Algèbres de Relations, Étude des algèbres de Kleene avec converse

Soutenance de stage de recherche de M2, réalisé dans l'équipe PLUME du LIP

Paul Brunet supervisé par Damien Pous

ENS de Lyon

Lundi 2 septembre 2013

Introduction

Algèbres de Kleene[(i)] : Abstraction pour prouver des résultats directement sur les expressions régulières.

(i). Conway, J. (1971). *Regular algebra and finite machines.*

Introduction

Algèbres de Kleene [(i)] : Abstraction pour prouver des résultats directement sur les expressions régulières.

Algèbres de Kleene Étendues : on souhaite ajouter des opérations aux expressions régulières.

(i). Conway, J. (1971). *Regular algebra and finite machines.*

Introduction

Algèbres de Kleene [(i)] : Abstraction pour prouver des résultats directement sur les expressions régulières.

Algèbres de Kleene Étendues : on souhaite ajouter des opérations aux expressions régulières.

Ici, on s'intéressera aux *Algèbres de Kleene avec Converse*.

(i). Conway, J. (1971). *Regular algebra and finite machines.*

Plan

Plan

Algèbres de Kleene

Une algèbre Kleene (telle que présentée par D. Kozen) est une structure algébrique $\langle K, +, \cdot, ^* , \mathbb{0}, \mathbb{1} \rangle$ telle que

Algèbres de Kleene

Une algèbre Kleene (telle que présentée par D. Kozen) est une structure algébrique $\langle K, +, \cdot, ^*, \mathbb{0}, \mathbb{1} \rangle$ telle que

$\langle K, +, \mathbb{0} \rangle$ monoïde $\quad a + (b + c) = (a + b) + c$

commutatif $\quad a + b = b + a$

idempotent : $\quad a + \mathbb{0} = a$

$\quad a + a = a$

Algèbres de Kleene

Une algèbre Kleene (telle que présentée par D. Kozen) est une structure algébrique $\langle K, +, \cdot, ^*, \mathbb{0}, \mathbb{1} \rangle$ telle que

$\langle K, +, \mathbb{0} \rangle$ monoïde	$a + (b + c)$	$=$	$(a + b) + c$
commutatif	$a + b$	$=$	$b + a$
idempotent :	$a + \mathbb{0}$	$=$	a
	$a + a$	$=$	a
$\langle K, \cdot, \mathbb{1} \rangle$ monoïde :	$a \cdot (b \cdot c)$	$=$	$(a \cdot b) \cdot c$
	$\mathbb{1} \cdot a$	$=$	a
	$a \cdot \mathbb{1}$	$=$	a

Algèbres de Kleene

Une algèbre Kleene (telle que présentée par D. Kozen) est une structure algébrique $\langle K, +, \cdot, ^*, \mathbb{0}, \mathbb{1} \rangle$ telle que

$\langle K, +, \mathbb{0} \rangle$ monoïde	$a + (b + c)$	$=$	$(a + b) + c$
commutatif	$a + b$	$=$	$b + a$
idempotent :	$a + \mathbb{0}$	$=$	a
	$a + a$	$=$	a
$\langle K, \cdot, \mathbb{1} \rangle$ monoïde :	$a \cdot (b \cdot c)$	$=$	$(a \cdot b) \cdot c$
	$\mathbb{1} \cdot a$	$=$	a
	$a \cdot \mathbb{1}$	$=$	a
Distributivité :	$a \cdot (b + c)$	$=$	$a \cdot b + a \cdot c$
	$(a + b) \cdot c$	$=$	$a \cdot c + b \cdot c$
	$\mathbb{0} \cdot a$	$=$	$\mathbb{0}$
	$a \cdot \mathbb{0}$	$=$	$\mathbb{0}$

$a \leqslant b \overset{\Delta}{\iff} a + b = b$

Algèbres de Kleene

Une algèbre Kleene (telle que présentée par D. Kozen) est une structure algébrique $\langle K, +, \cdot, ^*, \mathbb{0}, \mathbb{1} \rangle$ telle que

$\langle K, +, \mathbb{0} \rangle$ monoïde	$a + (b + c)$	$=$	$(a + b) + c$
commutatif	$a + b$	$=$	$b + a$
idempotent :	$a + \mathbb{0}$	$=$	a
	$a + a$	$=$	a
$\langle K, \cdot, \mathbb{1} \rangle$ monoïde :	$a \cdot (b \cdot c)$	$=$	$(a \cdot b) \cdot c$
	$\mathbb{1} \cdot a$	$=$	a
	$a \cdot \mathbb{1}$	$=$	a
Distributivité :	$a \cdot (b + c)$	$=$	$a \cdot b + a \cdot c$
	$(a + b) \cdot c$	$=$	$a \cdot c + b \cdot c$
	$\mathbb{0} \cdot a$	$=$	$\mathbb{0}$
	$a \cdot \mathbb{0}$	$=$	$\mathbb{0}$
Étoile :	$1 + aa^*$	$\leqslant$	a^*
	$1 + a^*a$	$\leqslant$	a^*
	$b + ax \leqslant x$	$\Rightarrow$	$a^*b \leqslant x$
	$b + xa \leqslant x$	$\Rightarrow$	$ba^* \leqslant x$

$a \leqslant b \overset{\Delta}{\iff} a + b = b$

Algèbres de Kleene

Une algèbre Kleene (telle que présentée par D. Kozen) est une structure algébrique $\langle K, +, \cdot, ^*, \mathbb{0}, \mathbb{1} \rangle$ telle que

$$\left.\begin{array}{rcl}
a + (b + c) & = & (a + b) + c \\
a + b & = & b + a \\
a + \mathbb{0} & = & a \\
a + a & = & a \\
a \cdot (b \cdot c) & = & (a \cdot b) \cdot c \\
\mathbb{1} \cdot a & = & a \\
a \cdot \mathbb{1} & = & a \\
a \cdot (b + c) & = & a \cdot b + a \cdot c \\
(a + b) \cdot c & = & a \cdot c + b \cdot c \\
\mathbb{0} \cdot a & = & \mathbb{0} \\
a \cdot \mathbb{0} & = & \mathbb{0} \\
1 + aa^* & \leqslant & a^* \\
1 + a^*a & \leqslant & a^* \\
b + ax \leqslant x & \Rightarrow & a^*b \leqslant x \\
b + xa \leqslant x & \Rightarrow & ba^* \leqslant x
\end{array}\right\} : \text{Théorie KA}$$

$$a \leqslant b \overset{\Delta}{\iff} a + b = b$$

Interprétations

Expressions régulières sur X [a]

a. Kleene, S. (1951). *Representation of Events in Nerve Nets and Finite Automata*

Soit X un ensemble fini, l'ensemble des expressions régulières sur X (noté Reg_X) sont générées par la grammaire :

$$e, f ::= \mathbb{0} | \mathbb{1} | x \in X | e + f | e \cdot f | e^*$$

Interprétations

Expressions régulières sur X [a]

a. Kleene, S. (1951). *Representation of Events in Nerve Nets and Finite Automata*

Soit X un ensemble fini, l'ensemble des expressions régulières sur X (noté Reg_X) sont générées par la grammaire :

$$e, f ::= \mathbb{0} | \mathbb{1} | x \in X | e + f | e \cdot f | e^*$$

Une interprétation des expressions régulières sur l'alphabet X dans une algèbre de Kleene $\langle K, +, \cdot, ^*, 0, 1 \rangle$ est spécifiée par une application

$$\sigma : X \longrightarrow K$$

On notera

$$\hat{\sigma} : Reg(X) \longrightarrow K$$

l'unique morphisme égal à σ sur X.

Langages et relations

Deux classes d'algèbres de Kleene nous intéressent particulièrement : les langages (sur un alphabet fini Σ, avec l'union et la concaténation) et les relations (sur un ensemble quelconque S, avec l'union et la composition).

Langages et relations

Deux classes d'algèbres de Kleene nous intéressent particulièrement : les langages (sur un alphabet fini Σ, avec l'union et la concaténation) et les relations (sur un ensemble quelconque S, avec l'union et la composition).

Notations

- $e \equiv_{Lang} f$: $e = f$ est vraie pour toute interprétation dans un modèle de langage ;
- et $e \equiv_{Rel} f$: de même dans les modèles relationnels

Langages et relations

Deux classes d'algèbres de Kleene nous intéressent particulièrement : les langages (sur un alphabet fini Σ, avec l'union et la concaténation) et les relations (sur un ensemble quelconque S, avec l'union et la composition).

Notations

- $e \equiv_{Lang} f$: $e = f$ est vraie pour toute interprétation dans un modèle de langage ;
- et $e \equiv_{Rel} f$: de même dans les modèles relationnels

Théorème

Les langages et les relations sont des modèles initiaux de KA, c'est à dire que, si on fixe X, alors pour toutes expressions e et f de Reg_X :

- $\mathrm{KA} \vdash e = f \Leftrightarrow e \equiv_{Lang} f$
- $\mathrm{KA} \vdash e = f \Leftrightarrow e \equiv_{Rel} f$

Décidabilité

On note $[\![e]\!] \subseteq X^*$ le *langage dénoté par* e.

Théorème

$$\mathsf{KA} \vdash e = f \Leftrightarrow [\![e]\!] = [\![f]\!]$$

Or l'égalité de tels langages est décidable, grâce au théorème de Kleene [(ii)].

(ii). Kleene, S. (1951). *Representation of Events in Nerve Nets and Finite Automata*

Bilan

Soient $e, f \in \mathrm{Reg}_X$,

$$e \equiv_{Lang} f \iff \mathrm{KA} \vdash e = f \iff e \equiv_{Rel} f$$

$$\Updownarrow$$

$$[\![e]\!] = [\![f]\!]$$

(décidable)

Plan

Expressions régulières avec converse

- Bloom, S. L., Ésik, Z., and Stefanescu, G. (1995). Notes on equational theories of relations.
  algebra universalis
- Ésik, Z. and Bernátsky, L. (1995). Equational properties of kleene algebras of relations with conversion.
  Theoretical Computer Science

Expressions régulières avec converse

- Bloom, S. L., Ésik, Z., and Stefanescu, G. (1995). Notes on equational theories of relations.
  algebra universalis
- Ésik, Z. and Bernátsky, L. (1995). Equational properties of kleene algebras of relations with conversion.
  Theoretical Computer Science

On ajoute à notre grammaire une opération unaire $^\vee$.

$\mathrm{Reg}_X^\vee$

$$e, f ::= \mathbb{0} | \mathbb{1} | x \in X | e + f | e \cdot f | e^* | \ e^\vee$$

Converse

Converse sur les langages : miroir

$$\begin{aligned} \epsilon^\vee &:= \epsilon \\ (w \cdot x)^\vee &:= x \cdot w^\vee \\ L^\vee &:= \{w^\vee \mid w \in L\} \end{aligned}$$

Converse

Converse sur les langages : miroir

$$\begin{aligned} \epsilon^\vee &:= \epsilon \\ (w \cdot x)^\vee &:= x \cdot w^\vee \\ L^\vee &:= \{w^\vee \mid w \in L\} \end{aligned}$$

Converse sur les relations : réciproque

$$xR^\vee y \overset{\Delta}{\Longleftrightarrow} yRx$$

Converse

Converse sur les langages : miroir

$$\begin{aligned} \epsilon^\vee &:= \epsilon \\ (w \cdot x)^\vee &:= x \cdot w^\vee \\ L^\vee &:= \{w^\vee \mid w \in L\} \end{aligned}$$

Converse sur les relations : réciproque

$$xR^\vee y \overset{\Delta}{\Longleftrightarrow} yRx$$

Les question qui se posent maintenant sont les suivantes :

Converse

Converse sur les langages : miroir

$$\begin{aligned} \epsilon^\vee &:= \epsilon \\ (w \cdot x)^\vee &:= x \cdot w^\vee \\ L^\vee &:= \{w^\vee \mid w \in L\} \end{aligned}$$

Converse sur les relations : réciproque

$$xR^\vee y \overset{\Delta}{\Longleftrightarrow} yRx$$

Les question qui se posent maintenant sont les suivantes :

1. Peut-on encoder ces opérations dans la même théorie équationnelle ?

Converse

Converse sur les langages : miroir

$$\begin{aligned} \epsilon^\vee &:= \epsilon \\ (w \cdot x)^\vee &:= x \cdot w^\vee \\ L^\vee &:= \{w^\vee \mid w \in L\} \end{aligned}$$

Converse sur les relations : réciproque

$$xR^\vee y \overset{\Delta}{\Longleftrightarrow} yRx$$

Les question qui se posent maintenant sont les suivantes :

1. Peut-on encoder ces opérations dans la même théorie équationnelle ?
2. Quels axiomes rajouter à KA pour modéliser ces opérations ?

Converse

Converse sur les langages : miroir

$$\begin{aligned} \epsilon^\vee &:= \epsilon \\ (w \cdot x)^\vee &:= x \cdot w^\vee \\ L^\vee &:= \{w^\vee \mid w \in L\} \end{aligned}$$

Converse sur les relations : réciproque

$$xR^\vee y \overset{\Delta}{\Longleftrightarrow} yRx$$

Les question qui se posent maintenant sont les suivantes :

1. Peut-on encoder ces opérations dans la même théorie équationnelle ?
2. Quels axiomes rajouter à KA pour modéliser ces opérations ?
3. Les langages et les relations sont-ils des modèles initiaux pour les théories obtenues ?

Converse

Converse sur les langages : miroir

$$\begin{aligned} \epsilon^\vee &:= \epsilon \\ (w \cdot x)^\vee &:= x \cdot w^\vee \\ L^\vee &:= \{w^\vee \mid w \in L\} \end{aligned}$$

Converse sur les relations : réciproque

$$xR^\vee y \overset{\Delta}{\Longleftrightarrow} yRx$$

Les question qui se posent maintenant sont les suivantes :

1. Peut-on encoder ces opérations dans la même théorie équationnelle ?
2. Quels axiomes rajouter à KA pour modéliser ces opérations ?
3. Les langages et les relations sont-ils des modèles initiaux pour les théories obtenues ?
4. Comment décider ces théories ?

Première question

Peut-on encoder ces opérations dans la même théorie équationnelle ?

Première question

Peut-on encoder ces opérations dans la même théorie équationnelle ?

NON

Première question

Peut-on encoder ces opérations dans la même théorie équationnelle ?

NON

$$a \leqslant a \cdot a^{\vee} \cdot a$$

Première question

Peut-on encoder ces opérations dans la même théorie équationnelle ?

NON

$$a \leqslant a \cdot a^{\vee} \cdot a$$

- $xRy \Rightarrow (xRy \wedge yR^{\vee}x \wedge xRy) \Rightarrow xR \circ R^{\vee} \circ Ry$

$$x \xrightarrow{R} y \;\Rightarrow\; x \underset{R^{\vee}}{\overset{R}{\rightleftarrows}} y \quad \text{c'est à dire} \quad x \xrightarrow{R} y \xrightarrow{R^{\vee}} x \xrightarrow{R} y$$

- si on considère $L = \{a\}$, alors $L \cdot L^{\vee} \cdot L = \{aaa\}$ et $a \notin \{aaa\}$.

Modèles de langages : CKA_1

Théorème

Une axiomatisation complète de la variété $Lang^\vee$ des langages générés par les opérations de concaténation, union, étoile et converse consiste en les axiomes de KA auxquels on ajoute :

$$(a+b)^\vee = a^\vee + b^\vee \tag{1}$$
$$(a \cdot b)^\vee = b^\vee \cdot a^\vee \tag{2}$$
$$(a^*)^\vee = (a^\vee)^* \tag{3}$$
$$a^{\vee\vee} = a \tag{4}$$

On appelle CKA_1 la théorie obtenue.

Soient $e, f \in \mathrm{Reg}_X^\vee$, ce théorème nous dit donc que

$$\mathrm{CKA}_1 \vdash e = f \Leftrightarrow e \equiv_{Lang^\vee} f.$$

Décidabilité de CKA_1

Soit X un alphabet fini,

- $X' := \{x' \mid x \in X\}$ est une copie primée disjointe de X,
- et $\mathbf{X} := X \cup X'$.

Décidabilité de CKA_1

Soit X un alphabet fini,

- $X' := \{x' \mid x \in X\}$ est une copie primée disjointe de X,
- et $\mathbf{X} := X \cup X'$.

1. On commence par voir les équations (1)-(4) comme des règles de réécriture :

$$\begin{aligned}(a+b)^\vee &\longmapsto a^\vee + b^\vee \\ (a\cdot b)^\vee &\longmapsto b^\vee \cdot a^\vee \\ (a^*)^\vee &\longmapsto (a^\vee)^* \\ a^{\vee\vee} &\longmapsto a\end{aligned}$$

On applique ces règles jusqu'à avoir fait descendre tous les $^\vee$ sur les variables.

Décidabilité de CKA_1

Soit X un alphabet fini,

- $X' := \{x' \mid x \in X\}$ est une copie primée disjointe de X,
- et $\mathbf{X} := X \cup X'$.

1. On commence par voir les équations (1)-(4) comme des règles de réécriture :

$$\begin{aligned}(a+b)^\vee &\longmapsto a^\vee + b^\vee \\ (a\cdot b)^\vee &\longmapsto b^\vee \cdot a^\vee \\ (a^*)^\vee &\longmapsto (a^\vee)^* \\ a^{\vee\vee} &\longmapsto a\end{aligned}$$

   On applique ces règles jusqu'à avoir fait descendre tous les $^\vee$ sur les variables.
2. On applique alors la substitution $[x^\vee \mapsto x']$. L'expression obtenue est $\mathbf{e} \in \mathrm{Reg}_{\mathbf{X}}$.

Décidabilité de CKA_1

Soit X un alphabet fini,

- $X' := \{x' \mid x \in X\}$ est une copie primée disjointe de X,
- et $\mathbf{X} := X \cup X'$.

1. On commence par voir les équations (1)-(4) comme des règles de réécriture :

$$\begin{aligned}(a+b)^\vee &\longmapsto a^\vee + b^\vee \\ (a\cdot b)^\vee &\longmapsto b^\vee \cdot a^\vee \\ (a^*)^\vee &\longmapsto (a^\vee)^* \\ a^{\vee\vee} &\longmapsto a\end{aligned}$$

   On applique ces règles jusqu'à avoir fait descendre tous les $^\vee$ sur les variables.
2. On applique alors la substitution $[x^\vee \mapsto x']$. L'expression obtenue est $\mathbf{e} \in \mathrm{Reg}_{\mathbf{X}}$.

Théorème

$$\mathrm{CKA}_1 \vdash e = f \Leftrightarrow [\![\mathbf{e}]\!] = [\![\mathbf{f}]\!]$$

Modèles de relations : CKA_2

$$a \leqslant aa^\vee a \tag{5}$$

Modèles de relations : CKA_2

$$a \leqslant aa^\vee a \tag{5}$$

Théorème

Une axiomatisation complète de la variété $Rel^\vee$ des relations générées par les opérations de composition, union, étoile et converse consiste en les axiomes de CKA_1 , auxquels on ajoute l'inéquation (5).

On a donc une équivalence pour toutes expressions $e, f \in \mathrm{Reg}_X^\vee$:

$$\mathrm{CKA}_2 \vdash e = f \Leftrightarrow e \equiv_{Rel^\vee} f$$

Décidabilité de CKA_2 : réduction à KA

Théorème

$$\mathrm{CKA}_2 \vdash e = f \Leftrightarrow \mathfrak{cl}([\![\mathbf{e}]\!]) = \mathfrak{cl}([\![\mathbf{f}]\!])$$

Décidabilité de CKA_2 : réduction à KA

Théorème

$$\mathrm{CKA}_2 \vdash e = f \Leftrightarrow \mathfrak{cl}(\llbracket \mathbf{e} \rrbracket) = \mathfrak{cl}(\llbracket \mathbf{f} \rrbracket)$$

$$\mathrm{CKA}_1 \vdash e = f \Longleftrightarrow e \equiv_{Lang^\vee} f$$

$$\mathrm{CKA}_2 \vdash e = f \Longleftrightarrow e \equiv_{Rel^\vee} f$$

Décidabilité de CKA_2 : réduction à KA

Théorème

$$\mathrm{CKA}_2 \vdash e = f \Leftrightarrow \mathfrak{cl}(\llbracket \mathbf{e} \rrbracket) = \mathfrak{cl}(\llbracket \mathbf{f} \rrbracket)$$

$$\begin{array}{c} \llbracket \mathbf{e} \rrbracket = \llbracket \mathbf{f} \rrbracket \\ \Updownarrow \\ \mathrm{CKA}_1 \vdash e = f \Longleftrightarrow e \equiv_{Lang^\vee} f \end{array}$$

$$\begin{array}{c} \mathrm{CKA}_2 \vdash e = f \Longleftrightarrow e \equiv_{Rel^\vee} f \\ \Updownarrow \\ \mathfrak{cl}(\llbracket \mathbf{e} \rrbracket) = \mathfrak{cl}(\llbracket \mathbf{f} \rrbracket) \end{array}$$

Décidabilité de CKA_2 : réduction à KA

Théorème

$$\mathrm{CKA}_2 \vdash e = f \Leftrightarrow \mathfrak{cl}(\llbracket \mathbf{e} \rrbracket) = \mathfrak{cl}(\llbracket \mathbf{f} \rrbracket)$$

$$\begin{array}{c} \llbracket \mathbf{e} \rrbracket = \llbracket \mathbf{f} \rrbracket \\ \Updownarrow \\ \mathrm{CKA}_1 \vdash e = f \Longleftrightarrow e \equiv_{Lang^\vee} f \end{array}$$

$$\begin{array}{c} \mathrm{CKA}_2 \vdash e = f \Longleftrightarrow e \equiv_{Rel^\vee} f \\ \Updownarrow \\ \mathfrak{cl}(\llbracket \mathbf{e} \rrbracket) = \mathfrak{cl}(\llbracket \mathbf{f} \rrbracket) \end{array}$$

Réduction et clôture

Définition : $\overline{w}$

Pour tout mot $w \in \mathbf{X}^*$, on définit inductivement $\overline{w}$:

$$\begin{array}{ll|l} \forall x \in X, & \overline{x} := x' & \overline{\epsilon} := \epsilon \\ \forall x' \in X', & \overline{x'} := x & \overline{wx} := \overline{x}\ \overline{w} \end{array}$$

Réduction et clôture

Définition : $\overline{w}$

Pour tout mot $w \in \mathbf{X}^*$, on définit inductivement $\overline{w}$:

$$\begin{array}{ll|l} \forall x \in X, & \overline{x} := x' & \overline{\epsilon} := \epsilon \\ \forall x' \in X', & \overline{x'} := x & wx := \overline{x}\ \overline{w} \end{array}$$

Définition : $u \rightsquigarrow v$

$$\frac{}{u_1 \cdot w\overline{w}w \cdot u_2 \quad \rightsquigarrow \quad u_1 \cdot w \cdot u_2}$$

Exemple :

$abbabb'a'abbaa'$

Réduction et clôture

Définition : $\overline{w}$

Pour tout mot $w \in \mathbf{X}^*$, on définit inductivement $\overline{w}$:

$$\begin{array}{ll|l} \forall x \in X, & \overline{x} := x' & \overline{\epsilon} := \epsilon \\ \forall x' \in X', & \overline{x'} := x & \overline{wx} := \overline{x}\ \overline{w} \end{array}$$

Définition : $u \rightsquigarrow v$

$$\frac{}{u_1 \cdot w\overline{w}w \cdot u_2 \quad \rightsquigarrow \quad u_1 \cdot w \cdot u_2}$$

Exemple :

$abbabb'a'abbaa' = abb \cdot ab \cdot b'a' \cdot ab \cdot baa'$

Réduction et clôture

Définition : $\overline{w}$

Pour tout mot $w \in \mathbf{X}^*$, on définit inductivement $\overline{w}$:

$$\begin{array}{ll|l} \forall x \in X, & \overline{x} := x' & \overline{\epsilon} := \epsilon \\ \forall x' \in X', & \overline{x'} := x & \overline{wx} := \overline{x}\ \overline{w} \end{array}$$

Définition : $u \rightsquigarrow v$

$$\frac{}{u_1 \cdot w\overline{w}w \cdot u_2 \quad \rightsquigarrow \quad u_1 \cdot w \cdot u_2}$$

Exemple :

$abbabb'a'abbaa' = abb \cdot ab \cdot b'a' \cdot ab \cdot baa' = abb \cdot ab \cdot \overline{ab} \cdot ab \cdot baa'$

Réduction et clôture

Définition : $\overline{w}$

Pour tout mot $w \in \mathbf{X}^*$, on définit inductivement $\overline{w}$:

$$\begin{array}{ll|l} \forall x \in X, & \overline{x} := x' & \overline{\epsilon} := \epsilon \\ \forall x' \in X', & \overline{x'} := x & \overline{wx} := \overline{x}\ \overline{w} \end{array}$$

Définition : $u \rightsquigarrow v$

$$\frac{}{u_1 \cdot w\overline{w}w \cdot u_2 \quad \rightsquigarrow \quad u_1 \cdot w \cdot u_2}$$

Exemple :

$abbabb'a'abbaa' = abb \cdot ab \cdot b'a' \cdot ab \cdot baa' = abb \cdot ab \cdot \overline{ab} \cdot ab \cdot baa' \rightsquigarrow abb \cdot ab \cdot baa'$

Réduction et clôture

Définition : $\overline{w}$

Pour tout mot $w \in \mathbf{X}^*$, on définit inductivement $\overline{w}$:

$$\begin{array}{ll|l} \forall x \in X, & \overline{x} := x' & \overline{\epsilon} := \epsilon \\ \forall x' \in X', & \overline{x'} := x & \overline{wx} := \overline{x}\ \overline{w} \end{array}$$

Définition : $u \rightsquigarrow v$

$$\frac{}{u_1 \cdot w\overline{w}w \cdot u_2 \quad \rightsquigarrow \quad u_1 \cdot w \cdot u_2}$$

Exemple :

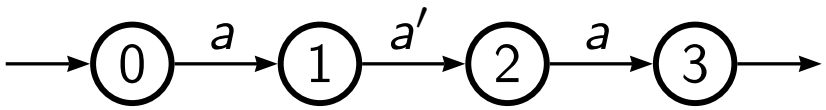
$$abbabb'a'abbaa' = abb \cdot ab \cdot b'a' \cdot ab \cdot baa' = abb \cdot ab \cdot \overline{ab} \cdot ab \cdot baa' \rightsquigarrow abb \cdot ab \cdot baa'$$

Définition : $\mathfrak{cl}(L)$

$$\mathfrak{cl}(L) := \{v \mid \exists u \in L : u \rightsquigarrow^* v\}$$

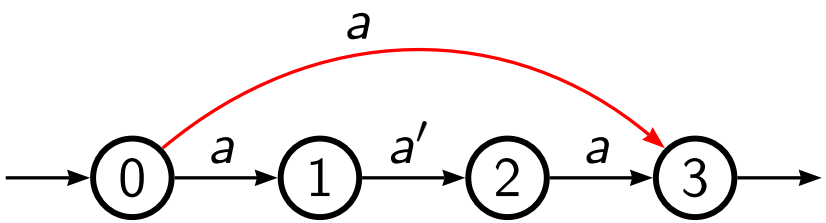
Calcul de la clôture d'un langage

$$\mathscr{A} \text{ reconnaissant } L \xmapsto{\mathfrak{cl}(.)} \mathscr{A}' \text{ reconnaissant } \mathfrak{cl}(L)$$



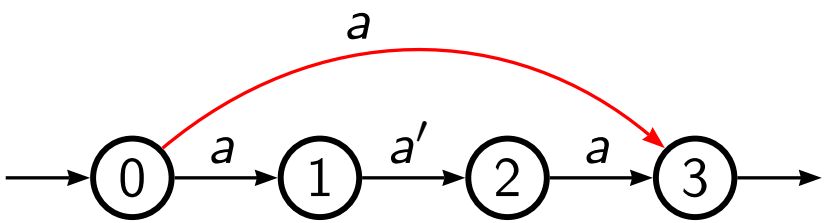
Calcul de la clôture d'un langage

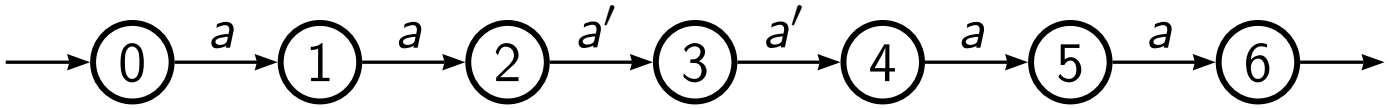
$\mathscr{A}$ reconnaissant $L \xmapsto{\mathfrak{cl}(.)} \mathscr{A}'$ reconnaissant $\mathfrak{cl}(L)$



Calcul de la clôture d'un langage

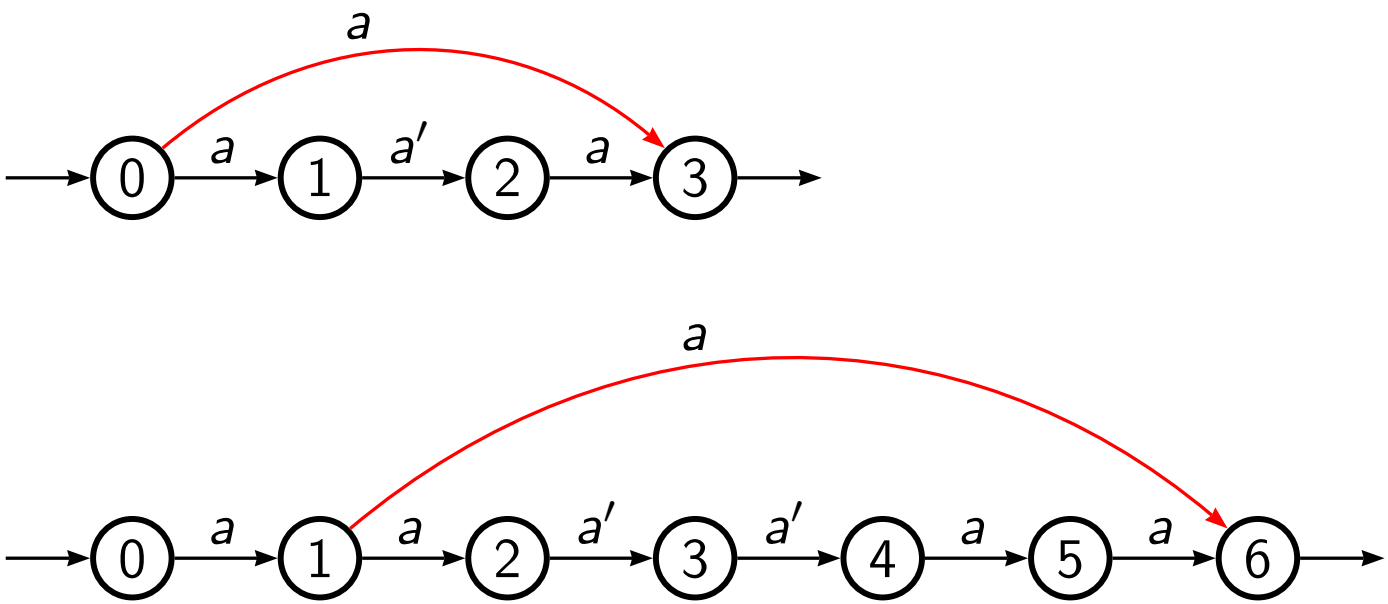
$\mathscr{A}$ reconnaissant $L \xmapsto{\mathfrak{cl}(.)} \mathscr{A}'$ reconnaissant $\mathfrak{cl}(L)$





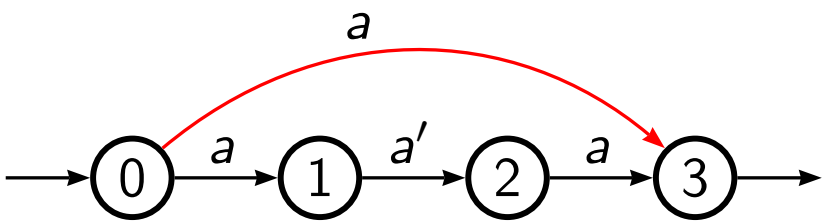
Calcul de la clôture d'un langage

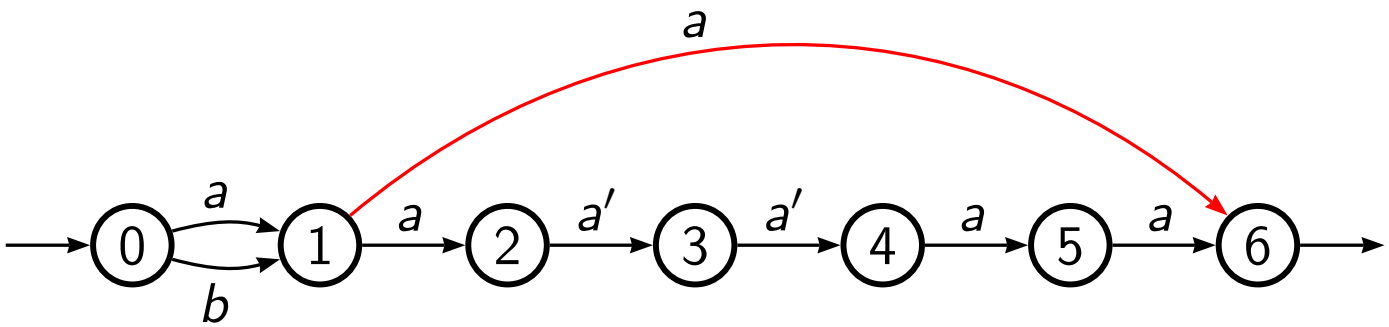
$\mathscr{A}$ reconnaissant $L \xmapsto{\mathfrak{cl}(.)} \mathscr{A}'$ reconnaissant $\mathfrak{cl}(L)$



Calcul de la clôture d'un langage

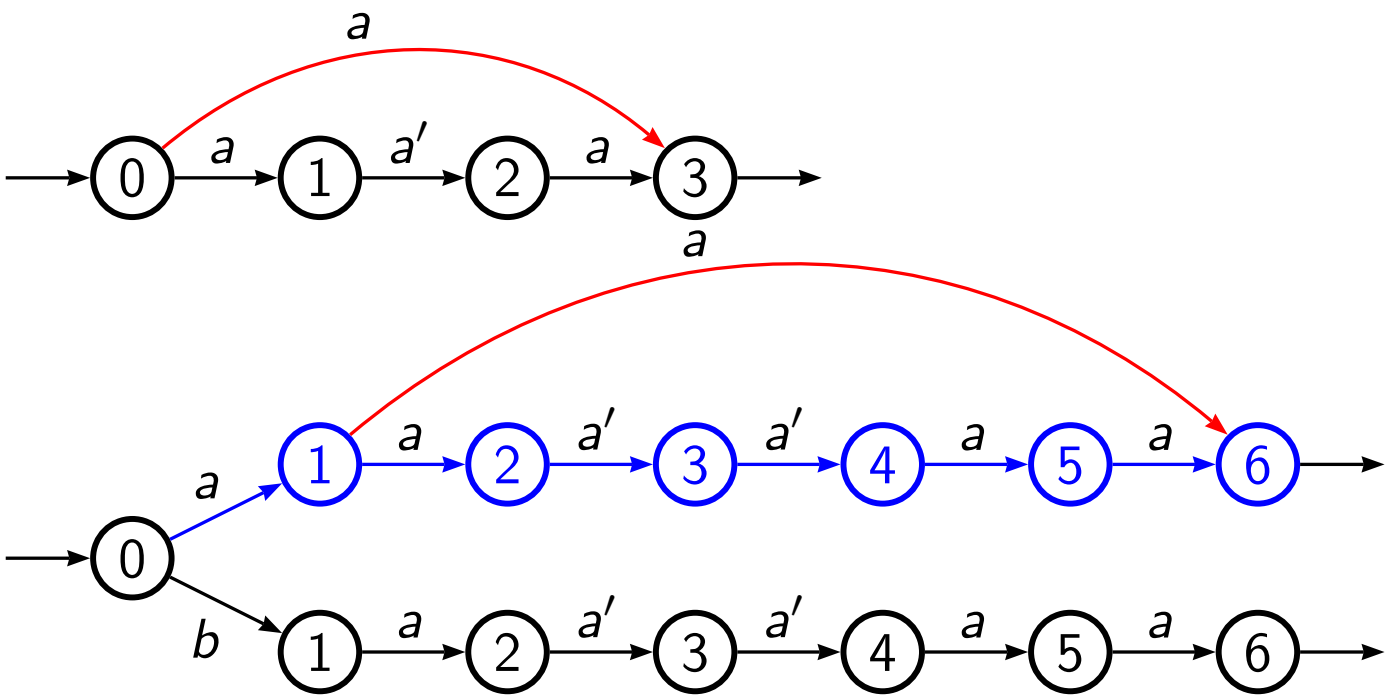
$\mathscr{A}$ reconnaissant $L \xmapsto{\mathfrak{cl}(.)} \mathscr{A}'$ reconnaissant $\mathfrak{cl}(L)$





Calcul de la clôture d'un langage

$\mathscr{A}$ reconnaissant $L \xmapsto{\mathfrak{cl}(.)} \mathscr{A}'$ reconnaissant $\mathfrak{cl}(L)$



Et il y a encore pire...

Idée générale

- Bloom, Ésik et Stefanescu proposent une construction, utilisant le monoïde des transitions de l'automate de départ, construisant un automate déterministe de taille bornée par $2^{2^{n \times 2^n + 1}}$ (avec n la taille de l'automate initial).

Idée générale

- Bloom, Ésik et Stefanescu proposent une construction, utilisant le monoïde des transitions de l'automate de départ, construisant un automate déterministe de taille bornée par $2^{2^{n \times 2^n + 1}}$ (avec n la taille de l'automate initial).
- Nous allons proposer une autre contruction, plus légère. Les états de notre automate seront constitués d'un couple formé
  - d'un état de l'automate initial
  - et d'un historique, permettant d'effectuer des sauts.

Idée générale

- Bloom, Ésik et Stefanescu proposent une construction, utilisant le monoïde des transitions de l'automate de départ, construisant un automate déterministe de taille bornée par $2^{2^{n \times 2^n + 1}}$ (avec n la taille de l'automate initial).
- Nous allons proposer une autre contruction, plus légère. Les états de notre automate seront constitués d'un couple formé
  - d'un état de l'automate initial
  - et d'un historique, permettant d'effectuer des sauts.

$$\text{Si} : q_0 \xrightarrow{u} q_1 \xrightarrow{x} q_3 \xrightarrow{w} q_2$$

$$\text{Avec} : \exists u_2 \in \text{suffixes}(ux) : w \rightsquigarrow^* \overline{u}_2 u_2$$

Idée générale

- Bloom, Ésik et Stefanescu proposent une construction, utilisant le monoïde des transitions de l'automate de départ, construisant un automate déterministe de taille bornée par $2^{2^{n \times 2^n + 1}}$ (avec n la taille de l'automate initial).
- Nous allons proposer une autre contruction, plus légère. Les états de notre automate seront constitués d'un couple formé
  - d'un état de l'automate initial
  - et d'un historique, permettant d'effectuer des sauts.

$$\text{Si} : q_0 \xrightarrow{u} q_1 \xrightarrow{x} q_3 \xrightarrow{w} q_2$$

$$\text{Avec} : \exists u_2 \in \text{suffixes}(ux) : w \rightsquigarrow^* \overline{u}_2 u_2$$

$$\text{Soit} : uxw \rightsquigarrow^* ux\overline{u}_2 u_2 = u_1 u_2 \overline{u}_2 u_2 \rightsquigarrow u_1 u_2 = ux$$

Idée générale

- Bloom, Ésik et Stefanescu proposent une construction, utilisant le monoïde des transitions de l'automate de départ, construisant un automate déterministe de taille bornée par $2^{2^{n \times 2^n + 1}}$ (avec n la taille de l'automate initial).
- Nous allons proposer une autre contruction, plus légère. Les états de notre automate seront constitués d'un couple formé
  - d'un état de l'automate initial
  - et d'un historique, permettant d'effectuer des sauts.

$$\text{Si} : q_0 \xrightarrow{u} q_1 \xrightarrow{x} q_3 \xrightarrow{w} q_2$$

$$\text{Avec} : \exists u_2 \in \text{suffixes}(ux) : w \rightsquigarrow^* \overline{u}_2 u_2$$

$$\text{Soit} : uxw \rightsquigarrow^* ux\overline{u}_2 u_2 = u_1 u_2 \overline{u}_2 u_2 \rightsquigarrow u_1 u_2 = ux$$

$$\text{Alors} : (q_0, [\epsilon]) \xrightarrow{u} (q_1, [u]) \xrightarrow{x} (q_2, [ux])$$

Idée générale

- Bloom, Ésik et Stefanescu proposent une construction, utilisant le monoïde des transitions de l'automate de départ, construisant un automate déterministe de taille bornée par $2^{2^{n \times 2^n + 1}}$ (avec n la taille de l'automate initial).
- Nous allons proposer une autre contruction, plus légère. Les états de notre automate seront constitués d'un couple formé
  - d'un état de l'automate initial
  - et d'un historique, permettant d'effectuer des sauts.

$$\text{Si} : q_0 \xrightarrow{u} q_1 \xrightarrow{x} q_3 \xrightarrow{w} q_2$$

$$\text{Avec} : \exists u_2 \in \text{suffixes}(ux) : w \rightsquigarrow^* \overline{u}_2 u_2$$

$$\text{Soit} : uxw \rightsquigarrow^* ux\overline{u}_2 u_2 = u_1 u_2 \overline{u}_2 u_2 \rightsquigarrow u_1 u_2 = ux$$

$$\text{Alors} : (q_0, [\epsilon]) \xrightarrow{u} (q_1, [u]) \xrightarrow{x} (q_2, [ux])$$

$\Gamma(w)$

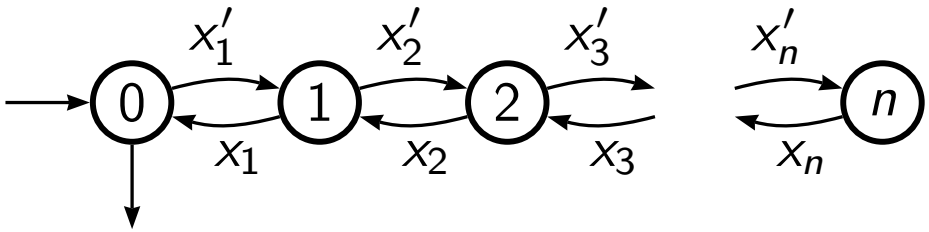
Définition : $\Gamma(w)$

$$\Gamma(\epsilon) = \{\epsilon\}$$
$$\Gamma(wx) = (\{\overline{x}\} \cdot \Gamma(w) \cdot \{x\})^*$$

Lemme

$$u \in \Gamma(w) \Leftrightarrow \exists v \in \text{suffixes}(w) : u \rightsquigarrow^* \overline{v}v$$

$\Gamma(x_n \cdots x_1)$ est reconnu par l'automate :



$\gamma(w)$

On considère un automate $\mathscr{A} = \langle Q, A, I, T, \Delta \rangle$, et on note $\Delta_x := \{(p, q) \mid p \xrightarrow{x} q \in \Delta\}$.

Définition : $\gamma(w)$

$$\gamma(\epsilon) = \mathrm{Id}_Q$$
$$\gamma(wx) = (\Delta_{\overline{x}} \cdot \gamma(w) \cdot \Delta_x)^*$$

Lemme

$$(p, q) \in \gamma(w) \Leftrightarrow \exists u \in \Gamma(w) : p \xrightarrow{u} q$$
$$\Leftrightarrow \exists u : \exists v \in \mathrm{suffixes}(w) : p \xrightarrow{u} q \wedge u \rightsquigarrow^* \overline{v} v$$

Automate clôture

Définition : G

$$[u] \coloneqq \{v \in \mathbf{X}^* \mid \gamma(v) = \gamma(u)\}$$

G est l'ensemble des classes d'équivalence de γ : $G \coloneqq \{[u] \mid u \in \mathbf{X}^*\}$

Automate clôture

Définition : G

$$[u] := \{v \in \mathbf{X}^* \mid \gamma(v) = \gamma(u)\}$$

G est l'ensemble des classes d'équivalence de γ : $G := \{[u] \mid u \in \mathbf{X}^*\}$

Théorème

La clôture du langage L est reconnue par l'automate $\mathscr{A}' := \langle Q \times G, \mathbf{X}, I \times \{[\epsilon]\}, F \times G, \Delta' \rangle$ avec :

$$\Delta' = \{((q_1, [w]), x, (q_2, [wx])) \mid (q_1, q_2) \in \Delta_x \circ \gamma(wx)\}$$

$$(p, [u]) \xrightarrow{x} (q, [ux]) \overset{\Delta}{\iff} \begin{array}{l} \exists r \in Q \\ \exists w \in \Gamma(ux) \end{array} : p \xrightarrow{x} r \xrightarrow{w} q$$

Automate clôture

Définition : G

$$[u] := \{v \in \mathbf{X}^* \mid \gamma(v) = \gamma(u)\}$$

G est l'ensemble des classes d'équivalence de γ : $G := \{[u] \mid u \in \mathbf{X}^*\}$

Théorème

La clôture du langage L est reconnue par l'automate
$\mathscr{A}' := \langle Q \times G, \mathbf{X}, I \times \{[\epsilon]\}, F \times G, \Delta' \rangle$ avec :

$$\Delta' = \{((q_1, [w]), x, (q_2, [wx])) \mid (q_1, q_2) \in \Delta_x \circ \gamma(wx)\}$$

$$(p, [u]) \xrightarrow{x} (q, [ux]) \overset{\Delta}{\iff} \begin{array}{l} \exists r \in Q \\ \exists v \in \text{suffixes}(ux) \\ \exists w \rightsquigarrow^* \overline{v}v \end{array} : p \xrightarrow{x} r \xrightarrow{w} q$$

Complexité et comparaison avec la construction originale

$$\Delta' = \{((q_1, [w]), x, (q_2, [wx])) \mid (q_1, q_2) \in \Delta_x \circ \gamma(wx)\}$$

On voit que notre construction donne un automate non déterministe de taille au plus $n \times 2^{n \times (n-1)}$.

Complexité et comparaison avec la construction originale

$$\Delta' = \{((q_1, [w]), x, (q_2, [wx])) \mid (q_1, q_2) \in \Delta_x \circ \gamma(wx)\}$$

On voit que notre construction donne un automate non déterministe de taille au plus $n \times 2^{n \times (n-1)}$.

De plus, on peut simplement le déterminiser :

$$\delta' : ((Q_1, [w]), x) \mapsto (\{q_2 \mid \exists q_1 \in Q_1 : (q_1, q_2) \in \Delta_x \circ \gamma(wx)\}, [wx])$$

Complexité et comparaison avec la construction originale

$$\Delta' = \{((q_1, [w]), x, (q_2, [wx])) \mid (q_1, q_2) \in \Delta_x \circ \gamma(wx)\}$$

On voit que notre construction donne un automate non déterministe de taille au plus $n \times 2^{n \times (n-1)}$.

De plus, on peut simplement le déterminiser :

$$\delta' : ((Q_1, [w]), x) \mapsto (\{q_2 \mid \exists q_1 \in Q_1 : (q_1, q_2) \in \Delta_x \circ \gamma(wx)\}, [wx])$$

Cet automate déterministe est de taille au plus $2^n \times 2^{n \times (n-1)} = 2^{n^2}$, ce qui est bien inférieur au $2^{2^{n \times 2^n + 1}}$ de la construction originale.

Plan

Résumé

$$\mathsf{CKA}_1 \vdash e = f \iff e \equiv_{Lang^\vee} f$$

$$\mathsf{CKA}_2 \vdash e = f \iff e \equiv_{Rel^\vee} f$$

Résumé

$$\begin{array}{rl} [\![\mathbf{e}]\!] = [\![\mathbf{f}]\!] & \text{(décidable)} \\ \Updownarrow & \\ \mathsf{CKA}_1 \vdash e = f \Longleftrightarrow e \equiv_{Lang^\vee} f & \end{array}$$

$$\begin{array}{rl} \mathsf{CKA}_2 \vdash e = f \Longleftrightarrow e \equiv_{Rel^\vee} f & \\ \Updownarrow & \\ \mathfrak{cl}([\![\mathbf{e}]\!]) = \mathfrak{cl}([\![\mathbf{f}]\!]) & \text{(décidable)} \end{array}$$

Bilan et perspectives

Bilan

1. Re-formulation, plus élémentaire, des preuves.

Bilan et perspectives

Bilan

1. Re-formulation, plus élémentaire, des preuves.
2. Nouvelle construction de l'automate clôture :
   - plus simple
   - plus légère
   - permettant de prouver que ce problème est dans PSPACE.

Bilan et perspectives

Bilan

1. Re-formulation, plus élémentaire, des preuves.
2. Nouvelle construction de l'automate clôture :
   - plus simple
   - plus légère
   - permettant de prouver que ce problème est dans PSPACE.
3. Implémentation en OCAML et tests de cette construction et de la construction originale.

Bilan et perspectives

Bilan

1. Re-formulation, plus élémentaire, des preuves.
2. Nouvelle construction de l'automate clôture :
   - plus simple
   - plus légère
   - permettant de prouver que ce problème est dans PSPACE.
3. Implémentation en OCAML et tests de cette construction et de la construction originale.
4. Preuve en COQ de la confluence de la relation $\rightsquigarrow$.

Bilan et perspectives

Bilan

1. Re-formulation, plus élémentaire, des preuves.
2. Nouvelle construction de l'automate clôture :
   - plus simple
   - plus légère
   - permettant de prouver que ce problème est dans PSPACE.
3. Implémentation en OCAML et tests de cette construction et de la construction originale.
4. Preuve en COQ de la confluence de la relation $\rightsquigarrow$.

Perspectives

1. Intégration à COQ, dans la librairie *Relation Algebra*[a].

a. `perso.ens-lyon.fr/damien.pous/ra/`

Bilan et perspectives

Bilan

1. Re-formulation, plus élémentaire, des preuves.
2. Nouvelle construction de l'automate clôture :
   - plus simple
   - plus légère
   - permettant de prouver que ce problème est dans PSPACE.
3. Implémentation en OCAML et tests de cette construction et de la construction originale.
4. Preuve en COQ de la confluence de la relation $\rightsquigarrow$.

Perspectives

1. Intégration à COQ, dans la librairie *Relation Algebra*[a].
2. Utilisation de l'approche par automates à d'autres extensions des algèbres de Kleene, par exemple les algèbres d'action.

a. `perso.ens-lyon.fr/damien.pous/ra/`

Plan