Support de travaux dirigés/pratiques - CCS

Calculus of **C**ommunicating **S**ystems, MILNER

Christian Attiogbé, Mars 2007

Présentation de CCS

Un **processus (ou agent)** modélise le comportement d'un système séquentiel ou concurrent.

Elément de base de la spécification des processus

– alphabet d'**actions** ;
– un ensemble d'**opérateurs** (préfixage noté **.**, composition parallèle notée |, choix noté +, ...)
O dénote (une action nulle) ou plus précisément un processus prédéfini qui est la terminaison.

Les actions (alphabet d'un processus) :

$\mathcal{A}$	: ensemble des étiquettes et des actions d'entrée
$\overline{\mathcal{A}}$	: $\{\overline{\alpha} \mid \alpha \in \mathcal{A}\}$
$\mathcal{A} \cup \overline{\mathcal{A}}$	: ensemble des actions externes
$Act = \mathcal{A} \cup \overline{\mathcal{A}} \cup \{\tau\}$	: toutes les actions (internes et externes)

Structure d'un processus CCS élémentaire

```
Agent P = action1 . SuiteComportement
Agent SuiteComportement = action2 . 0 + action3 . action4 . SuiteComport2
Agent SuiteComport2 = ...
```

Les processus CCS peuvent communiquer lorsqu'ils sont composés en parallèle.

Modèle de communication

Il est synchrone (entre les processus composés en parallèle)
La communication est réalisée avec des actions de même nom dites actions complémentaires : action, $\overline{\text{action}}$
(complémentaires peut s'interpréter comme : entrée, sortie).
Evolution interne : τ
Les actions complémentaires s'effectuent en parallèle ; on effectue simultanément action et $\overline{\text{action}}$

Restriction dans la communication : (hiding)

Un ensemble d'actions ($L = \{act1, act2, \cdots\}$ avec $L \subseteq Act$) est utilisé pour contraindre les synchronisations qui se passent comme une évolution interne.
On contrôle ainsi les synchronisations entre les processus.

Syntaxe abstraite de CCS (notation BNF)

Soit $a \in Act$ et $f \in Act \rightarrow Act$ une fonction (partielle) de renommage qui associe un nouveau nom à une action ; (concrètement on note f par nouveau1/ancien1, nouveau2/ancien2, ...

E	$::=$	0	terminaison
	$\mid$	$a.E$	préfixage
	$\mid$	$E + E$	choix non déterministe
	$\mid$	$E \mid E$	composition parallèle
	$\mid$	$E \setminus L$	masquage
	$\mid$	$E[f]$	renommage
	$\mid$	A	constante d'agent
	$\mid$	(E)	parenthésage

Sémantique (opérationnelle) de CCS

Sémantique opérationnelle

– **ensemble de règles d'inférence** définies par rapport à une syntaxe abstraite ;
– Chaque règle a une ou plusieurs prémisses et une seule conclusion.

Règles d'inférence

$$\frac{Premisses}{Conclusion}$$

Définition de la sémantique de CCS revient à définir un ensemble de règles pour chaque opérateur de la grammaire.

pref $\dfrac{}{\alpha.E \xrightarrow{\alpha} E}$

choixg $\dfrac{E \xrightarrow{\alpha} E'}{E+F \xrightarrow{\alpha} E'}$ choixd $\dfrac{F \xrightarrow{\alpha} F'}{E+F \xrightarrow{\alpha} F'}$

parallg $\dfrac{E \xrightarrow{\alpha} E'}{E|F \xrightarrow{\alpha} E'|F}$ paralld $\dfrac{F \xrightarrow{\alpha} F'}{E|F \xrightarrow{\alpha} E|F'}$

sync $\dfrac{E \xrightarrow{\alpha} E' \quad F \xrightarrow{\overline{\alpha}} F'}{E|F \xrightarrow{\tau} E'|F'}$ restr $\dfrac{E \xrightarrow{\alpha} E' \quad \alpha \notin L}{E\setminus L \xrightarrow{\alpha} E'\setminus L}$

Autres règles

parenthèsage, renommage, définition d'un agent par un autre

Structure d'une spécification CCS

```
Agent Processus1 = ...
Agent Processus1 = ...
Sys MonSysteme = (Processus1 | Processus2) \ ListeDeSyncho
```

Sémantique relationnelle

Soit $\mathcal{P}$ un ensemble de processus CCS. La sémantique de CCS, donnée inductivement par les règles de sémantique opérationnelle précédentes, est aussi formalisée par une relation d'évolution notée $\rightarrow$:
$\rightarrow \subseteq \mathcal{P} \times Act \times \mathcal{P}$
ainsi, lorsqu'un processus P (avec $P = \alpha.Reste$) peut évoluer en $Reste$ en faisant α, on a $(P, \alpha, Reste) \in \rightarrow$.
On écrit plus simplement $P \xrightarrow{\alpha} Reste$

1 Anayse des processus : bissimulation forte et équivalence observationnelle

Deux principales relations caractérisent les processus : la *bissimulation forte* notée $\sim$ et la *bissimulation faible* ou *équivalence observationnelle* notée $\approx$.

Bissimulation forte

Une relation $\mathcal{B} \subseteq \mathcal{P} \times \mathcal{P}$ est une bissimulation forte si étant donné $(P, Q) \in \mathcal{B}$,alors
$\forall \alpha \in \mathcal{A}ct$
– lorsque $P \xrightarrow{\alpha} P'$ alors, $\exists Q'$ tq $Q \xrightarrow{\alpha} Q'$ et $(P', Q') \in \mathcal{B}$
– lorsque $Q \xrightarrow{\alpha} Q'$ alors, $\exists P'$ tq $P \xrightarrow{\alpha} P'$ et $(P', Q') \in \mathcal{B}$
Ainsi, deux processus P et Q sont en fortement équivalents ou en fortement bissimilaire (noté $P \sim Q$) si $(P, Q) \in \mathcal{B}$ pour une certaine bissimulation $\mathcal{B}$.
En conséquence $\sim = \{\mathcal{B} \mid \mathcal{B}$ est une bissimulation forte$\}$

Exemple
$P =$ `login.passOK + abando.0` et
$Q =$ `(login.passOK.0 + abandon.0) + login.passOK.0`
sont fortement bissimilaires : (P$\sim$Q).

$P =$ `login.passOK + abando.0` et
$Q =$ `(login.passOK.0 + login.abandon.0)`
ne sont pas fortement bissimilaires.

Bissimulation faible / équivalence observationelle

Préliminaires :
$\alpha \in \mathcal{A}$ une action de l'alphabet
$\alpha^n \in \mathcal{A}^*$ $n \geq 0$ une répétition de α
$t \in \mathcal{A}^*$ une suite d'actions de l'alphabet
On définit une relation $\Longrightarrow$ sur $\mathcal{B}$ qui permet d'oublier les actions τ.

$P \xRightarrow{\alpha} Q$ si $P \xrightarrow{\tau^n} \xrightarrow{\alpha} \xrightarrow{\tau^m} Q$ avec $n \geq 0$ et $m \geq 0$

$P \xRightarrow{\epsilon} Q$ si $P \xrightarrow{\tau^n} Q$ avec $n \geq 0$

$\xrightarrow{t}$ spécifie exactement les actions τ

$\stackrel{t}{\Longrightarrow}$ spécifie au moins les occurrences des actions τ

$\stackrel{\widehat{t}}{\Longrightarrow}$ ignore les actions τ

Conséquence

$P \xrightarrow{t} P'$ implique $P \stackrel{t}{\Longrightarrow} P'$

$P \stackrel{t}{\Longrightarrow} P'$ implique $\stackrel{\widehat{t}}{\Longrightarrow}$

Définition ($\approx$)

Deux processus P et Q sont en observationnellement équivalents ou faiblement bissimilaires (noté $P \approx Q$) si et seulement si

$\forall \alpha \in \mathcal{A}ct$

– lorsque $P \xrightarrow{\alpha} P'$ alors, $\exists Q'$ tq $Q \stackrel{\widehat{\alpha}}{\Longrightarrow} Q'$ et $P' \approx Q'$

– lorsque $Q \xrightarrow{\alpha} Q'$ alors, $\exists P'$ tq $P \stackrel{\widehat{\alpha}}{\Longrightarrow} P'$ et $P' \approx Q'$

Value-passing CCS de Milner

Version initiale de CCS

Contrôle + passage de valeurs abstraites.

Langage de données abstraites

– Ensemble de valeurs et d'expressions abstraites.

Limiter à quelques constructions

– des actions simples mais paramétrées
(pas d'actions simples non paramétrées)

– pas d'actions indexées

Idée générale du Value-passing CCS

– **actions d'entrée paramétrées :** $a(x).P$
x est une variable de n'importe quel *type*

– **actions de sortie paramétrées :** $\overline{a}(e).P$
e est n'importe quelle expression (liberté excessive)

– définitions d'**agents paramétrées :** $A(x) \mathrel{\widehat{=}} P$

– **structure conditionnelle**
`if` *cond* `then` P_1 `else` P_2

Nous allons étudier tous ces aspects dans la suite du cours, avec le langage LOTOS (normalisé ISO). Il est inspiré de CCS, de CSP (une autre algèbre de processus, Hoare) et intègre des données définies dans un langage algébrique.