6.7 Recherche de stratégies gagnantes sur les jeux finis

Un jeu est dit *fini* si les relations de transitions des deux joueurs (δ_1 et δ_2) ne dépendent pas du second argument.

Exercice 24 Démontrer le lemme de détermination des jeux sans la condition de faiblesse, dans le cas d'un jeu fini.

On considère dans ce paragraphe des jeux finis et on s'intéresse à la complexité de la recherche de stratégies gagnantes. Ce problème est lié à la vérification de modèles (finis) et au problème d'accessibilité dans les graphes ET/OU.

Un graphe (dirigé) ET/OU est un ensemble de sommets S, un ensemble d'arêtes $R \subseteq S \times S$, ainsi qu'une application e de S dans $\{\wedge, \vee\}$. Etant donné un graphe ET/OU G, on note $\xrightarrow[G]{*}$ la relation *d'accessibilité* définie par: $\xrightarrow[G]{*}$ est le plus petit sous-ensemble de $S \times 2^S$ tel que:

- si $s^0 \in S_1$, alors $s^0 \xrightarrow[G]{*} S_1$
- si $e(s^0) = \vee$ et il existe un sommet s^1 tel que $(s^0, s^1) \in R$ et $s^1 \xrightarrow[G]{*} S_1$, alors $s^0 \xrightarrow[G]{*} S_1$
- si $e(s^0) = \wedge$ et pour tout sommet s^1 tel que $(s^0, s^1) \in R$, $s^1 \xrightarrow[G]{*} S_1$, alors $s^0 \xrightarrow[G]{*} S_1$.

Theorème 31 *L'accessibilité dans un graphe ET/OU est décidable en temps linéaire.*

Preuve: Etant donné un ensemble de sommets S_1, on calcule le plus petit point fixe de l'application qui à S associe l'ensemble des sommets desquels on peut accéder à S en au plus une étape. Plus précisément:

1. Initialement, $S = S_1$, $T = S_1$
2. Tant que $T \neq \emptyset$:
   - On calcule l'ensemble T des sommets s qui ne sont pas dans S et tels que
     - Ou bien $e(s) = \vee$ et l'un des successeurs de s est dans S.
     - Ou bien $e(s) = \wedge$ et tous les successeurs de s sont dans S
   - On remplace T par $T \cup S$
3. $s^0 \xrightarrow[G]{*} S_1$ si et seulement si $s^0 \in S$

Remarquons que cet algorithme naïf de calcul de point fixe est bien polynômial (quadratique). Mais on peut trouver un algorithme linéaire en étant un peu plus rusé, que nous expliquons ci-dessous.

On peut tout d'abord se ramener, en temps linéaire, au cas où chaque sommet du graphe a un degré sortant au plus deux.

A chaque sommet s du graphe est attaché un quadruplet (e(s),val(s),flag(s),ancestors(s)). $e(s) \in \{\wedge, \vee\}$, $val(s), flag(s) \in \{\mathbf{true}, \mathbf{false}\}$ et ancestors(s) est la liste des sommets tels qu'il existe une arête (s', s) dans le graphe. Initialement, flag(s) =**false**pour tous les sommets ayant deux successeurs et flag(s)=**false**pour les sommets ayant 0 ou 1 successeur. val(s) =**true**si et seulement si $s \in S_1$. Cette construction peut être effectuée en temps linéaire.

On maintient ensuite une piles P. Initialement, P contient les sommets de S_1. L'algorithme procède comme suit:

Tant que la pile P est non vide:

1. retirer le sommet de pile s.
2. pour chacun des sommets t de ancestors(s) faire:
   (a) Si $val(t) = \mathbf{true}$, ne rien faire
   (b) Si $e(t) = \vee$ et $val(t) = \mathbf{false}$, alors mettre à jour $val(t)$ à **true** et empiler t sur P
   (c) Si $e(t) = \wedge$ et $flags(t) = \mathbf{true}$, alors mettre à jour $val(t)$ à **true** et empiler t sur P
   (d) Si $e(t) = \wedge$ et $flags(t) = \mathbf{false}$, alors mettre à jour $flags(t)$ à **true**.

Si la pile P est vide, les sommets s desquels on peut accéder à S_1 sont ceux tels que $val(s) = \mathbf{true}$ et l'algorithme s'arrête

On vérifie que l'algorithme est linéaire: chaque sommet est empilé au plus une fois et, pour chaque sommet dépilé s, le nombre de mises à jour est proportionnel au nombre d'ancêtres de s; au total, le nombre de mises à jour est proportionnel au nombre d'arêtes.

Remarquons que, en cas d'accessibilité, on peut effectivement calculer un "chemin" en temps linéaire. Un tel chemin c associe à chaque sommet t tel que $e(t) = \vee$ un successeur à partir duquel S_1 est accessible. Dans la présentation ci-dessus, il suffit dans le cas 2b de garder $c(t) = s$.

□

En fait, le problème est complet pour PTIME, mais ce résultat est admis ici.

Theorème 32 *Supposant l'ensemble d'états fini, on peut décider quel joueur a une stratégie gagnante en temps linéaire (et calculer cette stratégie).*

Preuve: On montre par récurrence sur le rang maximal d'une configuration que l'on peut calculer en temps linéaire, pour chaque configuration q, quel est le joueur qui a une stratégie gagnante à partir de q et aussi donner effectivement cette stratégie.

Si tous les états sont de même rang, le joueur 1 a une stratégie gagnante si et seulement si ce rang est pair. Dans ce cas, la stratégie gagnante est arbitraire.

Supposons maintenant que M est le rang maximal. On décompose les positions du jeu en $Q_0 \uplus \ldots \uplus Q_R$, Q_i étant l'ensemble des configurations de rang i. Le jeu étant faible la restriction des transitions aux configurations de $Q_0 \uplus \ldots \uplus Q_{M-1}$ est encore un jeu à parité faible lorsque la configuration initiale est choisie parmi ces états. Par hypothèse de récurrence, on peut supposer associé à chacun des états $q \in Q_0 \uplus \ldots \uplus Q_{M-1}$ le joueur gagnant $J(q)$ à partir de cet état et sa stratégie $S(q)$.

Considérons alors le graphe ET/OU dont les sommets sont les configurations Q_M plus deux sommets particuliers 1 et 2. Les arêtes sont données par les relations de transition: pour $s, s' \in Q_M$, $(s, s') \in R$ si et seulement si $s' \in \delta_i(s)$. De plus, si $s' \in \delta_i(s)$ et $r(s') < M$, alors on ajoute l'arête $(s, J(s'))$. $e(s)$ est $\vee$ si s est une configuration dans laquelle c'est au joueur 1 de jouer. $e(s)$ est $\wedge$ sinon. Si 1 est accessible à partir de q, alors le joueur 1 a une stratégie gagnante à partir de q. Si 2 est accessible à partir de q, dans le graphe dual (où les $\wedge$ et les $\vee$ sont échangés) alors le joueur 2 a une stratégie gagnante à partir de q. Si ni 1 ni 2 ne sont accessibles à partir de q dans G et le graphe dual respectivement, alors, ou bien M est pair et le joueur 1 a une stratégie gagnante, ou bien M est impair et le joueur 2 a une stratégie gagnante.

Comme, d'après le théorème 31, l'ensemble des sommets desquels 1 est accessible (resp. 2) peut être calculé en temps linéaire, on peut calculer $J(q)$ pour $q \in Q_M$ en temps linéaire.

De plus, la stratégie du joueur 1 (par exemple) est fournie par le chemin qui permet d'accéder à 1, dans le cas où 1 est accessible. Dans l'autre cas (1 n'est pas accessible dans le graphe, 2 n'est pas accessible dans le graphe dual et M est pair), il suffit d'associer à chaque sommet $\wedge$ du graphe dual un des sommets successeurs qui n'est pas étiqueté par **true**. $\square$

Comme conséquence du théorème 32 et du lemme 25:

Corollaire 33 *Le problème du vide pour les automates alternants à parité faibles sur un alphabet à une lettre peut être résolu en temps linéaire.*

Theorème 34 *Supposant l'ensemble d'états fini, déterminer si le joueur 1 a une stratégie gagnante dans un jeu à parité est dans NP $\cap$ co-NP.*

Preuve: On devine la stratégie ainsi que l'ensemble S des états de rang pair minimal rencontrés infiniment souvent. Vérifier que la stratégie est gagnante revient à décider que l'on peut atteindre S dans un graphe ET/OU, puis que l'on peut atteindre S à partir d'un état de S, dans le même graphe. Comme l'accessibilité dans un graphe ET/OU est polynômial (théorème 31), cette vérification se fait en temps polynômial. Il en résulte que le problème est dans NP. Par détermination des jeux à parité, le joueur 1 n'a pas de stratégie gagnante si et seulement si le joueur 2 a une stratégie gagnante, ce qui est dans NP à nouveau. Donc le problème est dans NP $\cap$ co-NP. $\square$

Comme conséquence du théorème 32 et du lemme 25:

Corollaire 35 *Le problème du vide pour les automates alternants avec condition de parité sur un alphabet à une lettre est dans NP $\cap$ co-NP.*

7 Logique monadique de second ordre à k successeurs SkS

Ce paragraphe n'est qu'un rapide survol.

Syntaxe

- **true**, **false**
- $t \in X$
- $t = \epsilon$, $t = S_i(t')$, pour $i = 1, ..., k$, $t = t'$, $t \geq t'$
- $\phi \vee \psi$, $\phi \wedge \psi$, $\neg \phi$
- $\forall t.\phi$, $\exists t.\phi$
- $\exists^2 X.\phi$, $\forall^2 X.\phi$

Sémantique Domaine d'interprétation des variables du premier ordre: $\{1, ..., k\}^*$ avec les opérations: concaténation à droite par $1, ..., k$, mot vide, ordre préfixe.

Domaine d'interprétation des variables de second ordre: sous-ensembles de $\{1, ..., k\}^*$.

Exercice 25 Exprimer en SkS les prédicats suivants:

1. Pair(X) qui est satisfait par l'ensemble des positions de longueur paire et cet ensemble seulement.
2. Chemin(X) qui est satsifait par les ensembles qui représentent des chemins infinis
3. Infini(X) satisfait par les ensembles infinis de positions
4. Origine(x, X) satisfait par les chemins X dont l'orgine est x
5. Succ(X, Y) satisfait par les chemins X et Y tels que Y est X privé de son origine

Exercice 26 Codage d'un arbre quelconque à branchement fini: si un nœud possède k fils: 21, ... $2^k 1$. Plus précisément, $c(i \cdot w) = 2^i \cdot 1 \cdot c(w)$. On obtient ainsi une application dans l'arbre binaire, pour laquelle l'image du codage est expressible dans $S2S$ et le codage injectif. Si bien que $S_1, ..., S_n \models \phi$ ssi $c(S_1), ..., c(S_n) \models c(\phi)$.

De même que pour S1S, si $X_1, \ldots, X_n$ sont les variables libres (de second ordre) d'une formule ϕ de SkS, ϕ définit un langage (ensemble d'arbres) sur l'alphabet $\{0, 1\}^n$: un arbre (infini) de branchement k et étiqueté par $\{0, 1\}^n$ est une application de $\{1, ..., k\}^*$ dans $\{0, 1\}^n$. Un arbre T satisfait ϕ si les ensembles $S_1, ... S_n$ définis par: S_i est l'ensemble des positions de T dont l'étiquette a pour ième composante 1, satisfont ϕ. On note L_ϕ le langage ainsi défini.

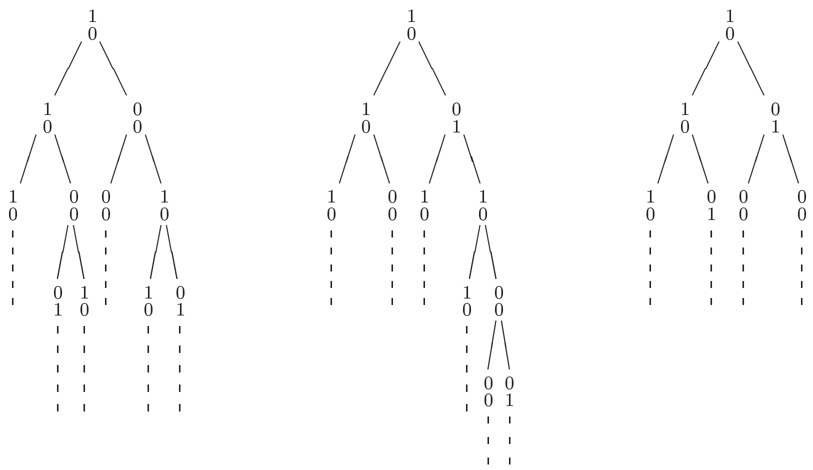

Figure 16: Quelques arbres binaires sur l'alphabet $\{0,1\}^2$

Exemple 36 Soit ϕ la formule

$$\begin{array}{rl} 0 \in X \wedge \forall x. & (x \in X \Rightarrow x \notin Y \wedge x \cdot 1 \in X \wedge x \cdot 2 \notin X) \\ \wedge & (x \in Y \Rightarrow x \notin X \wedge x \cdot 1 \notin Y \wedge x \cdot 2 \notin Y) \end{array}$$

Les arbres de la figure 16 satisfont ϕ.

7.1 Automates d'arbres

Un automate d'arbres est un tuple (Q, q_0, A, δ, F) où Q est un ensemble fini d' états, $q_0 \in Q$ est l'état initial, A est un ensemble fini de symboles d'entrée, F est une condition d'acceptance (nous serons plus précis ultérieurement) et δ est un ensemble de règles de la forme:

$$q(f(x_1, \dots, x_k)) \to f(q_1(x_1), \dots, q_k(x_k))$$

Autrement dit:

$$\delta \subseteq Q \times F \times Q^k$$

Un *calcul* ρ de l'automate sur un arbre infini T (de branchement constant k) étiqueté par A est un arbre étiqueté par Q (i.e. une application de $\{1, \dots, k\}^*$ dans Q tel que $\rho(\epsilon) = q_0$ et:

$$\left.\begin{array}{r} \rho(t) = q \\ \forall i = 1..k,\ \rho(t \cdot i) = q_i \\ T(t) = f \end{array}\right\} \Rightarrow \exists q(f(x_1, \dots, x_k)) \to f(q_1(x_1), \dots, q_k(x_k)) \in \delta$$

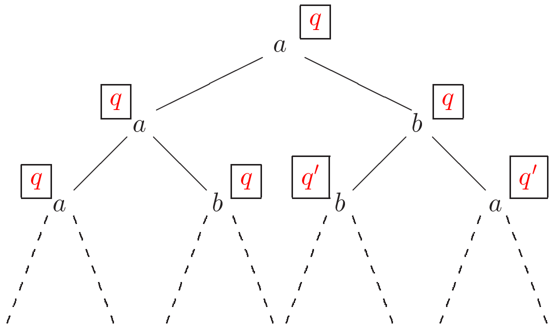

Figure 17: Un calcul d'automate d'arbres

Exemple 37 Soit l'automate défini par: $Q = \{q, q'\}, q_0 = q$, $A = \{a, b\}$, et δ est donné par:

$$\begin{array}{rcl} q(a(x,y)) & \to & a(q(x), q(y)) \\ q'(a(x,y)) & \to & a(q(x), q(y)) \\ q(b(x,y)) & \to & b(q'(x), q'(y)) \\ q'(b(x,y)) & \to & b(q'(x), q'(y)) \end{array}$$

Un calcul de l'automate est représenté sur la figure 17.

La *condition d'acceptance de Büchi* est donnée par $F \subseteq Q$; un calcul de l'automate est réussi si toutes ses branches contiennent une infinité d'éléments de F. Un arbre T est accepté par l'automate $\mathcal{A}$ s'il existe au moins un calcul réussi sur T.

Exemple 38 Si l'automate de l'exemple 37 est complété par la condition d'acceptance de Büchi $F = \{q\}$, l'automate accepte les arbres dont toutes les branches contiennent une infinité de a.

Exercice 27 Donner un automate de Büchi d'arbres qui accepte les arbres binaires sur l'alphabet $\{a, b\}$ tels qu'au moins une branche contient une infinité de a.

Par contre, le complémentaire du langage de cet exemple n'est pas accepté par un automate d'arbres de Büchi: avec cette condition d'acceptance, il n'y a pas stablité par complémentaire.

La *condition d'acceptance de Rabin* est donnée par un ensemble fini de paires de sous ensembles de Q: $F = \{(L_1, U_1), \ldots, (L_n, U_n)\}$. Un calcul ρ sur T est *réussi* si, pour tout chemin π de ρ, il existe un indice i tel que π ne contienne qu'un nombre fini d'éléments de L_i et un nombre infini d'éléments de U_i.