

Sémantique des systèmes concurrents

Stage de fin d'année

Jordan Ischard

Université d'Orléans

06 juin 2019

- 1 Introduction
 - Sujet du stage
 - Présentation de l'équipe
- 2 λ -calculs
 - Bases
 - Forme normale
 - Stratégie de réduction
- 3 Machines étudiées
 - Machine CC
 - Machine CK
 - Machine SCC
 - Machine CEK
 - Machine SECD
- 4 Programmation réactive synchrone concurrente
 - Concept
- 5 Machine TTS
 - Explication
 - Sémantique de la machine
 - Sémantique de la machine
- 6 Machine TTSI
 - Explication informelle
 - Sémantique de la machine
- 7 Conclusion
 - Résumer
 - Travaux futures

Sujet du stage : **Programmation réactive synchrone et Implantation d'une machine virtuelle**

Encadrants : **Madame Bousdira** et **Monsieur Dabrowski**

LIFO : Le Laboratoire d'Informatique Fondamentale d'Orléans (LIFO) est un laboratoire de l'Université d'Orléans et de l'INSA Centre-Val de Loire. Les recherches menées au LIFO concernent la science informatique et les STIC (Système de Traitement des Infractions Constatées).

LMV : L'objectif de l'équipe LMV est de contribuer à l'amélioration de la compréhension des problèmes de sûreté et de sécurité des systèmes informatiques.

Définition

Le λ -calcul est un système formel inventé par Alonzo Church dans les années 1930, qui fonde les concepts de fonction et d'application.

Variable	Abstraction	Application
X	$\lambda X.M$	$(M \ N)$

Règles de réduction

α $(\lambda X_1.M)$	\rightarrow_α	$(\lambda X_2.M[X_1 \leftarrow X_2])$ où $X_2 \notin FV(M)$
β $((\lambda X.M_1)M_2)$	\rightarrow_β	$M_1[X \leftarrow M_2]$
η $(\lambda X.(M \ X))$	\rightarrow_η	M où $X \notin FV(M)$

Théorème de la forme normale

Si on peut réduire L tels que $L =_n M$ et $L =_n N$ et que N et M sont en forme normale alors $M = N$ à n renommages près.

Règles de la stratégie de réduction

$$\begin{array}{lll} M & \longrightarrow_{\bar{n}} & N \\ M & \longrightarrow_{\bar{n}} & N \\ (\lambda X.M) & \longrightarrow_{\bar{n}} & (\lambda X.N) \\ (M \ N) & \longrightarrow_{\bar{n}} & (M' \ N) \\ (M \ N) & \longrightarrow_{\bar{n}} & (M \ N') \end{array}$$

Machine CC

- utilise la β -réduction
- Sépare l'expression en 2 sous-expressions
- Exploite la chaîne de contrôle uniquement

Machine CK

- Utilise la β -réduction
- principe de la continuation

Machine SCC

- Utilise la β -réduction
- Version simplifiée de la machine CC
- Exploite les deux sous-expressions

Machine CEK

- Version plus complète de la machine CK
- Ajout d'un environnement

Définition : La continuation

La continuation d'un système désigne son futur, c'est-à-dire la suite des instructions qu'il lui reste à exécuter à un moment précis.

Machine SECD

- Sauvegarde différente du CEK
- Appel par valeur
- Fonctionne avec son propre langage
- Composée de quatre éléments :
 - une pile S
 - un environnement E
 - une chaîne de contrôle C
 - un dépôt D

Règles de la machine SECD

- 1 $\langle S, \varepsilon, b \ C, D \rangle \mapsto_{secd} \langle b \ S, \varepsilon, C, D \rangle$
- 2 $\langle S, \varepsilon, X \ C, D \rangle \mapsto_{secd} \langle V \ S, \varepsilon, C, D \rangle$ où $V = \varepsilon(X)$
- 3 $\langle S, \varepsilon, \langle X, C' \rangle \ C, D \rangle \mapsto_{secd} \langle \langle \langle X, C' \rangle, \varepsilon \rangle \ S, \varepsilon, C, D \rangle$
- 4 $\langle V \ \langle \langle X, C' \rangle, \varepsilon' \rangle \ S, \varepsilon, ap \ C, D \rangle \mapsto_{secd} \langle \varepsilon, \varepsilon'[X \leftarrow V], C', \langle S, \varepsilon, C, D \rangle \rangle$
- 5 $\langle V \ S, \varepsilon, \emptyset, \langle S', \varepsilon', C', D \rangle \rangle \mapsto_{secd} \langle V \ S', \varepsilon', C', D \rangle$
- 6 $\langle b_1 \dots b_n \ S, \varepsilon, prim_{o^n} \ C, D \rangle \mapsto_{secd} \langle V \ S, \varepsilon, C, D \rangle$ où $V = \delta(o^n, b_1, \dots b_n)$

Synchrone

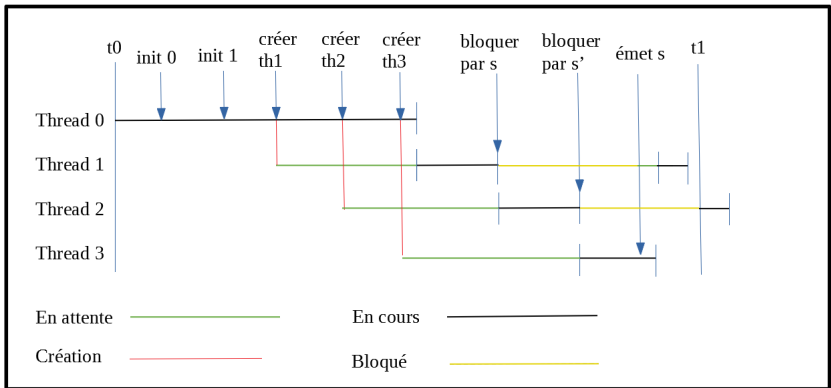
Synchrone signifie que les informations seront obtenues de manière immédiate.

Concurrente

Concurrente signifie que plusieurs processus vont se dérouler durant le même instant logique.

Machines TTS : Explication

$\lambda s.(\lambda s'.(\text{Spawn}(\text{present } s \text{ in } 6 \ 9) \ \text{Spawn}(\text{present } s' \text{ in } 3 \ 5) \ \text{Spawn}(\text{emit } s)) \ \text{init}) \ \text{init}$
 $\langle s, \langle s', \langle \langle s, 6 \rangle \langle 9 \rangle \text{present} \rangle \ \text{spawn} \langle s', \langle 3 \rangle \langle 5 \rangle \text{present} \rangle \ \text{spawn} \ \text{ap} \langle s, \text{emit} \rangle \ \text{spawn} \ \text{ap} \rangle \rangle \ \text{init} \ \text{ap} \ \text{init} \ \text{ap}$



Création d'un thread :

$$\langle \langle \langle X, C' \rangle, E \rangle S, E, \text{spawn } C, D \rangle, TL, SI \rangle \longrightarrow_{TTS} \langle \langle S, E, C, D \rangle, TL \langle S, E, C', D \rangle, SI \rangle$$

Initialisation d'un signal :

$$\langle \langle S, E, \text{init } C, D \rangle, TL, SI \rangle \longrightarrow_{TTS} \langle \langle s, S, E, C, D \rangle, TL, SI' \rangle$$

avec $\iota(SI) = (s, SI')$

Émettre :

$$\langle \langle s, S, E, \text{emit } C, D \rangle, TL, SI \rangle \longrightarrow_{TTS} \langle \langle S, E, C, D \rangle, TL \text{ST}, SI' \rangle$$

avec $\varepsilon(s, SI) = (\text{ST}, SI)'$

Récupération de thread :

$$\langle \langle S, E, \epsilon, \emptyset \rangle, \langle S', E', C, D \rangle TL, SI \rangle \longrightarrow_{TTS} \langle \langle S', E', C, D \rangle, TL, SI \rangle$$

Fin d'un instant logique :

$$\langle \langle S, E, \epsilon, \emptyset \rangle, \emptyset, SI \rangle \longrightarrow_{TTS} \langle \langle S, E, \epsilon, \emptyset \rangle, TL, SI' \rangle \text{ avec } \tau(SI) = (TL, SI')$$

Test d'un signal présent :

$\langle \langle \langle X', C'' \rangle, E \rangle \langle X, C' \rangle, E \rangle s S, E, \text{present } C, D \rangle, TL, SI \rangle \longrightarrow_{TTS}$
 $\langle \langle S, E, C' \ C, D \rangle, TL, SI \rangle$ avec $SI(s) = \langle \text{vraie}, ST \rangle$

Test d'un signal non présent avec thread remplaçable :

$\langle \langle \langle X', C'' \rangle, E \rangle \langle X, C' \rangle, E \rangle s S, E, \text{present } C, D \rangle, \langle S', E', C''', D' \rangle$
 $TL, SI \rangle \longrightarrow_{TTS} \langle \langle S', E', C''', D' \rangle, TL, SI' \rangle$

avec $SI(s) = \langle \text{faux}, ST \rangle$

et $SI'(s) = \langle \text{faux}, ST \ \langle \langle X', C'' \rangle, E \rangle \langle X, C' \rangle, E \rangle s S, E, \text{present } C, D \rangle \rangle$

Test d'un signal non présent sans thread remplaçable :

$\langle \langle \langle X', C'' \rangle, E \rangle \langle X, C' \rangle, E \rangle s S, E, C, D \rangle, \emptyset, SI \rangle \longrightarrow_{TTS}$
 $\langle \langle \emptyset, \epsilon, \emptyset, \emptyset \rangle, \emptyset, SI' \rangle$

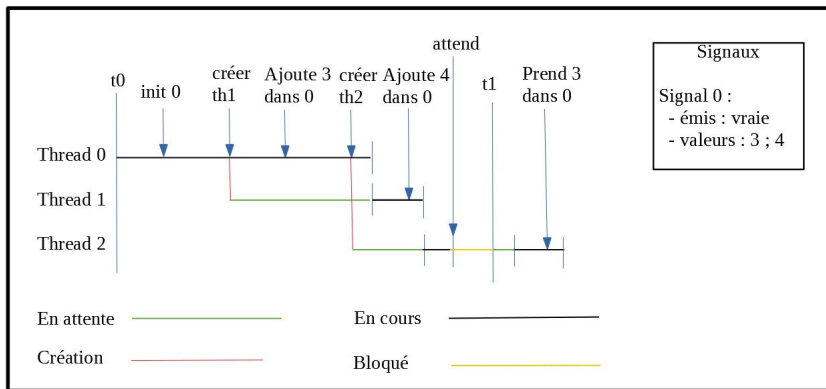
avec $SI(s) = \langle \text{faux}, ST \rangle$

et $SI'(s) = \langle \text{faux}, ST \ \langle \langle X', C'' \rangle, E \rangle \langle X, C' \rangle, E \rangle s S, E, \text{present } C, D \rangle \rangle$

Machines TTSI : Explication informelle

$\lambda s. (\text{Spawn}(\text{put } x \ 4) \ \text{put } x \ 3 \ \text{Spawn}(\text{wait } \text{get } x \ 1 \ 0)) \ \text{init}$

$\langle x, \langle, 4 \times \text{put} \rangle \ \text{spawn } 3 \times \text{put } ap \ \langle, -1 \ \langle, \rangle \ \langle, \rangle \ \text{present } x \ 1 \ 0 \ \text{get } ap \rangle \ \text{spawn } ap \ \rangle \ \text{init } ap$



Ajout d'une valeur :

$$\langle \langle I, s \text{ b } S, E, \text{put } C, D \rangle, TL, SI \rangle \longrightarrow_{TTSl} \langle \langle I, S, E, C, D \rangle, TL, SI [(s, I) \leftarrow b] \rangle$$

Prise d'une valeur :

$$\langle \langle I, s \text{ b } n \langle \langle X, C' \rangle, E' \rangle S, E, \text{get } C, D \rangle, TL, SI \rangle \longrightarrow_{TTSl}$$

$$\langle \langle I, \emptyset, E' [X \leftarrow V], C', \langle S, E, C, D \rangle \rangle, TL, SI \rangle$$

si pour $SI(s) = \langle \text{emit}, CS, SSI \rangle$ et $SSI(b) = \langle CI, IL \rangle$ on a $I \notin IL$ alors $\gamma(I, b, SSI(b)) = V$
sinon $n = V$

Émettre :

$$\langle \langle s \text{ } S, E, \text{emit } C, D \rangle, TL, SI \rangle \Longrightarrow_{TTS} \langle \langle S, E, C, D \rangle, TL \text{ } ST, SI' \rangle$$

avec $\varepsilon(s, SI) = (ST, SI)'$

En Résumé :

- Machine réactive pure
- Machine réactive avec partages de valeurs

Travaux futures :

- Preuve du déterminisme
- Gestion des erreurs