Université Bordeaux I — **Master Informatique 2010-2011**
Amphi Kastler – BATIMENT A9 — **Compilation**
INF476EX
Lionel Clément — **mardi 12 avril 2011 – de 14h00 à 15h30**

Examen de Compilation
CORRIGÉ

Exercice 1 : 8 points

Une expression de type est définie inductivement de la sorte :
- Un type de base : *integer*, *boolean* est une expression de type ;
- Une variable de type $`x$ est une expression de type ;
- Un symbole α est une expression de type ;
- Si T_1 et T_2 sont deux expressions de type, $T_1 \times T_2$ est une expression de type ;
- Si T_1 et T_2 sont deux expressions de type, $T_1 \rightarrow T_2$ est une expression de type ;
- Si T est une expression de type, *constructeur*(T) est une expression de type.
  Où *constructeur* vaut *pointer*, *list* ou *record*

Dans le code suivant, nous avons une liste de déclarations de types suivie de l'implémentation d'une fonction *count* qui prend deux arguments *lst* et *elt* et qui retourne le nombre d'éléments *elt* que contient cette liste.

```
1  null: ∀'a, list('a) → boolean ;
2  isList: ∀'a, 'a → boolean ;
3  head: ∀'a, list('a) → 'a ;
4  tail: ∀'a, list('a) → list('a) ;
5  cons: ∀'a, ('a × list('a)) → list('a) ;
6
7  function count (lst, elt)
8  {
9          if null(lst) then
10                 return 0
11         else {
12                 h = head(lst);
13                 if (isList(h)){
14                         return count(h, elt)+count(tail(lst), elt);
15                 } else {
16                         c=count(tail(lst), elt);
17                         if (head(lst) == elt)
18                                 return c+1;
19                         else
20                                 return c;
21                 }
22         }
23 }
```

Questions

1. Quelle est l'expression de type correspondant à la fonction *count* ?

Réponse

$list(`a) \times `a \rightarrow integer$

2. Pourquoi cette fonction est-elle polymorphe ?

Réponse

Le code de la fonction *count* peut s'exécuter avec des arguments de types différents. Par exemple avec une liste d'entiers et un entier, ou encore avec une liste de booléens et un booléen.

3. Soit
   L'expression $[e_1, e_2, \ldots, e_k]$ dont le type est $list(`a)$, et $e_i$ est de type $`a$ pour tout i.
   Expliquer pourquoi le type de l'expression $count([1, [true, false], 2, 3], 2)$ est mal formé.

Réponse

$count([1, [true, false], 2, 3], 2)$ fait appel récursivement à $count([[true, false], 2, 3], 2)$ puis à $count([true, false], 2)$ dont le type est $list(boolean) \times integer \rightarrow integer$ qui ne peut s'identifier à $list(`a) \times `a \rightarrow integer$

4. Soit
   – L'expression $[e_1, e_2, \ldots, e_k]$ dont le type est $list(`a)$, et $e_i$ est de type $`a$ pour tout i.
   – L'expression de valeur de type *integer* .
   – L'expression de valeur de type *boolean*.
   Expliquer précisément pourquoi le type de l'expression $count([1, [true, false], 2, 3], 2)$ est mal formé.

Réponse

$count([1, [true, false], 2, 3], 2)$ fait appel récursivement à $count([[true, false], 2, 3], 2)$ puis à $count([true, false], 2)$ dont le type est $list(boolean) \times integer \rightarrow integer$ qui ne peut s'identifier à $list(`a) \times `a \rightarrow integer$

5. Soit l'ajout de code suivant :

```
1 ∀'a,
2         str: 'a → 'a
3         foo: (('a → 'a) × list('a)) → 'a
```

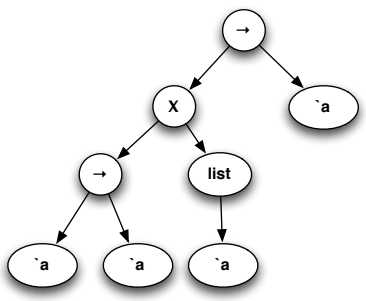
Expliquer comment se réalise le contrôle de type de l'expression $foo(str, [1, 2, 3])$.

Réponse

Le second argument de l'appel de la fonction foo est une liste d'entiers, on peut dont unifier $((`a \rightarrow `a) \times list(`a)) \rightarrow `a$ avec cette expression, l'unificateur le plus général est $\{(`a, integer)\}$.

6. Dessiner une représentation arborescente du type de la fonction foo

Réponse



Exercice 2 : 5 points

Soit G une grammaire algébrique $(\{E\}, \{+, \times, a\}, E, R)$ dont les règles de production R sont les suivantes :

E → E + E
E → E × E
E → a

Le tableau suivant résume l'analyse Earley de l'expression $a + a \times a$

état	lu	à lire	items
0	ϵ	$a + a \times a$	0, 0, E → • E + E 0, 0, E → • E × E 0, 0, E → • a
1	a	$+a \times a$	0, 1, E → a • 0, 1, E → E • + E 0, 1, E → E • × E
2	$a+$	$a \times a$	0, 2, E → E + • E 2, 2, E → • E + E 2, 2, E → • E × E 2, 2, E → • a
3	$a + a$	$\times a$	2, 3, E → a • 0, 3, E → E + E • 2, 3, E → E • + E 2, 3, E → E • × E 0, 3, E → E • + E 0, 3, E → E • × E
4	$a + a\times$	a	2, 4, E → E × • E 0, 4, E → E × • E 4, 4, E → • E + E 4, 4, E → • E × E 4, 4, E → • a
5	$a + a \times a$	ϵ	4, 5, E → a • 2, 5, E → E × E • 0, 5, E → E × E • 4, 5, E → E • + E 4, 5, E → E • × E 0, 5, E → E + E • 2, 5, E → E • + E 2, 5, E → E • × E 0, 5, E → E • + E 0, 5, E → E • × E

Questions

1. Expliquer comment peut-on conclure de ce tableau que la séquence est reconnue par l'analyseur.

Réponse

Les items

0, 5, E → E × E •
0, 5, E → E + E •

permettent de conclure que la séquence est reconnue ;
Le non-terminal E, axiome de la grammaire, est analysé de 0 à 5, c'est-à-dire sur l'ensemble de la séquence

2. La séquence produit une analyse ambigüe. En précisant que l'opérateur $\times$ est prioritaire sur l'opérateur $+$, peut-on modifier l'algorithme de sorte que seule l'analyse attendue soit faite ?
Remarque : Nous pourrons concentrer la démonstration sur la seule réduction

$$\frac{(0,0,E \to \bullet E \times E)(0,3,E \to E + E\bullet)}{(0,3,E \to E \bullet \times E)}$$

Réponse

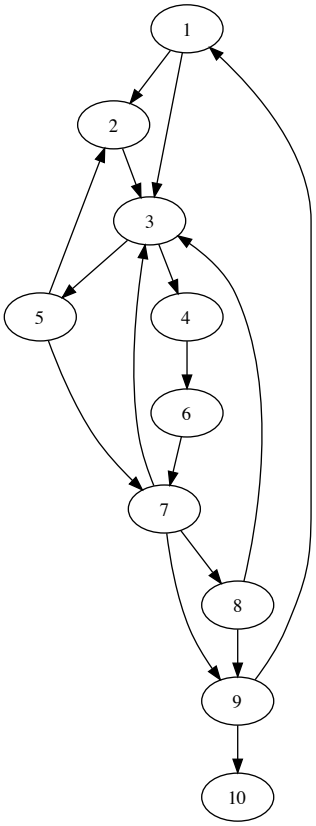
Nous pouvons modifier l'algorithme en assignant à chaque règle de production, un nombre indiquant l'ordre de priorité du terminal qu'il contient (nous ne traitons pas le cas de la présence de plusieurs terminaux). Nous appelons $P(R)$ la priorité de la règle de production P.
La règle de réduction

$$\frac{(i,j,A \to \alpha \bullet B\beta)(j,k,B \to \gamma\bullet)}{(i,k,A \to \alpha B \bullet \beta)}$$

est opérante, si et seulement si $P(A \to \alpha B\beta) < P(B \to \gamma)$.
Dans l'exemple donné, la conséquence est que l'item $(0,3,E \to E \bullet \times E)$ n'est pas produit, ce qui élimine l'analyse problématique.

Exercice 3 : 3 points

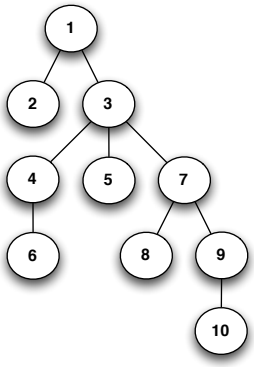
Soit le graphe de flot de contrôle suivant, où 1 représente le bloc initial :



1. Calculer le graphe des dominants

Réponse

Sommets	1	2	3	4	5	6	7	8	9	10
prédécesseurs	9	1 5	1 2 7 8	3	3	4	5 6	7	7 8	9
	1	1 2	1 2 ~~3~~	1 2 ~~3~~ 4	1 2 ~~3~~ ~~4~~ 5	1 2 ~~3~~ 4 ~~5~~ 6	1 2 ~~3~~ ~~4~~ ~~5~~ ~~6~~ 7	1 2 ~~3~~ ~~4~~ ~~5~~ ~~6~~ ~~7~~ 8	1 2 ~~3~~ ~~4~~ ~~5~~ ~~6~~ ~~7~~ ~~8~~ 9	1 2 ~~3~~ ~~4~~ ~~5~~ ~~6~~ ~~7~~ ~~8~~ ~~9~~ 10



2. Est-ce que ce graphe de flot de contrôle contient une ou plusieurs boucles ? Si oui, lesquelles ?

Réponse

(1, 2, 3, 4, 5, 6,7, 8 , 9), (3, 4, 5, 6, 7, 8), (3, 4, 5, 6, 7)

Exercice 4 : 4 points

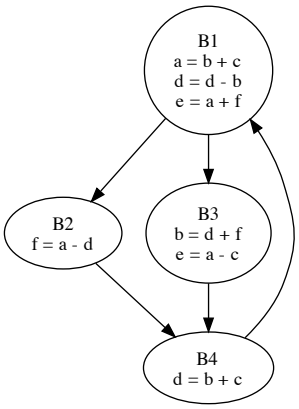
On considère les lignes de code suivantes :

```
1  L1:
2  a = b + c
3  d = d - b
4  e = a + f
5  if e == 0 then goto L3
6  L2:
7  f = a - d
8  goto L4
9  L3:
10 b = d + f
11 e = a - c
12 L4:
13 b=b + c
14 if b == 0 then goto L1
```

Questions

1. Dessiner le graphe de flot de contrôle correspondant.

Réponse



(Dragon page 595)

2. Pour chaque bloc de contrôle, indiquer les variables vivantes en entrée et en sortie.

Réponse

	Use	Def	In	Out
B1	b, c, d, f	a, d, e	b, c, d, f	a, b, c, d, f
B2	a, d	f	a, b, c, d	b, c, f
B3	a, c, d, f	b, e	a, c, d, f	b, c, f
B4	b, c	d	b, c, f	b, c, d, f