Interprétation Abstraite et Analyse Statique

Patrick COUSOT
École Normale Supérieure
45 rue d'Ulm
75230 Paris cedex 05

mailto:Patrick.Cousot@ens.fr
http://www.di.ens.fr/~cousot

10ème anniversaire du LIX 26 mai 1999

1. Motivations

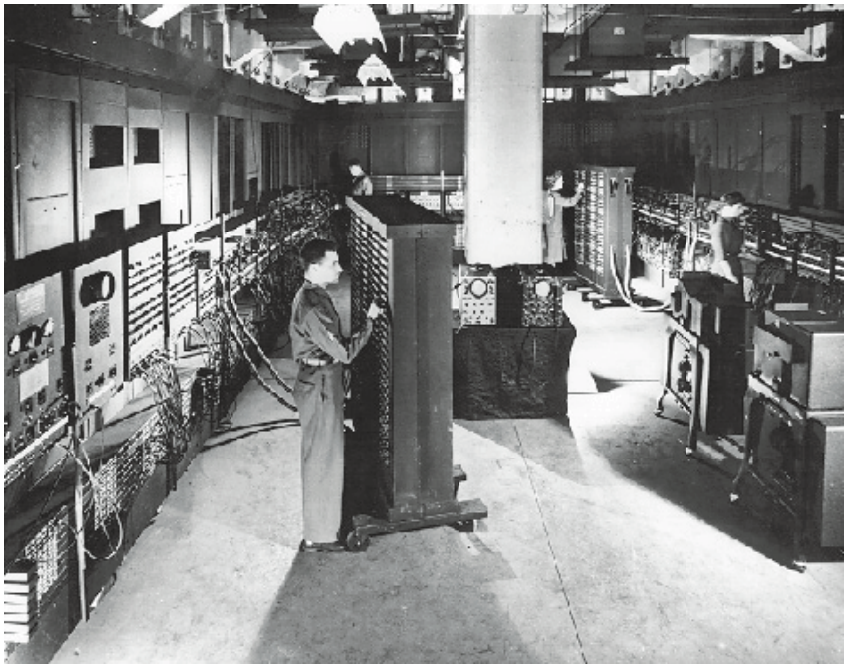
Quelle est la préoccupation fondamentale des informaticiens ?

Quelle est la préoccupation fondamentale des informaticiens ?

La production de logiciels fiables, leur maintenance et leur adaptation au fil du temps (20 à 30 ans au minimum).

Changement d'échelle des matériels informatiques

Ces 25 dernières années, le matériel informatique a vu ses performances multipliées par 10^4 à 10^6 ;



ENIAC (5000 flops)

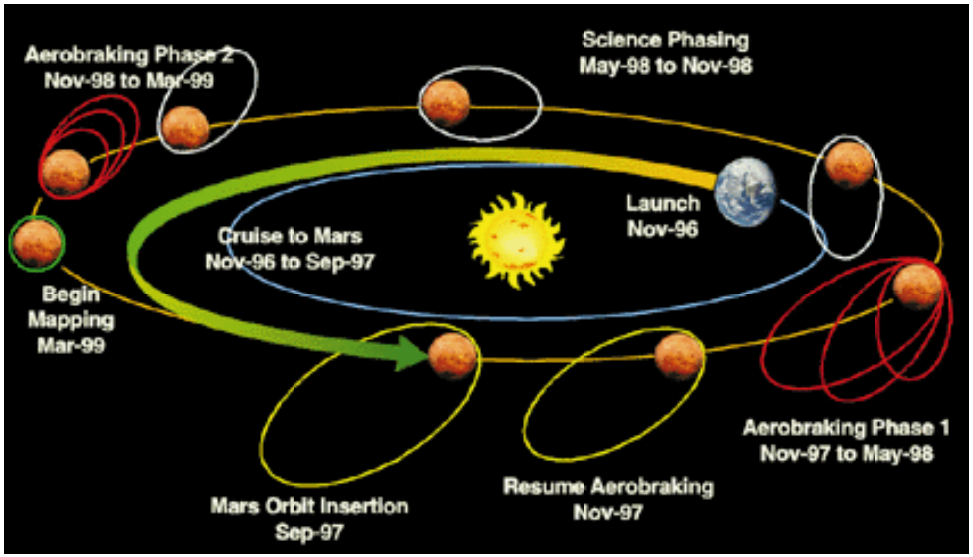
Intel/Sandia Teraflops System (10^{12} flops)

Révolution informatique

Un rapport 10^6 est indicatif d'une **révolution** :

– énergie : centrale nucléaire / esclave romain ;

– transport : terre — mars / Paris — Nice



Changement d'échelle dans les logiciels informatiques

– La taille des programmes exécutés par ces ordinateurs a cru dans des proportions similaires ;

Changement d'échelle dans les logiciels informatiques

– La taille des programmes exécutés par ces ordinateurs a cru dans des proportions similaires ;

– **Exemple** (éditeur de texte moderne, grand public) :

  – > 1 700 000 lignes de C [1] ;

  – 20 000 procédures ;

  – 400 fichiers ;

  – > 15 ans de développement.

1. au moins 3 mois de lecture à temps plein, 35 heures/semaine !

Bogues 

– Les erreurs dans les logiciels
  – qu'ils soient prévues (bogue de l'an 2000)
  – ou imprévues (échec du vol 5.01 d'Ariane)

  sont fréquentes ;

Bogues 

– Les erreurs dans les logiciels
  – qu'ils soient prévues (bogue de l'an 2000)
  – ou imprévues (échec du vol 5.01 d'Ariane)

  sont fréquentes ;
– Les bogues sont très difficiles à découvrir dans des logiciels énormes ;

Bogues

– Les erreurs dans les logiciels
  – qu'ils soient prévues (bogue de l'an 2000)
  – ou imprévues (échec du vol 5.01 d'Ariane)

  sont fréquentes ;
– Les bogues sont très difficiles à découvrir dans des logiciels énormes ;
– Les bogues peuvent avoir des conséquences catastrophiques très coûteuses ou inadmissibles (logiciels embarqués dans les transports) ;

Capacités des informaticiens

– Les capacités des informaticiens restent essentiellement inchangées au fil du temps ;

Capacités des informaticiens

– Les capacités des informaticiens restent essentiellement inchangées au fil du temps ;

– Les tailles des équipes de conception et de maintenance ne peuvent évoluer dans de grandes proportions ;

Capacités des informaticiens

– Les capacités des informaticiens restent essentiellement inchangées au fil du temps ;

– Les tailles des équipes de conception et de maintenance ne peuvent évoluer dans de grandes proportions ;

– Les méthodes manuelles classiques de validation des logiciels (revues de code, simulations, tests) ne passent pas à l'échelle.

Responsabilité des informaticiens

– Paradoxalement, la responsabilité des informaticiens n'est pas engagée (comparer à l'industrie automobile, l'aviation) ;

Responsabilité des informaticiens

– Paradoxalement, la responsabilité des informaticiens n'est pas engagée (comparer à l'industrie automobile, l'aviation) ;
– L'échec informatique peut devenir une problème de société important (peur collective ? nouvelles réglementations ?) ;

Responsabilité des informaticiens

– Paradoxalement, la responsabilité des informaticiens n'est pas engagée (comparer à l'industrie automobile, l'aviation) ;
– L'échec informatique peut devenir une problème de société important (peur collective ? nouvelles réglementations ?) ;

$\Longrightarrow$ Nécessité d'élargir la panoplie de méthodes et outils utilisés pour lutter contre les erreurs logicielles.

Idée

Utiliser l'ordinateur pour trouver les erreurs de programmation.

Question (extrêmement difficile)

Comment programmer les ordinateurs pour qu'ils analysent le travail qu'on leur donne à faire, avant qu'ils ne le fassent effectivement ?

Exemple simpliste : recette de cuisine

Recette de l'œuf coque :

– Prendre un œuf frais au réfrigérateur ;

– Le plonger dans l'eau bouillante salée ;

– Le retirer après 4 mn.

Exemple simpliste : recette de cuisine

Recette de l'œuf coque :

– Prendre un œuf frais au réfrigérateur ;

– Le plonger dans l'eau bouillante salée ;

– Le retirer après 4 h. 

Exemple simpliste : recette de cuisine

Recette de l'œuf coque :

– Prendre un œuf frais au réfrigérateur ;

– Le plonger dans l'eau bouillante salée ;

– Le retirer après 4 h. 

N'importe quel cuisinier est capable de trouver l'erreur avant de réaliser la recette !

Exemple simpliste : recette de cuisine

Recette de l'œuf coque :

– Prendre un œuf frais au réfrigérateur ;

– Le plonger dans l'eau bouillante salée ;

– Le retirer après 4 h. 

N'importe quel cuisinier est capable de trouver l'erreur avant de réaliser la recette !

Pourquoi pas les ordinateurs ?

Exemple simpliste : recette de cuisine

Recette de l'œuf coque :

– Prendre un œuf frais au réfrigérateur ;

– Le plonger dans l'eau bouillante salée ;

– Le retirer après 4 h. 

N'importe quel cuisinier est capable de trouver l'erreur avant de réaliser la recette !

Pourquoi pas les ordinateurs ?

Que faire ?

2. Interprétation abstraite

Exposé introductif

– Quatre notions à introduire :
  – Sémantique ,
  – Indécidabilité ,
  – Interprétation abstraite ,
  – Analyse statique ;

Exposé introductif informel

– Quatre notions à introduire :
  – Sémantique ,
  – Indécidabilité ,
  – Interprétation abstraite ,
  – Analyse statique ;
– Explication complètement informelle , évitant tout formalisme ;

Exposé introductif informel

– Quatre notions à introduire :
  – Sémantique ,
  – Indécidabilité ,
  – Interprétation abstraite ,
  – Analyse statique ;
– Explication complètement informelle , évitant tout formalisme ;
– Illustrée par des travaux effectués au LIX depuis sa création et les thèses soutenues (ou à soutenir) à l'École polytechnique.

2.1 Sémantique & indécidabilité

Il s'agit donc d'expliquer la sémantique, par exemple :

Syntaxe :

$$
\begin{array}{rcl}
\mathtt{x},\mathtt{f} \in \mathbb{X} & : & \text{variables} \\
e \in \mathbb{E} & : & \text{expressions} \\
e & ::= & \mathtt{x} \mid \boldsymbol{\lambda}\mathtt{x}\cdot e \mid e_1(e_2) \mid \\
 & & \boldsymbol{\mu}\mathtt{f}\cdot\boldsymbol{\lambda}\mathtt{x}\cdot e \mid e_1 - e_2 \mid \\
 & & \mathbf{1} \mid (e_1\,\mathbf{?}\,e_2\,\mathbf{:}\,e_3)
\end{array}
$$

Domaines sémantiques :

$$
\begin{array}{rcll}
\mathbb{W} & \triangleq & \{\omega\} & \text{erreur} \\
\mathsf{z} \in \mathbb{Z} & & & \text{entiers} \\
\mathsf{u},\mathsf{f},\varphi \in \mathbb{U} & \cong & \mathbb{W}_\bot \oplus \mathbb{Z}_\bot \oplus [\mathbb{U} \mapsto \mathbb{U}]_\bot & \text{valeurs} \\
\mathsf{R} \in \mathbb{R} & \triangleq & \mathbb{X} \mapsto \mathbb{U} & \text{environnements} \\
\phi \in \mathbb{S} & \triangleq & \mathbb{R} \mapsto \mathbb{U} & \text{domaine sémantique}
\end{array}
$$

Sémantique :

$$
\begin{array}{rcl}
\mathbf{S}[\![\mathtt{x}]\!] & \triangleq & \Lambda\mathsf{R}\bullet\mathsf{R}(\mathtt{x}) \\
\mathbf{S}[\![\boldsymbol{\lambda}\mathtt{x}\cdot e]\!] & \triangleq & \Lambda\mathsf{R}\bullet\uparrow(\Lambda\mathsf{u}\bullet(\mathsf{u}=\bot\vee\mathsf{u}=\Omega\;?\;\mathsf{u}\mid \\
 & & \qquad\qquad \mathbf{S}[\![e]\!]\mathsf{R}[\mathtt{x}\leftarrow\mathsf{u}])) :: [\mathbb{U}\mapsto\mathbb{U}]_\bot \\
\mathbf{S}[\![e_1(e_2)]\!] & \triangleq & \Lambda\mathsf{R}\bullet(\mathbf{S}[\![e_1]\!]\mathsf{R}=\bot\vee\mathbf{S}[\![e_2]\!]\mathsf{R}=\bot\;?\;\bot\mid \\
 & & \mathbf{S}[\![e_1]\!]\mathsf{R}=\mathsf{f}::[\mathbb{U}\mapsto\mathbb{U}]_\bot\;?\downarrow(\mathsf{f})(\mathbf{S}[\![e_2]\!]\mathsf{R})\mid\Omega) \\
\mathbf{S}[\![\boldsymbol{\mu}\mathtt{f}\cdot\boldsymbol{\lambda}\mathtt{x}\cdot e]\!] & \triangleq & \Lambda\mathsf{R}\bullet\mathbf{lfp}^{\sqsubseteq}_{\uparrow(\Lambda\mathsf{u}\bullet\bot)::[\mathbb{U}\mapsto\mathbb{U}]_\bot}\;\Lambda\varphi\bullet\mathbf{S}[\![\boldsymbol{\lambda}\mathtt{x}\cdot e]\!]\mathsf{R}[\mathtt{f}\leftarrow\varphi] \\
\mathbf{S}[\![\mathbf{1}]\!] & \triangleq & \Lambda\mathsf{R}\bullet\uparrow(1)::\mathbb{Z}_\bot \\
\mathbf{S}[\![e_1-e_2]\!] & \triangleq & \Lambda\mathsf{R}\bullet(\mathbf{S}[\![e_1]\!]\mathsf{R}=\bot\vee\mathbf{S}[\![e_2]\!]\mathsf{R}=\bot\;?\;\bot\mid \\
 & & \mathbf{S}[\![e_1]\!]\mathsf{R}=\mathsf{z}_1::\mathbb{Z}_\bot\wedge\mathbf{S}[\![e_2]\!]\mathsf{R}=\mathsf{z}_2::\mathbb{Z}_\bot\;? \\
 & & \uparrow(\downarrow(\mathsf{z}_1)-\downarrow(\mathsf{z}_2))::\mathbb{Z}_\bot\mid\Omega) \\
\mathbf{S}[\![(e_1\,\mathbf{?}\,e_2\,\mathbf{:}\,e_3)]\!] & \triangleq & \Lambda\mathsf{R}\bullet(\mathbf{S}[\![e_1]\!]\mathsf{R}=\bot\;?\;\bot\mid\mathbf{S}[\![e_1]\!]\mathsf{R}=\mathsf{z}::\mathbb{Z}_\bot\;? \\
 & & (\downarrow(\mathsf{z})=0\;?\;\mathbf{S}[\![e_2]\!]\mathsf{R}\mid\mathbf{S}[\![e_3]\!]\mathsf{R})\mid\Omega)
\end{array}
$$

Il s'agit donc d'expliquer la sémantique, par exemple :

Syntaxe :

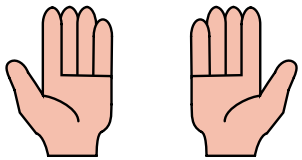
$$
\begin{array}{rcl}
\mathtt{x},\mathtt{f} \in \mathbb{X} & : & \text{variables} \\
e \in \mathbb{E} & : & \text{expressions} \\
e & ::= & \mathtt{x} \mid \boldsymbol{\lambda}\mathtt{x}\cdot e \mid e_1(e_2) \mid \\
 & & \boldsymbol{\mu}\mathtt{f}\cdot\boldsymbol{\lambda}\mathtt{x}\cdot e \mid e_1 - e_2 \mid \\
 & & \mathbf{1} \mid (e_1 \,?\, e_2 : e_3)
\end{array}
$$

Domaines sémantiques :

$$
\begin{array}{rcll}
\mathbb{W} & \triangleq & \{\omega\} & \text{erreur} \\
\mathsf{z} \in \mathbb{Z} & & & \text{entiers} \\
\mathsf{u},\mathsf{f},\varphi \in \mathbb{U} & \cong & \mathbb{W}_\bot \oplus \mathbb{Z}_\bot \oplus [\mathbb{U} \mapsto \mathbb{U}]_\bot & \text{valeurs} \\
\mathsf{R} \in \mathbb{R} & \triangleq & \mathbb{X} \mapsto \mathbb{U} & \text{environnements} \\
\phi \in \mathbb{S} & \triangleq & \mathbb{R} \mapsto \mathbb{U} & \text{domaine sémantique}
\end{array}
$$

Sémantique :

$$
\begin{array}{rcl}
\mathbf{S}[\![\mathtt{x}]\!] & \triangleq & \Lambda\mathsf{R}\cdot \mathsf{R}(\mathtt{x}) \\
\mathbf{S}[\![\boldsymbol{\lambda}\mathtt{x}\cdot e]\!] & \triangleq & \Lambda\mathsf{R}\cdot \uparrow(\Lambda\mathsf{u}\cdot(\mathsf{u} = \bot \vee \mathsf{u} = \Omega \,?\, \mathsf{u} \mid \\
 & & \qquad \mathbf{S}[\![e]\!]\mathsf{R}[\mathtt{x}\leftarrow\mathsf{u}])) :: [\mathbb{U} \mapsto \mathbb{U}]_\bot \\
\mathbf{S}[\![e_1(e_2)]\!] & \triangleq & \Lambda\mathsf{R}\cdot(\mathbf{S}[\![e_1]\!]\mathsf{R} = \bot \vee \mathbf{S}[\![e_2]\!]\mathsf{R} = \bot \,?\, \bot \mid \\
 & & \mathbf{S}[\![e_1]\!]\mathsf{R} = \mathsf{f} :: [\mathbb{U} \mapsto \mathbb{U}]_\bot \,?\, \downarrow(\mathsf{f})(\mathbf{S}[\![e_2]\!]\mathsf{R}) \mid \Omega) \\
\mathbf{S}[\![\boldsymbol{\mu}\mathtt{f}\cdot\boldsymbol{\lambda}\mathtt{x}\cdot e]\!] & \triangleq & \Lambda\mathsf{R}\cdot \mathbf{lfp}^{\sqsubseteq}_{\uparrow(\Lambda\mathsf{u}\cdot\bot)::[\mathbb{U}\mapsto\mathbb{U}]_\bot} \Lambda\varphi\cdot \mathbf{S}[\![\boldsymbol{\lambda}\mathtt{x}\cdot e]\!]\mathsf{R}[\mathtt{f}\leftarrow\varphi] \\
\mathbf{S}[\![\mathbf{1}]\!] & \triangleq & \Lambda\mathsf{R}\cdot \uparrow(1) :: \mathbb{Z}_\bot \\
\mathbf{S}[\![e_1 - e_2]\!] & \triangleq & \Lambda\mathsf{R}\cdot(\mathbf{S}[\![e_1]\!]\mathsf{R} = \bot \vee \mathbf{S}[\![e_2]\!]\mathsf{R} = \bot \,?\, \bot \mid \\
 & & \mathbf{S}[\![e_1]\!]\mathsf{R} = \mathsf{z}_1 :: \mathbb{Z}_\bot \wedge \mathbf{S}[\![e_2]\!]\mathsf{R} = \mathsf{z}_2 :: \mathbb{Z}_\bot \,? \\
 & & \uparrow(\downarrow(\mathsf{z}_1) - \downarrow(\mathsf{z}_2)) :: \mathbb{Z}_\bot \mid \Omega) \\
\mathbf{S}[\![(e_1 \,?\, e_2 : e_3)]\!] & \triangleq & \Lambda\mathsf{R}\cdot(\mathbf{S}[\![e_1]\!]\mathsf{R} = \bot \,?\, \bot \mid \mathbf{S}[\![e_1]\!]\mathsf{R} = \mathsf{z} :: \mathbb{Z}_\bot \,? \\
 & & (\downarrow(\mathsf{z}) = 0 \,?\, \mathbf{S}[\![e_2]\!]\mathsf{R} \mid \mathbf{S}[\![e_3]\!]\mathsf{R}) \mid \Omega)
\end{array}
$$

avec ça 

Il s'agit donc d'expliquer la sémantique, par exemple :

Syntaxe :

$$
\begin{array}{rcl}
\mathtt{x},\mathtt{f} \in \mathbb{X} & : & \text{variables} \\
e \in \mathbb{E} & : & \text{expressions} \\
e & ::= & \mathtt{x} \mid \boldsymbol{\lambda}\mathtt{x}\cdot e \mid e_1(e_2) \mid \\
 & & \boldsymbol{\mu}\mathtt{f}\cdot\boldsymbol{\lambda}\mathtt{x}\cdot e \mid e_1 - e_2 \mid \\
 & & \mathbf{1} \mid (e_1\,\mathbf{?}\,e_2\,\mathbf{:}\,e_3)
\end{array}
$$

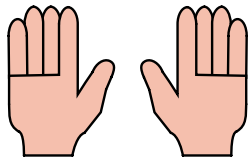
Domaines sémantiques :

$$
\begin{array}{rcll}
\mathbb{W} & \triangleq & \{\omega\} & \text{erreur} \\
\mathsf{z} \in \mathbb{Z} & & & \text{entiers} \\
\mathsf{u},\mathsf{f},\varphi \in \mathbb{U} & \cong & \mathbb{W}_\bot \oplus \mathbb{Z}_\bot \oplus [\mathbb{U} \mapsto \mathbb{U}]_\bot & \text{valeurs} \\
\mathsf{R} \in \mathbb{R} & \triangleq & \mathbb{X} \mapsto \mathbb{U} & \text{environnements} \\
\phi \in \mathbb{S} & \triangleq & \mathbb{R} \mapsto \mathbb{U} & \text{domaine sémantique}
\end{array}
$$

Sémantique :

$$
\begin{array}{rcl}
\mathbf{S}[\![\mathtt{x}]\!] & \triangleq & \Lambda\mathsf{R}\cdot \mathsf{R}(\mathtt{x}) \\
\mathbf{S}[\![\boldsymbol{\lambda}\mathtt{x}\cdot e]\!] & \triangleq & \Lambda\mathsf{R}\cdot \uparrow(\Lambda\mathsf{u}\cdot(\mathsf{u} = \bot \vee \mathsf{u} = \Omega\ ?\ \mathsf{u} \mid \\
 & & \qquad \mathbf{S}[\![e]\!]\mathsf{R}[\mathtt{x}\leftarrow\mathsf{u}])) :: [\mathbb{U} \mapsto \mathbb{U}]_\bot \\
\mathbf{S}[\![e_1(e_2)]\!] & \triangleq & \Lambda\mathsf{R}\cdot(\mathbf{S}[\![e_1]\!]\mathsf{R} = \bot \vee \mathbf{S}[\![e_2]\!]\mathsf{R} = \bot\ ?\ \bot \mid \\
 & & \mathbf{S}[\![e_1]\!]\mathsf{R} = \mathsf{f} :: [\mathbb{U} \mapsto \mathbb{U}]_\bot\ ?\ \downarrow(\mathsf{f})(\mathbf{S}[\![e_2]\!]\mathsf{R}) \mid \Omega) \\
\mathbf{S}[\![\boldsymbol{\mu}\mathtt{f}\cdot\boldsymbol{\lambda}\mathtt{x}\cdot e]\!] & \triangleq & \Lambda\mathsf{R}\cdot \mathbf{lfp}^{\sqsubseteq}_{\uparrow(\Lambda\mathsf{u}\cdot\bot)::[\mathbb{U}\mapsto\mathbb{U}]_\bot} \Lambda\varphi\cdot \mathbf{S}[\![\boldsymbol{\lambda}\mathtt{x}\cdot e]\!]\mathsf{R}[\mathtt{f}\leftarrow\varphi] \\
\mathbf{S}[\![\mathbf{1}]\!] & \triangleq & \Lambda\mathsf{R}\cdot \uparrow(1) :: \mathbb{Z}_\bot \\
\mathbf{S}[\![e_1 - e_2]\!] & \triangleq & \Lambda\mathsf{R}\cdot(\mathbf{S}[\![e_1]\!]\mathsf{R} = \bot \vee \mathbf{S}[\![e_2]\!]\mathsf{R} = \bot\ ?\ \bot \mid \\
 & & \mathbf{S}[\![e_1]\!]\mathsf{R} = \mathsf{z}_1 :: \mathbb{Z}_\bot \wedge \mathbf{S}[\![e_2]\!]\mathsf{R} = \mathsf{z}_2 :: \mathbb{Z}_\bot\ ? \\
 & & \uparrow(\downarrow(\mathsf{z}_1) - \downarrow(\mathsf{z}_2)) :: \mathbb{Z}_\bot \mid \Omega) \\
\mathbf{S}[\![(e_1\,\mathbf{?}\,e_2\,\mathbf{:}\,e_3)]\!] & \triangleq & \Lambda\mathsf{R}\cdot(\mathbf{S}[\![e_1]\!]\mathsf{R} = \bot\ ?\ \bot \mid \mathbf{S}[\![e_1]\!]\mathsf{R} = \mathsf{z} :: \mathbb{Z}_\bot\ ? \\
 & & (\downarrow(\mathsf{z}) = 0\ ?\ \mathbf{S}[\![e_2]\!]\mathsf{R} \mid \mathbf{S}[\![e_3]\!]\mathsf{R}) \mid \Omega)
\end{array}
$$

avec ça



Il s'agit donc d'expliquer la sémantique, par exemple :

Syntaxe :

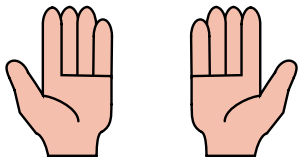
$$
\begin{array}{rcl}
\mathtt{x},\mathtt{f} \in \mathbb{X} & : & \text{variables} \\
e \in \mathbb{E} & : & \text{expressions} \\
e & ::= & \mathtt{x} \mid \boldsymbol{\lambda}\mathtt{x}\cdot e \mid e_1(e_2) \mid \\
 & & \boldsymbol{\mu}\mathtt{f}\cdot\boldsymbol{\lambda}\mathtt{x}\cdot e \mid e_1 - e_2 \mid \\
 & & \mathbf{1} \mid (e_1 \,?\, e_2 : e_3)
\end{array}
$$

Domaines sémantiques :

$$
\begin{array}{rcll}
\mathbb{W} & \triangleq & \{\omega\} & \text{erreur} \\
\mathsf{z} \in \mathbb{Z} & & & \text{entiers} \\
\mathsf{u},\mathsf{f},\varphi \in \mathbb{U} & \cong & \mathbb{W}_\bot \oplus \mathbb{Z}_\bot \oplus [\mathbb{U} \mapsto \mathbb{U}]_\bot & \text{valeurs} \\
\mathsf{R} \in \mathbb{R} & \triangleq & \mathbb{X} \mapsto \mathbb{U} & \text{environnements} \\
\phi \in \mathbb{S} & \triangleq & \mathbb{R} \mapsto \mathbb{U} & \text{domaine sémantique}
\end{array}
$$

Sémantique :

$$
\begin{array}{rcl}
\mathbf{S}[\![\mathtt{x}]\!] & \triangleq & \Lambda\mathsf{R}\cdot \mathsf{R}(\mathtt{x}) \\
\mathbf{S}[\![\boldsymbol{\lambda}\mathtt{x}\cdot e]\!] & \triangleq & \Lambda\mathsf{R}\cdot \uparrow(\Lambda\mathsf{u}\cdot(\mathsf{u} = \bot \vee \mathsf{u} = \Omega \,?\, \mathsf{u} \mid \\
 & & \qquad \mathbf{S}[\![e]\!]\mathsf{R}[\mathtt{x}\leftarrow\mathsf{u}])) :: [\mathbb{U} \mapsto \mathbb{U}]_\bot \\
\mathbf{S}[\![e_1(e_2)]\!] & \triangleq & \Lambda\mathsf{R}\cdot(\mathbf{S}[\![e_1]\!]\mathsf{R} = \bot \vee \mathbf{S}[\![e_2]\!]\mathsf{R} = \bot \,?\, \bot \mid \\
 & & \mathbf{S}[\![e_1]\!]\mathsf{R} = \mathsf{f} :: [\mathbb{U} \mapsto \mathbb{U}]_\bot \,?\, \downarrow(\mathsf{f})(\mathbf{S}[\![e_2]\!]\mathsf{R}) \mid \Omega) \\
\mathbf{S}[\![\boldsymbol{\mu}\mathtt{f}\cdot\boldsymbol{\lambda}\mathtt{x}\cdot e]\!] & \triangleq & \Lambda\mathsf{R}\cdot \mathbf{lfp}^{\sqsubseteq}_{\uparrow(\Lambda\mathsf{u}\cdot\bot)::[\mathbb{U}\mapsto\mathbb{U}]_\bot} \Lambda\varphi\cdot \mathbf{S}[\![\boldsymbol{\lambda}\mathtt{x}\cdot e]\!]\mathsf{R}[\mathtt{f}\leftarrow\varphi] \\
\mathbf{S}[\![\mathbf{1}]\!] & \triangleq & \Lambda\mathsf{R}\cdot \uparrow(1) :: \mathbb{Z}_\bot \\
\mathbf{S}[\![e_1 - e_2]\!] & \triangleq & \Lambda\mathsf{R}\cdot(\mathbf{S}[\![e_1]\!]\mathsf{R} = \bot \vee \mathbf{S}[\![e_2]\!]\mathsf{R} = \bot \,?\, \bot \mid \\
 & & \mathbf{S}[\![e_1]\!]\mathsf{R} = \mathsf{z}_1 :: \mathbb{Z}_\bot \wedge \mathbf{S}[\![e_2]\!]\mathsf{R} = \mathsf{z}_2 :: \mathbb{Z}_\bot \,? \\
 & & \uparrow(\downarrow(\mathsf{z}_1) - \downarrow(\mathsf{z}_2)) :: \mathbb{Z}_\bot \mid \Omega) \\
\mathbf{S}[\![(e_1 \,?\, e_2 : e_3)]\!] & \triangleq & \Lambda\mathsf{R}\cdot(\mathbf{S}[\![e_1]\!]\mathsf{R} = \bot \,?\, \bot \mid \mathbf{S}[\![e_1]\!]\mathsf{R} = \mathsf{z} :: \mathbb{Z}_\bot \,? \\
 & & (\downarrow(\mathsf{z}) = 0 \,?\, \mathbf{S}[\![e_2]\!]\mathsf{R} \mid \mathbf{S}[\![e_3]\!]\mathsf{R}) \mid \Omega)
\end{array}
$$

avec ça 

Sémantique

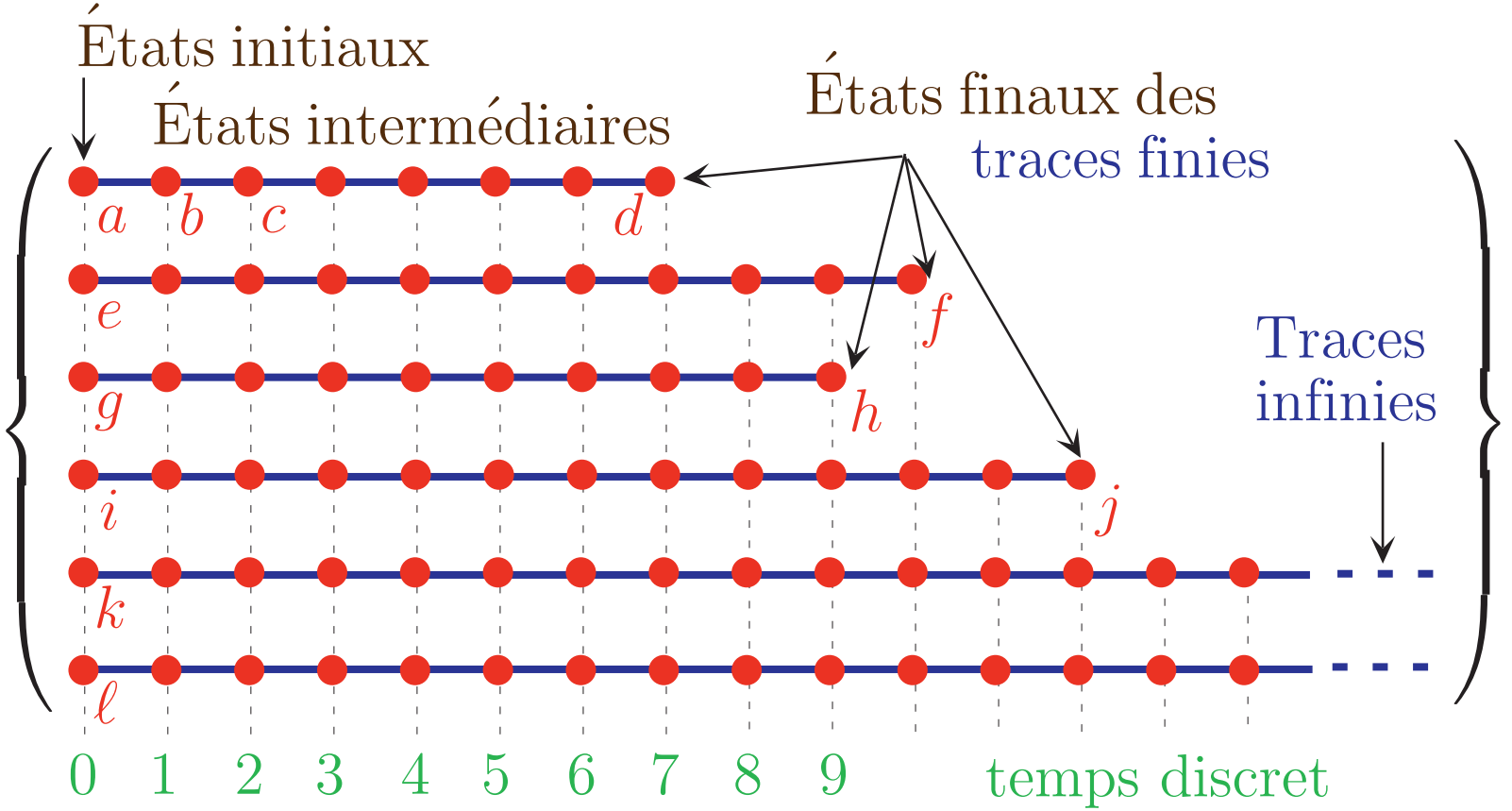
– La sémantique d'un programme fournit un modèle mathématique formel de tous les comportements possibles d'un système informatique exécutant ce programme (en interaction avec un environnement quelconque) ;

Sémantique

– La sémantique d'un programme fournit un modèle mathématique formel de tous les comportements possibles d'un système informatique exécutant ce programme (en interaction avec un environnement quelconque) ;

– La sémantique d'un langage définit la sémantique de tout programme écrit dans ce langage.

Exemple 1 : sémantique de traces

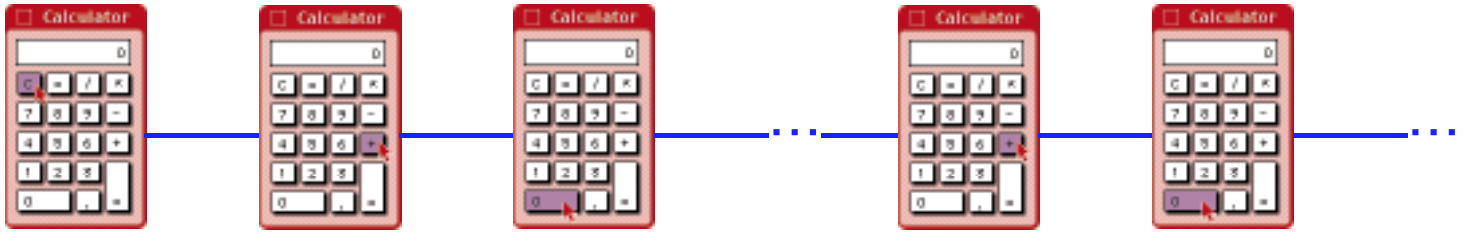


Exemples de traces de calculs

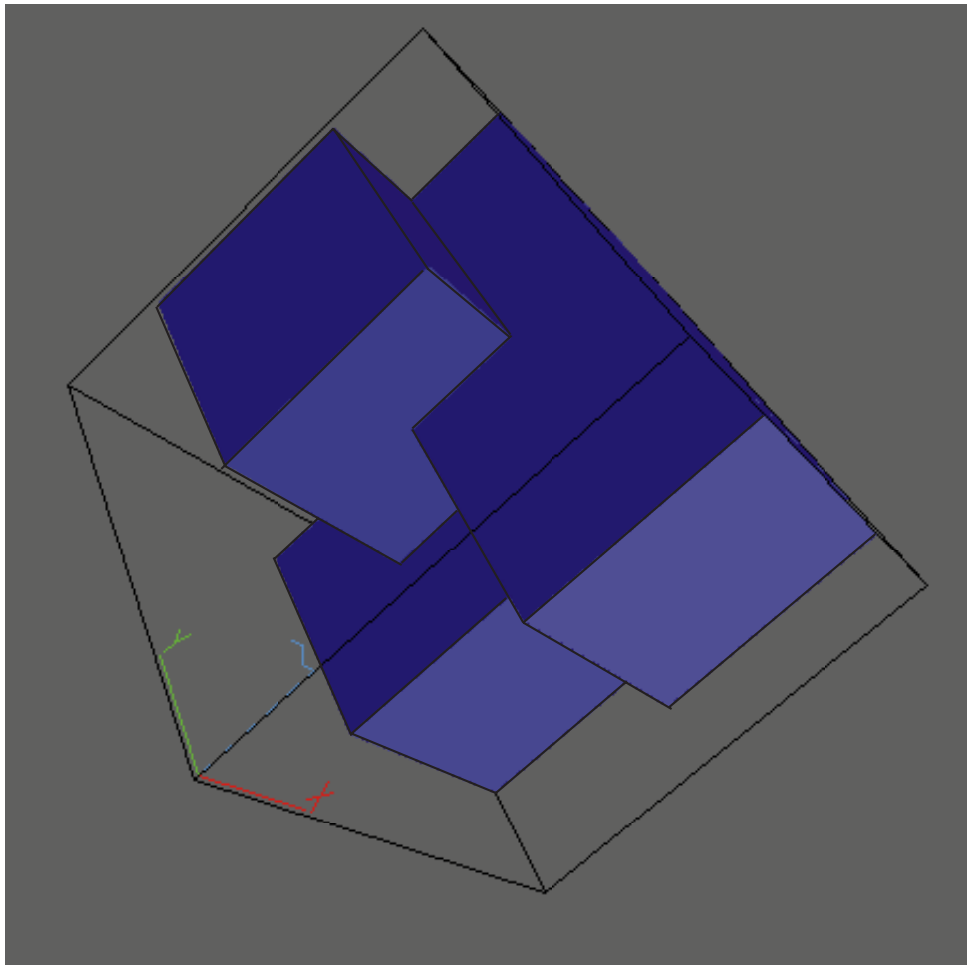
– Finie (C1+1=) :

– Erronée  (C1+1+1+1...) :

– Infinie (C+0+0+0...) :



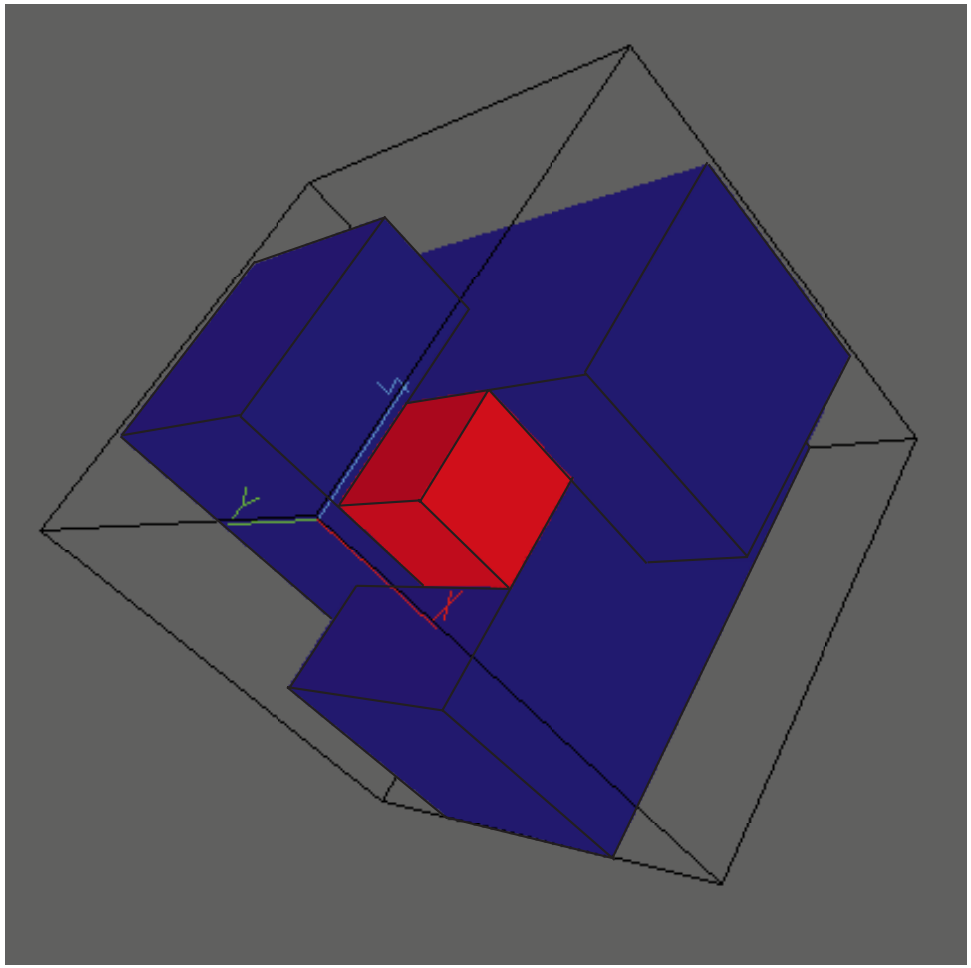
Exemple 2 : sémantique géométrique



```
⟦ Pa.Pb.Va.Vb
‖ Pb.Pc.Vb.Vc
‖ Pc.Pa.Vc.Va ⟧
```

thèse É. Goubault, X, 1995

Exemple 2 : sémantique géométrique (interblocage)



```
[[ Pa.Pb.Va.Vb
|| Pb.Pc.Vb.Vc
|| Pc.Pa.Vc.Va ]]
```

interblocage

thèse É. Goubault, X, 1995

Indécidabilité

– Toutes les questions intéressantes relatives à la sémantique d'un programme non trivial sont indécidables;

Indécidabilité

– Toutes les questions intéressantes relatives à la sémantique d'un programme non trivial sont indécidables:

  $\Rightarrow$ aucun ordinateur ne peut toujours y répondre exactement en un temps fini;

Indécidabilité

– Toutes les questions intéressantes relatives à la sémantique d'un programme non trivial sont indécidables:

  $\Rightarrow$ aucun ordinateur ne peut toujours y répondre exactement en un temps fini;

– On peut définir mathématiquement la sémantique d'un programme comme solution d'une équation de point fixe;

Indécidabilité

– Toutes les questions intéressantes relatives à la sémantique d'un programme non trivial sont indécidables:

  $\Rightarrow$ aucun ordinateur ne peut toujours y répondre exactement en un temps fini;

– On peut définir mathématiquement la sémantique d'un programme comme solution d'une équation de point fixe:

  $\Rightarrow$ aucun ordinateur ne peut résoudre exactement les équations en un temps fini.

Sémantiques et points fixes

Syntaxe :

$$
\begin{array}{rcl}
\mathtt{x},\mathtt{f} \in \mathbb{X} & : & \text{variables} \\
e \in \mathbb{E} & : & \text{expressions} \\
e & ::= & \mathtt{x} \mid \boldsymbol{\lambda}\mathtt{x}\cdot e \mid e_1(e_2) \mid \\
 & & \boldsymbol{\mu}\mathtt{f}\cdot\boldsymbol{\lambda}\mathtt{x}\cdot e \mid e_1 - e_2 \mid \\
 & & \mathbf{1} \mid (e_1\,?\,e_2 : e_3)
\end{array}
$$

Domaines sémantiques :

$$
\begin{array}{rcll}
\mathbb{W} & \triangleq & \{\omega\} & \text{erreur} \\
\mathtt{z} \in \mathbb{Z} & & & \text{entiers} \\
\mathtt{u},\mathtt{f},\varphi \in \mathbb{U} & \cong & \mathbb{W}_\bot \oplus \mathbb{Z}_\bot \oplus [\mathbb{U} \mapsto \mathbb{U}]_\bot & \text{valeurs} \\
\mathtt{R} \in \mathbb{R} & \triangleq & \mathbb{X} \mapsto \mathbb{U} & \text{environnements} \\
\phi \in \mathbb{S} & \triangleq & \mathbb{R} \mapsto \mathbb{U} & \text{domaine sémantique}
\end{array}
$$

Sémantique :

$$
\begin{array}{rcl}
\mathbf{S}[\![\mathtt{x}]\!] & \triangleq & \Lambda\mathtt{R}\bullet \mathtt{R}(\mathtt{x}) \\
\mathbf{S}[\![\boldsymbol{\lambda}\mathtt{x}\cdot e]\!] & \triangleq & \Lambda\mathtt{R}\bullet \uparrow(\Lambda\mathtt{u}\bullet(\mathtt{u} = \bot \vee \mathtt{u} = \Omega\ ?\ \mathtt{u} \mid \\
 & & \qquad \mathbf{S}[\![e]\!]\mathtt{R}[\mathtt{x}\leftarrow\mathtt{u}])) :: [\mathbb{U} \mapsto \mathbb{U}]_\bot \\
\mathbf{S}[\![e_1(e_2)]\!] & \triangleq & \Lambda\mathtt{R}\bullet(\mathbf{S}[\![e_1]\!]\mathtt{R} = \bot \vee \mathbf{S}[\![e_2]\!]\mathtt{R} = \bot\ ?\ \bot \mid \\
 & & \mathbf{S}[\![e_1]\!]\mathtt{R} = \mathtt{f} :: [\mathbb{U} \mapsto \mathbb{U}]_\bot\ ?\ \downarrow(\mathtt{f})(\mathbf{S}[\![e_2]\!]\mathtt{R}) \mid \Omega) \\
\mathbf{S}[\![\boldsymbol{\mu}\mathtt{f}\cdot\boldsymbol{\lambda}\mathtt{x}\cdot e]\!] & \triangleq & \Lambda\mathtt{R}\bullet \mathbf{lfp}^{\sqsubseteq}_{\uparrow(\Lambda\mathtt{u}\bullet\bot)::[\mathbb{U}\mapsto\mathbb{U}]_\bot} \Lambda\varphi\bullet \mathbf{S}[\![\boldsymbol{\lambda}\mathtt{x}\cdot e]\!]\mathtt{R}[\mathtt{f}\leftarrow\varphi] \\
\mathbf{S}[\![\mathbf{1}]\!] & \triangleq & \Lambda\mathtt{R}\bullet \uparrow(1) :: \mathbb{Z}_\bot \\
\mathbf{S}[\![e_1 - e_2]\!] & \triangleq & \Lambda\mathtt{R}\bullet(\mathbf{S}[\![e_1]\!]\mathtt{R} = \bot \vee \mathbf{S}[\![e_2]\!]\mathtt{R} = \bot\ ?\ \bot \mid \\
 & & \mathbf{S}[\![e_1]\!]\mathtt{R} = \mathtt{z}_1 :: \mathbb{Z}_\bot \wedge \mathbf{S}[\![e_2]\!]\mathtt{R} = \mathtt{z}_2 :: \mathbb{Z}_\bot\ ? \\
 & & \uparrow(\downarrow(\mathtt{z}_1) - \downarrow(\mathtt{z}_2)) :: \mathbb{Z}_\bot \mid \Omega) \\
\mathbf{S}[\![(e_1\,?\,e_2 : e_3)]\!] & \triangleq & \Lambda\mathtt{R}\bullet(\mathbf{S}[\![e_1]\!]\mathtt{R} = \bot\ ?\ \bot \mid \mathbf{S}[\![e_1]\!]\mathtt{R} = \mathtt{z} :: \mathbb{Z}_\bot\ ? \\
 & & (\downarrow(\mathtt{z}) = 0\ ?\ \mathbf{S}[\![e_2]\!]\mathtt{R} \mid \mathbf{S}[\![e_3]\!]\mathtt{R}) \mid \Omega)
\end{array}
$$

points fixes : intuition

Calculs =

points fixes : intuition

$\mathbf{Calculs} = \{\bullet \mid \bullet \text{ est un état final}\}$

points fixes : intuition

$$\mathbf{Calculs} = \{\bullet \mid \bullet \text{ est un état final}\}$$

$$\cup \{\bullet\!\!-\!\!-\!\!-\!\!\bullet\!\!-\!\!-\!\!-\!\!\bullet\!\!-\!\!-\!\!-\!\!\bullet \mid \bullet\!\!-\!\!-\!\!-\!\!\bullet \text{ est un pas élémentaire } \&$$

$$\bullet\!\!-\!\!-\!\!-\!\!\bullet\!\!-\!\!-\!\!-\!\!\bullet \in \mathbf{Calculs}\}$$

points fixes : intuition

Calculs $=$ $\{$ ● $\mid$ ● est un état final$\}$

$\cup\,\{$ ●——●——. . .——● $\mid$ ●——● est un pas élémentaire &

●——. . .——● $\in$ **Calculs**$\}$

$\cup\,\{$ ●——●——. . .——. . . $\mid$ ●——● est un pas élémentaire &

●——. . .——. . . $\in$ **Calculs**$^{\infty}\}$

points fixes : intuition

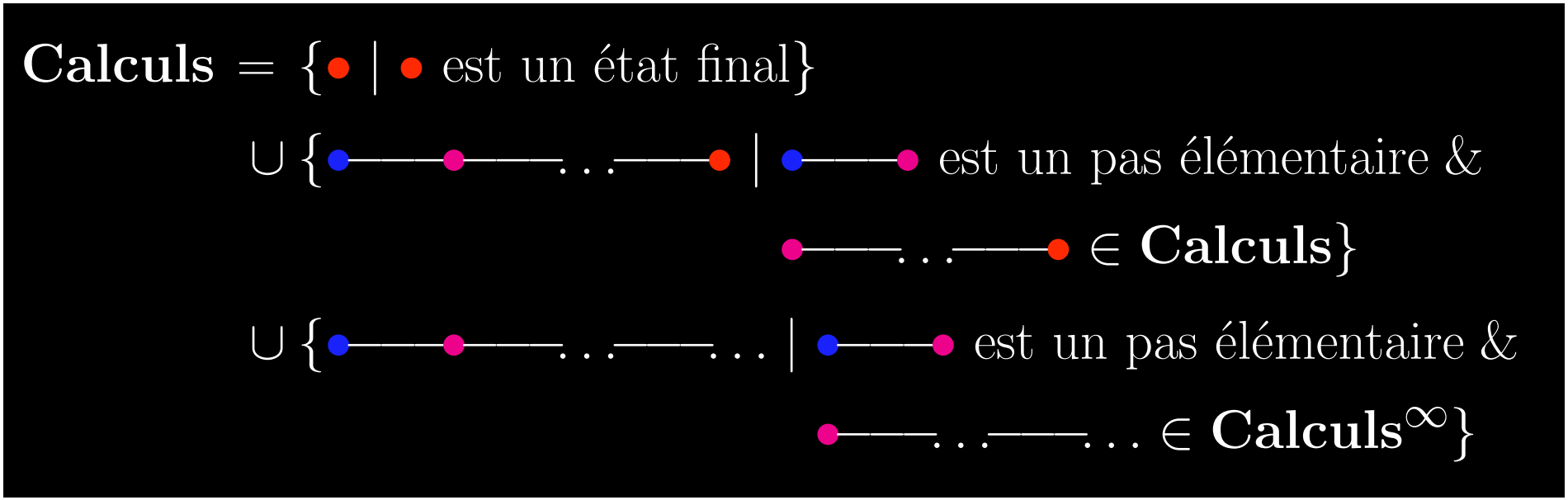
$$\mathbf{Calculs} = \{\bullet \mid \bullet \text{ est un état final}\}$$

$$\cup \{\bullet\!\!-\!\!-\!\!\bullet\!\!-\!\!-\!\!-\!\!\bullet \mid \bullet\!\!-\!\!-\!\!\bullet \text{ est un pas élémentaire } \& \; \bullet\!\!-\!\!-\!\!-\!\!\bullet \in \mathbf{Calculs}\}$$

$$\cup \{\bullet\!\!-\!\!-\!\!\bullet\!\!-\!\!-\!\!-\!\!\bullet\!\!-\!\!-\!\!-\!\!\bullet \mid \bullet\!\!-\!\!-\!\!\bullet \text{ est un pas élémentaire } \& \; \bullet\!\!-\!\!-\!\!-\!\!\bullet\!\!-\!\!-\!\!-\!\!\bullet \in \mathbf{Calculs}^{\infty}\}$$

En général, l'équation a plusieurs solutions.

Plus petits points fixes : intuition



En général, l'équation a plusieurs solutions. Choisir la plus petite pour l'ordre :

« *plus de traces finies & moins de traces infinies* ».

2.2 Interprétation abstraite

Interprétation abstraite

– L'interprétation abstraite est une théorie de l'approximation de sémantiques de langages (de programmation ou de spécification) ;

Interprétation abstraite

– L'interprétation abstraite est une théorie de l'approximation de sémantiques de langages (de programmation ou de spécification) ;

– L'interprétation abstraite permet de formaliser l'idée qu'une sémantique est plus ou moins précise selon le niveau d'observation auquel on se place.

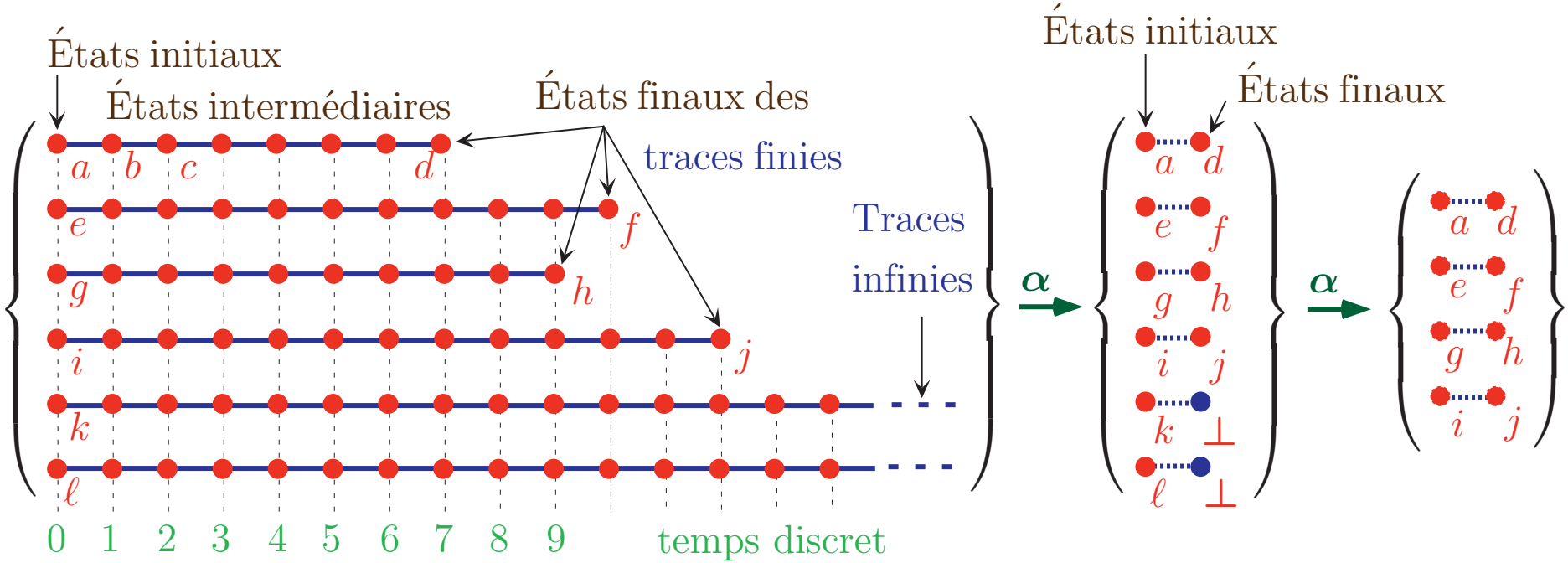
Exemples familiers d'abstractions

concret	abstrait
citoyen	
réseau routier	
film	
voiture	
article scientifique	
article scientifique	
nombre	

Exemples familiers d'abstractions

concret	abstrait
citoyen	carte d'identité
réseau routier	carte routière
film	affiche
voiture	marque
article scientifique	résumé (*abstract*)
article scientifique	mots clef
nombre	signe et/ou parité

Exemples de sémantiques approchées



Sémantique de traces | **Sémantique dénotationnelle** | **Sémantique naturelle**

Perte d'information

– La perte d'information ne permet pas de répondre à toutes les questions ;

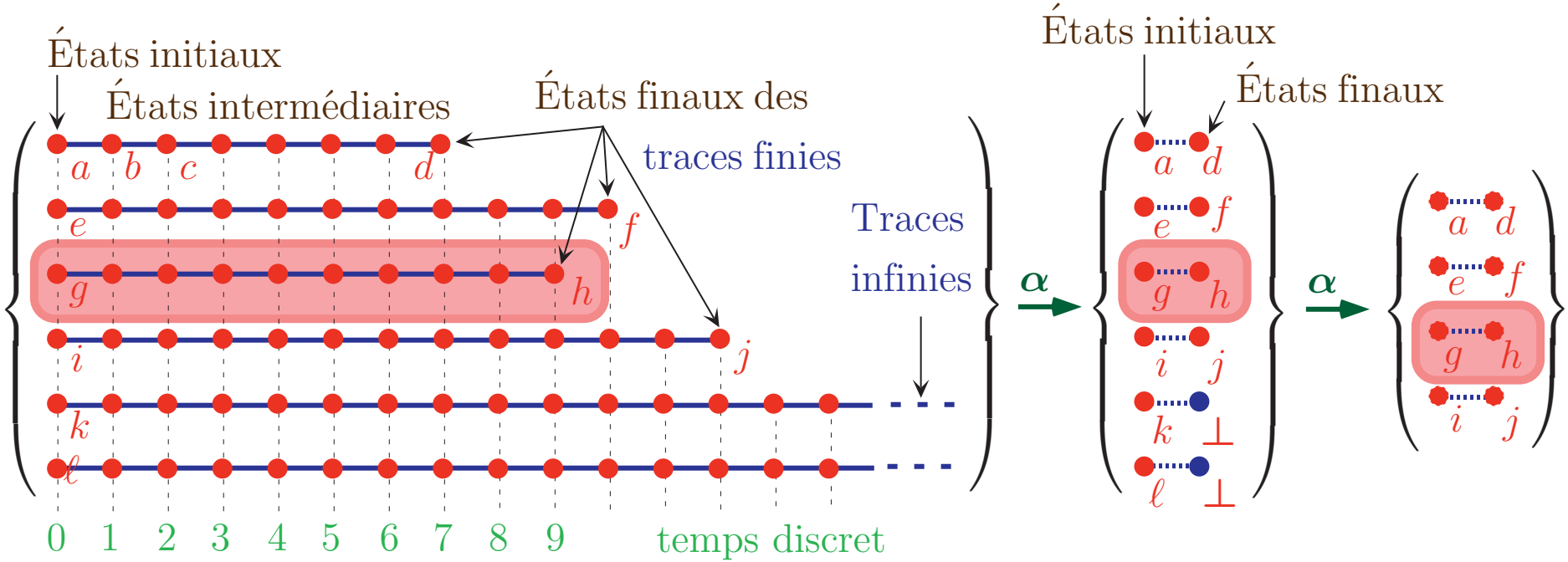
Perte d'information

– La perte d'information ne permet pas de répondre à toutes les questions ;
– Toutes les réponses données pour sémantique abstraite sont toujours justes pour la sémantique concrète.

Exemple de perte d'information

	Concret ←		→Abstrait
Question	sémantique de traces	sémantique dé--notationnelle	sémantique naturelle
Partant de l'état g peut-on terminer dans l'état h ?	—	—	—

Sémantiques



Sémantique de traces **Sémantique dénotationnelle** **Sémantique naturelle**

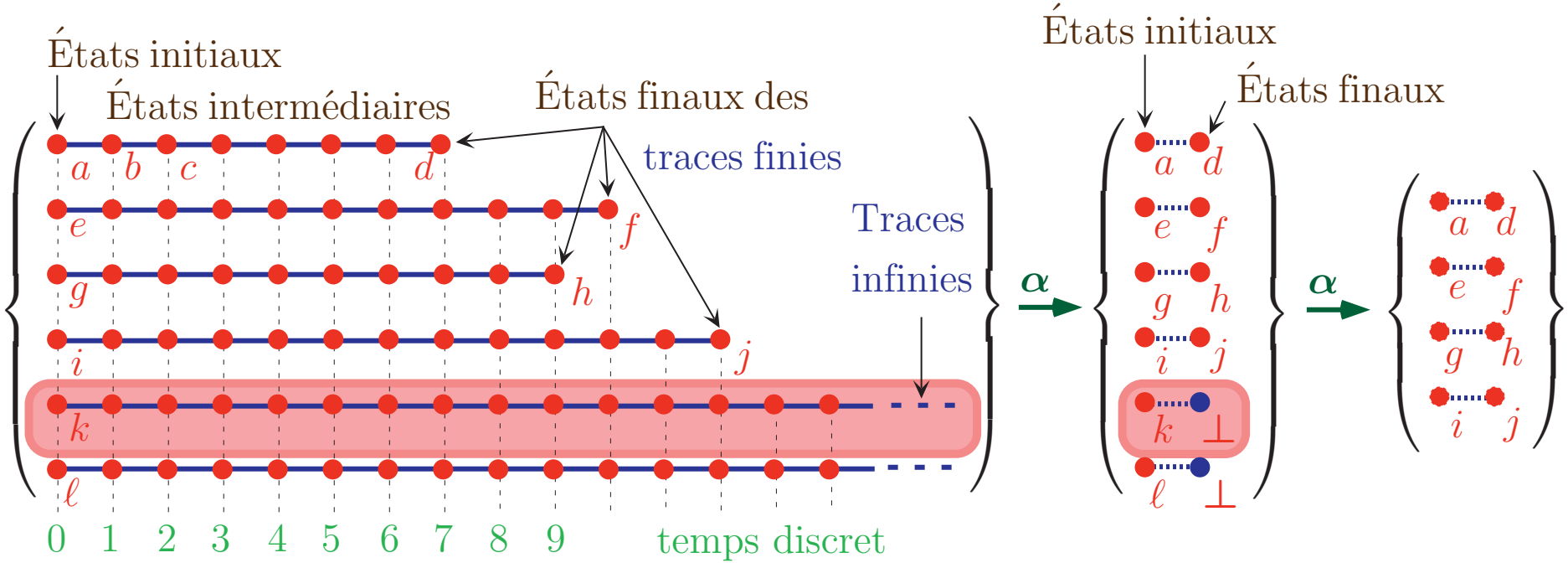
Exemple de perte d'information

	Concret ←		→Abstrait
Question	sémantique de traces	sémantique dé--notationnelle	sémantique naturelle
Partant de l'état g peut-on terminer dans l'état h ?	oui	oui	oui

Exemple de perte d'information

	Concret ←		→Abstrait
Question	sémantique de traces	sémantique dé--notationnelle	sémantique naturelle
Partant de l'état g peut-on terminer dans l'état h ?	oui	oui	oui
Le programme termine-t-il à partir de l'état k ?	—	—	—

Sémantiques



Sémantique de traces | **Sémantique dénotationnelle** | **Sémantique naturelle**

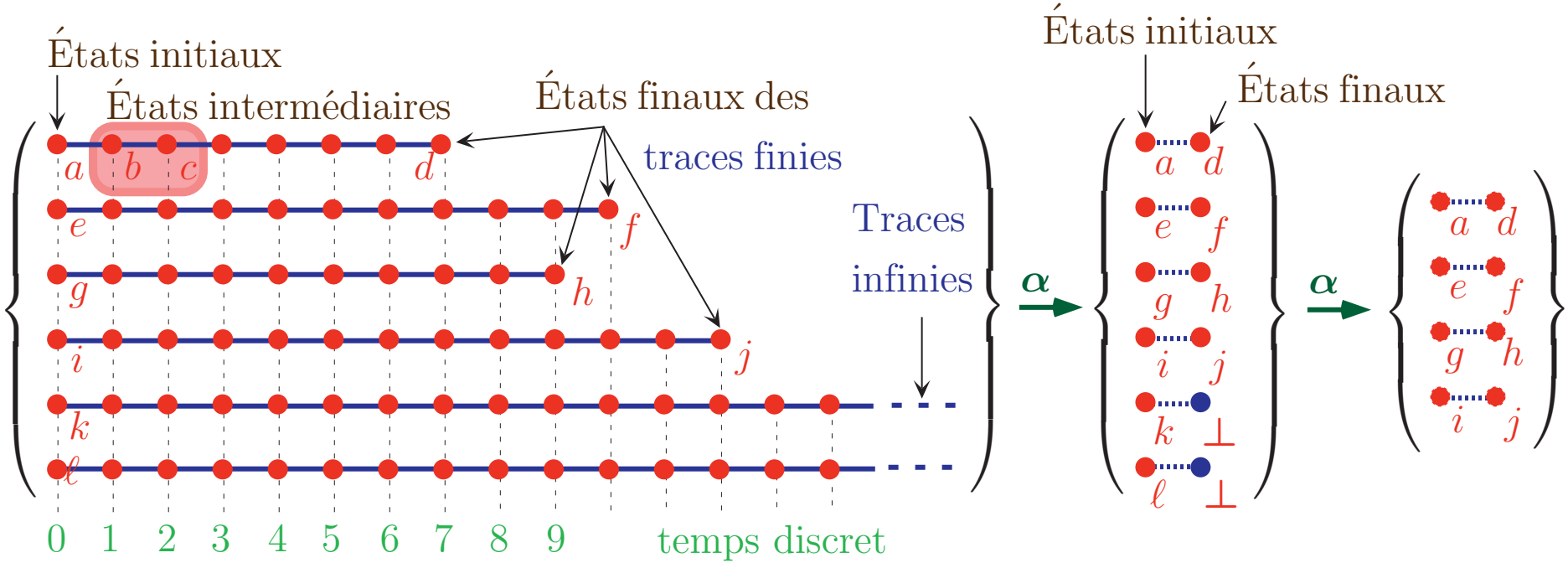
Exemple de perte d'information

	Concret ←		→Abstrait
Question	sémantique de traces	sémantique dé--notationnelle	sémantique naturelle
Partant de l'état g peut-on terminer dans l'état h ?	oui	oui	oui
Le programme termine-t-il à partir de l'état k ?	non	non	???

Exemple de perte d'information

	Concret ←		→Abstrait
Question	sémantique de traces	sémantique dé--notationnelle	sémantique naturelle
Partant de l'état g peut-on terminer dans l'état h ?	oui	oui	oui
Le programme termine-t-il à partir de l'état k ?	non	non	???
L'état b peut-il être immédiatement suivi de l'état c ?	—	—	—

Sémantiques



Sémantique de traces | **Sémantique dénotationnelle** | **Sémantique naturelle**

Exemple de perte d'information

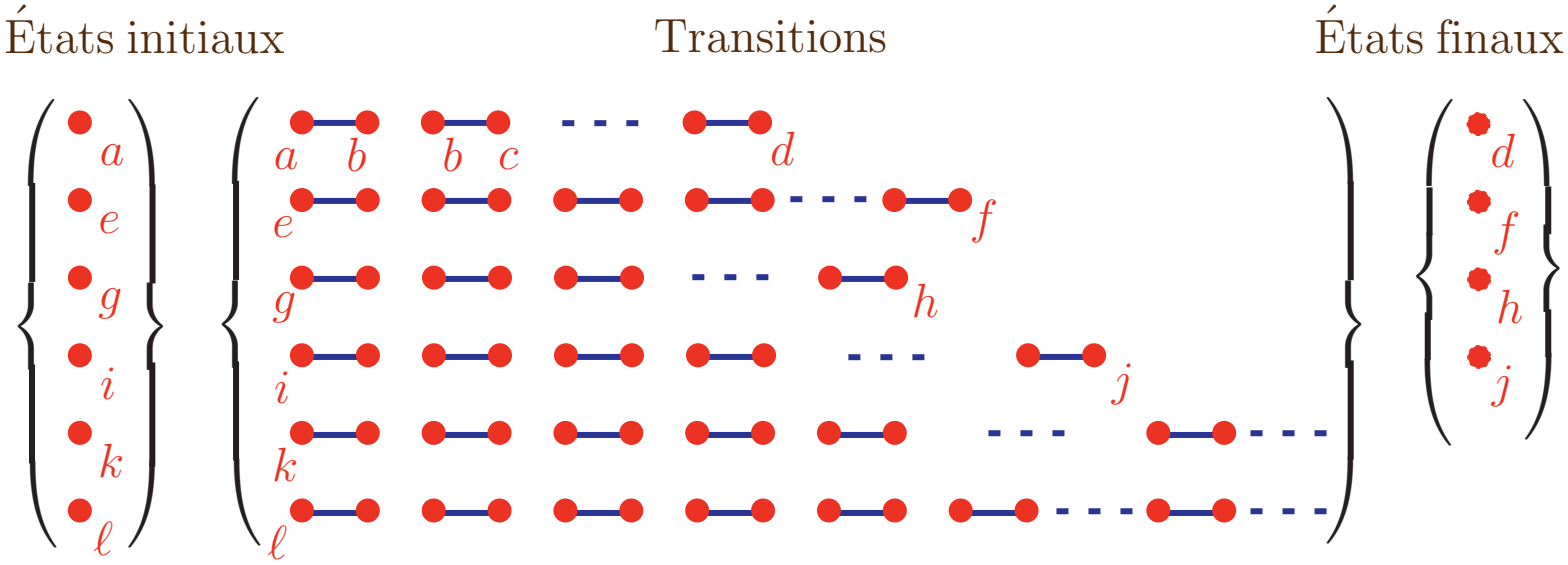
	Concret ←		→Abstrait
Question	sémantique de traces	sémantique dé--notationnelle	sémantique naturelle
Partant de l'état g peut-on terminer dans l'état h ?	oui	oui	oui
Le programme termine-t-il à partir de l'état k ?	non	non	???
L'état b peut-il être immédiatement suivi de l'état c ?	oui	???	???

Exemple de perte d'information

	Concret ←		→Abstrait
Question	sémantique de traces	sémantique dé--notationnelle	sémantique naturelle
Partant de l'état g peut-on terminer dans l'état h ?	oui	oui	oui
Le programme termine-t-il à partir de l'état k ?	non	non	???
L'état b peut-il être immédiatement suivi de l'état c ?	oui	???	???

Les sémantiques les plus concrètes permettent de répondre à plus de questions. Les sémantiques abstraites sont plus simples.

Exemple de sémantique approchée non comparable

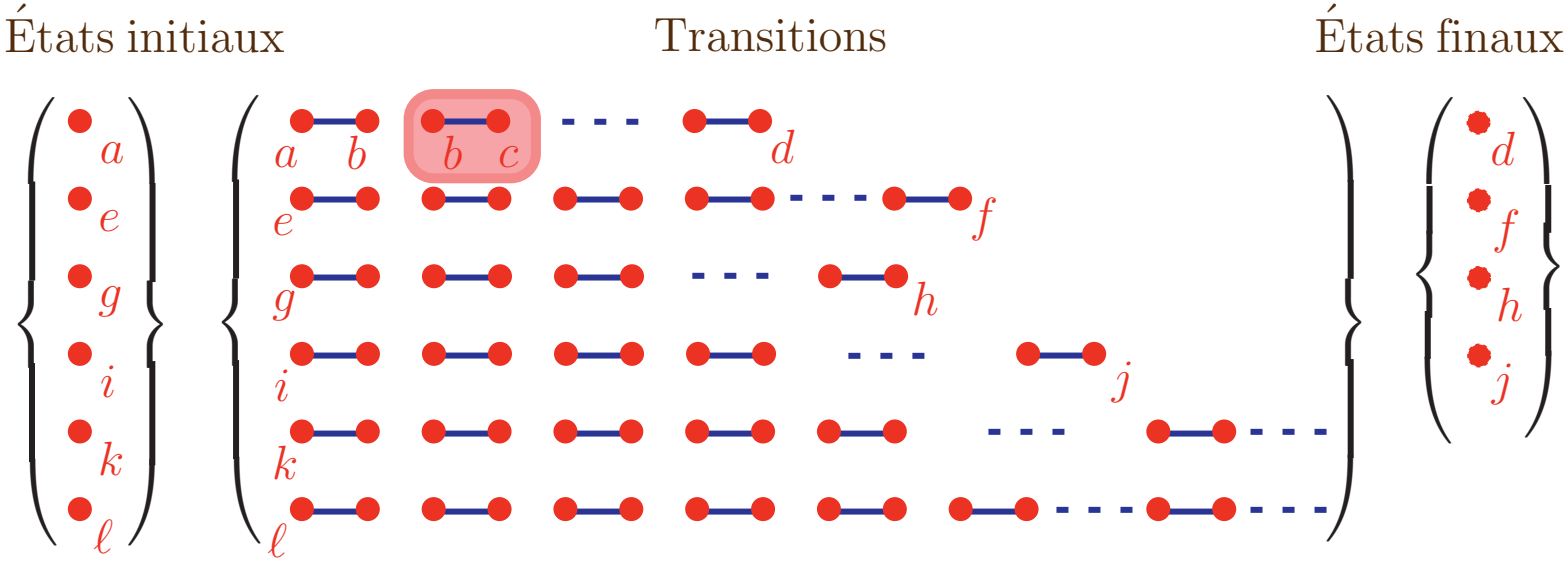


Sémantique opérationnelle

Quelle est la perte d'information ?

	Concret ←		→ Abstrait	
Question	sémantique de traces	sémantique dé--notationnelle	sémantique naturelle	sémantique opérationnelle
Partant de l'état g peut-on terminer dans l'état h ?	oui	oui	oui	—
Le programme termine-t-il à partir de l'état k ?	non	non	???	—
L'état b peut-il être immédiatement suivi de l'état c ?	oui	???	???	—

Sémantique opérationnelle



Sémantique opérationnelle

Perte d'information non comparable

	Concret←		→Abstrait	Incomparable
Question	sémantique de traces	sémantique dé--notationnelle	sémantique naturelle	sémantique opérationnelle
Partant de l'état g peut-on terminer dans l'état h ?	oui	oui	oui	???
Le programme termine-t-il à partir de l'état k ?	non	non	???	???
L'état b peut-il être immédiatement suivi de l'état c ?	oui	???	???	oui

Approximations calculables

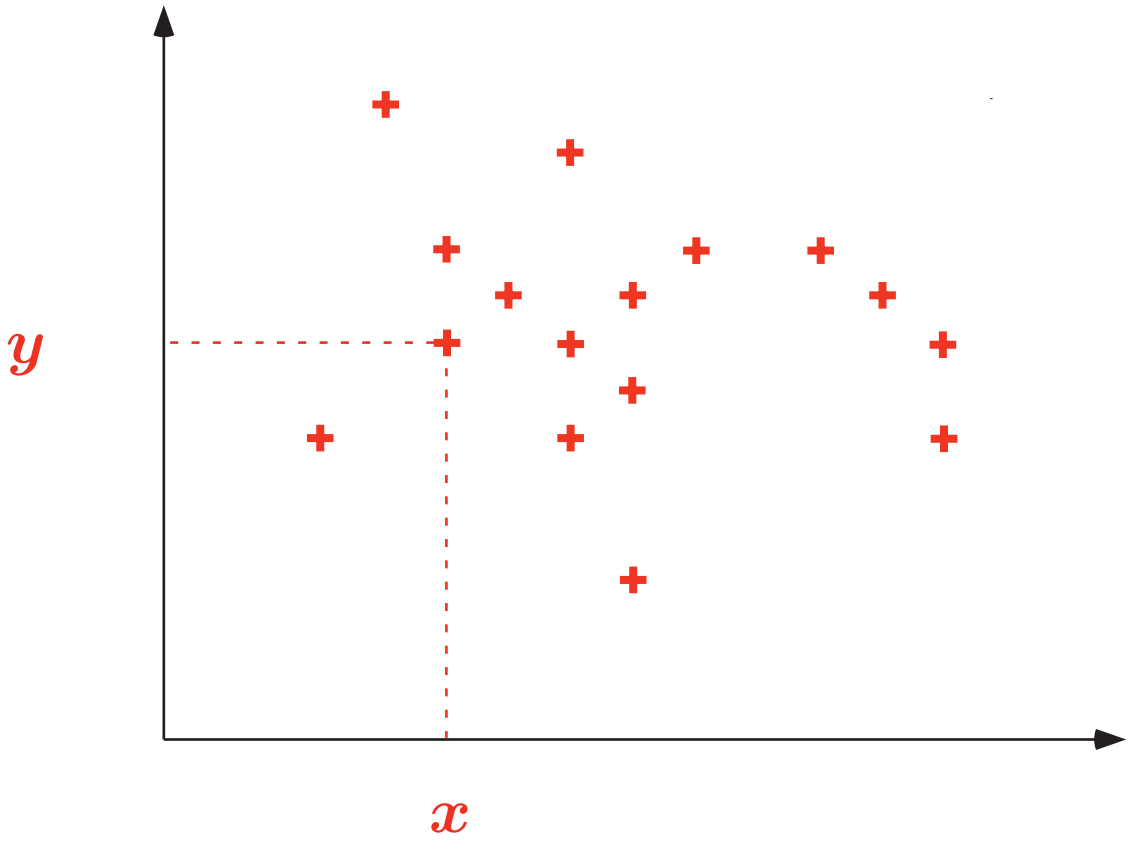
– Si l'approximation est suffisamment grossière, l'abstraction d'une sémantique peut permettre d'en donner une version moins précise mais calculable par ordinateur ;

Approximations calculables

– Si l'approximation est suffisamment grossière, l'abstraction d'une sémantique peut permettre d'en donner une version moins précise mais calculable par ordinateur ;

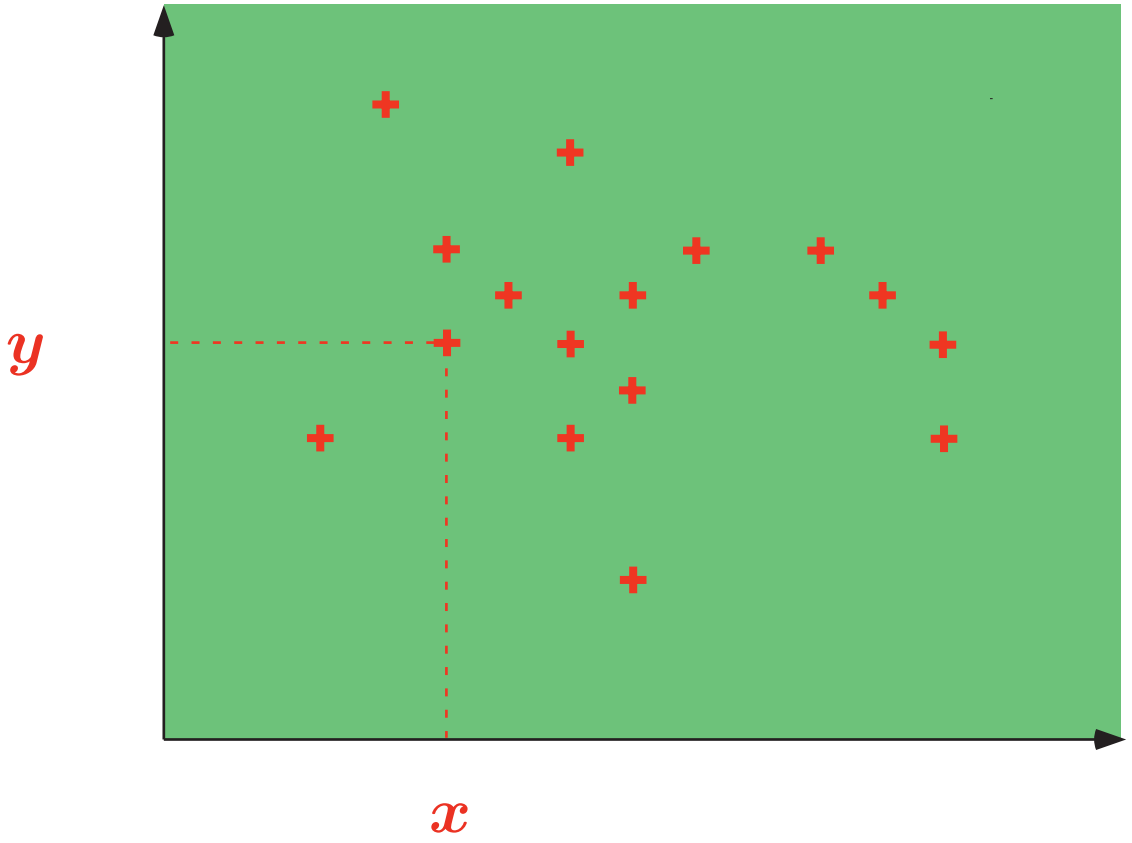
– Par calcul effectif de la sémantique abstraite, l'ordinateur est capable d'analyser le comportement de programmes et de logiciels avant même de les exécuter.

Exemple d'approximations calculables d'un ensemble [in]fini de points



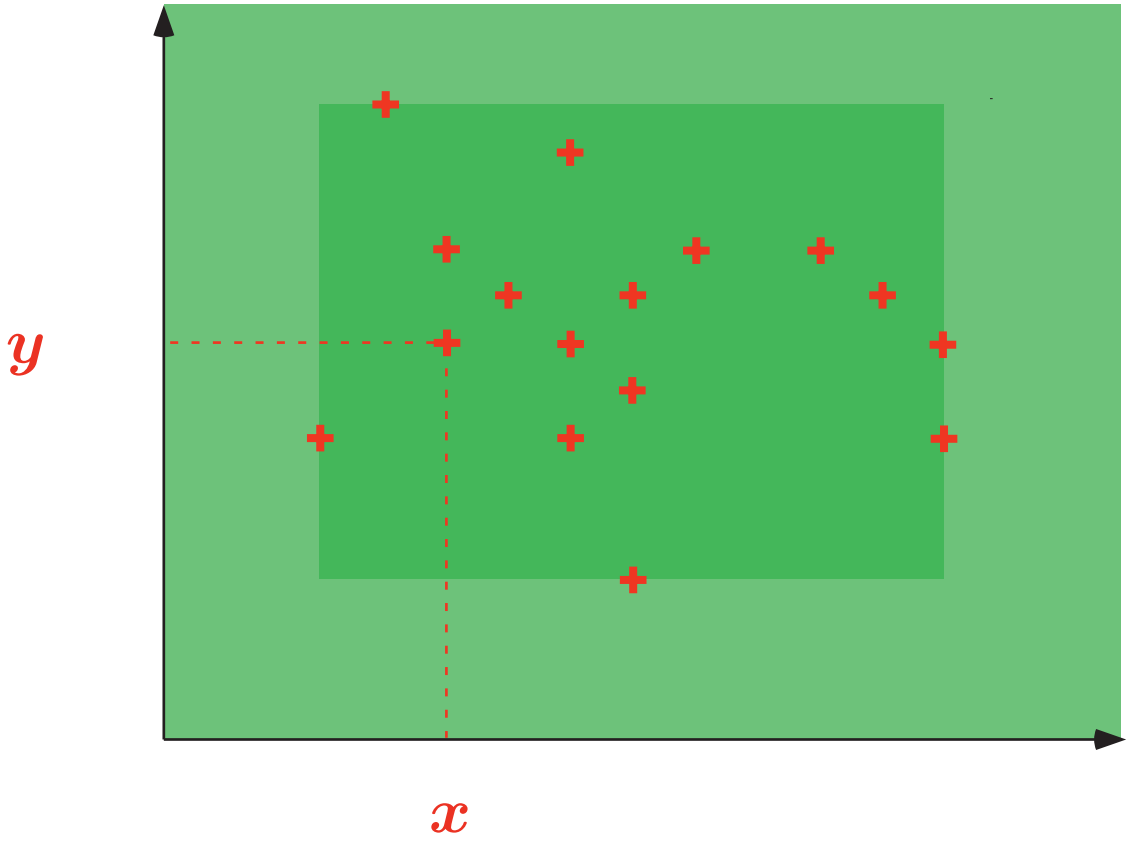
$$\{\ldots,\langle 19,\, 88\rangle,\ldots,\langle 19,\, 99\rangle,\ldots\}$$

Exemple d'approximations calculables d'un ensemble [in]fini de points (signes)



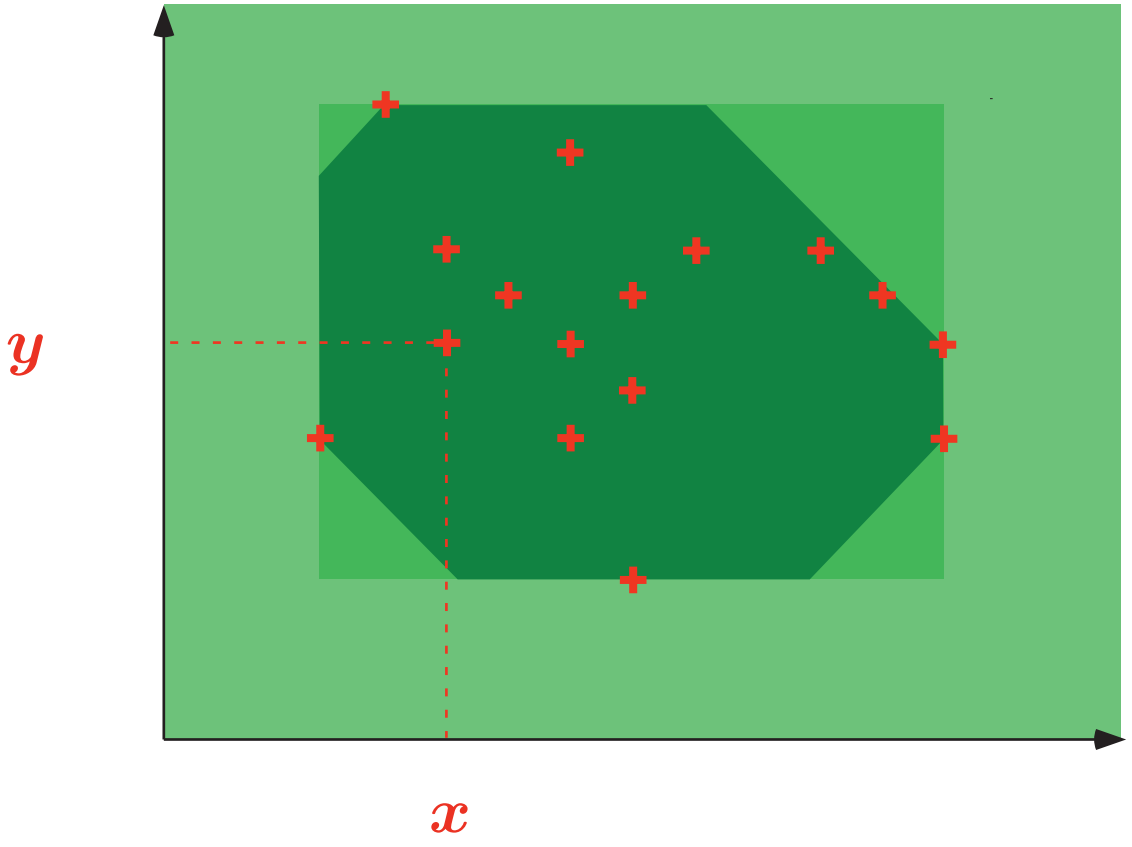
$$\begin{cases} x \geq 0 \\ y \geq 0 \end{cases}$$

Exemple d'approximations calculables d'un ensemble [in]fini de points (intervalles)



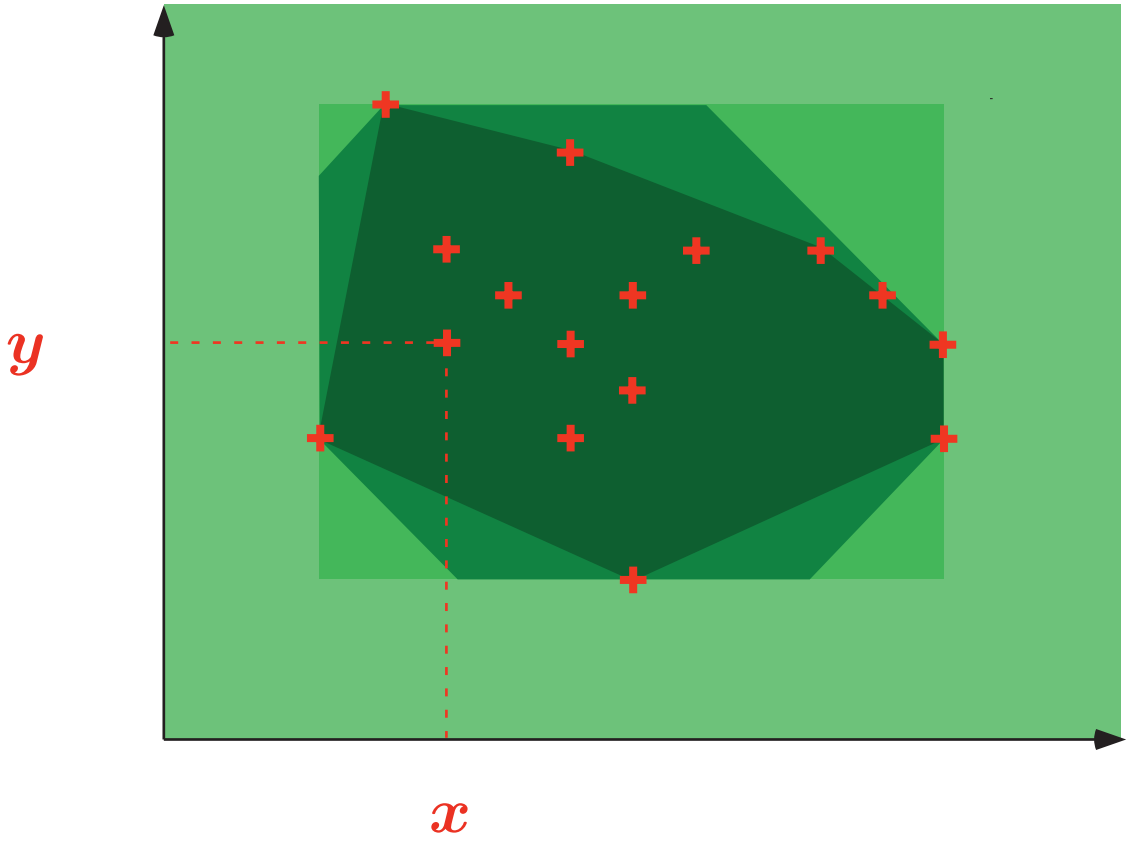
$$\begin{cases} x \in [19, 88] \\ y \in [19, 99] \end{cases}$$

Exemple d'approximations calculables d'un ensemble [in]fini de points (octogones)



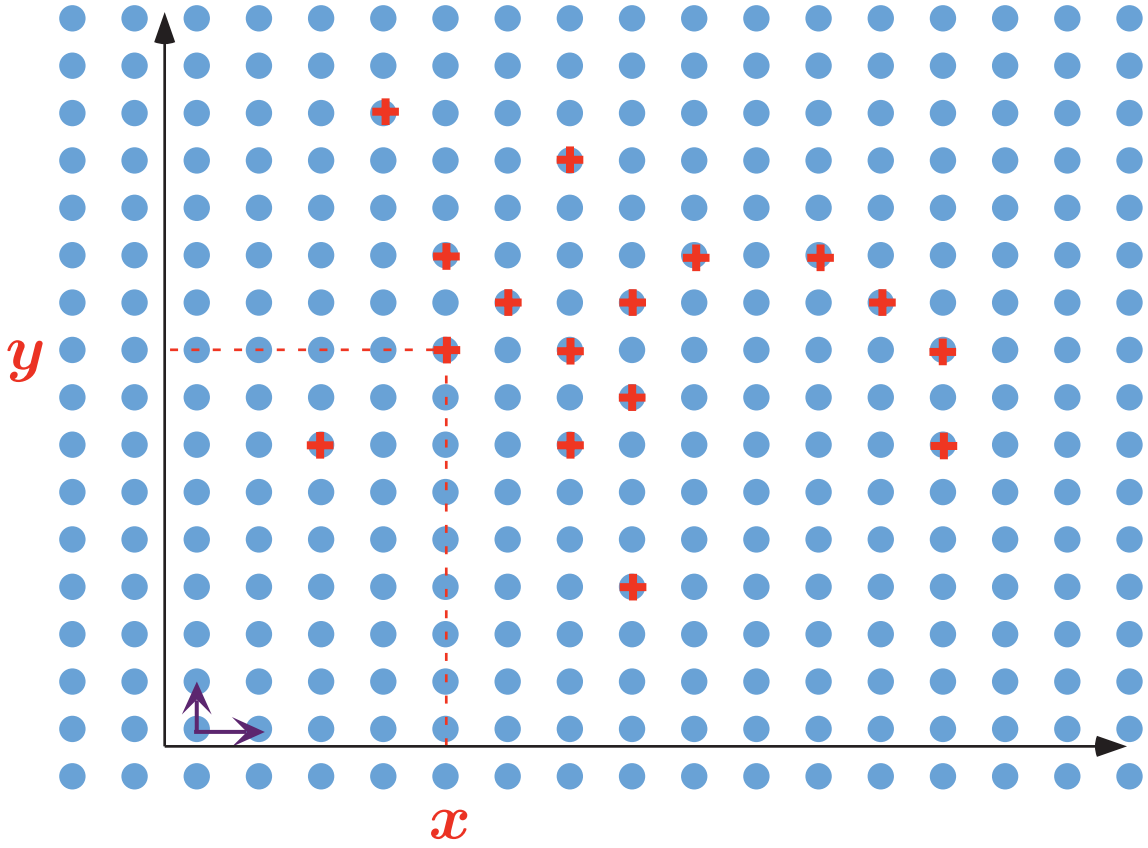
$$\begin{cases} 1 \leq x \leq 9 \\ x + y \leq 88 \\ 1 \leq y \leq 9 \\ x - y \leq 99 \end{cases}$$

Exemple d'approximations calculables d'un ensemble [in]fini de points (polyèdres)



$$\begin{cases} 19x + 88y \leq 2000 \\ 19x + 99y \geq 0 \end{cases}$$

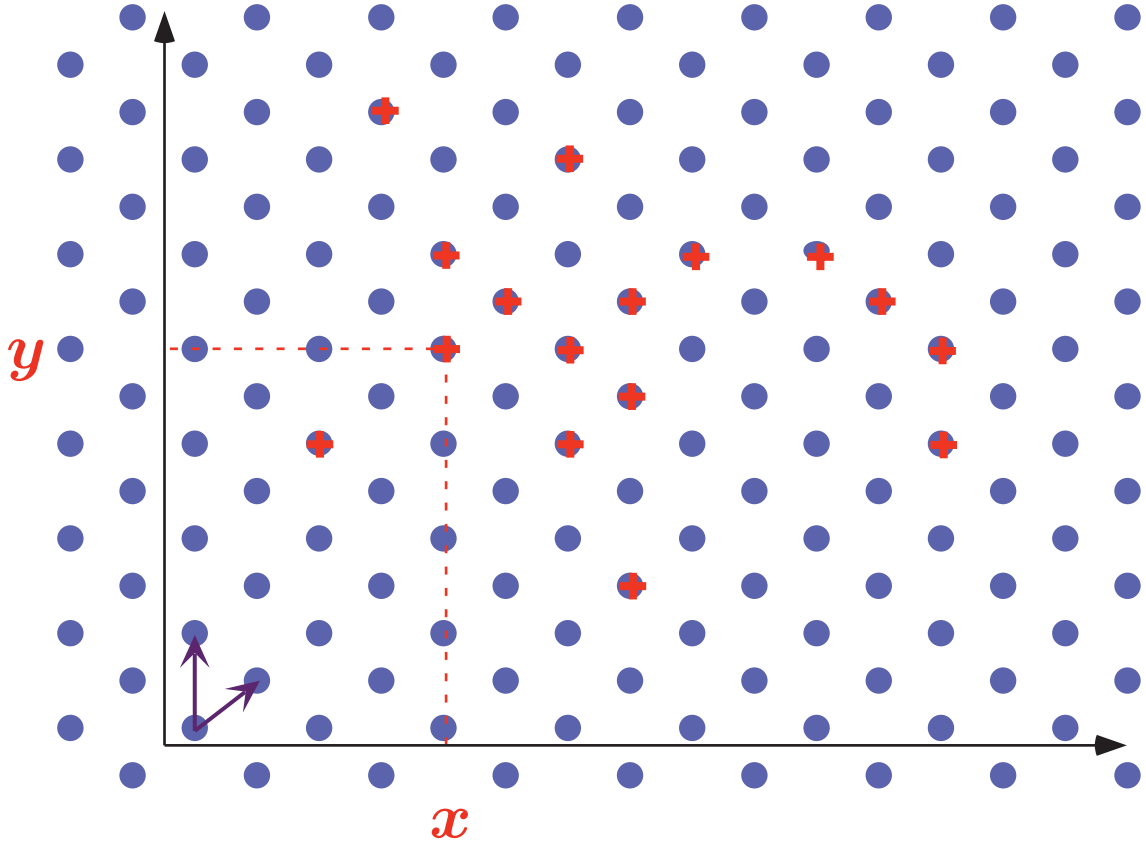
Exemple d'approximations calculables d'un ensemble [in]fini de points (congruences simples)



$$\begin{cases} x = 19 \bmod 88 \\ y = 19 \bmod 99 \end{cases}$$

thèse P. Granger, LIX, 1991

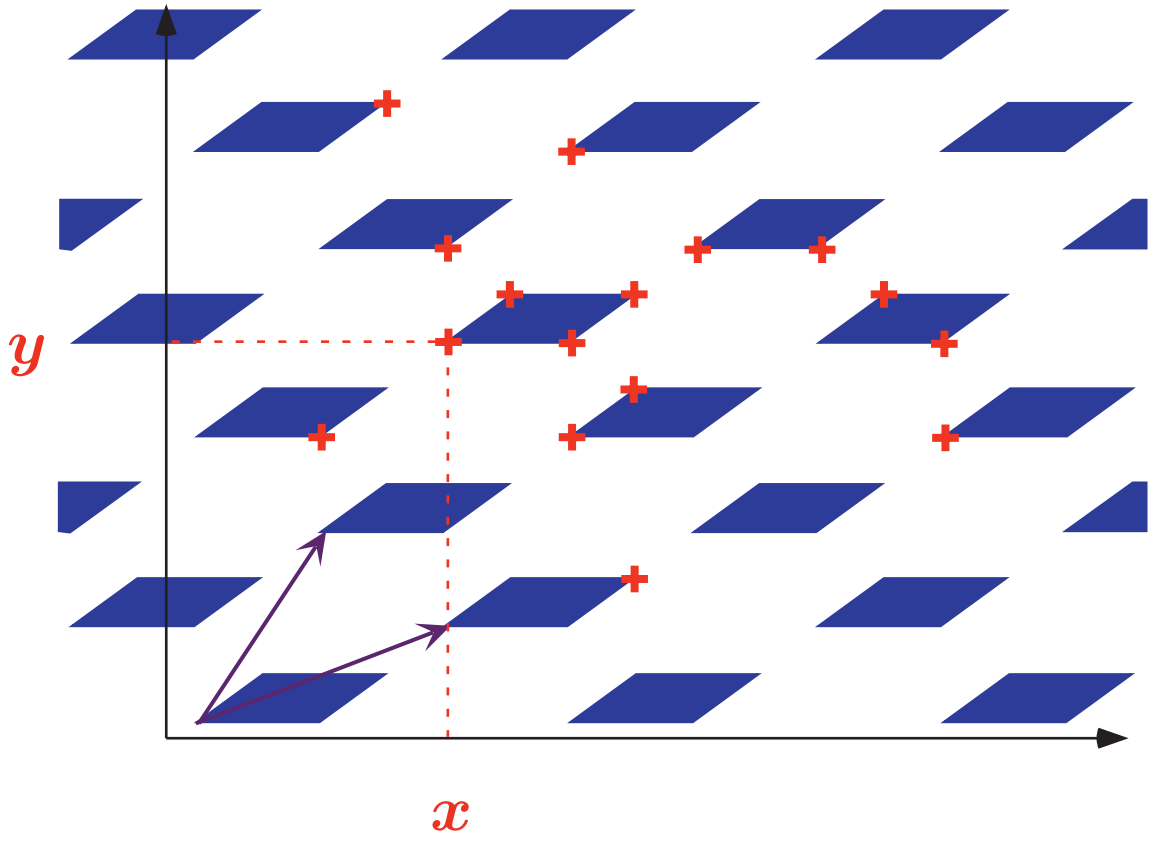
Exemple d'approximations calculables d'un ensemble [in]fini de points (congruences linéaires)



$$\begin{cases} 1x + 9y = 8 \bmod 8 \\ 1x - 9y = 9 \bmod 9 \end{cases}$$

thèse P. Granger, LIX, 1991

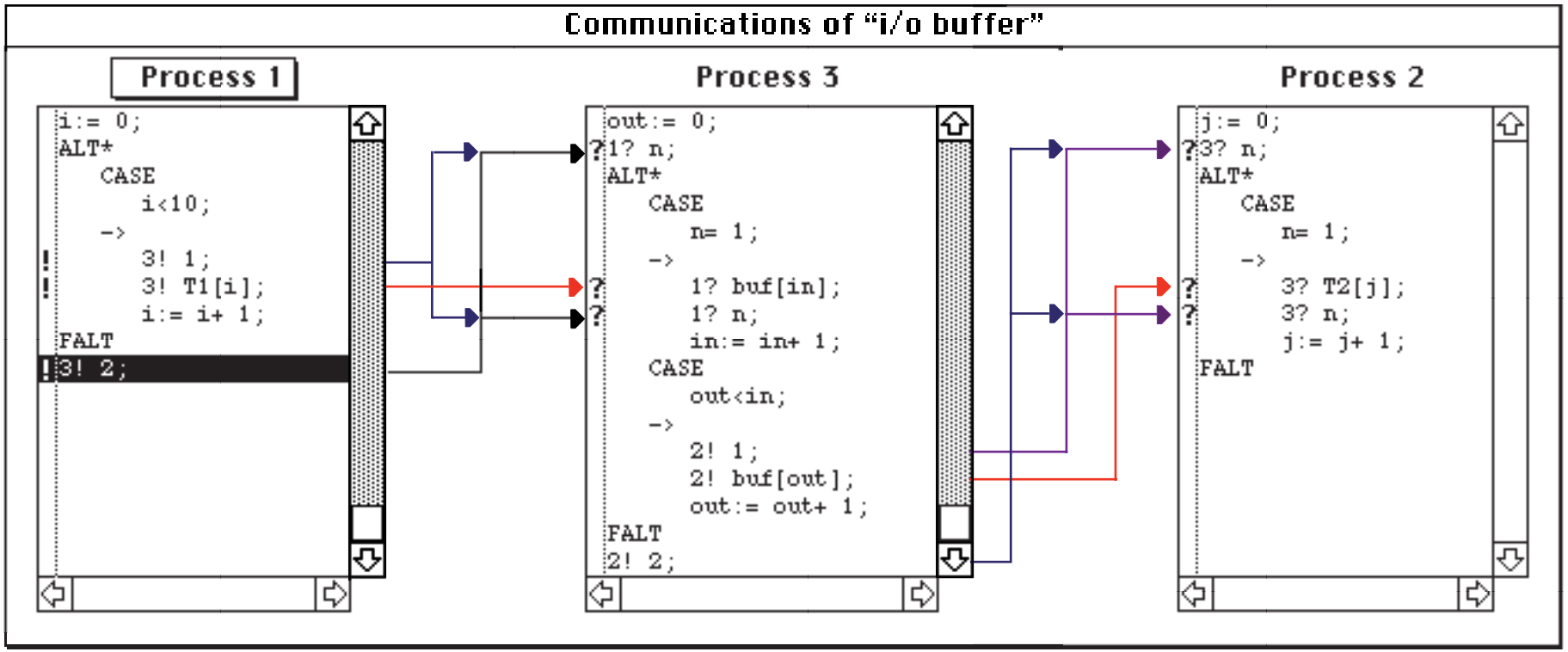
Exemple d'approximations calculables d'un ensemble [in]fini de points (congruences linéaires trapézoïdales)



$$\begin{cases} 1x + 9y \in [0,88] \bmod 10 \\ 1x - 9y \in [0,99] \bmod 11 \end{cases}$$

thèse F. Masdupuy, LIX, 1993

Exemple d'application des congruences : analyse de communications en OCCAM



thèse N. Mercouroff, LIX, 1990

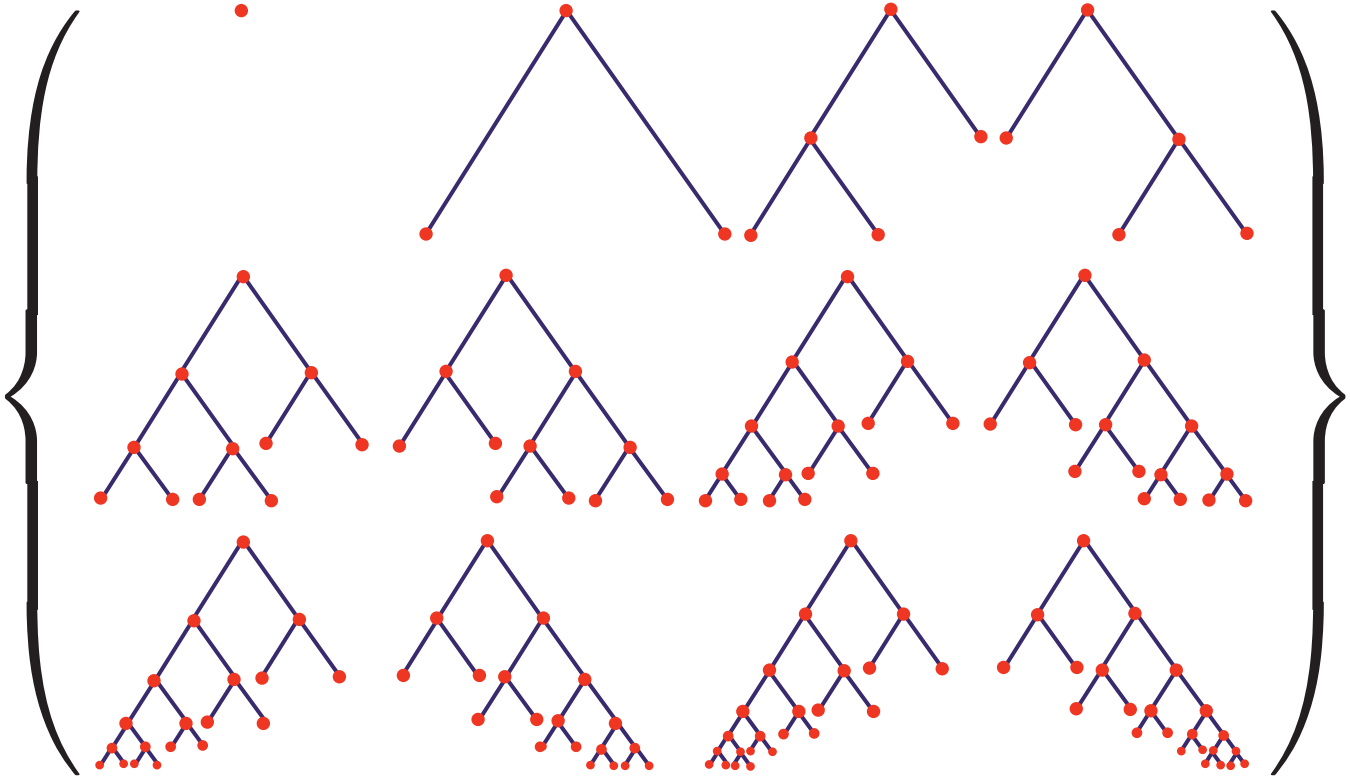
Plus difficile: structures non numériques

– La plupart des structures manipulées par les programmes ne sont pas numériques ;

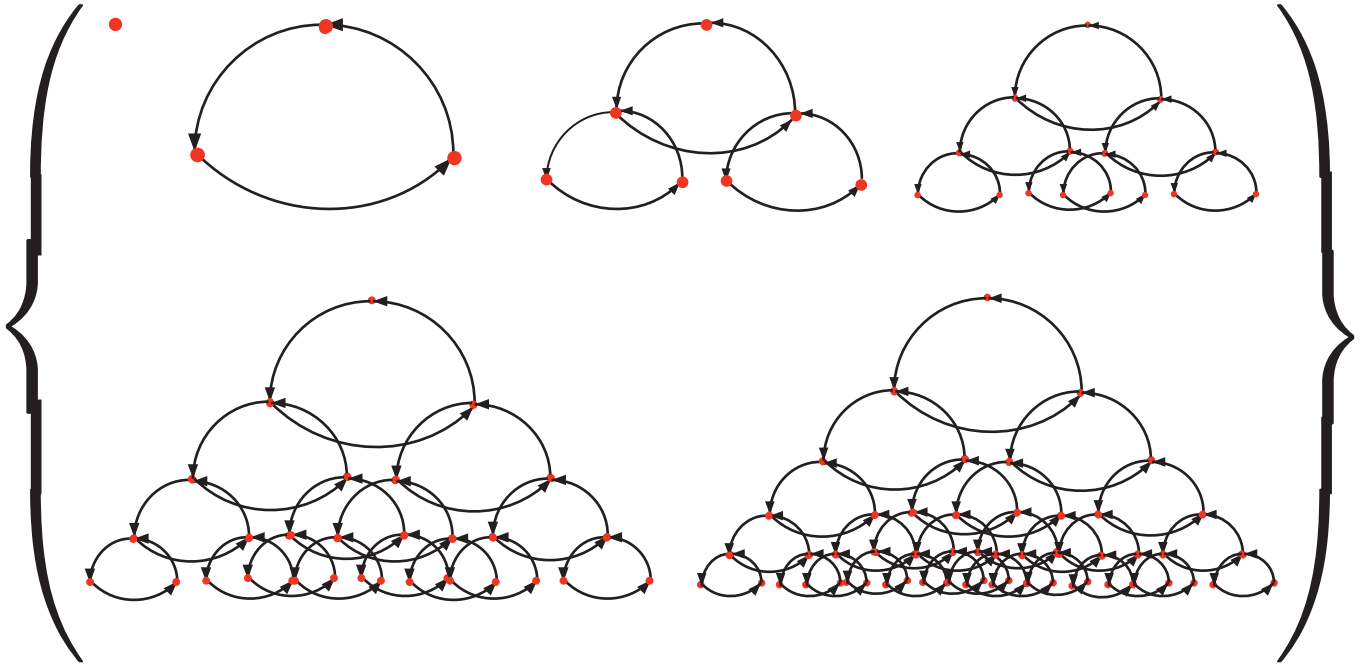
Plus difficile: structures non numériques

– La plupart des structures manipulées par les programmes ne sont pas numériques ;
– C'est le cas par exemple des structures :
  – de calcul (graphes d'appel, arbres de récursivité),
  – de données (arbres de recherche),
  – de communication (programmes distribués),
  – de transfert d'information (programmes mobiles), etc.

Exemple 1 : ensembles (infinis) d'arbres (infinis)



Exemple 2 : ensembles (infinis) de graphes (infinis) décorés



Approximations compactes précises

– Il est très difficile de trouver des représentations informatiques compactes de tels ensembles d'objets (langages, automates, arbres, graphes, etc.)

Approximations compactes précises

– Il est très difficile de trouver des représentations informatiques compactes de tels ensembles d'objets (langages, automates, arbres, graphes, etc.) telles que :
  – les diverses opérations ensemblistes sont implantées efficacement;

Approximations compactes précises

– Il est très difficile de trouver des représentations informatiques compactes de tels ensembles d'objets (langages, automates, arbres, graphes, etc.) telles que :
  – les diverses opérations ensemblistes sont implantées efficacement;
  – la taille mémoire n'explose pas combinatoirement pour des ensembles complexes ou irréguliers;

Approximations compactes précises

– Il est très difficile de trouver des représentations informatiques compactes de tels ensembles d'objets (langages, automates, arbres, graphes, etc.) telles que :
  – les diverses opérations ensemblistes sont implantées efficacement;
  – la taille mémoire n'explose pas combinatoirement pour des ensembles complexes ou irréguliers;
  – les approximations sont précises.

Approximations compactes précises

– Il est très difficile de trouver des représentations informatiques compactes de tels ensembles d'objets (langages, automates, arbres, graphes, etc.) telles que :
  – les diverses opérations ensemblistes sont implantées efficacement;
  – la taille mémoire n'explose pas combinatoirement pour des ensembles complexes ou irréguliers;
  – les approximations sont précises.

thèses I. Stransky, LIX, 1988, A. Deutsch, LIX, 1992, A. Venet, LIX, 1998, L. Mauborgne, X, 1999

2.3 Analyse statique

Difficultés de la programmation

– La programmation des ordinateurs est difficile ;

Difficultés de la programmation

– La programmation des ordinateurs est difficile ;

– Les raisonnements sur les programmes sont difficiles ;

Difficultés de la programmation

– La programmation des ordinateurs est difficile ;
– Les raisonnements sur les programmes sont difficiles ;
– Les erreurs sont fréquentes.

Exemple 1 : composition d'informatique en tronc commun à l'École polytechnique

Que fait ce programme PASCAL :

```
program P (input, output);
 procedure AlaLigne; begin writeln end;
 procedure P (X : integer; procedure Q);
  procedure R;
   begin write(X); Q; end;
 begin
  if X > 0 then begin R; P(X - 1, R); end;
 end;
begin
 P(5, AlaLigne);
end.
```

Exemple 1 : composition d'informatique en tronc commun à l'École polytechnique

Que fait ce programme PASCAL :

```
program P (input, output);
 procedure AlaLigne; begin writeln end;
 procedure P (X : integer; procedure Q);
  procedure R;
   begin write(X); Q; end;
 begin
  if X > 0 then begin R; P(X - 1, R); end;
 end;
begin
 P(5, AlaLigne);
end.
```

```
5
4   5
3   4   5
2   3   4   5
1   2   3   4   5
```

Moins de 5% des réponses sont correctes !

Exemple 2 : composition d'informatique en tronc commun à l'École polytechnique

Prouver que ce programme imprime une valeur ≥ 91:

```
program MacCarthy  (input,output);
 var x, m : integer;
 function MC(n : integer) : integer;
  begin
   if n > 100 then MC := n - 10
   else MC := MC(MC(n + 11));
  end;
begin
 read(x); m := MC(x); writeln(m);
end.
```

Exemple 2 : composition d'informatique en tronc commun à l'École polytechnique

Prouver que ce programme imprime une valeur ≥ 91:

```
program MacCarthy  (input,output);
 var x, m : integer;
 function MC(n : integer) : integer;
  begin
   if n > 100 then MC := n - 10
   else MC := MC(MC(n + 11));
  end;
begin
 read(x); m := MC(x); writeln(m);
end.
```

Plus de 50 % des preuves données en réponse sont incorrectes !

Analyse statique

– Objectif : découvrir les erreurs de programmation avant qu'elles ne produisent des catastrophes!

Analyse statique

– Objectif : découvrir les erreurs de programmation avant qu'elles ne produisent des catastrophes!

– L'analyse statique utilise l'*interprétation abstraite* pour dériver de la sémantique standard une sémantique calculable par l'ordinateur ;

Analyse statique

– Objectif : découvrir les erreurs de programmation avant qu'elles ne produisent des catastrophes!
– L'analyse statique utilise l'*interprétation abstraite* pour dériver de la sémantique standard une sémantique calculable par l'ordinateur ;
– De ce fait un ordinateur est capable d'analyser le comportement de logiciels avant même de les exécuter ;

Analyse statique

– Objectif : découvrir les erreurs de programmation avant qu'elles ne produisent des catastrophes!
– L'analyse statique utilise l'*interprétation abstraite* pour dériver de la sémantique standard une sémantique calculable par l'ordinateur ;
– De ce fait un ordinateur est capable d'analyser le comportement de logiciels avant même de les exécuter ;
– Ceci est essentiel pour les systèmes informatiques critiques (par exemple : avions, fusées, centrales nucléaires, etc.).

Exemple : analyse d'intervalles (1975)

Programme à analyser :

```
   x := 1;
1:
   while x <10000 do
2:
        x := x + 1
3:
   od;
4:
```

Exemple : analyse d'intervalles (1975)

Équations (interprétation abstraite de la sémantique) :

```
   x := 1;
1:
   while x <10000 do
2:
      x := x + 1
3:
   od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

Exemple : analyse d'intervalles (1975)

Résolution itérative chaotique croissante :

```
   x := 1;
1:
   while x <10000 do
2:
        x := x + 1
3:
   od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

$$\begin{cases} X_1 = \emptyset \\ X_2 = \emptyset \\ X_3 = \emptyset \\ X_4 = \emptyset \end{cases}$$

Exemple : analyse d'intervalles (1975)

Itération chaotique croissante :

```
   x := 1;
1:
   while x <10000 do
2:
        x := x + 1
3:
   od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

$$\begin{cases} X_1 = [1,1] \\ X_2 = \emptyset \\ X_3 = \emptyset \\ X_4 = \emptyset \end{cases}$$

Exemple : analyse d'intervalles (1975)

Itération chaotique croissante :

```
   x := 1;
1:
   while x <10000 do
2:
        x := x + 1
3:
   od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

$$\begin{cases} X_1 = [1,1] \\ X_2 = [1,1] \\ X_3 = \emptyset \\ X_4 = \emptyset \end{cases}$$

Exemple : analyse d'intervalles (1975)

Itération chaotique croissante :

```
    x := 1;
1:
    while x <10000 do
2:
        x := x + 1
3:
    od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

$$\begin{cases} X_1 = [1,1] \\ X_2 = [1,1] \\ X_3 = [2,2] \\ X_4 = \emptyset \end{cases}$$

Exemple : analyse d'intervalles (1975)

Itération chaotique croissante :

```
    x := 1;
1:
    while x <10000 do
2:
        x := x + 1
3:
    od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

$$\begin{cases} X_1 = [1,1] \\ X_2 = [1,2] \\ X_3 = [2,2] \\ X_4 = \emptyset \end{cases}$$

Exemple : analyse d'intervalles (1975)

Itération chaotique croissante : **convergence !**

```
    x := 1;
1:
    while x <10000 do
2:
        x := x + 1
3:
    od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

$$\begin{cases} X_1 = [1,1] \\ X_2 = [1,2] \\ X_3 = [2,3] \\ X_4 = \emptyset \end{cases}$$

Exemple : analyse d'intervalles (1975)

Itération chaotique croissante : **convergence !!**

```
    x := 1;
1:
    while x <10000 do
2:
        x := x + 1
3:
    od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

$$\begin{cases} X_1 = [1,1] \\ X_2 = [1,3] \\ X_3 = [2,3] \\ X_4 = \emptyset \end{cases}$$

Exemple : analyse d'intervalles (1975)

Itération chaotique croissante : **convergence !!!**

```
   x := 1;
1:
   while x <10000 do
2:
        x := x + 1
3:
   od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

$$\begin{cases} X_1 = [1,1] \\ X_2 = [1,3] \\ X_3 = [2,4] \\ X_4 = \emptyset \end{cases}$$

Exemple : analyse d'intervalles (1975)

Itération chaotique croissante : **convergence !!!!**

```
   x := 1;
1:
   while x <10000 do
2:
        x := x + 1
3:
   od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

$$\begin{cases} X_1 = [1,1] \\ X_2 = [1,4] \\ X_3 = [2,4] \\ X_4 = \emptyset \end{cases}$$

Exemple : analyse d'intervalles (1975)

Itération chaotique croissante : **convergence !!!!!**

```
    x := 1;
1:
    while x <10000 do
2:
        x := x + 1
3:
    od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

$$\begin{cases} X_1 = [1,1] \\ X_2 = [1,4] \\ X_3 = [2,5] \\ X_4 = \emptyset \end{cases}$$

Exemple : analyse d'intervalles (1975)

Itération chaotique croissante : **convergence !!!!!!**

```
   x := 1;
1:
   while x <10000 do
2:
        x := x + 1
3:
   od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

$$\begin{cases} X_1 = [1,1] \\ X_2 = [1,5] \\ X_3 = [2,5] \\ X_4 = \emptyset \end{cases}$$

Exemple : analyse d'intervalles (1975)

Itération chaotique croissante : **convergence !!!!!!!**

```
    x := 1;
1:
    while x <10000 do
2:
        x := x + 1
3:
    od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

$$\begin{cases} X_1 = [1,1] \\ X_2 = [1,5] \\ X_3 = [2,6] \\ X_4 = \emptyset \end{cases}$$

Exemple : analyse d'intervalles (1975)

Accélération de la convergence par extrapolation :

```
   x := 1;
1:
   while x <10000 do
2:
        x := x + 1
3:
   od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

$$\begin{cases} X_1 = [1,1] \\ X_2 = [1,+\infty] \\ X_3 = [2,6] \\ X_4 = \emptyset \end{cases} \Leftarrow \text{élargissement}$$

Exemple : analyse d'intervalles (1975)

Itération chaotique décroissante :

```
    x := 1;
1:
    while x <10000 do
2:
        x := x + 1
3:
    od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

$$\begin{cases} X_1 = [1,1] \\ X_2 = [1,+\infty] \\ X_3 = [2,+\infty] \\ X_4 = \emptyset \end{cases}$$

Exemple : analyse d'intervalles (1975)

Itération chaotique décroissante :

```
    x := 1;
1:
    while x <10000 do
2:
        x := x + 1
3:
    od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

$$\begin{cases} X_1 = [1,1] \\ X_2 = [1,9999] \\ X_3 = [2,+\infty] \\ X_4 = \emptyset \end{cases}$$

Exemple : analyse d'intervalles (1975)

Itération chaotique décroissante :

```
    x := 1;
1:
    while x <10000 do
2:
        x := x + 1
3:
    od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

$$\begin{cases} X_1 = [1,1] \\ X_2 = [1,9999] \\ X_3 = [2,+10000] \\ X_4 = \emptyset \end{cases}$$

Exemple : analyse d'intervalles (1975)

Solution finale :

```
   x := 1;
1:
   while x <10000 do
2:
        x := x + 1
3:
   od;
4:
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000,+\infty] \end{cases}$$

$$\begin{cases} X_1 = [1,1] \\ X_2 = [1,9999] \\ X_3 = [2,+10000] \\ X_4 = [+10000,+10000] \end{cases}$$

Exemple : analyse d'intervalles (1975)

Résultat de l'analyse d'intervalles :

```
    x := 1;
1: {x = 1}
    while x <10000 do
2: {x ∈ [1,9999]}
        x := x + 1
3: {x ∈ [2, + 10000]}
    od;
4: {x = 10000}
```

$$\begin{cases} X_1 = [1,1] \\ X_2 = (X_1 \cup X_3) \cap [-\infty,9999] \\ X_3 = X_2 \oplus [1,1] \\ X_4 = (X_1 \cup X_3) \cap [10000, +\infty] \end{cases}$$

$$\begin{cases} X_1 = [1,1] \\ X_2 = [1,9999] \\ X_3 = [2, + 10000] \\ X_4 = [+10000, + 10000] \end{cases}$$

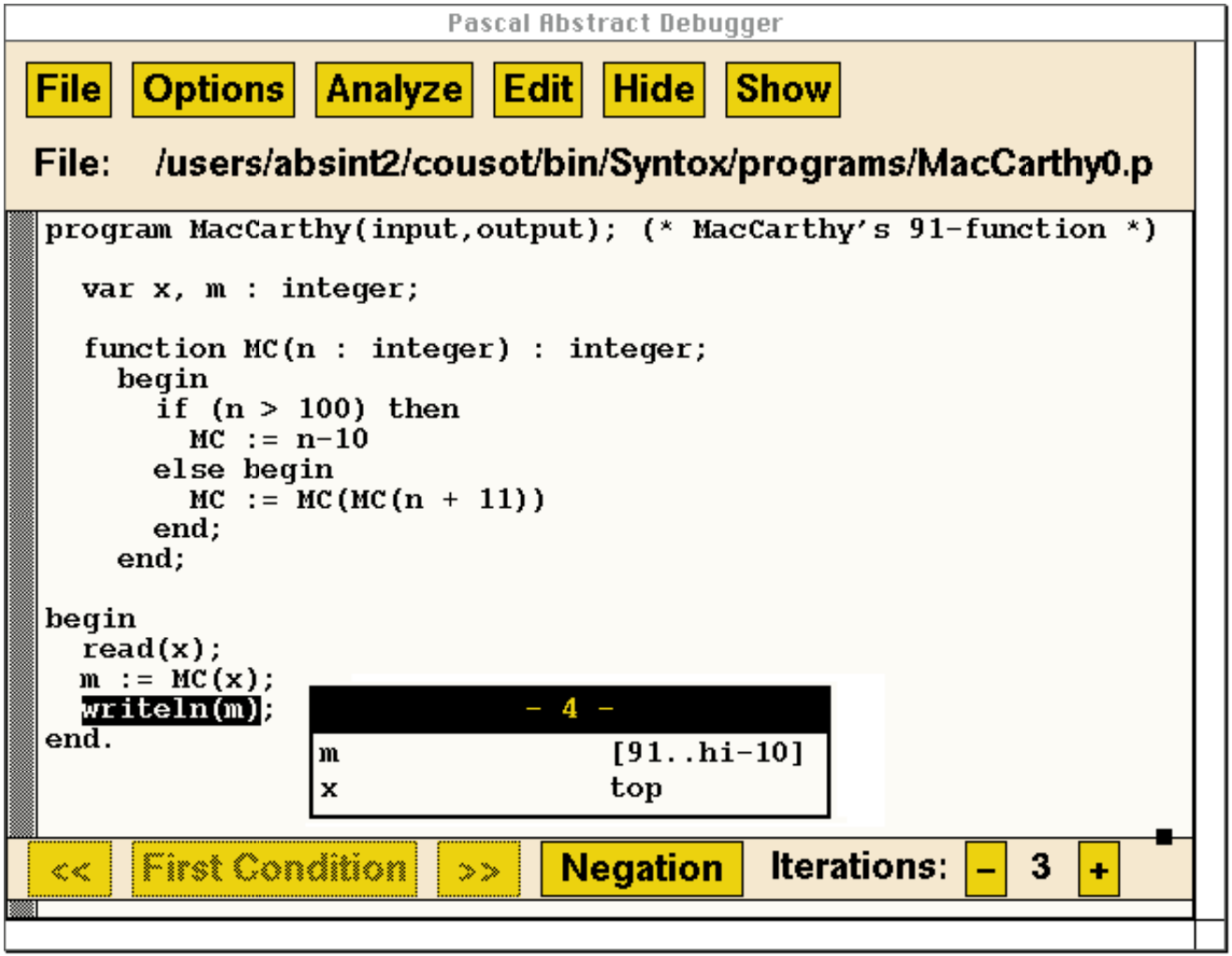
Exemple : analyse d'intervalles (1975)

Exploitation des résultats de l'analyse d'intervalles :

```
    x := 1;
1: {x = 1}
    while x <10000 do
2: {x ∈ [1,9999]}
        x := x + 1
3: {x ∈ [2, +10000]}
    od;
4: {x = 10000}
```

⟵ **pas de débordement**

Pour des langages impératifs comme PASCAL ...



thèse F. Bourdoncle, X, 1992

Application (1996/97)

– A. Deutsch [2] utilise l'interprétation abstraite (dont l'analyse d'intervalles) pour l'analyse statique des logiciels du programme de vol et de la centrale inertielle du lanceur Ariane 5 ;

– Réussite pour le vol 502 et l'ARD.

2. thèse au LIX sur l'interprétation abstraite en 1992.

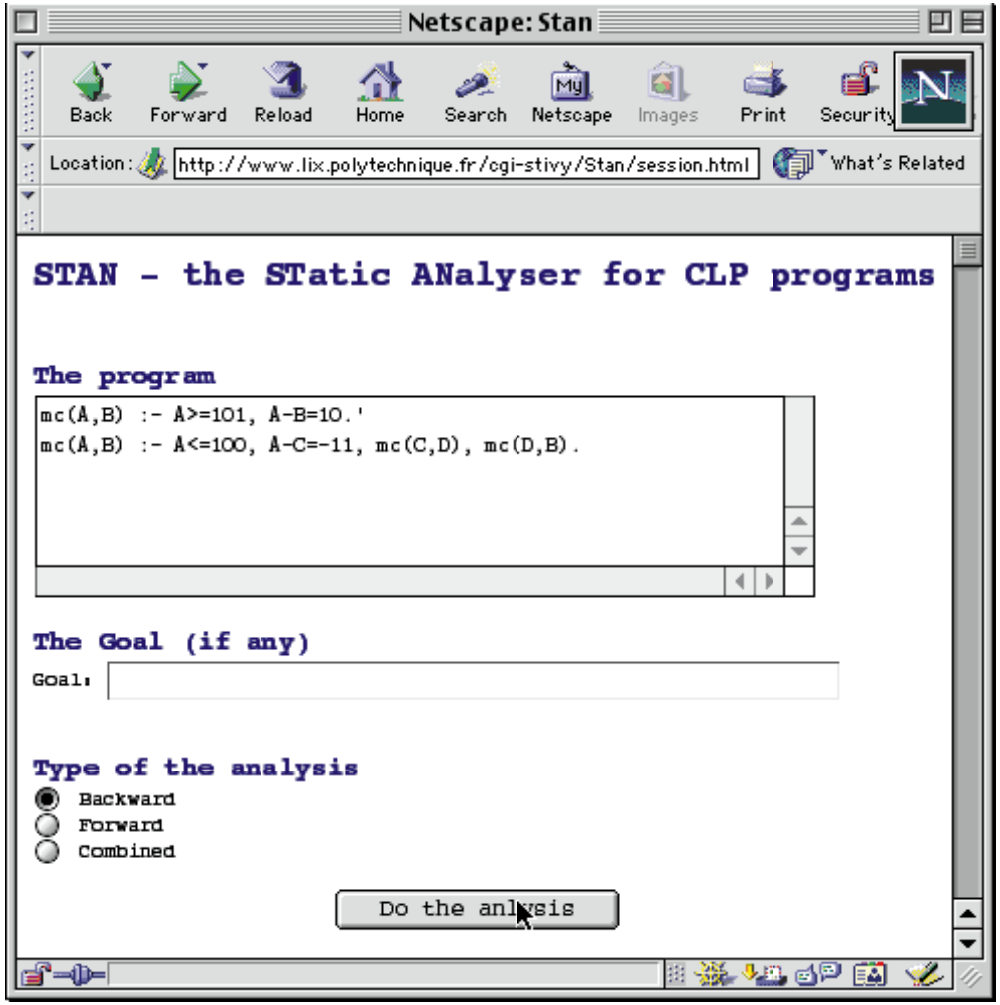
Quelques autres applications récentes de l'analyse statique par interprétation abstraite

– transformation & optimisation de programmes ;
– test abstrait ;
– inférence de types (systèmes indécidables) ;
– code mobile ;
– « model-checking [3] » abstrait de systèmes infinis ;
– différenciation automatique ;
– ...

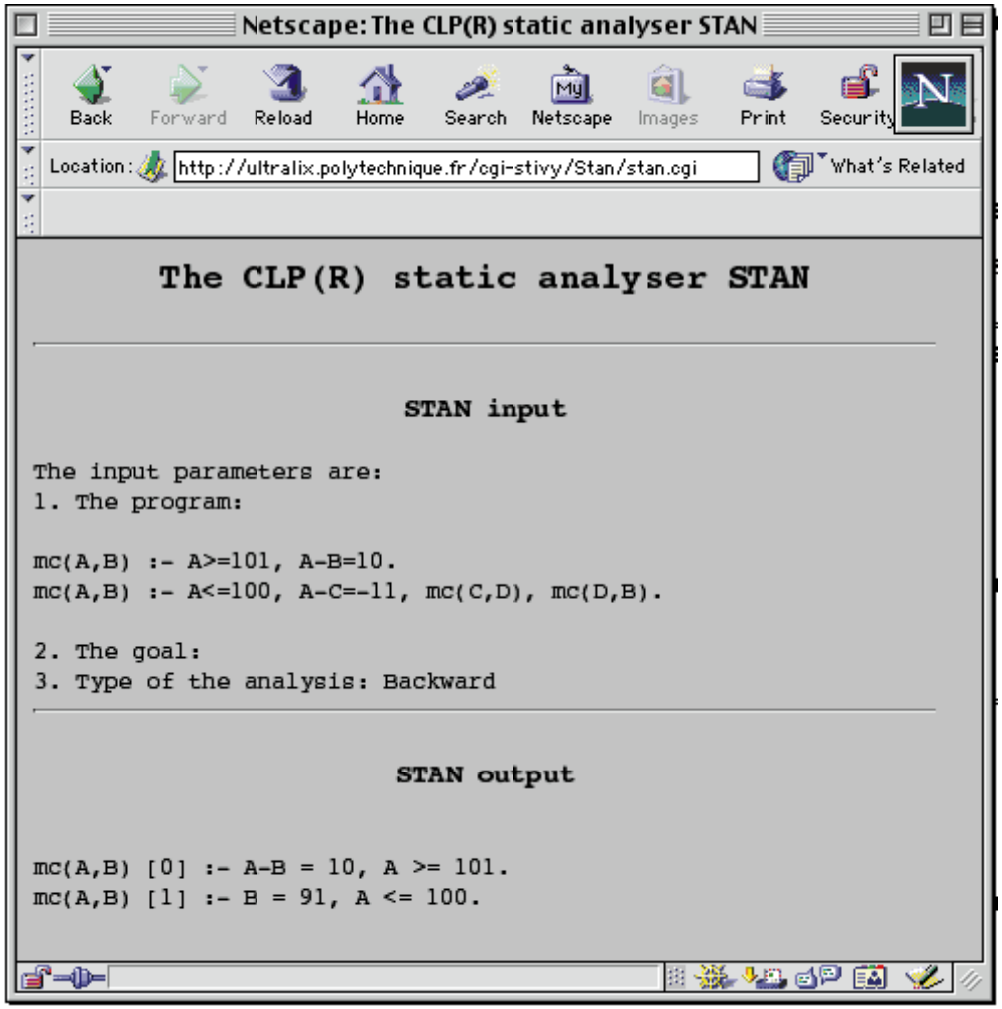
thèses F. Bourdoncle, X, 1992, B. Monsuez, X, 1994,
A. Venet, LIX, 1998, R. Cridlig, X, 1999

3. vérification de modèle.

Exemple d'application de l'analyse statique à la transformation & optimisation de programmes



Exemple d'application de l'analyse statique à la transformation & optimisation de programmes

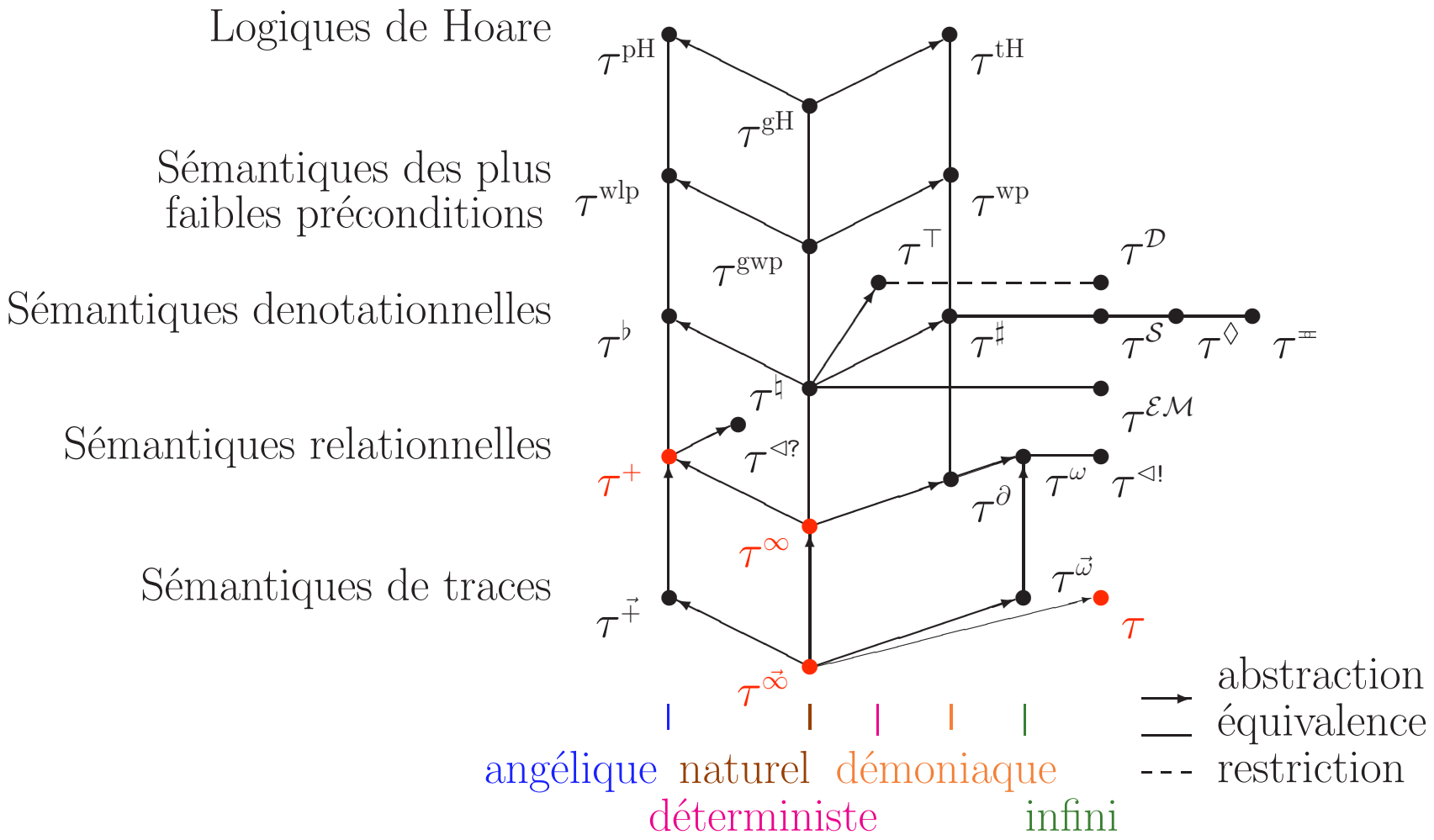


Quelques autres applications récentes de l'interprétation abstraite

– fondamentales :
  – conception de hiérarchies de sémantiques,
  – ...;
– appliquées :
  – sécurité (analyse de protocoles cryptographiques),
  – tatouage sémantique de logiciels,
  – « data mining [4] » ,
  – ....

4. fouille de données.

Treillis des sémantiques



Recherches à venir

Il nous reste beaucoup à faire sur le plan fondamental :

- ordre supérieur,
- modularité,
- flottants,
- analyses probabilistes,
- propriétés de vivacité avec équité,
- ...;

Quelques références

Pour débuter :

P. Cousot. Abstract interpretation. *ACM Computing Surveys* 28 (2), 1996, 324–328.

Sur la toile :

`http://www.di.ens.fr/~cousot/`

Industrialisation de l'analyse statique par interprétation abstraite

- Premières recherches : 1975 ;
- Premières industrialisations :
  -  Connected Components Corporation (U.S.A.), L. Harrison, 1993 ;
  -  AbsInt Angewandte Informatik GmbH (Allemagne), R. Wilhelm, 1998 ;
  - Polyspace Technologies (France), A. Deutsch & D. Pilaud, 1999.

En guise de conclusion

– Les problèmes fondamentaux de l'informatique sont difficiles à expliquer (seules les applications sont comprises) ;

En guise de conclusion

– Les problèmes fondamentaux de l'informatique sont difficiles à expliquer (seules les applications sont comprises) ;
– La société prendra certainement conscience de ces problèmes dans le futur (par exemple, à ses dépends au travers de catastrophes) ;

En guise de conclusion

– Les problèmes fondamentaux de l'informatique sont difficiles à expliquer (seules les applications sont comprises) ;
– La société prendra certainement conscience de ces problèmes dans le futur (par exemple, à ses dépends au travers de catastrophes) ;
– La recherche d'idées fondamentales sur le développement de logiciels est essentielle pour l'avenir ;

En guise de conclusion

– Les problèmes fondamentaux de l'informatique sont difficiles à expliquer (seules les applications sont comprises) ;
– La société prendra certainement conscience de ces problèmes dans le futur (par exemple, à ses dépends au travers de catastrophes) ;
– La recherche d'idées fondamentales sur le développement de logiciels est essentielle pour l'avenir ;
– Leurs applications peuvent difficilement se programmer à court terme (3 ans) ;

En guise de conclusion

- Les problèmes fondamentaux de l'informatique sont difficiles à expliquer (seules les applications sont comprises) ;
- La société prendra certainement conscience de ces problèmes dans le futur (par exemple, à ses dépends au travers de catastrophes) ;
- La recherche d'idées fondamentales sur le développement de logiciels est essentielle pour l'avenir ;
- Leurs applications peuvent difficilement se programmer à court terme (3 ans) ;
- Il faut donner des moyens convenables aux informaticiens.