Communicating Concurrent Kleene Algebra for Distributed Systems Specification

Maxime Buyse and Jason Jaskolka

February 23, 2021

Abstract

Communicating Concurrent Kleene Algebra (C^2KA) is a mathematical framework for capturing the communicating and concurrent behaviour of agents in distributed systems. It extends Hoare et al.'s Concurrent Kleene Algebra (CKA) with communication actions through the notions of stimuli and shared environments. C^2KA has applications in studying system-level properties of distributed systems such as safety, security, and reliability. In this work, we formalize results about C^2KA and its application for distributed systems specification. We first formalize the stimulus structure and behaviour structure (CKA). Next, we combine them to formalize C^2KA and its properties. Then, we formalize notions and properties related to the topology of distributed systems and the potential for communication via stimuli and via shared environments of agents, all within the algebraic setting of C^2KA.

Contents

1 Introduction

Most complex distributed systems participate in intensive communication and exchange with their environment, which often includes other systems. For example, many systems need input in terms of energy, resources, information, etc. As a result, the interactions between a system and its environment need to be carefully taken into account when modeling such systems.

In a distributed system, agents can communicate via their shared environments in the form of shared-variable communication where they transfer information through a shared medium (e.g., variables, buffers, etc.) and through their local communication channels in the form of message-passing communication where they transfer information explicitly through the exchange of data structures. However, the agents in the system may also be influenced by external stimuli. From the perspective of behaviourism, a *stimulus* constitutes the basis for behaviour. In this way, agent behaviour can, in some situations, be explained without the need to consider the internal states of an agent. A *closed system* is one that does not receive any stimuli that affect its behaviour and that does not share any environment. A system that is not a closed system is called an *open system*. When dealing with open systems, external stimuli are required to initiate agent behaviours. Such external stimuli result from systems outside the boundaries of the considered system and may impact the way in which the system agents behave. It is important to note that every stimulus *invokes a response* from an agent. When the behaviour of an agent changes as a result of the response, we say that the stimulus *influences* the behaviour of the agent.

Communicating Concurrent Kleene Algebra (C^{2}KA) [2, 5] is a mathematical framework for capturing the communicating and concurrent behaviour of agents in distributed systems. In this work, the term *agent* is used to refer to any system, component, or process whose behaviour consists of discrete actions and each interaction, direct or indirect, of an agent with its neighbouring agents is called a *communication* as in [6]. C^{2}KA extends the algebraic model of Concurrent Kleene Algebra [1], with communication actions through the notions of stimuli and shared environments. It offers an algebraic setting capable of capturing both the influence of stimuli on agent behaviour as well as the communication and concurrency of agents in a system and its environment at an abstract algebraic level, thereby allowing it to capture the dynamic behaviour of complex distributed systems.

In this work, we follow Jaskolka's doctoral dissertation [2] which provides a full treatment of C^{2}KA and its related notions and properties. Section 2 and Section 3 formalize the stimulus structure and behaviour structure, respectively. These structures comprise the two primary components of a C^{2}KA. Section 4 then combines these notions to formalize C^{2}KA and its properties. Section 5 follows this by presenting a formalization of the notions of orbits, stabilisers, and fixed points to establish an understanding of the topology of a distributed system specified using C^{2}KA. Finally, Section 6 formalizes results regarding the potential for communication via stimuli and via shared environments of distributed system agents within the algebraic setting of C^{2}KA.

2 Stimulus Structure

A stimulus constitutes the basis for behaviour. Because of this, each discrete, observable event introduced to a system, such as that which occurs through the communication among agents or from the system environment, is considered to be a stimulus which invokes a response from each system agent.

A *stimulus structure* is an idempotent semiring $\left(S, \oplus, \odot, \mathfrak{d}, \mathfrak{n}\right)$ with a multiplicatively absorbing $\mathfrak{d}$ and identity $\mathfrak{n}$. Within the context of stimuli, S is a set of stimuli which may be introduced to a system. The operator $\oplus$ is interpreted as a choice between two stimuli and the operator $\odot$ is interpreted as a sequential composition of two stimuli. The element $\mathfrak{d}$ represents the *deactivation stimulus* which influences all agents to become inactive and the element $\mathfrak{n}$ represents the *neutral stimulus* which has no influence on the behaviour of all agents. The natural ordering relation $\leq_{\mathcal{S}}$ on a stimulus structure $\mathcal{S}$ is called the sub-stimulus relation. For stimuli $s, t \in S$, we write $s \leq_{\mathcal{S}} t$ and say that s is a sub-stimulus of t if and only if $s \oplus t = t$.

theory *Stimuli*
  imports *Main*
begin

The class *stimuli* describes the stimulus structure for C^{2}KA. We do not use Isabelle's built-in theories for groups and orderings to allow a different notation for the operations on stimuli to be consistent with [2].

class *plus-ord* =
  fixes *leq*::$'a \Rightarrow\ 'a \Rightarrow$ *bool* ((-/ $\leq_{\mathcal{S}}$ -) [*51*, *51*] *50*)

```
  fixes add::'a ⇒ 'a ⇒ 'a (infixl ⊕ 65)
  assumes leq-def: x ≤_S y ⟷ x ⊕ y = y
  and add-assoc: (x ⊕ y) ⊕ z = x ⊕ (y ⊕ z)
  and add-comm: x ⊕ y = y ⊕ x
begin

notation
  leq  ('(≤')) and
  leq ((-/ ≤_S -)  [51, 51] 50)

end

class stimuli = plus-ord +
  fixes seq-comp::'a ⇒ 'a ⇒ 'a (infixl ⊙ 70)
  fixes neutral :: 'a (𝔫)
  and deactivation :: 'a (𝔡)
  and basic :: 'a set (S_a)
  assumes stim-idem [simp]: x ⊕ x = x
  and seq-nl [simp]: 𝔫 ⊙ x = x
  and seq-nr [simp]: x ⊙ 𝔫 = x
  and add-zero [simp]: 𝔡 ⊕ x = x
  and absorbingl [simp]: 𝔡 ⊙ x = 𝔡
  and absorbingr [simp]: x ⊙ 𝔡 = 𝔡
  and zero-not-basic: 𝔡 ∉ S_a
begin

lemma inf-add-S-right: x ≤_S y ⟹ x ≤_S y ⊕ z
  unfolding leq-def
  by (simp add: add-assoc [symmetric])

lemma inf-add-S-left: x ≤_S y ⟹ x ≤_S z ⊕ y
  by (simp add: add-comm inf-add-S-right)

lemma leq-refl [simp]: x ≤_S x
  unfolding leq-def
  by simp

end

end
```

3 Behaviour Structure

Hoare et al. [1] presented the framework of Concurrent Kleene Algebra (CKA) which captures the concurrent behaviour of agents. The framework of CKA is adopted to describe agent behaviours in distributed systems. For a CKA $\left(K, +, *, ;, ^*, ^;, 0, 1\right)$, K is a set of possible behaviours. The operator $+$ is interpreted as a choice between two behaviours, the operator $;$

is interpreted as a sequential composition of two behaviours, and the operator $*$ is interpreted as a parallel composition of two behaviours. The operators $^{;}$ and * are interpreted as a finite sequential iteration and a finite parallel iteration of behaviours, respectively. The element 0 represents the behaviour of the *inactive agent* and the element 1 represents the behaviour of the *idle agent*. Associated with a CKA $\mathcal{K}$ is a natural ordering relation $\leq_{\mathcal{K}}$ related to the semirings upon which the CKA is built which is called the sub-behaviour relation. For behaviours $a, b \in K$, we write $a \leq_{\mathcal{K}} b$ and say that a is a sub-behaviour of b if and only if $a + b = b$.

theory *CKA*
  imports *Main*
begin

no-notation
rtrancl ((-*) [*1000*] *999*)

notation
times (**infixl** $*$ *70*)
and *less-eq* ($'(\leq_{\mathcal{K}}')$)
and *less-eq* ((-/ $\leq_{\mathcal{K}}$ -) [*51*, *51*] *50*)

The class *cka* contains an axiomatisation of Concurrent Kleene Algebras and a selection of useful theorems.

class *join-semilattice* = *ordered-ab-semigroup-add* +
  assumes *leq-def*: $x \leq y \longleftrightarrow x + y = y$
  and *le-def*: $x < y \longleftrightarrow x \leq y \wedge x \neq y$
  and *add-idem* [*simp*]: $x + x = x$
begin

lemma *inf-add-K-right*: $a \leq_{\mathcal{K}} a + b$
  unfolding *leq-def*
  by (*simp add*: *add-assoc*[*symmetric*])

lemma *inf-add-K-left*: $a \leq_{\mathcal{K}} b + a$
  by (*simp only*: *add-commute*, *fact inf-add-K-right*)

end

class *dioid* = *semiring* + *one* + *zero* + *join-semilattice* +
  assumes *par-onel* [*simp*]: $1 * x = x$
  and *par-oner* [*simp*]: $x * 1 = x$
  and *add-zerol* [*simp*]: $0 + x = x$
  and *annil* [*simp*]: $0 * x = 0$
  and *annir* [*simp*]: $x * 0 = 0$

class *kleene-algebra* = *dioid* +
  fixes *star* :: $'a \Rightarrow 'a$ (-* [*101*] *100*)
  assumes *star-unfoldl*: $1 + x * x^* \leq_{\mathcal{K}} x^*$

```
  and star-unfoldr: 1 + x* ∗ x ≤𝒦 x*
  and star-inductl: z + x ∗ y ≤𝒦 y ⟹ x* ∗ z ≤𝒦 y
  and star-inductr: z + y ∗ x ≤𝒦 y ⟹ z ∗ x* ≤𝒦 y

class cka = kleene-algebra +
  fixes seq :: 'a ⇒ 'a ⇒ 'a (infixl ; 70)
  and seqstar :: 'a ⇒ 'a (-^; [101] 100)
  assumes seq-assoc: x ; (y ; z) = (x ; y) ; z
  and seq-rident [simp]: x ; 1 = x
  and seq-lident [simp]: 1 ; x = x
  and seq-rdistrib [simp]: (x + y);z = x;z + y;z
  and seq-ldistrib [simp]: x;(y + z) = x;y + x;z
  and seq-annir [simp]: x ; 0 = 0
  and seq-annil [simp]: 0 ; x = 0
  and seqstar-unfoldl: 1 + x ; x^; ≤𝒦 x^;
  and seqstar-unfoldr: 1 + x^; ; x ≤𝒦 x^;
  and seqstar-inductl: z + x ; y ≤𝒦 y ⟹ x^; ; z ≤𝒦 y
  and seqstar-inductr: z + y ; x ≤𝒦 y ⟹ z ; x^; ≤𝒦 y
  and exchange: (a∗b) ; (c∗d) ≤𝒦 (b;c) ∗ (a;d)
begin

interpretation cka: kleene-algebra plus less-eq less 1 0 seq seqstar
   by (unfold-locales, simp-all add: seq-assoc seqstar-unfoldl seqstar-unfoldr seqs-
tar-inductl seqstar-inductr)

lemma par-comm: a ∗ b = b ∗ a
proof –
  have (b∗a) ; (1∗1) ≤𝒦 (a;1) ∗ (b;1) by (simp only:  exchange)
  hence b ∗ a ≤𝒦 a ∗ b by (simp)
  hence a ∗ b ≤𝒦 b ∗ a ⟷ a ∗ b  = b ∗ a by (rule antisym-conv)
  moreover have a ∗ b ≤𝒦 b ∗ a proof –
  have (a∗b) ; (1∗1) ≤𝒦 (b;1) ∗ (a;1) by (rule  exchange)
    thus ?thesis  by (simp)
  qed
  ultimately show ?thesis by (auto)
qed

lemma exchange-2: (a∗b) ; (c∗d) ≤𝒦 (a;c) ∗ (b;d)
proof –
  have (b∗a) ; (c∗d) ≤𝒦 (a;c) ∗ (b;d) by (rule exchange)
  thus ?thesis by (simp add: par-comm)
qed

lemma seq-inf-par: a ; b ≤𝒦 a ∗ b
proof –
  have (1∗a) ; (1∗b) ≤𝒦 (a;1) ∗ (1;b) by (rule exchange)
  thus ?thesis by simp
qed
```

lemma *add-seq-inf-par*: $a;b + b;a \leq_{\mathcal{K}} a*b$
proof −
  have $a;b \leq_{\mathcal{K}} a*b$ **by** (*rule seq-inf-par*)
  moreover have $b;a \leq_{\mathcal{K}} b*a$ **by** (*rule seq-inf-par*)
  ultimately have $a;b + b;a \leq_{\mathcal{K}} a*b + b*a$ **by** (*simp add*: *add-mono*)
  thus *?thesis* **by** (*simp add*: *par-comm*)
qed

lemma *exchange-3*: $(a*b) ; c \leq_{\mathcal{K}} a * (b;c)$
proof −
  have $(a*b) ; (1*c) \leq_{\mathcal{K}} (a;1) * (b;c)$ **by** (*rule exchange-2*)
  thus *?thesis* **by** *simp*
qed

lemma *exchange-4*: $a ; (b*c) \leq_{\mathcal{K}} (a;b) * c$
proof −
  have $(1*a) ; (b*c) \leq_{\mathcal{K}} (a;b) * (1;c)$ **by** (*rule exchange*)
  thus *?thesis* **by** *simp*
qed

lemma *seqstar-inf-parstar*: $a^{;} \leq_{\mathcal{K}} a^*$
proof −
  have $a ; a^* \leq_{\mathcal{K}} a * a^*$ **by** (*rule seq-inf-par*)
  hence $1 + a ; a^* \leq_{\mathcal{K}} 1 + a * a^*$ **by** (*simp add*: *add-left-mono*)
  hence $1 + a ; a^* \leq_{\mathcal{K}} a^*$ **by** (*simp add*: *star-unfoldl order-trans*)
  hence $a^{;} ; 1 \leq_{\mathcal{K}} a^*$ **by** (*rule seqstar-inductl*)
  thus *?thesis* **by** *simp*
qed

end

end

4 Communicating Concurrent Kleene Algebra

C^{2}KA extends the algebraic foundation of CKA with the notions of semimodules and stimulus structures to capture the influence of stimuli on the behaviour of system agents.

A C^{2}KA is a mathematical system consisting of two semimodules which describe how a stimulus structure $\mathcal{S}$ and a CKA $\mathcal{K}$ mutually act upon one another to characterize the response invoked by a stimulus on an agent behaviour as a next behaviour and a next stimulus. The left $\mathcal{S}$-semimodule $\left({}_{\mathcal{S}}K, +\right)$ describes how the stimulus structure $\mathcal{S}$ acts upon the CKA $\mathcal{K}$ via the mapping $\circ$. The mapping $\circ$ is called the *next behaviour mapping* and it describes how a stimulus invokes a behavioural response from a given agent. From $\left({}_{\mathcal{S}}K, +\right)$, the next behaviour mapping $\circ$ distributes over $+$

and $\oplus$. Additionally, since $\left({}_{\mathcal{S}}K, +\right)$ is unitary, it is the case that the neutral stimulus has no influence on the behaviour of all agents and since $\left({}_{\mathcal{S}}K, +\right)$ is zero-preserving, the deactivation stimulus influences all agents to become inactive. The right $\mathcal{K}$-semimodule $\left(S_{\mathcal{K}}, \oplus\right)$ describes how the CKA $\mathcal{K}$ acts upon the stimulus structure $\mathcal{S}$ via the mapping λ. The mapping λ is called the *next stimulus mapping* and it describes how a new stimulus is generated as a result of the response invoked by a given stimulus on an agent behaviour. From $\left(S_{\mathcal{K}}, \oplus\right)$, the next stimulus mapping λ distributes over $\oplus$ and $+$. Also, since $\left(S_{\mathcal{K}}, \oplus\right)$ is unitary, it is the case that the idle agent forwards any stimulus that acts on it and since $\left(S_{\mathcal{K}}, \oplus\right)$ is zero-preserving, the inactive agent always generates the deactivation stimulus. A full account of C^2KA can be found in [2, 4, 5].

theory *C2KA*
  imports *CKA Stimuli*
begin

no-notation
comp (**infixl** ∘ *55*)
and *rtrancl* ((-*) [*1000*] *999*)

The locale *c2ka* contains an axiomatisation of C^2KA and some basic theorems relying on the axiomatisations of stimulus structures and CKA provided in Sections 2 and 3, respectively. We use a locale instead of a class in order to allow stimuli and behaviours to have two different types.

locale *c2ka* =
  fixes *next-behaviour* :: *'b::stimuli* ⇒ *'a::cka* ⇒ *'a* (**infixr** ∘ *75*)
  and *next-stimulus* :: (*'b::stimuli* × *'a::cka*) ⇒ *'b* (λ)
  assumes *lsemimodule1* [*simp*]: $s \circ (a + b) = (s \circ a) + (s \circ b)$
  and *lsemimodule2* [*simp*]: $(s \oplus t) \circ a = (s \circ a) + (t \circ a)$
  and *lsemimodule3* [*simp*]: $(s \odot t) \circ a = s \circ (t \circ a)$
  and *lsemimodule4* [*simp*]: $\mathfrak{n} \circ a = a$
  and *lsemimodule5* [*simp*]: $\mathfrak{d} \circ a = 0$
  and *rsemimodule1* [*simp*]: $\lambda(s \oplus t,\ a) = \lambda(s,\ a) \oplus \lambda(t,\ a)$
  and *rsemimodule2* [*simp*]: $\lambda(s,\ a + b) = \lambda(s,\ a) \oplus \lambda(s,\ b)$
  and *rsemimodule3* [*simp*]: $\lambda(s,\ a\ ;\ b) = \lambda(\lambda(s,\ a),\ b)$
  and *rsemimodule4* [*simp*]: $\lambda(s,\ 1) = s$
  and *rsemimodule5* [*simp*]: $\lambda(s,\ 0) = \mathfrak{d}$
  and *cascadingaxiom* [*simp*]: $s \circ (a\ ;\ b) = (s \circ a);(\lambda(s,\ a) \circ b)$
  and *cascadingoutputlaw*: $a \leq_{\mathcal{K}} c \lor b = 1 \lor (s \circ a);(\lambda(s,c) \circ b) = 0$
  and *sequentialoutputlaw* [*simp*]: $\lambda(s \odot t,\ a) = \lambda(s,\ t \circ a) \odot \lambda(t,\ a)$
  and *onefix*: $s = \mathfrak{d} \lor s \circ 1 = 1$
  and *neutralunmodified*: $a = 0 \lor \lambda(\mathfrak{n},\ a) = \mathfrak{n}$
begin

Lemmas *inf-K-S-next-behaviour* and *inf-K-S-next-stimulus* show basic results from the axiomatisation of C^2KA.

lemma *inf-K-S-next-behaviour*: $(a \leq_{\mathcal{K}} b \wedge s \leq_{\mathcal{S}} t) \Longrightarrow (s \circ a \leq_{\mathcal{K}} t \circ b)$
  unfolding *Stimuli.leq-def CKA.leq-def*
proof –
  assume *hyp*: $a + b = b \wedge s \oplus t = t$
  hence $s \circ a + t \circ b = s \circ a + (s \oplus t) \circ b$ **by** *simp*
  hence $s \circ a + t \circ b = s \circ a + s \circ b + t \circ b$ **by** (*simp add*: *algebra-simps*)
  moreover have $s \circ (a + b) = s \circ a + s \circ b$ **by** *simp*
  ultimately have $s \circ a + t \circ b = s \circ (a + b) + t \circ b$ **by** *simp*
  hence $s \circ a + t \circ b = s \circ b + t \circ b$ **by** (*simp add*: *hyp*)
  hence $s \circ a + t \circ b = (s \oplus t) \circ b$ **by** *simp*
  thus $s \circ a + t \circ b = t \circ b$ **by** (*simp add*: *hyp*)
qed

lemma *inf-K-S-next-stimulus*: $a \leq_{\mathcal{K}} b \wedge s \leq_{\mathcal{S}} t \Longrightarrow \lambda(s,a) \leq_{\mathcal{S}} \lambda(t,b)$
  unfolding *Stimuli.leq-def CKA.leq-def*
proof –
  assume *hyp*: $a + b = b \wedge s \oplus t = t$
  hence $\lambda(s,a) \oplus \lambda(t,b) = \lambda(s,a) \oplus \lambda(s \oplus t,b)$ **by** *simp*
  hence $\lambda(s,a) \oplus \lambda(t,b) = \lambda(s,a) \oplus \lambda(s,b) \oplus \lambda(t,b)$ **by** (*simp add*: *add-assoc*)
  moreover have $\lambda(s,a+b) = \lambda(s,a) \oplus \lambda(s,b)$ **by** *simp*
  ultimately have $\lambda(s,a) \oplus \lambda(t,b) = \lambda(s,a+b) \oplus \lambda(t,b)$ **by** *simp*
  hence $\lambda(s,a) \oplus \lambda(t,b) = \lambda(s,b) \oplus \lambda(t,b)$ **by** (*simp add*: *hyp*)
  hence $\lambda(s,a) \oplus \lambda(t,b) = \lambda(s \oplus t,b)$ **by** *simp*
  thus $\lambda(s,a) \oplus \lambda(t,b) = \lambda(t,b)$ **by** (*simp add*: *hyp*)
qed

The following lemmas show additional results from the axiomatisation of C^2KA which follow from lemmas *inf-K-S-next-behaviour* and *inf-K-S-next-stimulus*.

lemma *inf-K-next-behaviour*: $a \leq_{\mathcal{K}} b \Longrightarrow s \circ a \leq_{\mathcal{K}} s \circ b$
  by (*simp add*: *inf-K-S-next-behaviour*)

lemma *inf-S-next-behaviour*: $s \leq_{\mathcal{S}} t \Longrightarrow s \circ a \leq_{\mathcal{K}} t \circ a$
  by (*simp add*: *inf-K-S-next-behaviour*)

lemma *inf-add-seq-par-next-behaviour*: $s \circ (a;b + b;a) \leq_{\mathcal{K}} s \circ (a*b)$
  using *inf-K-next-behaviour add-seq-inf-par* **by** *blast*

lemma *inf-seqstar-parstar-next-behaviour*: $s \circ a^{;} \leq_{\mathcal{K}} s \circ a^{*}$
  by (*simp add*: *seqstar-inf-parstar inf-K-next-behaviour*)

lemma *inf-S-next-stimulus*: $s \leq_{\mathcal{S}} t \Longrightarrow \lambda(s,a) \leq_{\mathcal{S}} \lambda(t,a)$
  by (*simp add*: *inf-K-S-next-stimulus*)

lemma *inf-K-next-stimulus*: $a \leq_{\mathcal{K}} b \Longrightarrow \lambda(s,a) \leq_{\mathcal{S}} \lambda(s,b)$
  by (*simp add*: *inf-K-S-next-stimulus*)

lemma *inf-add-seq-par-next-stimulus*: $\lambda(s,\ a;b + b;a) \leq_{\mathcal{S}} \lambda(s,\ a*b)$
proof –

have $a;b \leq_{\mathcal{K}} a{*}b$ **by** (*rule seq-inf-par*)
moreover have $b;a \leq_{\mathcal{K}} b{*}a$ **by** (*rule seq-inf-par*)
ultimately have $a;b + b;a \leq_{\mathcal{K}} a{*}b + b{*}a$ **by** (*simp add*: *add-mono*)
hence $a;b + b;a \leq_{\mathcal{K}} a{*}b$ **by** (*simp add*: *par-comm*)
thus $\lambda(s,\ a;b + b;a) \leq_{\mathcal{S}} \lambda(s,\ a{*}b)$ **by** (*rule inf-K-next-stimulus*)
qed

lemma *inf-seqstar-parstar-next-stimulus*: $\lambda(s,\ a^{;}) \leq_{\mathcal{S}} \lambda(s,\ a^{*})$
by (*simp add*: *seqstar-inf-parstar inf-K-next-stimulus*)

end

end

5 Notions of Topology for C^{2}KA

Orbits, stabilisers, and fixed points are notions that allow us to perceive a kind of topology of a system with respect to the stimulus-response relationships among system agents. In this context, the term "topology" is used to capture the relationships (influence) and connectedness via stimuli of the agents in a distributed system. It intends to capture a kind of reachability in terms of the possible behaviours for a given agent.

A C^{2}KA consists of two semimodules $\left({}_{\mathcal{S}}K, +\right)$ and $\left(S_{\mathcal{K}}, \oplus\right)$ for which we have a left $\mathcal{S}$-act ${}_{\mathcal{S}}K$ and a right $\mathcal{K}$-act $S_{\mathcal{K}}$. Therefore, there are two complementary notions of orbits, stabilisers, and fixed points within the context of agent behaviours and stimuli, respectively. In this way, one can use these notions to think about distributed systems from two different perspectives, namely the behavioural perspective provided by the action of stimuli on agent behaviours described by $\left({}_{\mathcal{S}}K, +\right)$ and the external event (stimulus) perspective provided by the action of agent behaviours on stimuli described by $\left(S_{\mathcal{K}}, \oplus\right)$. In this section, only the treatment of these notions with respect to the left $\mathcal{S}$-semimodule $\left({}_{\mathcal{S}}K, +\right)$ and agent behaviours is provided. The same notions for the right $\mathcal{K}$-semimodule $\left(S_{\mathcal{K}}, \oplus\right)$ and stimuli can be provided in a very similar way.

When discussing the interplay between C^{2}KA and the notions of orbits, stabilisers, and fixed points, the partial order of sub-behaviours $\leq_{\mathcal{K}}$ is extended to sets in order to express sets of agent behaviours encompassing one another. For two subsets of agent behaviours $A, B \subseteq K$, we say that A *is encompassed by* B (or B *encompasses* A), written $A \lessdot_{\mathcal{K}} B$, if and only if $\forall\left(a \mid a \in A : \exists(b \mid b \in B : a \leq_{\mathcal{K}} b)\right)$. In essence, $A \lessdot_{\mathcal{K}} B$ indicates that every behaviour contained within the set A is a sub-behaviour of at least one behaviour in the set B. The encompassing relation $\lessdot_{\mathcal{S}}$ for stimuli can be defined similarly.

Throughout this section, let $\left({}_{\mathcal{S}}K, +\right)$ be the unitary and zero-preserving left $\mathcal{S}$-semimodule of a C^2KA and let $a \in K$.

theory *Topology-C2KA*
imports *C2KA*
begin

no-notation
comp (**infixl** ∘ *55*)
and *rtrancl* ((-*) [*1000*] *999*)

The locale *topology-c2ka* extends the axiomatisation of *c2ka* to support the notions of topology.

locale *topology-c2ka* = *c2ka* +
fixes *orbit* :: *'a::cka* ⇒ *'a::cka set* (*Orb*)
and *strong-orbit* :: *'a::cka* ⇒ *'a::cka set* ($Orb_{\mathcal{S}}$)
and *stabiliser* :: *'a::cka* ⇒ *'b::stimuli set* (*Stab*)
and *fixed* :: *'a::cka* ⇒ *bool*
and *encompassing-relation-behaviours* :: *'a set* ⇒ *'a set* ⇒ *bool* (**infix** $\lessdot_{\mathcal{K}}$ *50*)
and *encompassing-relation-stimuli* :: *'b set* ⇒ *'b set* ⇒ *bool* (**infix** $\lessdot_{\mathcal{S}}$ *50*)
and *induced* :: *'a::cka* ⇒ *'a::cka* ⇒ *bool* (**infix** ◁ *50*)
and *orbit-equivalent* :: *'a::cka* ⇒ *'a::cka* ⇒ *bool* (**infix** $\sim_{\mathcal{K}}$ *50*)
assumes *orb-def*: $x \in Orb(a) \longleftrightarrow (\exists\, s.\ (s \circ a = x))$
and *orbs-def*: $b \in Orb_{\mathcal{S}}(a) \longleftrightarrow Orb(b) = Orb(a)$
and *stab-def*: $s \in Stab(a) \longleftrightarrow s \circ a = a$
and *fixed-def*: $fixed(a) \longleftrightarrow (\forall\, s::'b.\ s \neq \mathfrak{d} \longrightarrow s \circ a = a)$
and *erb-def*: $A \lessdot_{\mathcal{K}} B \longleftrightarrow (\forall\, a::'a.\ a \in A \longrightarrow (\exists\, b.\ b \in B \wedge a \leq_{\mathcal{K}} b))$
and *ers-def*: $E \lessdot_{\mathcal{S}} F \longleftrightarrow (\forall\, a::'b.\ a \in E \longrightarrow (\exists\, b.\ b \in F \wedge a \leq_{\mathcal{S}} b))$
and *induced-def*: $a \lhd b \longleftrightarrow b \in Orb(a)$
and *orbit-equivalent-def*: $a \sim_{\mathcal{K}} b \longleftrightarrow Orb(a) = Orb(b)$
begin

5.1 Orbits

The *orbit* of a in $\mathcal{S}$ is the set given by $\mathrm{Orb}(a) = \{s \circ a \mid s \in S\}$. The orbit of an agent $a \in K$ represents the set of all possible behavioural responses from an agent behaving as a to any stimulus from $\mathcal{S}$. In this way, the orbit of a given agent can be perceived as the set of all possible future behaviours for that agent.

Lemma *inf-K-enc-orb* provides an isotonicity law with respect to the orbits and the encompassing relation for agent behaviours.

lemma *inf-K-enc-orb*: $a \leq_{\mathcal{K}} b \Longrightarrow Orb(a) \lessdot_{\mathcal{K}} Orb(b)$
unfolding *erb-def orb-def*
using *inf-K-next-behaviour* **by** *blast*

The following lemmas provide a selection of properties regarding orbits and the encompassing relation for agent behaviours.

lemma *enc-orb-add*: $Orb(a) \lessdot_{\mathcal{K}} Orb(a + b)$
using *inf-K-enc-orb inf-add-K-right* **by** *auto*

lemma *enc-orb-exchange*: $Orb((a{*}b)\ ;\ (c{*}d)) \lessdot_{\mathcal{K}} Orb((a;c) * (b;d))$
using *inf-K-enc-orb exchange-2* **by** *blast*

lemma *enc-orb-seq-par*: $Orb(a;b) \lessdot_{\mathcal{K}} Orb(a{*}b)$
using *inf-K-enc-orb seq-inf-par* **by** *auto*

lemma *enc-orb-add-seq-par*: $Orb(a;b + b;a) \lessdot_{\mathcal{K}} Orb(a{*}b)$
using *inf-K-enc-orb add-seq-inf-par* **by** *auto*

lemma *enc-orb-parseq*: $Orb((a{*}b);c) \lessdot_{\mathcal{K}} Orb(a{*}(b;c))$
using *inf-K-enc-orb exchange-3* **by** *blast*

lemma *enc-orb-seqpar*: $Orb(a;(b{*}c)) \lessdot_{\mathcal{K}} Orb((a;b){*}c)$
using *inf-K-enc-orb exchange-4* **by** *blast*

lemma *enc-orb-seqstar-parstar*: $Orb(a^{;}) \lessdot_{\mathcal{K}} Orb(a^{*})$
using *inf-K-enc-orb seqstar-inf-parstar* **by** *auto*

lemma *enc-orb-union*: $Orb(a) \lessdot_{\mathcal{K}} Orb(c) \wedge Orb(b) \lessdot_{\mathcal{K}} Orb(c)$
$\longleftrightarrow Orb(a) \cup Orb(b) \lessdot_{\mathcal{K}} Orb(c)$
unfolding *erb-def*
by *auto*

5.2 Stabilisers

The *stabiliser* of a in $\mathcal{S}$ is the set given by $\text{Stab}(a) = \{s \in S \mid s \circ a = a\}$. The stabiliser of an agent $a \in K$ represents the set of stimuli which have no observable influence (or act as neutral stimuli) on an agent behaving as a.

Lemma *enc-stab-inter-add* provides a property regarding stabilisers and the encompassing relation for stimuli.

lemma *enc-stab-inter-add*: $Stab(a) \cap Stab(b) \lessdot_{\mathcal{S}} Stab(a + b)$
unfolding *ers-def*
by (*auto simp add*: *stab-def*, *rename-tac s*, *rule-tac x=s* **in** *exI*, *simp*)

5.3 Fixed Points

An element $a \in K$ is called a *fixed point* if $\forall(s \mid s \in S\backslash\{\mathfrak{d}\} : s \circ a = a)$. When an agent behaviour is a fixed point, it is not influenced by any stimulus other than the deactivation stimulus $\mathfrak{d}$. It is important to note that since $({}_{\mathcal{S}}K, +)$ is zero-preserving, every agent behaviour becomes inactive when subjected to the deactivation stimulus $\mathfrak{d}$. Because of this, we exclude this special case when discussing fixed point agent behaviours.

lemma *zerofix* [*simp*]: $s \circ 0 = 0$

proof −
  have $0 = \mathfrak{d} \circ a$ **by** *simp*
  hence $s \circ 0 = s \circ (\mathfrak{d} \circ a)$ **by** *simp*
  hence $s \circ 0 = (s \odot \mathfrak{d}) \circ a$ **by** (*simp only*: *lsemimodule3* [*symmetric*])
  thus $s \circ 0 = 0$ **by** *simp*
qed

The following lemmas provide a selection of properties regarding fixed agent behaviours.

lemma *fixed-zero*: $\mathit{fixed}(0)$
  unfolding *fixed-def*
  by *simp*

lemma *fixed-a-b-add*: $\mathit{fixed}(a) \wedge \mathit{fixed}(b) \longrightarrow \mathit{fixed}(a + b)$
  unfolding *fixed-def*
  by *simp*

lemma *fix-not-deactivation*: $s \circ a = a \wedge \lambda(s,a) = \mathfrak{d} \Longrightarrow a = 0$
proof −
  assume E: $s \circ a = a \wedge \lambda(s,a) = \mathfrak{d}$
  hence $s \circ (a;1) = a$ **by** *simp*
  hence $(s \circ a)\ ;\ (\lambda(s,a) \circ 1) = a$ **by** (*simp only*: *cascadingaxiom*)
  hence $0 = a$ **by** (*simp add*: E)
  thus *?thesis* **by** *auto*
qed

lemma *fixed-a-b-seq*: $\mathit{fixed}(a) \wedge \mathit{fixed}(b) \longrightarrow \mathit{fixed}(a\ ;\ b)$
  unfolding *fixed-def*
proof (*rule impI*)
  assume *hyp*: $(\forall s.\ s \neq \mathfrak{d} \longrightarrow s \circ a = a) \wedge (\forall s.\ s \neq \mathfrak{d} \longrightarrow s \circ b = b)$
  have *C1*: $(\forall s.\ \lambda(s,a) = \mathfrak{d} \longrightarrow s \neq \mathfrak{d} \longrightarrow s \circ (a\ ;\ b) = a\ ;\ b)$
  proof −
    have E: $(\forall s.\ s \neq \mathfrak{d} \wedge \lambda(s,a) = \mathfrak{d} \longrightarrow s \circ (a\ ;\ b) = 0)$ **by** *simp*
    hence $(\forall s.\ s \neq \mathfrak{d} \wedge \lambda(s,a) = \mathfrak{d} \longrightarrow s \circ a = a \wedge \lambda(s,a) = \mathfrak{d})$
      by (*simp add*: *hyp*)
    moreover have $(\forall s.\ s \circ a = a \wedge \lambda(s,a) = \mathfrak{d} \longrightarrow a = 0)$
      by (*simp add*: *fix-not-deactivation*)
    ultimately have $(\forall s.\ s \neq \mathfrak{d} \wedge \lambda(s,a) = \mathfrak{d} \longrightarrow a = 0)$ **by** *auto*
    thus *?thesis* **by** (*auto simp add*: E)
  qed
  moreover have *C2*: $(\forall s.\ \lambda(s,a) \neq \mathfrak{d} \longrightarrow s \neq \mathfrak{d} \longrightarrow s \circ (a\ ;\ b) = a\ ;\ b)$
    by (*simp add*: *hyp*)
  ultimately show $(\forall s.\ s \neq \mathfrak{d} \longrightarrow s \circ (a\ ;\ b) = a\ ;\ b)$ **by** *blast*
qed

5.4 Strong Orbits and Induced Behaviours

The *strong orbit* of a in $\mathcal{S}$ is the set given by $\mathrm{Orb}_{\mathbb{S}}(a) = \{b \in K \mid \mathrm{Orb}(b) = \mathrm{Orb}(a)\}$. Two agents are in the same strong orbit, denoted $a \sim_{\mathcal{K}} b$ for $a, b \in K$, if and only if their orbits are identical. This is to say when $a \sim_{\mathcal{K}} b$, if an agent behaving as a is influenced by a stimulus to behave as b, then there exists a stimulus which influences the agent, now behaving as b, to revert back to its original behaviour a.

The influence of stimuli on agent behaviours is called the *induced behaviours* via stimuli. Let $a, b \in K$ be agent behaviours with $a \neq b$. We say that b is *induced by a via stimuli* (denoted by $a \lhd b$) if and only if $\exists (s \mid s \in S : s \circ a = b)$. The notion of induced behaviours allows us to make some predictions about the evolution of agent behaviours in a given system by providing some insight into how different agents can respond to any stimuli.

Lemma *fixed-not-induce* states that if an agent has a fixed point behaviour, then it does not induce any agent behaviours via stimuli besides the inactive behaviour 0.

lemma *fixed-not-induce*: $\mathit{fixed}(a) \longrightarrow (\forall\, b.\ b \neq \mathit{0} \wedge b \neq a \longrightarrow \neg(a \lhd b))$
proof –
  have $\bigwedge s.\ s = \mathfrak{d} \vee s \neq \mathfrak{d} \Longrightarrow (\forall\, t.\ t \neq \mathfrak{d} \longrightarrow t \circ a = a) \Longrightarrow s \circ a \neq \mathit{0}$
$\Longrightarrow s \circ a \neq a \Longrightarrow \mathit{False}$
    by (*erule disjE*, *simp-all*)
  hence $\bigwedge s.\ (\forall\, t.\ t \neq \mathfrak{d} \longrightarrow t \circ a = a) \Longrightarrow s \circ a \neq \mathit{0} \Longrightarrow s \circ a \neq a \Longrightarrow \mathit{False}$
    by *simp*
  thus *?thesis*
    unfolding *fixed-def induced-def orb-def*
    by *auto*
qed

Lemma *strong-orbit-both-induced* states that all agent behaviours which belong to the same strong orbit are mutually induced via some (possibly different) stimuli. This is to say that if two agent behaviours are in the same strong orbit, then there exists inverse stimuli for each agent behaviour in a strong orbit allowing an agent to revert back to its previous behaviour.

lemma *in-own-orbit*: $a \in \mathit{Orb}(a)$
  unfolding *orb-def*
  by (*rule-tac* x=$\mathfrak{n}$ **in** *exI*, *simp*)

lemma *strong-orbit-both-induced*: $a \sim_{\mathcal{K}} b \longrightarrow a \lhd b \wedge b \lhd a$
  unfolding *orbit-equivalent-def induced-def*
  by (*blast intro*: *in-own-orbit*)

Lemma *strong-orbit-induce-same* states that if two agent behaviours are in the same strong orbit, then a third behaviour can be induced via stimuli by either of the behaviours within the strong orbit. This is to say that each behaviour in a strong orbit can induce the same set of behaviours (perhaps via different stimuli).

lemma *strong-orbit-induce-same*: $a \sim_{\mathcal{K}} b \longrightarrow (a \lhd c \longleftrightarrow b \lhd c)$
unfolding *induced-def orbit-equivalent-def*
by *simp*

end

end

6 Notions of Communication for C^{2}KA

Distributed systems contain a significant number of interactions among their constituent agents. Any interaction, direct or indirect, of an agent with its neighbouring agents can be understood as a *communication* [6]. Therefore, any potential for communication between two system agents can be characterized by the existence of a communication path allowing for the transfer of data or control from one agent to another. Potential for communication allows system agents to have an *influence* over each other. The study of agent influence allows for the determination of the overall structure of the distributed system of which the agents comprise. A full treatment of the potential for communication within distributed systems specified using C^{2}KA has been given in [2] and [3] and is highlighted below.

Consider a distributed system with $\mathsf{A}, \mathsf{B} \in \mathcal{A}$ such that $\mathsf{A} \neq \mathsf{B}$. We write $\mathsf{A} \mapsto \langle a \rangle$ where A is the name given to the agent and $a \in K$ is the agent behaviour. For $\mathsf{A} \mapsto \langle a \rangle$ and $\mathsf{B} \mapsto \langle b \rangle$, we write $\mathsf{A} + \mathsf{B}$ to denote the agent $\langle a + b \rangle$. In a sense, we extend the operators on behaviours of K to their corresponding agents.

Communication via stimuli from agent A to agent B is said to have taken place only when a stimulus generated by A *influences* (i.e., causes an observable change in, directly or indirectly) the behaviour of B. Note that it is possible that more than one agent is influenced by the generation of the same stimulus by another agent in the system. Formally, we say that agent $\mathsf{A} \mapsto \langle a \rangle$ has the *potential for direct communication via stimuli* with agent $\mathsf{B} \mapsto \langle b \rangle$ (denoted by $\mathsf{A} \rightarrow_{\mathcal{S}} \mathsf{B}$) if and only if $\exists\big(s, t \mid s, t \in S_b \ \wedge\ t \leq_{\mathcal{S}} \lambda(s, a) \ :\ t \circ b \neq b\big)$ where S_b is the set of all basic stimuli. A stimulus is called *basic* if it is indivisible with regard to the sequential composition operator $\odot$ of a stimulus structure. Similarly, we say that agent A has the *potential for communication via stimuli with agent* B *using at most* n *basic stimuli* (denoted by $\mathsf{A} \rightarrow^{n}_{\mathcal{S}} \mathsf{B}$) if and only if $\exists\big(\mathsf{C} \mid \mathsf{C} \in \mathcal{A} \ \wedge\ \mathsf{C} \neq \mathsf{A} \ \wedge\ \mathsf{C} \neq \mathsf{B} \ :\ \mathsf{A} \rightarrow^{(n-1)}_{\mathcal{S}} \mathsf{C} \ \wedge\ \mathsf{C} \rightarrow_{\mathcal{S}} \mathsf{B}\big)$. More generally, we say that agent A has the *potential for communication via stimuli* with agent B (denoted by $\mathsf{A} \rightarrow^{+}_{\mathcal{S}} \mathsf{B}$) if and only if $\exists\big(n \mid n \geq 1 \ :\ \mathsf{A} \rightarrow^{n}_{\mathcal{S}} \mathsf{B}\big)$.

When $\mathsf{A} \to_{\mathcal{S}}^{+} \mathsf{B}$, there is a sequence of stimuli of arbitrary length which allows for the transfer of data or control from agent A to agent B in the system. To simplify the Isabelle theory, we do not implement the potential for communication using at most n basic stimuli. Instead, we give the definition of potential for direct communication via stimuli and the fact that $\mathsf{A} \to_{\mathcal{S}} \mathsf{B} \Longrightarrow \mathsf{A} \to_{\mathcal{S}}^{+} \mathsf{B}$ as axioms because these are the only properties that we use about potential for communication via stimuli.

Communication via shared environments from agent A to agent B (denoted by $\mathsf{A} \to_{\mathcal{E}}^{+} \mathsf{B}$) is said to have taken place only when A has the ability to alter an element of the environment that it shares with B such that B is able to observe the alteration that was made. Formally, we say that agent $\mathsf{A} \mapsto \langle a \rangle$ has the *potential for direct communication via shared environments* with agent $\mathsf{B} \mapsto \langle b \rangle$ (denoted by $\mathsf{A} \to_{\mathcal{E}} \mathsf{B}$) if and only if $a \,\mathrm{R}\, b$ where R is a given dependence relation. More generally, agent A has the *potential for communication via shared environments* with agent B (denoted by $\mathsf{A} \to_{\mathcal{E}}^{+} \mathsf{B}$) if and only if $a \,\mathrm{R}^{+}\, b$ where R^{+} is the transitive closure of the given dependence relation. This means that if two agents respect the given dependence relation, then there is a potential for communication via shared environments.

theory *Communication-C2KA*
imports *Topology-C2KA*
begin

The locale *communication-c2ka* extends *topology-c2ka* to include aspects of potential for communication among distributed system agents.

locale *communication-c2ka* = *topology-c2ka* +
fixes *dcs* :: $'a::cka \Rightarrow 'a::cka \Rightarrow bool$ (**infix** $\to_{\mathcal{S}}$ *50*)
and *pcs* :: $'a::cka \Rightarrow 'a::cka \Rightarrow bool$ (**infix** $\to_{\mathcal{S}}{}^{+}$ *50*)
and *dce* :: $'a::cka \Rightarrow 'a::cka \Rightarrow bool$ (**infix** $\to_{\mathcal{E}}$ *50*)
and *pce* :: $'a::cka \Rightarrow 'a::cka \Rightarrow bool$ (**infix** $\to_{\mathcal{E}}{}^{+}$ *50*)
and *pdc* :: $'a::cka \Rightarrow 'a::cka \Rightarrow bool$ (**infix** $\rightsquigarrow$ *50*)
and *pfc* :: $'a::cka \Rightarrow 'a::cka \Rightarrow bool$ (**infix** $\rightsquigarrow^{+}$ *50*)
and *stimuli-connected* :: $'a\ set \Rightarrow bool$
and *universally-influential* :: $'a::cka \times 'a\ set \Rightarrow bool$
assumes *dcs-def*: $a \to_{\mathcal{S}} b \longleftrightarrow$
$(\exists\ s\ t.\ s \in \mathcal{S}_a \wedge t \in \mathcal{S}_a \wedge t \leq_{\mathcal{S}} \lambda(s,a) \wedge t \circ b \neq b)$
and *pdc-def*: $a \rightsquigarrow b \longleftrightarrow (a \to_{\mathcal{S}} b \vee a \to_{\mathcal{E}} b)$
and *zero-dce*: $\neg(0 \to_{\mathcal{E}} a)$
and *one-dce*: $\neg(1 \to_{\mathcal{E}} a)$
and *dce-zero*: $\neg(a \to_{\mathcal{E}} 0)$
and *dce-one*: $\neg(a \to_{\mathcal{E}} 1)$
and *sum-dce*: $(A + B \to_{\mathcal{E}} C) \longleftrightarrow (A \to_{\mathcal{E}} C \vee B \to_{\mathcal{E}} C)$
and *dce-sum*: $(A \to_{\mathcal{E}} B + C) \longleftrightarrow (A \to_{\mathcal{E}} B \vee A \to_{\mathcal{E}} C)$
and *dcs-pcs*: $A \to_{\mathcal{S}} B \Longrightarrow A \to_{\mathcal{S}}{}^{+} B$
and *stimuli-connected-def*: *stimuli-connected*$(\mathcal{C}) \longleftrightarrow$
$(\forall\ X_1\ X_2.\ X_1 \cap X_2 = \{\} \wedge X_1 \cup X_2 = \mathcal{C} \wedge X_1 \neq \{\} \wedge X_2 \neq \{\} \longrightarrow$
$(\exists\ A\ B.\ A \in X_1 \wedge B \in X_2 \wedge (A \to_{\mathcal{S}}{}^{+} B \vee B \to_{\mathcal{S}}{}^{+} A)))$

and *universally-influential-def*: *universally-influential*$(A,\mathcal{C}) \longleftrightarrow$
$A \in \mathcal{C} \wedge (\forall\ B.\ B \in \mathcal{C} \wedge B \neq A \longrightarrow A \rightarrow_{\mathcal{S}}{}^{+} B)$
begin

6.1 Stimuli-Connected Systems & Universally Influential Agents

Two subsets X_1 and X_2 of $\mathcal{A}$ form a partition of $\mathcal{A}$ if and only if $X_1 \cap X_2 = \emptyset$ and $X_1 \cup X_2 = \mathcal{A}$. A distributed system of agents $\mathcal{A}$ is called *stimuli-connected* if and only if for every X_1 and X_2 nonempty that form a partition of $\mathcal{A}$, we have $\exists(\mathsf{A},\mathsf{B} \mid \mathsf{A} \in X_1 \wedge \mathsf{B} \in X_2 : \mathsf{A} \rightarrow_{\mathcal{S}}^{+} \mathsf{B} \vee \mathsf{B} \rightarrow_{\mathcal{S}}^{+} \mathsf{A})$. Otherwise, $\mathcal{A}$ is called *stimuli-disconnected.* In a stimuli-connected system, every agent is a participant, either as the source or sink, of at least one direct communication via stimuli.

An agent $\mathsf{A} \in \mathcal{A}$ is called *universally influential* if and only if $\forall\big(\mathsf{B} \mid \mathsf{B} \in \mathcal{A}\backslash\{\mathsf{A}\} : \mathsf{A} \rightarrow_{\mathcal{S}}^{+} \mathsf{B}\big)$. A universally influential agent is able to generate some stimuli that influences the behaviour, either directly or indirectly, of each other agent in the system.

Lemma *universally-influential-stimuli-connected* shows that the existence of a universally influential agent yields a stimuli-connected system.

lemma *universally-influential-stimuli-connected*:
$(\exists\ A.\ universally\text{-}influential(A,\mathcal{C})) \longrightarrow stimuli\text{-}connected(\mathcal{C})$
  unfolding *universally-influential-def stimuli-connected-def*
proof (*intro allI impI*)
  fix $X_1\ X_2$
  show $(\exists A.\ A \in \mathcal{C} \wedge (\forall B.\ B \in \mathcal{C} \wedge B \neq A \longrightarrow A \rightarrow_{\mathcal{S}}{}^{+} B)) \Longrightarrow$
$X_1 \cap X_2 = \{\} \wedge X_1 \cup X_2 = \mathcal{C} \wedge X_1 \neq \{\} \wedge X_2 \neq \{\} \Longrightarrow$
$(\exists\ A\ B.\ A \in X_1 \wedge B \in X_2 \wedge (A \rightarrow_{\mathcal{S}}{}^{+} B \vee B \rightarrow_{\mathcal{S}}{}^{+} A))$
  proof −
    assume $(\exists A.\ A \in \mathcal{C} \wedge (\forall B.\ B \in \mathcal{C} \wedge B \neq A \longrightarrow A \rightarrow_{\mathcal{S}}{}^{+} B))$
    from *this* **obtain** A **where** *Aui*: $A \in \mathcal{C} \wedge (\forall B.\ B \in \mathcal{C} \wedge B \neq A \longrightarrow$
$A \rightarrow_{\mathcal{S}}{}^{+} B)$ **by** *auto*
    show $X_1 \cap X_2 = \{\} \wedge X_1 \cup X_2 = \mathcal{C} \wedge X_1 \neq \{\} \wedge X_2 \neq \{\} \Longrightarrow$
$(\exists A\ B.\ A \in X_1 \wedge B \in X_2 \wedge (A \rightarrow_{\mathcal{S}}{}^{+} B \vee B \rightarrow_{\mathcal{S}}{}^{+} A))$
    proof −
      assume *partition*:$X_1 \cap X_2 = \{\} \wedge X_1 \cup X_2 = \mathcal{C} \wedge X_1 \neq \{\} \wedge X_2 \neq \{\}$
      show $(\exists A\ B.\ A \in X_1 \wedge B \in X_2 \wedge (A \rightarrow_{\mathcal{S}}{}^{+} B \vee B \rightarrow_{\mathcal{S}}{}^{+} A))$
      proof *cases*
        assume *in1*: $A \in X_1$
        from *partition* **obtain** B **where** *in2*: $B \in X_2$ **by** *auto*
        have $A = B \Longrightarrow False$
        proof −
          assume $A = B$
          hence $A \in X_2$ **by** (*simp add*: *in2*)
          moreover have $A \in X_1$ **by** (*rule in1*)

ultimately have $A \in X_1 \cap X_2$ **by** *simp*
hence $A \in \{\}$ **by** (*simp add*: *partition*)
thus *False* **by** *simp*
qed
hence $A \neq B$ **by** *auto*
moreover have $B \in \mathcal{C}$
proof –
from *partition* **have** $\mathcal{C} = X_1 \cup X_2$ **by** *auto*
hence $X_2 \subseteq \mathcal{C}$ **by** *simp*
thus *?thesis* **by** (*auto simp add*: *in2*)
qed
ultimately have $A \rightarrow_{\mathcal{S}}{}^{+} B$ **by** (*auto simp add*: *Aui in2*)
thus *?thesis*
by (*rule-tac x=A* **in** *exI*, *rule-tac x=B* **in** *exI*, *simp add*: *in1 in2*)
next
assume *notin1*: $A \notin X_1$
moreover have $A \in \mathcal{C}$ **by** (*simp add*: *Aui*)
moreover have $X_1 \cup X_2 = \mathcal{C}$ **by** (*simp add*: *partition*)
ultimately have *in2*: $A \in X_2$ **by** *auto*
from *partition* **obtain** B **where** *in1*: $B \in X_1$ **by** *auto*
have $B = A \Longrightarrow$ *False*
proof –
assume $B = A$
hence $B \in X_2$ **by** (*simp add*: *in2*)
moreover have $B \in X_1$ **by** (*rule in1*)
ultimately have $B \in X_1 \cap X_2$ **by** *simp*
hence $B \in \{\}$ **by** (*simp add*: *partition*)
thus *False* **by** *simp*
qed
hence $B \neq A$ **by** *auto*
moreover have $B \in \mathcal{C}$
proof –
from *partition* **have** $\mathcal{C} = X_1 \cup X_2$ **by** *auto*
hence $X_1 \subseteq \mathcal{C}$ **by** *simp*
thus *?thesis* **by** (*auto simp add*: *in1*)
qed
ultimately have $A \rightarrow_{\mathcal{S}}{}^{+} B$ **by** (*auto simp add*: *Aui in2*)
thus *?thesis*
by (*rule-tac x=B* **in** *exI*, *rule-tac x=A* **in** *exI*, *simp add*: *in1 in2*)
qed
qed
qed
qed

Lemma *fixed-no-stimcomm* shows that no agent has the potential for communication via stimuli with an agent that has a fixed point behaviour.

lemma *fixed-no-stimcomm*: $fixed(A) \longrightarrow (\forall\ B.\ \neg(B \rightarrow_{\mathcal{S}} A))$
unfolding *fixed-def*
proof (*rule impI*)

assume *hyp*: $\forall s.\ s \neq \mathfrak{0} \longrightarrow s \circ A = A$
have $\exists\ B.\ B \rightarrow_{\mathcal{S}} A \Longrightarrow$ *False*
proof −
assume $\exists\ B.\ B \rightarrow_{\mathcal{S}} A$
then obtain B **where** $B \rightarrow_{\mathcal{S}} A$ **by** *auto*
hence $\exists\ s\ t.\ s \in \mathcal{S}_a \wedge t \in \mathcal{S}_a \wedge t \leq_{\mathcal{S}} \lambda(s,B) \wedge t \circ A \neq A$
by (*simp only*: *dcs-def*)
then obtain $s\ t$ **where** *st*: $s \in \mathcal{S}_a \wedge t \in \mathcal{S}_a \wedge t \leq_{\mathcal{S}} \lambda(s,B) \wedge t \circ A \neq A$
by *auto*
hence $t \neq \mathfrak{0}$ **by** (*auto simp only*: *zero-not-basic*)
hence $t \circ A = A$ **by** (*simp add*: *hyp*)
thus *False* **by** (*auto simp add*: *st*)
qed
thus $(\forall\ B.\ \neg(B \rightarrow_{\mathcal{S}} A))$ **by** *auto*
qed

6.2 Preserving the Potential for Communication under Non-Determinism

6.2.1 Potential for Communication via Stimuli

The following results show how the potential for communication via stimuli can be preserved when non-determinism is introduced among agents. Specifically, Lemma *source-nondet-stimcomm* states that when non-determinism is added at the source of a potential communication path via stimuli, the potential for communication via stimuli is always preserved. On the other hand, Lemma *sink-nondet-stimcomm* states that when non-determinism is added at the sink of a potential communication path via stimuli, the potential for communication is preserved only if there does not exist any basic stimulus that is generated by the source that influences agent B and agent C to behave as a sub-behaviour of agent $\mathsf{B} + \mathsf{C}$. This condition ensures that agent $\mathsf{B} + \mathsf{C}$ cannot have a fixed point behaviour.

lemma *source-nondet-stimcomm*: $(B \rightarrow_{\mathcal{S}} C) \Longrightarrow ((A + B) \rightarrow_{\mathcal{S}} C)$
proof −
assume $B \rightarrow_{\mathcal{S}} C$
then obtain $s\ t$ **where** *st*: $s \in \mathcal{S}_a \wedge t \in \mathcal{S}_a \wedge t \leq_{\mathcal{S}} \lambda(s,B) \wedge t \circ C \neq C$
by (*auto simp only*: *dcs-def*)
show $(A + B) \rightarrow_{\mathcal{S}} C$
unfolding *dcs-def*
by (*rule-tac x=s* **in** *exI*, *rule-tac x=t* **in** *exI*, *auto simp add*: *st inf-add-S-left*)
qed

lemma *comm-source-nondet-stimcomm*: $(B \rightarrow_{\mathcal{S}} C) \Longrightarrow ((B + A) \rightarrow_{\mathcal{S}} C)$
by (*simp add*: *source-nondet-stimcomm algebra-simps*)

lemma *sink-sum-stimcomm*: $(\exists\ s\ t.\ s \in \mathcal{S}_a \wedge t \in \mathcal{S}_a \wedge t \leq_{\mathcal{S}} \lambda(s,A) \wedge$
$\neg(t \circ B \leq_{\mathcal{K}} B + C \wedge t \circ C \leq_{\mathcal{K}} B + C)) \Longrightarrow (A \rightarrow_{\mathcal{S}} B + C)$

proof –
assume $\exists\ s\ t.\ s \in \mathcal{S}_a \wedge t \in \mathcal{S}_a \wedge t \leq_{\mathcal{S}} \lambda(s,A) \wedge$
$\neg(t \circ B \leq_{\mathcal{K}} B + C \wedge t \circ C \leq_{\mathcal{K}} B + C)$
then obtain $s\ t$ **where** *st*: $s \in \mathcal{S}_a \wedge t \in \mathcal{S}_a \wedge t \leq_{\mathcal{S}} \lambda(s,A) \wedge$
$\neg(t \circ B \leq_{\mathcal{K}} B + C \wedge t \circ C \leq_{\mathcal{K}} B + C)$ **by** *auto*
have $t \circ (B + C) = B + C \Longrightarrow False$
proof –
assume *fixbc*: $t \circ (B + C) = B + C$
have $t \circ B\ \leq_{\mathcal{K}} t \circ (B + C)$
by (*simp*, *rule inf-add-K-right*)
moreover have $t \circ C\ \leq_{\mathcal{K}} t \circ (B + C)$
by (*simp*, *rule inf-add-K-left*)
ultimately have $t \circ B \leq_{\mathcal{K}} B + C \wedge t \circ C \leq_{\mathcal{K}} B + C$
by (*simp only*: *fixbc*)
thus *False* **by** (*simp only*: *st*)
qed
thus $A \rightarrow_{\mathcal{S}} B + C$
unfolding *dcs-def*
by (*rule-tac x=s* **in** *exI*, *rule-tac x=t* **in** *exI*, *auto simp only*: *st*)
qed

lemma *sink-nondet-stimcomm*: $A \rightarrow_{\mathcal{S}} B \Longrightarrow (\forall\ s\ t.\ s \in \mathcal{S}_a \wedge t \in \mathcal{S}_a \wedge t \leq_{\mathcal{S}}$
$\lambda(s,A)$
$\longrightarrow \neg(t \circ B \leq_{\mathcal{K}} B + C \wedge t \circ C \leq_{\mathcal{K}} B + C)) \Longrightarrow (A \rightarrow_{\mathcal{S}} B + C)$
proof (*rule sink-sum-stimcomm*)
assume *h1*: $A \rightarrow_{\mathcal{S}} B$
assume *h2*: $(\forall\ s\ t.\ s \in \mathcal{S}_a \wedge t \in \mathcal{S}_a \wedge t \leq_{\mathcal{S}} \lambda(s,A) \longrightarrow$
$\neg(t \circ B \leq_{\mathcal{K}} B + C \wedge t \circ C \leq_{\mathcal{K}} B + C))$
show $\exists\ s\ t.\ s \in \mathcal{S}_a \wedge t \in \mathcal{S}_a \wedge t \leq_{\mathcal{S}} \lambda(s,A) \wedge$
$\neg(t \circ B \leq_{\mathcal{K}} B + C \wedge t \circ C \leq_{\mathcal{K}} B + C)$
proof –
from *h1* **obtain** $s\ t$ **where** $s \in \mathcal{S}_a \wedge t \in \mathcal{S}_a \wedge t \leq_{\mathcal{S}} \lambda(s,A) \wedge t \circ B \neq B$
by (*auto simp only*: *dcs-def*)
from *this h2* **have** $s \in \mathcal{S}_a \wedge t \in \mathcal{S}_a \wedge t \leq_{\mathcal{S}} \lambda(s,A) \wedge$
$\neg(t \circ B \leq_{\mathcal{K}} B + C \wedge t \circ C \leq_{\mathcal{K}} B + C)$ **by** *auto*
thus *?thesis*
by (*rule-tac x=s* **in** *exI*, *rule-tac x=t* **in** *exI*, *auto*)
qed
qed

6.2.2 Potential for Communication via Shared Environments

Lemmas *source-nondet-envcomm* and *sink-nondet-envcomm* show how the potential for communication via shared environments is preserved when non-determinism is introduced at the source or the sink of a potential communication path via shared environments.

lemma *source-nondet-envcomm*: $B \rightarrow_{\mathcal{E}} C \Longrightarrow (A + B) \rightarrow_{\mathcal{E}} C$
by (*simp add*: *sum-dce*)

lemma *sink-nondet-envcomm*: $A \rightarrow_{\mathcal{E}} B \Longrightarrow A \rightarrow_{\mathcal{E}} (B + C)$
  by (*simp add*: *dce-sum*)

6.3 Preserving the Potential for Communication with Agent Behaviour Modifications

The following results identify the conditions constraining the modifications that can be made to the source or sink agent involved in a direct potential for communication to preserve the communication in a distributed system. In this way, it demonstrates the conditions under which a modification to an agent behaviour can be made while maintaining the communicating behaviour of the agents in the system.

Specifically, Lemma *sink-seq-stimcomm* shows how the sequential composition of an additional behaviour on the left of a sink agent will not affect the potential for communication provided that every stimulus that is generated by the source agent either does not fix the behaviour of the first component of the sequential composition, or causes the first component of the sequential composition to generate a stimulus that does not fix the behaviour of the second component of the sequential composition. Alternatively, Lemma *nondet-right-source-communication* shows how non-determinism added on the right of a source agent will not affect the potential for communication provided that the non-deterministic behaviours can be influenced by the source agent to stop being a sub-behaviour of the non-deterministic behaviour.

lemma *sink-seq-stimcomm*: $A \rightarrow_{\mathcal{S}} B$
$\Longrightarrow \forall\ s\ t.\ s \in \mathcal{S}_a \wedge t \in \mathcal{S}_a \wedge t \leq_{\mathcal{S}} \lambda(s,A) \longrightarrow \lambda(t,C) = t \Longrightarrow A;C \rightarrow_{\mathcal{S}} B$
proof −
  assume $A \rightarrow_{\mathcal{S}} B$
  then obtain $s\ t$ **where** *st*: $s \in \mathcal{S}_a \wedge t \in \mathcal{S}_a \wedge t \leq_{\mathcal{S}} \lambda(s,A) \wedge t \circ B \neq B$
    unfolding *dcs-def* **by** *auto*
  assume $\forall\ s\ t.\ s \in \mathcal{S}_a \wedge t \in \mathcal{S}_a \wedge t \leq_{\mathcal{S}} \lambda(s,A) \longrightarrow \lambda(t,C) = t$
  from *this st* **have** *tfix*: $\lambda(t,C) = t$ **by** *auto*
  have $\lambda(t,C) \leq_{\mathcal{S}} \lambda(\lambda(s,A),C)$ **by** (*simp add*: *inf-S-next-stimulus st*)
  hence $t \leq_{\mathcal{S}} \lambda\ (\lambda\ (s,\ A),\ C)$ **by** (*simp add*: *tfix*)
  thus $A;C \rightarrow_{\mathcal{S}} B$
    unfolding *dcs-def*
    by (*rule-tac x=s* **in** *exI*, *rule-tac x=t* **in** *exI*, *simp add*: *st*)
qed

lemma *nondet-right-source-communication*: $A \rightsquigarrow C \wedge C \rightsquigarrow B \Longrightarrow (\forall\ s\ t.\ s \in \mathcal{S}_a$
$\wedge\ t \in \mathcal{S}_a \wedge t \leq_{\mathcal{S}} \lambda(s,A)$
$\longrightarrow \neg(t \circ C \leq_{\mathcal{K}} C + D \wedge t \circ D \leq_{\mathcal{K}} C + D)) \Longrightarrow A \rightsquigarrow C{+}D \wedge C{+}D \rightsquigarrow B$
proof −
  assume *h1*:$A \rightsquigarrow C \wedge C \rightsquigarrow B$
  assume *h2*:($\forall\ s\ t.\ s \in \mathcal{S}_a \wedge t \in \mathcal{S}_a \wedge t \leq_{\mathcal{S}} \lambda(s,A)$

$\longrightarrow \neg(t \circ C \leq_{\mathcal{K}} C + D \wedge t \circ D \leq_{\mathcal{K}} C + D))$
from *h2* **have** *hs*: $A \rightarrow_{\mathcal{S}} C \Longrightarrow A \rightarrow_{\mathcal{S}} C{+}D$
by (*auto simp add*: *sink-nondet-stimcomm*)
have $A \rightsquigarrow C \Longrightarrow A \rightsquigarrow C{+}D$
unfolding *pdc-def*
using *dce-sum hs* **by** *blast*
moreover have $C \rightsquigarrow B \Longrightarrow C + D \rightsquigarrow B$
unfolding *pdc-def*
using *comm-source-nondet-stimcomm sum-dce* **by** *blast*
ultimately show $A \rightsquigarrow C{+}D \wedge C{+}D \rightsquigarrow B$
by (*simp add*: *h1*)
qed

end

end

References

[1] C. Hoare, B. Möller, G. Struth, and I. Wehrman. Concurrent Kleene algebra and its foundations. *Journal of Logic and Algebraic Programming*, 80(6):266–296, 2011.

[2] J. Jaskolka. *On the Modelling, Analysis, and Mitigation of Distributed Covert Channels*. PhD thesis, McMaster University, Hamilton, ON, Canada, March 2015.

[3] J. Jaskolka and R. Khedri. A formulation of the potential for communication condition using C^2KA. In A. Peron and C. Piazza, editors, *Proceedings of the 5th International Symposium on Games, Automata, Logics and Formal Verification*, volume 161 of *Electronic Proceedings in Theoretical Computer Science*, pages 161–174. Open Publishing Association, Verona, Italy, September 2014.

[4] J. Jaskolka, R. Khedri, and Q. Zhang. Foundations of communicating concurrent Kleene algebra. Technical Report CAS-13-07-RK, McMaster University, Hamilton, ON, Canada, November 2013.

[5] J. Jaskolka, R. Khedri, and Q. Zhang. Endowing concurrent Kleene algebra with communication actions. In P. Höfner, P. Jipsen, W. Kahl, and M. Müller, editors, *Proceedings of the 14th International Conference on Relational and Algebraic Methods in Computer Science*, volume 8428 of *Lecture Notes in Computer Science*, pages 19–36. Springer International Publishing Switzerland, 2014.

[6] R. Milner. *Communication and Concurrency*. Prentice-Hall International Series in Computer Science. Prentice Hall, 1989.