Théorème d'Immerman-Szelepcsényi

Benjamin Hellouin

Théorème 1 (Théorème d'Immerman-Szelepcsényi)**.**
Pour toute fonction $S(n) \geq \log(n)$, $\mathbf{NSPACE}(S(n)) = \mathbf{coNSPACE}(S(n))$.

On utilisera la vision de "deviner nondéterministiquement" pour avoir une meilleure intuition, mais on passe facilement à la vision avec certificat.

Définition 1. *Le problème décisionnel* ***PATH*** *est défini comme suit :*
Entrée : un graphe orienté $G = (V, E)$ *et deux sommets* s *(source) et* c *(cible) ;*
Sortie : OUI s'il existe un chemin de s *à* c *dans* G*, NON sinon.*

Lemme 1. ***PATH*** *est* ***NL****-complet.*

Voici un algorithme qui décide **PATH** en espace logarithmique :

$v \leftarrow s$
Tant que $v \neq c$ **faire**
  deviner le prochain sommet v'
  vérifier $(v, v') \in E$
  $v \leftarrow v'$

Soit P un problème de **NL** accepté par une machine M. Si on suppose que M n'a qu'une seule configuration acceptante (nettoyer le ruban et amener la tête à l'origine avant d'accepter), on voit que le problème d'acceptation d'une instance par M est équivalent à la recherche d'un chemin dans son graphe des configurations. De plus, une machine de Turing travaillant en espace logarithmique a un graphe de taille $2^{O(\log(n))} = P(n)$, donc **PATH** sera bien en espace logarithmique.

Lemme 2. ***PATH*** $\in$ ***coNL****.*

On note C_i = sommets atteignables en $\leq i$ étapes à partir de s. L'appartenance à C_i peut être décidée en espace logarithmique avec l'algorithme précédant, en y ajoutant un compteur (espace logarithmique) ; on veut prouver qu'il en est de même pour la non-appartenance, car le problème revient à décider si $c \notin C_{n-1}$.

Supposons qu'on a calculé en espace logarithmique la valeur de $|C_i|$, qui est écrite sur le ruban. L'algorithme suivant décide si $v_a \notin C_{i+1}$, où on a noté $V = \{v_1 \dots v_n\}$:

$c \leftarrow |C_i|$
Pour tout $1 \leq k \leq n$ **faire**
  deviner si $v_k \in C_i$
  Si oui **alors**
    vérifier que $v_k \in C_i$, $v_a \neq v_k$ et $(v_a, v_k) \notin E$

$c \leftarrow c-1, k \leftarrow j$

Si $c = 0$, on a vérifié que $v_a \notin C_{i+1}$.

On peut maintenant écrire l'algorithme général, qui calcule les $|C_i|$ par ordre croissant, sachant que $C_1 = 1$:

$|C_0| \leftarrow 1$
Pour tout $1 \leq i \leq n-2$ **faire**
  $|C_i| \leftarrow 0$
  Pour tout $1 \leq k \leq n$ **faire**
    en devinant quel cas s'applique, vérifier si $v_k \in / \notin C_i$
    le cas échéant, $|C_i| \leftarrow |C_i + 1|$
  effacer $|C_{i-1}|$

Conclusion. Une fois $|C_{n-2}|$ connu, on s'en sert pour vérifier que $t \notin C_{n-1}$ de la même manière qu'à l'étape 1.