Sémantique des systèmes concurrents

Stage de fin d'année

Jordan Ischard

Université d'Orléans

06 juin 2019

Sommaire

Introduction

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.

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

$$\alpha\ (\lambda X_1.M) \rightarrow_\alpha (\lambda X_2.M[X_1 \leftarrow X_2]) \text{ où } X_2 \notin FV(M)$$

$$\beta\ ((\lambda X.M_1)M_2) \rightarrow_\beta M_1[X \leftarrow M_2]$$

$$\eta\ (\lambda X.(M\ X)) \rightarrow_\eta M \text{ où } X \notin FV(M)$$

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

$$
\begin{array}{lll}
M & \longrightarrow_{\bar{n}} & N \quad \text{si } M \ \beta \ N \\
M & \longrightarrow_{\bar{n}} & N \quad \text{si } M \ \eta \ N \\
(\lambda X.M) & \longrightarrow_{\bar{n}} & (\lambda X.N) \text{ si on a } M \ \beta \ N \text{ ou } M \ \eta \ N \\
(M \ N) & \longrightarrow_{\bar{n}} & (M' \ N) \text{ si } M \longrightarrow_{\bar{n}} M' \text{ et } \forall \ L, (M \ N) \not\beta \ L \\
(M \ N) & \longrightarrow_{\bar{n}} & (M \ N') \text{ si } N \longrightarrow_{\bar{n}} N' \text{ et } \forall \ L, (M \ N) \not\beta \ L
\end{array}
$$

Machines étudiées

Machine CC

- utilise la β-réduction
- Sépare l'expression en 2 sous-expressions
- Exploite la chaîne de contôle 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 quatres éléments :
  - une pile S
  - un environnement ε
  - une chaîne de contrôle C
  - un dépôt D

Règles de la machine SECD

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

Programmation réactive synchrone concurrente

Programmation réactive

La programmation réactive est un paradigme de programmation visant à conserver une cohérence d'ensemble en propageant les modifications d'une source réactive aux éléments dépendants de cette source.

Synchrone

Synchrone signifie que les informations seront obtenues de manière immédiate.

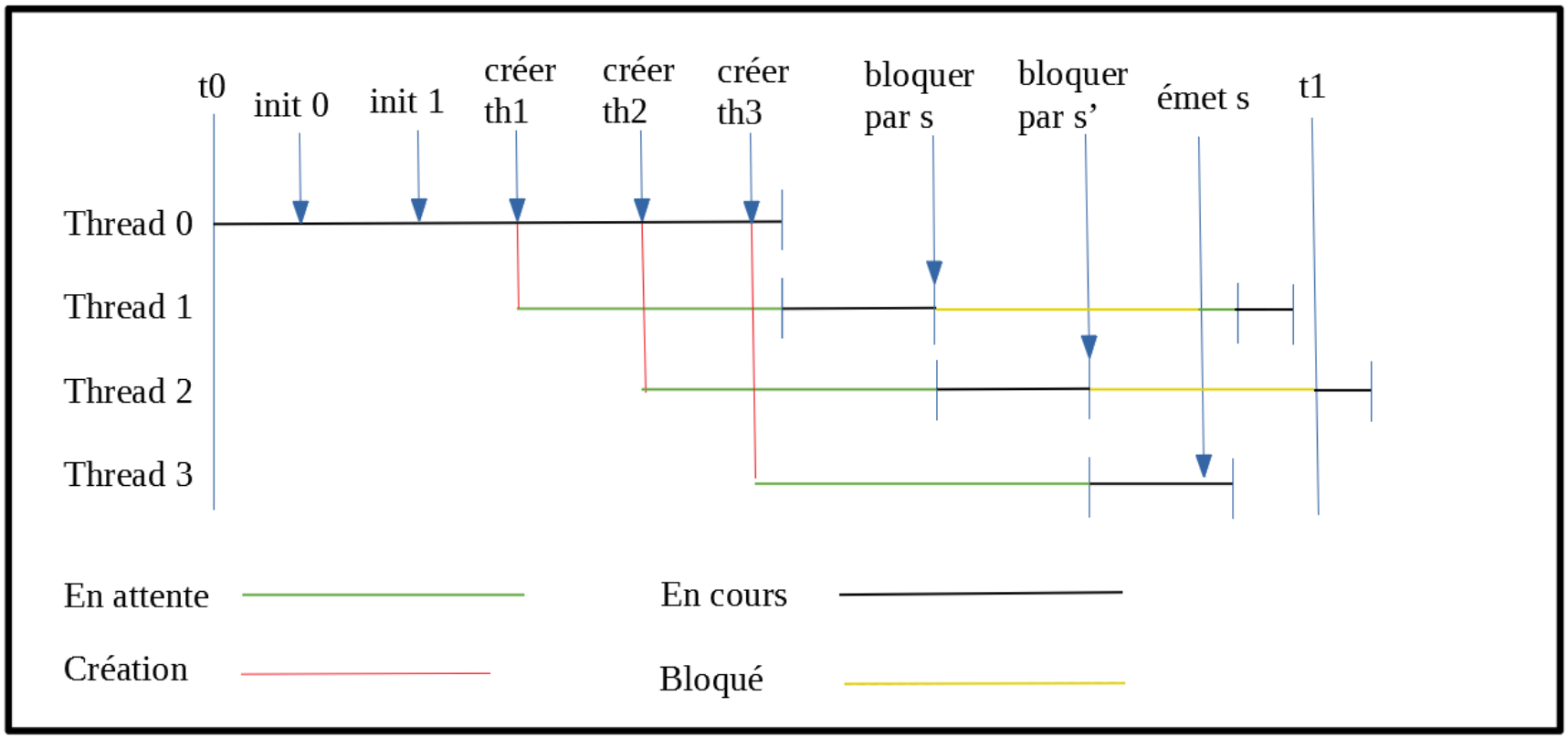
Concurrente

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

Machines TTS : Explication

$\lambda s.(\lambda s'.(\ \mathit{Spawn}(\mathit{present}\ s\ \mathit{in}\ 6\ 9)\ \mathit{Spawn}(\mathit{present}\ s'\ \mathit{in}\ 3\ 5)\ \mathit{Spawn}(\mathit{emit}\ s))\ \mathit{init})\ \mathit{init}$

$\langle s, \langle s', \langle, s\langle, 6\rangle\langle, 9\rangle \mathit{present}\rangle\ \mathit{spawn}\ \langle, s'\langle, 3\rangle\langle, 5\rangle \mathit{present}\rangle\ \mathit{spawn}\ \mathit{ap}\ \langle, s\ \mathit{emit}\rangle\ \mathit{spawn}\ \mathit{ap}\ \rangle\rangle\ \mathit{init}\ \mathit{ap}\ \mathit{init}\ \mathit{ap}$



Machines TTS : Sémantique de la machine

Création d'un thread :

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

Initialisation d'un signal :

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

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

Émettre :

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

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

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 :

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

Machines TTS : Sémantique de la machine

Test d'un signal présent :

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

Test d'un signal non présent avec thread remplaçable :

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

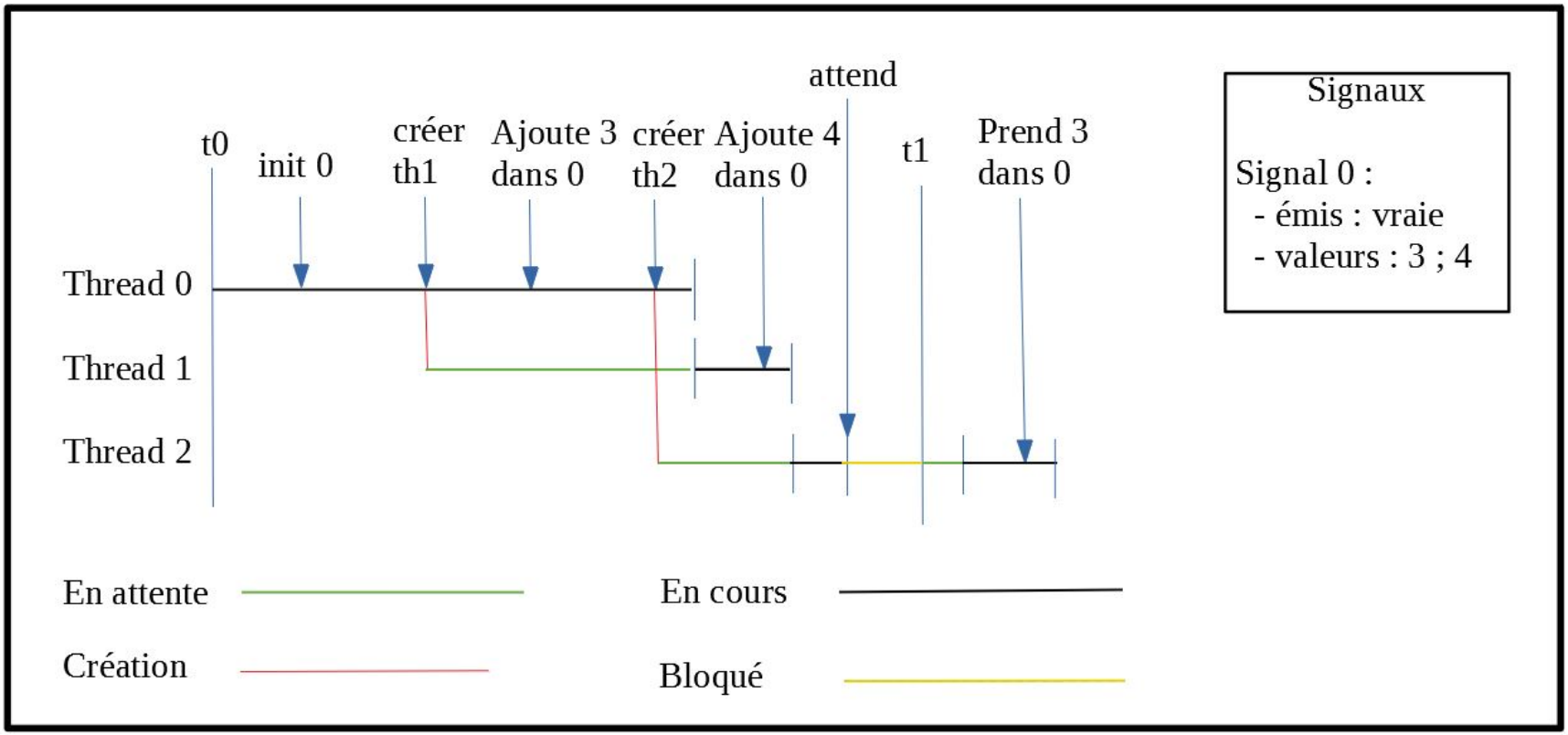
Test d'un signal non présent sans thread remplaçable :

$\langle\langle\langle\langle X', C''\rangle, E\rangle\ \langle\langle X, C'\rangle, E\rangle\ s\ S, E, C, D\rangle, \emptyset, SI\rangle \longrightarrow_{TTS}$
$\langle\langle\emptyset, \epsilon, \emptyset, \emptyset\rangle, \emptyset, SI'\rangle$
avec $SI(s) = \langle faux, ST\rangle$
et $SI'(s) = \langle faux, ST\ \langle\langle\langle X', C''\rangle, E\rangle\ \langle\langle X, C'\rangle, E\rangle\ s\ S, E, present\ C, D\rangle\rangle$

Machines TTSI : Explication informelle

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

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



Machines TTSI : Sémantique de la machine

Ajout d'une valeur :

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

Prise d'une valeur :

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

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

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

sinon $n = V$

~~**Émettre :**~~

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

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

Conclusion

En Résumé :

- Machine réactive pure
- Machine réactive avec partages de valeurs

Travaux futures :

- Preuve du déterminisme
- Gestion des erreurs