Université Bordeaux 1. Master Sciences & Technologies, Informatique. Examen UE IN7W11, Modèles de calcul. Responsable A. Muscholl
Session 1, 2011–2012. 12 décembre 2011, 14h-17h.

Documents autorisés : transparents du cours et notes de TD.
Le barème est indicatif. A noter que le barème total dépasse 20 points.
On attachera une grande importance à la clarté et à la concision des justifications.

Exercice 1 (7 points) Répondez par "Vrai" ou "Faux" aux questions suivantes. **Détachez et joignez cette première feuille à votre copie.** Notation : $0,5$ par réponse juste et $-0,5$ par réponse fausse (l'absence de réponse vaut 0 points). Le résultat final sera le maximum entre 0 et la somme des points réalisés.

Rappels : $A \leq_P B$ signifie que le problème A se réduit au problème B par une réduction calculable en temps polynomial. Par PCP on note le problème *Correspondance de Post* (dominos), et par SAT le problème de satisfaisabilité de formules propositionnelles. **P** est la classe des problèmes où on peut *trouver* une solution en temps polynomial, et **NP** la classe des problèmes où on peut *vérifier* une solution en temps polynomial.

		Vrai	Faux
1	On peut énumérer les sous-parties de $\mathbb{N}$ de taille au plus 99.	☐	☐
	vrai : on énumère en ordre croissant sur la taille t ET la plus grande valeur m dans l'ensemble. Par exemple : $(s,m) = (1,1)$: $\{1\}, (s,m) = (1,2)$: $\{2\}, (s,m) = (2,1)$: $-, (s,m) = (2,2)$: $\{1,2\}$ etc	☐	☐
2	On peut énumérer les fonctions récursives.	☐	☐
	vrai : il suffit d'énumérer les expressions les définissants, et ces expressions sont des mots	☐	☐
3	Les mots d'un langage semi-décidable peuvent être énumerés en ordre hiérarchique.	☐	☐
	faux (X est *décidable* ssi on peut énumérer X en ordre hiérarchique, or il ya des langages semi-décidables, mais pas décidables	☐	☐
4	Si $A \subseteq \mathbb{N}$ et son complémentaire $\mathbb{N} \setminus A$ sont des ensembles semi-décidables, alors A est décidable.	☐	☐
	vrai : voir cours	☐	☐
5	Si A, B sont des ensembles tels que $A \subseteq B$ et B est décidable, alors A est aussi décidable.	☐	☐
	faux : par exemple $A \subseteq \{0,1\}^*$ où A est indécidable	☐	☐
6	Si A se réduit à B et B se réduit à C, alors A se réduit à C.	☐	☐
	vrai : voir cours	☐	☐
7	On sait décider s'il y a une infinité de nombres premiers.	☐	☐
	vrai : soit il y a un nombre infini de nombres premiers, ou il n'y en a pas - la réponse est soit toujours OUI, ou toujours NON	☐	☐
8	On sait décider si un programme WHILE s'arrête sur au moins une entrée.	☐	☐
	faux : voir cours	☐	☐
9	On sait réduire le problème PCP au problème SAT.	☐	☐
	faux : PCP est indécidable, on ne peut pas le réduire à un problème décidable comme SAT	☐	☐
10	On sait décider si une instance de PCP a une solution de longueur au plus 99.	☐	☐
	vrai : il suffit de chercher une telle solution - il n'y a qu'un nombre borné	☐	☐
11	$\mathbf{P} = \mathbf{NP}$ si au moins un problème **NP**-difficile appartient à **P**.	☐	☐
	vrai : voir cours	☐	☐
12	On sait décider si deux programmes WHILE calculent la même fonction.	☐	☐
	faux : voir cours	☐	☐
13	Si $A \in \mathbf{NP}$ et $A \leq_P B$, alors $B \in \mathbf{NP}$.	☐	☐
	faux : B peut être même indécidable	☐	☐
14	Si $B \in \mathbf{NP}$ et $A \leq_P B$, alors $A \in \mathbf{NP}$.	☐	☐
	vrai : voir cours	☐	☐

Exercice 2 (4 points) Justifiez que les ensembles suivants sont dénombrables.

- L'ensemble des polynômes à une variable, avec coéfficients entiers.
  Un polynome à une variable - par exemple $3x^3 - 2x + 4$ *- peut etre decrit comme une liste de ses coefficients. Dans l'exemple, ca serait* $(3, 0, -2, 4)$. *Or on peut montrer que l'ensemble des listes dont les éléments provienennt d'un ensemble dénombrable, est lui même dénombrable. Pour les listes d'entiers on peut par exemple les énumérer selon leur longueur & la valeur absolue maximale dans la liste. C'est a dire, pour chaque couple* (s, v) *d'entiers positifs, on énumère toutes les listes de longueur* s *dont la valeur absolue maximale est* v. *On a vu en cours comment énumérer les couples d'entiers positifs.*
- L'ensemble des matrices avec entrées rationnelles. *Une matrice* $n \times m$ *peut être representée par une liste, en énumérant les lignes une par une. Or, comme on a vu, l'ensembles des listes de rationnels est dénombrable - car l'ensemble des rationnels est dénombrable.*

Exercice 3 (6 points) Soit *Premier* le problème suivant :

- *Entrée :* entier n.
- *Sortie :* OUI si n est premier, NON sinon.

On peut voir *Premier* aussi comme une fonction, en remplaçant OUI par 1 et NON par 0.

Décrivez une solution pour *Premier* dans **un** des modèles de calcul suivants (*vous pouvez donc en choisir un !*)

1. Fonctions primitives-récursives.
   Vous avez vu en TD que la fonction reste(m, n) *qui donne le reste à la division entière de* m *par* $n > 0$, *est primitive-récursive. La fonction* $P(n, m) = 1 - sgn(reste(m, n+2))$ *vaut 1 si* $n + 2$ *est un diviseur de* m, *et 0 sinon.*
   On peut appliquer maintenant la minimisation bornée pour trouver le plus petit diviseur n *de* $m \geq 3$ *:*
   $$\min_B P(m-3, m) = \begin{cases} \min\{n \leq m-3 \mid P(n,m) = 1\} & \text{si tel } n \text{ existe} \\ m-2 & \text{sinon} \end{cases}$$
   Donc, m *est premier SSI* $\min_B P(m-3, m) = m - 2$ *SSI* $sgn(m - 2 - \min_B P(m - 3, m)) = 0$.
2. Programmes LOOP.

   $n = k = 2$; res $= 1$;
   LOOP $(m - 3)$ DO
       LOOP $(m - 3)$ DO
           IF $(m = n \cdot k)$ THEN res $= 0$;
           $k = k + 1$;
       $n = n + 1$

3. Machines de Turing.

   Vous pouvez utiliser une machine de Turing M_{div} qui travaille sur une entrée de la forme $\underbrace{1\cdots1}_{n}\#\underbrace{1\cdots1}_{k}$. La machine M_{div} finit son calcul avec le même contenu sur la bande (et la tête sur le premier 1), ainsi qu'en état final si k divise n, et état non-final sinon.

Exercice 4 (8 points)

1. Montrez que le problème suivant est décidable. Justifiez bien votre réponse.
   – *Entrée :* programme WHILE P, avec variables $x_0, \ldots, x_{k-1}$, et entier N.
   – *Sortie :* OUI, si le calcul de P à partir des valeurs initiales $\underbrace{0, \ldots, 0}_{k}$ est tel qu'aucune des variables ne dépasse la valeur N au cours du calcul.

   Attention : un programme WHILE peut tourner à l'infini, donc une simulation naive de P ne donne pas un algorithme de décision.

   Plutôt : on simule P à partir des valeurs initiales, en gardant toutes les valeurs (= tuplets) intermédiaires en mémoire. Si une de ces valeurs dépasse N, on répond NON. Si ce n'est pas le cas, on peut arrêter dès que on rencontre un tuplet déjà mémorisé, avec la même instruction actuelle de P, et on répind OUI.

2. Quel est le temps de calcul de votre algorithme pour la question précédente ?

   Exponentiel en N : le nombre de k-tuplets avec entrées $\leq N$ est $\leq (N+1)^k$.

3. Montrez que le problème suivant est indécidable. Justifiez bien votre réponse.
   – *Entrée :* programme WHILE P avec variables $x_0, \ldots, x_{k-1}$.
   – *Sortie :* OUI s'il existe un entier N tel que le calcul de P à partir des valeurs initiales $\underbrace{0, \ldots, 0}_{k}$ est tel qu'aucune des variables ne dépasse N.

   La différence avec le premier point est qu'on ne connait pas la borne N : c'est comme pour l'arrêt d'un programme, où on ne connait pas a priori la longueur du calcul.

   Pour montrer l'indécidabilité on peut réduire le problème de l'arrêt d'un programme WHILE à ce problème : on modifie le programme WHILE P pour lequel on pose la question de l'arrêt en rajoutant une variable x, qui est incrémentée à chaque instruction de P. Si P termine, alors il y a une valeur maximale pour le nouveau programme. Si ce n'est pas le cas, la variable x n'est pas bornée.

Exercice 5 (5 points) Rappels : un problème P appartient à la classe **NP** s'il existe un vérificateur polynomial V pour P : l'algorithme V reçoit en entrée un couple $\langle x, y\rangle$ et son temps de calcul est polynomial en fonction de la taille de x et de y. Une instance x de P est positive, si et seulement si il existe un y de taille polynomiale en x, et tel que V accepte $\langle x, y\rangle$.

Vous pouvez supposer pour cet exercice, ainsi que l'exercice suivant, que y est une séquence de bits (0 ou 1). La taille de y est donc la longueur de la séquence.

Justifiez que tout problème de la classe **NP** possède un algorithme dont le temps de calcul est exponentiel.

Pour savoir si x est instance positive, il faut trouver un y tel que l'algorithme V accepte $\langle x, y\rangle$. On peut faire une recherche exhaustive : le nombre de y est exponentiel en $|x|$.

Exercice 6 (5 points) Soit P un problème et $f : \mathbb{N} \to \mathbb{N}$ une fonction. Un f-vérificateur V pour P est un vérificateur polynomial (en fonction de la taille de x et de y) où la taille de y est bornée par $f(|x|)$.

Soit $\mathcal{C}_f$ la classe des problèmes qui possèdent un f-vérificateur.

1. Dans quelle classe de problèmes se trouve $\mathcal{C}_f$ quand f est un polynôme ? Justifiez votre réponse.

   Clsse **NP**, *voir définition de* **NP**.

2. Montrez que $\mathcal{C}_f \subseteq \mathbf{P}$, quand f est une fonction constante, $f(n) = c$ pour tout $n \in \mathbb{N}$. Généralisez votre argument pour $f(n) = \log(n)$.

   Si f est constant, cad ne dépend pas de x, alors le nombre de y est constant aussi. Donc on peut considérer les y un par un et voir si V accepte le couple $\langle x, y\rangle$.

   Si $f = \log$ le nombre de y est polynomial en $|x|$ ($2^{\log(n)} = n$). Donc on peut procéder comme avant, et appeler V un nombre polynomial de fois.

Exercice 7 (5 points) Pour le problème *Chemin long* on prend en entrée un graphe orienté $G = (V, E)$ (donc V ensemble de sommets et E ensemble d'arcs) et un entier k. On veut savoir s'il existe dans G un chemin *simple* avec k sommets. Un chemin simple est une suite $v_1, \ldots, v_k$ de sommets $v_i \in V$ tel que $v_i \neq v_j$ pour tout $i \neq j$, et $(v_i, v_{i+1}) \in E$ pour tout $1 \leq i < k$.

Dans cet exercice on construit une réduction polynomiale de *Chemin long* au problème SAT (satisfaisabilité des formules booléennes propositionnelles). La formule φ_G associée au graphe G sera la conjonction des trois formules demandées ci-dessous.

La formule φ_G a $k|V|$ variables booléennes de la forme $x_{v,i}$, pour $v \in V$ et $1 \leq i \leq k$. On veut que la variable $x_{v,i}$ prend la valeur “vrai” si le sommet v est le i-ème sommet du chemin.

1. Ecrivez une formule propositionnelle qui exprime que pour chaque $i \in \{1, \ldots, k\}$, il y au moins un sommet qui est le i-ème sommet du chemin.

$$\phi_1 = \wedge_{i=1}^{k} \vee_{v \in V} x_{v,i}$$

2. Ecrivez une formule propositionnelle qui exprime que pour chaque $i \in \{1, \ldots, k\}$, il n'y a pas deux sommets différents qui sont le i-ème sommet du chemin

$$\phi_2 = \wedge_{i=1}^{k} \wedge_{u \neq v} (\neg x_{u,i} \vee \neg x_{v,i})$$

3. Ecrivez une formule propositionnelle qui exprime que la solution représente bien un chemin.

$$\wedge_{i=1}^{k-1} \vee_{(u,v)\in E} \left(x_{u,i} \wedge x_{v,i+1}\right)$$