

# Concurrent Kleene Algebra: Free Model and Completeness

*Tobias Kappé*   Paul Brunet   Alexandra Silva   Fabio Zanasi

University College London

Leiden, 28 May 2018

Let's write a program that outputs  $n > 0$  space-separated 😊's.

# Introduction

Let's write a program that outputs  $n > 0$  space-separated ☺'s.

```
i := 1
while i < n do
  | print ☺
  | print ␣
  | i := i + 1
end
print ☺
```

# Introduction

Let's write a program that outputs  $n > 0$  space-separated ☺'s.

```
i := 1
while i < n do
  | print ☺
  | print ⊣
  | i := i + 1
end
print ☺
```

```
i := 1
print ☺
while i < n do
  | print ⊣
  | print ☺
  | i := i + 1
end
```

# Introduction

Let's write a program that outputs  $n > 0$  space-separated ☺'s.

```
i := 1
while i < n do
  | print ☺
  | print ⊥
  | i := i + 1
end
print ☺
```

```
i := 1
print ☺
while i < n do
  | print ⊥
  | print ☺
  | i := i + 1
end
```

Are these programs equivalent?

Programs are *expressions*

Programs are *expressions*, thus we should be able to reason *equationally*.

Programs are *expressions*, thus we should be able to reason *equationally*.

*Kleene Algebra* (KA) provides an *algebraic framework* to do this.

| program       | expression               |
|---------------|--------------------------|
| atomic action | $a, b, \dots \in \Sigma$ |

| program         | expression               |
|-----------------|--------------------------|
| atomic action   | $a, b, \dots \in \Sigma$ |
| abort execution | 0                        |

| program         | expression               |
|-----------------|--------------------------|
| atomic action   | $a, b, \dots \in \Sigma$ |
| abort execution | 0                        |
| no-operation    | 1                        |

| program                 | expression               |
|-------------------------|--------------------------|
| atomic action           | $a, b, \dots \in \Sigma$ |
| abort execution         | 0                        |
| no-operation            | 1                        |
| nondeterministic choice | $e + f$                  |

| program                 | expression               |
|-------------------------|--------------------------|
| atomic action           | $a, b, \dots \in \Sigma$ |
| abort execution         | 0                        |
| no-operation            | 1                        |
| nondeterministic choice | $e + f$                  |
| sequential composition  | $e \cdot f$              |

| program                 | expression               |
|-------------------------|--------------------------|
| atomic action           | $a, b, \dots \in \Sigma$ |
| abort execution         | $0$                      |
| no-operation            | $1$                      |
| nondeterministic choice | $e + f$                  |
| sequential composition  | $e \cdot f$              |
| repetition              | $e^*$                    |

# Introduction

```
i := 1
while i < n do
  | print ☺
  | print ⊥
  | i := i + 1
end
print ☺
```

```
i := 1
print ☺
while i < n do
  | print ⊥
  | print ☺
  | i := i + 1
end
```

# Introduction

```
i := 1  
while i < n do  
  | print ☺  
  | print ⊥  
  | i := i + 1  
end  
print ☺
```

}  $(\text{☺} \cdot \perp)^* \cdot \text{☺}$

```
i := 1  
print ☺  
while i < n do  
  | print ⊥  
  | print ☺  
  | i := i + 1  
end
```

# Introduction

```
i := 1  
while i < n do  
  | print ☺  
  | print ⊥  
  | i := i + 1  
end  
print ☺
```

$$\left. \vphantom{\begin{array}{l} i := 1 \\ \text{while } i < n \text{ do} \\ \quad | \text{ print } \odot \\ \quad | \text{ print } \perp \\ \quad | i := i + 1 \\ \text{end} \\ \text{print } \odot \end{array}} \right\} (\odot \cdot \perp)^* \cdot \odot$$

```
i := 1  
print ☺  
while i < n do  
  | print ⊥  
  | print ☺  
  | i := i + 1  
end
```

$$\left. \vphantom{\begin{array}{l} i := 1 \\ \text{print } \odot \\ \text{while } i < n \text{ do} \\ \quad | \text{ print } \perp \\ \quad | \text{ print } \odot \\ \quad | i := i + 1 \\ \text{end} \end{array}} \right\} \odot \cdot (\perp \cdot \odot)^*$$

## Axioms of KA:

$$e + 0 \equiv e \qquad e + e \equiv e \qquad e + f \equiv f + e \qquad e + (f + g) \equiv (e + f) + g$$

$$e \cdot 0 \equiv 0 \equiv 0 \cdot e \qquad e \cdot 1 \equiv e \equiv 1 \cdot e \qquad e \cdot (f \cdot g) \equiv (e \cdot f) \cdot g$$

$$e \cdot (f + g) \equiv e \cdot f + e \cdot g \qquad (e + f) \cdot g \equiv e \cdot g + f \cdot g$$

$$1 + e \cdot e^* \equiv e^* \qquad e \cdot f + g \leq f \implies e^* \cdot g \leq f$$

$$1 + e^* \cdot e \equiv e^* \qquad e \cdot f + g \leq e \implies g \cdot f^* \leq e$$

## Axioms of KA:

$$e + 0 \equiv e \qquad e + e \equiv e \qquad e + f \equiv f + e \qquad e + (f + g) \equiv (e + f) + g$$

$$e \cdot 0 \equiv 0 \equiv 0 \cdot e \qquad e \cdot 1 \equiv e \equiv 1 \cdot e \qquad e \cdot (f \cdot g) \equiv (e \cdot f) \cdot g$$

$$e \cdot (f + g) \equiv e \cdot f + e \cdot g \qquad (e + f) \cdot g \equiv e \cdot g + f \cdot g$$

$$1 + e \cdot e^* \equiv e^* \qquad e \cdot f + g \leq f \implies e^* \cdot g \leq f$$

$$1 + e^* \cdot e \equiv e^* \qquad e \cdot f + g \leq e \implies g \cdot f^* \leq e$$

$$\odot \cdot (\sqcup \cdot \odot)^* \equiv (\odot \cdot \sqcup)^* \cdot \odot$$

## Theorem (Kozen 1990)

*The axioms for KA are sound & **complete** for equivalence:*

$$e \equiv f \iff \mathcal{L}(e) = \mathcal{L}(f)$$

*$\mathcal{L}(e)$  is the regular language interpretation of  $e$ .*

## Theorem (Kozen 1990)

*The axioms for KA are sound & **complete** for equivalence:*

$$e \equiv f \iff \mathcal{L}(e) = \mathcal{L}(f)$$

*$\mathcal{L}(e)$  is the regular language interpretation of  $e$ .*

Upshot:

- to check KA equivalence is to check regular language equivalence
- through Kleene's theorem, this means checking DFA equivalence
- sophisticated (near-linear) algorithms exist to do this

# Adding concurrency

Which *new* axioms do we need for parallel composition?

# Adding concurrency

Which *new* axioms do we need for parallel composition?

$$e \parallel f \equiv f \parallel e$$

# Adding concurrency

Which *new* axioms do we need for parallel composition?

$$e \parallel f \equiv f \parallel e$$

$$e \parallel (f \parallel g) \equiv (e \parallel f) \parallel g$$

# Adding concurrency

Which *new* axioms do we need for parallel composition?

$$e \parallel f \equiv f \parallel e$$

$$e \parallel (f \parallel g) \equiv (e \parallel f) \parallel g$$

$$e \parallel 1 \equiv e$$

# Adding concurrency

Which *new* axioms do we need for parallel composition?

$$e \parallel f \equiv f \parallel e$$

$$e \parallel (f \parallel g) \equiv (e \parallel f) \parallel g$$

$$e \parallel 1 \equiv e$$

$$e \parallel 0 \equiv 0$$

# Adding concurrency

Which *new* axioms do we need for parallel composition?

$$e \parallel f \equiv f \parallel e$$

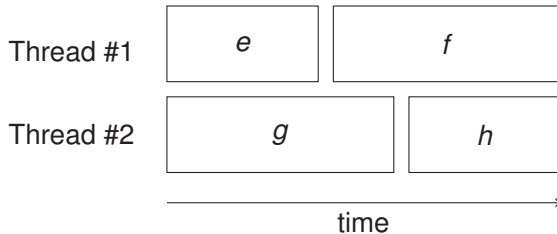
$$e \parallel (f \parallel g) \equiv (e \parallel f) \parallel g$$

$$e \parallel 1 \equiv e$$

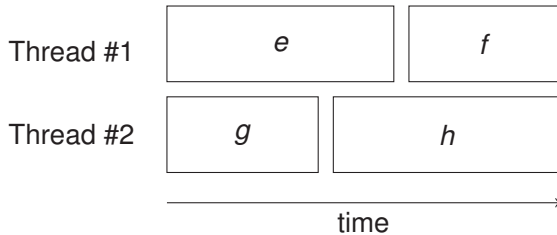
$$e \parallel 0 \equiv 0$$

$$e \parallel (f + g) \equiv e \parallel f + e \parallel g$$

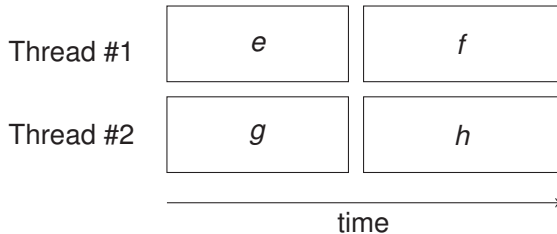
# Adding concurrency



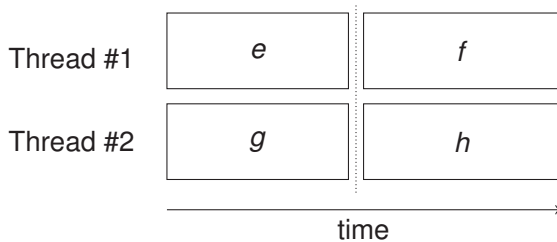
# Adding concurrency



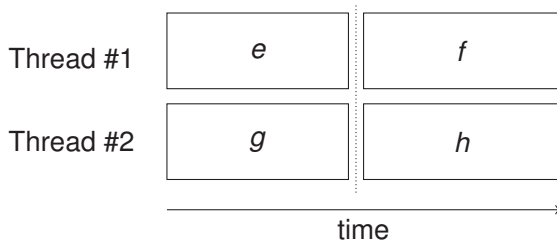
# Adding concurrency



# Adding concurrency

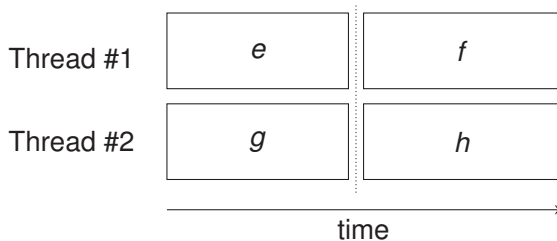


# Adding concurrency



Equationally:  $(e \parallel g) \cdot (f \parallel h) \leq (e \cdot f) \parallel (g \cdot h)$ .

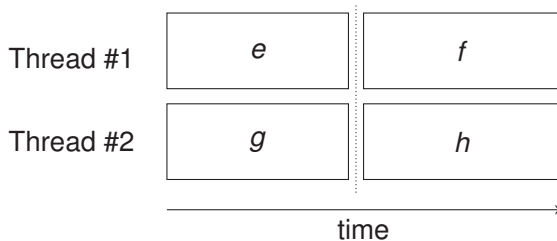
# Adding concurrency



Equationally:  $(e \parallel g) \cdot (f \parallel h) \leq (e \cdot f) \parallel (g \cdot h)$ .

$$p \leq q \iff p + q \equiv q$$

# Adding concurrency



Equationally:  $(e \parallel g) \cdot (f \parallel h) \leq (e \cdot f) \parallel (g \cdot h)$ .

Nondeterministic interleaving as special case:  $e \cdot f + f \cdot e \leq e \parallel f$ .

# Adding concurrency

## Question

*Can we have a regular interpretation  $\llbracket - \rrbracket$  such that  $e \equiv f \iff \llbracket e \rrbracket = \llbracket f \rrbracket$ ?*

# Adding concurrency

## Question

*Can we have a regular interpretation  $\llbracket - \rrbracket$  such that  $e \equiv f \iff \llbracket e \rrbracket = \llbracket f \rrbracket$ ?*

NB:  $\llbracket - \rrbracket$  should generalize  $\mathcal{L}(-)$ : for  $\parallel$ -less terms,  $\mathcal{L}(e)$  should resemble  $\llbracket e \rrbracket$ .

# Regular interpretation: first attempt

*Partially ordered multiset* (pomset):

$$a \cdot b \cong a \longrightarrow b$$

# Regular interpretation: first attempt

*Partially ordered multiset* (pomset):

$$a \cdot b \cong a \longrightarrow b$$

$$a \parallel b \cong$$

$b$   
 $a$

# Regular interpretation: first attempt

*Partially ordered multiset* (pomset):

$$a \cdot b \cong a \longrightarrow b$$

$$c \cdot (a \parallel b) \cong c \begin{array}{l} \nearrow b \\ \searrow a \end{array}$$

# Regular interpretation: first attempt

*Partially ordered multiset (pomset):*

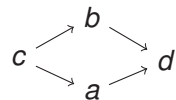
$$a \cdot b \cong a \longrightarrow b$$

$$c \cdot (a \parallel b) \cdot d \cong \begin{array}{ccccc} & & b & & \\ & \nearrow & & \searrow & \\ c & & & & d \\ & \searrow & & \nearrow & \\ & & a & & \end{array}$$

# Regular interpretation: first attempt

*Partially ordered multiset* (pomset):

$$a \cdot b \cong a \longrightarrow b$$

$$c \cdot (a \parallel b) \cdot d \cong$$


Composition lifts to *sets of pomsets* in the obvious way.

# Regular interpretation: first attempt

Straightforward semantics:  $\llbracket - \rrbracket : \mathcal{T} \rightarrow 2^{\text{Pomsets}}$  given by

$$\llbracket 0 \rrbracket = \emptyset$$

$$\llbracket 1 \rrbracket = \{1\}$$

$$\llbracket a \rrbracket = \{a\}$$

$$\llbracket e + f \rrbracket = \llbracket e \rrbracket \cup \llbracket f \rrbracket$$

$$\llbracket e \cdot f \rrbracket = \llbracket e \rrbracket \cdot \llbracket f \rrbracket$$

$$\llbracket e \parallel f \rrbracket = \llbracket e \rrbracket \parallel \llbracket f \rrbracket$$

$$\llbracket e^* \rrbracket = \llbracket e \rrbracket^*$$

# Regular interpretation: first attempt

Straightforward semantics:  $\llbracket - \rrbracket : \mathcal{T} \rightarrow 2^{\text{Pomsets}}$  given by

$$\llbracket 0 \rrbracket = \emptyset$$

$$\llbracket 1 \rrbracket = \{1\}$$

$$\llbracket a \rrbracket = \{a\}$$

$$\llbracket e + f \rrbracket = \llbracket e \rrbracket \cup \llbracket f \rrbracket$$

$$\llbracket e \cdot f \rrbracket = \llbracket e \rrbracket \cdot \llbracket f \rrbracket$$

$$\llbracket e \parallel f \rrbracket = \llbracket e \rrbracket \parallel \llbracket f \rrbracket$$

$$\llbracket e^* \rrbracket = \llbracket e \rrbracket^*$$

Problem:  $\llbracket - \rrbracket$  is not sound for the exchange law.

# Regular interpretation: first attempt

Straightforward semantics:  $\llbracket - \rrbracket : \mathcal{T} \rightarrow 2^{\text{Pomsets}}$  given by

$$\llbracket 0 \rrbracket = \emptyset$$

$$\llbracket 1 \rrbracket = \{1\}$$

$$\llbracket a \rrbracket = \{a\}$$

$$\llbracket e + f \rrbracket = \llbracket e \rrbracket \cup \llbracket f \rrbracket$$

$$\llbracket e \cdot f \rrbracket = \llbracket e \rrbracket \cdot \llbracket f \rrbracket$$

$$\llbracket e \parallel f \rrbracket = \llbracket e \rrbracket \parallel \llbracket f \rrbracket$$

$$\llbracket e^* \rrbracket = \llbracket e \rrbracket^*$$

Problem:  $\llbracket - \rrbracket$  is not sound for the exchange law.

For instance:  $a \cdot b \leq a \parallel b$  should imply that  $\llbracket a \cdot b \rrbracket \subseteq \llbracket a \parallel b \rrbracket$ , but

$$\llbracket a \cdot b \rrbracket = \left\{ a \longrightarrow b \right\}$$

$$\llbracket a \parallel b \rrbracket = \left\{ a \quad b \right\}$$

# Regular interpretation: first attempt

Axioms to build  $\approx$  are axioms for  $\equiv$ , minus exchange law.

# Regular interpretation: first attempt

Axioms to build  $\approx$  are axioms for  $\equiv$ , minus exchange law.

Theorem (Laurence and Struth 2014)

*The axioms for  $\approx$  are sound & complete w.r.t.  $\langle - \rangle$ :*

$$e \approx f \iff \langle e \rangle = \langle f \rangle$$

# Regular interpretation: second attempt

We define the *subsumption order*  $\sqsubseteq$  on pomsets.

Intuition:  $U \sqsubseteq V$  if

- i  $U$  and  $V$  have the same events, and
- ii  $U$  has all order in  $V$  (and possibly more)

# Regular interpretation: second attempt

We define the *subsumption order*  $\sqsubseteq$  on pomsets.

Intuition:  $U \sqsubseteq V$  if

- i  $U$  and  $V$  have the same events, and
- ii  $U$  has all order in  $V$  (and possibly more)

For example:

$$a \longrightarrow b \sqsubseteq a \quad b$$

# Regular interpretation: second attempt

We define the *subsumption order*  $\sqsubseteq$  on pomsets.

Intuition:  $U \sqsubseteq V$  if

- i  $U$  and  $V$  have the same events, and
- ii  $U$  has all order in  $V$  (and possibly more)

For example:

$$\begin{array}{ccc} a \longrightarrow c & & a \longrightarrow c \\ & \nearrow \searrow & \\ b \longrightarrow d & \sqsubseteq & b \longrightarrow d \end{array}$$

## Regular interpretation: second attempt

“Fixed” semantics:  $\llbracket e \rrbracket = \langle e \rangle \downarrow$ .

downward closure w.r.t.  $\sqsubseteq$

## Regular interpretation: second attempt

“Fixed” semantics:  $\llbracket e \rrbracket = \langle e \rangle \downarrow$ .

Previous problem no longer occurs:

$$\llbracket a \cdot b \rrbracket = \left\{ a \rightarrow b \right\} \subseteq \left\{ a \rightarrow b, a \leftarrow b, a \quad b \right\} = \llbracket a \parallel b \rrbracket$$

## Regular interpretation: second attempt

“Fixed” semantics:  $\llbracket e \rrbracket = \langle e \rangle \downarrow$ .

Previous problem no longer occurs:

$$\llbracket a \cdot b \rrbracket = \{ a \rightarrow b \} \subseteq \{ a \rightarrow b, a \leftarrow b, a \quad b \} = \llbracket a \parallel b \rrbracket$$

Lemma (Hoare et al. 2009)

*The axioms for  $\equiv$  are sound w.r.t.  $\llbracket - \rrbracket$ , i.e.,  $e \equiv f$  implies  $\llbracket e \rrbracket = \llbracket f \rrbracket$ .*

# Closure

## Definition

Let  $e \in \mathcal{T}$ ; a *closure* of  $e$  is a term  $e\downarrow$  such that

$$1 \quad e\downarrow \equiv e$$

$$2 \quad \llbracket e \rrbracket = \langle e\downarrow \rangle$$

# Closure

## Definition

Let  $e \in \mathcal{T}$ ; a *closure* of  $e$  is a term  $e\downarrow$  such that

1  $e\downarrow \equiv e$

2  $\llbracket e \rrbracket = \langle e\downarrow \rangle$

## Lemma (Laurence and Struth 2017)

*If closures exist for all terms, then  $\equiv$  is complete w.r.t.  $\llbracket - \rrbracket$ , i.e.,  $\llbracket e \rrbracket = \llbracket f \rrbracket$  implies  $e \equiv f$ .*

# Closure

## Definition

Let  $e \in \mathcal{T}$ ; a *closure* of  $e$  is a term  $e\downarrow$  such that

$$1 \quad e\downarrow \equiv e$$

$$2 \quad \llbracket e \rrbracket = \langle e\downarrow \rangle$$

## Lemma (Laurence and Struth 2017)

*If closures exist for all terms, then  $\equiv$  is complete w.r.t.  $\llbracket - \rrbracket$ , i.e.,  $\llbracket e \rrbracket = \llbracket f \rrbracket$  implies  $e \equiv f$ .*

## Proof.

If  $\llbracket e \rrbracket = \llbracket f \rrbracket$ ,



# Closure

## Definition

Let  $e \in \mathcal{T}$ ; a *closure* of  $e$  is a term  $e\downarrow$  such that

$$1 \quad e\downarrow \equiv e$$

$$2 \quad \llbracket e \rrbracket = \langle e\downarrow \rangle$$

## Lemma (Laurence and Struth 2017)

*If closures exist for all terms, then  $\equiv$  is complete w.r.t.  $\llbracket - \rrbracket$ , i.e.,  $\llbracket e \rrbracket = \llbracket f \rrbracket$  implies  $e \equiv f$ .*

## Proof.

If  $\llbracket e \rrbracket = \llbracket f \rrbracket$ , then  $\langle e\downarrow \rangle = \langle f\downarrow \rangle$ ,



# Closure

## Definition

Let  $e \in \mathcal{T}$ ; a *closure* of  $e$  is a term  $e\downarrow$  such that

$$1 \quad e\downarrow \equiv e$$

$$2 \quad \llbracket e \rrbracket = \langle e\downarrow \rangle$$

## Lemma (Laurence and Struth 2017)

*If closures exist for all terms, then  $\equiv$  is complete w.r.t.  $\llbracket - \rrbracket$ , i.e.,  $\llbracket e \rrbracket = \llbracket f \rrbracket$  implies  $e \equiv f$ .*

## Proof.

If  $\llbracket e \rrbracket = \llbracket f \rrbracket$ , then  $\langle e\downarrow \rangle = \langle f\downarrow \rangle$ , thus  $e\downarrow \approx f\downarrow$ . □

# Closure

## Definition

Let  $e \in \mathcal{T}$ ; a *closure* of  $e$  is a term  $e\downarrow$  such that

$$1 \quad e\downarrow \equiv e$$

$$2 \quad \llbracket e \rrbracket = \langle e\downarrow \rangle$$

## Lemma (Laurence and Struth 2017)

*If closures exist for all terms, then  $\equiv$  is complete w.r.t.  $\llbracket - \rrbracket$ , i.e.,  $\llbracket e \rrbracket = \llbracket f \rrbracket$  implies  $e \equiv f$ .*

## Proof.

If  $\llbracket e \rrbracket = \llbracket f \rrbracket$ , then  $\langle e\downarrow \rangle = \langle f\downarrow \rangle$ , thus  $e\downarrow \approx f\downarrow$ . Therefore,  $e\downarrow \equiv f\downarrow$  . □

# Closure

## Definition

Let  $e \in \mathcal{T}$ ; a *closure* of  $e$  is a term  $e\downarrow$  such that

$$1 \quad e\downarrow \equiv e$$

$$2 \quad \llbracket e \rrbracket = \langle e\downarrow \rangle$$

## Lemma (Laurence and Struth 2017)

*If closures exist for all terms, then  $\equiv$  is complete w.r.t.  $\llbracket - \rrbracket$ , i.e.,  $\llbracket e \rrbracket = \llbracket f \rrbracket$  implies  $e \equiv f$ .*

## Proof.

If  $\llbracket e \rrbracket = \llbracket f \rrbracket$ , then  $\langle e\downarrow \rangle = \langle f\downarrow \rangle$ , thus  $e\downarrow \approx f\downarrow$ . Therefore,  $e \equiv e\downarrow \equiv f\downarrow \equiv f$ . □

# Main contribution

## Theorem

*If  $e \in \mathcal{T}$ , then we can **compute** a term  $e\downarrow$  that is a closure of  $e$ .*

# Main contribution

## Theorem

*If  $e \in \mathcal{T}$ , then we can **compute** a term  $e_{\downarrow}$  that is a closure of  $e$ .*

## Corollary

*The axioms for CKA are sound & complete w.r.t.  $\llbracket - \rrbracket$ :*

$$e \equiv f \iff \llbracket e \rrbracket = \llbracket f \rrbracket$$

# Main contribution

## Theorem

*If  $e \in \mathcal{T}$ , then we can **compute** a term  $e_{\downarrow}$  that is a closure of  $e$ .*

## Corollary

*The axioms for CKA are sound & complete w.r.t.  $\llbracket - \rrbracket$ :*

$$e \equiv f \iff \llbracket e \rrbracket = \llbracket f \rrbracket$$

The latter can be decided; c.f. [Brunet, Pous, and Struth 2017].

- Explore coalgebraic perspective:
  - Efficient equivalence checking through bisimulation?
  - Can completeness be shown coalgebraically?
- Add “parallel star” operator — closure method does not apply.
- Extend *Kleene Algebra with Tests* (KAT) to add concurrency.
- Extend extend NetKAT with concurrency.

Thank you for your attention



Implementation: <https://doi.org/10.5281/zenodo.926651>.

Extended paper: <https://arxiv.org/abs/1710.02787>.

## Bonus: computing the closure

So, how does one compute a closure?

## Bonus: computing the closure

So, how does one compute a closure?

### Lemma

*If  $e, f$  have closures  $e\downarrow$  and  $f\downarrow$  respectively, then*

- 1  $e\downarrow + f\downarrow$  is a closure of  $e + f$
- 2  $e\downarrow \cdot f\downarrow$  is a closure of  $e \cdot f$
- 3  $e\downarrow^*$  is a closure of  $e^*$

## Bonus: computing the closure

So, how does one compute a closure?

### Lemma

*If  $e, f$  have closures  $e\downarrow$  and  $f\downarrow$  respectively, then*

- 1  $e\downarrow + f\downarrow$  is a closure of  $e + f$
- 2  $e\downarrow \cdot f\downarrow$  is a closure of  $e \cdot f$
- 3  $e\downarrow^*$  is a closure of  $e^*$

One case remains: parallel composition.

## Bonus: computing the closure

Sketch: given  $e \parallel f$ , apply exchange law syntactically, “in the limit”.

## Bonus: computing the closure

Sketch: given  $e \parallel f$ , apply exchange law syntactically, “in the limit”.

For instance: if  $e = a \cdot b$  and  $f = c \cdot d$ :

$$\blacksquare (a \parallel c) \cdot (b \parallel d) \leq e \parallel f$$

$$(e = a \bullet b, f = c \bullet d)$$

## Bonus: computing the closure

Sketch: given  $e \parallel f$ , apply exchange law syntactically, “in the limit”.

For instance: if  $e = a \cdot b$  and  $f = c \cdot d$ :

$$\blacksquare (a \parallel c) \cdot (b \parallel d) \leq e \parallel f$$

$$(e = a \bullet b, f = c \bullet d)$$

$$\blacksquare (a \parallel 1) \cdot (b \parallel (c \cdot d)) \leq e \parallel f$$

$$(e = a \bullet b, f = 1 \bullet c \cdot d)$$

## Bonus: computing the closure

Sketch: given  $e \parallel f$ , apply exchange law syntactically, “in the limit”.

For instance: if  $e = a \cdot b$  and  $f = c \cdot d$ :

$$\blacksquare (a \parallel c) \cdot (b \parallel d) \leq e \parallel f$$

$$(e = a \bullet b, f = c \bullet d)$$

$$\blacksquare (a \parallel 1) \cdot (b \parallel (c \cdot d)) \leq e \parallel f$$

$$(e = a \bullet b, f = 1 \bullet c \cdot d)$$

$$\blacksquare (1 \parallel c) \cdot ((a \cdot b) \parallel d) \leq e \parallel f$$

$$(e = 1 \bullet a \cdot b, f = c \bullet d)$$

$$\blacksquare \dots$$

## Bonus: computing the closure

Sketch: given  $e \parallel f$ , apply exchange law syntactically, “in the limit”.

For instance: if  $e = a \cdot b$  and  $f = c \cdot d$ :

$$\blacksquare (a \parallel c) \cdot (b \parallel d) \leq e \parallel f$$

$$(e = a \bullet b, f = c \bullet d)$$

$$\blacksquare (a \parallel 1) \cdot (b \parallel (c \cdot d)) \leq e \parallel f$$

$$(e = a \bullet b, f = 1 \bullet c \cdot d)$$

$$\blacksquare (1 \parallel c) \cdot ((a \cdot b) \parallel d) \leq e \parallel f$$

$$(e = 1 \bullet a \cdot b, f = c \bullet d)$$

$$\blacksquare \dots$$

Goal: find enough of these terms to cover all pomsets in  $\llbracket e \parallel f \rrbracket$ .

## Bonus: computing the closure

Obstacles to overcome:

- How to split terms  $e$  and  $f$  into heads and tails?

## Bonus: computing the closure

Obstacles to overcome:

- How to split terms  $e$  and  $f$  into heads and tails?
- What to do about recursion? For instance,

$$(e \parallel f) \cdot (e^* \parallel f^*) \leq e^* \parallel f^*$$

# Bonus: computing the closure

Obstacles to overcome:

- How to split terms  $e$  and  $f$  into heads and tails?
- What to do about recursion? For instance,

👉 splicing relations

$$(e \parallel f) \cdot (e^* \parallel f^*) \leq e^* \parallel f^*$$

# Bonus: computing the closure

Obstacles to overcome:

- How to split terms  $e$  and  $f$  into heads and tails?
- What to do about recursion? For instance,

👉 splicing relations

👉 fixpoints of inequations

$$(e \parallel f) \cdot (e^* \parallel f^*) \leq e^* \parallel f^*$$

# Bonus: computing the closure

## Definition

Let  $e \in \mathcal{T}$ . We define  $\nabla_e \subseteq \mathcal{T} \times \mathcal{T}$  as the smallest relation such that

$$\begin{array}{c} \overline{1 \nabla_1 1} \qquad \overline{a \nabla_a 1} \qquad \overline{1 \nabla_a a} \qquad \overline{1 \nabla_{e^*} 1} \qquad \frac{\ell \nabla_e r}{\ell \nabla_{e+f} r} \\[10pt] \frac{\ell \nabla_e r}{\ell \nabla_{e \cdot f} r \cdot f} \quad \frac{\ell \nabla_e r}{\ell \nabla_{e \cdot f} r \cdot f} \quad \frac{\ell \nabla_f r}{e \cdot \ell \nabla_{e \cdot f} r} \quad \frac{\ell_0 \nabla_e r_0 \quad \ell_1 \nabla_f r_1}{\ell_0 \parallel \ell_1 \nabla_{e \parallel f} r_0 \parallel r_1} \quad \frac{\ell \nabla_e r}{e^* \cdot \ell \nabla_{e^*} r \cdot e^*} \end{array}$$

## Lemma

Let  $e \in \mathcal{T}$  and  $U \cdot V \in \llbracket e \rrbracket_{\text{WCKA}}$ ; there exist  $\ell \nabla_e r$  such that  $U \in \llbracket \ell \rrbracket$  and  $V \in \llbracket r \rrbracket$ .

## Bonus: computing the closure

Suppose that for all  $g, h \in \mathcal{T}$ , we have that  $X_{g \parallel h}$  is a closure of  $g \parallel h$ .

Then we find

$$e \parallel f + \sum_{\substack{\ell_e \parallel \ell_f \\ \nabla_e r_e \\ \nabla_f r_f}} (\ell_e \parallel \ell_f) \cdot (r_e \parallel r_f) \leq X_{e \parallel f}$$

## Bonus: computing the closure

Suppose that for all  $g, h \in \mathcal{T}$ , we have that  $X_{g \parallel h}$  is a closure of  $g \parallel h$ .

Then we find

$$e \parallel f + \sum_{\substack{\ell_e \nabla_e r_e \\ \ell_f \nabla_f r_f}} (\ell_e \parallel \ell_f) \cdot X_{r_e \parallel r_f} \leq X_{e \parallel f}$$

## Bonus: computing the closure

Suppose that for all  $g, h \in \mathcal{T}$ , we have that  $X_{g \parallel h}$  is a closure of  $g \parallel h$ .

Then we find

$$e \parallel f + \sum_{\substack{\ell_e \nabla_e r_e \\ \ell_f \nabla_f r_f}} (\ell_e \parallel \ell_f) \cdot X_{r_e \parallel r_f} \leq X_{e \parallel f}$$

For  $X_{r_e \parallel r_f}$ , we find another inequation, et cetera. . .

## Bonus: computing the closure

Suppose that for all  $g, h \in \mathcal{T}$ , we have that  $X_{g \parallel h}$  is a closure of  $g \parallel h$ .

Then we find

$$e \parallel f + \sum_{\substack{\ell_e \nabla_e r_e \\ \ell_f \nabla_f r_f}} (\ell_e \parallel \ell_f) \cdot X_{r_e \parallel r_f} \leq X_{e \parallel f}$$

For  $X_{r_e \parallel r_f}$ , we find another inequation, et cetera. . .

### Lemma

*Continuing this, we get a finite system of inequations  $\langle M, \vec{b} \rangle_{e \parallel f}$ .*

## Bonus: computing the closure

### Theorem

Let  $e \otimes f$  be the least solution to  $X_{e \parallel f}$  in  $\langle M, \vec{b} \rangle_{e \parallel f}$ . Then the following hold:

1  $e \otimes f \equiv e \parallel f$

2  $\langle e \otimes f \rangle = \llbracket e \parallel f \rrbracket$

In other words,  $e \otimes f$  is a closure of  $e \parallel f$ .