Complexité - TD 5

Benjamin Bordais
bordais@lsv.fr
`www.lsv.fr/~bordais/`

13 Décembre 2021

Question 1 (Planification) *On note $I = \{1, \ldots, n\}$. Un problème de planification est donné par :*

— *n variables booléennes $\{x_i\}_{i \in I}$;*
— *m opérations où chaque opération est définie par une condition de la forme $\bigwedge_{i \in I'}(x_i = \alpha_i)$ avec $I' \subseteq I$ et une mise à jour de la forme $\{x_i \leftarrow \beta_i\}_{i \in I''}$ avec $I'' \subseteq I$.*
Voici un exemple d'opération : Si $x_1 = \mathtt{V} \wedge x_3 = \mathtt{F}$ Alors $x_1 \leftarrow \mathtt{F}; x_2 \leftarrow \mathtt{V}$
— *Une configuration initiale s_{init} et une configuration finale s_{fin}.*
— *Une opération est applicable à une configuration si la condition de l'opération s'évalue à Vrai. Son application consiste à effectuer ses mises à jour pour obtenir la nouvelle configuration. Par exemple l'opération précédente est applicable à la configuration $(\mathtt{V}, \mathtt{F}, \mathtt{F})$ et conduit à la configuration $(\mathtt{F}, \mathtt{V}, \mathtt{F})$.*

Le problème consiste à déterminer s'il existe une suite d'applications des opérations (avec éventuellement plusieurs applications d'une même opération) qui conduise de la configuration initiale à la configuration finale.

1. *Montrez que le problème de planification est dans* PSPACE.
2. *Montrez que le problème de planification est* PSPACE*-dur. Pour cela, on montrera une réduction* LOGSPACE *du problème ci-dessous vers le problème de planification :*
   — *Entrée : Une machine $\mathcal{M}$, un espace t (en unaire) et un mot w ;*
   — *Sortie : $\mathcal{M}$ accepte w en espace t.*
   (Indice : pour simplifier, on supposera que $\mathcal{M}$ a une seule bande de travail, et l'efface à l'issue du calcul (une fois dans l'état acceptant ou rejetant), ainsi que replace les têtes de lecture au début des bandes correspondantes.)

Question 2 (Vacuité d'un automate déterministe concurrent) *Un automate déterministe concurrent $\mathcal{A}$ est donné par un ensemble d'automates déterministes $\{\mathcal{A}_i\}_{i \leq n}$ appelés composants. Un état de l'automate déterministe concurrent est un tuple $(s_1, \ldots, s_n)$ composé d'un état par composant. Lorsqu'une lettre a est lue, les automates $\mathcal{A}_i$ qui ont une transition issue de s_i étiquetée par la lettre effectuent simultanément leur transition tandis que les autres conservent leur état. Pour qu'une lettre puisse être lue, au moins un automate doit effectuer une transition. Un mot est reconnu s'il conduit à un tuple d'états terminaux. Le problème de la vacuité consiste à savoir s'il existe au moins un mot accepté par l'automate.*

1. *Montrez que le problème de la vacuité des automates déterministes concurrents est dans* PSPACE.
2. *Montrez que le problème de la vacuité est* PSPACE*-difficile depuis la problème de planification.*

Question 3 (Géographie) *Le jeu Géographie se joue de la manière suivante :*

- *Le jeu commence avec un nom donné de ville, par exemple* Cachan *;*
- *le premier joueur donne le nom d'une ville dont la première lettre coïncide avec la dernière lettre de la ville précédente, par exemple* Nice *;*
- *le deuxième joueur donne un autre nom de ville, commençant également par la dernière lettre de la vile précédente, par exemple* Évry *;*
- *le premier joueur joue à nouveau et ainsi de suite avec la restriction qu'aucun joueur ne peut donner le nom d'une ville déjà vu dans le jeu ;*
- *le perdant est le premier joueur qui ne peut trouver de nom de ville pour continuer.*

Ce jeu peut être décrit à l'aide d'un graphe orienté dont les nœuds représente les villes et où une arête (X, Y) signifie que la dernière lettre de X est la première lettre de Y. Ce graphe a également un nœud distingué qui correspond à la ville initiale. Chaque joueur choisit un nœud du graphe successeur du nœud précédent, le joueur 1 choisit en premier et ensuite les deux joueurs alternent. Un joueur gagne si, à un moment donné, l'autre joueur ne peut plus choisir de nœud qui n'a pas déjà été visité.

Géography Généralisée (i.e. GG*) correspond au problème suivant :*

- *Entrée : un graphe orienté G, et un nœud initial s.*
- *Sortie : le joueur 1 a une stratégie gagnante pour* GG *sur un graphe G depuis s*

1. *Montrer que* GG *est dans* PSPACE.
2. *Prouver que* GG *est* PSPACE*-dur via une réduction depuis* QBF.

Question 4 (Espace poly-logarithmique) *On rappelle le théorème de hiérarchie en espace :*

Théorème 0.1 *Pour deux fonctions constructibles en espace f et g telles que $f(n) = o(g(n))$, on a $\mathsf{DSPACE}(f) \subsetneq \mathsf{DSPACE}(g)$.*

On définit à présent la classe de complexité :

$$\mathsf{PolyLog} = \bigcup_{k>0} \mathsf{SPACE}(\log^k(n))$$

.

1. *Montrer que* PolyLog *n'a pas de problème complet pour des réductions en espace logarithmique. Que peut-on déduire quant à la comparaison entre les classes* P *et* PolyLog *?*
2. *On rappelle que $\mathsf{PSPACE} = \bigcup_{k>0} \mathsf{SPACE}(n^k)$. Est-ce que* PSPACE *a un problème complet pour des réductions en espace logarithmique ? Pourquoi est-ce que la preuve de la question précédente ne s'applique pas à* PSPACE *?*