpoint-def xi-mu-nu-def*)
qed

lemma *xi-mu-nu-has-apx-least-fixpoint*: *xi-mu-nu f* $\rightarrow$ *has-apx-least-fixpoint f*
by (*metis xi-mu-nu-def*)

lemma *nu-below-mu-nu-xi-mu-nu*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\wedge$ *isotone f* $\wedge$ *apx-isotone f* $\wedge$ *nu-below-mu-nu f* $\rightarrow$ *xi-mu-nu f*
by (*metis apx-meet-below-nu-xi-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*)

lemma *xi-mu-nu-nu-below-mu-nu*: *has-least-fixpoint f* $\wedge$ *has-greatest-fixpoint f* $\wedge$ *xi-mu-nu f* $\rightarrow$ *nu-below-mu-nu f*
by (*metis apx-meet-below-nu-nu-below-mu-nu-2 has-apx-least-fixpoint-xi-apx-meet nu-below-mu-nu-2-nu-below-mu-nu xi-apx-meet-apx-meet-below-nu xi-mu-nu-has-apx-least-fixpoint*)

definition *xi-mu-nu-L* :: $('a \Rightarrow 'a) \Rightarrow$ *bool*
where *xi-mu-nu-L f* $\leftrightarrow$ *has-apx-least-fixpoint f* $\wedge$ $\xi\ f = \mu\ f\ +\ n(\nu\ f)\ ;\ L$

definition *nu-below-mu-nu-L* :: $('a \Rightarrow 'a) \Rightarrow$ *bool*
where *nu-below-mu-nu-L f* $\leftrightarrow$ $n(L)\ ;\ \nu\ f \leq \mu\ f\ +\ n(\nu\ f)\ ;\ T$

definition *mu-nu-apx-nu-L* :: $('a \Rightarrow 'a) \Rightarrow$ *bool*
where *mu-nu-apx-nu-L f* $\leftrightarrow$ $\mu\ f\ +\ n(\nu\ f)\ ;\ L \sqsubseteq \nu\ f$

definition *mu-nu-apx-meet-L* :: $('a \Rightarrow 'a) \Rightarrow$ *bool*
where *mu-nu-apx-meet-L f* $\leftrightarrow$ *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* $\rightarrow$ $\mu\ f\ +\ n(\nu\ f)\ ;\ L = \mu\ f\ +\ (\nu\ f \frown L)$
proof
assume *nu-below-mu-nu-L f*
hence $\nu\ f \frown L \leq (\mu\ f\ +\ n(\nu\ f)\ ;\ T) \frown L$
by (*smt meet-L-below-n-L meet-least-upper-bound meet-right-upper-bound nu-below-mu-nu-L-def order-trans*)
also have ... $\leq \mu\ f\ +\ (n(\nu\ f)\ ;\ T \frown L)$
by (*metis add-left-dist-meet add-right-upper-bound meet-right-isotone*)
also have ... $\leq \mu\ f\ +\ n(\nu\ f)\ ;\ L$
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

lemma *nu-below-mu-nu-L-nu-below-mu-nu*: *nu-below-mu-nu-L f* $\rightarrow$ *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*)

lemma *nu-below-mu-nu-L-xi-mu-nu-L*: *has-least-fixpoint f* ∧ *has-greatest-fixpoint f* ∧ *isotone f* ∧ *apx-isotone f* ∧ *nu-below-mu-nu-L f* → *xi-mu-nu-L f*
by (*metis n-equal-l nu-below-mu-nu-L-nu-below-mu-nu nu-below-mu-nu-xi-mu-nu xi-mu-nu-L-def xi-mu-nu-def*)

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*)

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*)

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 *n*(*L*) ; *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

lemma *xi-mu-nu-L-mu-nu-apx-nu-L*: *has-greatest-fixpoint f* ∧ *xi-mu-nu-L f* → *mu-nu-apx-nu-L f*
by (*metis mu-nu-apx-nu-L-def xi-apx-below-nu xi-mu-nu-L-def*)

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*)

lemma *xi-mu-nu-L-nu-below-mu-nu-L*: *has-least-fixpoint f* ∧ *has-greatest-fixpoint f* ∧ *xi-mu-nu-L f* → *nu-below-mu-nu-L f*
by (*metis mu-nu-apx-nu-L-nu-below-mu-nu-L xi-mu-nu-L-mu-nu-apx-nu-L*)

end

class *itering-apx* = *itering-T* + *idl-semiring-lattice-apx*

begin

lemma *circ-L*: *L*° = *L* + *1*
by (*metis add-commutative mult-top-circ n-L-top-L*)

lemma *n-circ-import*: *n*(*y*) ; *x* ≤ *x* ; *n*(*y*) → *n*(*y*) ; *x*° = *n*(*y*) ; (*n*(*y*) ; *x*)°

by (*metis circ-import n-mult-idempotent n-sub-one order-refl*)

lemma *circ-apx-isotone*: $x \sqsubseteq y \rightarrow x^\circ \sqsubseteq y^\circ$
proof
assume $x \sqsubseteq y$
hence *1*: $x \leq y + L \wedge n(L) \ ; \ y \leq x + n(x) \ ; \ T$
by (*metis apx-def*)
have $n(L) \ ; \ y^\circ \leq (n(L) \ ; \ y)^\circ$
by (*smt circ-reflexive circ-transitive-equal n-sub-one n-circ-import n-nL-semi-commute mult-left-isotone order-trans*)
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*: $n(L) \ ; \ y^\circ \leq x^\circ + n(x^\circ) \ ; \ T$
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 *sdloa-apx* = *sdloa-T* + *idl-semiring-lattice-apx* + *Omega* +
assumes *Omega-def*: $x^\Omega = n(x^\omega) \ ; \ L + x^*$

begin

lemma *n-omega-export*: $n(y) \ ; \ x \leq x \ ; \ n(y) \rightarrow n(y) \ ; \ x^\omega = (n(y) \ ; \ x)^\omega$
by (*metis mult-associative n-mult-idempotent omega-import omega-slide order-refl*)

lemma *L-mult-star*: $L \ ; \ x^* = L$
by (*metis less-eq-def mult-associative n-L-below-L star.circ-back-loop-fixpoint*)

lemma *mult-L-star*: $(x \ ; \ L)^* = 1 + x \ ; \ L$
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*)

lemma *mult-L-add-star*: $(x\ ;\ L + y)^* = y^*\ +\ y^*\ ;\ x\ ;\ L$
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*: $(x\ ;\ L + y)^\omega \leq y^\omega\ +\ y^*\ ;\ x\ ;\ L$
proof −
have $(x\ ;\ L + y)^\omega \leq y^*\ ;\ x\ ;\ L + (y^*\ ;\ x\ ;\ L)^*\ ;\ y^\omega$
by (*metis add-commutative mult-associative omega-decompose add-left-isotone mult-L-omega-below*)
also have $... \leq y^\omega\ +\ y^*\ ;\ x\ ;\ L$
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*: $x \leq y \rightarrow x^\Omega \leq y^\Omega$
by (*metis Omega-def add-isotone mult-left-isotone n-isotone omega-isotone star-isotone*)

lemma *n-star-below-Omega*: $x^* \leq x^\Omega$
by (*metis add-right-upper-bound Omega-def*)

lemma *mult-L-star-mult-below*: $(x\ ;\ L)^*\ ;\ y \leq y\ +\ x\ ;\ L$
by (*metis add-right-isotone mult-associative mult-right-isotone n-L-below-L star-left-induct*)

end

sublocale *sdloa-apx* < *star*!: *itering-apx* **where** *circ* = *star* **..**

class *sdloa-apx-extra* = *sdloa-apx* + *idl-semiring-lattice-apx* +
assumes *n-split-omega-mult*: $n(L)\ ;\ x^\omega \leq x^*\ ;\ n(x^\omega)\ ;\ T$
assumes *tarski*: $x\ ;\ L \leq x\ ;\ L\ ;\ x\ ;\ L$

begin

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

lemma *mult-L-add-omega*: $(x\ ;\ L + y)^\omega = y^\omega\ +\ y^*\ ;\ x\ ;\ L$
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

lemma *tarski-mult-top-idempotent*: $x\ ;\ L = x\ ;\ L\ ;\ x\ ;\ L$
by (*metis mult-L-omega mult-associative omega-unfold*)

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

lemma *n-split-omega-add-zero*: $n(L)\ ;\ x^\omega \leq x^*\ ;\ 0 + n(x^\omega)\ ;\ T$
proof −
have $n(x^\omega)\ ;\ T + x\ ;\ (x^*\ ;\ 0 + n(x^\omega)\ ;\ T) = n(x^\omega)\ ;\ T + x\ ;\ x^*\ ;\ 0 + x\ ;\ n(x^\omega)\ ;\ T$
by (*metis add-associative mult-associative mult-left-dist-add*)
also have $... \leq n(x^\omega)\ ;\ T + x\ ;\ x^*\ ;\ 0 + x\ ;\ 0 + n(x^\omega)\ ;\ T$
by (*metis add-associative add-right-isotone n-n-top-split-n-top omega-unfold*)
also have $... = x\ ;\ x^*\ ;\ 0 + n(x^\omega)\ ;\ T$
by (*smt add-associative add-commutative add-left-top add-right-zero mult-associative mult-left-dist-add*)
also have $... \leq x^*\ ;\ 0 + n(x^\omega)\ ;\ T$
by (*metis add-left-isotone mult-left-isotone star.left-plus-below-circ*)
finally have $x^*\ ;\ n(x^\omega)\ ;\ T \leq x^*\ ;\ 0 + n(x^\omega)\ ;\ T$
by (*metis mult-associative star-left-induct*)
thus *?thesis*
by (*metis n-split-omega-mult order-trans*)
qed

lemma *n-split-omega-add*: $n(L)\ ;\ x^\omega \leq x^* + n(x^\omega)\ ;\ T$
by (*metis add-left-isotone n-split-omega-add-zero order-trans zero-right-mult-decreasing*)

lemma *n-dist-omega-star*: $n(y^\omega + y^*\ ;\ z) = n(y^\omega) + n(y^*\ ;\ z)$
proof −
have $n(y^\omega + y^*\ ;\ z) \leq n(n(L)\ ;\ y^\omega + y^*\ ;\ z)$
by (*smt mult-associative mult-left-dist-add mult-left-isotone mult-left-one n-export n-mult-commutative n-n-nL n-sub-one*)
also have $... \leq n(y^*\ ;\ 0 + n(y^\omega)\ ;\ T + y^*\ ;\ z)$
by (*metis add-commutative add-right-isotone n-isotone n-split-omega-add-zero*)
also have $... = n(y^\omega) + n(y^*\ ;\ z)$
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*: $(x\ ;\ L + y)^\Omega \leq n(y^\omega)\ ;\ L + y^* + y^*\ ;\ x\ ;\ L$

proof −
have $(x\ ;\ L + y)^{\Omega} \leq n(y^{\omega} + y^{*}\ ;\ x\ ;\ L)\ ;\ L + (x\ ;\ L + y)^{*}$
by (*metis add-left-isotone mult-L-add-omega-below mult-left-isotone n-isotone Omega-def*)
also have $\ldots = n(y^{\omega})\ ;\ L + n(y^{*}\ ;\ x\ ;\ L)\ ;\ L + (x\ ;\ L + y)^{*}$
by (*metis mult-associative mult-right-dist-add n-dist-omega-star*)
also have $\ldots \leq n(y^{\omega})\ ;\ L + y^{*} + y^{*}\ ;\ x\ ;\ L$
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*: $n(y\ ;\ x^{\omega})\ ;\ L \leq y\ ;\ x^{*}\ ;\ 0 + y\ ;\ n(x^{\omega})\ ;\ L$
proof −
have $n(y\ ;\ x^{\omega})\ ;\ L \leq n(L)\ ;\ y\ ;\ x^{\omega} \frown L$
by (*metis n-L-decreasing-meet-L n-export n-mult-commutative n-n-nL mult-associative*)
also have $\ldots \leq y\ ;\ n(L)\ ;\ x^{\omega} \frown L$
by (*smt meet-left-isotone meet-right-divisibility mult-right-sub-dist-meet-right n-nL-semi-commute*)
also have $\ldots \leq y\ ;\ (x^{*}\ ;\ 0 + n(x^{\omega})\ ;\ T) \frown L$
by (*metis meet-commutative meet-right-isotone mult-associative mult-right-isotone n-split-omega-add-zero*)
also have $\ldots = (y\ ;\ x^{*}\ ;\ 0 \frown L) + (y\ ;\ n(x^{\omega})\ ;\ T \frown L)$
by (*metis meet-commutative meet-left-dist-add mult-associative mult-left-dist-add*)
also have $\ldots \leq (y\ ;\ x^{*}\ ;\ 0 \frown L) + y\ ;\ n(x^{\omega})\ ;\ L$
by (*metis add-right-isotone n-vector-meet-L*)
also have $\ldots \leq y\ ;\ x^{*}\ ;\ 0 + y\ ;\ n(x^{\omega})\ ;\ L$
by (*metis add-left-isotone meet-left-upper-bound*)
finally show *?thesis*
by *metis*
qed

lemma *n-mult-omega-L-star-zero*: $y\ ;\ x^{*}\ ;\ 0 + n(y\ ;\ x^{\omega})\ ;\ L = y\ ;\ x^{*}\ ;\ 0 + y\ ;\ n(x^{\omega})\ ;\ L$
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

lemma *n-mult-omega-L-star*: $y\ ;\ x^{*} + n(y\ ;\ x^{\omega})\ ;\ L = y\ ;\ x^{*} + y\ ;\ n(x^{\omega})\ ;\ L$
by (*metis zero-right-mult-decreasing n-mult-omega-L-star-zero add-relative-same-increasing*)

lemma *n-mult-omega-L-below*: $n(y\ ;\ x^{\omega})\ ;\ L \leq y\ ;\ x^{*} + y\ ;\ n(x^{\omega})\ ;\ L$
by (*metis add-right-upper-bound n-mult-omega-L-star*)

lemma *n-omega-L-below-zero*: $n(x^{\omega})\ ;\ L \leq x\ ;\ x^{*}\ ;\ 0 + x\ ;\ n(x^{\omega})\ ;\ L$
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*: $n(x^{\omega})\ ;\ L \leq x^{*} + x\ ;\ n(x^{\omega})\ ;\ L$
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*: $x\ ;\ x^*\ ;\ 0 + n(x^\omega)\ ;\ L = x\ ;\ x^*\ ;\ 0 + x\ ;\ n(x^\omega)\ ;\ L$
by (*metis n-mult-omega-L-star-zero omega-unfold*)

lemma *n-omega-L-star*: $x^* + n(x^\omega)\ ;\ L = x^* + x\ ;\ n(x^\omega)\ ;\ L$
by (*metis star.circ-mult-upper-bound star.left-plus-below-circ zero-least n-omega-L-star-zero add-relative-same-increasing*)

lemma *n-omega-L-star-zero-star*: $x^*\ ;\ 0 + n(x^\omega)\ ;\ L = x^*\ ;\ 0 + x^*\ ;\ n(x^\omega)\ ;\ L$
by (*metis n-mult-omega-L-star-zero star-mult-omega mult-associative star.circ-transitive-equal*)

lemma *n-omega-L-star-star*: $x^* + n(x^\omega)\ ;\ L = x^* + x^*\ ;\ n(x^\omega)\ ;\ L$
by (*metis zero-right-mult-decreasing n-omega-L-star-zero-star add-relative-same-increasing*)

lemma *n-Omega-left-unfold*: $1 + x\ ;\ x^\Omega = x^\Omega$
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*: $(x\ ;\ y)^\Omega\ ;\ x \leq x\ ;\ (y\ ;\ x)^\Omega$
proof −
have $(x\ ;\ y)^\Omega\ ;\ x \leq x\ ;\ y\ ;\ n((x\ ;\ y)^\omega)\ ;\ L + (x\ ;\ y)^*\ ;\ x$
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 $\ldots \leq x\ ;\ (y\ ;\ 0 + n(y\ ;\ (x\ ;\ y)^\omega)\ ;\ L) + (x\ ;\ y)^*\ ;\ x$
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 $\ldots = x\ ;\ (y\ ;\ x)^\Omega$
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*: $(x + y)^\Omega = x^\Omega\ ;\ (y\ ;\ x^\Omega)^\Omega$
proof −
have *1*: $(x + y)^\Omega = n((x^*\ ;\ y)^\omega)\ ;\ L + n((x^*\ ;\ y)^*\ ;\ x^\omega)\ ;\ L + (x^*\ ;\ y)^*\ ;\ x^*$
by (*smt Omega-def mult-right-dist-add n-dist-omega-star omega-decompose star.circ-add*)
have $n((x^*\ ;\ y)^\omega)\ ;\ L \leq (x^*\ ;\ y)^* + x^*\ ;\ (y\ ;\ n((x^*\ ;\ y)^\omega)\ ;\ L)$
by (*metis n-omega-L-below mult-associative*)
also have $\ldots \leq (x^*\ ;\ y)^* + x^*\ ;\ y\ ;\ 0 + x^*\ ;\ n((y\ ;\ x^*)^\omega)\ ;\ L$
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 $\ldots = (x^*\ ;\ y)^* + x^*\ ;\ n((y\ ;\ x^*)^\omega)\ ;\ L$
by (*metis add-commutative less-eq-def star.circ-sub-dist-1 zero-right-mult-decreasing*)
also have $\ldots \leq x^*\ ;\ (y\ ;\ x^*)^* + x^*\ ;\ n((y\ ;\ x^*)^\omega)\ ;\ L$
by (*metis add-left-isotone mult-right-isotone star.circ-increasing star.circ-isotone star-decompose-3*)
also have $\ldots \leq x^*\ ;\ (y\ ;\ x^\Omega)^\Omega$
by (*metis Omega-def add-commutative mult-associative mult-left-dist-add mult-right-isotone n-Omega-isotone n-star-below-Omega*)
also have $\ldots \leq x^\Omega\ ;\ (y\ ;\ x^\Omega)^\Omega$
by (*metis n-star-below-Omega mult-left-isotone*)
finally have *2*: $n((x^*\ ;\ y)^\omega)\ ;\ L \leq x^\Omega\ ;\ (y\ ;\ x^\Omega)^\Omega$
by *metis*
have $n((x^*\ ;\ y)^*\ ;\ x^\omega)\ ;\ L \leq n(x^\omega)\ ;\ L + x^*\ ;\ (y\ ;\ x^*)^* + x^*\ ;\ (y\ ;\ x^*)^*\ ;\ y\ ;\ n(x^\omega)\ ;\ L$
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 ... = $n(x^\omega)$; L ; $(y$; $x^\Omega)^*$ + x^* ; $(y$; $x^\Omega)^*$
by (*smt Omega-def add-associative mult-L-add-star mult-associative mult-left-dist-add L-mult-star*)
also have ... ≤ x^Ω ; $(y$; $x^\Omega)^\Omega$
by (*metis mult-right-dist-add Omega-def n-star-below-Omega mult-right-isotone*)
finally have *3*: $n((x^*$; $y)^*$; $x^\omega)$; $L \leq x^\Omega$; $(y$; $x^\Omega)^\Omega$
by *metis*
have $(x^*$; $y)^*$; $x^* \leq x^\Omega$; $(y$; $x^\Omega)^\Omega$
by (*metis star-slide mult-isotone mult-right-isotone n-star-below-Omega order-trans star-isotone*)
hence *4*: $(x + y)^\Omega \leq x^\Omega$; $(y$; $x^\Omega)^\Omega$ **using** *1 2 3*
by (*metis add-least-upper-bound*)
have *5*: x^Ω ; $(y$; $x^\Omega)^\Omega \leq n(x^\omega)$; $L + x^*$; $n((y$; $x^\Omega)^\omega)$; $L + x^*$; $(y$; $x^\Omega)^*$
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 $n(x^\omega)$; $L \leq n((x^*$; $y)^*$; $x^\omega)$; L
by (*metis add-commutative add-left-upper-bound mult-left-isotone n-isotone star.circ-loop-fixpoint*)
hence *6*: $n(x^\omega)$; $L \leq (x + y)^\Omega$ **using** *1*
by (*metis Omega-def add-left-upper-bound n-Omega-isotone order-trans*)
have x^* ; $n((y$; $x^\Omega)^\omega)$; $L \leq x^*$; $n((y$; $x^*)^\omega + (y$; $x^*)^*$; y ; $n(x^\omega)$; $L)$; 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 ... ≤ x^* ; $0 + n(x^*$; $((y$; $x^*)^\omega + (y$; $x^*)^*$; y ; $n(x^\omega)$; $L))$; L
by (*metis n-n-L-split-n-L*)
also have ... ≤ $x^* + n((x^*$; $y)^\omega + x^*$; $(y$; $x^*)^*$; y ; $n(x^\omega)$; $L)$; L
by (*smt add-left-isotone mult-associative mult-left-dist-add omega-slide zero-right-mult-decreasing*)
also have ... ≤ $x^* + n((x^*$; $y)^\omega + (x^*$; $y)^*$; $n(x^\omega)$; $L)$; L
by (*smt add-right-divisibility add-right-isotone mult-left-isotone n-isotone star.circ-mult*)
also have ... ≤ $x^* + n((x + y)^\omega)$; L
by (*metis add-right-isotone mult-associative mult-left-isotone mult-right-isotone n-L-decreasing n-isotone omega-decompose*)
also have ... ≤ $(x + y)^\Omega$
by (*metis add-left-isotone star.circ-sub-dist Omega-def add-commutative*)
finally have *7*: x^* ; $n((y$; $x^\Omega)^\omega)$; $L \leq (x + y)^\Omega$
by *metis*
have x^* ; $(y$; $x^\Omega)^* \leq (x^*$; $y)^*$; $x^* + (x^*$; $y)^*$; $n(x^\omega)$; 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 ... ≤ $(x^*$; $y)^*$; $x^* + n((x^*$; $y)^*$; $x^\omega)$; L
by (*metis add-associative add-right-isotone add-right-zero mult-left-dist-add n-n-L-split-n-L*)
also have ... ≤ $(x + y)^\Omega$
by (*smt Omega-def add-commutative add-right-isotone mult-left-isotone n-right-upper-bound omega-decompose star.circ-add*)
finally have $n(x^\omega)$; $L + x^*$; $n((y$; $x^\Omega)^\omega)$; $L + x^*$; $(y$; $x^\Omega)^* \leq (x + y)^\Omega$ **using** *6 7*
by (*metis add-least-upper-bound*)
hence x^Ω ; $(y$; $x^\Omega)^\Omega \leq (x + y)^\Omega$ **using** *5*
by (*smt order-trans*)
thus *?thesis* **using** *4*
by (*metis antisym*)
qed

end

sublocale *sdloa-apx-extra* < *nL-omega*!: *idl-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 *sdloa-apx-extra*

begin

lemma *omega-apx-isotone*: $x \sqsubseteq y \rightarrow x^\omega \sqsubseteq y^\omega$
proof
assume $x \sqsubseteq y$
hence *1*: $x \leq y + L \wedge n(L) \ ; \ y \leq x + n(x) \ ; \ T$
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*: $x^* \ ; \ n(x) \ ; \ T \leq x^\omega + n(x^\omega) \ ; \ T$
by (*metis mult-associative star-left-induct*)
have $n(L) \ ; \ y^\omega = (n(L) \ ; \ y)^\omega$
by (*metis n-omega-export n-nL-semi-commute*)
also have $... \leq (x + n(x) \ ; \ T)^\omega$ **using** *1*
by (*metis omega-isotone*)
also have $... = (x^* \ ; \ n(x) \ ; \ T)^\omega + (x^* \ ; \ n(x) \ ; \ T)^* \ ; \ x^\omega$
by (*metis mult-associative omega-decompose*)
also have $... \leq x^* \ ; \ n(x) \ ; \ T + (x^* \ ; \ n(x) \ ; \ T)^* \ ; \ x^\omega$
by (*metis add-left-isotone mult-top-omega*)
also have $... = x^* \ ; \ n(x) \ ; \ T + (1 + x^* \ ; \ n(x) \ ; \ T \ ; \ (x^* \ ; \ n(x) \ ; \ T)^*) \ ; \ x^\omega$
by (*metis mult-associative star.circ-left-top star.mult-top-circ*)
also have $... \leq x^\omega + x^* \ ; \ n(x) \ ; \ T$
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*: $n(L) \ ; \ y^\omega \leq x^\omega + n(x^\omega) \ ; \ T$
by *metis*
have $x^\omega \leq (y + L)^\omega$ **using** *1*
by (*metis omega-isotone*)
also have $... = (y^* \ ; \ L)^\omega + (y^* \ ; \ L)^* \ ; \ y^\omega$
by (*metis omega-decompose*)
also have $... = y^* \ ; \ L \ ; \ (y^* \ ; \ L)^\omega + (y^* \ ; \ L)^* \ ; \ y^\omega$
by (*metis omega-unfold*)
also have $... \leq y^* \ ; \ L + (y^* \ ; \ L)^* \ ; \ y^\omega$
by (*metis add-left-isotone n-L-below-L mult-associative mult-right-isotone*)

also have ... $= y^* \mathbin{;} L + (1 + y^* \mathbin{;} L \mathbin{;} (y^* \mathbin{;} L)^*) \mathbin{;} y^\omega$
by (*metis star.circ-left-unfold*)
also have ... $\leq y^* \mathbin{;} L + y^\omega$
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 ... $\leq y^* \mathbin{;} 0 + L + y^\omega$
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*: $x \sqsubseteq y \to n(x^\omega) \mathbin{;} L + x^* \mathbin{;} z \sqsubseteq n(y^\omega) \mathbin{;} L + y^* \mathbin{;} z$
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*: $x \sqsubseteq y \to (x^\omega \frown L) + x^* \mathbin{;} z \sqsubseteq (y^\omega \frown L) + y^* \mathbin{;} z$
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*: $y \sqsubseteq z \to n(x^\omega) \mathbin{;} L + x^* \mathbin{;} y \sqsubseteq n(x^\omega) \mathbin{;} L + x^* \mathbin{;} z$
by (*metis add-apx-right-isotone mult-apx-right-isotone*)

lemma *combined-apx-right-isotone-2*: $y \sqsubseteq z \to (x^\omega \frown L) + x^* \mathbin{;} y \sqsubseteq (x^\omega \frown L) + x^* \mathbin{;} z$
by (*metis add-apx-right-isotone mult-apx-right-isotone*)

lemma *combined-apx-isotone*: $x \sqsubseteq y \wedge w \sqsubseteq z \to n(x^\omega) \mathbin{;} L + x^* \mathbin{;} w \sqsubseteq n(y^\omega) \mathbin{;} L + y^* \mathbin{;} z$
by (*metis add-apx-isotone mult-apx-isotone omega-apx-isotone star.circ-apx-isotone n-L-apx-isotone*)

lemma *combined-apx-isotone-2*: $x \sqsubseteq y \wedge w \sqsubseteq z \to (x^\omega \frown L) + x^* \mathbin{;} w \sqsubseteq (y^\omega \frown L) + y^* \mathbin{;} z$
by (*metis add-apx-isotone mult-apx-isotone omega-apx-isotone star.circ-apx-isotone meet-L-apx-isotone*)

lemma *n-split-nu-mu*: $n(L) \mathbin{;} (y^\omega + y^* \mathbin{;} z) \leq y^* \mathbin{;} z + n(y^\omega + y^* \mathbin{;} z) \mathbin{;} T$
proof −
have $n(L) \mathbin{;} (y^\omega + y^* \mathbin{;} z) \leq n(L) \mathbin{;} y^\omega + y^* \mathbin{;} z$
by (*metis add-right-isotone mult-left-dist-add mult-left-isotone mult-left-one n-sub-one*)
also have ... $\leq y^* \mathbin{;} 0 + n(y^\omega) \mathbin{;} T + y^* \mathbin{;} z$
by (*metis add-commutative add-right-isotone n-split-omega-add-zero*)
also have ... $\leq y^* \mathbin{;} z + n(y^\omega + y^* \mathbin{;} z) \mathbin{;} T$
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*: $n(L) \mathbin{;} (y^\omega + y^* \mathbin{;} z) \leq y^* \mathbin{;} z + ((y^\omega + y^* \mathbin{;} z) \frown L) + n(y^\omega + y^* \mathbin{;} z) \mathbin{;} T$
by (*smt add-left-isotone add-left-upper-bound add-right-isotone add-right-upper-bound n-split-nu-mu order-trans*)

lemma *loop-exists*: $n(L) \mathbin{;} \nu\ (\lambda x \mathbin{.} y \mathbin{;} x + z) \leq \mu\ (\lambda x \mathbin{.} y \mathbin{;} x + z) + n(\nu\ (\lambda x \mathbin{.} y \mathbin{;} x + z)) \mathbin{;} T$
by (*metis n-split-nu-mu omega-loop-nu star-loop-mu*)

lemma *loop-exists-2*: $n(L)\ ;\ \nu\ (\lambda x\ .\ y\ ;\ x + z) \leq \mu\ (\lambda x\ .\ y\ ;\ x + z) + (\nu\ (\lambda x\ .\ y\ ;\ x + z) \frown L) + n(\nu\ (\lambda x\ .\ y\ ;\ x + z))\ ;\ T$
by (*metis n-split-nu-mu-2 omega-loop-nu star-loop-mu*)

lemma *loop-apx-least-fixpoint*: *is-apx-least-fixpoint* $(\lambda x\ .\ y\ ;\ x + z)\ (\mu\ (\lambda x\ .\ y\ ;\ x + z) + n(\nu\ (\lambda x\ .\ y\ ;\ x + z))\ ;\ L)$
proof −
have *xi-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-xi-mu-nu-L*)
thus *?thesis*
by (*smt apx-least-fixpoint-char xi-mu-nu-L-def*)
qed

lemma *loop-apx-least-fixpoint-2*: *is-apx-least-fixpoint* $(\lambda x\ .\ y\ ;\ x + z)\ (\mu\ (\lambda x\ .\ y\ ;\ x + z) + (\nu\ (\lambda x\ .\ y\ ;\ x + z) \frown L))$
proof −
have *xi-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-xi-mu-nu*)
thus *?thesis*
by (*smt apx-least-fixpoint-char xi-mu-nu-def*)
qed

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

lemma *loop-semantics*: $\xi\ (\lambda x\ .\ y\ ;\ x + z) = \mu\ (\lambda x\ .\ y\ ;\ x + z) + n(\nu\ (\lambda x\ .\ y\ ;\ x + z))\ ;\ L$
by (*metis apx-least-fixpoint-char loop-apx-least-fixpoint*)

lemma *loop-semantics-2*: $\xi\ (\lambda x\ .\ y\ ;\ x + z) = \mu\ (\lambda x\ .\ y\ ;\ x + z) + (\nu\ (\lambda x\ .\ y\ ;\ x + z) \frown L)$
by (*metis apx-least-fixpoint-char loop-apx-least-fixpoint-2*)

lemma *loop-semantics-xi-mu-nu*: $\xi\ (\lambda x\ .\ y\ ;\ x + z) = n(y^\omega)\ ;\ L + y^*\ ;\ z$
proof −
have $\xi\ (\lambda x\ .\ y\ ;\ x + z) = y^*\ ;\ z + n(y^\omega + y^*\ ;\ z)\ ;\ L$
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

lemma *loop-semantics-xi-mu-nu-2*: $\xi\ (\lambda x\ .\ y\ ;\ x + z) = (y^\omega \frown L) + y^*\ ;\ z$
proof −
have $\xi\ (\lambda x\ .\ y\ ;\ x + z) = y^*\ ;\ z + ((y^\omega + y^*\ ;\ z) \frown L)$
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

lemma *loop-semantics-apx-left-isotone*: $w \sqsubseteq y \rightarrow \xi\ (\lambda x\ .\ w\ ;\ x + z) \sqsubseteq \xi\ (\lambda x\ .\ y\ ;\ x + z)$
by (*metis loop-semantics-xi-mu-nu combined-apx-left-isotone*)

lemma *loop-semantics-apx-right-isotone*: $w \sqsubseteq z \to \xi\ (\lambda x\ .\ y\ ;\ x + w) \sqsubseteq \xi\ (\lambda x\ .\ y\ ;\ x + z)$
by (*metis loop-semantics-xi-mu-nu combined-apx-right-isotone*)

lemma *loop-semantics-apx-isotone*: $v \sqsubseteq y \wedge w \sqsubseteq z \to \xi\ (\lambda x\ .\ v\ ;\ x + w) \sqsubseteq \xi\ (\lambda x\ .\ y\ ;\ x + z)$
by (*metis loop-semantics-xi-mu-nu combined-apx-isotone*)

end

context *boolean-algebra*

begin

notation
inf (**infixl** $\sqcap$ *70*) **and**
sup (**infixl** $\sqcup$ *65*) **and**
uminus (- $'$ [*80*] *80*)

lemma *shunting-1*: $x \leq y \leftrightarrow x \sqcap y\ ' = bot$
apply *rule*
apply (*smt compl-inf-bot inf-absorb1 inf-bot-right inf-commute inf-left-commute*)
apply (*metis inf-commute inf-sup-distrib1 inf-top-left le-iff-inf sup-commute sup-bot-left sup-compl-top*)
done

lemma *shunting-2*: $x \leq y \leftrightarrow x\ ' \sqcup y = top$
by (*metis compl-bot-eq compl-inf double-compl shunting-1*)

definition *conjugate* :: $('a \Rightarrow 'a) \Rightarrow ('a \Rightarrow 'a) \Rightarrow bool$
where *conjugate f g* $\leftrightarrow$ $(\forall x\ y\ .\ f\ x \sqcap y = bot \leftrightarrow x \sqcap g\ y = bot)$

lemma *conjugate-unique*: *conjugate f g* $\wedge$ *conjugate f h* $\to g = h$
proof
assume *conjugate f g* $\wedge$ *conjugate f h*
hence $\forall x\ y\ .\ g\ y \leq x\ ' \leftrightarrow h\ y \leq x\ '$
by (*smt double-compl inf-commute shunting-1 conjugate-def*)
hence $\forall y\ .\ g\ y = h\ y$
by (*metis double-compl eq-iff*)

```
  thus g = h
    by (metis lifted-antisymmetric lifted-less-eq-def order-refl)
qed

lemma conjugate-symmetric: conjugate f g → conjugate g f
  by (smt conjugate-def inf-commute)



definition additive :: ('a ⇒ 'a) ⇒ bool
  where additive f ↔ (∀ x y . f (x ⊔ y) = f x ⊔ f y)



lemma additive-isotone: additive f → isotone f
  by (metis additive-def isotone-def le-iff-sup)



lemma conjugate-additive: conjugate f g → additive f
proof
  assume 1: conjugate f g
  have 2: ∀ x y z . f (x ⊔ y) ≤ z ↔ f x ≤ z ∧ f y ≤ z
  proof
    fix x
    show ∀ y z . f (x ⊔ y) ≤ z ↔ f x ≤ z ∧ f y ≤ z
    proof
      fix y
      show ∀ z . f (x ⊔ y) ≤ z ↔ f x ≤ z ∧ f y ≤ z
      proof
        fix z
        have (f (x ⊔ y) ≤ z) = (f (x ⊔ y) ⊓ z ' = bot)
          by (metis shunting-1)
        also have ... = ((x ⊔ y) ⊓ g(z ') = bot) using 1
          by (metis conjugate-def)
        also have ... = (x ⊔ y ≤ (g(z '))')
          by (metis double-compl shunting-1)
        also have ... = (x ≤ (g(z '))' ∧ y ≤ (g(z '))')
          by (metis le-sup-iff)
        also have ... = (x ⊓ g(z ') = bot ∧ y ⊓ g(z ') = bot)
          by (metis double-compl shunting-1)
        also have ... = (f x ⊓ z ' = bot ∧ f y ⊓ z ' = bot) using 1
          by (metis conjugate-def)
        also have ... = (f x ≤ z ∧ f y ≤ z)
          by (metis shunting-1)
        finally show f (x ⊔ y) ≤ z ↔ f x ≤ z ∧ f y ≤ z
          by metis
```

```
    qed
  qed
 qed
 have ∀ x y . f (x ⊔ y) = f x ⊔ f y
 proof
  fix x
  show ∀ y . f (x ⊔ y) = f x ⊔ f y
  proof
   fix y
   have f(x ⊔ y) ≤ f(x) ⊔ f(y) using 2
    by (metis sup-ge1 sup-ge2)
   thus f (x ⊔ y) = f x ⊔ f y using 2
    by (metis le-supI order-refl antisym)
  qed
 qed
 thus additive f
  by (metis additive-def)
qed

lemma conjugate-isotone: conjugate f g → isotone f
 by (metis additive-isotone conjugate-additive)



lemma conjugate-char-1: conjugate f g ↔ (∀ x y . f(x ⊓ (g y)′) ≤ f x ⊓ y ′ ∧ g(y ⊓ (f x)′) ≤ g y ⊓ x ′)
proof
 assume 1: conjugate f g
 show ∀ x y . f(x ⊓ (g y)′) ≤ f x ⊓ y ′ ∧ g(y ⊓ (f x)′) ≤ g y ⊓ x ′
 proof
  fix x
  show ∀ y . f(x ⊓ (g y)′) ≤ f x ⊓ y ′ ∧ g(y ⊓ (f x)′) ≤ g y ⊓ x ′
  proof
   fix y
   have f(x ⊓ (g y)′) ≤ y ′ using 1
    by (smt compl-inf-bot conjugate-def double-compl inf-assoc inf-bot-right shunting-1)
   hence 2: f(x ⊓ (g y)′) ≤ f x ⊓ y ′ using 1
    by (metis conjugate-isotone inf-le1 isotone-def le-inf-iff)
   have g(y ⊓ (f x)′) ≤ x ′ using 1
    by (smt compl-inf-bot conjugate-def double-compl inf-assoc inf-bot-right shunting-1 inf-commute)
   hence g(y ⊓ (f x)′) ≤ g y ⊓ x ′ using 1
    by (metis conjugate-isotone inf-le1 isotone-def le-inf-iff conjugate-symmetric)
   thus f(x ⊓ (g y)′) ≤ f x ⊓ y ′ ∧ g(y ⊓ (f x)′) ≤ g y ⊓ x ′ using 2
    by metis
  qed
 qed
next
 assume ∀ x y . f(x ⊓ (g y)′) ≤ f x ⊓ y ′ ∧ g(y ⊓ (f x)′) ≤ g y ⊓ x ′
```

```
  thus conjugate f g
    by (smt conjugate-def double-compl inf-commute inf-le2 le-iff-inf shunting-1)
qed

lemma conjugate-char-2: conjugate f g ↔ f bot = bot ∧ g bot = bot ∧ (∀ x y . f x ⊓ y ≤ f(x ⊓ g y) ∧ g y ⊓ x ≤ g(y ⊓ f x))
proof
  assume 1: conjugate f g
  show f bot = bot ∧ g bot = bot ∧ (∀ x y . f x ⊓ y ≤ f(x ⊓ g y) ∧ g y ⊓ x ≤ g(y ⊓ f x))
  proof
    show f bot = bot using 1
      by (metis conjugate-def inf-idem inf-bot-left)
  next
    show g bot = bot ∧ (∀ x y . f x ⊓ y ≤ f(x ⊓ g y) ∧ g y ⊓ x ≤ g(y ⊓ f x))
    proof
      show g bot = bot using 1
        by (metis conjugate-def inf-idem inf-bot-right)
    next
      show ∀ x y . f x ⊓ y ≤ f(x ⊓ g y) ∧ g y ⊓ x ≤ g(y ⊓ f x)
      proof
        fix x
        show ∀ y . f x ⊓ y ≤ f(x ⊓ g y) ∧ g y ⊓ x ≤ g(y ⊓ f x)
        proof
          fix y
          show f x ⊓ y ≤ f(x ⊓ g y) ∧ g y ⊓ x ≤ g(y ⊓ f x)
          proof
            have f x ⊓ y = (f(x ⊓ g y) ⊔ f(x ⊓ (g y)′)) ⊓ y using 1
              by (metis additive-def conjugate-additive inf-sup-distrib1 inf-top-right sup-compl-top)
            also have ... ≤ (f(x ⊓ g y) ⊔ (f x ⊓ y ′)) ⊓ y using 1
              by (metis conjugate-char-1 inf-mono order-refl sup-mono)
            also have ... ≤ f(x ⊓ g y)
              by (smt inf-idem inf-assoc inf-commute inf-compl-bot inf-sup-distrib1 le-iff-inf sup-commute sup-bot-left)
            finally show f x ⊓ y ≤ f(x ⊓ g y)
              by metis
          next
            have g y ⊓ x = (g(y ⊓ f x) ⊔ g(y ⊓ (f x)′)) ⊓ x using 1
              by (metis additive-def conjugate-additive conjugate-symmetric inf-sup-distrib1 inf-top-right sup-compl-top)
            also have ... ≤ (g(y ⊓ f x) ⊔ (g y ⊓ x ′)) ⊓ x using 1
              by (metis conjugate-char-1 inf-mono order-refl sup-mono)
            also have ... ≤ g(y ⊓ f x)
              by (smt inf-idem inf-assoc inf-commute inf-compl-bot inf-sup-distrib1 le-iff-inf sup-commute sup-bot-left)
            finally show g y ⊓ x ≤ g(y ⊓ f x)
              by metis
          qed
        qed
      qed
    qed
  qed
```

next
  assume $f\ bot = bot \wedge g\ bot = bot \wedge (\forall x\ y\ .\ f\ x \sqcap y \leq f(x \sqcap g\ y) \wedge g\ y \sqcap x \leq g(y \sqcap f\ x))$
  thus *conjugate f g*
    by (*smt conjugate-def inf-commute le-bot*)
qed

end

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

class *relation-algebra* = *boolean-algebra* + *mult* + *one* + *conv* +
  assumes *comp-associative* : $(x\ ;\ y)\ ;\ z = x\ ;\ (y\ ;\ z)$
  assumes *comp-right-dist-sup*: $(x \sqcup y)\ ;\ z = (x\ ;\ z) \sqcup (y\ ;\ z)$
  assumes *comp-right-one* : $x\ ;\ 1 = x$
  assumes *conv-involutive* : $x^{\smile\smile} = x$
  assumes *conv-dist-sup* : $(x \sqcup y)^\smile = x^\smile \sqcup y^\smile$
  assumes *conv-dist-comp* : $(x\ ;\ y)^\smile = y^\smile\ ;\ x^\smile$
  assumes *conv-complement-sub*: $x^\smile\ ;\ (x\ ;\ y)' \sqcup y\ ' = y\ '$

begin

lemma *conv-order*: $x \leq y \leftrightarrow x^\smile \leq y^\smile$
  by (*metis conv-dist-sup conv-involutive le-iff-sup*)

lemma *conv-zero*: $bot^\smile = bot$
  by (*metis conv-dist-sup conv-involutive sup-bot-right sup-eq-bot-iff*)

lemma *conv-top*: $top^\smile = top$
  by (*metis conv-involutive conv-order eq-iff top-greatest*)

lemma *conv-complement-0*: $x^\smile \sqcup (x\ ')^\smile = top$
  by (*metis conv-dist-sup conv-top sup-compl-top*)

lemma *conv-complement-1*: $(x^\smile)' \sqcup (x\ ')^\smile = (x\ ')^\smile$
  by (*smt compl-sup-top conv-dist-sup conv-top inf-compl-bot sup-idem sup-bot-right sup-commute sup-inf-distrib1 sup-top-right*)

lemma *conv-complement*: $(x\ ')^\smile = (x^\smile)'$
  by (*metis conv-complement-1 conv-dist-sup conv-involutive sup-commute*)

lemma *conv-dist-inf*: $(x \sqcap y)^\smile = x^\smile \sqcap y^\smile$
  by (*smt conv-complement compl-inf double-compl conv-dist-sup*)

lemma *conv-meet-zero-iff*: $bot = x^\smile \sqcap y \leftrightarrow bot = x \sqcap y^\smile$
by (*metis conv-dist-inf conv-involutive conv-zero*)

lemma *conv-one*: $1^\smile = 1$
by (*metis comp-right-one conv-dist-comp conv-involutive*)

lemma *comp-left-dist-sup*: $(x \ ; \ y) \sqcup (x \ ; \ z) = x \ ; \ (y \sqcup z)$
by (*metis comp-right-dist-sup conv-involutive conv-dist-sup conv-dist-comp*)

lemma *comp-right-isotone*: $x \leq y \rightarrow z \ ; \ x \leq z \ ; \ y$
by (*metis comp-left-dist-sup le-iff-sup*)

lemma *comp-left-isotone*: $x \leq y \rightarrow x \ ; \ z \leq y \ ; \ z$
by (*metis comp-right-dist-sup le-iff-sup*)

lemma *comp-left-conjugate*: *conjugate* $(\lambda y \ . \ x \ ; \ y)$ $(\lambda y \ . \ x^\smile \ ; \ y)$
proof −
let *?f* $= \lambda y \ . \ x \ ; \ y$
let *?g* $= \lambda y \ . \ x^\smile \ ; \ y$
have $\forall z \ y \ . \ ?f(z \sqcap (?g \ y)') \leq ?f \ z \sqcap y \ ' \wedge \ ?g(y \sqcap (?f \ z)') \leq ?g \ y \sqcap z \ '$
proof
fix *z*
show $\forall y \ . \ ?f(z \sqcap (?g \ y)') \leq ?f \ z \sqcap y \ ' \wedge \ ?g(y \sqcap (?f \ z)') \leq ?g \ y \sqcap z \ '$
proof
fix *y*
show $?f(z \sqcap (?g \ y)') \leq ?f \ z \sqcap y \ ' \wedge \ ?g(y \sqcap (?f \ z)') \leq ?g \ y \sqcap z \ '$
proof
have $?f(z \sqcap (?g \ y)') \leq ?f(z) \sqcap ?f((?g \ y)')$
by (*metis comp-right-isotone inf-greatest inf-le1 inf-le2*)
also have ... $\leq ?f(z) \sqcap y \ '$
by (*metis conv-complement-sub conv-involutive inf-mono le-iff-sup order-refl*)
finally show $?f(z \sqcap (?g \ y)') \leq ?f(z) \sqcap y \ '$
by *metis*
next
have $?g(y \sqcap (?f \ z)') \leq ?g(y) \sqcap ?g((?f \ z)')$
by (*metis comp-right-isotone inf-greatest inf-le1 inf-le2*)
also have ... $\leq ?g(y) \sqcap z \ '$
by (*metis conv-complement-sub inf-mono le-iff-sup order-refl*)
finally show $?g(y \sqcap (?f \ z)') \leq ?g \ y \sqcap z \ '$
by *metis*
qed
qed
qed
thus *conjugate ?f ?g*
by (*metis conjugate-char-1*)
qed

lemma *complement-conv-sub*: $(y\ ;\ x)'\ ;\ x^\smile \leq y\ '$
by (*smt conv-complement-sub conv-order conv-involutive conv-dist-comp conv-complement le-iff-sup*)

lemma *comp-right-conjugate*: *conjugate* $(\lambda y\ .\ y\ ;\ x)$ $(\lambda y\ .\ y\ ;\ x^\smile)$
proof −
let *?f* $= \lambda y\ .\ y\ ;\ x$
let *?g* $= \lambda y\ .\ y\ ;\ x^\smile$
have $\forall z\ y\ .\ ?f(z \sqcap (?g\ y)') \leq ?f\ z \sqcap y\ ' \wedge\ ?g(y \sqcap (?f\ z)') \leq ?g\ y \sqcap z\ '$
proof
fix *z*
show $\forall y\ .\ ?f(z \sqcap (?g\ y)') \leq ?f\ z \sqcap y\ ' \wedge\ ?g(y \sqcap (?f\ z)') \leq ?g\ y \sqcap z\ '$
proof
fix *y*
show $?f(z \sqcap (?g\ y)') \leq ?f\ z \sqcap y\ ' \wedge\ ?g(y \sqcap (?f\ z)') \leq ?g\ y \sqcap z\ '$
proof
have $?f(z \sqcap (?g\ y)') \leq ?f(z) \sqcap ?f((?g\ y)')$
by (*metis comp-left-isotone inf-greatest inf-le1 inf-le2*)
also have $\ldots \leq ?f(z) \sqcap y\ '$
by (*metis complement-conv-sub conv-involutive inf-mono le-iff-sup order-refl*)
finally show $?f(z \sqcap (?g\ y)') \leq ?f(z) \sqcap y\ '$
by *metis*
next
have $?g(y \sqcap (?f\ z)') \leq ?g(y) \sqcap ?g((?f\ z)')$
by (*metis comp-left-isotone inf-greatest inf-le1 inf-le2*)
also have $\ldots \leq ?g(y) \sqcap z\ '$
by (*metis complement-conv-sub inf-mono le-iff-sup order-refl*)
finally show $?g(y \sqcap (?f\ z)') \leq ?g\ y \sqcap z\ '$
by *metis*
qed
qed
qed
thus *conjugate ?f ?g*
by (*smt conjugate-char-1*)
qed

lemma *schroeder-1*: $x\ ;\ y \sqcap z = bot \leftrightarrow x^\smile\ ;\ z \sqcap y = bot$
by (*smt comp-left-conjugate conjugate-def inf-commute*)

lemma *schroeder-2*: $x\ ;\ y \sqcap z = bot \leftrightarrow z\ ;\ y^\smile \sqcap x = bot$
by (*smt comp-right-conjugate conjugate-def inf-commute*)

lemma *schroeder-3*: $x\ ;\ y \leq z \leftrightarrow x^\smile\ ;\ z\ ' \leq y\ '$
by (*metis double-compl schroeder-1 shunting-1*)

lemma *schroeder-4*: $x\ ;\ y \leq z \leftrightarrow z\ '\ ;\ y^\smile \leq x\ '$
by (*metis double-compl schroeder-2 shunting-1*)

lemma *dedekind-1*: $x \mathbin{;} y \sqcap z \leq x \mathbin{;} (y \sqcap (x^\smile \mathbin{;} z))$
by (*metis comp-left-conjugate conjugate-char-2*)

lemma *dedekind-2*: $y \mathbin{;} x \sqcap z \leq (y \sqcap (z \mathbin{;} x^\smile)) \mathbin{;} x$
by (*smt comp-right-conjugate conjugate-char-2*)

lemma *comp-left-zero*: $bot \mathbin{;} x = bot$
by (*metis comp-right-conjugate conjugate-char-2*)

lemma *comp-right-zero*: $x \mathbin{;} bot = bot$
by (*metis comp-left-conjugate conjugate-char-2*)

lemma *comp-left-one*: $1 \mathbin{;} x = x$
by (*metis comp-right-one conv-dist-comp conv-involutive*)

lemma *comp-right-top-increasing*: $x \leq x \mathbin{;} top$
by (*metis comp-right-isotone comp-right-one top-greatest*)

lemma *comp-left-top-increasing*: $x \leq top \mathbin{;} x$
by (*metis comp-left-isotone comp-left-one top-greatest*)

lemma *top-top*: $top \mathbin{;} top = top$
by (*metis comp-left-top-increasing top-unique*)

lemma *theorem24xxiii*: $x \mathbin{;} y \sqcap (x \mathbin{;} z)' = x \mathbin{;} (y \sqcap z\ ') \sqcap (x \mathbin{;} z)'$
proof −
have $x \mathbin{;} y \sqcap (x \mathbin{;} z)' \leq x \mathbin{;} (y \sqcap (x^\smile \mathbin{;} (x \mathbin{;} z)'))$
by (*metis dedekind-1*)
also have $\ldots \leq x \mathbin{;} (y \sqcap z\ ')$
by (*metis comp-right-isotone conv-complement-sub inf-mono le-iff-sup order-refl*)
finally have $x \mathbin{;} y \sqcap (x \mathbin{;} z)' \leq x \mathbin{;} (y \sqcap z\ ') \sqcap (x \mathbin{;} z)'$
by (*metis inf-le2 le-inf-iff*)
thus *?thesis*
by (*metis comp-right-isotone eq-iff inf-commute inf-le1 le-infI le-infI2*)
qed

lemma *theorem24xxiv*: $(x \mathbin{;} y)' \sqcup (x \mathbin{;} z) = (x \mathbin{;} (y \sqcap z\ '))' \sqcup (x \mathbin{;} z)$
by (*metis compl-inf double-compl theorem24xxiii*)

lemma *vector-complement*: $x = x \mathbin{;} top \rightarrow x\ ' = x\ ' \mathbin{;} top$
by (*metis comp-right-top-increasing complement-conv-sub conv-top eq-iff*)

lemma *vector-meet-comp*: $x = x \mathbin{;} top \rightarrow (x \sqcap y) \mathbin{;} z = x \sqcap (y \mathbin{;} z)$
proof
assume *1*: $x = x \mathbin{;} top$
hence $(x \sqcap y) \mathbin{;} z \leq x \sqcap (y \mathbin{;} z)$
by (*metis comp-left-isotone comp-right-isotone inf-assoc inf-commute inf-le2 le-iff-inf le-infI top-greatest*)

thus $(x \sqcap y)\ ;\ z = x \sqcap (y\ ;\ z)$ **using** *1*
by (*smt antisym comp-left-isotone comp-right-isotone dedekind-2 inf-commute inf-mono order-refl order-trans top-greatest*)
qed

lemma *vector-meet*: $x = x\ ;\ top \wedge y = y\ ;\ top \rightarrow x \sqcap y = (x \sqcap y)\ ;\ top$
by (*metis vector-meet-comp*)

lemma *vector-meet-one-comp*: $x = x\ ;\ top \rightarrow (x \sqcap 1)\ ;\ y = x \sqcap y$
by (*metis comp-left-one vector-meet-comp*)

lemma *covector-meet-comp-1*: $x = x\ ;\ top \rightarrow (y \sqcap x^\smile)\ ;\ z = (y \sqcap x^\smile)\ ;\ (x \sqcap z)$
proof
assume *1*: $x = x\ ;\ top$
have $(y \sqcap x^\smile)\ ;\ z \leq (y \sqcap x^\smile)\ ;\ (z \sqcap ((y^\smile \sqcap x)\ ;\ top))$
by (*metis inf-top-right dedekind-1 conv-dist-inf conv-involutive*)
also have $... \leq (y \sqcap x^\smile)\ ;\ (x \sqcap z)$ **using** *1*
by (*metis comp-left-isotone comp-right-isotone inf-le2 inf-mono order-refl inf-commute*)
finally show $(y \sqcap x^\smile)\ ;\ z = (y \sqcap x^\smile)\ ;\ (x \sqcap z)$
by (*metis comp-right-isotone eq-iff inf-le2*)
qed

lemma *covector-meet-comp-2*: $x = x\ ;\ top \rightarrow y\ ;\ (x \sqcap z) = (y \sqcap x^\smile)\ ;\ (x \sqcap z)$
proof
assume *1*: $x = x\ ;\ top$
have $y\ ;\ (x \sqcap z) \leq (y \sqcap (top\ ;\ (x \sqcap z)^\smile))\ ;\ (x \sqcap z)$
by (*metis dedekind-2 inf-top-right*)
also have $... \leq (y \sqcap x^\smile)\ ;\ (x \sqcap z)$ **using** *1*
by (*metis comp-left-isotone conv-dist-comp conv-order conv-top eq-refl inf-le1 inf-mono*)
finally show $y\ ;\ (x \sqcap z) = (y \sqcap x^\smile)\ ;\ (x \sqcap z)$
by (*metis comp-left-isotone eq-iff inf-le1*)
qed

lemma *coreflexive-conv*: $x \leq 1 \rightarrow x^\smile = x$
proof
assume *1*: $x \leq 1$
hence $x \leq x\ ;\ (1 \sqcap (x^\smile\ ;\ 1))$
by (*metis comp-right-one le-iff-inf dedekind-1*)
also have $... \leq x^\smile$ **using** *1*
by (*metis comp-left-isotone comp-right-one conv-dist-inf conv-one inf-absorb2 comp-left-one*)
thus $x^\smile = x$
by (*metis antisym calculation conv-involutive conv-order order-trans*)
qed

lemma *coreflexive-comp-top-meet*: $x \leq 1 \rightarrow x\ ;\ top \sqcap y = x\ ;\ y$
proof
assume *1*: $x \leq 1$
hence $x\ ;\ top \sqcap y \leq x\ ;\ y$

by (*metis comp-left-isotone comp-left-one coreflexive-conv dedekind-1 inf-top-left order-trans*)
thus $x\ ;\ top \sqcap y = x\ ;\ y$ **using** *1*
by (*metis antisym comp-left-isotone comp-left-one comp-right-isotone le-inf-iff top-greatest*)
qed

lemma *coreflexive-comp-top-complement-meet-one*: $x \leq 1 \rightarrow (x\ ;\ top)' \sqcap 1 = x\ ' \sqcap 1$
proof
assume *1*: $x \leq 1$
hence *2*: $x\ ;\ x^\smile\ ;\ (x\ ' \sqcap 1) \leq 1\ ;\ 1\ ;\ x\ '$
by (*metis comp-left-one coreflexive-comp-top-meet coreflexive-conv inf-commute inf-idem le-iff-inf le-infI2*)
have *3*: $x\ ;\ x^\smile\ ;\ (x\ ' \sqcap 1) \leq x\ ;\ 1\ ;\ 1$ **using** *1*
by (*metis comp-left-isotone comp-right-isotone inf-le2 order-trans coreflexive-conv*)
have $x\ ' \sqcap 1 \sqcap (x\ ;\ top) \leq x\ ;\ x^\smile\ ;\ (x\ ' \sqcap 1)$
by (*metis dedekind-1 inf-commute comp-associative inf-top-left*)
also have $... \leq bot$ **using** *2 3*
by (*metis le-inf-iff comp-left-one comp-right-one compl-inf-bot*)
finally have $x\ ' \sqcap 1 \leq (x\ ;\ top)' \sqcap 1$
by (*metis bot-unique double-compl shunting-1 inf-le2 le-inf-iff*)
thus $(x\ ;\ top)' \sqcap 1 = x\ ' \sqcap 1$
by (*metis antisym comp-right-top-increasing compl-le-compl-iff inf-mono order-refl*)
qed

lemma *coreflexive-comp-meet*: $x \leq 1 \wedge y \leq 1 \rightarrow x\ ;\ y = x \sqcap y$
by (*smt comp-right-one coreflexive-comp-top-meet inf-absorb1 inf-left-commute*)

lemma *coreflexive-comp-meet-comp*: $x \leq 1 \wedge y \leq 1 \rightarrow (x\ ;\ z) \sqcap (y\ ;\ z) = (x \sqcap y)\ ;\ z$
by (*smt comp-associative comp-left-isotone comp-right-one coreflexive-comp-top-meet inf-left-commute le-iff-inf*)

lemma *coreflexive-comp-meet-complement*: $x \leq 1 \rightarrow (x\ ;\ y) \sqcap z\ ' = (x\ ;\ y) \sqcap (x\ ;\ z)'$
by (*smt compl-le-compl-iff coreflexive-comp-top-meet inf-assoc inf-commute inf-left-idem inf-top-left le-iff-inf theorem24xxiii*)

lemma *vector-export-comp*: $(x\ ;\ top \sqcap 1)\ ;\ y = x\ ;\ top \sqcap y$
by (*metis comp-associative top-top vector-meet-one-comp*)

abbreviation $N :: 'a \Rightarrow 'a$
where $N(x) \equiv ((x\ ')\ ;\ top)' \sqcap 1$

lemma *N-comp*: $N(x)\ ;\ y = ((x\ ')\ ;\ top)' \sqcap y$
by (*metis comp-associative top-top vector-complement vector-export-comp*)

lemma *N-comp-top*: $N(x)\ ;\ top = ((x\ ')\ ;\ top)'$
by (*metis N-comp inf-top-right*)

lemma *vector-N*: $x = x\ ;\ top \rightarrow N(x) = x \sqcap 1$
by (*metis double-compl vector-complement*)

lemma *N-vector*: $N(x\ ;\ top) = x\ ;\ top \sqcap 1$
by (*metis comp-associative top-top vector-N*)

lemma *N-vector-top*: $N(x\ ;\ top)\ ;\ top = x\ ;\ top$
by (*metis N-vector inf-top-right vector-export-comp*)

lemma *N-below-meet-one*: $N(x) \leq x \sqcap 1$
by (*metis comp-right-top-increasing compl-le-swap2 inf-commute inf-le1 inf-mono le-inf-iff*)

lemma *N-below*: $N(x) \leq x$
by (*metis N-below-meet-one le-infE*)

lemma *N-comp-N*: $N(x)\ ;\ N(y) = ((x\ ')\ ;\ top)' \sqcap ((y\ ')\ ;\ top)' \sqcap 1$
by (*metis N-comp inf-assoc*)

lemma *N-zero*: $N(bot) = bot$
by (*metis compl-bot-eq compl-top-eq inf-bot-left top-top*)

lemma *N-top*: $N(top) = 1$
by (*metis inf-top-left top-top vector-N*)

lemma *n-split-omega-mult*: $xs\ ;\ xo = xo \wedge xo\ ;\ top = xo \rightarrow N(top)\ ;\ xo = xs\ ;\ N(xo)\ ;\ top$
by (*metis N-top N-vector-top comp-associative comp-left-one*)

end

sublocale *relation-algebra* $<$ *idl-semiring-T* **where** *plus* $=$ *sup* **and** *zero* $=$ *bot* **and** $T = top$
apply *unfold-locales*
apply (*metis sup-assoc*)
apply (*metis sup-commute*)
apply (*metis sup-idem*)
apply (*metis sup-bot-left*)
apply (*metis comp-associative*)
apply (*metis comp-right-one conv-dist-comp conv-involutive*)
apply (*metis comp-right-one*)
apply (*metis comp-left-dist-sup order-refl*)
apply (*metis comp-right-dist-sup*)
apply (*metis comp-left-zero*)
apply (*metis le-iff-sup*)
apply (*metis less-le-not-le*)
apply (*metis comp-left-dist-sup order-refl*)
apply (*metis sup-top-left*)
done

sublocale *relation-algebra* $<$ *idl-semiring-lattice* **where** *plus* $=$ *sup* **and** *zero* $=$ *bot* **and** $T = top$ **and** *meet* $=$ *inf*
apply *unfold-locales*

apply (*metis inf-assoc*)
apply (*metis inf-commute*)
apply (*metis inf-idem*)
apply (*metis inf-bot-left*)
apply (*metis inf-top-left*)
apply (*metis inf-sup-distrib1*)
apply (*metis sup-inf-distrib1*)
apply (*metis inf-sup-absorb*)
apply (*metis sup-inf-absorb*)
done

sublocale *relation-algebra* < *idl-semiring-lattice-nL* **where** *plus* = *sup* **and** *zero* = *bot* **and** *T* = *top* **and** *meet* = *inf* **and** *n* = *N* **and** *L* = *top*
apply *unfold-locales*
apply (*metis N-comp-top comp-associative compl-inf double-compl inf-sup-distrib2 top-top vector-meet-comp*)
apply (*metis N-comp compl-sup double-compl mult-associative mult-right-dist-add top-top N-comp-N*)
apply (*metis N-comp-N compl-inf compl-sup meet-absorb mult-right-dist-add*)
apply (*metis N-top inf-idem meet-right-upper-bound*)
apply (*metis N-comp-top compl-le-swap2 top-right-mult-increasing*)
apply (*metis N-top eq-refl mult-left-one mult-right-one*)
apply (*metis N-top N-zero comp-right-zero mult-left-one mult-left-zero mult-right-top sup-bot-right*)
apply (*metis N-vector-top comp-right-zero sup-bot-left*)
apply (*metis N-comp-top conv-complement-sub double-compl le-supI2 less-eq-def mult-associative mult-left-isotone schroeder-3*)
apply (*metis meet-left-upper-bound*)
done

sublocale *relation-algebra* < *idl-semiring-lattice-apx* **where** *plus* = *sup* **and** *zero* = *bot* **and** *T* = *top* **and** *meet* = *inf* **and** *n* = *N* **and** *L* = *top* **and** *apx* = *greater-eq*
apply *unfold-locales*
apply (*metis N-top mult-left-one n-less-eq-char sup-top-right top-greatest*)
done

class *while* =
fixes *while* :: $'a \Rightarrow 'a \Rightarrow 'a$ (**infixr** $*$ *60*)

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

begin

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

lemma *while-plus-one*: $x * y = y + (x * y)$

by (*metis less-eq-def while-increasing*)

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

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

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

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

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

lemma *while-mult-upper-bound*: $x \leq z * 1 \wedge y \leq z * w \rightarrow x \ ; \ y \leq z * w$
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*: $(y * 1) \ ; \ (y * v) \leq y * v$
by (*metis order-refl while-mult-upper-bound*)

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

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

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

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

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

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

lemma *while-sumstar-1-below*: $x * ((y \ ; \ (x * 1)) * z) \leq ((x * 1) \ ; \ y) * (x * z)$
proof −
have *1*: $x \ ; \ (((x * 1) \ ; \ y) * (x * z)) \leq ((x * 1) \ ; \ y) * (x * z)$
by (*smt add-isotone add-right-upper-bound mult-associative mult-left-dist-add mult-right-sub-dist-add-right while-left-unfold*)
have $x * ((y \ ; \ (x * 1)) * z) \leq (x * z) + (x * (y \ ; \ (((x * 1) \ ; \ y) * ((x * 1) \ ; \ z))))$
by (*metis eq-refl while-left-dist-add while-productstar*)

also have ... $\leq (x * z) + (x * ((x * 1)\ ;\ y\ ;\ (((x * 1)\ ;\ y) * ((x * 1)\ ;\ z))))$
by (*metis add-right-isotone mult-associative mult-left-one mult-right-sub-dist-add-left while-left-unfold while-right-isotone*)
also have ... $\leq (x * z) + (x * (((x * 1)\ ;\ y) * ((x * 1)\ ;\ z)))$
by (*metis add-right-isotone add-right-upper-bound while-left-unfold while-right-isotone*)
also have ... $\leq x * (((x * 1)\ ;\ y) * (x * z))$
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 ... $\leq (((x * 1)\ ;\ y) * (x * z)) + (x * 0)$ **using** *1*
by (*metis while-simulate-absorb*)
also have ... $= ((x * 1)\ ;\ y) * (x * z)$
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*: $((x * 1)\ ;\ y) * (x * z) \leq (x * y) * (x * z)$
by (*metis mult-left-one while-left-isotone while-sub-associative*)

lemma *while-add-1-below*: $x * ((y\ ;\ (x * 1)) * z) \leq (x + y) * z$
proof −
have $((x * 1)\ ;\ y) * ((x * 1)\ ;\ z) \leq (x + y) * z$
by (*metis while-isotone while-one-mult-below while-sumstar*)
hence $(y\ ;\ (x * 1)) * z \leq z + y\ ;\ ((x + y) * z)$
by (*metis add-right-isotone mult-right-isotone while-productstar*)
also have ... $\leq (x + y) * z$
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

lemma *while-while-while*: $((x * 1) * 1) * y = (x * 1) * y$
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*: $(1 * 1) * y = 1 * y$
by (*metis while-while-while while-zero*)

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

lemma *while-add-2*: $(x + y) * z \leq (x * (y * 1)) * z$
by (*metis while-add-below while-left-isotone*)

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

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

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

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

lemma *while-decompose-10*: $(x * (y * 1)) * z = (x * (y * 1)) * (x * (y * z))$
proof −
have *1*: $(x * (y * 1)) * z \leq (x * (y * 1)) * (x * (y * z))$
by (*metis add-associative less-eq-def while-left-dist-add while-plus-one*)
have $x + (y * 1) \leq x * (y * 1)$
by (*metis add-least-upper-bound while-add-below while-increasing*)
hence $(x * (y * 1)) * (x * (y * z)) \leq (x * (y * 1)) * z$
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*: $z \ ; (y * (y \ ; x)) + z \ ; x = z \ ; (y * x)$
by (*metis add-commutative mult-left-dist-add while-right-unfold*)

lemma *while-back-loop-prefixpoint*: $z \ ; (y * 1) \ ; y + z \leq z \ ; (y * 1)$
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*)

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

by (*metis while-left-unfold*)
also have ... $\leq (x * z) + (x * (y\ ;\ 0))$
by (*metis add-right-isotone mult-associative mult-left-zero while-sub-associative*)
also have ... $= x * ((y\ ;\ 0) * z)$
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*: $(x + y\ ;\ 0) * y = x * y$
by (*metis less-eq-def while-mult-zero-add while-zero-2 zero-right-mult-decreasing*)

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

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

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

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

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

lemma *while-while-mult*: $1 * (x * y) = (x * 1) * y$
proof −
have $(x * 1) * y \leq (x * 1)\ ;\ ((x * 1) * y)$
by (*metis order-refl while-increasing while-sup-one-left-unfold*)
also have ... $\leq 1 * ((x * 1)\ ;\ y)$
by (*metis mult-left-one order-refl while-mult-upper-bound while-simulate*)
also have ... $\leq 1 * (x * y)$
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*: $x\ ;\ z \leq z\ ;\ y \rightarrow x * (z\ ;\ v) \leq z\ ;\ (y * v) + (x * 0)$
by (*metis add-right-zero mult-left-zero while-simulate-left*)

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

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

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

lemma *while-decompose-5-below*: $(x * (y * 1)) * z \leq (y * (x * 1)) * z$
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*)

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

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

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

finally show $y * (x * 1) \leq x * (y * 1)$
.
next
assume $y * (x * 1) \leq x * (y * 1)$
thus $y \ ; (x * 1) \leq x * (y * 1)$
by (*metis order-trans while-mult-increasing*)
qed

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

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

lemma *while-extra-while*: $(y \ ; (x * 1)) * z = (y \ ; (y * (x * 1))) * z$
proof −
have $y \ ; (y * (x * 1)) \leq y \ ; (x * 1) \ ; (y \ ; (x * 1) * 1)$
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*: $(y \ ; (y * (x * 1))) * z \leq (y \ ; (x * 1)) * z$
by (*metis while-simulate-right-plus-1 mult-left-one*)
have $(y \ ; (x * 1)) * z \leq (y \ ; (y * (x * 1))) * z$
by (*metis while-increasing while-left-isotone while-mult-star-exchange*)
thus *?thesis* **using** *1*
by (*metis antisym*)
qed

lemma *while-separate-4*: $y \ ; x \leq x \ ; (x * (1 + y)) \rightarrow (x + y) * z = x * (y * z)$
proof
assume *1*: $y \ ; x \leq x \ ; (x * (1 + y))$
hence $(1 + y) \ ; x \leq x \ ; (x * (1 + y))$
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*: $(1 + y) \ ; (x * 1) \leq x * (1 + y)$
by (*metis mult-right-one while-simulate-right-plus-1*)
have $y \ ; x \ ; (x * 1) \leq x \ ; (x * ((1 + y) \ ; (x * 1)))$ **using** *1*
by (*smt less-eq-def mult-associative mult-right-dist-add while-associative-1 while-increasing*)
also have $... \leq x \ ; (x * (1 + y))$ **using** *2*
by (*metis mult-right-isotone order-refl while-mult-transitive*)
also have $... \leq x \ ; (x * 1) \ ; (y * 1)$
by (*metis add-least-upper-bound mult-associative mult-right-isotone while-increasing while-one-increasing while-one-while while-right-isotone*)
finally have $y * (x \ ; (x * 1)) \leq x \ ; (x * 1) \ ; (y * 1) + (y * 0)$
by (*metis mult-associative mult-right-one while-simulate-left-plus-1*)
hence $(y * 1) \ ; (y * x) \leq x \ ; (x * y * 1) + (y * 0)$
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 $(y * 1) \ ; ((y * x) * (y * z)) \leq x * ((y * 1) \ ; (y * z) + (y * 0) \ ; ((y * x) * (y * z)))$
by (*metis while-simulate-right-plus*)
also have $... \leq x * ((y * z) + (y * 0))$
by (*metis add-isotone mult-left-zero order-refl while-absorb-2 while-one-mult-below while-right-isotone while-sub-associative*)

also have ... $= x * y * z$
by (*metis add-right-zero while-left-dist-add*)
finally show $(x + y) * z = x * (y * z)$
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*: $y \ ; \ x \leq x \ ; \ (x * (x + y)) \rightarrow (x + y) * z = x * (y * z)$
by (*smt order-trans while-add-sub-add-one-mult while-separate-4*)

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

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

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

lemma *separation*: $y \ ; \ x \leq x \ ; \ (y * 1) \rightarrow (x + y) * z = x * (y * z)$
proof
assume $y \ ; \ x \leq x \ ; \ (y * 1)$
hence $y * x \leq x \ ; \ (y * 1) + (y * 0)$
by (*metis mult-right-one while-simulate-left-plus-1*)
also have ... $\leq x \ ; \ (x * y * 1) + (y * 0)$
by (*metis add-left-isotone while-increasing while-mult-star-exchange*)
finally have $(y * 1) \ ; \ (y * x) \leq x \ ; \ (x * y * 1) + (y * 0)$
by (*metis order-refl order-trans while-absorb-2 while-one-mult-below*)
hence $(y * 1) \ ; \ ((y * x) * (y * z)) \leq x * ((y * 1) \ ; \ (y * z) + (y * 0) \ ; \ ((y * x) * (y * z)))$
by (*metis while-simulate-right-plus*)
also have ... $\leq x * ((y * z) + (y * 0))$
by (*metis add-isotone mult-left-zero order-refl while-absorb-2 while-one-mult-below while-right-isotone while-sub-associative*)
also have ... $= x * y * z$
by (*metis add-right-zero while-left-dist-add*)
finally show $(x + y) * z = x * (y * z)$
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-left*: $y \ ; \ x \leq x \ ; \ (y * 1) \rightarrow y * (x * z) \leq x * (y * z)$
by (*metis add-commutative separation while-sub-dist-3*)

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

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

lemma *while-simulate-6*: $y \ ; \ x \leq x \ ; \ (x + y) \rightarrow y * (x * z) \leq x * (y * z)$

by (*smt order-trans while-increasing while-mult-star-exchange while-simulate-5*)

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

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

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

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

lemma *while-preserve*: $p \leq p \ ; \ p \wedge p \leq 1 \wedge p \ ; \ x \leq x \ ; \ p \rightarrow p \ ; \ (x * y) = p \ ; \ (x * (p \ ; \ y))$
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*: $(x * 1) \ ; \ x \leq x * 1$
by (*metis order-refl while-mult-upper-bound while-one-increasing*)

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

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

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

end

class *binary-itering-T* = *idl-semiring-T* + *binary-itering*

begin

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

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

end

class *binary-itering-1* = *binary-itering* +
assumes *while-denest-0*: $w \ ; (x * (y \ ; z)) \leq (w \ ; (x * y)) * (w \ ; (x * y) \ ; z)$

begin

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

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

lemma *while-zero-zero*: $(x * 0) * 0 = x * 0$
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*)

lemma *while-mult-zero-zero*: $(x \ ; \ (y * 0)) * 0 = x \ ; \ (y * 0)$
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

lemma *while-denest-2*: $w \ ; \ ((x * (y \ ; \ w)) * z) = w \ ; \ (((x * y) \ ; \ w) * z)$
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

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

lemma *while-02*: $x * ((x * w) * ((x * y) \ ; \ z)) = (x * w) * ((x * y) \ ; \ z)$
proof −
have $x \ ; \ ((x * w) * ((x * y) \ ; \ z)) = x \ ; \ (x * y) \ ; \ z + x \ ; \ (x * w) \ ; \ ((x * w) * ((x * y) \ ; \ z))$
by (*metis mult-associative mult-left-dist-add while-left-unfold*)
also have $... \leq (x * w) * ((x * y) \ ; \ z)$
by (*metis add-isotone mult-right-sub-dist-add-right while-left-unfold*)
finally have $x * ((x * w) * ((x * y) \ ; \ z)) \leq ((x * w) * ((x * y) \ ; \ z)) + (x * 0)$
by (*metis while-simulate-absorb*)
also have $... = (x * w) * ((x * y) \ ; \ z)$
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*: $(x * y) * (x * z) \leq (x * y) * ((x * 1) \ ; \ z)$
proof −
have $(x * y) * (x * z) = (x * z) + ((x * y) * ((x * y) \ ; \ (x * z)))$
by (*metis while-right-unfold*)
also have $... \leq (x * z) + ((x * y) * (x * (y \ ; \ (x * z))))$
by (*metis add-right-isotone while-right-isotone while-sub-associative*)
also have $... \leq (x * z) + ((x * y) * (x * ((x * y) * (x * z))))$
by (*smt add-right-isotone order-trans while-increasing while-mult-upper-bound while-one-increasing while-right-isotone*)
also have $... \leq (x * z) + ((x * y) * (x * ((x * y) * ((x * 1) \ ; \ z))))$

by (*metis add-right-isotone mult-left-isotone mult-left-one order-trans while-increasing while-right-isotone while-sumstar while-transitive*)
also have ... = $(x * z) + ((x * y) * ((x * 1) ; z))$
by (*metis while-02 while-transitive*)
also have ... = $(x * y) * ((x * 1) ; z)$
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*: $(x * y) * ((x * 1) ; z) \leq x * ((y ; (x * 1)) * z)$
proof −
have $(x * y) * ((x * 1) ; z) = (x * 1) ; z + (x * y) ; ((x * y) * ((x * 1) ; z))$
by (*metis while-left-unfold*)
also have ... $\leq (x * z) + (x * (y ; ((x * y) * ((x * 1) ; z))))$
by (*metis add-isotone while-one-mult-below while-sub-associative*)
also have ... = $(x * z) + (x * (y ; (((x * 1) ; y) * ((x * 1) ; z))))$
by (*metis mult-left-one while-denest-2*)
also have ... = $x * ((y ; (x * 1)) * z)$
by (*metis while-left-dist-add while-productstar*)
finally show *?thesis*
.
qed

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

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

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

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

lemma *while-denest-4*: $(x * w) * (x * (y ; z)) = (x * w) * ((x * y) ; z)$
proof −
have $(x * w) * (x * (y ; z)) = x * ((w ; (x * 1)) * (y ; z))$
by (*metis while-sumstar while-sumstar-2*)
also have ... $\leq (x * w) * ((x * y) ; z)$
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

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

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

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

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

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

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

by (*smt mult-associative mult-right-isotone while-simulate*)
also have ... = s ; (l * q ; (r * (x ; (b * q) ; (r * 1)) * z)) **using** *1*
by (*metis while-elimination*)
also have ... ≤ s ; (l * r * (x ; (b * q) ; (r * 1)) * 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 * (r + x ; (b * q)) * z)
by (*metis while-sumstar-2*)
also have ... ≤ s ; ((x ; (b * q) + r + l) * z)
by (*metis add-commutative mult-right-isotone while-sub-dist-3*)
finally show s ; ((x + b + r + l) * (q ; z)) ≤ s ; ((x ; (b * q) + r + l) * z)
.
qed

end

class *binary-itering-1-T* = *binary-itering-T* + *binary-itering-1*

begin

end

class *strict-itering* = *itering-4* + *while* +
assumes *while-def*: x * y = x° ; y

begin

subclass *binary-itering-1*
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

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

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

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

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

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

end

class *strict-itering-T* = *itering-T* + *strict-itering*

begin

subclass *binary-itering-1-T* **..**

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

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

end

class *nonstrict-itering* = *sdloa-T* + *while* +
assumes *while-def*: $x * y = x^{\omega} + x^{*}\ ;\ y$

begin

subclass *binary-itering-T*
proof (*unfold-locales*)
fix *x y z*
show $(x\ ;\ y) * z = z + x\ ;\ ((y\ ;\ x) * (y\ ;\ z))$
by (*metis add-commutative mult-associative mult-left-dist-add omega-loop-fixpoint omega-slide star.circ-slide while-def*)
next
fix *x y z*
show $(x + y) * z = (x * y) * (x * z)$
proof −
have *1*: $(x + y) * z = (x^{*}\ ;\ y)^{\omega} + (x^{*}\ ;\ y)^{*}\ ;\ (x^{\omega} + x^{*}\ ;\ z)$
by (*smt add-associative mult-associative mult-left-dist-add omega-decompose star.circ-add while-def*)

hence *2*: $(x + y) * z \leq (x * y) * (x * z)$
by (*smt add-isotone add-right-upper-bound less-eq-def mult-left-isotone omega-sub-dist star.circ-sub-dist while-def*)
let *?rhs* = $x^*\ ;\ y\ ;\ ((x^\omega + x^*\ ;\ y)^\omega + (x^\omega + x^*\ ;\ y)^*\ ;\ (x^\omega + x^*\ ;\ z)) + (x^\omega + x^*\ ;\ z)$
have $x^\omega\ ;\ (x^\omega + x^*\ ;\ y)^\omega \leq x^\omega$
by (*metis omega-sub-vector*)
hence $x^\omega\ ;\ (x^\omega + x^*\ ;\ y)^\omega + x^*\ ;\ y\ ;\ (x^\omega + x^*\ ;\ y)^\omega \leq$ *?rhs*
by (*smt add-commutative add-isotone add-left-upper-bound mult-left-dist-add order-trans*)
hence *3*: $(x^\omega + x^*\ ;\ y)^\omega \leq$ *?rhs*
by (*metis mult-right-dist-add omega-unfold*)
have $x^\omega\ ;\ (x^\omega + x^*\ ;\ y)^*\ ;\ (x^\omega + x^*\ ;\ z) \leq x^\omega$
by (*metis mult-associative omega-sub-vector*)
hence $x^\omega\ ;\ (x^\omega + x^*\ ;\ y)^*\ ;\ (x^\omega + x^*\ ;\ z) + x^*\ ;\ y\ ;\ (x^\omega + x^*\ ;\ y)^*\ ;\ (x^\omega + x^*\ ;\ z) \leq$ *?rhs*
by (*smt add-commutative add-isotone add-right-upper-bound mult-associative mult-left-dist-add order-trans*)
hence $(x^\omega + x^*\ ;\ y)^*\ ;\ (x^\omega + x^*\ ;\ z) \leq$ *?rhs*
by (*smt add-associative add-right-upper-bound less-eq-def mult-associative mult-right-dist-add star.circ-loop-fixpoint*)
hence $(x^\omega + x^*\ ;\ y)^\omega + (x^\omega + x^*\ ;\ y)^*\ ;\ (x^\omega + x^*\ ;\ z) \leq$ *?rhs* **using** *3*
by (*metis add-least-upper-bound*)
hence $(x^\omega + x^*\ ;\ y)^\omega + (x^\omega + x^*\ ;\ y)^*\ ;\ (x^\omega + x^*\ ;\ z) \leq (x^*\ ;\ y)^\omega + (x^*\ ;\ y)^*\ ;\ (x^\omega + x^*\ ;\ z)$
by (*metis add-commutative omega-induct*)
thus *?thesis* **using** *1 2*
by (*smt antisym while-def*)
qed
next
fix $x\ y\ z$
show $x * (y + z) = (x * y) + (x * z)$
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 $(x * y)\ ;\ z \leq x * (y\ ;\ z)$
by (*metis mult-associative mult-right-dist-add omega-loop-fixpoint omega-loop-greatest-fixpoint while-def*)
next
fix $v\ w\ x\ y\ z$
show $x\ ;\ z \leq z\ ;\ (y * 1) + w \rightarrow x * (z\ ;\ v) \leq z\ ;\ (y * v) + (x * (w\ ;\ (y * v)))$
proof
assume $x\ ;\ z \leq z\ ;\ (y * 1) + w$
hence *1*: $x\ ;\ z \leq z\ ;\ y^\omega + z\ ;\ y^* + w$
by (*metis mult-left-dist-add mult-right-one while-def*)
let *?rhs* = $z\ ;\ (y^\omega + y^*\ ;\ v) + x^\omega + x^*\ ;\ w\ ;\ (y^\omega + y^*\ ;\ v)$
have *2*: $z\ ;\ v \leq$ *?rhs*
by (*metis add-least-upper-bound add-left-upper-bound mult-left-dist-add omega-loop-fixpoint*)
have $x\ ;\ z\ ;\ (y^\omega + y^*\ ;\ v) \leq$ *?rhs*
proof −
have $x\ ;\ z\ ;\ (y^\omega + y^*\ ;\ v) \leq (z\ ;\ y^\omega + z\ ;\ y^* + w)\ ;\ (y^\omega + y^*\ ;\ v)$ **using** *1*
by (*metis mult-left-isotone*)
also have ... $= z\ ;\ (y^\omega\ ;\ (y^\omega + y^*\ ;\ v) + y^*\ ;\ (y^\omega + y^*\ ;\ v)) + w\ ;\ (y^\omega + y^*\ ;\ v)$
by (*smt mult-associative mult-left-dist-add mult-right-dist-add*)
also have ... $= z\ ;\ (y^\omega\ ;\ (y^\omega + y^*\ ;\ v) + y^\omega + y^*\ ;\ v) + w\ ;\ (y^\omega + y^*\ ;\ v)$

```
    by (smt add-associative mult-associative mult-left-dist-add star.circ-transitive-equal star-mult-omega)
  also have ... ≤ z ; (y^ω + y^* ; v) + x^* ; w ; (y^ω + y^* ; v)
    by (smt add-commutative add-isotone add-left-top mult-left-dist-add mult-left-one mult-right-dist-add mult-right-sub-dist-add-left omega-vector order-refl star.circ-plus-one)
  finally show ?thesis
    by (smt add-associative add-commutative less-eq-def)
 qed
 hence x ; ?rhs ≤ ?rhs
 by (smt add-associative add-commutative add-left-upper-bound less-eq-def mult-associative mult-left-dist-add mult-right-dist-add omega-unfold star.circ-increasing star.circ-transitive-equal)
 hence z ; v + x ; ?rhs ≤ ?rhs using 2
   by (metis add-least-upper-bound)
 hence x^* ; z ; v ≤ ?rhs
   by (metis mult-associative star-left-induct)
 hence x^ω + x^* ; z ; v ≤ ?rhs
   by (metis add-least-upper-bound add-left-upper-bound)
 thus x * (z ; v) ≤ z ; (y * v) + (x * (w ; (y * v)))
   by (smt add-associative mult-associative mult-left-dist-add while-def)
qed
next
 fix v w x y z
 show z ; x ≤ y ; (y * z) + w → z ; (x * v) ≤ y * (z ; v + w ; (x * v))
 proof
   assume z ; x ≤ y ; (y * z) + w
   hence z ; x ≤ y ; (y^ω + y^* ; z) + w
     by (metis while-def)
   hence 1: z ; x ≤ y^ω + y ; y^* ; z + w
     by (metis mult-associative mult-left-dist-add omega-unfold)
   let ?rhs = y^ω + y^* ; z ; v + y^* ; w ; (x^ω + x^* ; v)
   have 2: z ; x^ω ≤ ?rhs
   proof −
     have z ; x^ω ≤ y ; y^* ; z ; x^ω + y^ω ; x^ω + w ; x^ω using 1
       by (smt add-commutative less-eq-def mult-associative mult-right-dist-add omega-unfold)
     also have ... ≤ y ; y^* ; z ; x^ω + y^ω + w ; x^ω
       by (metis add-left-isotone add-right-isotone omega-sub-vector)
     also have ... = y ; y^* ; (z ; x^ω) + (y^ω + w ; x^ω)
       by (metis add-associative mult-associative)
     finally have z ; x^ω ≤ (y ; y^*)^ω + (y ; y^*)^* ; (y^ω + w ; x^ω)
       by (metis add-commutative omega-induct)
     also have ... = y^ω + y^* ; w ; x^ω
       by (metis left-plus-omega less-eq-def mult-associative mult-left-dist-add mult-left-sub-dist-add-left star.left-plus-circ star-mult-omega)
     also have ... ≤ ?rhs
       by (metis add-isotone add-left-upper-bound mult-left-sub-dist-add-left)
     finally show ?thesis
       by metis
   qed
   let ?rhs2 = y^ω + y^* ; z + y^* ; w ; (x^ω + x^*)
   have ?rhs2 ; x ≤ ?rhs2
   proof −
```

```
    have 3: y^ω ; x ≤ ?rhs2
      by (metis add-associative less-eq-def omega-sub-vector)
    have y* ; z ; x ≤ y* ; (y^ω + y ; y* ; z + w) using 1
      by (metis mult-associative mult-right-isotone)
    also have ... = y^ω + y* ; y ; y* ; z + y* ; w
      by (metis mult-associative mult-left-dist-add star-mult-omega)
    also have ... = y^ω + y ; y* ; z + y* ; w
      by (metis mult-associative star.circ-transitive-equal star-simulation-right-equal)
    also have ... ≤ y^ω + y* ; z + y* ; w
      by (metis add-left-isotone add-right-isotone mult-left-isotone star.left-plus-below-circ)
    also have ... ≤ y^ω + y* ; z + y* ; w ; x*
      by (metis add-right-isotone add-right-upper-bound star.circ-back-loop-fixpoint)
    finally have 4: y* ; z ; x ≤ ?rhs2
      by (smt add-associative add-commutative less-eq-def mult-left-dist-add)
    have (x^ω + x*) ; x ≤ x^ω + x*
      by (metis add-isotone mult-right-dist-add omega-sub-vector star.circ-plus-same star.left-plus-below-circ)
    hence y* ; w ; (x^ω + x*) ; x ≤ ?rhs2
      by (smt add-associative add-commutative less-eq-def mult-associative mult-left-dist-add)
    thus ?thesis using 3 4
      by (smt add-associative less-eq-def mult-right-dist-add)
  qed
  hence z + ?rhs2 ; x ≤ ?rhs2
    by (smt add-commutative add-least-upper-bound add-right-divisibility while-def omega-loop-fixpoint)
  hence 5: z ; x* ≤ ?rhs2
    by (metis star-right-induct)
  have z ; x* ; v ≤ ?rhs
  proof -
    have z ; x* ; v ≤ ?rhs2 ; v using 5
      by (metis mult-left-isotone)
    also have ... = y^ω ; v + y* ; z ; v + y* ; w ; (x^ω ; v + x* ; v)
      by (metis mult-associative mult-right-dist-add)
    also have ... ≤ y^ω + y* ; z ; v + y* ; w ; (x^ω ; v + x* ; v)
      by (metis add-left-isotone omega-sub-vector)
    also have ... ≤ ?rhs
      by (metis add-left-isotone add-right-isotone mult-right-isotone omega-sub-vector)
    finally show ?thesis
      by metis
  qed
  hence z ; (x^ω + x* ; v) ≤ ?rhs using 2
    by (smt add-associative less-eq-def mult-associative mult-left-dist-add)
  thus z ; (x * v) ≤ y * (z ; v + w ; (x * v))
    by (metis add-associative mult-associative mult-left-dist-add while-def)
 qed
qed

lemma while-top: T * x = T
 by (metis add-left-top star.circ-top star-omega-top while-def)
```

lemma *while-one-top*: $1 * x = T$
by (*metis add-left-top omega-one while-def*)

lemma *while-finite-associative*: $x^\omega = 0 \rightarrow (x * y) \ ; z = x * (y \ ; z)$
by (*metis add-left-zero mult-associative while-def*)

lemma *star-below-while*: $x^* \ ; y \leq x * y$
by (*metis add-right-upper-bound while-def*)

lemma *while-sub-mult-one*: $x \ ; (1 * y) \leq 1 * x$
by (*metis top-greatest while-one-top*)

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

lemma *while-simulate-4-plus*: $y \ ; x \leq x \ ; (x * (1 + y)) \rightarrow y \ ; x \ ; x^* \leq x \ ; (x * (1 + y))$
proof
have *1*: $x \ ; (x * (1 + y)) = x^\omega + x \ ; x^* + x \ ; x^* \ ; y$
by (*metis add-associative mult-associative mult-left-dist-add mult-right-one omega-unfold while-def*)
assume $y \ ; x \leq x \ ; (x * (1 + y))$
hence $y \ ; x \ ; x^* \leq (x^\omega + x \ ; x^* + x \ ; x^* \ ; y) \ ; x^*$ **using** *1*
by (*metis mult-left-isotone*)
also have $... = x^\omega \ ; x^* + x \ ; x^* \ ; x^* + x \ ; x^* \ ; y \ ; x^*$
by (*metis mult-right-dist-add*)
also have $... = x \ ; x^* \ ; (y \ ; x \ ; x^*) + x^\omega + x \ ; x^* + x \ ; x^* \ ; y$
by (*smt add-associative add-commutative mult-associative omega-mult-star-2 star.circ-back-loop-fixpoint star.circ-plus-same star.circ-transitive-equal*)
finally have $y \ ; x \ ; x^* \leq x \ ; x^* \ ; (y \ ; x \ ; x^*) + (x^\omega + x \ ; x^* + x \ ; x^* \ ; y)$
by (*metis add-associative*)
hence $y \ ; x \ ; x^* \leq (x \ ; x^*)^\omega + (x \ ; x^*)^* \ ; (x^\omega + x \ ; x^* + x \ ; x^* \ ; y)$
by (*metis add-commutative omega-induct*)
also have $... = x^\omega + x^* \ ; (x^\omega + x \ ; x^* + x \ ; x^* \ ; y)$
by (*metis left-plus-omega star.left-plus-circ*)
finally show $y \ ; x \ ; x^* \leq x \ ; (x * (1 + y))$ **using** *1*
by (*metis while-def while-mult-star-exchange while-transitive*)
qed

lemma *while-simulate-4-omega*: $y \ ; x \leq x \ ; (x * (1 + y)) \rightarrow y \ ; x^\omega \leq x^\omega$
proof
have *1*: $x \ ; (x * (1 + y)) = x^\omega + x \ ; x^* + x \ ; x^* \ ; y$
by (*metis add-associative mult-associative mult-left-dist-add mult-right-one omega-unfold while-def*)
assume $y \ ; x \leq x \ ; (x * (1 + y))$
hence $y \ ; x^\omega \leq (x^\omega + x \ ; x^* + x \ ; x^* \ ; y) \ ; x^\omega$ **using** *1*
by (*smt less-eq-def mult-associative mult-right-dist-add omega-unfold*)
also have $... = x^\omega \ ; x^\omega + x \ ; x^* \ ; x^\omega + x \ ; x^* \ ; y \ ; x^\omega$
by (*metis mult-right-dist-add*)
also have $... = x \ ; x^* \ ; (y \ ; x^\omega) + x^\omega$

by (*metis add-commutative less-eq-def mult-associative omega-sub-vector omega-unfold star-mult-omega*)
finally have $y \ ; \ x^\omega \leq x \ ; \ x^* \ ; \ (y \ ; \ x^\omega) + x^\omega$
by *metis*
hence $y \ ; \ x^\omega \leq (x \ ; \ x^*)^\omega + (x \ ; \ x^*)^* \ ; \ x^\omega$
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-unfold-below*: $x = z + y \ ; \ x \rightarrow x \leq y * z$
by (*metis omega-induct-equal while-def*)

lemma *while-unfold-below-1*: $x = y \ ; \ x \rightarrow x \leq y * 1$
by (*metis add-right-upper-bound omega-induct while-def*)

lemma *while-square-1*: $x * 1 = (x \ ; \ x) * (x + 1)$
by (*metis mult-right-one omega-square star-square-2 while-def*)

lemma *while-absorb-below-one*: $y \ ; \ x \leq x \rightarrow y * x \leq 1 * x$
by (*metis top-greatest while-one-top*)

lemma *while-loop-is-greatest-postfixpoint*: *is-greatest-postfixpoint* $(\lambda x \ . \ y \ ; \ x + z) \ (y * z)$
proof −
have $(y * z) \leq (\lambda x \ . \ y \ ; \ x + z) \ (y * z)$
by (*metis is-fixpoint-def order-refl while-loop-is-fixpoint*)
thus *?thesis*
by (*smt add-commutative is-greatest-postfixpoint-def omega-induct while-def*)
qed

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

end

class *nonstrict-itering-zero* = *nonstrict-itering* +
assumes *mult-right-zero*: $x \ ; \ 0 = 0$

begin

lemma *while-finite-associative-2*: $x * 0 = 0 \rightarrow (x * y) \ ; \ z = x * (y \ ; \ z)$
by (*metis add-left-zero add-right-zero mult-associative mult-right-zero while-def*)

end

class *nonstrict-itering-tarski* = *nonstrict-itering* +
  assumes *tarski*: $x \leq x \mathbin{;} T \mathbin{;} x \mathbin{;} T$

begin

lemma *tarski-mult-top-idempotent*: $x \mathbin{;} T = x \mathbin{;} T \mathbin{;} x \mathbin{;} T$
  by (*metis add-commutative less-eq-def mult-associative star.circ-back-loop-fixpoint star.circ-left-top tarski top-mult-top*)

lemma *tarski-top-omega-below*: $x \mathbin{;} T \leq (x \mathbin{;} T)^\omega$
  by (*metis mult-associative omega-induct-mult order-refl tarski-mult-top-idempotent*)

lemma *tarski-top-omega*: $x \mathbin{;} T = (x \mathbin{;} T)^\omega$
  by (*metis antisym mult-top-omega tarski-top-omega-below*)

lemma *tarski-below-top-omega*: $x \leq (x \mathbin{;} T)^\omega$
  by (*metis tarski-top-omega top-right-mult-increasing*)

lemma *tarski-mult-omega-omega*: $(x \mathbin{;} y^\omega)^\omega = x \mathbin{;} y^\omega$
  by (*metis mult-associative omega-vector tarski-top-omega*)

lemma *tarski-omega-idempotent*: $x^{\omega\omega} = x^\omega$
  by (*metis omega-vector tarski-top-omega*)

lemma *while-denest-2a*: $w \mathbin{;} ((x * (y \mathbin{;} w)) * z) = w \mathbin{;} (((x * y) \mathbin{;} w) * z)$
proof −
  have $(x^\omega + x^* \mathbin{;} y \mathbin{;} w)^\omega = (x^* \mathbin{;} y \mathbin{;} w)^* \mathbin{;} x^\omega \mathbin{;} (((x^* \mathbin{;} y \mathbin{;} w)^* \mathbin{;} x^\omega)^\omega + ((x^* \mathbin{;} y \mathbin{;} w)^* \mathbin{;} x^\omega)^* \mathbin{;} (x^* \mathbin{;} y \mathbin{;} w)^\omega) + (x^* \mathbin{;} y \mathbin{;} w)^\omega$
    by (*metis add-commutative omega-decompose omega-loop-fixpoint*)
  also have $\ldots \leq (x^* \mathbin{;} y \mathbin{;} w)^* \mathbin{;} x^\omega + (x^* \mathbin{;} y \mathbin{;} w)^\omega$
    by (*metis add-left-isotone mult-associative mult-right-isotone omega-sub-vector*)
  finally have *1*: $w \mathbin{;} (x^\omega + x^* \mathbin{;} y \mathbin{;} w)^\omega \leq (w \mathbin{;} x^* \mathbin{;} y)^* \mathbin{;} w \mathbin{;} x^\omega + (w \mathbin{;} x^* \mathbin{;} y)^\omega$
    by (*smt add-commutative less-eq-def mult-associative mult-left-dist-add while-def while-slide*)
  have $(x^\omega + x^* \mathbin{;} y \mathbin{;} w)^* \mathbin{;} z = (x^* \mathbin{;} y \mathbin{;} w)^* \mathbin{;} x^\omega \mathbin{;} ((x^* \mathbin{;} y \mathbin{;} w)^* \mathbin{;} x^\omega)^* \mathbin{;} (x^* \mathbin{;} y \mathbin{;} w)^* \mathbin{;} z + (x^* \mathbin{;} y \mathbin{;} w)^* \mathbin{;} z$
    by (*smt add-commutative mult-associative star.circ-add star.circ-loop-fixpoint*)
  also have $\ldots \leq (x^* \mathbin{;} y \mathbin{;} w)^* \mathbin{;} x^\omega + (x^* \mathbin{;} y \mathbin{;} w)^* \mathbin{;} z$
    by (*smt add-commutative add-right-isotone mult-associative mult-right-isotone omega-sub-vector*)
  finally have $w \mathbin{;} (x^\omega + x^* \mathbin{;} y \mathbin{;} w)^* \mathbin{;} z \leq (w \mathbin{;} x^* \mathbin{;} y)^* \mathbin{;} w \mathbin{;} x^\omega + (w \mathbin{;} x^* \mathbin{;} y)^* \mathbin{;} w \mathbin{;} z$
    by (*metis mult-associative mult-left-dist-add mult-right-isotone star.circ-slide*)
  hence $w \mathbin{;} (x^\omega + x^* \mathbin{;} y \mathbin{;} w)^\omega + w \mathbin{;} (x^\omega + x^* \mathbin{;} y \mathbin{;} w)^* \mathbin{;} z \leq (w \mathbin{;} x^* \mathbin{;} y)^* \mathbin{;} (w \mathbin{;} x^\omega)^\omega + (w \mathbin{;} x^* \mathbin{;} y)^\omega + (w \mathbin{;} x^* \mathbin{;} y)^* \mathbin{;} w \mathbin{;} z$ **using** *1*
    by (*smt add-associative add-commutative less-eq-def mult-associative tarski-mult-omega-omega*)
  also have $\ldots \leq (w \mathbin{;} x^\omega + w \mathbin{;} x^* \mathbin{;} y)^* \mathbin{;} (w \mathbin{;} x^\omega + w \mathbin{;} x^* \mathbin{;} y)^\omega + (w \mathbin{;} x^\omega + w \mathbin{;} x^* \mathbin{;} y)^\omega + (w \mathbin{;} x^\omega + w \mathbin{;} x^* \mathbin{;} y)^* \mathbin{;} w \mathbin{;} z$
    by (*metis add-isotone add-left-upper-bound add-right-upper-bound mult-isotone mult-left-isotone omega-isotone star.circ-isotone*)
  also have $\ldots = (w \mathbin{;} x^\omega + w \mathbin{;} x^* \mathbin{;} y)^\omega + (w \mathbin{;} x^\omega + w \mathbin{;} x^* \mathbin{;} y)^* \mathbin{;} w \mathbin{;} z$
    by (*metis add-idempotent star-mult-omega*)
  finally have $w \mathbin{;} ((x^\omega + x^* \mathbin{;} y \mathbin{;} w)^\omega + (x^\omega + x^* \mathbin{;} y \mathbin{;} w)^* \mathbin{;} z) \leq w \mathbin{;} ((x^\omega + x^* \mathbin{;} y) \mathbin{;} w)^\omega + w \mathbin{;} ((x^\omega + x^* \mathbin{;} y) \mathbin{;} w)^* \mathbin{;} z$
    by (*smt mult-associative mult-left-dist-add omega-slide star.circ-slide*)
  hence *2*: $w \mathbin{;} ((x * (y \mathbin{;} w)) * z) \leq w \mathbin{;} (((x * y) \mathbin{;} w) * z)$

by (*smt mult-associative mult-left-dist-add while-def while-slide*)
have $w \ ; (((x * y) \ ; w) * z) \leq w \ ; ((x * (y \ ; w)) * z)$
by (*metis mult-right-isotone while-left-isotone while-sub-associative*)
thus *?thesis* **using** *2*
by (*metis antisym*)
qed

lemma *while-denest-3*: $(x * w) * x^\omega = (x * w)^\omega$
proof −
have *1*: $(x * w) * x^\omega = (x * w)^\omega + (x * w)^* \ ; x^{\omega\omega}$
by (*metis tarski-omega-idempotent while-def*)
also have ... $\leq (x * w)^\omega + (x * w)^* \ ; (x^\omega + x^* \ ; w)^\omega$
by (*metis add-left-upper-bound add-right-isotone mult-right-isotone omega-isotone*)
also have ... $= (x * w)^\omega$
by (*metis add-idempotent star-mult-omega while-def*)
finally show *?thesis* **using** *1*
by (*metis add-left-upper-bound antisym-conv*)
qed

lemma *while-denest-4a*: $(x * w) * (x * (y \ ; z)) = (x * w) * ((x * y) \ ; z)$
proof −
have $(x * w) * (x * (y \ ; z)) = (x * w)^\omega + ((x * w) * (x^* \ ; y \ ; z))$
by (*smt mult-associative while-denest-3 while-def while-left-dist-add*)
also have ... $\leq (x * w)^\omega + ((x * w) * ((x * y) \ ; z))$
by (*metis add-right-isotone mult-left-isotone star-below-while while-right-isotone*)
finally have *1*: $(x * w) * (x * (y \ ; z)) \leq (x * w) * ((x * y) \ ; z)$
by (*smt add-left-upper-bound less-eq-def while-def*)
have $(x * w) * ((x * y) \ ; z) \leq (x * w) * (x * (y \ ; z))$
by (*metis while-right-isotone while-sub-associative*)
thus *?thesis* **using** *1*
by (*metis antisym*)
qed

subclass *binary-itering-1-T*
apply *unfold-locales*
apply (*smt mult-associative while-denest-2a while-denest-4a while-increasing while-slide*)
done

lemma *while-mult-top*: $(x \ ; T) * z = z + x \ ; T$
proof −
have *1*: $z + x \ ; T \leq (x \ ; T) * z$
by (*metis add-least-upper-bound while-denest-1 while-increasing while-one-top*)
have $(x \ ; T) * z = z + x \ ; T \ ; ((x \ ; T) * z)$
by (*metis while-left-unfold*)
also have ... $\leq z + x \ ; T$
by (*metis add-right-isotone mult-associative mult-right-isotone top-greatest*)
finally show *?thesis* **using** *1*

by (*metis antisym*)
qed

lemma *tarski-top-omega-below-2*: $x\ ;\ T \leq (x\ ;\ T) * 0$
by (*metis add-right-divisibility while-mult-top*)

lemma *tarski-top-omega-2*: $x\ ;\ T = (x\ ;\ T) * 0$
by (*metis add-left-zero while-mult-top*)

lemma *tarski-below-top-omega-2*: $x \leq (x\ ;\ T) * 0$
by (*metis tarski-top-omega-2 top-right-mult-increasing*)

end

class *nonstrict-itering-tarski-zero* = *nonstrict-itering-tarski* + *nonstrict-itering-zero*

begin

lemma $1 = (x\ ;\ 0) * 1$
by (*metis mult-right-zero while-zero*)

end

sublocale *binary-itering-1* < *idl-conway-semiring* **where** *circ* = $(\lambda x\ .\ x * 1)$
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* = *binary-itering-T* + *idl-semiring-lattice-apx*

begin

lemma *n-while-import*: $n(y)\ ;\ x \leq x\ ;\ n(y) \rightarrow n(y)\ ;\ (x * z) = n(y)\ ;\ ((n(y)\ ;\ x) * z)$
by (*metis while-import n-mult-idempotent n-sub-one order-refl*)

lemma *n-while-preserve*: $n(y)\ ;\ x \leq x\ ;\ n(y) \rightarrow n(y)\ ;\ (x * z) = n(y)\ ;\ (x * (n(y)\ ;\ z))$
by (*metis while-preserve n-mult-idempotent n-sub-one order-refl*)

lemma *while-L-L*: $L * L = L$
by (*metis n-L-top-L while-mult-star-exchange while-right-top*)

lemma *while-L-below-add*: $L * x \leq x + L$
by (*metis while-left-unfold add-right-isotone n-L-below-L*)

lemma *while-L-split*: $x * L \leq (x * y) + L$
proof −
have $x * L \leq (x * 0) + L$
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*: $x * (n(x * y)\ ;\ T) \leq (x * 0) + n(x * y)\ ;\ T$
proof −
have $x\ ;\ n(x * y)\ ;\ T \leq x\ ;\ 0 + n(x\ ;\ (x * y))\ ;\ T$
by (*metis n-n-top-split-n-top*)
also have $... \leq n(x * y)\ ;\ T + x\ ;\ 0$
by (*metis add-commutative add-right-isotone mult-left-isotone n-isotone while-left-plus-below*)
finally have $x * (n(x * y)\ ;\ T) \leq n(x * y)\ ;\ T + (x * (x\ ;\ 0))$
by (*metis mult-associative mult-right-one while-simulate-left mult-left-zero while-left-top*)
also have $... \leq (x * 0) + n(x * y)\ ;\ T$
by (*metis add-least-upper-bound add-left-isotone while-right-plus-below*)
finally show *?thesis*
by *metis*
qed

lemma *circ-apx-right-isotone*: $x \sqsubseteq y \rightarrow z * x \sqsubseteq z * y$
proof
assume $x \sqsubseteq y$
hence *1*: $x \leq y + L \wedge n(L)\ ;\ y \leq x + n(x)\ ;\ T$
by (*metis apx-def*)
hence $z * x \leq (z * y) + (z * L)$
by (*metis while-left-dist-add while-right-isotone*)
hence *2*: $z * x \leq (z * y) + L$
by (*smt add-least-upper-bound add-left-upper-bound while-L-split order-trans*)
have $z * (n(z * x)\ ;\ T) \leq (z * 0) + n(z * x)\ ;\ T$
by (*metis while-n-while-top-split*)
also have $... \leq (z * x) + n(z * x)\ ;\ T$
by (*metis add-left-isotone while-right-isotone zero-least*)
finally have *3*: $z * (n(x)\ ;\ T) \leq (z * x) + n(z * x)\ ;\ T$
by (*metis mult-left-isotone n-isotone order-trans while-increasing while-right-isotone*)
have $n(L)\ ;\ (z * y) \leq z * (n(L)\ ;\ y)$
by (*metis n-nL-semi-commute while-simulate*)
also have $... \leq (z * x) + (z * (n(x)\ ;\ T))$ **using** *1*

by (*metis while-left-dist-add while-right-isotone*)
also have ... $\leq (z * x) + n(z * x)\ ;\ T$ **using** *3*
by (*metis add-least-upper-bound add-left-upper-bound*)
finally show $z * x \sqsubseteq z * y$ **using** *2*
by (*metis apx-def*)
qed

end

class *binary-itering-1-apx* = *binary-itering-apx* + *binary-itering-1-T* +
assumes *n-below-while-zero*: $n(x) \leq n(x * 0)$

begin

lemma *circ-apx-right-isotone*: $x \sqsubseteq y \rightarrow x * z \sqsubseteq y * z$
proof
assume $x \sqsubseteq y$
hence *1*: $x \leq y + L \wedge n(L)\ ;\ y \leq x + n(x)\ ;\ T$
by (*metis apx-def*)
hence $x * z \leq ((y * 1)\ ;\ L) * (y * z)$
by (*metis while-left-isotone while-sumstar-3*)
also have ... $\leq (y * z) + (y * 1)\ ;\ L$
by (*metis while-productstar add-right-isotone mult-right-isotone n-L-below-L while-slide*)
also have ... $\leq (y * z) + L$
by (*metis add-commutative add-least-upper-bound add-right-upper-bound order-trans while-L-split while-one-mult-below*)
finally have *2*: $x * z \leq (y * z) + L$
by *metis*
have $n(L)\ ;\ (y * z) \leq (n(L)\ ;\ y) * z$
by (*metis n-nL-semi-commute n-while-import n-sub-one mult-left-one mult-left-isotone*)
also have ... $\leq ((x * 1)\ ;\ n(x)\ ;\ T) * (x * z)$ **using** *1*
by (*metis while-left-isotone mult-associative while-sumstar-3*)
also have ... $\leq (x * z) + (x * 1)\ ;\ n(x)\ ;\ T$
by (*metis while-productstar add-left-top add-right-isotone mult-associative mult-left-sub-dist-add-right while-slide*)
also have ... $\leq (x * z) + (x * (n(x)\ ;\ T))$
by (*metis add-right-isotone mult-associative while-one-mult-below*)
also have ... $\leq (x * z) + (x * (n(x * z)\ ;\ T))$
by (*metis n-below-while-zero zero-least while-right-isotone n-isotone mult-left-isotone add-right-isotone order-trans*)
also have ... $\leq (x * z) + n(x * z)\ ;\ T$
by (*smt add-associative add-right-isotone while-n-while-top-split add-right-zero while-left-dist-add*)
finally show $x * z \sqsubseteq y * z$ **using** *2*
by (*metis apx-def*)
qed

end

class *sdloa-apx-binary* = *sdloa-apx-extra* + *while* +
  assumes *while-def*: $x * y = n(x^\omega)\ ;\ L + x^*\ ;\ y$

begin

lemma *while-omega-meet-L-star*: $x * y = (x^\omega \frown L) + x^*\ ;\ y$
  by (*metis loop-semantics-xi-mu-nu loop-semantics-xi-mu-nu-2 while-def*)

lemma *while-one-mult-while-below-1*: $(y * 1)\ ;\ (y * v) \leq y * v$
proof −
  have $(y * 1)\ ;\ (y * v) \leq y * (y * v)$
    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 ... $= n(y^\omega)\ ;\ L + y^*\ ;\ n(y^\omega)\ ;\ L + y^*\ ;\ y^*\ ;\ v$
    by (*metis while-def mult-left-dist-add add-associative mult-associative*)
  also have ... $= n(y^\omega)\ ;\ L + n(y^*\ ;\ y^\omega)\ ;\ L + y^*\ ;\ y^*\ ;\ v$
    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*: $x^*\ ;\ y \leq x * y$
  by (*metis add-right-upper-bound while-def*)

subclass *binary-itering-T*
proof *unfold-locales*
  fix *x y z*
  have $z + x\ ;\ ((y\ ;\ x) * (y\ ;\ z)) = x\ ;\ (y\ ;\ x)^*\ ;\ y\ ;\ z + x\ ;\ n((y\ ;\ x)^\omega)\ ;\ L + z$
    by (*smt add-associative add-commutative mult-associative mult-left-dist-add while-def*)
  also have ... $= x\ ;\ (y\ ;\ x)^*\ ;\ y\ ;\ z + n(x\ ;\ (y\ ;\ x)^\omega)\ ;\ L + z$
    by (*metis mult-associative mult-right-isotone zero-least n-mult-omega-L-star-zero add-relative-same-increasing*)
  also have ... $= (x\ ;\ y)^*\ ;\ z + n(x\ ;\ (y\ ;\ x)^\omega)\ ;\ L$
    by (*smt add-associative add-commutative mult-associative star.circ-loop-fixpoint star-slide*)
  also have ... $= (x\ ;\ y) * z$
    by (*smt omega-slide while-def add-commutative*)
  finally show $(x\ ;\ y) * z = z + x\ ;\ ((y\ ;\ x) * (y\ ;\ z))$
    by *metis*
next
  fix *x y z*
  have $(x * y) * (x * z) = n((n(x^\omega)\ ;\ L + x^*\ ;\ y)^\omega)\ ;\ L + (n(x^\omega)\ ;\ L + x^*\ ;\ y)^*\ ;\ (x * z)$
    by (*metis while-def*)
  also have ... $= n((x^*\ ;\ y)^\omega + (x^*\ ;\ y)^*\ ;\ n(x^\omega)\ ;\ L)\ ;\ L + ((x^*\ ;\ y)^* + (x^*\ ;\ y)^*\ ;\ n(x^\omega)\ ;\ L)\ ;\ (x * z)$
    by (*metis mult-L-add-star mult-L-add-omega*)
  also have ... $= n((x^*\ ;\ y)^\omega)\ ;\ L + n((x^*\ ;\ y)^*\ ;\ n(x^\omega)\ ;\ L)\ ;\ L + (x^*\ ;\ y)^*\ ;\ (x * z) + (x^*\ ;\ y)^*\ ;\ n(x^\omega)\ ;\ L\ ;\ (x * z)$
    by (*metis mult-associative n-dist-omega-star mult-right-dist-add add-associative*)
  also have ... $= n((x^*\ ;\ y)^\omega)\ ;\ L + n((x^*\ ;\ y)^*\ ;\ n(x^\omega)\ ;\ L)\ ;\ L + (x^*\ ;\ y)^*\ ;\ 0 + (x^*\ ;\ y)^*\ ;\ (x * z) + (x^*\ ;\ y)^*\ ;\ n(x^\omega)\ ;\ L\ ;\ (x * z)$

by (*smt add-associative add-left-zero mult-left-dist-add*)
also have ... = $n((x^* \ ; \ y)^\omega) \ ; \ L + ((x^* \ ; \ y)^* \ ; \ n(x^\omega) \ ; \ L \ ; \ (x * z) + (x^* \ ; \ y)^* \ ; \ n(x^\omega) \ ; \ L + (x^* \ ; \ y)^* \ ; \ (x * z))$
by (*smt n-n-L-split-n-n-L-L add-commutative add-associative*)
also have ... = $n((x^* \ ; \ y)^\omega) \ ; \ L + ((x^* \ ; \ y)^* \ ; \ n(x^\omega) \ ; \ L + (x^* \ ; \ y)^* \ ; \ (x * z))$
by (*smt mult-L-omega omega-sub-vector less-eq-def*)
also have ... = $n((x^* \ ; \ y)^\omega) \ ; \ L + (x^* \ ; \ y)^* \ ; \ (x * z)$
by (*metis add-left-divisibility mult-associative mult-right-isotone while-def less-eq-def*)
also have ... = $(x^* \ ; \ y)^* \ ; \ x^* \ ; \ z + (x^* \ ; \ y)^* \ ; \ n(x^\omega) \ ; \ L + n((x^* \ ; \ y)^\omega) \ ; \ L$
by (*metis add-commutative mult-associative mult-left-dist-add while-def*)
also have ... = $(x^* \ ; \ y)^* \ ; \ x^* \ ; \ z + n((x^* \ ; \ y)^* \ ; \ x^\omega) \ ; \ L + n((x^* \ ; \ y)^\omega) \ ; \ L$
by (*metis add-relative-same-increasing add-right-divisibility mult-right-isotone n-mult-omega-L-star-zero zero-least*)
also have ... = $(x + y) * z$
by (*metis add-associative add-commutative omega-decompose star.circ-add while-def mult-right-dist-add n-dist-omega-star*)
finally show $(x + y) * z = (x * y) * (x * z)$
by *metis*
next
fix $x \ y \ z$
show $x * (y + z) = (x * y) + (x * z)$
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 $(x * y) \ ; \ z \leq x * (y \ ; \ z)$
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 $x \ ; \ z \leq z \ ; \ (y * 1) + w \rightarrow x * (z \ ; \ v) \leq z \ ; \ (y * v) + (x * (w \ ; \ (y * v)))$
proof
assume *1*: $x \ ; \ z \leq z \ ; \ (y * 1) + w$
have $z \ ; \ v + x \ ; \ (z \ ; \ (y * v) + x^* \ ; \ (w \ ; \ (y * v))) \leq z \ ; \ v + x \ ; \ z \ ; \ (y * v) + x^* \ ; \ (w \ ; \ (y * v))$
by (*metis add-associative add-right-isotone mult-associative mult-left-dist-add mult-left-isotone star.left-plus-below-circ*)
also have ... $\leq z \ ; \ v + z \ ; \ (y * 1) \ ; \ (y * v) + w \ ; \ (y * v) + x^* \ ; \ (w \ ; \ (y * v))$ **using** *1*
by (*metis add-associative add-left-isotone add-right-isotone mult-left-isotone mult-right-dist-add*)
also have ... $\leq z \ ; \ v + z \ ; \ (y * v) + x^* \ ; \ (w \ ; \ (y * v))$
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 ... = $z \ ; \ (y * v) + x^* \ ; \ (w \ ; \ (y * v))$
by (*metis add-commutative 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 $x^* \ ; \ z \ ; \ v \leq z \ ; \ (y * v) + x^* \ ; \ (w \ ; \ (y * v))$
by (*metis mult-associative star-left-induct*)
thus $x * (z \ ; \ v) \leq z \ ; \ (y * v) + (x * (w \ ; \ (y * v)))$
by (*smt add-associative add-commutative add-right-isotone mult-associative while-def*)
qed
next
fix $v \ w \ x \ y \ z$
show $z \ ; \ x \leq y \ ; \ (y * z) + w \rightarrow z \ ; \ (x * v) \leq y * (z \ ; \ v + w \ ; \ (x * v))$
proof
assume $z \ ; \ x \leq y \ ; \ (y * z) + w$
hence *1*: $z \ ; \ x \leq y \ ; \ y^* \ ; \ z + (y \ ; \ n(y^\omega) \ ; \ L + w)$

by (*smt add-associative add-commutative mult-associative mult-left-dist-add while-def*)
hence $z\ ;\ x^*\ \leq\ y^*\ ;\ (z\ +\ (y\ ;\ n(y^\omega)\ ;\ L\ +\ w)\ ;\ x^*)$
by (*metis star.circ-simulate-right-plus*)
also have $\ldots\ =\ y^*\ ;\ z\ +\ y^*\ ;\ y\ ;\ n(y^\omega)\ ;\ L\ +\ y^*\ ;\ w\ ;\ x^*$
by (*smt add-associative mult-associative mult-left-dist-add mult-right-dist-add L-mult-star*)
also have $\ldots\ =\ y^*\ ;\ z\ +\ n(y^*\ ;\ y\ ;\ y^\omega)\ ;\ L\ +\ y^*\ ;\ w\ ;\ x^*$
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 $\ldots\ =\ n(y^\omega)\ ;\ L\ +\ y^*\ ;\ z\ +\ y^*\ ;\ w\ ;\ x^*$
by (*metis add-commutative omega-unfold right-plus-omega*)
finally have $z\ ;\ x^*\ ;\ v\ \leq\ n(y^\omega)\ ;\ L\ ;\ v\ +\ y^*\ ;\ z\ ;\ v\ +\ y^*\ ;\ w\ ;\ x^*\ ;\ v$
by (*smt less-eq-def mult-right-dist-add*)
also have $\ldots\ \leq\ n(y^\omega)\ ;\ L\ +\ y^*\ ;\ (z\ ;\ v\ +\ w\ ;\ x^*\ ;\ v)$
by (*metis n-L-below-L mult-associative mult-right-isotone add-left-isotone mult-left-dist-add add-associative*)
also have $\ldots\ \leq\ n(y^\omega)\ ;\ L\ +\ y^*\ ;\ (z\ ;\ v\ +\ w\ ;\ (x\ *\ v))$
by (*metis add-commutative add-right-isotone mult-associative mult-left-sub-dist-add-left mult-right-isotone while-def*)
finally have *2*: $z\ ;\ x^*\ ;\ v\ \leq\ y\ *\ (z\ ;\ v\ +\ w\ ;\ (x\ *\ v))$
by (*metis while-def*)
have *3*: $y^*\ ;\ y\ ;\ y^*\ ;\ 0\ \leq\ y^*\ ;\ w\ ;\ x^\omega$
by (*metis add-commutative add-left-zero mult-associative mult-left-sub-dist-add-left star.circ-loop-fixpoint star.circ-transitive-equal*)
have $z\ ;\ x^\omega\ \leq\ y\ ;\ y^*\ ;\ z\ ;\ x^\omega\ +\ (y\ ;\ n(y^\omega)\ ;\ L\ +\ w)\ ;\ x^\omega$ **using** *1*
by (*metis mult-associative mult-left-isotone mult-right-dist-add omega-unfold*)
hence $z\ ;\ x^\omega\ \leq\ y^\omega\ +\ y^*\ ;\ y\ ;\ n(y^\omega)\ ;\ L\ ;\ x^\omega\ +\ y^*\ ;\ w\ ;\ x^\omega$
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 $\ldots\ \leq\ y^\omega\ +\ y^*\ ;\ y\ ;\ n(y^\omega)\ ;\ L\ +\ y^*\ ;\ w\ ;\ x^\omega$
by (*metis add-left-isotone add-right-isotone mult-associative mult-right-isotone n-L-below-L*)
also have $\ldots\ =\ y^\omega\ +\ n(y^*\ ;\ y\ ;\ y^\omega)\ ;\ L\ +\ y^*\ ;\ w\ ;\ x^\omega$ **using** *3*
by (*smt add-associative add-commutative add-relative-same-increasing n-mult-omega-L-star-zero*)
also have $\ldots\ =\ y^\omega\ +\ y^*\ ;\ w\ ;\ x^\omega$
by (*metis mult-associative omega-unfold star-mult-omega add-commutative less-eq-def n-L-decreasing*)
finally have $n(z\ ;\ x^\omega)\ ;\ L\ \leq\ n(y^\omega)\ ;\ L\ +\ n(y^*\ ;\ w\ ;\ x^\omega)\ ;\ L$
by (*metis mult-associative mult-left-isotone mult-right-dist-add n-dist-omega-star n-isotone*)
also have $\ldots\ \leq\ n(y^\omega)\ ;\ L\ +\ y^*\ ;\ (w\ ;\ (n(x^\omega)\ ;\ L\ +\ x^*\ ;\ 0))$
by (*smt add-commutative add-right-isotone mult-associative mult-left-dist-add n-mult-omega-L-below-zero*)
also have $\ldots\ \leq\ n(y^\omega)\ ;\ L\ +\ y^*\ ;\ (w\ ;\ (n(x^\omega)\ ;\ L\ +\ x^*\ ;\ v))$
by (*metis add-right-isotone mult-right-isotone zero-least*)
also have $\ldots\ \leq\ n(y^\omega)\ ;\ L\ +\ y^*\ ;\ (z\ ;\ v\ +\ w\ ;\ (n(x^\omega)\ ;\ L\ +\ x^*\ ;\ v))$
by (*metis add-right-isotone mult-left-sub-dist-add-right*)
finally have *4*: $n(z\ ;\ x^\omega)\ ;\ L\ \leq\ y\ *\ (z\ ;\ v\ +\ w\ ;\ (x\ *\ v))$
by (*metis while-def*)
have $z\ ;\ (x\ *\ v)\ =\ z\ ;\ n(x^\omega)\ ;\ L\ +\ z\ ;\ x^*\ ;\ v$
by (*metis while-def mult-left-dist-add mult-associative*)
also have $\ldots\ =\ n(z\ ;\ x^\omega)\ ;\ L\ +\ z\ ;\ x^*\ ;\ v$
by (*metis add-commutative add-relative-same-increasing mult-right-isotone n-mult-omega-L-star-zero zero-least*)
finally show $z\ ;\ (x\ *\ v)\ \leq\ y\ *\ (z\ ;\ v\ +\ w\ ;\ (x\ *\ v))$ **using** *2 4*
by (*metis add-least-upper-bound*)
qed
qed

lemma *while-top*: $T * x = L + T \ ; x$
  by (*metis n-top-L star.circ-top star-omega-top while-def*)

lemma *while-one-top*: $1 * x = L + x$
  by (*smt mult-left-one n-top-L omega-one star-one while-def*)

lemma *while-finite-associative*: $x^\omega = 0 \rightarrow (x * y) \ ; z = x * (y \ ; z)$
  by (*metis add-left-zero mult-associative n-zero-L-zero while-def*)

lemma *while-while-one*: $y * (x * 1) = n(y^\omega) \ ; L + y^* \ ; n(x^\omega) \ ; L + y^* \ ; x^*$
  by (*metis add-associative mult-left-dist-add mult-right-one while-def mult-associative*)

subclass *binary-itering-1-T*
proof *unfold-locales*
  fix *w x y z*
  have $w \ ; (x * y \ ; z) = n(w \ ; n(x^\omega) \ ; L) \ ; L + w \ ; x^* \ ; y \ ; z$
    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 $... \leq n((w \ ; n(x^\omega) \ ; L)^\omega) \ ; L + w \ ; x^* \ ; y \ ; z$
    by (*metis eq-refl mult-L-omega*)
  also have $... \leq n((w \ ; (x * y))^\omega) \ ; L + w \ ; x^* \ ; y \ ; z$
    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 $... \leq n((w \ ; (x * y))^\omega) \ ; L + w \ ; (x * y) \ ; z$
    by (*metis star-below-while mult-associative mult-left-isotone mult-right-isotone add-right-isotone*)
  also have $... \leq n((w \ ; (x * y))^\omega) \ ; L + (w \ ; (x * y))^* \ ; (w \ ; (x * y) \ ; z)$
    by (*metis add-right-isotone add-right-upper-bound star.circ-loop-fixpoint*)
  finally show $w \ ; (x * y \ ; z) \leq (w \ ; (x * y)) * (w \ ; (x * y) \ ; z)$
    by (*metis while-def*)
qed

subclass *binary-itering-1-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*: $y \ ; x \leq x \ ; (x * (1 + y)) \rightarrow y \ ; x \ ; x^* \leq x \ ; (x * (1 + y))$
proof
  assume *1*: $y \ ; x \leq x \ ; (x * (1 + y))$
  have $x \ ; (x * (1 + y)) = x \ ; n(x^\omega) \ ; L + x \ ; x^* \ ; (1 + y)$
    by (*metis mult-associative mult-left-dist-add while-def*)
  also have $... = n(x \ ; x^\omega) \ ; L + x \ ; x^* \ ; (1 + y)$
    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*: $x \ ; (x * (1 + y)) = n(x^\omega) \ ; L + x \ ; x^* + x \ ; x^* \ ; y$
    by (*metis add-associative mult-left-dist-add mult-right-one omega-unfold*)
  hence $x \ ; x^* \ ; y \ ; x \leq x \ ; x^* \ ; n(x^\omega) \ ; L + x \ ; x^* \ ; x^* \ ; x + x \ ; x^* \ ; x \ ; x^* \ ; y$ **using** *1*
    by (*metis mult-associative mult-right-isotone mult-left-dist-add star-plus*)
  also have $... = n(x \ ; x^* \ ; x^\omega) \ ; L + x \ ; x^* \ ; x^* \ ; x + x \ ; x^* \ ; x \ ; x^* \ ; y$

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^* ; $x + x$; x ; x^* ; y
by (*metis mult-associative omega-unfold star.circ-plus-same star.circ-transitive-equal star-mult-omega*)
also have ... ≤ $n(x^\omega)$; $L + x$; $x^* + x$; x^* ; 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^* ; y ; $x \leq n(x^\omega)$; $L + x$; $x^* + x$; x^* ; y
by *metis*
have ($n(x^\omega)$; $L + x$; $x^* + x$; x^* ; y) ; $x \leq n(x^\omega)$; $L + x$; x^* ; $x + x$; x^* ; y ; x
by (*metis mult-right-dist-add n-L-below-L mult-associative mult-right-isotone add-left-isotone*)
also have ... ≤ $n(x^\omega)$; $L + x$; $x^* + x$; x^* ; y ; x
by (*smt add-commutative add-left-isotone mult-associative mult-right-isotone star.left-plus-below-circ star-plus*)
also have ... ≤ $n(x^\omega)$; $L + x$; $x^* + x$; x^* ; y **using** *3*
by (*metis add-least-upper-bound add-left-upper-bound*)
finally show y ; x ; $x^* \leq x$; $(x * (1 + y))$ **using** *1 2*
by (*metis add-least-upper-bound star-right-induct*)
qed

lemma *while-simulate-4-omega*: y ; $x \leq x$; $(x * (1 + y)) \rightarrow y$; $x^\omega \leq x^\omega$
proof
assume *1*: y ; $x \leq x$; $(x * (1 + y))$
have x ; $(x * (1 + y)) = x$; $n(x^\omega)$; $L + x$; x^* ; $(1 + y)$
by (*metis mult-associative mult-left-dist-add while-def*)
also have ... = $n(x$; $x^\omega)$; $L + x$; x^* ; $(1 + y)$
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 ; $(x * (1 + y)) = n(x^\omega)$; $L + x$; $x^* + x$; x^* ; 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^* ; $x^\omega + x$; x^* ; y ; x^ω **using** *1*
by (*smt less-eq-def mult-associative mult-right-dist-add omega-unfold*)
also have ... ≤ x ; x^* ; $(y$; $x^\omega) + x^\omega$
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 ; $x^\omega \leq (x$; $x^*)^\omega + (x$; $x^*)^*$; x^ω
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*: $x * 1 = (x$; $x) * (x + 1)$
by (*metis mult-right-one omega-square star-square-2 while-def*)

lemma *while-absorb-below-one*: y ; $x \leq x \rightarrow y * x \leq 1 * x$
by (*metis star-left-induct-mult add-isotone n-galois n-sub-nL while-def while-one-top*)

lemma *while-mult-L*: $(x$; $L) * z = z + x$; 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 ; $L \leq (x$; $L) * 0$
by (*metis add-right-divisibility while-mult-L*)

lemma *tarski-top-omega-2*: $x \ ; \ L = (x \ ; \ L) * 0$
  by (*metis add-left-zero while-mult-L*)

end

end