ENSEIRB 2006-2007 - Département informatique - Première année
IF105 - Analyse d'algorithmes

Examen (session1)

Veillez à soigner la rédaction de vos solutions, particulièrement vos justifications et vos preuves. Soyez précis et concis. Bon courage et bonne chance !

Exercice 1 (3 points)

1. Donnez en quelques phrases la définition d'un problème de décision.
2. Donnez le principe d'un algorithme "brute-force" pour résoudre un problème de décision
3. En vous appuyant sur la structure de votre algorithme, à quelle(s) condition(s) un problème est-il dans la classe de complexité $\mathcal{NP}$?
4. À quelle(s) condition(s) un problème est-il dans la classe de complexité $\mathcal{P}$?

Exercice 2 (3 points)

Pour chacune des affirmations suivantes, indiquez si elle est vraie ou fausse en **justifiant** votre réponse. Les réponses non justifiées, même exactes, ne rapportent aucun point.

1. Les problèmes de la classe de complexité $\mathcal{P}$ sont ceux pour lesquels il est impossible de trouver une solution déterministe polynômiale
2. Il est possible de prouver la correction de n'importe quel while-program en utilisant la logique de Hoare
3. Il existe des problèmes que l'on peut résoudre avec une machine de Turing à 2 rubans, mais qu'il est impossible de résoudre avec une machine de Turing n'ayant qu'un seul ruban
4. Si un problème est résolu par une machine de Turing non-déterministe, alors il est aussi résolu par une machine de Turing déterministe
5. Les problèmes de correction et de terminaison des machines de Turing sont indécidables
6. L'ensemble des configurations accessibles d'une machine de Turing est calculable

Exercice 3 (6 points)

Le while-program suivant recherche un élément `x` (de type `elmt`) dans une liste chaînée `l` (de type `list`). E est une variable de type `ensemble`.

```
l' := l; E := ∅;
while (l' ≠ NULL) ∧ (head(l') ≠ x) do
  E := E∪{l'}
  l' := next(l');
endwhile
```

Le TAD `list` est muni des opérations :

```
NULL : list
head : list -> elmt
next : list -> list
```

qui définissent la liste vide (`NULL`), la valeur en tête de liste (`head`) et la liste suivante (`next`) respectivement.

On s'intéresse à la preuve de **terminaison** de cet algorithme par la méthode des ensembles bien fondés. On rappelle qu'un ensemble $(E, \leq)$ est bien fondé s'il n'existe pas de suite infinie strictement décroissante (relativement à $\leq$) d'éléments de E.

1. Pour tout ensemble X, on note 2^X l'ensemble des **parties finies** de X. Montrez que $(2^X, \subseteq)$ est un ensemble bien fondé.

2. Les valeurs de `l` et `l'` peuvent être assimilées aux adresses des premières cellules de ces listes. Soit A l'ensemble des adresses possibles et A_1 l'ensemble des adresses apparaissant dans `l`, pour toute liste `l`. Dans cet algorithme, que stocke l'ensemble `E` ?

   En vous appuyant sur cette remarque, proposez une mesure pour prouver la terminaison de cet algorithme. Vous motiverez votre choix.

3. Prouvez à l'aide de la méthode de Hoare que ce while-program **termine**. On pourra utiliser l'invariance de $E \cup A_{l'} = A_l$. Indiquez, en justifiant votre réponse, quelle(s) précondition(s) vous devez imposer sur `l` pour la preuve.

Exercice 4 (8 points)

Soit le while-program suivant :

```
p := x*y; mincm := p;
while ((p≫x) /\ (p≫y)) do
  if ((p%x=0) /\ (p%y=0)) then
    mincm := p
  endif;
  p := p-1
endwhile
```

qui calcule le plus petit commun multiple (PPCM) de `x` et de `y`, la variable `mincm` contenant le résultat attendu. L'opérateur `%` désigne le modulo, c'est à dire le reste de la division entière.

Les spécifications pour cet algorithme sont :

- PRECOND : $x \in \mathbb{N} \wedge y \in \mathbb{N}$
- POSTCOND : $mincm = ppcm(x, y)$

1. Simulez le fonctionnement de cet algorithme pour les valeurs $x = 3$ et $y = 4$. Vous remplirez le tableau suivant :

x	y	p	mincm
3	4		