Intégration de méthodes formelles

Intégration de méthodes formelles :
algèbres de processus et langage de données

Slide 1

Christian Attiogbé
`Christian.Attiogbe@irin.univ-nantes.fr`

Intégration de méthodes formelles

Intégration de méthodes

☞ Combinaison des facettes données et processus.
⇒ **Traitement des aspects statique et dynamique relevés dans un cahier de charges**

Slide 2

Une approche : combinaison d'un langage de processus(dynamique) avec un langage de données (statique)

Exemple : Passage de valeurs aux algèbres de processus

Références :

G. Bruns, *A Language and Translator for Value-passing CCS*, TR, 1994
R. Milner, *Communication and Concurrency*, Prentice Hall, 1989

Slide 3

Intégration de méthodes formelles

Etude du passage de valeurs à une algèbre de processus

Plusieurs algèbres de processus : CCS CSP, ACP
(avec des variantes données, temps)
Outils d'analyse formelle : basés sur des machines à états finis

Extensions aux données :
→ meilleure adéquation au traitement des cahiers de charge
→ données ⇒ explosion des états, **systèmes infinis**
→ raisonnement formel plus délicat
Raisonnement formel :
– étude des propriétés dynamiques (liveness)
– étude des propriétés de sûreté (safety)

Slide 4

Intégration de méthodes formelles

Présentation de CCS (aspects sémantiques)

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

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

Elément de base de la modélisation : **actions**
→ alphabet d'actions
Action nulle : 0 (terminaison)

Un ensemble d'**opérateurs** (préfixage, composition parallèle, choix)

Slide 5

Modèle de communication

synchrone (entre les processus composés en parallèle)

Réalisé avec des actions de même nom : action, $\overline{\mathsf{action}}$
(complémentaires : entrée, sortie).
Evolution interne : τ (simultanément action et $\overline{\mathsf{action}}$)

Restriction dans la communication : (hiding)
Un ensemble d'actions (L) utilisé pour l'évolution interne.
$\rightarrow$ contrôler les synchronisations

Slide 6

Syntaxe abstraite

$$
\begin{array}{lcl}
\hline
E & ::= & 0 \\
 & | & a.E \\
 & | & E + E \\
 & | & E \;|\; E \\
 & | & E \setminus L \\
 & | & A \\
 & | & (E) \\
\hline
\end{array}
$$

Slide 7

CCS : Sémantique opérationnelle

Sémantique opérationnelle :
- **ensemble de règles d'inférence** définies par rapport à une syntaxe abstraite
- Chaque règle à une ou plusieurs prémisses et une seule conclusion

Règles d'inférence

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

Définition de la sémantique $\Rightarrow$ un ensemble de règles pour chaque opérateur de la grammaire.

Slide 8

Intégration de méthodes formelles

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 $\dfrac{E \xrightarrow{\alpha} E' \quad \alpha \notin L}{E\backslash L \xrightarrow{\alpha} E'\backslash L}$

Autres règles : parenthèsage, renommage, définition d'un agent par un autre

Intégration de méthodes formelles

Slide 9

Value-passing CCS de Milner

Version initiale de CCS

Contrôle + passage de valeur très **abstrait**.

Langage de données abstrait

– Ensemble de valeurs et d'expressions très abstraites.

Limiter à quelques constructions

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

Intégration de méthodes formelles

Slide 10

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 :** $A(x) \widehat{=} P$

– **structure conditionnelle**

`if` $cond$ `then` P_1 `else` P_2

Intégration de méthodes formelles

Besoin d'un langage de données

☞ **Particulariser/renforcer** le langage de données abstrait de Milner

→ Définir un langage de donnée (ou en prendre un existant)
→ Définir la sémantique du langage
ou adapter la sémantique du langage

☞ **Adapter ou étendre la sémantique** de CCS (Milner)

Slide 11

Intégration de méthodes formelles

Approche de Glenn Bruns (extension du value-passing de CCS)

Un langage de données qui lève les restrictions du CCS de Milner en ajoutant :

- **ensemble, tuples, listes** de données (entier, booléen et chaines)
- autorise les actions simples, indexées, parametrées, indexées/parametrées
- autorise l'association d'un ensemble de valeurs spécifiques aux déclarations d'actions

Slide 12

Intégration de méthodes formelles

Illustration par des exemples

Slide 13

Spécification du domaine des valeurs d'un paramètre :

```
label a({1,2})

agent A(x:{1,2}) = 'a(x+1).0
```

Actions indexées et definitions de constantes

```
value N = {1,2}
value B = {true,false}
label a_B(N)
label b_B
agent A(x:N) = b_y.'a_y(x+1).0
```

A se synchronise sur l'action indexée b_y, puis sort $x + 1$ sur l'action indexée a_y.

Intégration de méthodes formelles

Slide 14

```
const n     = 3
const Index = {0, 1, 2}
const Iset  = pow(Index)
label a_Index, b_Index

agent Schedspec(i:Index, X:Iset) =
      if member(i,X) then
         sum(j:X,b_j.Schedspec(i,diff(X,{j})))
      else
         a_i.Schedspec((i+1) mod n, union(X,{i}))
       + sum(j:X, b_j.Schedspec(i,diff(X,{j})))
```

Slide 15

Conception

CCS $\Rightarrow$ **Indépendance** entre
– le langage d'agents (l'algèbre de processus) et
– le langage de données

Le langage de données ne fait pas d'hypothèse sur l'algèbre de processus.

L'algèbre de processus ne considère que des valeurs et des expressions.

☞ adaptation facile du langage de données.

Slide 16

Aspects sémantiques

Définition du(des) langage(s) $\rightarrow$ sémantique

Sémantique opérationnelle structurée (Plotkin, 1981)

Règles d'inférence

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

Une règle d'inférence de la définition donne les transitions possibles pour un agent P dans un environnement Env

$$Env \vdash decl \xrightarrow{\alpha} Env'$$

(environnement = contexte où les processus évoluent)

Intégration de méthodes formelles

Initialement Milner a décrit les règles sémantiques de CCS avec un environnement implicite.

☞ On rend l'environnement explicite : Env (noté simplement E dans la suite)

Slide 17

Mises à jour de l'environnement effectuées par la principale règle de la définition

$$Env \vdash decl \Rightarrow Env'$$

Intégration de méthodes formelles

Langage (processus) étendu : Syntaxe

Mots réservés : `label`,`agent`,`const`, etc

Les identificateurs (Identifiants)

Slide 18

x	$\in$	Var	variables de valeurs
a	$\in$	$Name$	noms CCS
A	$\in$	$Const$	constantes d'agent

Slide 19

Langage étendu : Syntaxe (suite)

Grammaire : Principales catégories de termes

		$Program$	les programmes/spécifications
		$Decl$	les déclarations
		$Param$	paramètres d'agent
P	$\in$	$AExpr$	expressions d'agent
		$LName$	noms longs
		$Label$	étiquettes CCS
		$ReLabFun$	fonctions de renommage
		$Lpair$	paire d'étiquettes
e	$\in$	$VExpr$	expressions de valeurs

Slide 20

Langage étendu : Syntaxe (suite)

Grammaire pour les déclarations (et les spécifications)

$Specif$	$::=$	$decl\langle, declseq\rangle$	
$decl$	$::=$	$\texttt{const}\ x = e$	décl. de constante
		$\texttt{label}\ lname\langle, lnameseq\rangle$	décl. d'étiquette
		$\texttt{agent}\ A = P$	décl. d'agent
		$\texttt{agent}\ A(param\langle, paramseq\rangle) = P$	
		décl. d'agent paramétré	
$param$	$::=$	$x : e$	

Slide 21

Langage étendu : Syntaxe (suite)

Grammaire pour les expressions d'agent

P	$::=$	0	agent inactif
	$\mid$	$\mathtt{t}.P$	préfixe τ
	$\mid$	$a\langle\_e\rangle.P$	entrée simple
	$\mid$	$'a\langle\_e\rangle.P$	sortie simple
	$\mid$	$a\langle\_e\rangle(x).P$	entrée paramétrée
	$\mid$	$'a\langle\_e\rangle(e).P$	sortie paramétrée
	$\mid$	$P + P$	choix (somme)
	$\mid$	$P \mid P$	composition parallèle
	$\mid$	$P\backslash\{labelseq\}$	restriction
	$\mid$	$P[relabfun]$	renommage

Slide 22

	$\mid$	$\mathtt{if}\ e\ \mathtt{then}\ P\ \mathtt{else}\ P$	
	$\mid$	$\mathtt{sum}(x : e, P)$	somme indexée
	$\mid$	$\mathtt{comp}(x : e, P)$	composition indexée
	$\mid$	A	constante d'agent
	$\mid$	$A(e\langle, eseq\rangle)$	constante d'agent param.
	$\mid$	(P)	parenthésage
$lname$	$::=$	$a\langle\_e\rangle$	nom simple
	$\mid$	$a\langle\_e\rangle(e)$	nom paramétré
$label$	$::=$	$\langle'\rangle lname$	
$relabfun$	$::=$	$lpair\langle, lpairseq\rangle$	
$lpair$	$::=$	$label/label$	

Intégration de méthodes formelles

Slide 23

Langage étendu : Sémantique

Sémantique des expressions du CCS avec passage de valeurs : un **système de transitions étiquettées**.

Un système de transition étiquettée : (S, L, R)

- S un ensemble d'états,
- L un ensemble d'étiquettes,
- $R \subseteq S \times L \times S$ une relation de transition étiquetée.

(s, l, s') s'écrit $s \xrightarrow{l} s'$

Intégration de méthodes formelles

Slide 24

Un **état** est une expression d'agent P,

une **étiquette** est une action CCS (paramétrée, indexée ou paramétrée indexée).

Les **règles sémantiques** $\rightarrow$ définir la relation de transition entre les expressions d'agent par rapport à un environnement.

L'**environnement** est élaboré par les déclarations.

Intégration de méthodes formelles

Langage étendu : Objets sémantiques manipulés

Slide 25

$$
\begin{array}{lll}
 & & \mathrm{Var} \\
 & & \mathrm{Name} \\
 & & \mathrm{Const} \\
\mathrm{v,b,s,t} & \in & \mathrm{Val} \\
\mathrm{E} & \in & \mathrm{Env} = \mathrm{NEnv} \times \mathrm{CEnv} \times \mathrm{AEnv} \\
\mathrm{NE} & \in & \mathrm{NEnv} = \mathrm{Name} \xrightarrow{fin} \mathrm{Val} \\
\mathrm{CE} & \in & \mathrm{CEnv} = \mathrm{Var} \xrightarrow{fin} \mathrm{AType} \\
\mathrm{AE} & \in & \mathrm{AEnv} = \mathrm{Const} \xrightarrow{fin} \mathrm{AType} \\
\mathrm{nt} & \in & \mathrm{NType} = \{\mathrm{sim}\} \cup (\{\mathrm{idx}\}\times\mathrm{Val}) \cup (\{\mathrm{par}\}\times\mathrm{Var}) \\
 & & \qquad\qquad \cup\ (\{\mathrm{idxpar}\} \times\mathrm{Val}\times \mathrm{Var})
\end{array}
$$

Intégration de méthodes formelles

Objets sémantiques manipulés (suite)

Slide 26

$$
\begin{array}{lll}
 & & \mathrm{AType} = \cup_{k\geq 0}\ (\mathrm{AExpr} \times \mathrm{Var}^k) \\
 & & \mathrm{EName} = \mathrm{Name} \times \mathrm{NType} \\
\mathrm{l} & \in & \mathrm{Label} = \mathrm{EName} \cup \overline{EName} \\
\alpha & \in & \mathrm{Action} = \mathrm{Label} \cup \{\tau\} \\
 & & \mathrm{RFunc} = \mathrm{Action} \rightarrow \mathrm{Action} \\
\mathrm{f} & \in & \mathrm{LMap} = \mathrm{Label} \xrightarrow{fin} \mathrm{Label}
\end{array}
$$

Slide 27

Sémantique ...

Les noms de type (NType) : utilisés pour typer les noms dans les décl. d'étiquettes

`label a(1,2)` : (par, 1,2)

i.e. action **par**amétrée avec des valeurs dans `1,2`

`[a(2)/`b`]` : (par, 2)

i.e. action **par**amétrée avec la valeur 2

Intégration de méthodes formelles

Slide 28

Sémantique : Règles d'inférence

2 types de règles :

$$E \vdash phrase \Rightarrow A$$

comment les déclarations et expressions sont évaluées, et modifient l'environnement.

$$E \vdash P \xrightarrow{\alpha} P'$$

Les possibilités de transition d'un processus dans un environnement donné.

Intégration de méthodes formelles

Slide 29

Une expression d'agent P d'une spécification *prog* peut effectuer la transition $P \xrightarrow{\alpha} P'$ si *prog* fournit l'environnement E

$$(\{\},\{\},\{\}) \vdash prog \Rightarrow E$$

tel que

$$E \vdash P \xrightarrow{\alpha} P'$$

Intégration de méthodes formelles

Slide 30

Règles sémantiques ...

Les spécifications (Programs) $E \vdash prog \Rightarrow E'$

$$\frac{E \vdash decl \Rightarrow E_1 \quad \langle E \vdash declseq \Rightarrow E_2 \rangle}{E \vdash decl\langle, declseq\rangle \Rightarrow E_1\langle + E_2\rangle}$$

Les déclarations $E \vdash decl \Rightarrow E'$

$$\frac{E \vdash e \Rightarrow v}{E \vdash \texttt{const}\ x = e \Rightarrow \{x \mapsto v\} \text{ in Env}}$$

Slide 31

Intégration de méthodes formelles

Règles sémantiques ...

$$\frac{lnameseq = (ln_1, \cdots, ln_k) \quad E \vdash ln_i \Rightarrow (n_i, nt_i)\ (1 \geq i \geq k)}{E \vdash \texttt{label}\ lnameseq \Rightarrow \{n_1 \mapsto nt_1, \cdots, n_k \mapsto nt_k\} \text{ in Env}}$$

$$E \vdash \texttt{agent}\, A = P \Rightarrow \{A \mapsto (P, ())\} \text{ in Env}$$

$$\frac{param\langle, paramseq\rangle = x_1 : e_1, \cdots, x_k : e_k}{E \vdash \texttt{agent}\ A(param\langle, paramseq\rangle) = P \Rightarrow \{A \mapsto (P, (x_1, \cdots, x_k))\} \text{ in Env}}$$

Slide 32

Intégration de méthodes formelles

Règles sémantiques ...

Les expressions d'agent $\qquad E \vdash P \xrightarrow{\alpha} P'$

$$E \vdash .P \xrightarrow{\tau} P$$

$$\frac{\langle E \vdash e \Rightarrow v\rangle}{E \vdash a\langle\_e\rangle.P \xrightarrow{a_{\langle v\rangle}} P} \qquad \frac{\langle E \vdash e \Rightarrow v\rangle}{E \vdash' a\langle\_e\rangle.P \xrightarrow{\overline{a}_{\langle v\rangle}} P}$$

$$\frac{E(a) = (\langle idx\rangle par\langle, s_1\rangle, s_2) \quad v_2 \in e_2 \quad \langle E \vdash e \Rightarrow v_1\rangle}{E \vdash a\langle\_e\rangle(x).P \xrightarrow{a(v_2)_{\langle v_1\rangle}} P\{v_2/x\}}$$

idem pour $\overline{a}$

Intégration de méthodes formelles

Règles sémantiques ...

Slide 33

$$\frac{E \vdash P_1 \xrightarrow{\alpha} P'_1}{E \vdash P_1 + P_2 \xrightarrow{\alpha} P'_1} \qquad \frac{E \vdash P_2 \xrightarrow{\alpha} P'_2}{E \vdash P_1 + P_2 \xrightarrow{\alpha} P'_2}$$

$$\frac{E \vdash P_1 \xrightarrow{\alpha} P'_1}{E \vdash P_1 \mid P_2 \xrightarrow{\alpha} P'_1 \mid P_2} \qquad \frac{E \vdash P_2 \xrightarrow{\alpha} P'_2}{E \vdash P_1 \mid P_2 \xrightarrow{\alpha} P_1 \mid P'_2}$$

$$\frac{E \vdash P_1 \xrightarrow{l} P'_1 \quad E \vdash P_2 \xrightarrow{\bar{l}} P'_2}{E \vdash P_1 \mid P_2 \xrightarrow{\tau} P'_1 \mid P'_2}$$

Intégration de méthodes formelles

Règles sémantiques ...

Slide 34

même travail pour les autres constructions

- restriction
- renommage
- conditionnelle
- choix (`sum`)
- compostion parallèle (`comp`)
- paranthésage
- appel d'agent

Langage de données étendu

Slide 35

Classes d'expressions

UnOp	opérateurs unaires
BinOp	opérateurs binaires

Slide 36

Grammaire pour les expressions de valeus

e	$::=$	$scon$	constante spéciale
		x	variable
		$unop\ e$	
		$e\ binop\ e$	
		$fname(e\langle, eseq\rangle)$	application de fonction
		$\{eseq\}$	ensemble énuméré
		$\{e \mid e \text{ where } x : e\}$	ens. en compréhension
		$<< eseq >>$	tuple
		$e\#e$	sélection dans un tuple
		$[eseq]$	liste
		(e)	expression parenthésée

Intégration de méthodes formelles

Règles sémantiques

Expressions $E \vdash vexpr \Rightarrow v$

Slide 37

$$E \vdash scon \Rightarrow \mathtt{val}(scon)$$

$$\frac{E(x) = v}{E \vdash x \Rightarrow v}$$

$$\frac{E \vdash e_1 \Rightarrow (v_1, \cdots, v_k) \quad E \vdash e_2 \Rightarrow i}{E \vdash e_1 \# e_2 \Rightarrow v_i}$$

Intégration de méthodes formelles

Slide 38

même traitement pour toutes les constructions du langage de données :

booléens : `not`, `and`, `or`, **etc**

ensembles : `prod`, `pow`, `union`, `diff`, **etc**

tuples,

listes : `cons`, `hd`, `tl`, **etc**

etc

Slide 39

Conclusion

☞ **Leçons sur la combinaison des algèbres de processus avec des données diverses**

- Relative facilité due à l'indépendance processus, données
- Applicable à d'autres algèbres de processus,
- Applicable à d'autres langages de données,
- Raisonnement formel dans même cadre, ramener en partie si possible à systèmes finis

☞ **comment définir la sémantique** des langages de spécification

☞ **Intégration d'autres facettes**

- intégration des contraintes de temps ?
- autres types d'intégration