Corrigé TD 4
Complexité

Luc Lapointe
luc.lapointe@ens-paris-saclay.fr
home.lmf.cnrs.fr/LucLapointe/

Vous pouvez reprendre les exercices des TD précédents si vous ne les aviez pas déjà tous traités.

Rappel Un langage est dans PSPACE s'il existe une machine de Turing M déterministe décidant le langage L qui termine sur toute entrée et telle qu'il existe un polynôme p tel que l'espace pris par la machine M sur toute entrée x de taille n prend un espace au plus $p(n)$. En particulier, le temps pris peut être exponentiel en n.

Exercice 1 – Mon tout premier problème PSPACE-complet

Montrer que le problème suivant est PSPACE-complet :

Entrée: Le code d'une machine de Turing M, un mot w, un entier t écrit en unaire.
Question: Est-ce que M accepte w en espace t ?

Idée de solution

Dans PSPACE : utiliser une machine universelle déterministe pour simuler M en se limitant à t cellules sur le ruban, en décrémentant 1^t à chaque blanc réécrit.

PSPACE-dur : On fait directement une réduction depuis un problème dans PSPACE quelconque. Soit $\mathcal{P}$ un problème dans PSPACE. Il existe donc une machine déterministe M qui résout $\mathcal{P}$ et s'exécute en espace au plus $p(n)$ sur une entrée de taille n, où p est un polynôme. On considère la réduction suivante : à x une entrée de $\mathcal{P}$, la réduction associe $\left(M, x, 1^{p(|x|)}\right)$.

Exercice 2 – QBF

On rappelle la définition de QBF (*Quantified Boolean Formula*), qui est PSPACE-complet :

Entrée: Une formule booléenne quantifiée

$$\varphi = Q_1 x_1 ... Q_n x_n : \psi(x_1, ..., x_n)$$

avec ψ une formule de la logique propositionnelle sans variable libre,
et Q_i valant $\exists$ ou $\forall$.
Question: Est-ce que φ est satisfiable ?

Que dire si tous les quantificateurs sont existentiels ? Et s'ils sont tous universels ?

Idée de solution

Si tous les quantificateurs sont existentiels, le problème décrit est SAT, qui est NP-complet. S'ils sont tous universels, c'est Tautologie, qui est co-NP-complet. Les deux ont été traités dans le précédent TD.

Exercice 3 – Jeu et complexité

Exercice tiré de [1].

Dans cet exercice, un *jeu* est un graphe orienté $G = (S, A)$ où l'ensemble des sommets $S = S_A \uplus S_B$ est partitionné en sommets appartenant à deux adversaires A et B, avec un sommet distingué s_0 qui sert de sommet initial. Le graphe est tel que chaque sommet a un successeur. Une *partie* correspond alors à un chemin fini ou infini $\rho = s_0 s_1 s_2 \ldots \in S^* \cup S^\omega$ avec s_0 le sommet initial. Si la partie en est au sommet $s_i \in S_A$ alors c'est le joueur A qui choisit le prochain sommet $s_{\{i+1\}} \in \mathrm{Succ}(s_i)$, sinon c'est la joueuse B. Une *condition de victoire* détermine lorsqu'une partie infinie est gagnée par A. On considère que si A ne gagne pas, c'est B qui gagne. Un joueur $J \in \{A, B\}$ a une *stratégie gagnante* à partir d'un sommet $s \in S$ si, à partir de toute partie qui a mené jusqu'à s, J a une manière de jouer pour gagner à coup sûr toute partie infinie.

1. Supposons que la condition de victoire soit une condition d'atteignabilité : étant donné un ensemble cible $T \subseteq S$, le joueur A gagne si et seulement si un sommet de T est vu. Montrer que décider le vainqueur d'un jeu d'atteignabilité à partir du sommet $s_0 \in S$ peut se faire en temps polynomial.
2. Considérons un entier $k \in \mathbb{N}$. Une condition d'atteignabilité k-généralisée est définie comme suit : étant donné k ensembles cibles $T_1, \ldots, T_k \subseteq S$, le joueur A gagne si et seulement si, pour tout $i \in [\![1, k]\!]$, un sommet de T_i est vu. Montrer que décider le vainqueur d'un jeu d'atteignabilité k-généralisé à partir du sommet $s_0 \in S$ peut être fait en temps polynomial.
3. Une condition d'atteignabilité généralisée est définie comme suit : étant donné plusieurs ensembles cibles $T_1, \ldots, T_k \subseteq S$, le joueur A gagne si et seulement si, pour tout $i \in [\![1, k]\!]$, un sommet de T_i est vu. La différence avec l'objectif k-généralisé est que le nombre de cibles n'est pas borné a priori. Montrer que décider le vainqueur d'un jeu d'atteignabilité généralisé est PSPACE-complet.
   Indice : Pour l'appartenance à PSPACE, on pourra utiliser que si le joueur A peut gagner avec k cibles, il peut voir toutes les cibles en au plus $k \cdot n$ étapes avec $n := |S|$.
4. Montrer que décider le vainqueur d'un jeu d'atteignabilité généralisé lorsque $S_B = \emptyset$ (i.e. le joueur A joue tout seul) est NP-complet.

Idée de solution

Les solutions détaillées sont disponibles dans [1]. Je ne donne que des ébauches extrêmement sommaires ici.

1. Calculer les attracteurs. Un *attracteur* pour A est un ensemble de sommets à partir desquels A peut s'assurer une victoire. Ils peuvent se calculer par itération : l'attracteur d'ordre 0 est T. L'attracteur d'ordre $n+1$ est l'ensemble des sommets contrôlés par A avec une arête pointant vers l'attracteur d'ordre n, ou contrôlés par B avec toutes les arêtes pointant vers l'attracteur d'ordre n. S'arrêter dès qu'un point fixe est atteint. La complexité est quadratique en la taille du graphe.
2. Rajouter 2^k sommets au graphe pour retenir quels ensembles ont été visités ou non, et résoudre comme un jeu d'accessibilité à un ensemble.
3. Dans PSPACE : On utilise PSPACE $=$ NPSPACE. On joue une partie en retenant les sommets visités. Lorsqu'on rencontre un sommet rencontré par le joueur actif, on devine le bon coup. Lorsqu'on rencontre un sommet contrôlé par l'adversaire, on énumère tous ses coups possibles (par exemple avec l'ordre lexicographique) en vérifiant que toutes les possibilités adverses mènent à une victoire. Lorsqu'on rencontre un sommet déjà visité, où qu'on dépasse $k * |S|$ sommets, on conclut à une défaite.
   PSPACE-dur : Réduction depuis QBF. Le graphe du jeu est une suite de sommets « choix » qui doivent aller soit dans x_i soit dans $\neg x_i$. Les quantificateurs existentiels, resp. universels, donnent des choix contrôlés par le joueur actif, resp. l'adversaire. Pour les régions gagnantes, on peut passer la formule ψ en entrée de QBF en une forme normale conjonctive en utilisant un espace logarithmique, avec par exemple la transformation de Tseitin.
4. Dans NP : Deviner un chemin de taille $k * |S|$.
   NP-dur : Même transformation qu'avant, mais en partant de SAT.

Bibliographie

[1] N. Fijalkow et F. Horn, « The surprizing complexity of generalized reachability games », *arXiv preprint arXiv:1010.2420*, 2010.