Code à trois adresses

Épreuve pratique d'algorithmique et de programmation
Concours commun des écoles normales supérieures

Durée de l'épreuve: 3 heures 30 minutes

Juillet 2005

ATTENTION !

N'oubliez en aucun cas de recopier votre u_0
à l'emplacement prévu sur votre fiche réponse

Important.
Lorsque la description d'un algorithme est demandée, vous devez présenter son fonctionnement de façon schématique, courte et précise. Vous ne devez en aucun cas recopier le code de vos procédures!
Quand on demande la complexité en temps ou en mémoire d'un algorithme en fonction d'un paramètre n, on demande l'ordre de grandeur en fonction du paramètre, par exemple: $O(n^2)$, $O(n \log n)$,...
Il est recommandé de commencer par lancer vos programmes sur de petites valeurs des paramètres et de ***tester vos programmes sur des petits exemples que vous aurez résolus préalablement à la main***.

1 Les codes booléens

Le code à trois adresses est un langage très simple utilisé comme langage intermédiaire en compilation. La restriction principale de ce langage est que chaque instruction ne concerne au plus que 3 variables : une variable de destination du résultat et deux variables dont la valeur est utilisée.
Pour cette épreuve, nous allons étudier des codes à trois adresses manipulant des booléens. Les seules opérations possibles sont le **et** logique et le **ou** logique. Le code sera constitué d'une liste numérotée d'instructions (en partant de 0), chaque instruction étant soit une opération (de la forme **x := y *op* z**) entre deux variables (**y** et **z**) dont le résultat est rangé dans une troisième variable (**x**), soit un branchement conditionnel (de la forme **si x aller en a**) qui en fonction de la valeur d'une variable (**x**) fait sauter à une instruction donnée (l'instruction numéro **a**).

Exemple

0 x := y **et** y
1 y := y **et** z
2 **si** x **aller en** 0
3 z := x **ou** y

Ce programme prend en entrée des valeurs vrai ou faux pour les variables x, y et z. Puis à l'instruction 0 il met la valeur de y en x. Il passe ensuite à l'instruction 1 pour mettre la valeur de z dans y si y était vrai. À l'instruction 2, il passe à l'instruction 0 si x est vrai et à l'instruction 3 sinon. Donc, si au départ y et z sont vrais, le programme boucle indéfiniment. Sinon, le programme termine après avoir exécuté l'instruction 3.

2 Génération de codes à trois adresses

2.1 Suite pseudo-aléatoire

Pour générer des codes à trois adresses, on utilisera la suite d'entiers positifs (u_n) définie par :
– u_0 est donné sur votre table.
– $u_n = (10\,663 \times u_{n-1}) \bmod 100\,693$

Question 1 *Que valent* ***a)*** u_{564} *et* ***b)*** u_{15564} *?*

Question 2 *Quel est le plus grand* u_i *pour* ***a)*** $9 < i < 1\,000$ *et* ***b)*** $999 < i < 10\,000$ *?*

2.2 Paramètres

Pour la génération de codes à trois adresses, on utilisera trois paramètres entiers : le nombre de variables, V, le nombre d'instructions, I, et la probabilité d'avoir un branchement conditionnel, $P_{\texttt{si}}$. On supposera les variables numérotées de 0 à $V - 1$. Le code généré par la suite pseudo-aléatoires sera constitué de I instructions, chaque instruction

numéro i (entre 0 et $I-1$ donc) étant définie par $(u_{5i}, u_{5i+1}, u_{5i+2}, u_{5i+3}, u_{5i+4})$. On définit $\mathrm{Var}(x)$ comme étant la variable numéro x mod V, $\mathrm{Instr}(x)$ comme étant l'instruction numéro x mod I, et $\mathbf{Op}(x)$ comme étant **et** si x est pair et **ou** sinon.
L'instruction numéro i du code est l'instruction de branchement "**si** $\mathrm{Var}(u_{5i+2})$ **aller en** $\mathrm{Instr}(u_{5i+1})$" si et seulement si u_{5i} est un multiple de $P_{\texttt{si}}$. Sinon, l'instruction i sera "$\mathrm{Var}(u_{5i+1}) := \mathrm{Var}(u_{5i+2})$ $\mathbf{Op}(u_{5i+4})$ $\mathrm{Var}(u_{5i+3})$".

Exemple

Si la suite u_i commence par $1, 3, 1, 1, 2, 1, 1, 1, 2, 2, 2, 10, 3$ et si $P_{\texttt{si}} = 2$, $V = 3$ et $I = 10$, alors les premières instructions seront :

$$\begin{array}{ll} 0 & x_0 := x_1 \textbf{ et } x_1 \\ 1 & x_1 := x_1 \textbf{ et } x_2 \\ 2 & \textbf{si } x_0 \textbf{ aller en } 0 \end{array}$$

En effet, u_0 vaut 1 qui n'est pas multiple de 2, donc la première instruction est "$\mathrm{Var}(u_1) := \mathrm{Var}(u_2)$ $\mathbf{Op}(u_4)$ $\mathrm{Var}(u_3)$".

Question 3 *Quelle est l'instruction numéro $I/2$ pour $V = 3$, $P_{\texttt{si}} = 7$ et* ***a)*** *$I = 10$,* ***b)*** *$I = 100$ ou* ***c)*** *$I = 10\,000$?*

2.3 Affectations simples

Les affectations simples sont les instructions de la forme **x := y et y** ou **x := y ou y**.

Question 4 *Comptez le nombre d'affectations simples pour $P_{\texttt{si}} = 100$ et* ***a)*** *$I = 10$ et $V = 3$,* ***b)*** *$I = 100$ et $V = 10$ et* ***c)*** *$I = 10\,000$ et $V = 100$.*

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé, ainsi que sa complexité.*

2.4 Variables modifiées

Les variables modifiées par le programme sont les **x** telles qu'il existe une opération non triviale de la forme **x :=y op z** dans le programme. Une opération est non triviale si et seulement si elle n'est pas de la forme **z :=z op z**.

Question 5 *Comptez le nombre de variables modifiées pour $P_{\texttt{si}} = 100$ et* ***a)*** *$I = 10$ et $V = 10$,* ***b)*** *$I = 100$ et $V = 100$,* ***c)*** *$I = 10\,000$ et $V = 1\,000$ et* ***d)*** *$I = 1\,000$ et $V = 1\,000\,000$.*

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé, ainsi que sa complexité.*

3 Exécution des programmes

Un environnement est une fonction associant à chaque variable une valeur booléenne (vrai ou faux). L'exécution d'un programme consiste à prendre un environnement et à le modifier en exécutant les instructions du programme les unes à la suite des autres

jusqu'à ce que l'exécution s'arrête. L'exécution de l'instruction i du programme avec l'environnement E, si $i < I$, consiste à passer à l'instruction $SuivInst(i, E)$ avec l'environnement $SuivEnv(i, E)$. Si $i \geq I$, alors on dira que l'exécution s'arrête. Les fonctions $SuivInst(i, E)$ et $SuivEnv(i, E)$ sont définies par :

- Si l'instruction i est **x := y op z**, alors
  - $SuivInst(i, E)$ est $i + 1$.
  - $SuivEnv(i, E)$ est l'environnement qui associe à **x** la valeur de $E(\mathbf{y})\mathbf{op}E(\mathbf{z})$, et à toute variable **t** différente de **x**, $E(\mathbf{t})$.
- Si l'instruction i est **si x aller en a**, alors
  - $SuivInst(i, E)$ est **a** si $E(\mathbf{x})$ est vrai, $SuivInst(i, E)$ vaut $i + 1$ sinon.
  - $SuivEnv(i, E) = E$.
- $SuivInst(i, E)$ et $SuivEnv(i, E)$ ne sont pas définis pour $i \geq I$.

Formellement l'exécution du programme en partant de l'environnement E est définie comme les suites (i_j) et (E_j), avec $i_0 = 0$, $E_0 = E$, $i_{n+1} = SuivInst(i_n, E_n)$ et $E_{n+1} = SuivEnv(i_n, E_n)$. Si il existe un k avec $i_k = I$, alors la suite est finie et E_k est appelé l'environnement de fin du programme (pour l'environnement de départ E).

Question 6 *Montrez que, quelle que soit l'instruction i d'un programme, il est toujours possible de trouver un environnement de départ dont l'exécution passe par i.*

Dans la suite on partira de l'environnement qui associe vrai aux variables de numéro pair et faux aux variables de numéro impair.

Question 7 *Quelles sont les valeurs $E(0)$ et $E(1)$ associées aux deux premières variables avec E l'environnement de fin quand $P_{\texttt{si}} = 100\,694$ et* ***a)*** *$I = 10$ et $V = 3$,* ***b)*** *$I = 100$ et $V = 10$,* ***c)*** *$I = 10\,000$ et $V = 100$,* ***d)*** *$I = 1\,000$ et $V = 1\,000\,000$, et* ***e)*** *$I = 50\,000$ et $V = 10\,000\,000$.*

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé, ainsi que sa complexité.*

En général, il est possible qu'un programme ne termine pas (en partant d'un environnement de départ particulier). Dans ce cas, on dit qu'il boucle, et la longueur de la boucle est égale au nombre d'instructions exécutées avant de retrouver un couple (environnement, instruction) déjà vu.

Exemple

```
0  x := y et y
1  si x aller en 0
2  x := z ou z
3  si x aller en 1
```

Si l'environnement de départ associe faux à **y** et vrai à **z**, alors le programme boucle et la boucle est de longueur 5 (instruction 1 avec **x** vaut faux, puis (2, faux), (3, vrai), (1, vrai) et enfin (0, vrai), soit 5 instructions avant de retrouver (1, faux)).

Question 8 *Pour les programmes de la liste suivante, dire si le programme boucle, et si oui la taille de la boucle :*
- ***a)*** $P_{\texttt{si}} = 5$, $V = 5$, $I = 50$
- ***b)*** $P_{\texttt{si}} = 5$, $V = 10$, $I = 100$
- ***c)*** $P_{\texttt{si}} = 100$, $V = 50$, $I = 1\,000$
- ***d)*** $P_{\texttt{si}} = 1\,000$, $V = 500$, $I = 10\,000$
- ***e)*** $P_{\texttt{si}} = 10\,000$, $V = 5\,000$, $I = 100\,000$

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé, ainsi que sa complexité.*

4 Découpage en blocs

On définit un point d'entrée de bloc d'un programme comme étant soit 0, soit un entier i tel qu'il existe un k et l'instruction k du programme est un branchement conditionnel **si x aller en a** et $i = k + 1$ ou $i = \mathbf{a}$.
Un bloc du programme est une suite d'instructions du programme, consécutives de i à j, et de taille maximale telle que i soit le seul point d'entrée de bloc dans l'intervalle $[i, j]$.

Question 9 *Quel est le nombre de blocs et la taille du plus grand bloc pour les programmes défini par* $V = 3$ *et* ***a)*** $P_{\texttt{si}} = 5$ *et* $I = 10$, ***b)*** $P_{\texttt{si}} = 5$ *et* $I = 100$, ***c)*** $P_{\texttt{si}} = 20$ *et* $I = 10\,000$.

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé, ainsi que sa complexité.*

5 Variables utiles

Les variables utiles sont celles qui servent à déterminer si un programme boucle ou non sur une de ses entrées. Formellement, on définit les variables utiles à l'instruction i, $U(i)$, comme les plus petits ensembles qui satisfont les conditions suivantes :
- Si $i \geq I$ alors $U(i)$ est vide.
- Si l'instruction i est $\mathbf{x} := \mathbf{y}\ \mathbf{op}\ \mathbf{z}$,
  - si $\mathbf{x} \in U(i+1)$, alors $U(i) = (U(i+1) - \{\mathbf{x}\}) \cup \{\mathbf{y}, \mathbf{z}\}$
  - sinon, $U(i) = U(i+1)$
- Si l'instruction i est **si x aller en a**,
  - si $\mathbf{a} > i$ et $U(i+1)$ et $U(\mathbf{a})$ sont vides, alors $U(i)$ est vide aussi
  - sinon, $U(i) = \{x\} \cup U(i+1) \cup U(\mathbf{a})$

Question 10 *À combien de points d'entrées de bloc la variable numéro 0 est-elle utile si* $P_{\texttt{si}}$*=10 et* ***a)*** $V = 3$ *et* $I = 10$, ***b)*** $V = 10$ *et* $I = 100$ *et* ***c)*** $V = 100$ *et* $I = 5\,000$.

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé, ainsi que sa complexité.*

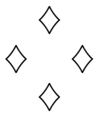
Question 11 *Calculez le nombre de variables utiles en 0 pour* $P_{\texttt{si}}$*=10 et* ***a)*** $V = 3$ *et* $I = 10$, ***b)*** $V = 10$ *et* $I = 100$ *et* ***c)*** $V = 100$ *et* $I = 5\,000$.

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé, ainsi que sa complexité.*

Question 12 *Calculez le nombre d'environnements d'entrée qui font boucler le programme pour*

- ***a)*** $P_{\texttt{si}} = 5$, $V = 3$ *et* $I = 10$,
- ***b)*** $P_{\texttt{si}} = 20$, $V = 10$ *et* $I = 100$,
- ***c)*** $P_{\texttt{si}} = 100$, $V = 20$ *et* $I = 400$.

★ *Vous présenterez à l'oral l'algorithme que vous avez utilisé, ainsi que sa complexité.*



Code à trois adresses

Nom, prénom, u_0: .....

Question 1

a)

b)

Question 2

a)

b)

Question 3

a)

b)

c)

Question 4

a)

b)

c)

Question 5

a)

b)

c)

d)

Question 6

Question 7

a)

b)

c)

d)

e)

Question 8

a)

b)

c)

d)

e)

Question 9

a)

b)

c)

Question 10

a)

b)

c)

Question 11

a)

b)

c)

Question 12

a)

b)

c)