

# Sur-approximations non régulières et terminaison pour l'analyse d'accessibilité

Vivien Pelletier

Encadré par :  
Pierre Réty et Yohan Boichut

23 octobre 2017



# Sommaire

- 1 Introduction
- 2 Préliminaires
- 3 Sur-approximations d'ensembles de descendants
- 4 Transformation de systèmes de réécriture avec stratégie
- 5 Conclusion

# Sommaire

## 1 Introduction

## 2 Préliminaires

## 3 Sur-approximations d'ensembles de descendants

- Méthode de compléction
- Descendants innermost
- Élimination des clauses copiantes

## 4 Transformation de systèmes de réécriture avec stratégie

## 5 Conclusion

## Système complexe

Un système complexe est un ensemble constitué d'un grand nombre d'entités en interaction qui empêchent l'observateur de prévoir sa rétroaction, son comportement ou évolution par le calcul.

# Systèmes complexes

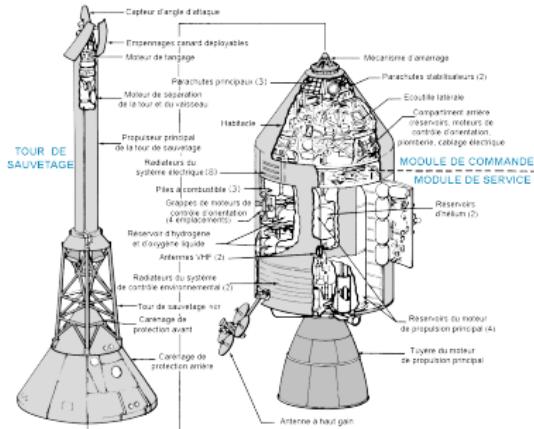
## Système complexe

Un système complexe est un ensemble constitué d'un grand nombre d'entités en interaction qui empêchent l'observateur de prévoir sa rétroaction, son comportement ou évolution par le calcul.

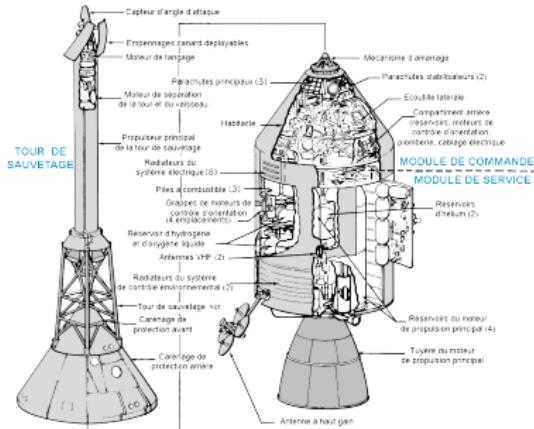
## Systèmes complexes

- programmes informatiques,
- protocoles de sécurité,
- circuits logiques ...

# Le génie logiciel



# Le génie logiciel



## Génie logiciel

L'ensemble des activités de conception et de mise en œuvre des produits et des procédures tendant à rationaliser la production du logiciel et son suivi.



Conformité et fiabilité d'un système complexe ?

Conformité et fiabilité d'un système complexe ?

- Les tests

Conformité et fiabilité d'un système complexe ?

- Les tests
  - si l'ensemble d'entrées est trop grand ou infini ?

Conformité et fiabilité d'un système complexe ?

- Les tests
  - si l'ensemble d'entrées est trop grand ou infini ?
- Les méthodes formelles

## Conformité et fiabilité d'un système complexe ?

- Les tests
  - si l'ensemble d'entrées est trop grand ou infini ?
- Les méthodes formelles
  - analyse statique par interprétation abstraite

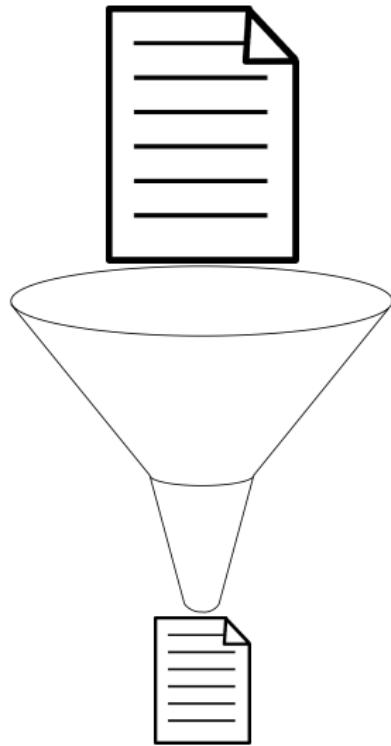
Conformité et fiabilité d'un système complexe ?

- Les tests
  - si l'ensemble d'entrées est trop grand ou infini ?
- Les méthodes formelles
  - analyse statique par interprétation abstraite
  - vérification déductive

Conformité et fiabilité d'un système complexe ?

- Les tests
  - si l'ensemble d'entrées est trop grand ou infini ?
- Les méthodes formelles
  - analyse statique par interprétation abstraite
  - vérification déductive
  - vérification de modèles

# Analyse statique par interprétation abstraite

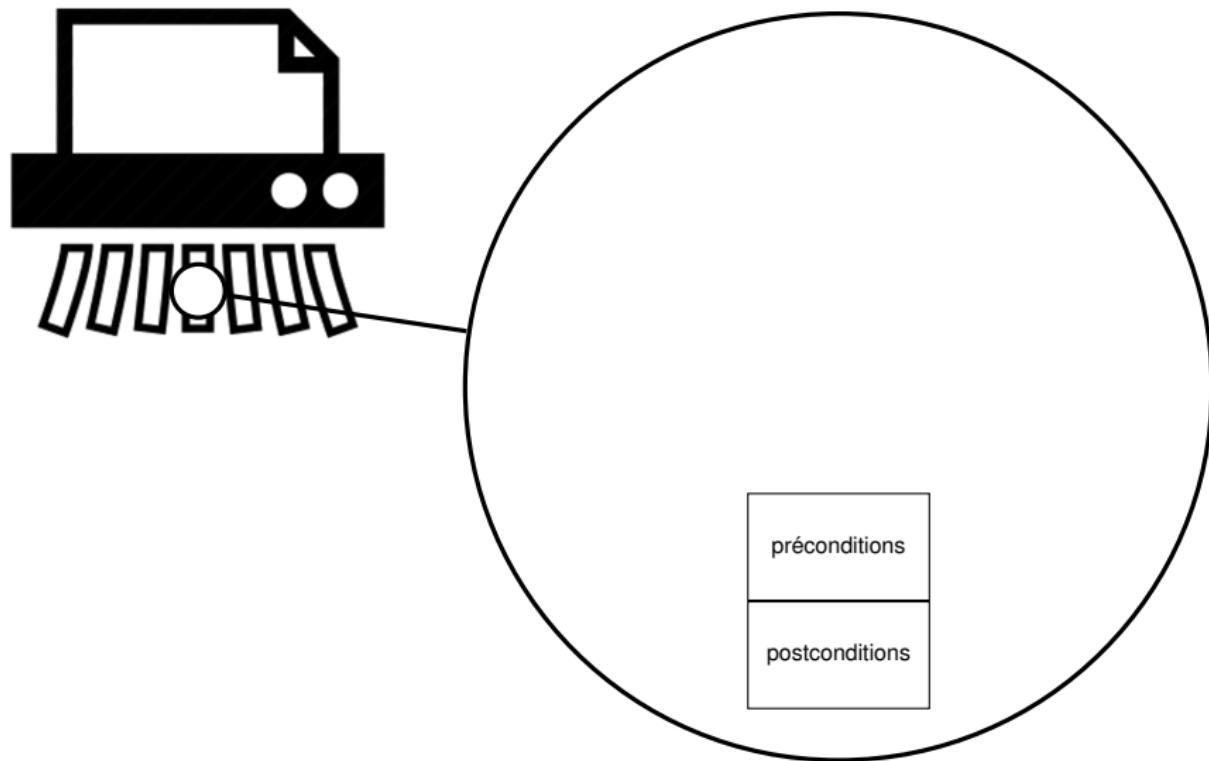


- Constat
  - trop d'informations
- Solution
  - utilisation d'abstractions
- Difficulté
  - garder suffisamment d'informations
  - mais pas trop

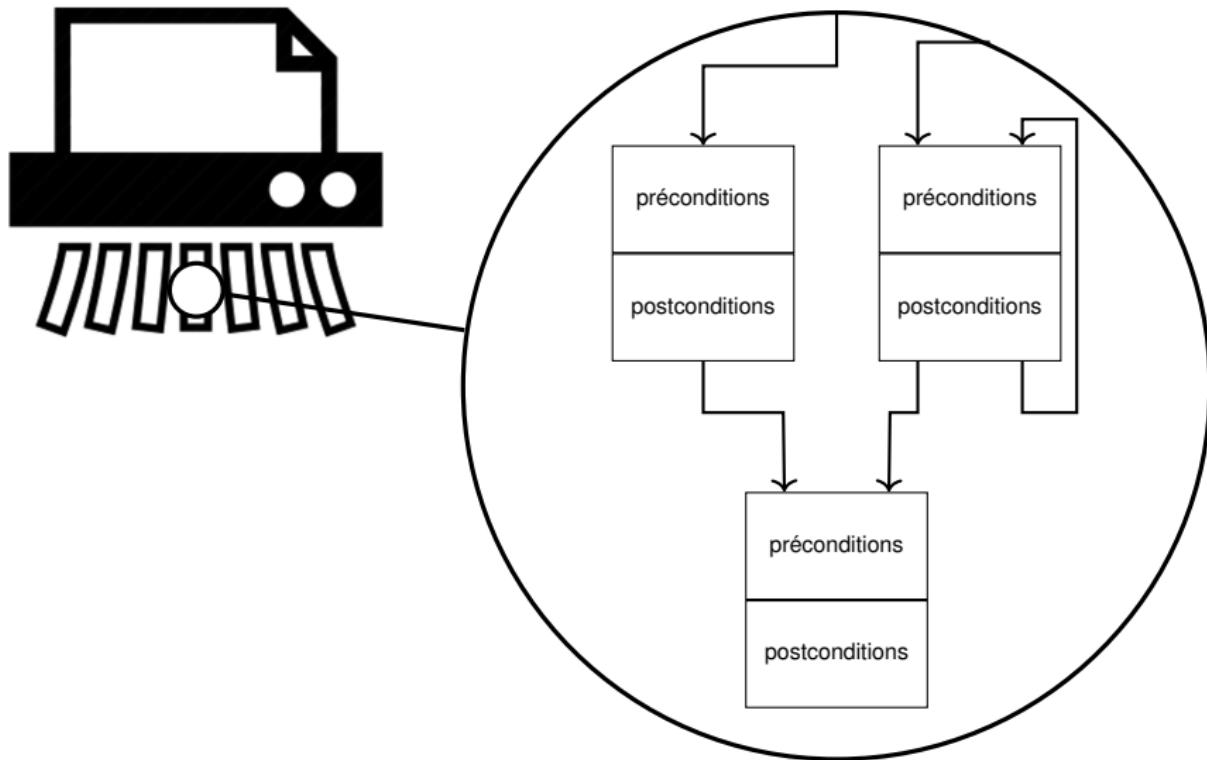
# Vérification déductive



# Vérification déductive

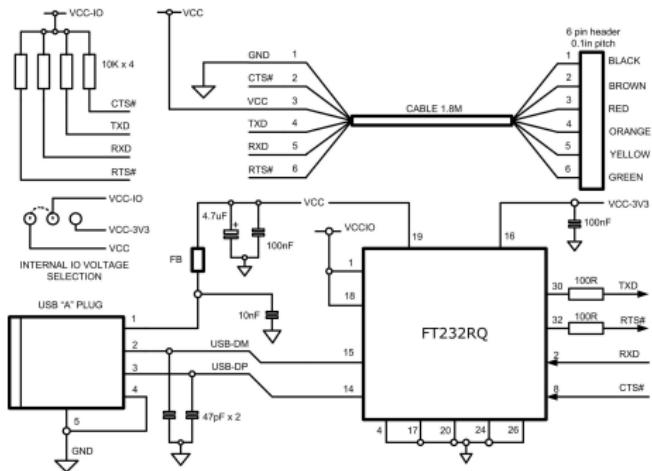
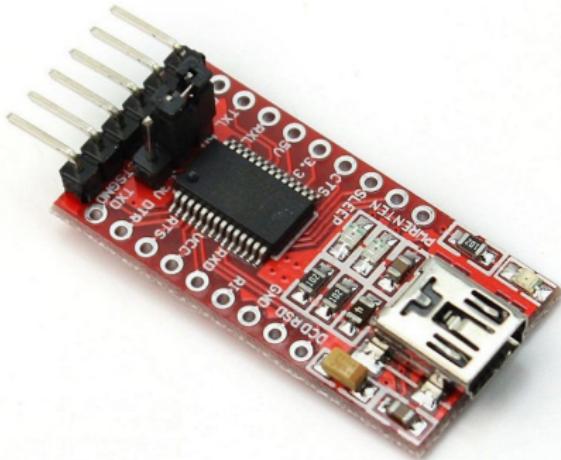


# Vérification déductive



# Vérification de modèles

- Analyse exhaustive
- Représentation astucieuse



## Configurations

- Ensemble des configurations accessibles

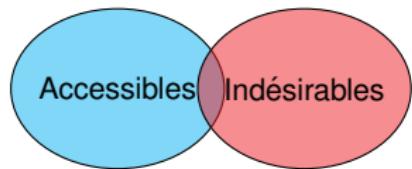
## Configurations

- Ensemble des configurations accessibles
- Ensemble des configurations indésirables

# Analyse d'accessibilité

## Configurations

- Ensemble des configurations accessibles
- Ensemble des configurations indésirables

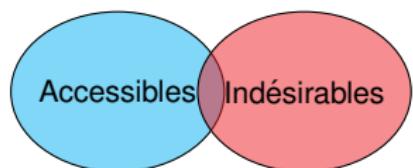


Il existe une configuration  
indésirable accessible

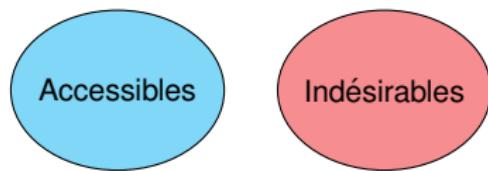
# Analyse d'accessibilité

## Configurations

- Ensemble des configurations accessibles
- Ensemble des configurations indésirables



Il existe une configuration  
indésirable accessible



Aucune configuration  
indésirable  
n'est accessible

# Calcul de l'ensemble des configurations accessibles

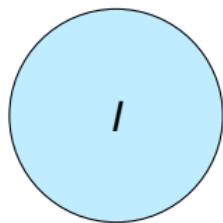
- Configurations initiales : /

# Calcul de l'ensemble des configurations accessibles

- Configurations initiales :  $I$
- Dynamique :  $R$

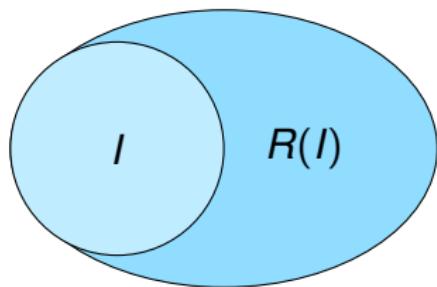
# Calcul de l'ensemble des configurations accessibles

- Configurations initiales :  $I$
- Dynamique :  $R$



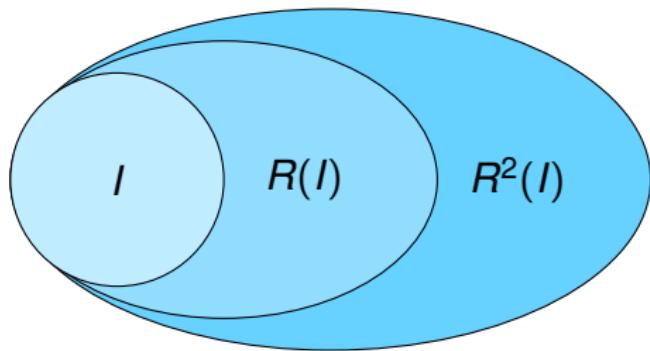
# Calcul de l'ensemble des configurations accessibles

- Configurations initiales :  $I$
- Dynamique :  $R$



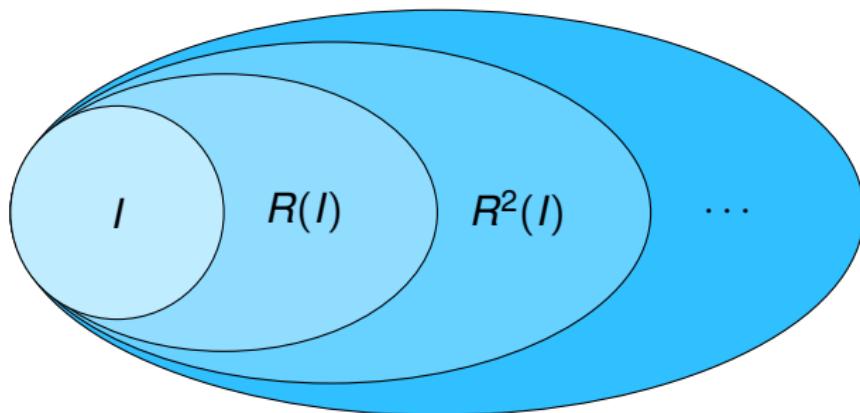
# Calcul de l'ensemble des configurations accessibles

- Configurations initiales :  $I$
- Dynamique :  $R$



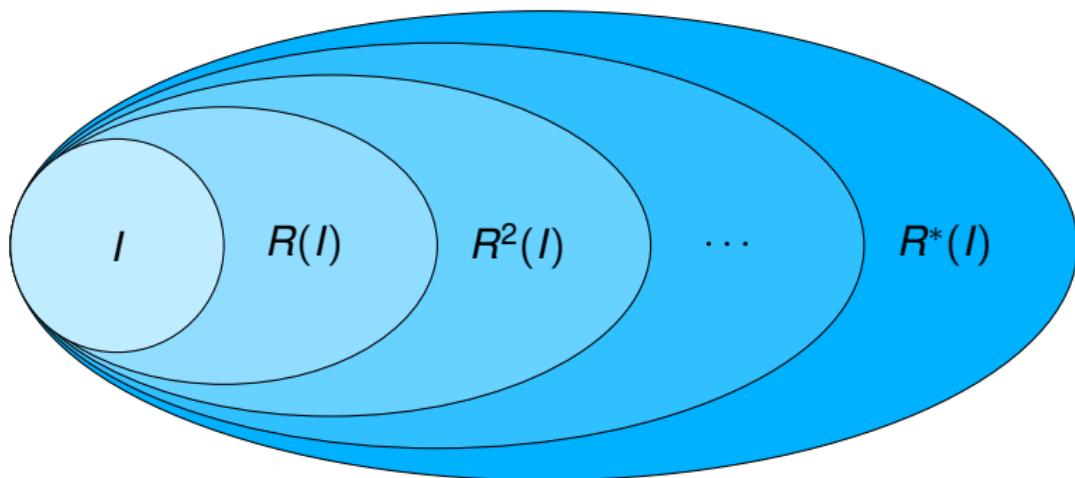
# Calcul de l'ensemble des configurations accessibles

- Configurations initiales :  $I$
- Dynamique :  $R$

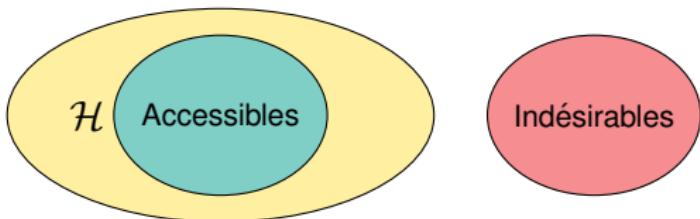


# Calcul de l'ensemble des configurations accessibles

- Configurations initiales :  $I$
- Dynamique :  $R$

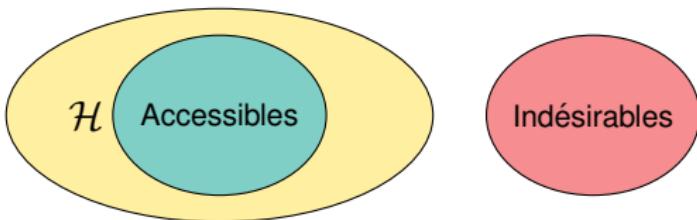


# Sur-approximations

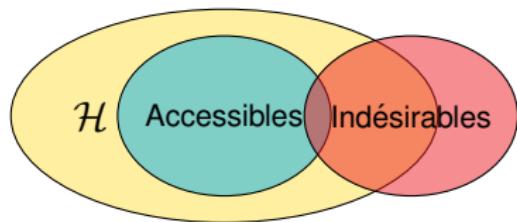


Aucune configuration indésirable accessible

# Sur-approximations

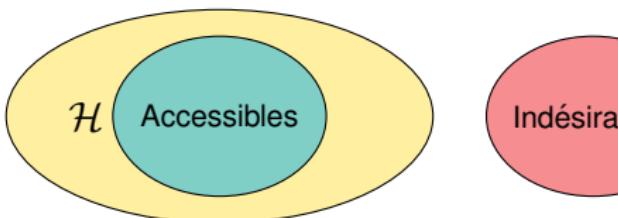


Aucune configuration indésirable accessible

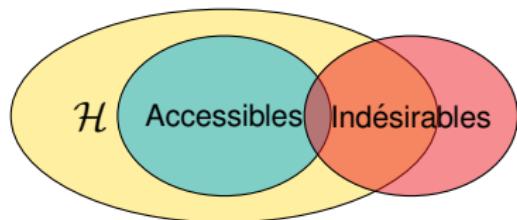


Il existe une configuration  
indésirable accessible

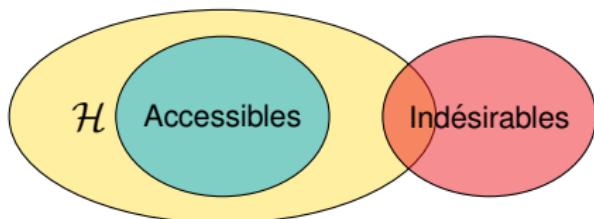
# Sur-approximations



Aucune configuration indésirable accessible



Il existe une configuration  
indésirable accessible



Aucune configuration  
indésirable accessible,  
c'est un faux-positif

# Notre modélisation

- Configuration : un terme

# Notre modélisation

- Configuration : un terme
- Configurations initiales : un langage de termes

# Notre modélisation

- Configuration : un terme
- Configurations initiales : un langage de termes
- Dynamique du système : un système de réécriture

# Notre modélisation

- Configuration : un terme
- Configurations initiales : un langage de termes
- Dynamique du système : un système de réécriture
- Configurations indésirables : un langage de termes

# Sommaire

1 Introduction

2 Préliminaires

3 Sur-approximations d'ensembles de descendants

- Méthode de compléction
- Descendants innermost
- Élimination des clauses copiantes

4 Transformation de systèmes de réécriture avec stratégie

5 Conclusion

# Les termes

- Des symboles d'arité fixe :  $f^{\backslash 2}$

# Les termes

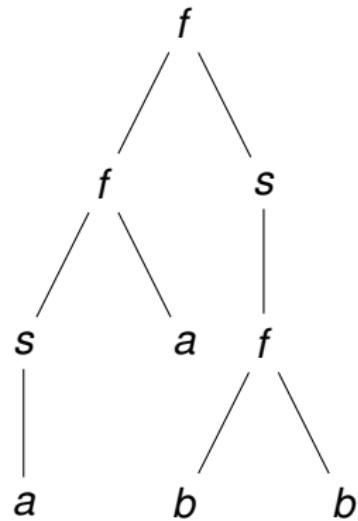
- Des symboles d'arité fixe :  $f^2$
- Un alphabet :  $\{a^0, b^0, s^1, h^1, f^2\}$

# Les termes

- Des symboles d'arité fixe :  $f^2$
- Un alphabet :  $\{a^0, b^0, s^1, h^1, f^2\}$
- Un terme :  $f(f(s(a), a), s(f(b, b)))$

# Les termes

- Des symboles d'arité fixe :  $f^2$
- Un alphabet :  $\{a^0, b^0, s^1, h^1, f^2\}$
- Un terme :  $f(f(s(a), a), s(f(b, b)))$



# Langages formels

Un ensemble de termes

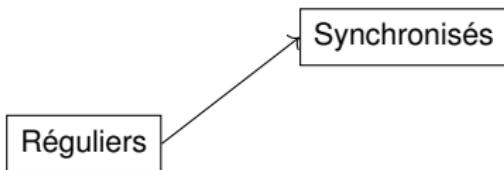
- Langages réguliers :  $\{f(s^*(a), s^*(a))\}$

Réguliers

# Langages formels

Un ensemble de termes

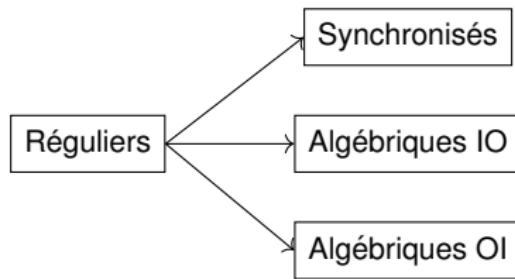
- Langages réguliers :  $\{f(s^*(a), s^*(a))\}$
- Langages synchronisés :  $\{f(s^n(a), s^n(a)) \mid n \geq 0\}$



# Langages formels

Un ensemble de termes

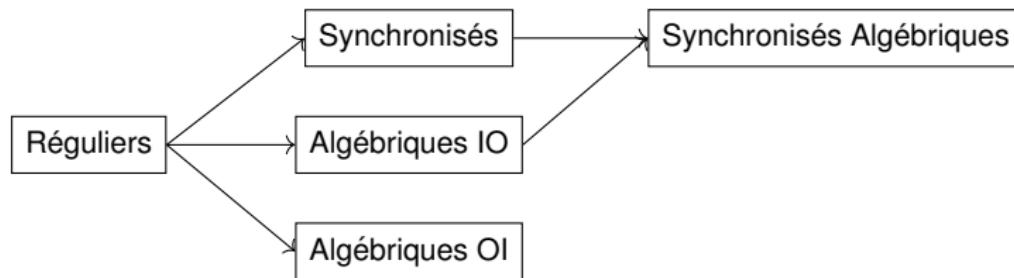
- Langages réguliers :  $\{f(s^*(a), s^*(a))\}$
- Langages synchronisés :  $\{f(s^n(a), s^n(a)) \mid n \geq 0\}$
- Langages algébriques :  $\{s^n(h^n(a)) \mid n \geq 0\}$



# Langages formels

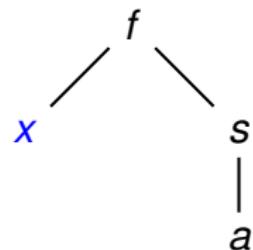
Un ensemble de termes

- Langages réguliers :  $\{f(s^*(a), s^*(a))\}$
- Langages synchronisés :  $\{f(s^n(a), s^n(a)) \mid n \geq 0\}$
- Langages algébriques :  $\{s^n(h^n(a)) \mid n \geq 0\}$
- Langages synchronisés algébriques :  
 $\{f(s^n(h^n(a)), s^n(h^n(b))) \mid n \geq 0\}$



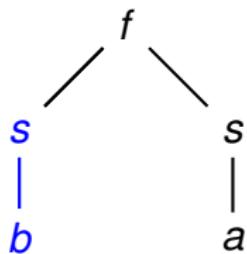
- Ensemble de variables :  $\mathcal{X} = \{x, y\}$

# Variables



- Ensemble de variables :  $\mathcal{X} = \{x, y\}$
- La substitution :  $\sigma = (x/s(b))$

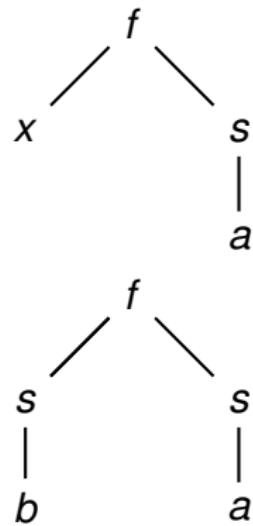
# Variables



- Ensemble de variables :  $\mathcal{X} = \{x, y\}$
- La substitution :  $\sigma = (x/s(b))$

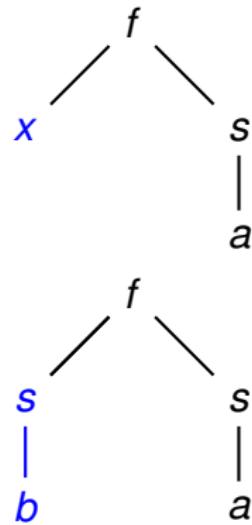
# Variables

- Ensemble de variables :  $\mathcal{X} = \{x, y\}$
- La substitution :  $\sigma = (x/s(b))$
- Le filtrage :  $\sigma(t) = t'$



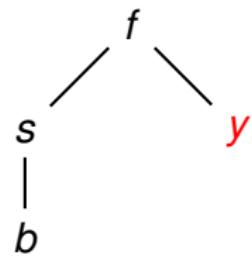
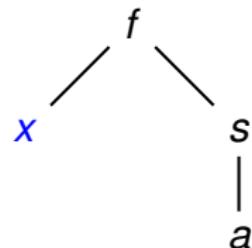
# Variables

- Ensemble de variables :  $\mathcal{X} = \{x, y\}$
- La substitution :  $\sigma = (x/s(b))$
- Le filtrage :  $\sigma(t) = t'$



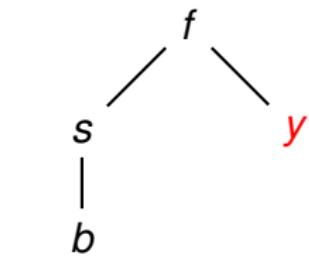
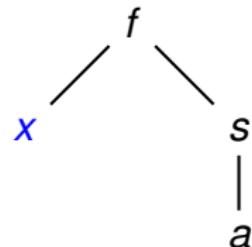
# Variables

- Ensemble de variables :  $\mathcal{X} = \{x, y\}$
- La substitution :  $\sigma = (x/s(b))$
- Le filtrage :  $\sigma(t) = t'$
- L'unification :  $\alpha(t) = \alpha(t')$

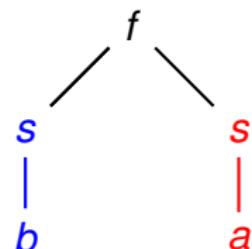


# Variables

- Ensemble de variables :  $\mathcal{X} = \{x, y\}$
- La substitution :  $\sigma = (x/s(b))$
- Le filtrage :  $\sigma(t) = t'$
- L'unification :  $\alpha(t) = \alpha(t')$



$$\alpha = (x/s(b), y/s(a))$$



## Règle de réécriture

- $l$  et  $r$  des termes

## Règle de réécriture

- $l$  et  $r$  des termes
- $r$  n'est pas une variable

## Règle de réécriture

- $l$  et  $r$  des termes
- $r$  n'est pas une variable
- $Var(r) \subseteq Var(l)$

## Règle de réécriture

- $I$  et  $r$  des termes
- $r$  n'est pas une variable
- $\text{Var}(r) \subseteq \text{Var}(I)$
- $I \rightarrow r$  est une règle de réécriture

## Règle de réécriture

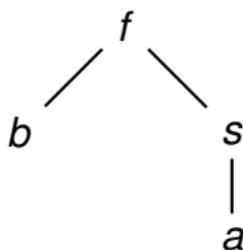
- $l$  et  $r$  des termes
- $r$  n'est pas une variable
- $\text{Var}(r) \subseteq \text{Var}(l)$
- $l \rightarrow r$  est une règle de réécriture

$$s(x) \rightarrow s(s(x))$$

## Règle de réécriture

- $I$  et  $r$  des termes
- $r$  n'est pas une variable
- $\text{Var}(r) \subseteq \text{Var}(I)$
- $I \rightarrow r$  est une règle de réécriture

$$s(x) \rightarrow s(s(x))$$



# Systèmes de réécriture

## Règle de réécriture

- $I$  et  $r$  des termes
- $r$  n'est pas une variable
- $\text{Var}(r) \subseteq \text{Var}(I)$
- $I \rightarrow r$  est une règle de réécriture

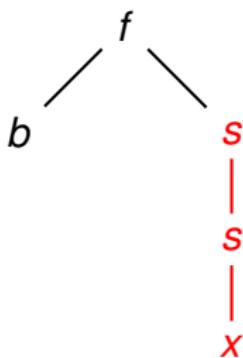
$$\begin{aligned} s(x) &\rightarrow s(s(x)) \\ \sigma &= (x/a) \\ f & \\ / \quad \backslash & \\ b & \quad s \quad s \\ & | \quad | \\ & a \quad x \end{aligned}$$

## Règle de réécriture

- $I$  et  $r$  des termes
- $r$  n'est pas une variable
- $\text{Var}(r) \subseteq \text{Var}(I)$
- $I \rightarrow r$  est une règle de réécriture

$$s(x) \rightarrow s(s(x))$$

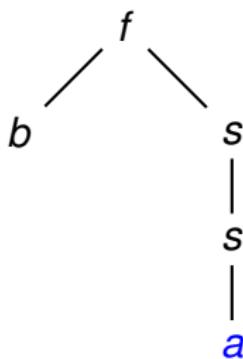
$$\sigma = (x/a)$$



## Règle de réécriture

- $I$  et  $r$  des termes
- $r$  n'est pas une variable
- $\text{Var}(r) \subseteq \text{Var}(I)$
- $I \rightarrow r$  est une règle de réécriture

$$s(x) \rightarrow s(s(x))$$
$$\sigma = (x/a)$$



# Sommaire

1 Introduction

2 Préliminaires

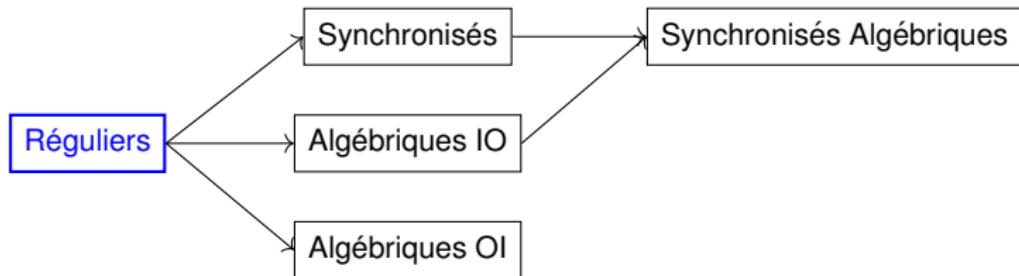
3 Sur-approximations d'ensembles de descendants

- Méthode de compléction
- Descendants innermost
- Élimination des clauses copiantes

4 Transformation de systèmes de réécriture avec stratégie

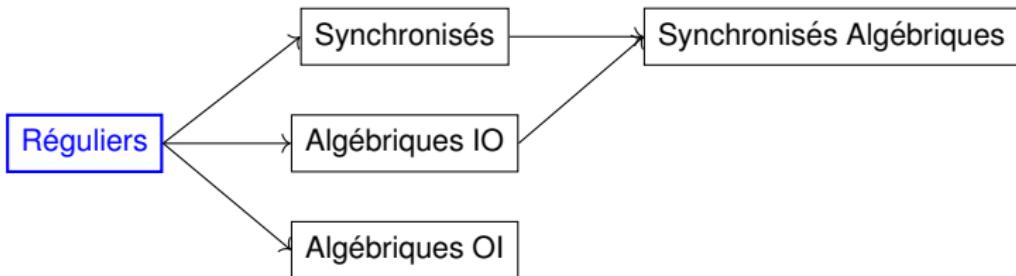
5 Conclusion

# Les différentes techniques



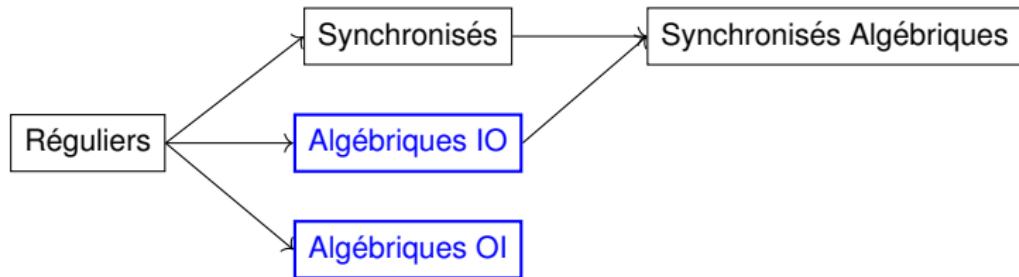
- RTA, 1998, T. Genet, Decidable approximations of sets of descendants and sets of normal forms
- CADE, 2000, T. Genet et F. Klay, Rewriting for cryptographic protocol verification
- JSC, 2010, T. Genet et V. Rusu, Equational approximations for tree automata completion

# Les différentes techniques



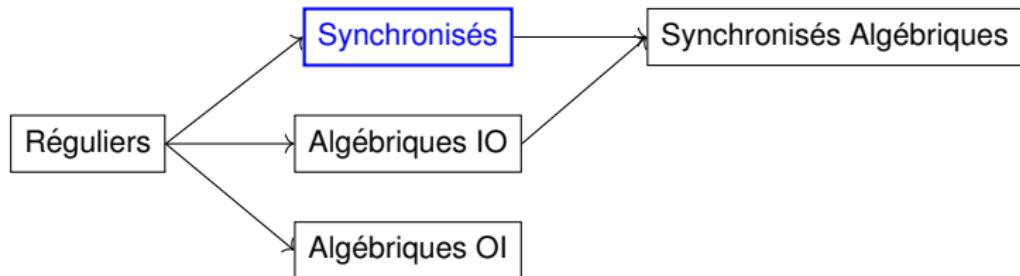
- IPL, 2008, Y. Boichut et P.-C. Héam, A theoretical limit for safety verification techniques with regular fix-point computations

# Les différentes techniques



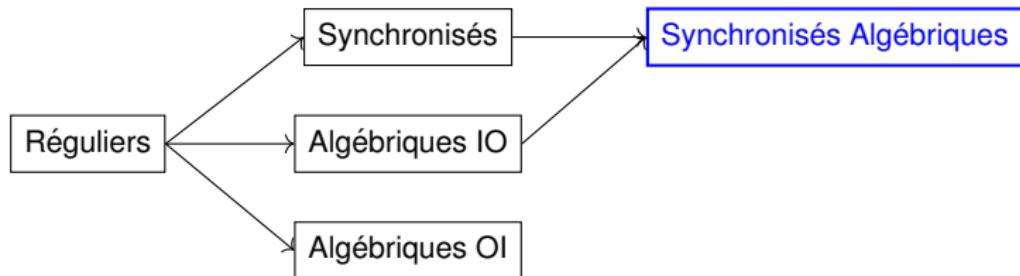
- RTA, 2011, J. Kochems et L. Ong, Improved functional flow and reachability analyses using indexed linear tree grammars

# Les différentes techniques



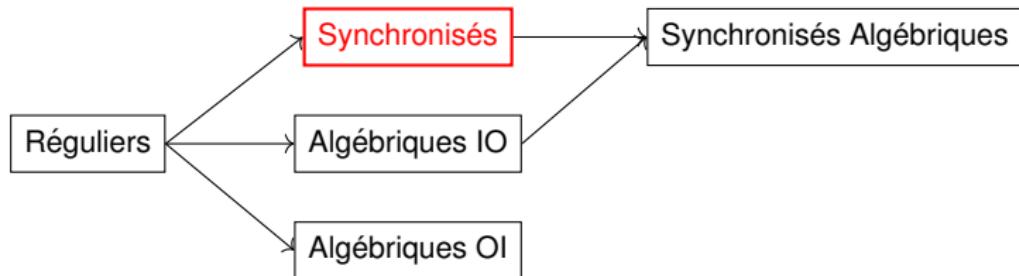
- RTA, 2013, Y. Boichut, J. Chabin, et P. Réty, Over-approximating descendants by synchronized tree languages

# Les différentes techniques



- LATA, 2015, Y. Boichut, J. Chabin, et P. Réty, Towards more precise rewriting approximations

# Les différentes techniques



- WRLA, 2016, Y. Boichut, V. Pelletier, et P. Réty, Synchronized tree languages for reachability in non-right-linear term rewrite systems

# Sommaire

1 Introduction

2 Préliminaires

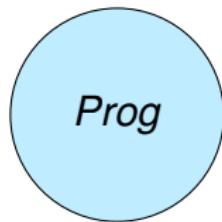
3 Sur-approximations d'ensembles de descendants

- Méthode de compléction
- Descendants innermost
- Élimination des clauses copiantes

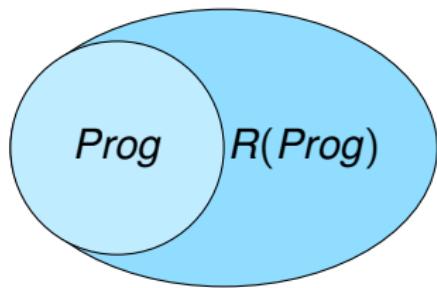
4 Transformation de systèmes de réécriture avec stratégie

5 Conclusion

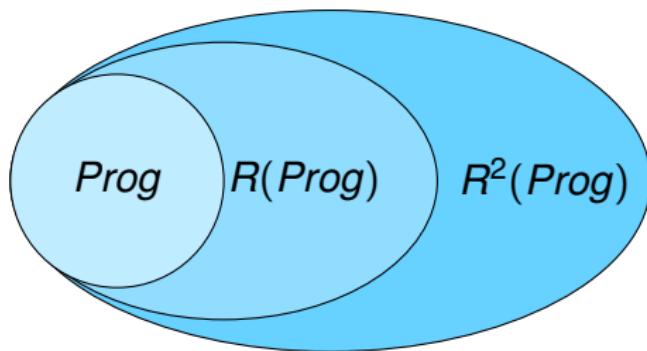
# Méthode de complétion



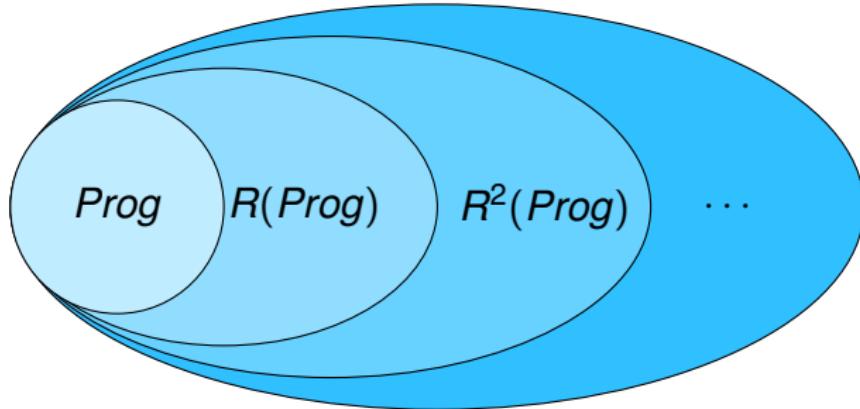
# Méthode de complétion



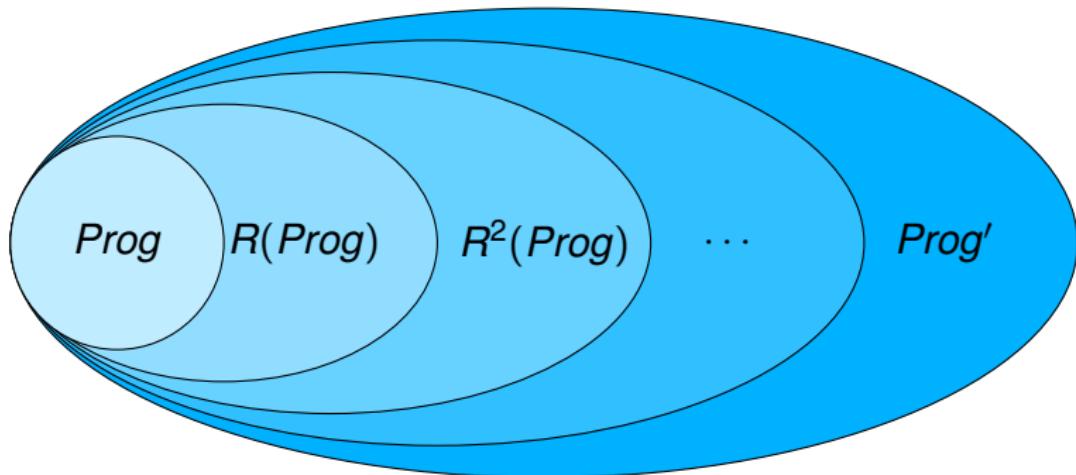
# Méthode de complétion



# Méthode de complétion



# Méthode de complétion



- Prédicats avec un nombre fixe d'arguments

- Prédicats avec un nombre fixe d'arguments
- Atome : prédicat avec des termes en argument

- Prédicats avec un nombre fixe d'arguments
- Atome : prédicat avec des termes en argument
- Conjonctions d'atomes :  $B = A_1, \dots, A_n$

- Prédicats avec un nombre fixe d'arguments
- Atome : prédicat avec des termes en argument
- Conjonctions d'atomes :  $B = A_1, \dots, A_n$
- Propriétés

- Prédicats avec un nombre fixe d'arguments
- Atome : prédicat avec des termes en argument
- Conjonctions d'atomes :  $B = A_1, \dots, A_n$
- Propriétés
  - plat : tous les arguments des atomes de  $B$  sont des variables

- Prédicats avec un nombre fixe d'arguments
- Atome : prédicat avec des termes en argument
- Conjonctions d'atomes :  $B = A_1, \dots, A_n$
- Propriétés
  - plat : tous les arguments des atomes de  $B$  sont des variables
  - linéaire : chaque variable dans  $B$  n'apparaît qu'une seule fois

- Prédicats avec un nombre fixe d'arguments
- Atome : prédicat avec des termes en argument
- Conjonctions d'atomes :  $B = A_1, \dots, A_n$
- Propriétés
  - plat : tous les arguments des atomes de  $B$  sont des variables
  - linéaire : chaque variable dans  $B$  n'apparaît qu'une seule fois
  - $\emptyset$  : conjonction d'atomes vide : plat et linéaire

## CS-clause

- Prédicats avec un nombre fixe d'arguments
- Atome : prédicat avec des termes en argument
- Conjonctions d'atomes :  $B = A_1, \dots, A_n$
- Propriétés
  - plat : tous les arguments des atomes de  $B$  sont des variables
  - linéaire : chaque variable dans  $B$  n'apparaît qu'une seule fois
  - $\emptyset$  : conjonction d'atomes vide : plat et linéaire

### CS-clause

$H \leftarrow B$  avec  $B$  plat et linéaire

## CS-clause

- Prédicats avec un nombre fixe d'arguments
- Atome : prédicat avec des termes en argument
- Conjonctions d'atomes :  $B = A_1, \dots, A_n$
- Propriétés
  - plat : tous les arguments des atomes de  $B$  sont des variables
  - linéaire : chaque variable dans  $B$  n'apparaît qu'une seule fois
  - $\emptyset$  : conjonction d'atomes vide : plat et linéaire

### CS-clause

$H \leftarrow B$  avec  $B$  plat et linéaire

### Exemple

- $P(f(g(x, y)), a) \leftarrow Q(x, y)$  : CS-clause

## CS-clause

- Prédicats avec un nombre fixe d'arguments
- Atome : prédicat avec des termes en argument
- Conjonctions d'atomes :  $B = A_1, \dots, A_n$
- Propriétés
  - plat : tous les arguments des atomes de  $B$  sont des variables
  - linéaire : chaque variable dans  $B$  n'apparaît qu'une seule fois
  - $\emptyset$  : conjonction d'atomes vide : plat et linéaire

### CS-clause

$H \leftarrow B$  avec  $B$  plat et linéaire

### Exemple

- $P(f(g(x, y)), a) \leftarrow Q(x, y)$  : CS-clause
- $P(x) \leftarrow Q(x), Q_2(g(x))$  : Pas une CS-clause

## Propriétés des CS-clauses

Une clause  $H \leftarrow B$  est

## Propriétés des cs-clauses

Une clause  $H \leftarrow B$  est

- copiante si  $H$  n'est pas linéaire

## Propriétés des cs-clauses

Une clause  $H \leftarrow B$  est

- copiante si  $H$  n'est pas linéaire
- normalisée si tous les arguments sont au plus de profondeur 1

## Propriétés des cs-clauses

Une clause  $H \leftarrow B$  est

- copiante si  $H$  n'est pas linéaire
- normalisée si tous les arguments sont au plus de profondeur 1

## Exemple

- $P(f(g(x, a)), x) \leftarrow Q(x)$  : copiante et non normalisée

## Propriétés des cs-clauses

Une clause  $H \leftarrow B$  est

- copiante si  $H$  n'est pas linéaire
- normalisée si tous les arguments sont au plus de profondeur 1

## Exemple

- $P(f(g(x, a)), x) \leftarrow Q(x)$  : copiante et non normalisée
- $P(f(x, y), a) \leftarrow Q'(x, y)$  : non copiante et normalisée

- cs-programme : Ensemble de cs-clauses

- cs-programme : Ensemble de cs-clauses
- Opérateurs

- cs-programme : Ensemble de cs-clauses
  - Opérateurs
- 
- $G \rightsquigarrow G'$  : (dérivation classique)  
 $\exists H \leftarrow B \in \text{Prog}$  et  $\exists \sigma$   
t.q.  $A \in G$  avec  $\sigma(A) = \sigma(H)$  (unification)  
et  $G' = \sigma(G)$  avec  $\sigma(A)$  remplacé par  $\sigma(B)$

- cs-programme : Ensemble de cs-clauses
  - Opérateurs
- 
- $G \rightsquigarrow G'$  : (dérivation classique)  
 $\exists H \leftarrow B \in \text{Prog}$  et  $\exists \sigma$   
t.q.  $A \in G$  avec  $\sigma(A) = \sigma(H)$  (unification)  
et  $G' = \sigma(G)$  avec  $\sigma(A)$  remplacé par  $\sigma(B)$
  - $G \rightarrow G'$  : (dérivation faible)  
comme ci-dessus mais avec  $A = \sigma(H)$  (filtrage)

# CS-programme

- cs-programme : Ensemble de cs-clauses
- Opérateurs

- $G \rightsquigarrow G'$  : (dérivation classique)  
 $\exists H \leftarrow B \in \text{Prog}$  et  $\exists \sigma$   
t.q.  $A \in G$  avec  $\sigma(A) = \sigma(H)$  (unification)  
et  $G' = \sigma(G)$  avec  $\sigma(A)$  remplacé par  $\sigma(B)$
- $G \rightarrow G'$  : (dérivation faible)  
comme ci-dessus mais avec  $A = \sigma(H)$  (filtrage)

## Langage d'un cs-programme

Soit  $\vec{t}$  des termes clos.  $\vec{t} \in \mathcal{L}_{\text{Prog}}(P)$  si et seulement si  $P(\vec{t}) \rightsquigarrow_{\text{Prog}}^* \emptyset$ .

# Exemple

- $\text{Prog} =$

# Exemple

- $\text{Prog} =$ 
  - ①  $P(f(x, y)) \leftarrow Q(x, y)$

# Exemple

- $\text{Prog} =$

- 1  $P(f(x, y)) \leftarrow Q(x, y)$
- 2  $Q(v(x), w(y)) \leftarrow Q(x, y)$

# Exemple

- $\text{Prog} =$

- 1  $P(f(x, y)) \leftarrow Q(x, y)$
- 2  $Q(v(x), w(y)) \leftarrow Q(x, y)$
- 3  $Q(a, a)$

# Exemple

- $\text{Prog} =$ 
  - 1  $P(f(x, y)) \leftarrow Q(x, y)$
  - 2  $Q(v(x), w(y)) \leftarrow Q(x, y)$
  - 3  $Q(a, a)$
- $f(v(a), w(a)) \in \mathcal{L}_{\text{Prog}}(P) ?$

# Exemple

- $\text{Prog} =$ 
  - 1  $P(f(x, y)) \leftarrow Q(x, y)$
  - 2  $Q(v(x), w(y)) \leftarrow Q(x, y)$
  - 3  $Q(a, a)$
- $f(v(a), w(a)) \in \mathcal{L}_{\text{Prog}}(P) ?$ 
  - $P(f(v(a), w(a))) \rightsquigarrow Q(v(a), w(a))$

# Exemple

- $\text{Prog} =$ 
  - 1  $P(f(x, y)) \leftarrow Q(x, y)$
  - 2  $Q(v(x), w(y)) \leftarrow Q(x, y)$
  - 3  $Q(a, a)$
- $f(v(a), w(a)) \in \mathcal{L}_{\text{Prog}}(P) ?$ 
  - $P(f(v(a), w(a))) \rightsquigarrow Q(v(a), w(a))$
  - $Q(v(a), w(a)) \rightsquigarrow Q(a, a)$

# Exemple

- $\text{Prog} =$ 
  - 1  $P(f(x, y)) \leftarrow Q(x, y)$
  - 2  $Q(v(x), w(y)) \leftarrow Q(x, y)$
  - 3  $Q(a, a)$
- $f(v(a), w(a)) \in \mathcal{L}_{\text{Prog}}(P) ?$ 
  - $P(f(v(a), w(a))) \rightsquigarrow Q(v(a), w(a))$
  - $Q(v(a), w(a)) \rightsquigarrow Q(a, a)$
  - $Q(a, a) \rightsquigarrow \emptyset$

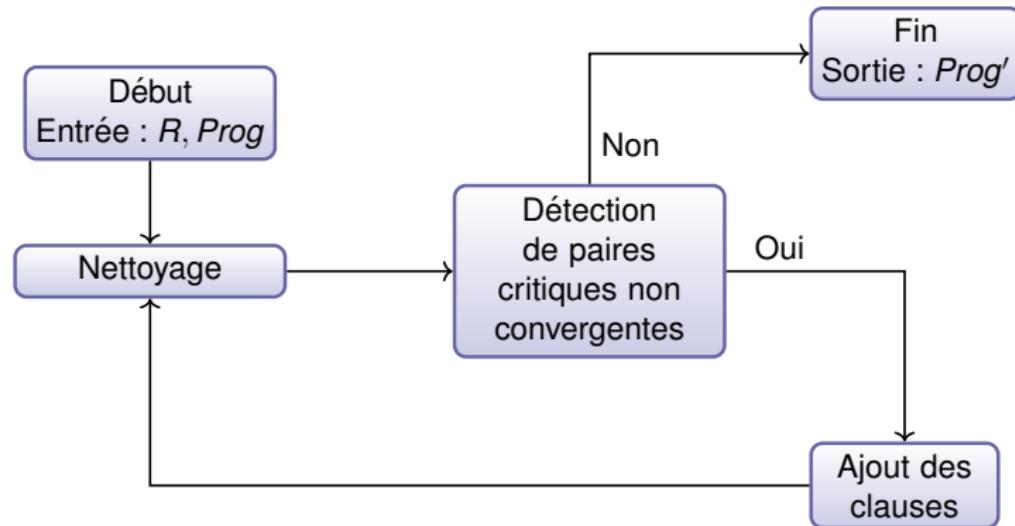
# Exemple

- $\text{Prog} =$ 
  - 1  $P(f(x, y)) \leftarrow Q(x, y)$
  - 2  $Q(v(x), w(y)) \leftarrow Q(x, y)$
  - 3  $Q(a, a)$
- $f(v(a), w(a)) \in \mathcal{L}_{\text{Prog}}(P) ?$ 
  - $P(f(v(a), w(a))) \rightsquigarrow Q(v(a), w(a))$
  - $Q(v(a), w(a)) \rightsquigarrow Q(a, a)$
  - $Q(a, a) \rightsquigarrow \emptyset$
  - Oui,  $f(v(a), w(a)) \in \mathcal{L}_{\text{Prog}}(P)$

# Exemple

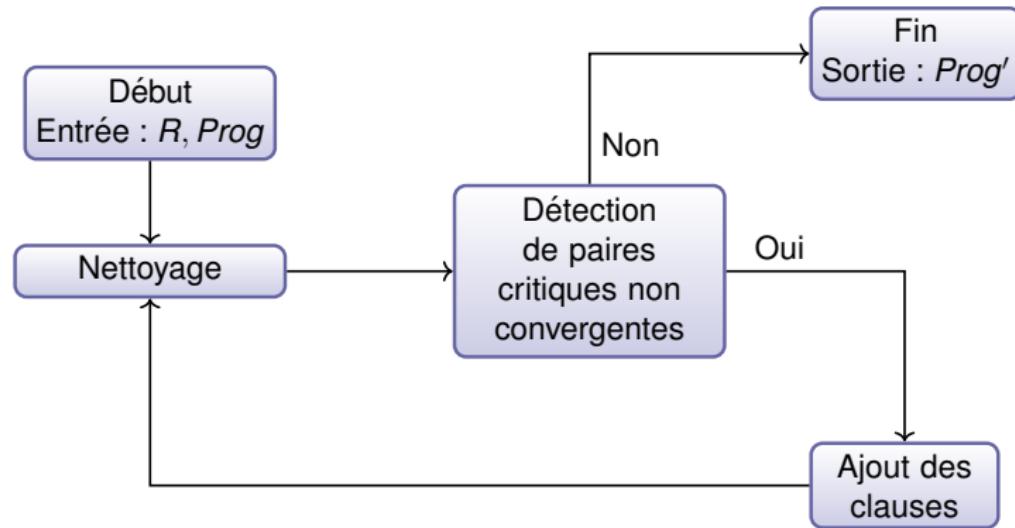
- $\text{Prog} =$ 
  - 1  $P(f(x, y)) \leftarrow Q(x, y)$
  - 2  $Q(v(x), w(y)) \leftarrow Q(x, y)$
  - 3  $Q(a, a)$
- $f(v(a), w(a)) \in \mathcal{L}_{\text{Prog}}(P) ?$ 
  - $P(f(v(a), w(a))) \rightsquigarrow Q(v(a), w(a))$
  - $Q(v(a), w(a)) \rightsquigarrow Q(a, a)$
  - $Q(a, a) \rightsquigarrow \emptyset$
  - Oui,  $f(v(a), w(a)) \in \mathcal{L}_{\text{Prog}}(P)$
- $\mathcal{L}_{\text{Prog}}(P) = \{f(v^n(a), w^n(a)) | n \in \mathbb{N}\}$

# L'algorithme de compléction



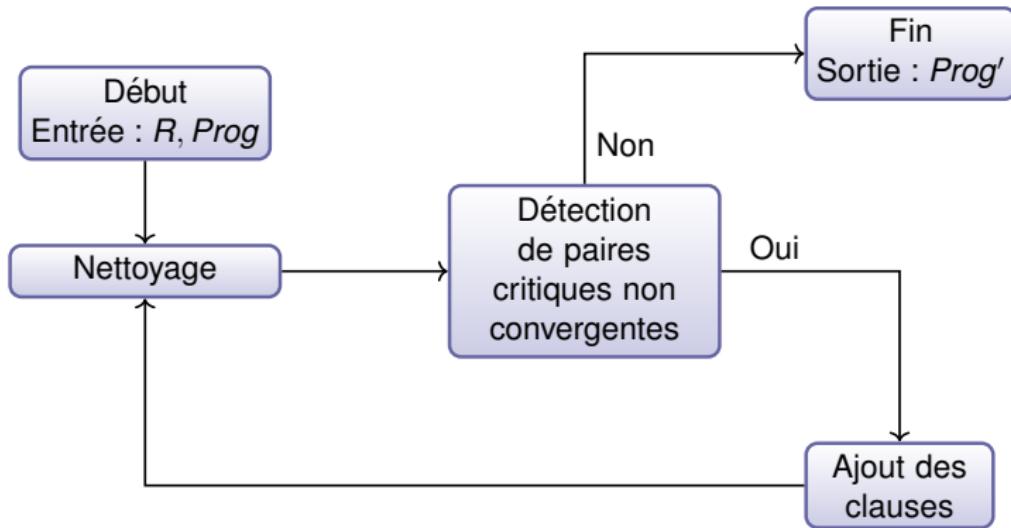
- Un cs-programme initial

# L'algorithme de compléction



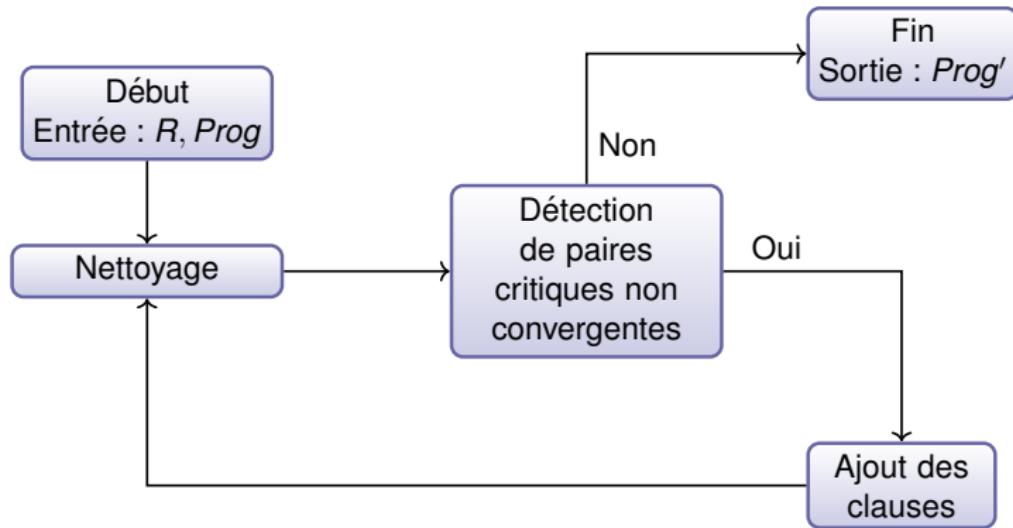
- Un cs-programme initial
  - non copiant

# L'algorithme de compléction



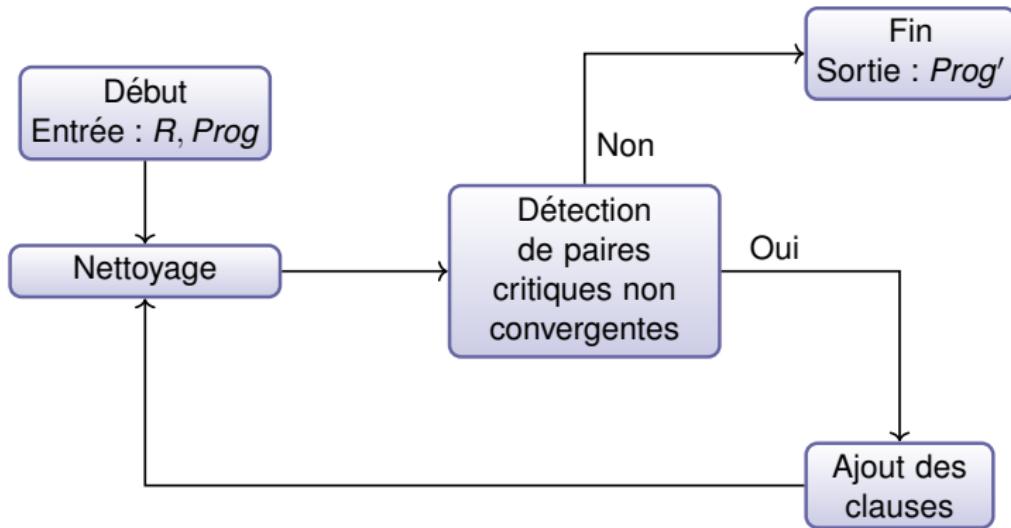
- Un cs-programme initial
  - non copiant
  - normalisé

# L'algorithme de compléction



- Un cs-programme initial
  - non copiant
  - normalisé
- Un système de réécriture

# L'algorithme de complétion



- Un cs-programme initial
  - non copiant
  - normalisé
- Un système de réécriture
  - linéaire

# Paires critiques

- $I \rightarrow r \in R$

# Paires critiques

- $I \rightarrow r \in R$
- $P(\dots, t_i, \dots) \leftarrow B \in \text{Prog}$

# Paires critiques

- $I \rightarrow r \in R$
- $P(..., t_i, ...) \leftarrow B \in Prog$

$P(..., I, ...)$

# Paires critiques

- $I \rightarrow r \in R$
- $P(\dots, t_i, \dots) \leftarrow B \in \text{Prog}$

$P(\dots, I, \dots)$

~~~~~ $\xrightarrow{[\sigma]}^+$

$G$  avec  $G$  plat

# Paires critiques

- $I \rightarrow r \in R$
- $P(\dots, t_i, \dots) \leftarrow B \in \text{Prog}$

$P(\dots, I, \dots)$

~~~~~ $\xrightarrow{[\sigma]}^+$

$G$  avec  $G$  plat



$\sigma(P(\dots, r, \dots)) \leftarrow G$  : paire critique

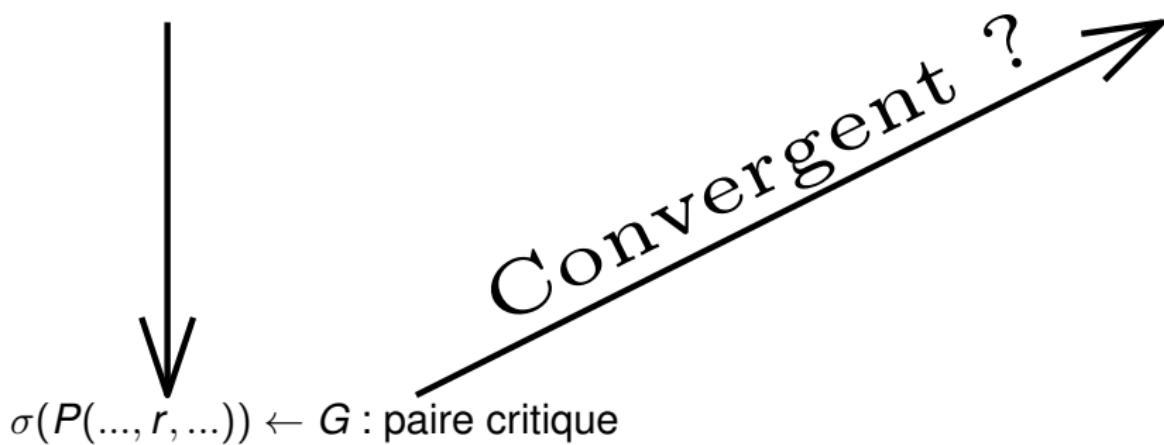
# Paires critiques

- $I \rightarrow r \in R$
- $P(\dots, t_i, \dots) \leftarrow B \in \text{Prog}$

$P(\dots, I, \dots)$

~~~~~ $\xrightarrow{[\sigma]}^+$

$G$  avec  $G$  plat



# Exemple

- $R : \{g(h(y)) \rightarrow f(y)\}$

# Exemple

- $R : \{g(h(y)) \rightarrow f(y)\}$
- $Prog : \{P(g(x)) \leftarrow Q(x). \ Q(h(x)) \leftarrow Q'(x). \ Q'(a).\}$

# Exemple

- $R : \{g(h(y)) \rightarrow f(y)\}$
  - $Prog : \{P(g(x)) \leftarrow Q(x). \ Q(h(x)) \leftarrow Q'(x). \ Q'(a).\}$
- $P(g(h(y)))$

# Exemple

- $R : \{g(h(y)) \rightarrow f(y)\}$
- $Prog : \{P(g(x)) \leftarrow Q(x). \ Q(h(x)) \leftarrow Q'(x). \ Q'(a).\}$

$P(g(h(y))) \rightsquigarrow Q(h(y))$

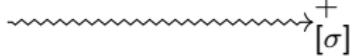
# Exemple

- $R : \{g(h(y)) \rightarrow f(y)\}$
- $Prog : \{P(g(x)) \leftarrow Q(x). \text{ } Q(h(x)) \leftarrow Q'(x). \text{ } Q'(a).\}$

$P(g(h(y))) \rightsquigarrow Q(h(y)) \rightsquigarrow Q'(y)$

# Exemple

- $R : \{g(h(y)) \rightarrow f(y)\}$
- $Prog : \{P(g(x)) \leftarrow Q(x). \ Q(h(x)) \leftarrow Q'(x). \ Q'(a).\}$

$P(g(h(y)))$              $Q'(y)$  et  $Q'(y)$  plat

# Exemple

- $R : \{g(h(y)) \rightarrow f(y)\}$
- $Prog : \{P(g(x)) \leftarrow Q(x). \ Q(h(x)) \leftarrow Q'(x). \ Q'(a).\}$

$P(g(h(y)))$

~~~~~ $\xrightarrow{[\sigma]}^+$

$Q'(y)$  et  $Q'(y)$  plat



$\sigma(P(f(y))) \leftarrow Q'(y) : \text{paire critique}$

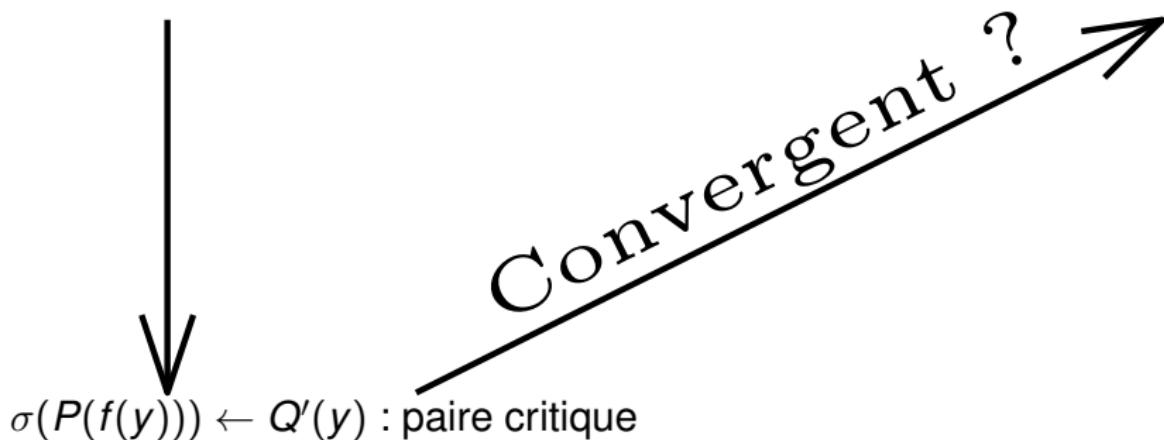
# Exemple

- $R : \{g(h(y)) \rightarrow f(y)\}$
- $Prog : \{P(g(x)) \leftarrow Q(x). \ Q(h(x)) \leftarrow Q'(x). \ Q'(a).\}$

$P(g(h(y)))$

~~~~~ $\xrightarrow{[\sigma]}$

$Q'(y)$  et  $Q'(y)$  plat



# Le problème

- $\text{Prog} = \{P(g(x)) \leftarrow Q(x). \ Q(a). \ Q(b).\}$

# Le problème

- $Prog = \{P(g(x)) \leftarrow Q(x). \ Q(a). \ Q(b).\}$
- $R = \{g(x) \rightarrow f(x, x), \ a \rightarrow b\}$

# Le problème

- $\text{Prog} = \{ P(g(x)) \leftarrow Q(x). \ Q(a). \ Q(b). \}$
- $R = \{ g(x) \rightarrow f(x, x), \ a \rightarrow b \}$
- 2 paires critiques :  $P(f(x, x)) \leftarrow Q(x)$

# Le problème

- $Prog = \{P(g(x)) \leftarrow Q(x). \text{ } Q(a). \text{ } Q(b).\}$
- $R = \{g(x) \rightarrow f(x, x), \text{ } a \rightarrow b\}$
- 2 paires critiques :  $P(f(x, x)) \leftarrow Q(x)$  et  $Q(b)$

# Le problème

- $\text{Prog} = \{P(g(x)) \leftarrow Q(x). \ Q(a). \ \textcolor{blue}{Q(b).}\}$
- $R = \{g(x) \rightarrow f(x, x), \ a \rightarrow b\}$
- 2 paires critiques :  $P(f(x, x)) \leftarrow Q(x)$  et  $Q(b)$
- $Q(b)$  est convergente

# Le problème

- $Prog = \{P(g(x)) \leftarrow Q(x). \quad Q(a). \quad Q(b).\}$
- $R = \{g(x) \rightarrow f(x, x), \quad a \rightarrow b\}$
- 2 paires critiques :  $P(f(x, x)) \leftarrow Q(x)$  et  $Q(b)$
- $Q(b)$  est convergente
- On ajoute  $P(f(x, x)) \leftarrow Q(x)$  à  $Prog$

# Le problème

- $Prog = \{P(g(x)) \leftarrow Q(x). \quad Q(a). \quad Q(b).\}$
- $R = \{g(x) \rightarrow f(x, x), \quad a \rightarrow b\}$
- 2 paires critiques :  $P(f(x, x)) \leftarrow Q(x)$  et  $Q(b)$
- $Q(b)$  est convergente
- On ajoute  $P(f(x, x)) \leftarrow Q(x)$  à  $Prog$
- Toutes les pairs critiques sont convergentes

# Le problème

- $Prog = \{P(g(x)) \leftarrow Q(x). \quad Q(a). \quad Q(b).\}$
- $R = \{g(x) \rightarrow f(x, x), \quad a \rightarrow b\}$
- 2 paires critiques :  $P(f(x, x)) \leftarrow Q(x)$  et  $Q(b)$
- $Q(b)$  est convergente
- On ajoute  $P(f(x, x)) \leftarrow Q(x)$  à  $Prog$
- Toutes les pairs critiques sont convergentes
- $f(a, a) \in \mathcal{L}_{Prog}(P)$

# Le problème

- $Prog = \{P(g(x)) \leftarrow Q(x). \quad Q(a). \quad Q(b).\}$
- $R = \{g(x) \rightarrow f(x, x), \quad a \rightarrow b\}$
- 2 paires critiques :  $P(f(x, x)) \leftarrow Q(x)$  et  $Q(b)$
- $Q(b)$  est convergente
- On ajoute  $P(f(x, x)) \leftarrow Q(x)$  à  $Prog$
- Toutes les pairs critiques sont convergentes
- $f(a, a) \in \mathcal{L}_{Prog}(P)$
- $f(a, a) \rightarrow_R f(b, a)$

# Le problème

- $Prog = \{P(g(x)) \leftarrow Q(x). \quad Q(a). \quad Q(b).\}$
- $R = \{g(x) \rightarrow f(x, x), \quad a \rightarrow b\}$
- 2 paires critiques :  $P(f(x, x)) \leftarrow Q(x)$  et  $Q(b)$
- $Q(b)$  est convergente
- On ajoute  $P(f(x, x)) \leftarrow Q(x)$  à  $Prog$
- Toutes les pairs critiques sont convergentes
- $f(a, a) \in \mathcal{L}_{Prog}(P)$
- $f(a, a) \rightarrow_R f(b, a)$
- mais  $f(b, a)$  et  $f(a, b) \notin \mathcal{L}_{Prog}(P)$

# Sommaire

1 Introduction

2 Préliminaires

3 Sur-approximations d'ensembles de descendants

- Méthode de compléction
- Descendants innermost
- Élimination des clauses copiantes

4 Transformation de systèmes de réécriture avec stratégie

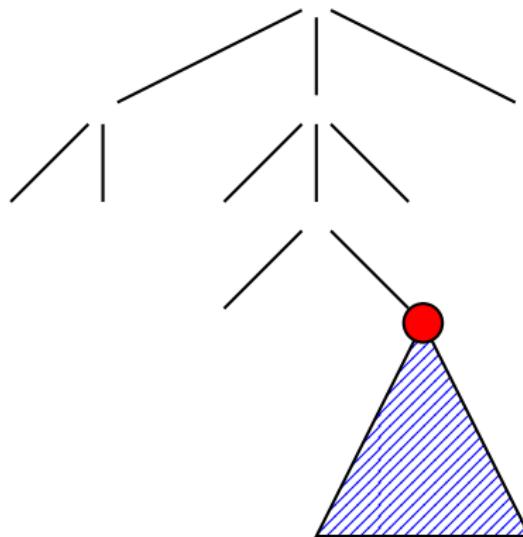
5 Conclusion

# Stratégie innermost

- Restriction des réécritures

# Stratégie innermost

- Restriction des réécritures
- Réécritures de sous-termes non réductibles



# Irréducibilité forte

## Irréducibilité forte

Pour tout  $l \rightarrow r \in R$ .

- Un terme  $t$  est *fortement irréductible* (par  $R$ ) si aucun de ses sous-termes ne peut s'unifier avec  $l$ .

# Irréducibilité forte

## Irréducibilité forte

Pour tout  $l \rightarrow r \in R$ .

- Un terme  $t$  est *fortement irréductible* (par  $R$ ) si aucun de ses sous-termes ne peut s'unifier avec  $l$ .
- Une substitution  $\theta$  est *fortement irréductible* si pour tout  $x \in \mathcal{X}$ ,  $\theta(x)$  est fortement irréductible.

# Irréducibilité forte

## Irréducibilité forte

Pour tout  $I \rightarrow r \in R$ .

- Un terme  $t$  est *fortement irréductible* (par  $R$ ) si aucun de ses sous-termes ne peut s'unifier avec  $I$ .
- Une substitution  $\theta$  est *fortement irréductible* si pour tout  $x \in \mathcal{X}$ ,  $\theta(x)$  est fortement irréductible.

## Exemple

- $f(b) \rightarrow b$

# Irréducibilité forte

## Irréducibilité forte

Pour tout  $I \rightarrow r \in R$ .

- Un terme  $t$  est *fortement irréductible* (par  $R$ ) si aucun de ses sous-termes ne peut s'unifier avec  $I$ .
- Une substitution  $\theta$  est *fortement irréductible* si pour tout  $x \in \mathcal{X}$ ,  $\theta(x)$  est fortement irréductible.

## Exemple

- $f(b) \rightarrow b$
- $t = f(x) :$

# Irréducibilité forte

## Irréducibilité forte

Pour tout  $I \rightarrow r \in R$ .

- Un terme  $t$  est *fortement irréductible* (par  $R$ ) si aucun de ses sous-termes ne peut s'unifier avec  $I$ .
- Une substitution  $\theta$  est *fortement irréductible* si pour tout  $x \in \mathcal{X}$ ,  $\theta(x)$  est fortement irréductible.

## Exemple

- $f(b) \rightarrow b$
- $t = f(x)$  : irréductible

# Irréducibilité forte

## Irréducibilité forte

Pour tout  $l \rightarrow r \in R$ .

- Un terme  $t$  est *fortement irréductible* (par  $R$ ) si aucun de ses sous-termes ne peut s'unifier avec  $l$ .
- Une substitution  $\theta$  est *fortement irréductible* si pour tout  $x \in \mathcal{X}$ ,  $\theta(x)$  est fortement irréductible.

## Exemple

- $f(b) \rightarrow b$
- $t = f(x)$  : irréductible
- $\theta = (x/a)$  :

# Irréducibilité forte

## Irréducibilité forte

Pour tout  $l \rightarrow r \in R$ .

- Un terme  $t$  est *fortement irréductible* (par  $R$ ) si aucun de ses sous-termes ne peut s'unifier avec  $l$ .
- Une substitution  $\theta$  est *fortement irréductible* si pour tout  $x \in \mathcal{X}$ ,  $\theta(x)$  est fortement irréductible.

## Exemple

- $f(b) \rightarrow b$
- $t = f(x)$  : irréductible
- $\theta = (x/a)$  : est fortement irréductible

# Irréducibilité forte

## Irréducibilité forte

Pour tout  $l \rightarrow r \in R$ .

- Un terme  $t$  est *fortement irréductible* (par  $R$ ) si aucun de ses sous-termes ne peut s'unifier avec  $l$ .
- Une substitution  $\theta$  est *fortement irréductible* si pour tout  $x \in \mathcal{X}$ ,  $\theta(x)$  est fortement irréductible.

## Exemple

- $f(b) \rightarrow b$
- $t = f(x)$  : irréductible
- $\theta = (x/a)$  : est fortement irréductible
- $\theta(t) = f(a)$  :

# Irréducibilité forte

## Irréducibilité forte

Pour tout  $l \rightarrow r \in R$ .

- Un terme  $t$  est *fortement irréductible* (par  $R$ ) si aucun de ses sous-termes ne peut s'unifier avec  $l$ .
- Une substitution  $\theta$  est *fortement irréductible* si pour tout  $x \in \mathcal{X}$ ,  $\theta(x)$  est fortement irréductible.

## Exemple

- $f(b) \rightarrow b$
- $t = f(x)$  : irréductible
- $\theta = (x/a)$  : est fortement irréductible
- $\theta(t) = f(a)$  : fortement irréductible

## NC et SNC

Soit  $A$  un atome ( $A$  peut contenir des variables).

- L'étape  $A \rightsquigarrow_{[H \leftarrow B, \sigma]} G$  est NC (resp. SNC) si pour toute variables  $x \in \text{Var}(H)$  non linéaires,  $x$  est irréducible (resp. fortement irréducible) par  $R$ .

## NC et SNC

Soit  $A$  un atome ( $A$  peut contenir des variables).

- L'étape  $A \rightsquigarrow_{[H \leftarrow B, \sigma]} G$  est NC (resp. SNC) si pour toute variables  $x \in \text{Var}(H)$  non linéaires,  $x$  est irréducible (resp. fortement irréducible) par  $R$ .
- Une dérivation est NC (resp. SNC) si toutes ses étapes le sont.

## NC et SNC

Soit  $A$  un atome ( $A$  peut contenir des variables).

- L'étape  $A \rightsquigarrow_{[H \leftarrow B, \sigma]} G$  est NC (resp. SNC) si pour toute variables  $x \in \text{Var}(H)$  non linéaires,  $x$  est irréducible (resp. fortement irréducible) par  $R$ .
- Une dérivation est NC (resp. SNC) si toutes ses étapes le sont.
- Une clause non copiante est SNC

## NC et SNC

Soit  $A$  un atome ( $A$  peut contenir des variables).

- L'étape  $A \rightsquigarrow_{[H \leftarrow B, \sigma]} G$  est NC (resp. SNC) si pour toute variables  $x \in \text{Var}(H)$  non linéaires,  $x$  est irréducible (resp. fortement irréducible) par  $R$ .
- Une dérivation est NC (resp. SNC) si toutes ses étapes le sont.
- Une clause non copiante est SNC

## Exemple

- $P(g(x, x)) \leftarrow Q(x)$

## NC et SNC

Soit  $A$  un atome ( $A$  peut contenir des variables).

- L'étape  $A \rightsquigarrow_{[H \leftarrow B, \sigma]} G$  est NC (resp. SNC) si pour toute variables  $x \in \text{Var}(H)$  non linéaires,  $x$  est irréducible (resp. fortement irréducible) par  $R$ .
- Une dérivation est NC (resp. SNC) si toutes ses étapes le sont.
- Une clause non copiante est SNC

## Exemple

- $P(g(x, x)) \leftarrow Q(x)$
- $R = \{h(a) \rightarrow b\}$

## NC et SNC

Soit  $A$  un atome ( $A$  peut contenir des variables).

- L'étape  $A \rightsquigarrow_{[H \leftarrow B, \sigma]} G$  est NC (resp. SNC) si pour toute variables  $x \in \text{Var}(H)$  non linéaires,  $x$  est irréducible (resp. fortement irréducible) par  $R$ .
- Une dérivation est NC (resp. SNC) si toutes ses étapes le sont.
- Une clause non copiante est SNC

## Exemple

- $P(g(x, x)) \leftarrow Q(x)$
- $R = \{h(a) \rightarrow b\}$
- $P(g(h(y), h(y)))$

## NC et SNC

Soit  $A$  un atome ( $A$  peut contenir des variables).

- L'étape  $A \rightsquigarrow_{[H \leftarrow B, \sigma]} G$  est NC (resp. SNC) si pour toute variables  $x \in \text{Var}(H)$  non linéaires,  $x$  est irréducible (resp. fortement irréducible) par  $R$ .
- Une dérivation est NC (resp. SNC) si toutes ses étapes le sont.
- Une clause non copiante est SNC

## Exemple

- $P(g(x, x)) \leftarrow Q(x)$
- $R = \{h(a) \rightarrow b\}$
- $P(g(h(y), h(y))) \rightsquigarrow Q(h(y)) :$

## NC et SNC

Soit  $A$  un atome ( $A$  peut contenir des variables).

- L'étape  $A \rightsquigarrow_{[H \leftarrow B, \sigma]} G$  est NC (resp. SNC) si pour toute variables  $x \in \text{Var}(H)$  non linéaires,  $x$  est irréducible (resp. fortement irréducible) par  $R$ .
- Une dérivation est NC (resp. SNC) si toutes ses étapes le sont.
- Une clause non copiante est SNC

## Exemple

- $P(g(x, x)) \leftarrow Q(x)$
- $R = \{h(a) \rightarrow b\}$
- $P(g(h(y), h(y))) \rightsquigarrow Q(h(y))$  : NC mais pas SNC

# Clôture par réécriture innermost

Théorème [WRLA, 2016, Y. Boichut, V. Pelletier et P. Réty]

Soit  $R$  un système de réécriture linéaire gauche.

# Clôture par réécriture innermost

Théorème [WRLA, 2016, Y. Boichut, V. Pelletier et P. Réty]

Soit  $R$  un système de réécriture linéaire gauche.

Soit  $\text{Prog}$  un cs-programme normalisé et non copiant.

# Clôture par réécriture innermost

Théorème [WRLA, 2016, Y. Boichut, V. Pelletier et P. Réty]

Soit  $R$  un système de réécriture linéaire gauche.

Soit  $\text{Prog}$  un cs-programme normalisé et non copiant.

Soit  $\text{Prog}' \supseteq \text{Prog}$  un cs-programme

# Clôture par réécriture innermost

Théorème [WRLA, 2016, Y. Boichut, V. Pelletier et P. Réty]

Soit  $R$  un système de réécriture linéaire gauche.

Soit  $\text{Prog}$  un cs-programme normalisé et non copiant.

Soit  $\text{Prog}' \supseteq \text{Prog}$  un cs-programme

tel que toutes les paires critiques de  $\text{Prog}$  sont convergentes  
par dérivations SNC dans  $\text{Prog}'$ .

# Clôture par réécriture innermost

Théorème [WRLA, 2016, Y. Boichut, V. Pelletier et P. Réty]

Soit  $R$  un système de réécriture linéaire gauche.

Soit  $\text{Prog}$  un cs-programme normalisé et non copiant.

Soit  $\text{Prog}' \supseteq \text{Prog}$  un cs-programme

tel que toutes les paires critiques de  $\text{Prog}$  sont convergentes  
par dérivations SNC dans  $\text{Prog}'$ .

Si un terme  $t \in \mathcal{L}_{\text{Prog}}(P)$  et  $t \rightarrow_R^* t'$  avec une stratégie innermost, alors  
 $t' \in \mathcal{L}_{\text{Prog}'}(P)$ .

# Sommaire

1 Introduction

2 Préliminaires

3 Sur-approximations d'ensembles de descendants

- Méthode de compléction
- Descendants innermost
- Élimination des clauses copiantes

4 Transformation de systèmes de réécriture avec stratégie

5 Conclusion

# Origine des clauses copiantes

- Les réécritures non couvertes proviennent des clauses copiantes

# Origine des clauses copiantes

- Les réécritures non couvertes proviennent des clauses copiantes
  - Les clauses copiantes proviennent

# Origine des clauses copiantes

- Les réécritures non couvertes proviennent des clauses copiantes
  - Les clauses copiantes proviennent
    - de clauses copiantes présentes initialement

# Origine des clauses copiantes

- Les réécritures non couvertes proviennent des clauses copiantes
  - Les clauses copiantes proviennent
    - de clauses copiantes présentes initialement
    - du système de réécriture non linéaire droit

# Origine des clauses copiantes

- Les réécritures non couvertes proviennent des clauses copiantes
  - Les clauses copiantes proviennent
    - de clauses copiantes présentes initialement
    - du système de réécriture non linéaire droit
- Solution

# Origine des clauses copiantes

- Les réécritures non couvertes proviennent des clauses copiantes
  - Les clauses copiantes proviennent
    - de clauses copiantes présentes initialement
    - du système de réécriture non linéaire droit
- Solution
  - transformer les clauses copiantes en non copiantes

# Exemple

- $\text{Prog} =$

# Exemple

- $\text{Prog} =$ 
  - ➊  $P(f(x, x)) \leftarrow Q(x)$

# Exemple

- $\text{Prog} =$

- 1  $P(f(x, x)) \leftarrow Q(x)$
- 2  $Q(s(x)) \leftarrow Q(x)$

# Exemple

- $\text{Prog} =$

- 1  $P(f(x, x)) \leftarrow Q(x)$
- 2  $Q(s(x)) \leftarrow Q(x)$
- 3  $Q(a)$

# Exemple

- $\text{Prog} =$ 
  - 1  $P(f(x, x)) \leftarrow Q(x)$
  - 2  $Q(s(x)) \leftarrow Q(x)$
  - 3  $Q(a)$
- $\text{uncopying}(P(f(x, x)) \leftarrow Q(x))$

# Exemple

- $\text{Prog} =$ 
  - 1  $P(f(x, x)) \leftarrow Q(x)$
  - 2  $Q(s(x)) \leftarrow Q(x)$
  - 3  $Q(a)$
- $\text{uncopying}(P(f(x, x)) \leftarrow Q(x))$
- $P(f(x, y)) \leftarrow Q^2(x, y)$

# Exemple

- $\text{Prog} =$ 
  - 1  $P(f(x, x)) \leftarrow Q(x)$
  - 2  $Q(s(x)) \leftarrow Q(x)$
  - 3  $Q(a)$
- $\text{uncopying}(P(f(x, x)) \leftarrow Q(x))$
- $P(f(x, y)) \leftarrow Q^2(x, y)$
- $Q^2(s(x), s(y)) \leftarrow Q^2(x, y)$

# Exemple

- $\text{Prog} =$

- 1  $P(f(x, x)) \leftarrow Q(x)$
- 2  $Q(s(x)) \leftarrow Q(x)$
- 3  $Q(a)$

- $\text{uncopying}(P(f(x, x)) \leftarrow Q(x))$
- $P(f(x, y)) \leftarrow Q^2(x, y)$
- $Q^2(s(x), s(y)) \leftarrow Q^2(x, y)$
- $Q^2(a, a) \leftarrow$

# Exemple

- $\text{Prog} =$ 
  - 1  $P(f(x, x)) \leftarrow Q(x)$
  - 2  $Q(s(x)) \leftarrow Q(x)$
  - 3  $Q(a)$
- $\text{uncopying}(P(f(x, x)) \leftarrow Q(x))$
- $P(f(x, y)) \leftarrow Q^2(x, y)$
- $Q^2(s(x), s(y)) \leftarrow Q^2(x, y)$
- $Q^2(a, a) \leftarrow$
- $\mathcal{L}_{\text{Prog}}(Q^2) = \{t \cdot t' \mid t, t' \in \mathcal{L}_{\text{Prog}}(Q) \wedge t = t'\}$

# Exemple

- $\text{Prog} =$ 
  - 1  $P(f(x, x)) \leftarrow Q(x)$
  - 2  $Q(s(x)) \leftarrow Q(x)$
  - 3  $Q(a)$
- $\text{uncopying}(P(f(x, x)) \leftarrow Q(x))$
- $P(f(x, y)) \leftarrow Q^2(x, y)$
- $Q^2(s(x), s(y)) \leftarrow Q^2(x, y)$
- $Q^2(a, a) \leftarrow$
- $\mathcal{L}_{\text{Prog}}(Q^2) = \{t \cdot t' \mid t, t' \in \mathcal{L}_{\text{Prog}}(Q) \wedge t = t'\}$
- Plus de clauses copiantes

# Terminaison de l'élimination de clauses copiantes

- Ne termine pas dans le cas général

# Terminaison de l'élimination de clauses copiantes

- Ne termine pas dans le cas général
- $\text{Prog} =$

# Terminaison de l'élimination de clauses copiantes

- Ne termine pas dans le cas général
- $\text{Prog} =$ 
  - ➊  $P(c(x, x)) \leftarrow P(x)$

# Terminaison de l'élimination de clauses copiantes

- Ne termine pas dans le cas général
- $\text{Prog} =$ 
  - ①  $P(c(x, x)) \leftarrow P(x)$
  - ②  $P(a)$

# Terminaison de l'élimination de clauses copiantes

- Ne termine pas dans le cas général
- $\text{Prog} =$ 
  - 1  $P(c(x, x)) \leftarrow P(x)$
  - 2  $P(a)$
- $\text{uncopying}(P(c(x, x)) \leftarrow P(x))$

# Terminaison de l'élimination de clauses copiantes

- Ne termine pas dans le cas général
- $\text{Prog} =$ 
  - ①  $P(c(x, x)) \leftarrow P(x)$
  - ②  $P(a)$
- $\text{uncopying}(P(c(x, x)) \leftarrow P(x))$
- $P(c(x, y)) \leftarrow P^2(x, y)$

# Terminaison de l'élimination de clauses copiantes

- Ne termine pas dans le cas général

- $\text{Prog} =$

- ①  $P(c(x, x)) \leftarrow P(x)$
- ②  $P(a)$

- $\text{uncopying}(P(c(x, x)) \leftarrow P(x))$
- $P(c(x, y)) \leftarrow P^2(x, y)$
- $P^2(c(x, x), c(y, y)) \leftarrow P^2(x, y)$

# Terminaison de l'élimination de clauses copiantes

- Ne termine pas dans le cas général

- $\text{Prog} =$

- ①  $P(c(x, x)) \leftarrow P(x)$
- ②  $P(a)$

- $\text{uncopying}(P(c(x, x)) \leftarrow P(x))$
- $P(c(x, y)) \leftarrow P^2(x, y)$
- $P^2(c(x, x), c(y, y)) \leftarrow P^2(x, y)$
- $P^2(a, a) \leftarrow$

# Terminaison de l'élimination de clauses copiantes

- Ne termine pas dans le cas général

- $\text{Prog} =$

- 1  $P(c(x, x)) \leftarrow P(x)$
- 2  $P(a)$

- $\text{uncopying}(P(c(x, x)) \leftarrow P(x))$
- $P(c(x, y)) \leftarrow P^2(x, y)$
- $P^2(c(x, x), c(y, y)) \leftarrow P^2(x, y)$
- $P^2(a, a) \leftarrow$
- $\text{uncopying}(P^2(c(x, x), c(y, y)) \leftarrow P^2(x, y))$

# Terminaison de l'élimination de clauses copiantes

- Ne termine pas dans le cas général
- $Prog =$ 
  - ①  $P(c(x, x)) \leftarrow P(x)$
  - ②  $P(a)$
- $uncopying(P(c(x, x)) \leftarrow P(x))$
- $P(c(x, y)) \leftarrow P^2(x, y)$
- $P^2(c(x, x), c(y, y)) \leftarrow P^2(x, y)$
- $P^2(a, a) \leftarrow$
- $uncopying(P^2(c(x, x), c(y, y)) \leftarrow P^2(x, y))$
- $P^2(c(x, x'), c(y, y')) \leftarrow P^4(x, x', y, y')$

# Terminaison de l'élimination de clauses copiantes

- Ne termine pas dans le cas général

- $\text{Prog} =$

- 1  $P(c(x, x)) \leftarrow P(x)$
- 2  $P(a)$

- $\text{uncopying}(P(c(x, x)) \leftarrow P(x))$
- $P(c(x, y)) \leftarrow P^2(x, y)$
- $P^2(c(x, x), c(y, y)) \leftarrow P^2(x, y)$
- $P^2(a, a) \leftarrow$
  
- $\text{uncopying}(P^2(c(x, x), c(y, y)) \leftarrow P^2(x, y))$
- $P^2(c(x, x'), c(y, y')) \leftarrow P^4(x, x', y, y')$
- $P^4(c(x, x), c(x', x'), c(y, y), c(y', y')) \leftarrow P^4(x, x', y, y')$

# Terminaison de l'élimination de clauses copiantes

- Ne termine pas dans le cas général

- $\text{Prog} =$

- 1  $P(c(x, x)) \leftarrow P(x)$
- 2  $P(a)$

- $\text{uncopying}(P(c(x, x)) \leftarrow P(x))$
- $P(c(x, y)) \leftarrow P^2(x, y)$
- $P^2(c(x, x), c(y, y)) \leftarrow P^2(x, y)$
- $P^2(a, a) \leftarrow$
  
- $\text{uncopying}(P^2(c(x, x), c(y, y)) \leftarrow P^2(x, y))$
- $P^2(c(x, x'), c(y, y')) \leftarrow P^4(x, x', y, y')$
- $P^4(c(x, x), c(x', x'), c(y, y), c(y', y')) \leftarrow P^4(x, x', y, y')$
- $P^4(a, a, a, a) \leftarrow$

# Terminaison de l'élimination de clauses copiantes

- Ne termine pas dans le cas général

- $\text{Prog} =$

- 1  $P(c(x, x)) \leftarrow P(x)$
- 2  $P(a) \leftarrow$

- $\text{uncopying}(P(c(x, x)) \leftarrow P(x))$
- $P(c(x, y)) \leftarrow P^2(x, y)$
- $P^2(c(x, x), c(y, y)) \leftarrow P^2(x, y)$
- $P^2(a, a) \leftarrow$
- $\text{uncopying}(P^2(c(x, x), c(y, y)) \leftarrow P^2(x, y))$
- $P^2(c(x, x'), c(y, y')) \leftarrow P^4(x, x', y, y')$
- $P^4(c(x, x), c(x', x'), c(y, y), c(y', y')) \leftarrow P^4(x, x', y, y')$
- $P^4(a, a, a, a) \leftarrow$
- et ainsi de suite

# Forcer la terminaison

- $\text{UncopyingLimit} = 2$

# Forcer la terminaison

- $\text{UncopyingLimit} = 2$
- $P^2(c(x, x'), c(y, y')) \leftarrow P^2(x, x'), P(y), P(y')$

# Forcer la terminaison

- $\text{UncopyingLimit} = 2$
- $P^2(c(x, x'), c(y, y')) \leftarrow P^2(x, x'), P(y), P(y')$
- $\{t \cdot t \cdot t' \cdot t'' \mid t, t', t'' \in L(P)\} \supset L(P^4)$

# Sommaire

1 Introduction

2 Préliminaires

3 Sur-approximations d'ensembles de descendants

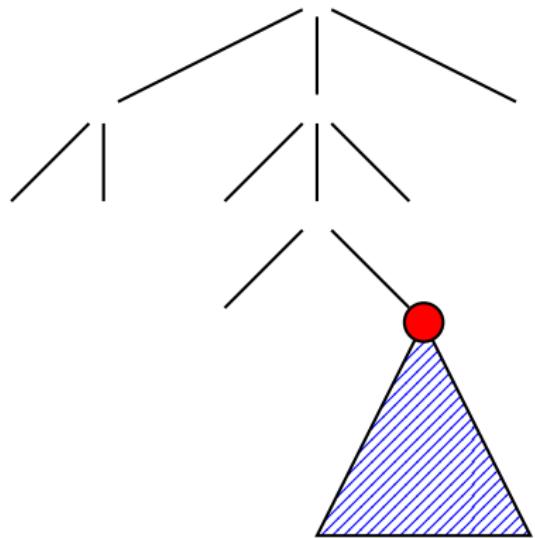
- Méthode de compléction
- Descendants innermost
- Élimination des clauses copiantes

4 Transformation de systèmes de réécriture avec stratégie

5 Conclusion

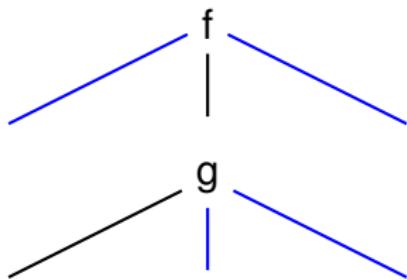
# Stratégies de réécriture

- *Innermost*



# Stratégies de réécriture

- *Innermost*
- *context-sensitive* (csTRS)



JFP, 2004, J. Giesl et A. Middeldorp, Transformation techniques for context-sensitive rewrite systems

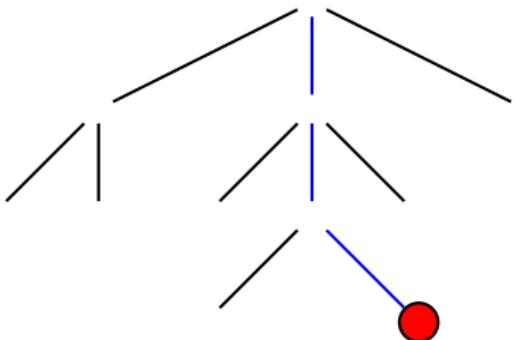
# Stratégies de réécriture

- *Innermost*
- *context-sensitive* (cSTRS)
- stratégie programmable
- Opérations élémentaires
- Opérations de contrôle de combinaison
- Opérations transversale

RTA, 2015, H. Cirstea, S. Lenglet et P.-E. Moreau, A faithful encoding of programmable strategies into term rewriting systems

# Stratégies de réécriture

- *Innermost*
- *context-sensitive* (csTRS)
- stratégie programmable
- *prefix-constrained* (pCTRS)



SCSS, 2017, N. Andrianarivelo, V. Pelletier et P. Réty, Transforming prefix-constrained or controlled rewrite systems

- $L : I \rightarrow r$

- $L : I \rightarrow r$
- $Dir(\Sigma) = \{\langle f, i \rangle \mid f \in \Sigma, 1 \leq i \leq arity(f)\}$

- $L : I \rightarrow r$
- $Dir(\Sigma) = \{\langle f, i \rangle \mid f \in \Sigma, 1 \leq i \leq arity(f)\}$
- $L$  automate de mots construit sur  $Dir(\Sigma)$

- $L : I \rightarrow r$
- $Dir(\Sigma) = \{\langle f, i \rangle \mid f \in \Sigma, 1 \leq i \leq arity(f)\}$
- $L$  automate de mots construit sur  $Dir(\Sigma)$

## Exemple

- $\Sigma = \{f^{\backslash 2}, g^{\backslash 2}, a^{\backslash 0}, b^{\backslash 0}\}$

- $L : I \rightarrow r$
- $Dir(\Sigma) = \{\langle f, i \rangle \mid f \in \Sigma, 1 \leq i \leq arity(f)\}$
- $L$  automate de mots construit sur  $Dir(\Sigma)$

## Exemple

- $\Sigma = \{f^{\backslash 2}, g^{\backslash 2}, a^{\backslash 0}, b^{\backslash 0}\}$
- $R = \{L : a \rightarrow b\}$

- $L : I \rightarrow r$
- $Dir(\Sigma) = \{\langle f, i \rangle \mid f \in \Sigma, 1 \leq i \leq arity(f)\}$
- $L$  automate de mots construit sur  $Dir(\Sigma)$

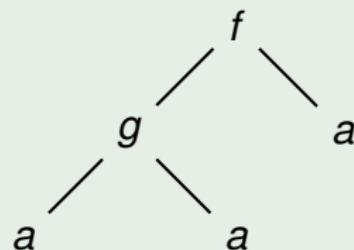
## Exemple

- $\Sigma = \{f^{\backslash 2}, g^{\backslash 2}, a^{\backslash 0}, b^{\backslash 0}\}$
- $R = \{L : a \rightarrow b\}$
- $L = (\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*$

- $L : I \rightarrow r$
- $Dir(\Sigma) = \{\langle f, i \rangle \mid f \in \Sigma, 1 \leq i \leq arity(f)\}$
- $L$  automate de mots construit sur  $Dir(\Sigma)$

## Exemple

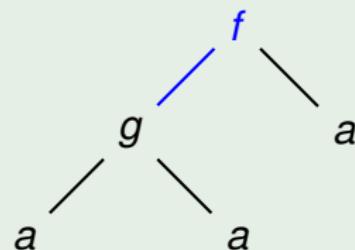
- $\Sigma = \{f^2, g^2, a^0, b^0\}$
- $R = \{L : a \rightarrow b\}$
- $L = (\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*$



- $L : I \rightarrow r$
- $Dir(\Sigma) = \{\langle f, i \rangle \mid f \in \Sigma, 1 \leq i \leq arity(f)\}$
- $L$  automate de mots construit sur  $Dir(\Sigma)$

## Exemple

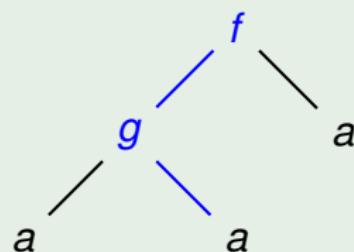
- $\Sigma = \{f^2, g^2, a^0, b^0\}$
- $R = \{L : a \rightarrow b\}$
- $L = (\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*$



- $L : I \rightarrow r$
- $Dir(\Sigma) = \{\langle f, i \rangle \mid f \in \Sigma, 1 \leq i \leq arity(f)\}$
- $L$  automate de mots construit sur  $Dir(\Sigma)$

## Exemple

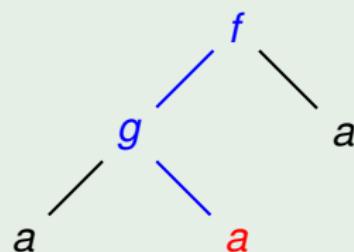
- $\Sigma = \{f^2, g^2, a^0, b^0\}$
- $R = \{L : a \rightarrow b\}$
- $L = (\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*$



- $L : I \rightarrow r$
- $Dir(\Sigma) = \{\langle f, i \rangle \mid f \in \Sigma, 1 \leq i \leq arity(f)\}$
- $L$  automate de mots construit sur  $Dir(\Sigma)$

## Exemple

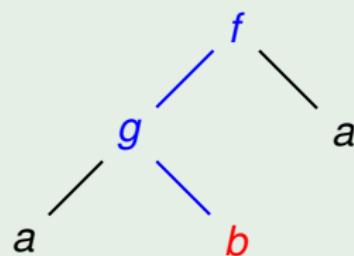
- $\Sigma = \{f^2, g^2, a^0, b^0\}$
- $R = \{L : a \rightarrow b\}$
- $L = (\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*$



- $L : I \rightarrow r$
- $Dir(\Sigma) = \{\langle f, i \rangle \mid f \in \Sigma, 1 \leq i \leq arity(f)\}$
- $L$  automate de mots construit sur  $Dir(\Sigma)$

## Exemple

- $\Sigma = \{f^2, g^2, a^0, b^0\}$
- $R = \{L : a \rightarrow b\}$
- $L = (\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*$



# Transformation de pCTRS

- $R' = R'_1 \cup R'_2 \cup R'_3 \cup R'_4$
- $R'_1$  : Règles d'amorçage
- $R'_2$  : Règles de parcourt de l'automate
- $R'_3$  : Règles de réécriture du pCTRS
- $R'_4$  : Règles de remonté du jeton

# Transformation de pCTRS

- $R' = R'_1 \cup R'_2 \cup R'_3 \cup R'_4$
- $R'_1 = \{ \text{top}(j(x)) \rightarrow \text{top}(q_l(x)) \mid q_l \in Q_l\}$
- $R'_2$  : Règles de parcourt de l'automate
- $R'_3$  : Règles de réécriture du pCTRS
- $R'_4$  : Règles de remonté du jeton

# Transformation de pCTRS

- $R' = R'_1 \cup R'_2 \cup R'_3 \cup R'_4$
- $R'_1 = \{ \text{top}(j(x)) \rightarrow \text{top}(q_l(x)) \mid q_l \in Q_l \}$
- $R'_2 = \{ q(f(x_1, \dots, x_n)) \rightarrow \delta(x_1, \dots, q'(x_i), \dots, x_n) \mid \delta = (q, \langle f, i \rangle, q') \in \Delta \}$
- $R'_3$  : Règles de réécriture du pCTRS
- $R'_4$  : Règles de remonté du jeton

# Transformation de pCTRS

- $R' = R'_1 \cup R'_2 \cup R'_3 \cup R'_4$
- $R'_1 = \{ \text{top}(j(x)) \rightarrow \text{top}(q_I(x)) \mid q_I \in Q_I \}$
- $R'_2 = \{ q(f(x_1, \dots, x_n)) \rightarrow \delta(x_1, \dots, q'(x_i), \dots, x_n) \mid \delta = (q, \langle f, i \rangle, q') \in \Delta \}$
- $R'_3 = \{ q_f(l_k) \rightarrow j(r_k) \mid q_f \in Q_f^k, (L_k : l_k \rightarrow r_k) \in R \}$
- $R'_4$  : Règles de remonté du jeton

# Transformation de pCTRS

- $R' = R'_1 \cup R'_2 \cup R'_3 \cup R'_4$
- $R'_1 = \{ \text{top}(j(x)) \rightarrow \text{top}(q_I(x)) \mid q_I \in Q_I \}$
- $R'_2 = \{ q(f(x_1, \dots, x_n)) \rightarrow \delta(x_1, \dots, q'(x_i), \dots, x_n) \mid \delta = (q, \langle f, i \rangle, q') \in \Delta \}$
- $R'_3 = \{ q_f(l_k) \rightarrow j(r_k) \mid q_f \in Q_f^k, (L_k : l_k \rightarrow r_k) \in R \}$
- $R'_4 = \{ \delta(x_1, \dots, j(x_i), \dots, x_n) \rightarrow j(f(x_1, \dots, x_i, \dots, x_n)) \mid \delta = (q, \langle f, i \rangle, q') \in \Delta \}$

# Transformation de pCTRS

- $R' = R'_1 \cup R'_2 \cup R'_3 \cup R'_4$
- $R'_1 = \{ \text{top}(j(x)) \rightarrow \text{top}(q_I(x)) \mid q_I \in Q_I \}$
- $R'_2 = \{ q(f(x_1, \dots, x_n)) \rightarrow \delta(x_1, \dots, q'(x_i), \dots, x_n) \mid \delta = (q, \langle f, i \rangle, q') \in \Delta \}$
- $R'_3 = \{ q_f(l_k) \rightarrow j(r_k) \mid q_f \in Q_f^k, (L_k : l_k \rightarrow r_k) \in R \}$
- $R'_4 = \{ \delta(x_1, \dots, j(x_i), \dots, x_n) \rightarrow j(f(x_1, \dots, x_i, \dots, x_n)) \mid \delta = (q, \langle f, i \rangle, q') \in \Delta \}$

Théorème : [SCSS, 2017, N. Andrianarivelo, V. Pelletier et P. Réty]

Soit  $t \in T(\Sigma)$ . On a  $t \rightarrow_{R_{pc}}^* t'$  si et seulement si  $\text{top}(j(t)) \rightarrow_{R'}^* \text{top}(j(t'))$ .

# Transformation de pCTRS

- $R' = R'_1 \cup R'_2 \cup R'_3 \cup R'_4$
- $R'_1 = \{ \text{top}(j(x)) \rightarrow \text{top}(q_I(x)) \mid q_I \in Q_I \}$
- $R'_2 = \{ q(f(x_1, \dots, x_n)) \rightarrow \delta(x_1, \dots, q'(x_i), \dots, x_n) \mid \delta = (q, \langle f, i \rangle, q') \in \Delta \}$
- $R'_3 = \{ q_f(l_k) \rightarrow j(r_k) \mid q_f \in Q_f^k, (L_k : l_k \rightarrow r_k) \in R \}$
- $R'_4 = \{ \delta(x_1, \dots, j(x_i), \dots, x_n) \rightarrow j(f(x_1, \dots, x_i, \dots, x_n)) \mid \delta = (q, \langle f, i \rangle, q') \in \Delta \}$

Théorème : [SCSS, 2017, N. Andrianarivelo, V. Pelletier et P. Réty]

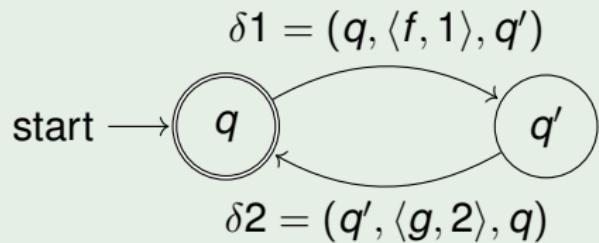
Soit  $t \in T(\Sigma)$ . On a  $t \xrightarrow{*_{R_{pc}}} t'$  si et seulement si  $\text{top}(j(t)) \xrightarrow{*_{R'}} \text{top}(j(t'))$ .

Théorème : [SCSS, 2017, N. Andrianarivelo, V. Pelletier et P. Réty]

Le pCTRS  $R$  est terminant sur  $\Sigma$  si et seulement si  $R'$  est terminant sur  $\Sigma'$ .

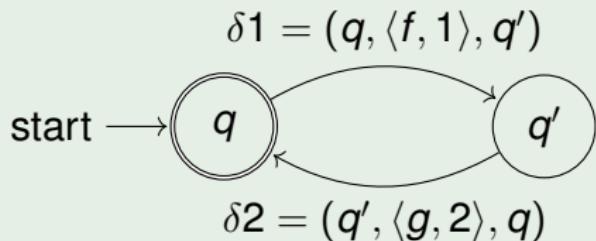
# Exemple

- $\Sigma = \{f^2, g^2, a^0, b^0\}$
- $R = \{(\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*: a \rightarrow b\}$
- $Dir(\Sigma) = \{\langle f, 1 \rangle, \langle f, 2 \rangle, \langle g, 1 \rangle, \langle g, 2 \rangle\}$



# Exemple

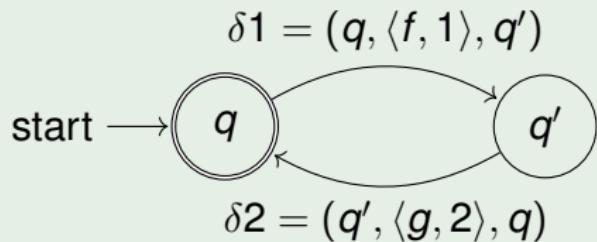
- $\Sigma = \{f^{\setminus 2}, g^{\setminus 2}, a^{\setminus 0}, b^{\setminus 0}\}$
- $R = \{(\langle f, 1 \rangle. \langle g, 2 \rangle)^*: a \rightarrow b\}$
- $Dir(\Sigma) = \{\langle f, 1 \rangle, \langle f, 2 \rangle, \langle g, 1 \rangle, \langle g, 2 \rangle\}$



- $R'_1 = \{top(j(x)) \rightarrow top(q(x))\}$

# Exemple

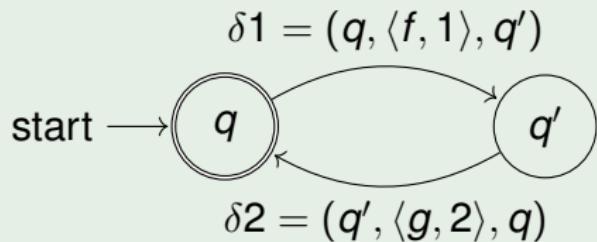
- $\Sigma = \{f^{\backslash 2}, g^{\backslash 2}, a^{\backslash 0}, b^{\backslash 0}\}$
- $R = \{(\langle f, 1 \rangle. \langle g, 2 \rangle)^*: a \rightarrow b\}$
- $Dir(\Sigma) = \{\langle f, 1 \rangle, \langle f, 2 \rangle, \langle g, 1 \rangle, \langle g, 2 \rangle\}$



- $R'_1 = \{top(j(x)) \rightarrow top(q(x))\}$
- $R'_2 = \{q(f(x, y)) \rightarrow \delta_1(q'(x), y), q'(g(x, y)) \rightarrow \delta_2(x, q(y))\}$

# Exemple

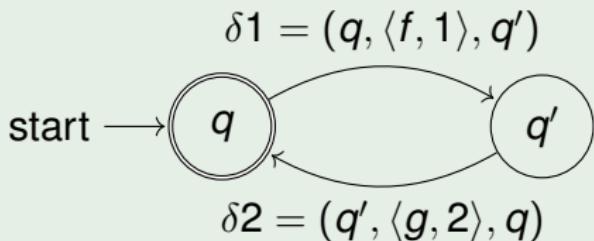
- $\Sigma = \{f^{\backslash 2}, g^{\backslash 2}, a^{\backslash 0}, b^{\backslash 0}\}$
- $R = \{(\langle f, 1 \rangle. \langle g, 2 \rangle)^*: a \rightarrow b\}$
- $Dir(\Sigma) = \{\langle f, 1 \rangle, \langle f, 2 \rangle, \langle g, 1 \rangle, \langle g, 2 \rangle\}$



- $R'_1 = \{top(j(x)) \rightarrow top(q(x))\}$
- $R'_2 = \{q(f(x, y)) \rightarrow \delta_1(q'(x), y), q'(g(x, y)) \rightarrow \delta_2(x, q(y))\}$
- $R'_3 = \{q(a) \rightarrow j(b)\}$

# Exemple

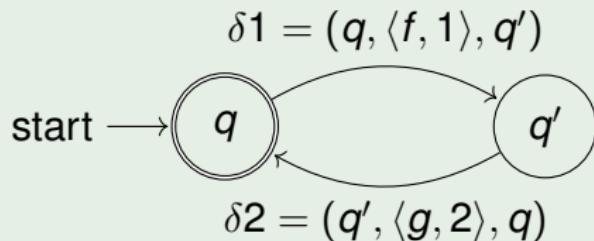
- $\Sigma = \{f^{\backslash 2}, g^{\backslash 2}, a^{\backslash 0}, b^{\backslash 0}\}$
- $R = \{(\langle f, 1 \rangle. \langle g, 2 \rangle)^*: a \rightarrow b\}$
- $Dir(\Sigma) = \{\langle f, 1 \rangle, \langle f, 2 \rangle, \langle g, 1 \rangle, \langle g, 2 \rangle\}$



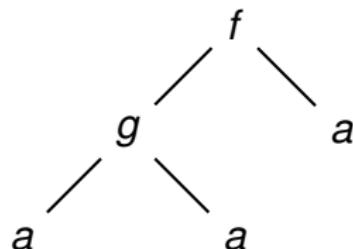
- $R'_1 = \{top(j(x)) \rightarrow top(q(x))\}$
- $R'_2 = \{q(f(x, y)) \rightarrow \delta_1(q'(x), y), q'(g(x, y)) \rightarrow \delta_2(x, q(y))\}$
- $R'_3 = \{q(a) \rightarrow j(b)\}$
- $R'_4 = \{\delta_1(j(x), y) \rightarrow j(f(x, y)), \delta_2(x, j(y)) \rightarrow j(g(x, y))\}$

# Exemple

- $\Sigma = \{f^2, g^2, a^0, b^0\}$
- $R = \{(\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*: a \rightarrow b\}$
- $Dir(\Sigma) = \{\langle f, 1 \rangle, \langle f, 2 \rangle, \langle g, 1 \rangle, \langle g, 2 \rangle\}$

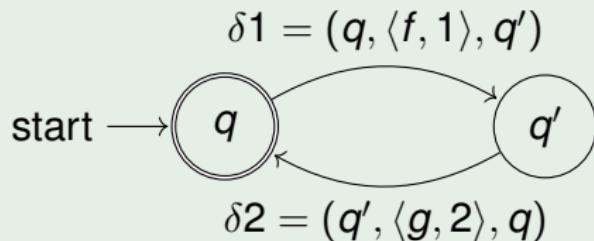


- $R'_1 = \{top(j(x)) \rightarrow top(q(x))\}$
- $R'_2 = \{q(f(x, y)) \rightarrow \delta_1(q'(x), y), q'(g(x, y)) \rightarrow \delta_2(x, q(y))\}$
- $R'_3 = \{q(a) \rightarrow j(b)\}$
- $R'_4 = \{\delta_1(j(x), y) \rightarrow j(f(x, y)), \delta_2(x, j(y)) \rightarrow j(g(x, y))\}$

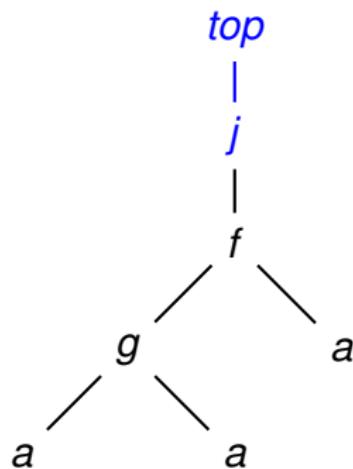


# Exemple

- $\Sigma = \{f^{\backslash 2}, g^{\backslash 2}, a^{\backslash 0}, b^{\backslash 0}\}$
- $R = \{(\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*: a \rightarrow b\}$
- $Dir(\Sigma) = \{\langle f, 1 \rangle, \langle f, 2 \rangle, \langle g, 1 \rangle, \langle g, 2 \rangle\}$

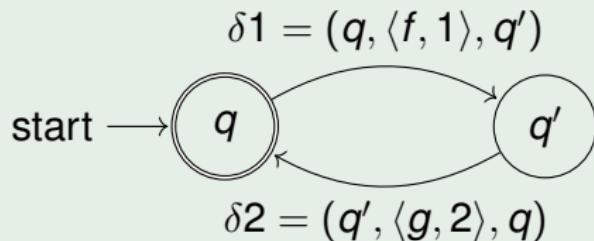


- $R'_1 = \{top(j(x)) \rightarrow top(q(x))\}$
- $R'_2 = \{q(f(x, y)) \rightarrow \delta_1(q'(x), y), q'(g(x, y)) \rightarrow \delta_2(x, q(y))\}$
- $R'_3 = \{q(a) \rightarrow j(b)\}$
- $R'_4 = \{\delta_1(j(x), y) \rightarrow j(f(x, y)), \delta_2(x, j(y)) \rightarrow j(g(x, y))\}$

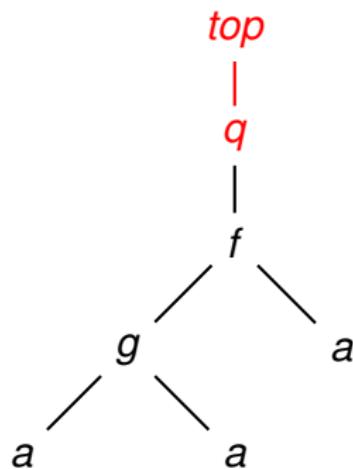


# Exemple

- $\Sigma = \{f^{\backslash 2}, g^{\backslash 2}, a^{\backslash 0}, b^{\backslash 0}\}$
- $R = \{(\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*: a \rightarrow b\}$
- $Dir(\Sigma) = \{\langle f, 1 \rangle, \langle f, 2 \rangle, \langle g, 1 \rangle, \langle g, 2 \rangle\}$

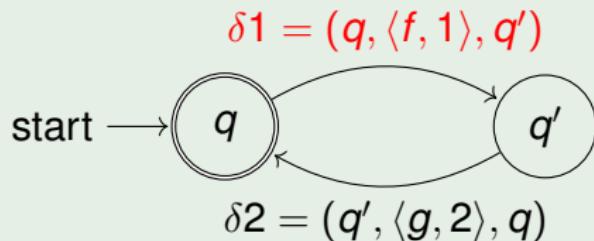


- $R'_1 = \{top(j(x)) \rightarrow top(q(x))\}$
- $R'_2 = \{q(f(x, y)) \rightarrow \delta_1(q'(x), y), q'(g(x, y)) \rightarrow \delta_2(x, q(y))\}$
- $R'_3 = \{q(a) \rightarrow j(b)\}$
- $R'_4 = \{\delta_1(j(x), y) \rightarrow j(f(x, y)), \delta_2(x, j(y)) \rightarrow j(g(x, y))\}$

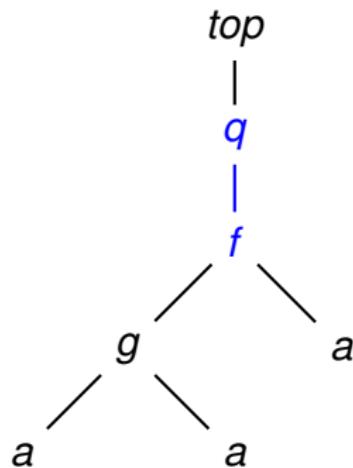


# Exemple

- $\Sigma = \{f^{\setminus 2}, g^{\setminus 2}, a^{\setminus 0}, b^{\setminus 0}\}$
- $R = \{(\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*: a \rightarrow b\}$
- $Dir(\Sigma) = \{\langle f, 1 \rangle, \langle f, 2 \rangle, \langle g, 1 \rangle, \langle g, 2 \rangle\}$

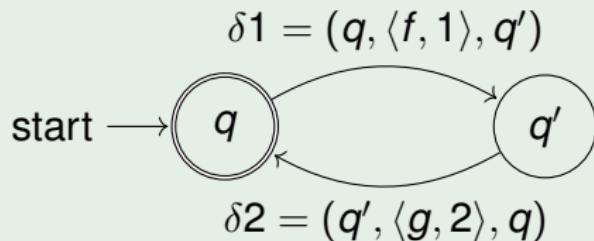


- $R'_1 = \{top(j(x)) \rightarrow top(q(x))\}$
- $R'_2 = \{q(f(x, y)) \rightarrow \delta_1(q'(x), y), q'(g(x, y)) \rightarrow \delta_2(x, q(y))\}$
- $R'_3 = \{q(a) \rightarrow j(b)\}$
- $R'_4 = \{\delta_1(j(x), y) \rightarrow j(f(x, y)), \delta_2(x, j(y)) \rightarrow j(g(x, y))\}$

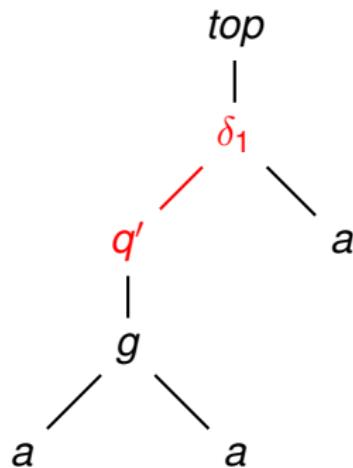


# Exemple

- $\Sigma = \{f^{\backslash 2}, g^{\backslash 2}, a^{\backslash 0}, b^{\backslash 0}\}$
- $R = \{(\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*: a \rightarrow b\}$
- $Dir(\Sigma) = \{\langle f, 1 \rangle, \langle f, 2 \rangle, \langle g, 1 \rangle, \langle g, 2 \rangle\}$

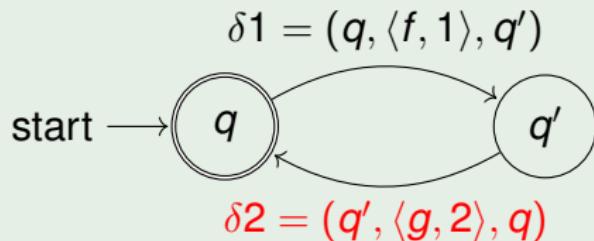


- $R'_1 = \{top(j(x)) \rightarrow top(q(x))\}$
- $R'_2 = \{q(f(x, y)) \rightarrow \delta_1(q'(x), y), q'(g(x, y)) \rightarrow \delta_2(x, q(y))\}$
- $R'_3 = \{q(a) \rightarrow j(b)\}$
- $R'_4 = \{\delta_1(j(x), y) \rightarrow j(f(x, y)), \delta_2(x, j(y)) \rightarrow j(g(x, y))\}$

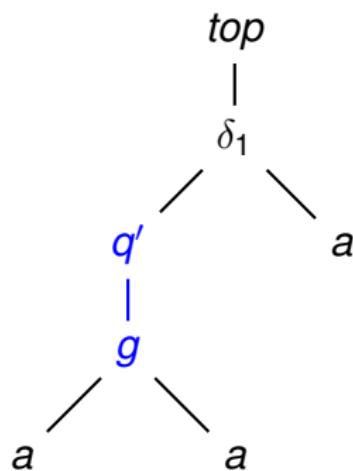


# Exemple

- $\Sigma = \{f^{\backslash 2}, g^{\backslash 2}, a^{\backslash 0}, b^{\backslash 0}\}$
- $R = \{(\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*: a \rightarrow b\}$
- $Dir(\Sigma) = \{\langle f, 1 \rangle, \langle f, 2 \rangle, \langle g, 1 \rangle, \langle g, 2 \rangle\}$

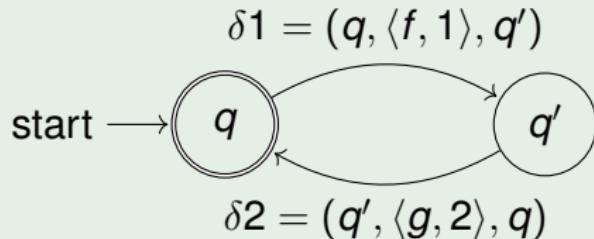


- $R'_1 = \{top(j(x)) \rightarrow top(q(x))\}$
- $R'_2 = \{q(f(x, y)) \rightarrow \delta_1(q'(x), y),$   
 $q'(g(x, y)) \rightarrow \delta_2(x, q(y))\}$
- $R'_3 = \{q(a) \rightarrow j(b)\}$
- $R'_4 = \{\delta_1(j(x), y) \rightarrow j(f(x, y)),$   
 $\delta_2(x, j(y)) \rightarrow j(g(x, y))\}$

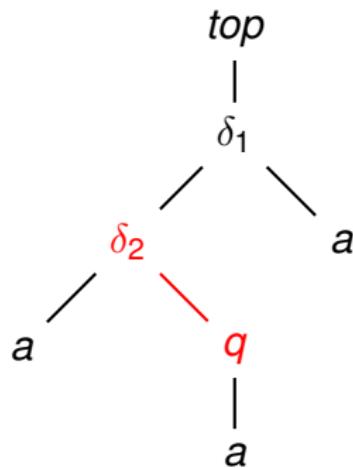


# Exemple

- $\Sigma = \{f^{\backslash 2}, g^{\backslash 2}, a^{\backslash 0}, b^{\backslash 0}\}$
- $R = \{(\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*: a \rightarrow b\}$
- $Dir(\Sigma) = \{\langle f, 1 \rangle, \langle f, 2 \rangle, \langle g, 1 \rangle, \langle g, 2 \rangle\}$

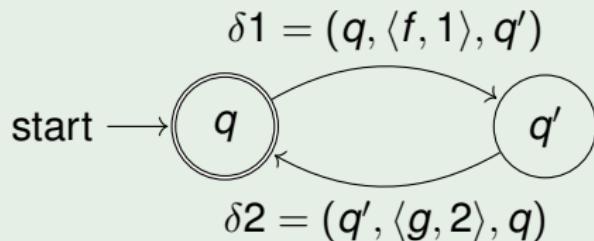


- $R'_1 = \{top(j(x)) \rightarrow top(q(x))\}$
- $R'_2 = \{q(f(x, y)) \rightarrow \delta_1(q'(x), y), q'(g(x, y)) \rightarrow \delta_2(x, q(y))\}$
- $R'_3 = \{q(a) \rightarrow j(b)\}$
- $R'_4 = \{\delta_1(j(x), y) \rightarrow j(f(x, y)), \delta_2(x, j(y)) \rightarrow j(g(x, y))\}$

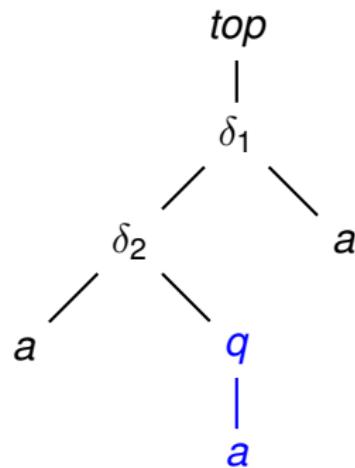


# Exemple

- $\Sigma = \{f^{\backslash 2}, g^{\backslash 2}, a^{\backslash 0}, b^{\backslash 0}\}$
- $R = \{(\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*: a \rightarrow b\}$
- $Dir(\Sigma) = \{\langle f, 1 \rangle, \langle f, 2 \rangle, \langle g, 1 \rangle, \langle g, 2 \rangle\}$

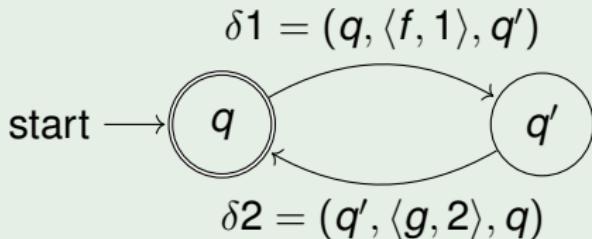


- $R'_1 = \{top(j(x)) \rightarrow top(q(x))\}$
- $R'_2 = \{q(f(x, y)) \rightarrow \delta_1(q'(x), y), q'(g(x, y)) \rightarrow \delta_2(x, q(y))\}$
- $R'_3 = \{q(a) \rightarrow j(b)\}$
- $R'_4 = \{\delta_1(j(x), y) \rightarrow j(f(x, y)), \delta_2(x, j(y)) \rightarrow j(g(x, y))\}$

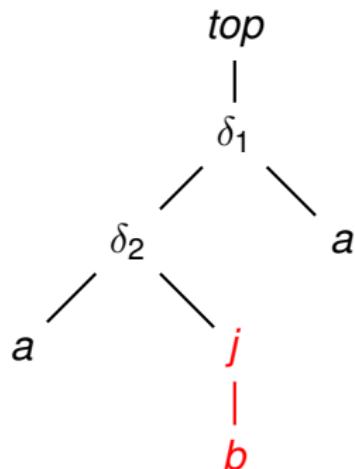


# Exemple

- $\Sigma = \{f^{\backslash 2}, g^{\backslash 2}, a^{\backslash 0}, b^{\backslash 0}\}$
- $R = \{(\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*: a \rightarrow b\}$
- $Dir(\Sigma) = \{\langle f, 1 \rangle, \langle f, 2 \rangle, \langle g, 1 \rangle, \langle g, 2 \rangle\}$

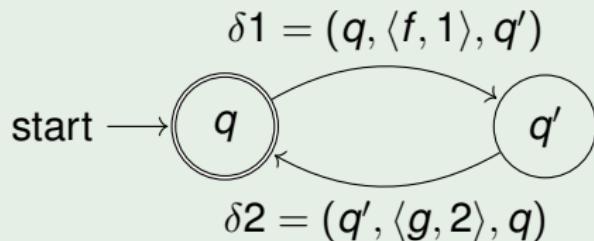


- $R'_1 = \{top(j(x)) \rightarrow top(q(x))\}$
- $R'_2 = \{q(f(x, y)) \rightarrow \delta_1(q'(x), y), q'(g(x, y)) \rightarrow \delta_2(x, q(y))\}$
- $R'_3 = \{q(a) \rightarrow j(b)\}$
- $R'_4 = \{\delta_1(j(x), y) \rightarrow j(f(x, y)), \delta_2(x, j(y)) \rightarrow j(g(x, y))\}$

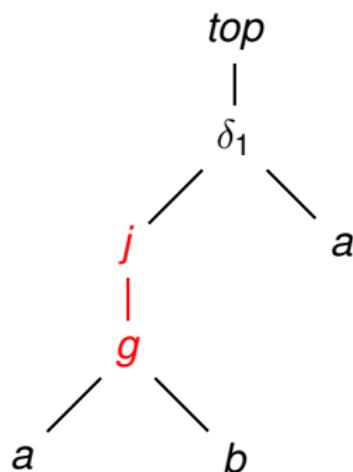


# Exemple

- $\Sigma = \{f^{\backslash 2}, g^{\backslash 2}, a^{\backslash 0}, b^{\backslash 0}\}$
- $R = \{(\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*: a \rightarrow b\}$
- $Dir(\Sigma) = \{\langle f, 1 \rangle, \langle f, 2 \rangle, \langle g, 1 \rangle, \langle g, 2 \rangle\}$

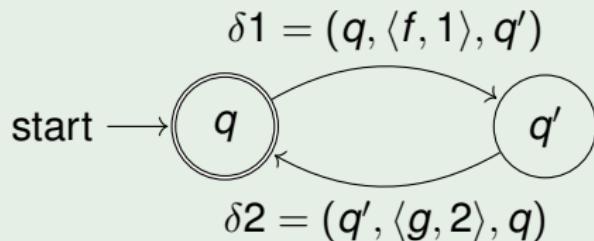


- $R'_1 = \{top(j(x)) \rightarrow top(q(x))\}$
- $R'_2 = \{q(f(x, y)) \rightarrow \delta_1(q'(x), y), q'(g(x, y)) \rightarrow \delta_2(x, q(y))\}$
- $R'_3 = \{q(a) \rightarrow j(b)\}$
- $R'_4 = \{\delta_1(j(x), y) \rightarrow j(f(x, y)), \delta_2(x, j(y)) \rightarrow j(g(x, y))\}$

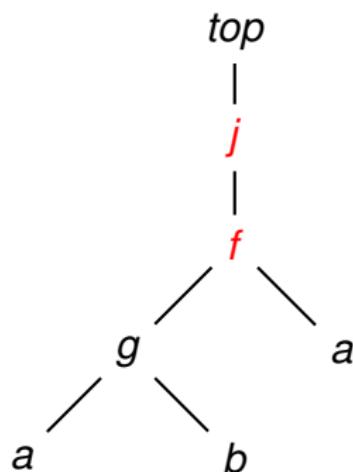


# Exemple

- $\Sigma = \{f^{\backslash 2}, g^{\backslash 2}, a^{\backslash 0}, b^{\backslash 0}\}$
- $R = \{(\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*: a \rightarrow b\}$
- $Dir(\Sigma) = \{\langle f, 1 \rangle, \langle f, 2 \rangle, \langle g, 1 \rangle, \langle g, 2 \rangle\}$

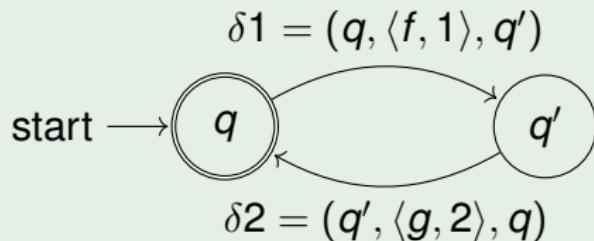


- $R'_1 = \{top(j(x)) \rightarrow top(q(x))\}$
- $R'_2 = \{q(f(x, y)) \rightarrow \delta_1(q'(x), y),$   
 $q'(g(x, y)) \rightarrow \delta_2(x, q(y))\}$
- $R'_3 = \{q(a) \rightarrow j(b)\}$
- $R'_4 = \{\delta_1(j(x), y) \rightarrow j(f(x, y)),$   
 $\delta_2(x, j(y)) \rightarrow j(g(x, y))\}$

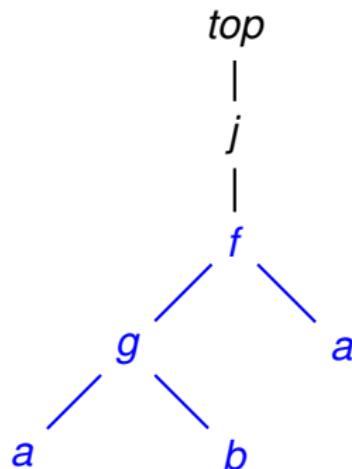


# Exemple

- $\Sigma = \{f^{\backslash 2}, g^{\backslash 2}, a^{\backslash 0}, b^{\backslash 0}\}$
- $R = \{(\langle f, 1 \rangle \cdot \langle g, 2 \rangle)^*: a \rightarrow b\}$
- $Dir(\Sigma) = \{\langle f, 1 \rangle, \langle f, 2 \rangle, \langle g, 1 \rangle, \langle g, 2 \rangle\}$



- $R'_1 = \{top(j(x)) \rightarrow top(q(x))\}$
- $R'_2 = \{q(f(x, y)) \rightarrow \delta_1(q'(x), y),$   
 $q'(g(x, y)) \rightarrow \delta_2(x, q(y))\}$
- $R'_3 = \{q(a) \rightarrow j(b)\}$
- $R'_4 = \{\delta_1(j(x), y) \rightarrow j(f(x, y)),$   
 $\delta_2(x, j(y)) \rightarrow j(g(x, y))\}$



# Sommaire

1 Introduction

2 Préliminaires

3 Sur-approximations d'ensembles de descendants

- Méthode de compléction
- Descendants innermost
- Élimination des clauses copiantes

4 Transformation de systèmes de réécriture avec stratégie

5 Conclusion

# Conclusion

- Calcul de sur-approximations non régulières

# Conclusion

- Calcul de sur-approximations non régulières
  - Descendants innermost

# Conclusion

- Calcul de sur-approximations non régulières
  - Descendants innermost
  - Élimination de clauses copiantes

# Conclusion

- Calcul de sur-approximations non régulières
  - Descendants innermost
  - Élimination de clauses copiantes
- Transformation de pCTRS

# Conclusion

- Calcul de sur-approximations non régulières
  - Descendants innermost
  - Élimination de clauses copiantes
- Transformation de pCTRS
- Transformation d'un programme logique en cs-programme avec sur-approximation

- Calcul de sur-approximations non régulières

# Perspectives

- Calcul de sur-approximations non régulières
  - Améliorer la précision des approximations

- Calcul de sur-approximations non régulières
  - Améliorer la précision des approximations
  - Développer des heuristiques

- Calcul de sur-approximations non régulières
  - Améliorer la précision des approximations
  - Développer des heuristiques
  - Porter les résultats existants sur les réguliers

- Calcul de sur-approximations non régulières
  - Améliorer la précision des approximations
  - Développer des heuristiques
  - Porter les résultats existants sur les réguliers
- Transformation de pCTRS

- Calcul de sur-approximations non régulières
  - Améliorer la précision des approximations
  - Développer des heuristiques
  - Porter les résultats existants sur les réguliers
- Transformation de pCTRS
  - Explorer d'autres stratégies

- Calcul de sur-approximations non régulières
  - Améliorer la précision des approximations
  - Développer des heuristiques
  - Porter les résultats existants sur les réguliers
- Transformation de pCTRS
  - Explorer d'autres stratégies
- Transformation d'un programme logique en cs-programme avec sur-approximation

- Calcul de sur-approximations non régulières
  - Améliorer la précision des approximations
  - Développer des heuristiques
  - Porter les résultats existants sur les réguliers
- Transformation de pCTRS
  - Explorer d'autres stratégies
- Transformation d'un programme logique en cs-programme avec sur-approximation
  - Améliorer la précision des approximations

# Remarques et questions

Merci de votre attention