Examen final de compilation

ÉNSIIE, semestre 3

mercredi 8 janvier 2014

Durée : 1h45.
Tout document personnel autorisé (pas de prêt entre voisins).
Tout appareil électronique interdit (sauf traducteur pour les étudiants étrangers).
Ce sujet comporte cinq exercices indépendants, qui peuvent être traités dans l'ordre voulu. Il contient 3 pages.
Le barème est donné à titre indicatif, il est susceptible d'être modifié. Le total est sur 20 points.

Exercice 1 : Syntaxe (2 points)

Donner l'arbre de syntaxe abstraite des instructions Pseudo Pascal suivantes :

1. `i := 0`
2. `t[i-1] := t[i+1]`
3. `while i < n do begin t[i] := readln(); i := i + 1 end`

Exercice 2 : Analyse syntaxique (3 points)

On considère la grammaire suivante, avec symbole de départ A :

$$\begin{aligned} A &\to aBa \\ &\mid c \\ B &\to bAb \\ &\mid c \end{aligned}$$

1. Construire l'automate déterministe LR(0) pour cette grammaire.
2. Cette grammaire est-elle LR(0) ?

Exercice 3 : Réécriture (5 points)

On considère trois opérateurs `noop`, `undo`, `compose` attendant respectivement 0, 1 et 2 arguments, ainsi que le système de réécriture suivant :

$$\texttt{undo}(\texttt{undo}(x)) \to x$$
$$\texttt{compose}(\texttt{undo}(x), x) \to \texttt{noop}$$
$$\texttt{compose}(\texttt{noop}, x) \to x$$

1. Donner la ou les forme(s) normale(s) $\texttt{compose}(\texttt{compose}(\texttt{undo}(y), y), \texttt{undo}(\texttt{undo}(x)))$.
2. Le système de réécriture est-il fortement normalisant ?
3. Calculer les paires critiques du système (les causes potentielles de non confluence). Lesquelles sont-elles joignables ?
4. Orienter les paires critiques non joignables de façon à ce que le système soit fortement normalisant. Les nouvelles paires critiques sont-elles joignables ?
5. Compléter le système jusqu'à obtenir un système confluent.

Exercice 4 : Convention d'appel (5 points)

On considère le programme Pseudo Pascal suivant :

```
1  program
2  function other (i, j : integer) : integer;
3  begin
4     if i = 1 and j = 2 then
5        other := 3
6     else if i = 1 and j = 3 then
7        other := 2
8     else
9        other := 1
10 end;
11
12 procedure hanoi (n : integer; i, j : integer);
13 var k :  integer;
14 begin
15    if n > 0 then begin
16       k := other(i, j);
17       hanoi (n - 1, k, j);
18       hanoi (n - 1, i, k)
19    end else begin end
20 end;
21
22 begin
23    hanoi(2, 1, 3)
24 end.
```

On suppose qu'on utilise la convention d'appel MIPS comme vue en cours. On suppose d'autre part que la variable locale `k` dans `hanoi` est stockée dans le registre *callee-saved* `$s0`.

Attention : on ne considère que la convention d'appel MIPS, pas la convention d'appel par pile.

1. Quels registres `hanoi` doit-elle sauvegarder ? En déduire la trame de `hanoi`.
2. Même question pour `other`.
3. Détailler l'évolution de la pile et du contenu des registres `$a0 $a1 $a2 $v0 $ra $s0` au cours du programme.

Exercice 5 : Allocation de registres (5 points)

Le programme Pseudo-Pascal suivant calcule la factorielle de `n` :

```
1  x1 := n;
2  x2 := x1;
3  x1 := 1;
4  t := x2;
5  x2 := x1;
6  x1 := t;
7  while x1 > 0 do begin
8     x3 := x2;
9     x2 := x1;
10    t := x3;
11    x3 := x2;
12    x2 := x1;
13    x1 := t;
14    t := x2;
15    x2 := x1;
16    x1 := t;
17    x1 := x1 * x2;
18    x2 := x3;
19    t := x2;
20    x2 := x1;
21    x1 := t;
22    x1 := x1 - 1 end;
23 x1 := x2;
24 fact := x1
```

1. Dans le graphe de flot de contrôle, quels sont le ou les prédécesseur(s) de l'instruction de la ligne 7 ? de la ligne 23 ?
2. Indiquer dans un tableau quelles variables sont vivantes en chacun des points du programme.
3. Donner le graphe d'interférence du programme (avec les arêtes de préférence éventuelles). On indiquera sur les arêtes le numéro d'une instruction qui a causé l'interférence ou la préférence.
4. Essayer de 3-colorier ce graphe en appliquant l'algorithme de George et Appel. On détaillera bien chaque étape.