Examen terminal - Théorie de la complexité

THX

18 décembre 2018

La précision et la clarté de la rédaction est prise en compte dans l'évaluation.
Écrivez assez grand. Rédigez soigneusement. Les exercices sont indépendants.

Exercice 1. Soient deux chaînes de café C_1 et C_2 – deux franchises – qui veulent s'installer dans une région et sont donc en concurrence. Elles souhaitent placer des cafés sur des emplacements. Les chaines jouent au jeu suivant : d'abord C_1 ouvre un café sur un certain emplacement, puis C_2 ouvre un café, puis C_1, puis C_2, et ainsi de suite. Chaque chaîne essaie de trouver l'emplacement qui lui donne le plus de profit. Aussi, le choix de l'emplacement est régulé : il est interdit d'ouvrir deux cafés proches.

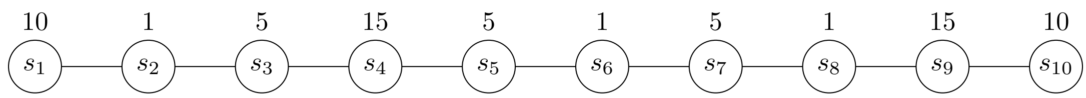
On modélise une région par un graphe non orienté $G = (S, A)$: un sommet $s \in S$ est un emplacement possible et $(s, s') \in A$ si les emplacements s et s' sont proches. Une fonction de gain $g : S \to \mathbb{N}$ donne le profit $g(s)$ que gagne une des chaînes si elle place un café en s. La régulation correspond au fait que l'ensemble de tous les emplacements occupés par un café forment un *ensemble indépendant* de G. Bien sûr, il y a au plus un seul café par emplacement.

On s'intéresse donc au jeu suivant. Les chaînes C_1 et C_2 jouent tour à tour, et c'est C_1 qui commence. A chaque tour, un des joueurs sélectionne un sommet de S. A chaque instant, l'ensemble des sommets sélectionnés forment un *ensemble indépendant* de G (on rappelle que $S' \subseteq S$ est indépendant si pour tout arête $(s, s') \in S'$, $(s, s') \notin A$).

On considère le problème de décision **CAFÉS EN CONCURRENCE** suivant :

- Entrée : un graphe non orienté $G = (S, A)$, une fonction de coût $c : S \to \mathbb{N}$, une borne B ;
- Sortie : oui, si C_2 a une stratégie telle que, quoique joue C_1, C_2 sélectionne un ensemble d'emplacements dont le gain total est au moins B ; non sinon.

Question 1. Expliquer une stratégie pour C_2 pour gagner au moins $B = 20$ sur le graphe suivant :

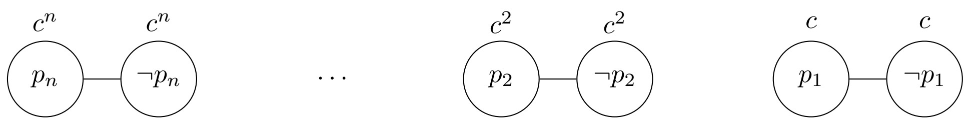


Question 2. Donner un algorithme *alternant en temps polynomial* qui décide le problème **CAFÉS EN CONCURRENCE**.

Question 3. En déduire que **CAFÉS EN CONCURRENCE** est dans PSPACE.

Question 4. Donner une réduction pour montrer que **CAFÉS EN CONCURRENCE** est PSPACE-dur. Rappeler les points qu'il faut montrer et expliquer l'intuition de votre construction (on ne donnera pas la démonstration complète).

Pour cela, on peut s'aider du problème TQBF (true quantified binary formula) où la formule propositionnelle apparaîssant dans une formule booléenne quantifiée est sous forme normale conjonctive. On peut utiliser le gadget suivant, avec une valeur de c bien choisie, et ajouter des sommets de gain 1 pour chaque clause :



TOURNEZ LA PAGE SVP

Exercice 2. Soit M une machine déterministe à un ruban qui décide un langage L en temps $2^{p(n)}$ (on suppose que $2^{p(n)} > n$), où n est la taille du mot w donnée en entrée. Pour tout mot w de taille n, pour tout $t \in \{0, \ldots, 2^{p(n)} - 1\}$, pour tout $x \in \{0, \ldots, 2^{p(n)} - 1\}$, on note $T_w[t, x]$ le contenu de la case (t, x) du "tableau" de l'exécution de M sur w.

Plus précisément, si, au temps t la machine est dans l'état q et qu'il y a la lettre a écrite dans la $x^{\text{ème}}$ case du ruban, alors :

$$T_w[t, x] = \begin{cases} (q, a) & \text{si le curseur est sous la } x^{\text{ème}} \text{ case au temps } t, \\ a & \text{sinon.} \end{cases}$$

Question 5. Dessiner un schéma du tableau T_w en indiquant la configuration initiale.

La valeur $T_w[t, x]$ est soit une lettre a, soit un couple (q, a) où q est un état de M et a est une lettre. Soit K le nombre de bits qu'il faut pour coder une valeur $T_w[t, x]$.

Question 6. Si l'alphabet de travail de M contient exactement 8 lettres, et que la machine a 31 états, donner la valeur de K minimale pour coder une valeur $T_w[t, x]$.

Pour tout $k \in \{1, \ldots, K\}$, on pose L_k le langage des mots encodant des triplets $\langle w, t, x \rangle$ où w est un mot,
$t \in \{0, \ldots, 2^{p(n)} - 1\}$ écrit en binaire, $x \in \{0, \ldots, 2^{p(n)} - 1\}$ écrit en binaire, tels que le $k^{\text{ème}}$ bit de $T_w[t, x]$ soit 1, avec n la taille du mot w.

Question 7. Quelle est la taille d'un mot qui encode un tel triplet $\langle w, t, x \rangle$?

Question 8. Expliquer pourquoi $L_k \in EXPTIME$.

Dans la suite, on considère des circuits booléens $C_1, \ldots, C_K$, qui prennent en entrée des représentations binaires de $t, x \in \{0, \ldots, 2^{p(n)} - 1\}$ dont on souhaite qu'ils calculent respectivement les bits numéro $1, \ldots, K$ de la valeur $T_w[t, x]$. On note $C_k[t, x]$ la sortie du circuit C_k sur l'entrée t, x. Dans la question suivante, nous allons donner un Π_1^P-algorithme (c'est-à-dire un algorithme alternant avec que des choix universels et en temps polynomial) qui vérifie la correction des $C_1, \ldots, C_K$ pour un certain mot w.

Question 9. Écrire un Π_1^P-algorithme pour le problème suivant :

- Entrée : un mot w et des circuits booléens $C_1, \ldots, C_K$;
- Sortie : oui si pour tout $k \in \{1, \ldots, K\}$, pour tout $t, x \in \{0, \ldots, 2^{p(n)} - 1\}$, $C_k[t, x]$ est le $k^{\text{ème}}$ bit de $T_w[t, x]$;

  non, sinon.

Question 10. Prouver la correction de votre algorithme.

Question 11. Conclure que $EXPTIME \subseteq P/poly$ implique que $EXPTIME = \Sigma_2^P$.