Université Bordeaux 1. Master Sciences & Technologies, Informatique. Examen *Modèles de calcul* IN7W11, année 2012-2013, session 1. Date : 13 décembre 2012, 14h-17h (durée 3h).

Documents autorisés : 2 feuilles format A4, recto-verso.
Le barème est indicatif, à noter qu'il dépasse 20 points. On attachera une grande importance à la clarté et à la concision des justifications.

Exercice 1 (3 points)

> *Rappel :* Un ensemble D est *dénombrable* si on peut énumérer ses éléments (éventuellement avec répétitions). De manière équivalente, D est dénombrable s'il existe une bijection entre $\mathbb{N}$ et D. Exemples d'ensembles dénombrables : les ensembles finis, $\mathbb{N}$, $\mathbb{N}^k$, $\mathbb{Z}$, ...

1. Soit D un ensemble dénombrable. Justifiez que l'ensemble des *listes finies* avec éléments dans D, est dénombrable.
2. Une image de taille n est un tableau $n \times n$ avec entrées entières dans l'interval $[0, 255]$. Déduisez de la question précédente que l'ensemble des images de taille n, où $n \in \mathbb{N}$, est dénombrable.

Exercice 2 (5 points)

> *Rappel :* Une fonction primitive-récursive est une fonction construite à partir des fonctions $succ : \mathbb{N} \to \mathbb{N}$ et $zero^k : \mathbb{N}^k \to \mathbb{N}$, en utilisant les projections, la composition et la récursion :
>
> $$h(0, x_1, \ldots, x_k) = f(x_1, \ldots, x_n)$$
> $$h(n+1, x_1, \ldots, x_k) = g(n, h(n, x_1, \ldots, x_k), x_1, \ldots, x_k)$$
>
> Une fonction est primitive-récursive si et seulement si elle elle est calculable par un programme LOOP (avec instructions $x := y \pm c$, $x := c$, LOOP (x) DO P OD, IF $(x = 0)$ THEN P ELSE Q FI, RETURN c).
> Exemples : addition, multiplication, différence tronquée $\dot{-}$, sign, ...

1. Soit $f(x, k) : \mathbb{N}^2 \to \mathbb{N}$ une fonction primitive récursive. Montrez que la fonction suivante est primitive-récursive :
$$F(x, n) = \sum_{k=0}^{n} f(x, k)$$
2. Montrez que le prédicat $pair : \mathbb{N} \to \{0, 1\}$ est primitif-récursif :
$$pair(n) = \begin{cases} 1 & \text{si } n \text{ est pair} \\ 0 & \text{sinon} \end{cases}$$

3. Montrez que la fonction *racine* : $\mathbb{N} \to \mathbb{N}$ est primitive-récursive :

$$racine(n) = \lceil \sqrt{n} \rceil$$

Exercice 3 (3 points)

> *Rappel :* Un *semi-algorithme* pour un problème A est un programme P qui s'arrête sur toutes les instances positives de A, avec la réponse "OUI". Sur les instances négatives, P peut soit s'arrêter avec "NON", ou ne pas terminer. Un *algorithme* s'arrête sur toute instance.

1. Proposez un **semi-algorithme** qui résout la question suivante :
   Entrée : Polynôme $p(x_1, \ldots, x_k)$ dans k variables, avec coefficients entiers.
   Question : Est-ce qu'il existe $x_1, \ldots, x_k \in \mathbb{N}$ tel que $p(x_1, \ldots, x_k) = 0$?
2. Proposez un **algorithme** qui résout la question suivante :
   Entrée : Polynôme $p(x_1, \ldots, x_k)$ dans k variables, avec coefficients entiers, et entier $K \in \mathbb{N}$.
   Question : Est-ce qu'il existe $x_1, \ldots, x_k \in \mathbb{N}$ tel que $p(x_1, \ldots, x_k) = 0$ et $\max(x_1, \ldots, x_k) \leq K$?

Pour les deux questions, vous pouvez décrire le (semi-)algorithme **de manière informelle**.

Exercice 4 (2 points)

> *Rappels :* Un problème A est NP-complet s'il appartient à NP et il est NP-difficile. "A est NP-difficile" signifie que tout problème B dans NP peut se réduire à A par une réduction polynomiale.

Justifiez :

Si on peut résoudre un problème NP-complet en temps polynomial, alors P=NP.

Exercice 5 (3 points)

> Soit u, v deux sommets dans un graphe *orienté* G. On note par $d_{\max}(u, v)$ la valeur *maximale* ℓ pour laquelle il existe un chemin $u = w_0, w_1, \ldots, w_\ell = v$ sans sommet répété (donc, $w_i \neq w_j$ pour tout $i \neq j$) et de longueur ℓ de u à v dans G.

Montrez que le problème suivant est NP-complet.

Entrée : Un graphe orienté $G = (V, E)$ et un entier K.
Question : Est-ce qu'il y a deux sommets $u, v \in V$ dans G pour lesquels $d_{\max}(u, v) \geq K$?

Indication : Vous pouvez utiliser le fait que *Chemin hamiltonien* (existence d'un chemin qui passe par chaque sommet exactement une fois) est NP-complet.

Exercice 6 (2 points)

> Une formule booléenne propositionnelle est définie de manière inductive : une variable x ou sa négation $\neg x$ est une formule. Si φ_1, φ_2 sont des formules, alors $\varphi_1 \vee \varphi_2$ et $\varphi_1 \wedge \varphi_2$ sont des formules. Une formule $\varphi(x_1, \ldots, x_k)$ est satisfaisable s'il existe une valuation $\sigma : \{x_1, \ldots, x_k\} \to \{true, false\}$ pour laquelle φ s'évalue à *true*. *SAT* est la question si une formule booléenne propositionnelle est satisfaisble.

On considère dans cet exercice que P est un algorithme pour le problème *SAT* : P prend en entrée une formule booléenne propositionelle φ et répond "OUI" si et seulement si φ est satisfaisable.

1. Proposez un algorithme P' qui prend en entrée une formule booléenne propositionnelle et qui **calcule** une valuation qui la satisfait, si la formule est satisfaisable. L'algorithme P' peut se servir de l'algorithme P comme d'une "boîte noire", et son temps de calcul (excepté les appels à P) doit être *polynomial*.
2. Quel est le temps de calcul de P' en fonction du temps de calcul de P ?

Exercice 7 (2 points)

On considère le problème *Partition* suivant :

Entrée : Entiers positifs $x_1, \ldots, x_n$.
Question : Est-ce qu'on peut trouver un ensemble $I \subseteq \{1, \ldots, n\}$ d'indices tel que

$$\sum_{i \in I} x_i = \sum_{i \in \{1,\ldots,n\} \setminus I} x_i$$

1. Soit $S = \sum_{i=1}^{n} x_i$. Proposez un algorithme qui résout *Partition* en temps $O(nS)$.
2. Pourquoi la question précédente n'implique pas que *Partition* est résoluble en temps polynomial ?

Exercice 8 (4 points)

Le problème *Isomorphisme de graphes* est le suivant :

Entrée : Deux graphes orientés $G = (V, E)$ et $G' = (V', E')$.
Question : Est-ce qu'il existe une bijection $f : V \to V'$ tel que $(u, v) \in E$ si et seulement si $(f(u), f(v)) \in E'$?

Proposez une réduction polynomiale de *Isomorphisme de graphes* vers *SAT*.

Indication : Les variables de la formule associée à une instance de *Isomorphisme de graphes* ont la forme $x_{u,u'}$ pour chaque paire de sommets $(u, u') \in V \times V'$. On veut que $x_{u,u'} = true$ si et seulement si $f(u) = u'$.