AVANCÉES RÉCENTES
SUR LES
ALGÈBRES DE KLEENE CONCURRENTES

SÉMINAIRE ACADIE - ENSEEIHT

Avril 2021

Paul Brunet
University College London

UCL

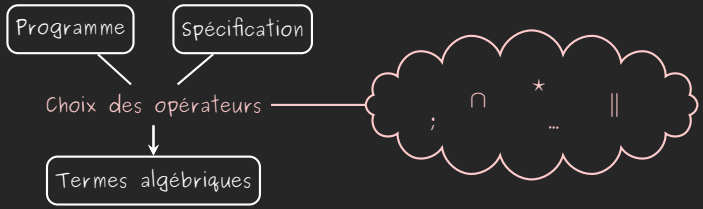
AVANCÉES RÉCENTES SUR CKA
Plan

Projet de recherche : Algèbres de programmes

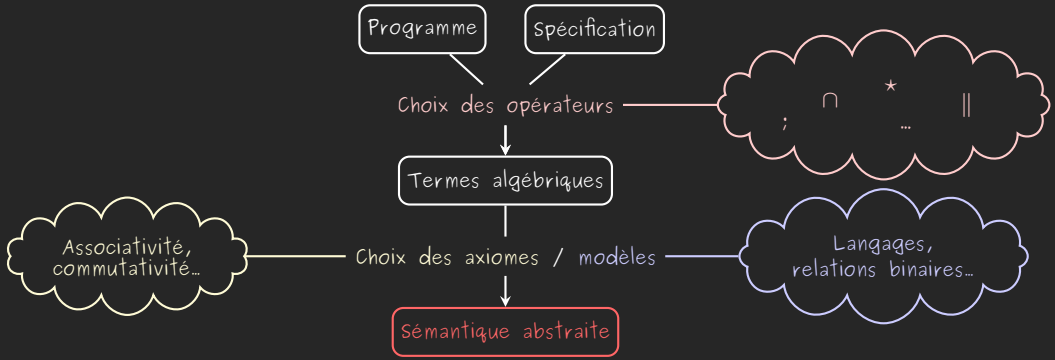
Programme

Spécification

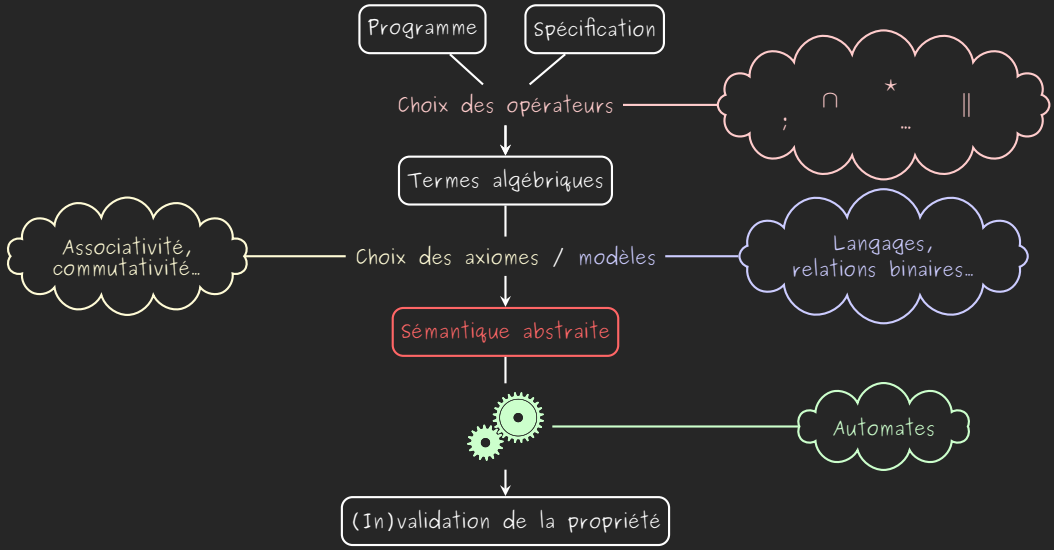
PROJET DE RECHERCHE : ALGÈBRES DE PROGRAMMES



Projet de recherche : Algèbres de programmes



Projet de recherche : Algèbres de programmes



ALGÈBRE DE KLEENE – LA THÉORIE DES LANGAGES RÉGULIERS

$$e, f \in \mathbf{E}_A^{ka} ::= 0 \mid 1 \mid a \in A \mid e \cdot f \mid e + f \mid e^*$$

Interprétation : langages réguliers

$$[\![\,]\!] : \mathbf{E}_A^{ka} \to \mathcal{P}(A^*)$$

exemple :

$$[\![a \cdot ((a + b) \cdot a)^*]\!] = \left\{ \begin{array}{l} \text{mots sur l'alphabet } \{a, b\} \text{ de longueur} \\ \text{impaire et tels qu'une lettre sur deux est} \\ \text{un } a \end{array} \right\}$$
$$= \{a, aaa, aba, aaaaa, abaaa, aaaba, ababa, \ldots\}$$

ALGÈBRE DE KLEENE – LA THÉORIE DES LANGAGES RÉGULIERS

Les axiomes de KA

$$e + e = e \qquad e + f = f + e \qquad e + (f + g) = (e + f) + g$$
$$e + 0 = e \qquad e \cdot 1 = e = 1 \cdot e \qquad e \cdot (f \cdot g) = (e \cdot f) \cdot g$$
$$e \cdot 0 = 0 = 0 \cdot e \qquad e \cdot (f + g) = e \cdot f + e \cdot g \qquad (e + f) \cdot g = e \cdot g + f \cdot g$$
$$e^* = 1 + e \cdot e^* \qquad e \cdot f \leq f \Rightarrow e^* \cdot f \leq f$$

Théorème

$$KA \vdash e = f \Leftrightarrow [\![e]\!] = [\![f]\!].$$

Kozen, "A completeness theorem for Kleene algebras and the algebra of regular events", LiCS '90

KAT – LA THÉORIE DES PROGRAMMES IMPÉRATIFS

Syntaxe

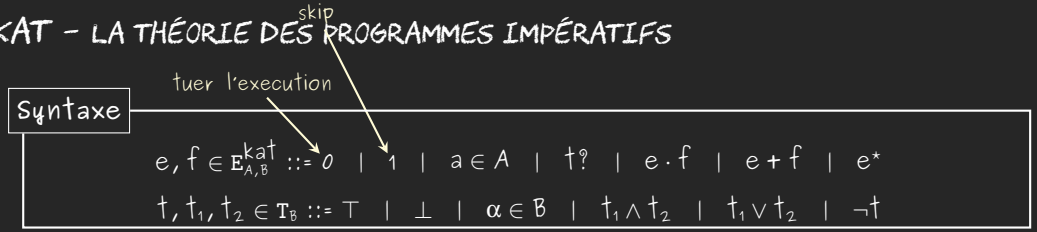
$$e, f \in \mathrm{E}^{\mathsf{kat}}_{A,B} ::= 0 \mid 1 \mid a \in A \mid t? \mid e \cdot f \mid e + f \mid e^*$$

$$t, t_1, t_2 \in \mathrm{T}_B ::= \top \mid \bot \mid \alpha \in B \mid t_1 \wedge t_2 \mid t_1 \vee t_2 \mid \neg t$$

Encode un langage `While` simple :

$$\texttt{if}\ b\ \texttt{then}\ p\ \texttt{else}\ q \mapsto b? \cdot p + (\neg b)? \cdot q \qquad \texttt{while}\ b\ \texttt{do}\ p \mapsto (b? \cdot p)^* \cdot (\neg b)?$$

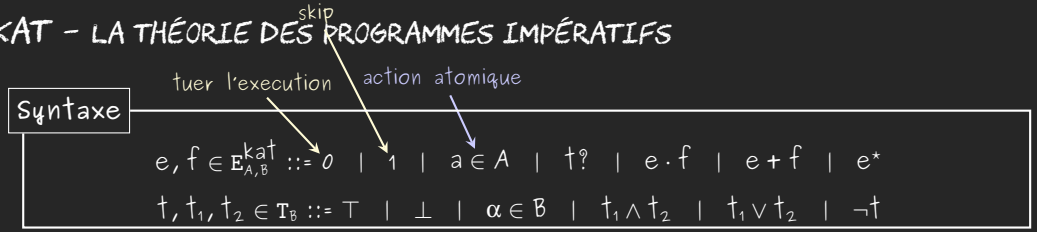
KAT - LA THÉORIE DES PROGRAMMES IMPÉRATIFS



Encode un langage `While` simple :

$\texttt{if}\ b\ \texttt{then}\ p\ \texttt{else}\ q \mapsto b? \cdot p + (\neg b)? \cdot q$ $\qquad$ $\texttt{while}\ b\ \texttt{do}\ p \mapsto (b? \cdot p)^* \cdot (\neg b)?$

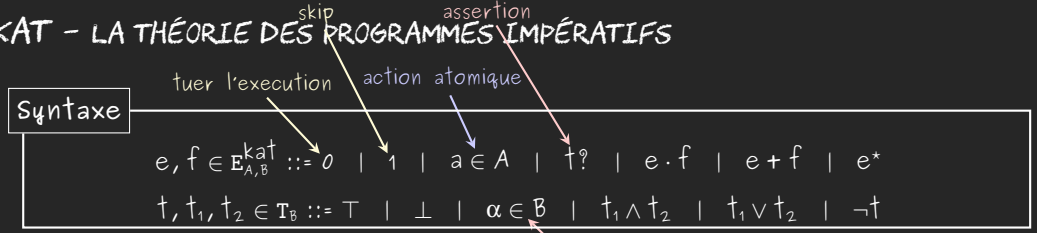
KAT - LA THÉORIE DES PROGRAMMES IMPÉRATIFS



Encode un langage `While` simple :

$\texttt{if}\ b\ \texttt{then}\ p\ \texttt{else}\ q \mapsto b? \cdot p + (\neg b)? \cdot q$ $\qquad$ $\texttt{while}\ b\ \texttt{do}\ p \mapsto (b? \cdot p)^* \cdot (\neg b)?$

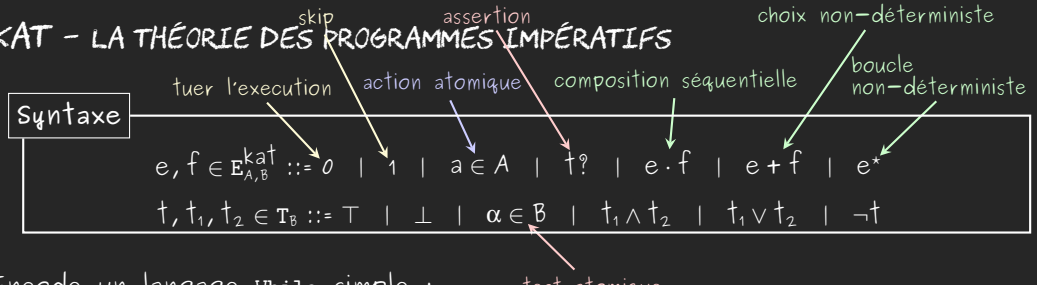
KAT - LA THÉORIE DES PROGRAMMES IMPÉRATIFS



Encode un langage `While` simple :

$\texttt{if}\ b\ \texttt{then}\ p\ \texttt{else}\ q \mapsto b? \cdot p + (\neg b)? \cdot q$ $\qquad$ $\texttt{while}\ b\ \texttt{do}\ p \mapsto (b? \cdot p)^* \cdot (\neg b)?$

KAT - LA THÉORIE DES PROGRAMMES IMPÉRATIFS



Encode un langage `While` simple :

$$\texttt{if } b \texttt{ then } p \texttt{ else } q \mapsto b? \cdot p + (\neg b)? \cdot q \qquad \texttt{while } b \texttt{ do } p \mapsto (b? \cdot p)^* \cdot (\neg b)?$$

KAT - LA THÉORIE DES PROGRAMMES IMPÉRATIFS

skip — assertion — choix non-déterministe — tuer l'execution — action atomique — composition séquentielle — boucle non-déterministe

Syntaxe

$$e, f \in \mathrm{E}^{\mathrm{kat}}_{A,B} ::= 0 \mid 1 \mid a \in A \mid t? \mid e \cdot f \mid e + f \mid e^*$$

$$t, t_1, t_2 \in \mathrm{T}_B ::= \top \mid \bot \mid \alpha \in B \mid t_1 \wedge t_2 \mid t_1 \vee t_2 \mid \neg t$$

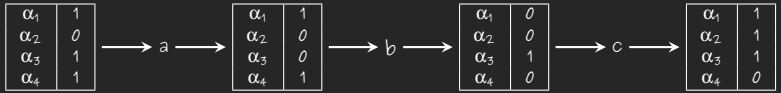
test atomique

Encode un langage `While` simple :

$$\texttt{if } b \texttt{ then } p \texttt{ else } q \mapsto b? \cdot p + (\neg b)? \cdot q \qquad \texttt{while } b \texttt{ do } p \mapsto (b? \cdot p)^* \cdot (\neg b)?$$

Interprétation : langages de chaînes gardées

chaînes guardées : séquences alternant états $\in 2^B$ & actions $\in A$.



KAT - LA THÉORIE DES PROGRAMMES IMPÉRATIFS

Les axiomes de KAT :

- ☞ Le axiomes de KA.
- ☞ Pour les tests, les axiomes des algèbres booléennes.
- ☞ Les axiomes "glue" suivants :

$$(t_1 \vee t_2)? = t_1? + t_2? \qquad (t_1 \wedge t_2)? = t_1? \cdot t_2? \qquad \top? = 1 \qquad \bot? = 0$$

Théorème

$$\mathrm{KAT} \vdash e = f \Leftrightarrow [\![e]\!] = [\![f]\!].$$

Kozen & Smith, "Kleene algebra with tests : Completeness and decidability", CSL '96

KAT – LA THÉORIE DES PROGRAMMES IMPÉRATIFS

Les axiomes de KAT :

- ☞ Le axiomes de KA.
- ☞ Pour les tests, les axiomes des algèbres booléennes.
- ☞ Les axiomes "glue" suivants :

$$(t_1 \vee t_2)? = t_1? + t_2? \qquad (t_1 \wedge t_2)? = t_1? \cdot t_2? \qquad \top? = 1 \qquad \bot? = 0$$

Théorème

$$\mathrm{KAT} \vdash e = f \Leftrightarrow [\![e]\!] = [\![f]\!].$$

Kozen & Smith, "Kleene algebra with tests : Completeness and decidability", CSL '96

Encode la logique de Hoare propositionnelle :

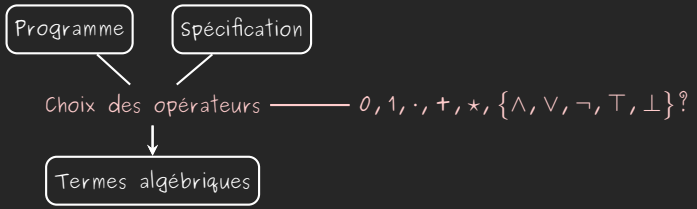
$$\begin{aligned} \{b\}\,p\,\{c\} &\Leftrightarrow b? \cdot p \leq p \cdot c? \\ &\Leftrightarrow b? \cdot p = b? \cdot p \cdot c? \\ &\Leftrightarrow b? \cdot p \cdot (\neg c)? = 0 \end{aligned}$$

KAT comme algèbre de programmes

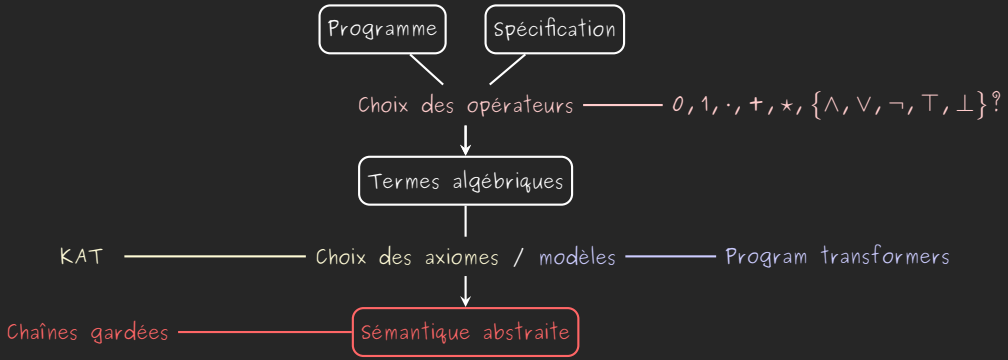
Programme

Spécification

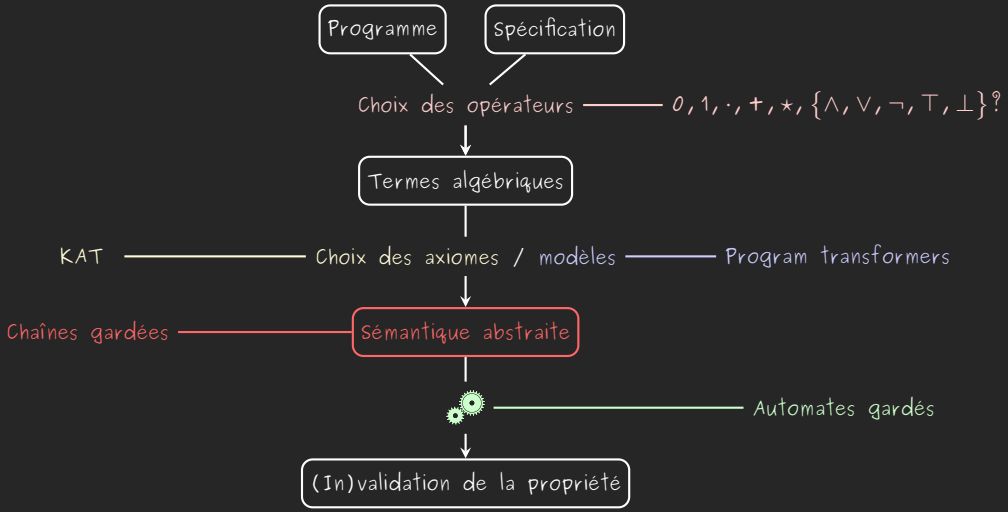
KAT comme algèbre de programmes



KAT comme algèbre de programmes

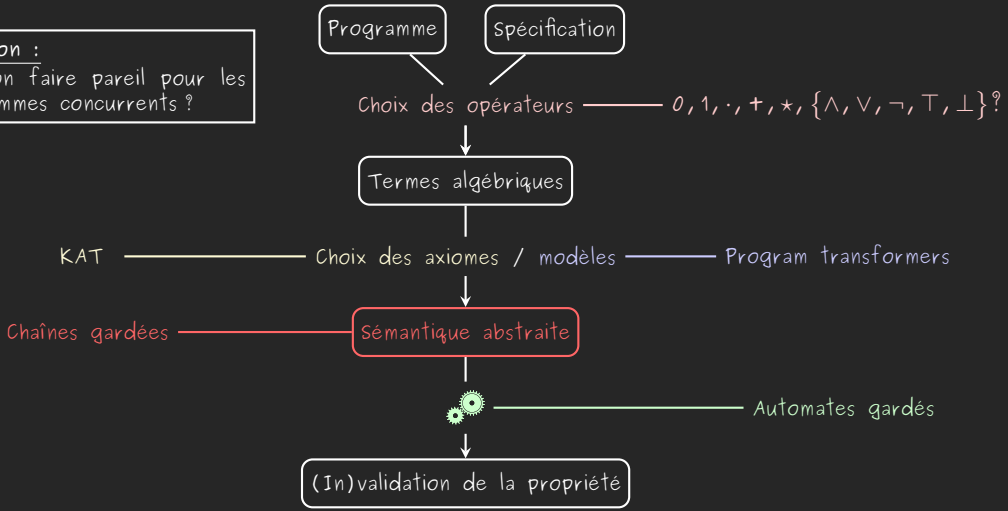


KAT comme algèbre de programmes



KAT comme algèbre de programmes

Question :
Peut-on faire pareil pour les programmes concurrents ?



AVANCÉES RÉCENTES SUR CKA
Plan

AVANCÉES RÉCENTES SUR CKA
Plan

I. Introduction


II. Algèbre de Kleene concurrente

III. Observations booléennes

IV. Observations partielles

V. Travaux en cours et à venir

Algèbre de Kleene concurrente

Concurrent Kleene Algebra

C.A.R. Tony Hoare[1], Bernhard Möller[2], Georg Struth[3], and Ian Wehrman[4]

[1] Microsoft Research, Cambridge, UK
[2] Universität Augsburg, Germany
[3] University of Sheffield, UK
[4] University of Texas at Austin, USA

2009

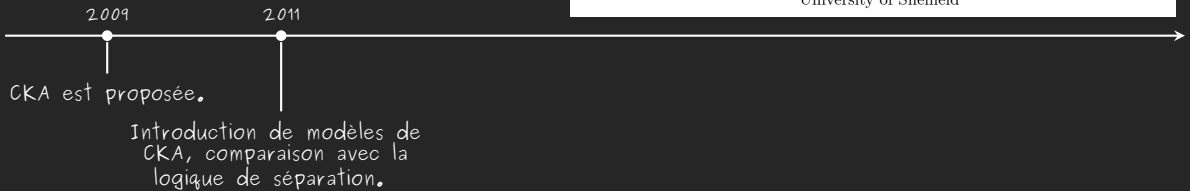
CKA est proposée.

ALGÈBRE DE KLEENE CONCURRENTE

On Locality and the Exchange Law for Concurrent Processes

C.A.R. Hoare[1], Akbar Hussain[2], Bernhard Möller[3], Peter W. O'Hearn[2], Rasmus Lerchedahl Petersen[2], and Georg Struth[4]

[1] Microsoft Research Cambridge
[2] Queen Mary University of London
[3] Universität Augsburg
[4] University of Sheffield



Algèbre de Kleene concurrente

Completeness Theorems for Bi-Kleene Algebras and Series-Parallel Rational Pomset Languages

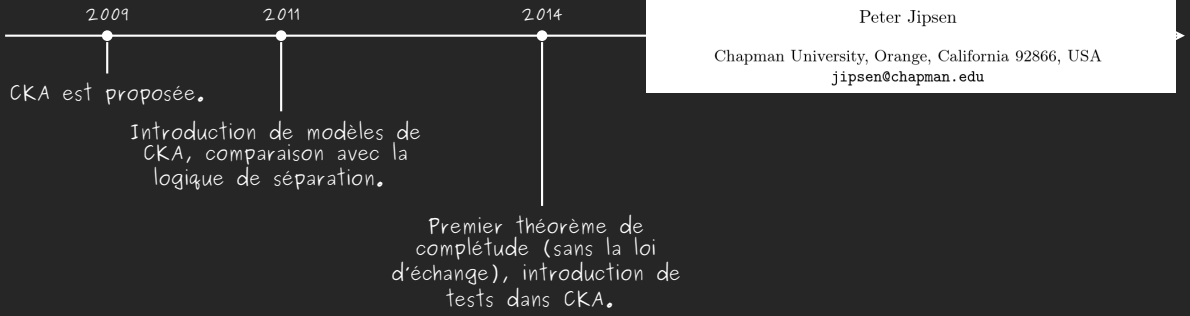
Michael R. Laurence and Georg Struth

Department of Computer Science, University of Sheffield, UK
{m.laurence,g.struth}@sheffield.ac.uk

Concurrent Kleene Algebra with Tests

Peter Jipsen

Chapman University, Orange, California 92866, USA
jipsen@chapman.edu

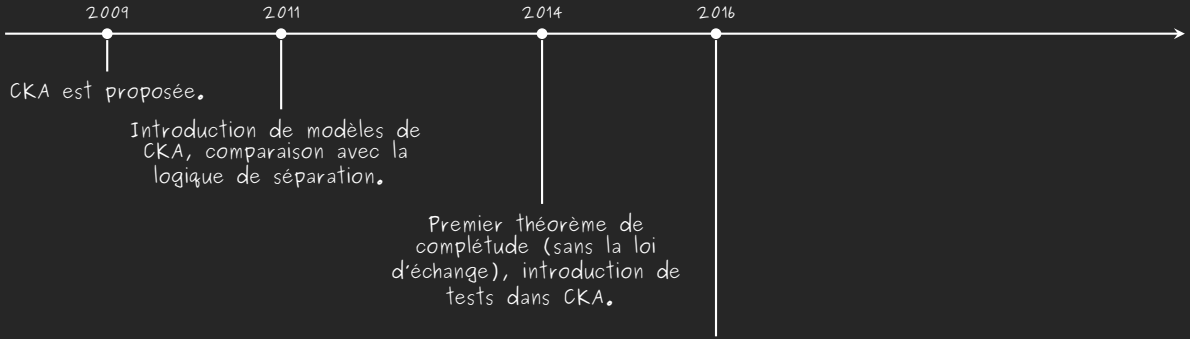


ALGÈBRE DE KLEENE CONCURRENTE

Concurrent Kleene algebra with tests and branching automata

Peter Jipsen *, M. Andrew Moshier

Chapman University, Orange, CA 92866, USA

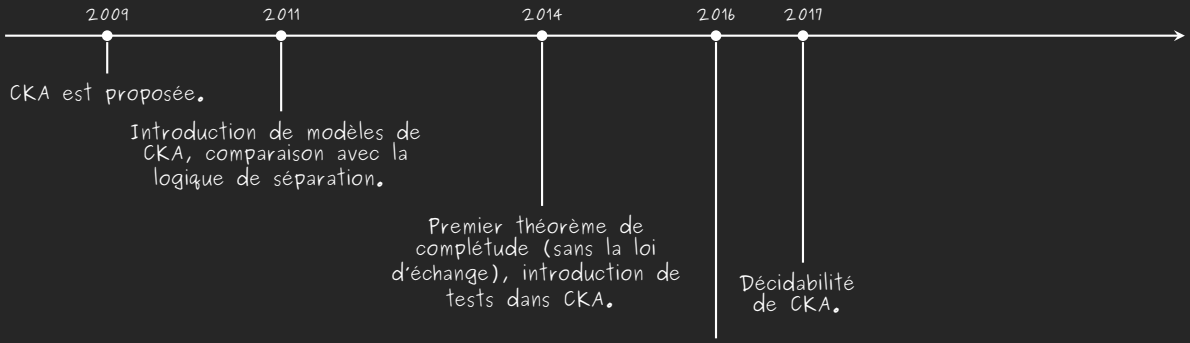


Algèbre de Kleene concurrente

On Decidability of Concurrent Kleene Algebra[*†]

Paul Brunet[1], Damien Pous[2], and Georg Struth[3]

1 Univ. Lyon, CNRS, ENS de Lyon, UCB Lyon 1, LIP, France
2 Univ. Lyon, CNRS, ENS de Lyon, UCB Lyon 1, LIP, France
3 Department of Computer Science, The University of Sheffield, UK

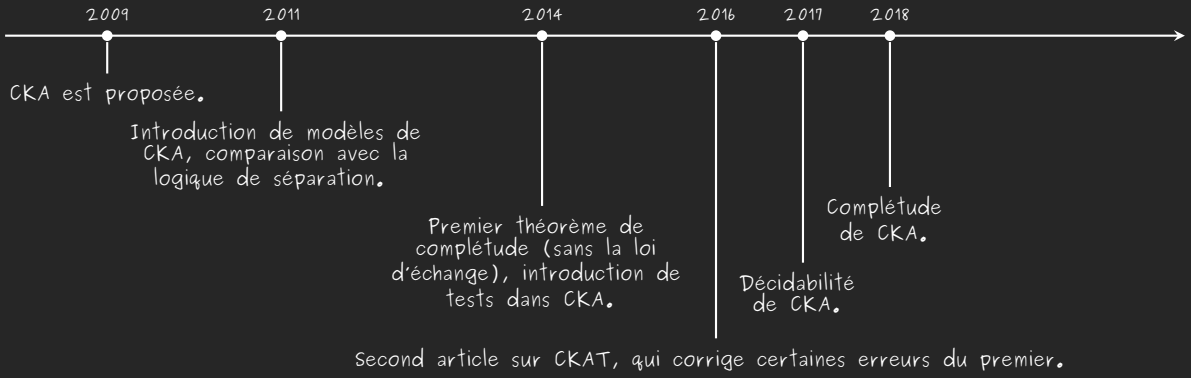


Algèbre de Kleene concurrente

Concurrent Kleene Algebra: Free Model and Completeness

Tobias Kappé(✉), Paul Brunet, Alexandra Silva, and Fabio Zanasi

University College London, London, UK
tkappe@cs.ucl.ac.uk



ALGÈBRE DE KLEENE

Concurrent Kleene Algebra with Observations: from Hypotheses to Completeness

Tobias Kappé (✉), Paul Brunet, Alexandra Silva, Jana Wagemaker, and Fabio Zanasi

University College London, London, United Kingdom; tkappe@cs.ucl.ac.uk

Pomsets with Boxes: Protection, Separation, and Locality in Concurrent Kleene Algebra

Paul Brunet
University College London, UK
paul.brunet-zamansky.fr
paul@brunet-zamansky.fr

David Pym
University College London, UK
www.cantab.net/users/david.pym/
d.pym@ucl.ac.uk

2009 2011 2014 2016 2017 2018 2020

Partially Observable Concurrent Kleene Algebra

Jana Wagemaker
Radboud University, Nijmegen
j.wagemaker@cs.ru.nl

Paul Brunet
University College London

Simon Docherty
University College London

Tobias Kappé
University College London

Jurriaan Rot
Radboud University, Nijmegen and University College London

Alexandra Silva
University College London

CKA es

de
a loi
ion de

CKA avec observations (totales/partielles), CKA avec boîtes.

Complétude de CKA.

Décidabilité de CKA.

Second article sur CKAT, qui corrige certaines erreurs du premier.

BI-ALGÈBRE DE KLEENE

$$e, f \in \mathrm{E}_A^{\mathrm{bika}} ::= 1 \mid 0 \mid a \in A \mid e \cdot f \mid e \parallel f \mid e + f \mid e^* \mid e^!$$

Définition

Une bi-algèbre de Kleene est une structure $\langle \mathscr{A}, 0, 1, \cdot, \parallel, +, \star, ! \rangle$ telle que :

- ☞ $\langle \mathscr{A}, 0, 1, \cdot, +, \star \rangle$ est une KA
- ☞ $\langle \mathscr{A}, 0, 1, \parallel, +, ! \rangle$ est une KA commutative.

BI-ALGÈBRE DE KLEENE

$$e, f \in \mathrm{E}_A^{\mathrm{bika}} ::= 1 \mid 0 \mid a \in A \mid e \cdot f \mid e \parallel f \mid e + f \mid e^* \mid e^!$$

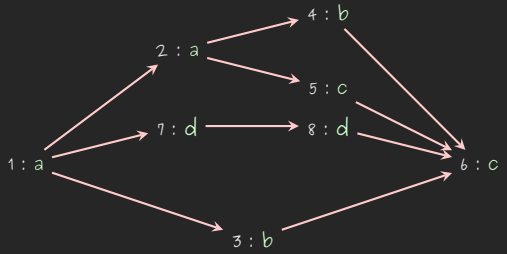
Définition

Une bi-algèbre de Kleene est une structure $\langle \mathscr{A}, 0, 1, \cdot, \parallel, +, \star, ! \rangle$ telle que :

- ☞ $\langle \mathscr{A}, 0, 1, \cdot, +, \star \rangle$ est une KA
- ☞ $\langle \mathscr{A}, 0, 1, \parallel, +, ! \rangle$ est une KA commutative.

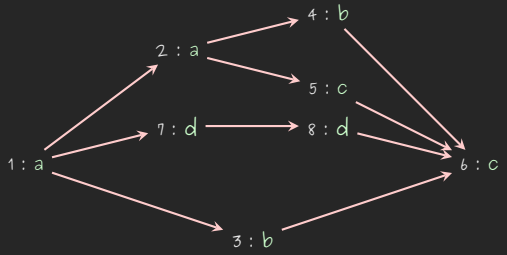
À quoi ressemble la bi-KA libre ?

POMSETS - TRACES CONCURRENTES



POMSETS – TRACES CONCURRENTES

A est un alphabet d'actions.



ensemble fini d'évènements

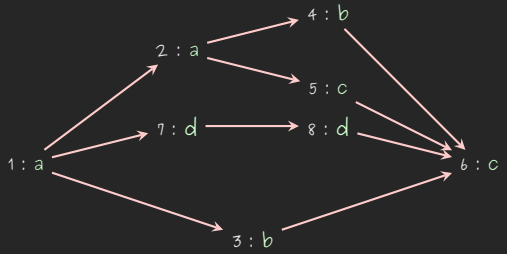
fonction d'étiquetage
$: E_P \to A$

$P = \langle E_P, \leqslant_P, \lambda_P \rangle$

ordre partiel
$\subseteq E_P \times E_P$

POMSETS - TRACES CONCURRENTES

A est un alphabet d'actions.



ensemble fini d'évènements

fonction d'étiquetage
$: E_P \to A$

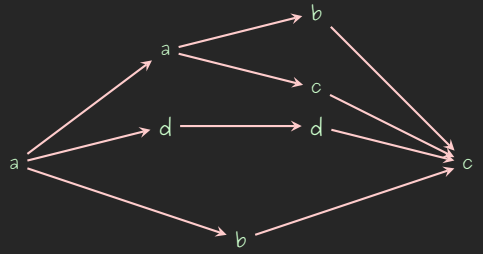
$P = \langle E_P, \leqslant_P, \lambda_P \rangle$

ordre partiel
$\subseteq E_P \times E_P$

À isomorphisme près $\equiv$.

POMSETS – TRACES CONCURRENTES

A est un alphabet d'actions.



ensemble fini d'évènements

ordre partiel
$\subseteq E_P \times E_P$

$P = \langle E_P, \leqslant_P, \lambda_P \rangle$

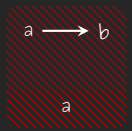
fonction d'étiquetage
: $E_P \to A$

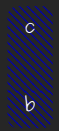
À isomorphisme près $\equiv$.

COMBINER LES POMSETS

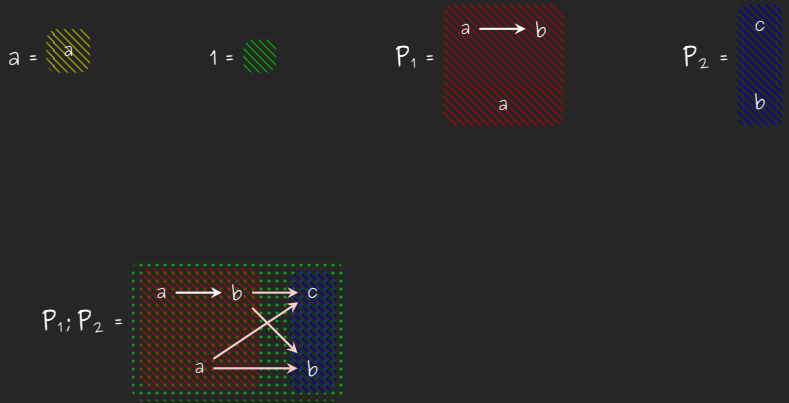
$a =$ 

$1 =$ 

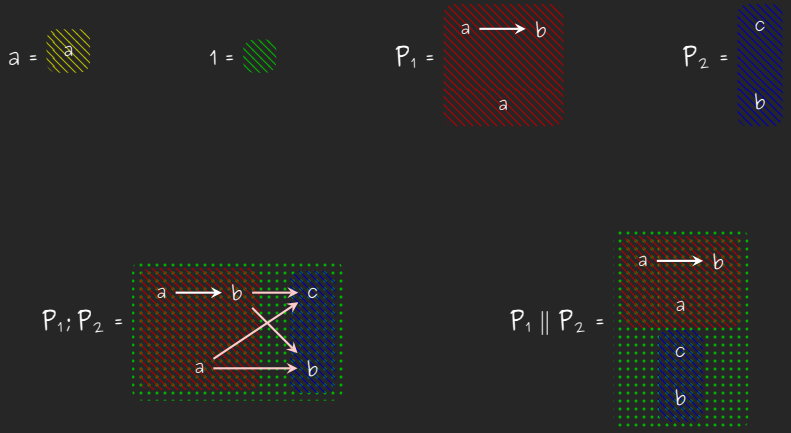
$P_1 =$ 

$P_2 =$ 

COMBINER LES POMSETS



COMBINER LES POMSETS



COMPLÉTUDE DE BIKA

$$[\![1]\!] := \{1\} \qquad [\![0]\!] := \emptyset$$

$$[\![a]\!] := \{a\} \qquad [\![e + f]\!] := [\![e]\!] \cup [\![f]\!]$$

$$[\![e \cdot f]\!] := \left\{ P ; Q \;\middle|\; P \in [\![e]\!], Q \in [\![f]\!] \right\} \qquad [\![e \parallel f]\!] := \left\{ P \parallel Q \;\middle|\; P \in [\![e]\!], Q \in [\![f]\!] \right\}$$

$$[\![e^*]\!] := \{P_1 ; \cdots ; P_n \mid n \in \mathbb{N}, P_i \in [\![e]\!]\} \qquad [\![e^\prime]\!] := \{P_1 \parallel \cdots \parallel P_n \mid n \in \mathbb{N}, P_i \in [\![e]\!]\}$$

Théorème

$$\mathsf{biKA} \vdash e = f \Leftrightarrow [\![e]\!] \equiv [\![f]\!].$$

Laurence & Struth, "Completeness Theorems for Bi-Kleene Algebras and Series-Parallel Rational Pomset Languages", RAMiCS '14

ENTRELACEMENTS ET SUBSOMPTION

Loi d'échange faible

$$(a \parallel b) \cdot (c \parallel d) \leq (a \cdot c) \parallel (b \cdot d).$$

$P \sqsubseteq Q$ quand il y a un homomorphisme de Q à P, c.à.d. une application bijective $\varphi : E_Q \to E_P$ telle que $\lambda_P \circ \varphi = \lambda_Q$ et $\varphi(\leq_Q) \subseteq \leq_P$.

$L^{\sqsubseteq} := \{P \mid \exists Q \in L : P \sqsubseteq Q\}$.

Algèbre de Kleene concurrente

Loi d'échange faible

$$(a \parallel b) \cdot (c \parallel d) \leq (a \cdot c) \parallel (b \cdot d).$$

Pas d'itération parallèle

CKA

Une algèbre de Kleene concurrente est une bi-algèbre de Kleene faible $\langle \mathscr{A}, 0, 1, \cdot, \parallel, +, \star \rangle$ qui satisfait la loi d'échange faible.

COMPLÉTUDE ET DÉCIDABILITÉ DE CKA

Théorème

Tester que deux expressions dénotent le même langage clos est un problème ExpSpace−complet.

B., Pous, & Struth, "On Decidability of Concurrent Kleene Algebra", CONCUR '17

Théorème

$$\mathsf{CKA} \vdash e = f \Leftrightarrow [\![e]\!]^{\sqsubseteq} = [\![f]\!]^{\sqsubseteq}.$$

Kappé, B., Silva, & Zanasi, "Concurrent Kleene Algebra : Free Model and Completeness", ESOP '18

AVANCÉES RÉCENTES SUR CKA
Plan

I. Introduction

II. Algèbre de Kleene concurrente


III. Observations booléennes

IV. Observations partielles

V. Travaux en cours et à venir

CONCURRENT KLEENE ALGEBRA WITH TESTS

Slogan

☞ KAT : KA avec une sous-algèbre booléenne.

☞ CKAT : CKA avec une sous-algèbre booléenne.

CONCURRENT KLEENE ALGEBRA WITH TESTS

Slogan

☞ KAT : KA avec une sous-algèbre booléenne.

☞ CKAT : CKA avec une sous-algèbre booléenne.

$$
\begin{aligned}
t? \cdot p \cdot (\neg t)? &\leq p \parallel (t? \cdot (\neg t)?) && \text{(axiomes de CKA)} \\
&= p \parallel (t \wedge \neg t)? && (a \wedge b)? = a? \cdot b? \\
&= p \parallel \bot? && \text{(axiomes booléens)} \\
&= p \parallel 0 && (\bot? = 0) \\
&= 0 && \text{(axiomes de CKA)}
\end{aligned}
$$

CONCURRENT KLEENE ALGEBRA WITH TESTS

Slogan

☞ KAT : KA avec une sous-algèbre booléenne.

☞ CKAT : CKA avec une sous-algèbre booléenne.

$$
\begin{aligned}
t? \cdot p \cdot (\neg t)? &\leq p \parallel (t? \cdot (\neg t)?) && \text{(axiomes de CKA)} \\
&= p \parallel (t \wedge \neg t)? && (a \wedge b)? = a? \cdot b? \\
&= p \parallel \bot? && \text{(axiomes booléens)} \\
&= p \parallel 0 && (\bot? = 0) \\
&= 0 && \text{(axiomes de CKA)}
\end{aligned}
$$

↔ Pour tout programme et toute assertion, le triplet $\{t\}\, p\, \{t\}$ est valide.

↔ Chaque test est un invariant de tous les programmes.

~~CONCURRENT KLEENE ALGEBRA WITH TESTS~~

Slogan

☞ KAT : KA avec une sous-algèbre booléenne.

☞ CKAT : CKA avec une sous-algèbre booléenne.

$$
\begin{aligned}
t? \cdot p \cdot (\neg t)? &\leq p \parallel (t? \cdot (\neg t)?) && \text{(axiomes de CKA)} \\
&= p \parallel (t \wedge \neg t)? && (a \wedge b)? = a? \cdot b? \\
&= p \parallel \bot? && \text{(axiomes booléens)} \\
&= p \parallel 0 && (\bot? = 0) \\
&= 0 && \text{(axiomes de CKA)}
\end{aligned}
$$

↔ Pour tout programme et toute assertion, le triplet $\{t\}\, p \,\{t\}$ est valide.

↔ Chaque test est un invariant de tous les programmes.

À QUI LA FAUTE?

$$
\begin{aligned}
t?\cdot p\cdot(\neg t)? &\leq p \parallel (t?\cdot(\neg t)?) && \text{(axiomes de CKA)}\\
&= p \parallel (t\wedge\neg t)? && (a\wedge b)? = a?\cdot b?\\
&= p \parallel \bot? && \text{(axiomes booléens)}\\
&= p \parallel 0 = 0 && (\bot? = 0 \text{ + axiomes de CKA})
\end{aligned}
$$

À QUI LA FAUTE?

$$
\begin{aligned}
t?\cdot p\cdot(\neg t)? &\leq p \parallel (t?\cdot(\neg t)?) && \text{(axiomes de CKA)}\\
&= p \parallel (t\wedge\neg t)? && (a\wedge b)? = a?\cdot b?\\
&= p \parallel \bot? && \text{(axiomes booléens)}\\
&= p \parallel 0 = 0 && (\bot? = 0 \text{ + axiomes de CKA})
\end{aligned}
$$

À QUI LA FAUTE?

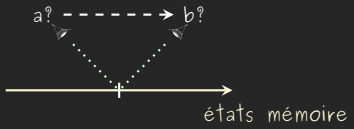
$$
\begin{aligned}
t? \cdot p \cdot (\neg t)? &\leq p \parallel (t? \cdot (\neg t)?) \\
&= p \parallel (t \wedge \neg t)? \\
&= p \parallel \bot? \\
&= p \parallel 0 = 0
\end{aligned}
$$

(axiomes de CKA)

$$(a \wedge b)? = a? \cdot b?$$

(axiomes booléens)

($\bot? = 0$ + axiomes de CKA)



À QUI LA FAUTE?

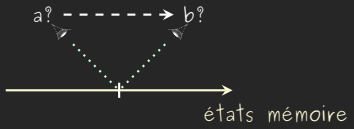
$$
\begin{aligned}
t? \cdot p \cdot (\neg t)? &\leq p \parallel (t? \cdot (\neg t)?) \\
&= p \parallel (t \wedge \neg t)? \\
&= p \parallel \bot? \\
&= p \parallel 0 = 0
\end{aligned}
$$

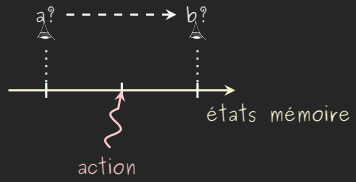
(axiomes de CKA)

$$(a \wedge b)? = a? \cdot b?$$

(axiomes booléens)

($\bot? = 0$ + axiomes de CKA)





À QUI LA FAUTE?

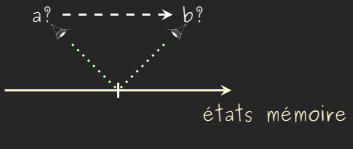
$$
\begin{aligned}
t? \cdot p \cdot (\neg t)? &\leq p \parallel (t? \cdot (\neg t)?) \\
&= p \parallel (t \wedge \neg t)? \\
&= p \parallel \bot? \\
&= p \parallel 0 = 0
\end{aligned}
$$

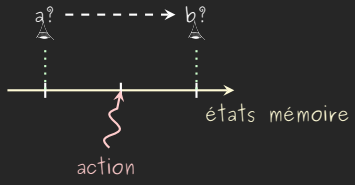
(axiomes de CKA)

~~$(a \wedge b)? = a? \cdot b?$~~

(axiomes booléens)

($\bot? = 0$ + axiomes de CKA)





$$(a \wedge b)? \leq a? \cdot b?$$

CKAO - SYNTAXE

$$e, f \in \mathbf{E}_{A,B}^{\mathrm{ckao}} ::= 0 \mid 1 \mid a \in A \mid t? \mid e \cdot f \mid e \parallel f \mid e + f \mid e^*$$

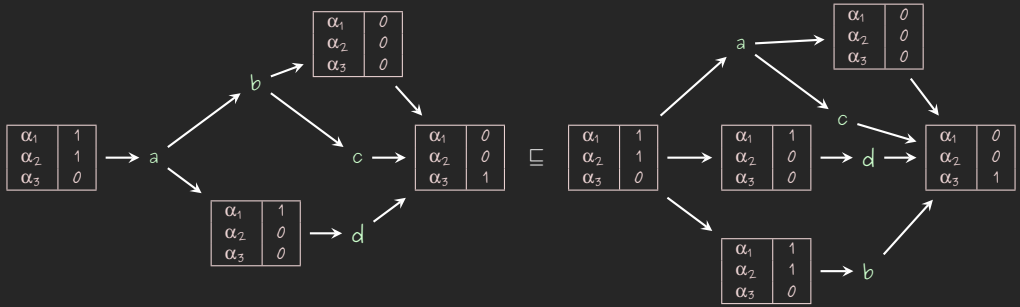
$$t, t_1, t_2 \in \mathbf{O}_A^{\mathrm{bool}} ::= \top \mid \bot \mid \alpha \in B \mid t_1 \wedge t_2 \mid t_1 \vee t_2 \mid \neg t$$

Axiomes de CKAO

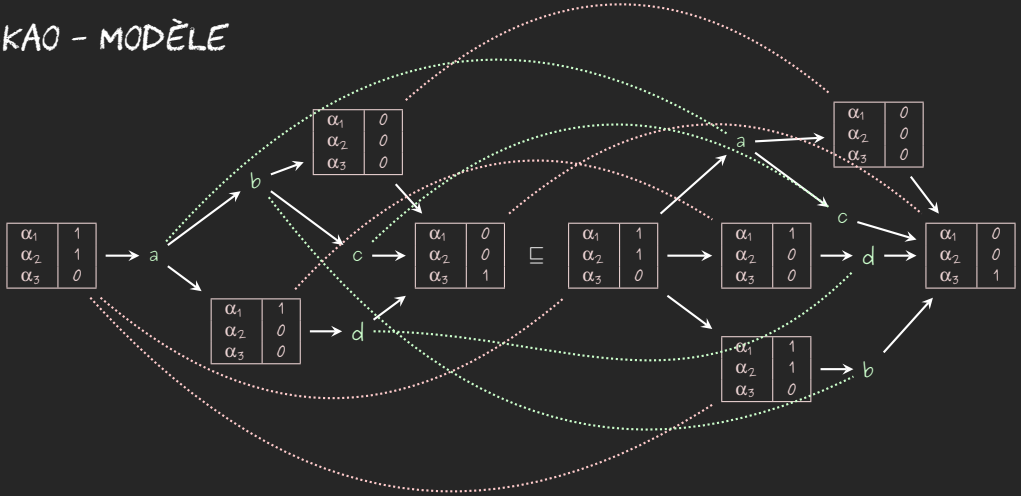
- ☞ Tous les axiomes de CKA.
- ☞ Pour les obsertions, les axiomes booléens.
- ☞ Les axiomes "glue" suivants :

$$(t_1 \vee t_2)? = t_1? + t_2? \qquad (t_1 \wedge t_2)? \leq t_1? \cdot t_2? \qquad \bot? = 0$$

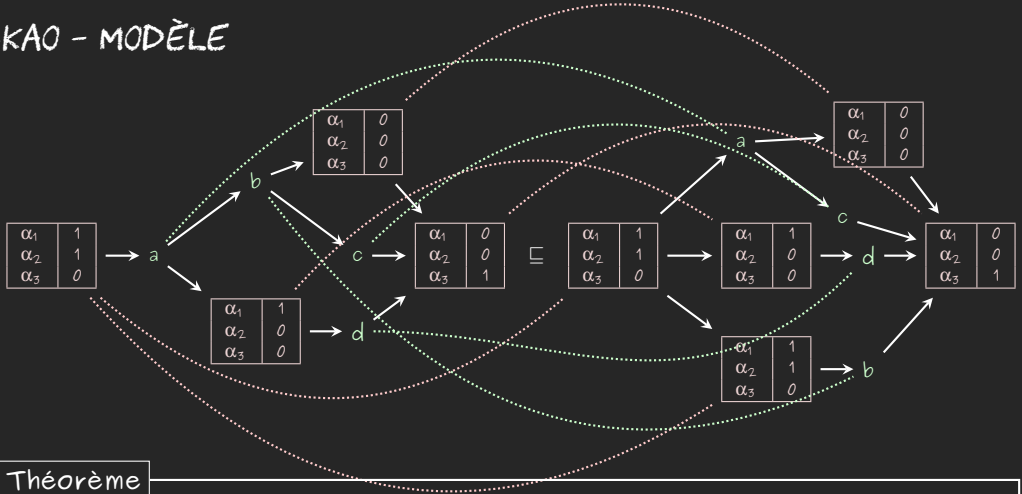
CKAO - MODÈLE


α1 1
α2 1
α3 0
a
b
α1 0
α2 0
α3 0
c
α1 0
α2 0
α3 1
α1 1
α2 0
α3 0
d
⊑
α1 1
α2 1
α3 0
a
α1 0
α2 0
α3 0
c
α1 1
α2 0
α3 0
d
α1 0
α2 0
α3 1
α1 1
α2 1
α3 0
b

CKAO - MODÈLE


α_1 1
α_2 1
α_3 0
a
b
α_1 0
α_2 0
α_3 0
c
α_1 1
α_2 0
α_3 0
d
α_1 0
α_2 0
α_3 1
$\sqsubseteq$
α_1 1
α_2 1
α_3 0
a
α_1 0
α_2 0
α_3 0
c
α_1 1
α_2 0
α_3 0
d
α_1 0
α_2 0
α_3 1
α_1 1
α_2 1
α_3 0
b

CKAO - MODÈLE



Théorème

$$\mathsf{CKAO} \vdash e = f \Leftrightarrow [\![e]\!]\downarrow = [\![f]\!]\downarrow_\bullet$$

Kappé, B., Silva, Wagemaker, & Zanasi, "Concurrent Kleene Algebra with Observations : from Hypotheses to Completeness", FoSSaCS '20

INTERLUDE – (C)KA MODULO HYPOTHÈSES

H : ensemble d'hypothèses $e \leq f$ sur un alphabet A donné.

- ☞ structure algébrique de l'alphabet (e.g. $\alpha \wedge \beta = \beta \wedge \alpha$) ;
- ☞ propriétés dynamiques (e.g. $\alpha \leq \alpha \cdot \alpha$)
- ☞ autres propriétés du système modélisé.

Théorème

$$\mathrm{CKA} + H \vdash e = f \Rightarrow [\![e]\!]\downarrow^H = [\![f]\!]\downarrow^H$$

- ☞ Doumane, Kuperberg, Pous, & Pradic, "Kleene Algebra with Hypotheses", FoSSaCS '19
- ☞ Kappé, B., Silva, Wagemaker, & Zanasi, "Concurrent Kleene Algebra with Observations : from Hypotheses to Completeness", FoSSaCS '20

AVANCÉES RÉCENTES SUR CKA
Plan

I. Introduction

II. Algèbre de Kleene concurrente

III. Observations booléennes

IV. Observations partielles

V. Travaux en cours et à venir

LITMUS TEST - COHÉRENCE SÉQUENTIELLE

```
{ r0 == 0 && r1 == 0 }

 x := 1    ||    y := 1
 r0 := y   ||    r1 := x

{ r0 == 1 || r1 == 1 }
```

Ingrédients :

- ☞ Affectations $x \leftarrow 1$
- ☞ Observations $r_0 \mapsto 0$

DE QUELLES OBSERVATIONS AVONS NOUS BESOIN ?
Première tentative : algèbre booléenne

☞ Observations atomiques : $\mathrm{VAR} \mapsto \mathrm{VAL}$ e.g. $r_0 \mapsto 1$

DE QUELLES OBSERVATIONS AVONS NOUS BESOIN ?
Première tentative : algèbre booléenne

☞ Observations atomiques : $\mathrm{Var} \mapsto \mathrm{Val}$ e.g. $r_0 \mapsto 1$

☞ Formules booléennes : ensembles d'états mémoire $\mathrm{Var} \to \mathrm{Val}$ e.g.

r_0	1
r_1	0

DE QUELLES OBSERVATIONS AVONS NOUS BESOIN ?
Première tentative : algèbre booléenne

☞ Observations atomiques : $\mathrm{VAR} \mapsto \mathrm{VAL}$ e.g. $r_0 \mapsto 1$

☞ Formules booléennes : ensembles d'états mémoire $\mathrm{VAR} \to \mathrm{VAL}$ e.g.

r_0	1
r_1	0

☞ Affectations : $\sum_{s \in \mathrm{State}} s \cdot (v \leftarrow n) \cdot s[v \mapsto n]$, c.à.d.

$$[\![x \leftarrow 1]\!] := \left\{ \begin{array}{|c|c|}\hline x & 0 \\ \hline y & 0 \\ \hline \end{array} \to [x \leftarrow 1] \to \begin{array}{|c|c|}\hline x & 1 \\ \hline y & 0 \\ \hline \end{array} \;,\; \begin{array}{|c|c|}\hline x & 0 \\ \hline y & 1 \\ \hline \end{array} \to [x \leftarrow 1] \to \begin{array}{|c|c|}\hline x & 1 \\ \hline y & 1 \\ \hline \end{array} \;, \ldots \right\}$$

DE QUELLES OBSERVATIONS AVONS NOUS BESOIN ?
Première tentative : algèbre booléenne

☞ Observations atomiques : $\mathrm{VAR} \mapsto \mathrm{VAL}$ e.g. $r_0 \mapsto 1$

☞ Formules booléennes : ensembles d'états mémoire $\mathrm{VAR} \to \mathrm{VAL}$ e.g. $\{ r_0 \mapsto 1, r_1 \mapsto 0 \}$

☞ Affectations : $\sum_{s \in \mathrm{State}} s \cdot (v \leftarrow n) \cdot s[v \mapsto n]$, c.à.d.

$$[\![x \leftarrow 1]\!] := \left\{ \begin{array}{|c|c|}\hline x & 0 \\ \hline y & 0 \\ \hline \end{array} \to [x \leftarrow 1] \to \begin{array}{|c|c|}\hline x & 1 \\ \hline y & 0 \\ \hline \end{array}, \begin{array}{|c|c|}\hline x & 0 \\ \hline y & 1 \\ \hline \end{array} \to [x \leftarrow 1] \to \begin{array}{|c|c|}\hline x & 1 \\ \hline y & 1 \\ \hline \end{array}, \ldots \right\}$$

Problème : composition parallèle ?

$$\begin{array}{|c|c|}\hline x & 0 \\ \hline y & 0 \\ \hline \end{array} \to [x \leftarrow 1] \to \begin{array}{|c|c|}\hline x & 1 \\ \hline y & 0 \\ \hline \end{array}$$

$$\begin{array}{|c|c|}\hline x & 0 \\ \hline y & 0 \\ \hline \end{array} \to [y \leftarrow 1] \to \begin{array}{|c|c|}\hline x & 0 \\ \hline y & 1 \\ \hline \end{array}$$

ALGÈBRE D'OBSERVATIONS PARTIELLES

Idée : au lieu d'états mémoire $V_{AR} \to V_{AL}$, on considère des fonctions partielles $V_{AR} \rightharpoonup V_{AL}$.

PCDL d'observations

$$t, t_1, t_2 \in \mathcal{O}_A^{\text{pocka}} ::= \top \mid \bot \mid \alpha \in B \mid t_1 \wedge t_2 \mid t_1 \vee t_2 \mid \overline{t}$$

Mêmes axiomes que BA pour $\vee, \wedge, \top, \bot$, plus :

☞ $p \leq \overline{q} \Leftrightarrow p \wedge q = \bot$ définition du pseudo-complément

☞ $\overline{v \mapsto n} = \bigvee_{m \neq n} v \mapsto m$

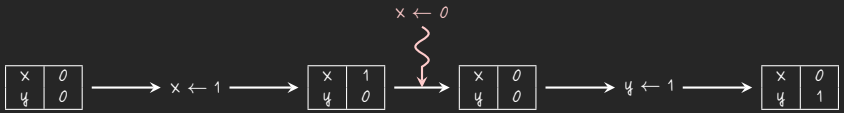
PCDL : trellis distributif pseudo-complémenté

Théorème

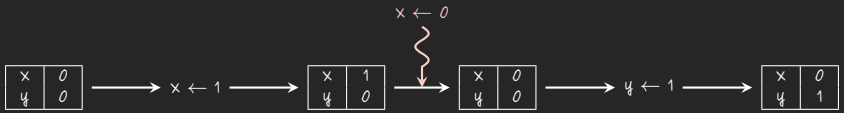
$$\text{POCKA} \vdash e = f \Leftrightarrow [\![e]\!]\downarrow^{\text{pocka}} = [\![f]\!]\downarrow^{\text{pocka}}.$$

Wagemaker, B., Docherty, Kappé, Rot, & Silva, "Partially Observable Concurrent Kleene Algebra", CONCUR '20

CAUSALITÉ Vs COMPOSITIONNALITÉ



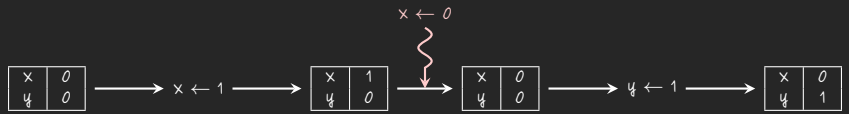
CAUSALITÉ Vs COMPOSITIONNALITÉ



Solution : il faut « clore » le système explicitement.

$$[\![e]\!] \quad \rightarrow \quad [\![e]\!] \cap \text{CausalPomsets}.$$

CAUSALITÉ VS COMPOSITIONNALITÉ



Solution : il faut « clore » le système explicitement.

$$[\![e]\!] \quad \to \quad [\![e]\!] \cap \text{CausalPomsets}.$$

Litmus test :

$$t := (r_0 \mapsto 0 \wedge r_1 \mapsto 0) \cdot ((x \leftarrow 1 \cdot r_0 \leftarrow y) \parallel (y \leftarrow 1 \cdot r_1 \leftarrow x)) \cdot \overline{(r_0 \mapsto 1 \vee r_1 \mapsto 1)}$$

$$[\![t]\!] \cap \text{CausalPomsets} = \emptyset$$

AVANCÉES RÉCENTES SUR CKA
Plan

I. Introduction

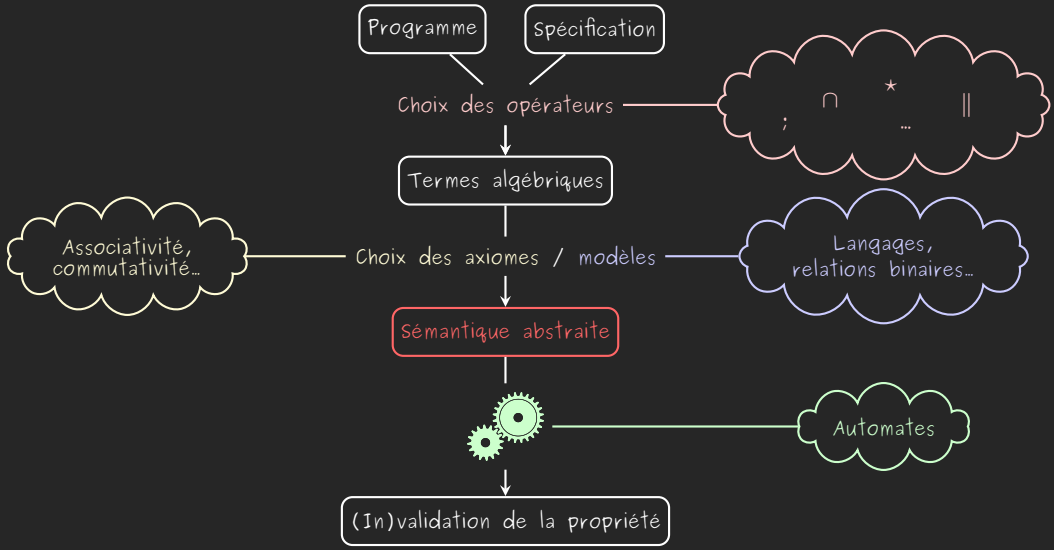
II. Algèbre de Kleene concurrente

III. Observations booléennes

IV. Observations partielles


V. Travaux en cours et à venir

PROJET DE RECHERCHE : ALGÈBRES DE PROGRAMMES



MÉTA-THÉORIE

☞ Construction modulaire d'algèbres de programme

MÉTA-THÉORIE

- ☞ Construction modulaire d'algèbres de programme
  - ▶ Quotienter un modèle : algèbres à hypothèses

MÉTA-THÉORIE

- ☞ Construction modulaire d'algèbres de programme
  - Quotienter un modèle : algèbres à hypothèses
    - KA : Doumane, Kuperberg, Pous, & Pradic, "Kleene Algebra with Hypotheses", FoSSaCS '19

MÉTA-THÉORIE

- ☞ Construction modulaire d'algèbres de programme
  - Quotienter un modèle : algèbres à hypothèses
    - KA : Doumane, Kuperberg, Pous, & Pradic, "Kleene Algebra with Hypotheses", FoSSaCS '19
    - CKA : Kappé, B., Silva, Wagemaker, & Zanasi, "Concurrent Kleene Algebra with Observations : from Hypotheses to Completeness", FoSSaCS '20

MÉTA-THÉORIE

- ☞ Construction modulaire d'algèbres de programme
  - Quotienter un modèle : algèbres à hypothèses
    - KA : Doumane, Kuperberg, Pous, & Pradic, "Kleene Algebra with Hypotheses", FoSSaCS '19
    - CKA : Kappé, B., Silva, Wagemaker, & Zanasi, "Concurrent Kleene Algebra with Observations : from Hypotheses to Completeness", FoSSaCS '20
    - en général ?

MÉTA-THÉORIE

☞ Construction modulaire d'algèbres de programme

- Quotienter un modèle : algèbres à hypothèses
  - KA : Doumane, Kuperberg, Pous, & Pradic, "Kleene Algebra with Hypotheses", FoSSaCS '19
  - CKA : Kappé, B., Silva, Wagemaker, & Zanasi, "Concurrent Kleene Algebra with Observations : from Hypotheses to Completeness", FoSSaCS '20
  - en général ?
- Donner une structure à l'alphabet : algèbres empilées (e.g. KAT, CKAO, POCKA, …)

MÉTA-THÉORIE

☞ Construction modulaire d'algèbres de programme
- Quotienter un modèle : algèbres à hypothèses
  - KA : Doumane, Kuperberg, Pous, & Pradic, "Kleene Algebra with Hypotheses", FoSSaCS '19
  - CKA : Kappé, B., Silva, Wagemaker, & Zanasi, "Concurrent Kleene Algebra with Observations : from Hypotheses to Completeness", FoSSaCS '20
  - en général ?
- Donner une structure à l'alphabet : algèbres empilées (e.g. KAT, CKAO, POCKA, …)
- Ajouter des opérateurs : "Fusion" d'algèbres

MÉTA-THÉORIE

☞ Construction modulaire d'algèbres de programme

- Quotienter un modèle : algèbres à hypothèses
  - KA : Doumane, Kuperberg, Pous, & Pradic, "Kleene Algebra with Hypotheses", FoSSaCS '19
  - CKA : Kappé, B., Silva, Wagemaker, & Zanasi, "Concurrent Kleene Algebra with Observations : from Hypotheses to Completeness", FoSSaCS '20
  - en général ?
- Donner une structure à l'alphabet : algèbres empilées (e.g. KAT, CKAO, POCKA, …)
- Ajouter des opérateurs : "Fusion" d'algèbres
  - (e.g. KA + KA commutative = biKA)

MÉTA-THÉORIE

- ☞ Construction modulaire d'algèbres de programme
  - Quotienter un modèle : algèbres à hypothèses
    - KA : Doumane, Kuperberg, Pous, & Pradic, "Kleene Algebra with Hypotheses", FoSSaCS '19
    - CKA : Kappé, B., Silva, Wagemaker, & Zanasi, "Concurrent Kleene Algebra with Observations : from Hypotheses to Completeness", FoSSaCS '20
    - en général ?
  - Donner une structure à l'alphabet : algèbres empilées (e.g. KAT, CKAO, POCKA, …)
  - Ajouter des opérateurs : "Fusion" d'algèbres
    - (e.g. KA + KA commutative = biKA)
- ☞ Techniques :

MÉTA-THÉORIE

- ☞ Construction modulaire d'algèbres de programme
  - Quotienter un modèle : algèbres à hypothèses
    - KA : Doumane, Kuperberg, Pous, & Pradic, "Kleene Algebra with Hypotheses", FoSSaCS '19
    - CKA : Kappé, B., Silva, Wagemaker, & Zanasi, "Concurrent Kleene Algebra with Observations : from Hypotheses to Completeness", FoSSaCS '20
    - en général ?
  - Donner une structure à l'alphabet : algèbres empilées (e.g. KAT, CKAO, POCKA, …)
  - Ajouter des opérateurs : "Fusion" d'algèbres
    - (e.g. KA + KA commutative = biKA)
- ☞ Techniques :
  - catégories, réécriture, algèbre universelle, théorie des automates, réseaux de Petri, théorie de la démonstration…

MÉTA-THÉORIE

- ☞ Construction modulaire d'algèbres de programme
  - Quotienter un modèle : algèbres à hypothèses
    - KA : Doumane, Kuperberg, Pous, & Pradic, "Kleene Algebra with Hypotheses", FoSSaCS '19
    - CKA : Kappé, B., Silva, Wagemaker, & Zanasi, "Concurrent Kleene Algebra with Observations : from Hypotheses to Completeness", FoSSaCS '20
    - en général ?
  - Donner une structure à l'alphabet : algèbres empilées (e.g. KAT, CKAO, POCKA, …)
  - Ajouter des opérateurs : "Fusion" d'algèbres
    - (e.g. KA + KA commutative = biKA)
- ☞ Techniques :
  - catégories, réécriture, algèbre universelle, théorie des automates, réseaux de Petri, théorie de la démonstration…
- ☞ Formalisation en Coq 

Application à la vérification de programmes

☞ Instances du modèle

Application à la vérification de programmes

- ☞ Instances du modèle
  - ▶ Langages métiers (DSL)
    en s'inspirant de NetKAT (langage basé sur KAT permettant de spécifier et vérifier les SDN)

Application à la vérification de programmes

☞ Instances du modèle
- Langages métiers (DSL)
  en s'inspirant de NetKAT (langage basé sur KAT permettant de spécifier et vérifier les SDN)
- Algèbre dédiée à un seul type de bugs
  e.g. détection d'interblocage

Application à la vérification de programmes

- ☞ Instances du modèle
  - Langages métiers (DSL)
    en s'inspirant de NetKAT (langage basé sur KAT permettant de spécifier et vérifier les SDN)
  - Algèbre dédiée à un seul type de bugs
    e.g. détection d'interblocage
- ☞ Implémentation d'outils automatiques

Application à la vérification de programmes

- ☞ Instances du modèle
  - Langages métiers (DSL)
    en s'inspirant de NetKAT (langage basé sur KAT permettant de spécifier et vérifier les SDN)
  - Algèbre dédiée à un seul type de bugs
    e.g. détection d'interblocage
- ☞ Implémentation d'outils automatiques
- ☞ Model-checking : logiques de comportements

  B. & Pym, "Pomsets with Boxes : Protection, Separation, and Locality in Concurrent Kleene Algebra.", FSCD '20

C'EST TOUT !

Merci de votre attention !

https://paul.brunet-zamansky.fr

AVANCÉES RÉCENTES SUR CKA
Plan

QUI SUIS-JE?

Diplômes

- 🎓 Licence d'informatique (ENS de Cachan antenne de Bretagne) 2009
- 🎓 Master MPRI (Université Paris VII) 2013
- 🎓 Doctorat d'informatique (Université de Lyon) 2016
  « Algèbres de Relations : des algorithmes aux preuves formelles »
  thèse effectuée avec Damien Pous au LIP (ENS de Lyon)

QUI SUIS-JE?

Diplômes

- Licence d'informatique (ENS de Cachan antenne de Bretagne) 2009
- Master MPRI (Université Paris VII) 2013
- Doctorat d'informatique (Université de Lyon) 2016
  « Algèbres de Relations : des algorithmes aux preuves formelles »
  thèse effectuée avec Damien Pous au LIP (ENS de Lyon)

Enseignements

- University College London 2017–2019
  - Calculabilité and complexité (niveau L3/M1)
- ÉNS de Lyon 2016
  - Sémantique and Vérification (niveau M1)
- UCB Lyon 1 2013–2016
  - Calculabilité and complexité (niveau M1)
  - Théorie des langages formels (niveau L3)
  - Logique classique (niveau L3)
  - Algorithmique et Programmation Impérative (niveau L1)
  - Algorithmique numérique (niveau L3)

ACTIVITÉS DE RECHERCHE, VUES DE LOIN...

Emploi

- Postdoc à l'ENS de Lyon avec Damien Pous (2016)
- Postdoc à UCL (GB) avec Alexandra Silva (2017–2018)
- Postdoc à UCL (GB) avec David Pym (2018–…)

Co-auteurs

- D. Pous (ENS de Lyon)
- A. Silva, D. Pym, F. Zanasi, T. Kappé, S. Docherty (UCL)
- B. Luttik (TU Eindhoven)
- J. Rot, J. Wagemaker (Radboud University, Nijmegen)
- G. Struth (University of Sheffield)
- I. Stucke (Kiel University)

Publications

Journaux

- JLAMP (2016, 2019)
- LMCS (2017)

Conférences

- LiCS (2015)
- ICALP (2019)
- CONCUR (2017x2, 2019, 2020)
- FSCD (2020)
- FoSSaCS (2020)
- ESOP (2018)
- MFCS (2016, 2017)
- CSL (2020)
- RAMiCS (2014)

Et aussi

- Membre de comités de programme : WoLLIC 2021, CONCUR 2021.
- Jury de thèse de Joshua Moerman en juillet 2019 (Radboud University Nijmegen, NL)

ARTICLES, PAR THÈMES

Relations

- ☞ Réciproque RAMiCS 2014, JLAMP 2016
- ☞ Intersection LiCS 2015, LMCS 2017
- ☞ *Programmes relationels* ITP 2016

Thèse 2016

Concurrence

- ☞ biKA CONCUR 2017 (Kappé et al.), JLAMP 2019
- ☞ CKA CONCUR 2017 (Pous et al.), ESOP 2018
- ☞ Observations FoSSaCS 2020, CONCUR 2020
- ☞ Boîtes FSCD 2020

Nominaux

- ☞ NKA MFCS 2016
- ☞ Parenthèses ICALP 2019

Traces

- ☞ Treillis MFCS 2017, CSL 2020
- ☞ Observations CONCUR 2019