Algorithmique des graphes appliquée au jeu ChocoBar

Master 1 2004-2005
A effectuer par groupe de 2 étudiants

A rendre le vendredi 17 décembre 2004 à 21h

Table des matières

1 Introduction au jeu ChocoBar

Le jeu ChocoBar est un jeu à deux joueurs, joué sur une barre de chocolat ayant $M \times N$ carrés. Le carré du coin supérieur gauche est empoisonné. Les joueurs jouent chacun à leur tour en mangeant un ou plusieurs carrés de chocolat. Quand c'est son tour, un joueur doit prendre un des carrés et tous ceux qui sont en dessous et à droite du carré choisit. En considérant que la barre initiale est une matrice $M \times N$, l'entree $(0, 0)$ est empoisonnée, et l'état initial du jeu est la barre initiale, c'est-à-dire toutes les entrées (i, j) telles que $0 \leq i \leq M-1$ et $0 \leq j \leq N-1$. Choisir le carré (i_0, j_0) implique qu'il faut manger tous les carrés (i, j) restants tels que $i_0 \leq i$ et $j_0 \leq j$. Le joueur qui doit manger le carré empoisonné $(0, 0)$ meurt dans d'atroces soufrances et donc perd le jeu ! Ainsi, un joueur qui ne veut pas se suicider ne choisira le carré $(0, 0)$ que s'il est forcé de le faire, c'est-à-dire si c'est le seul choix possible quand son tour arrive.

Par exemple, si $M = 5$ et $N = 3$, la configuration initiale du jeu est la suivante :

```
X X X
X X X
X X X
```

```
X X X
X X X
```

Si le premier joueur choisit de jouer $(4, 2)$, la configuration suivante du jeu est :

```
X X X
X X X
X X X
X X X
X X
```

Si le deuxième joueur choisit le jouer $(1, 1)$, la configuration suivante du jeu est :

```
X X X
X
X
X
X
```

et ainsi de suite.

Le jeu ChocoBar fait partie d'une classe de jeux appelée "jeux à 2 joueurs à information complète"[1]. Ces jeux sont joués par deux joueurs et, à chaque instant, chacun possède toutes les informations sur l'état du jeu. Aucun élément de hasard n'intervient. D'autres exemples de tels jeux sont : le jeu d'échecs, le jeu de dames, le jeu géographie (indiquer le nom d'un pays qui commence par la dernière lettre du nom du pays indiqué par l'adversaire), etc...

2 But du projet

Le but du projet est de modéliser ce jeu à l'aide de graphes et de programmer des algorithmes qui permettent de trouver la stratégie gagnante.

3 Modélisation

Ce jeu se modélise naturellement par son *graphe des positions* $G = (V, E)$. Chaque sommet du graphe $v \in V$, appelé *position*, correspond à un état du jeu (ce qui reste de la barre de chocolat) et le nom du joueur qui doit jouer. On a donc une partition de l'ensemble des sommets $V = V_A \biguplus V_B$ entre l'ensemble V_A des sommets appartenant au joueur A et l'ensemble V_B des sommets appartenant au joueur B.

Un arc relie deux positions u et v ($(u, v) \in E$) si la position v peut être atteinte en un coup à partir de la position u. On veut que les joueurs alternent leurs coups, c'est-à-dire que $E \subseteq V_A \times V_B \cup V_B \times V_A$. Une *partie* finie (resp. infinie) dans G est un chemin $v_0, v_1, \ldots, v_k$ (resp. $v_0, v_1, \ldots$) dans G.

Dans le graphe du jeu G, un sous-ensemble de sommets $F \subset V$ est considéré comme ensemble des *sommets finaux*. Dans le jeu du ChocoBar, les sommets

[1] Dans la "théorie des jeux", qui est une branche bien développée des mathématiques, il existe d'autres types de jeux, par exemple les jeux de coloriage de graphes, les jeux probabilistes, etc.

finaux (au nombre de deux) sont tels qu'il ne reste que le carré $(1, 1)$. Le sommet initial, noté $v_0 \in V$, est celui où la barre est complète et où c'est au joueur A de jouer.

Une *stratégie*[2] pour un joueur, disons A, est une fonction f qui indique à A quel coup choisir à partir de tout sommet de V_A. Formellement, une stratégie pour A est une fonction $f : V_A \to V_B$ telle que pour tout $v \in V_A$, $(v, f(v)) \in E$.

La *stratégie* f est *gagnante* pour A si toutes les parties où A joue selon f sont gagnantes pour A, quels que soient les coups de B.

Pour le jeu ChocoBar, comme pour tout jeu fini et déterministe, il a été prouvé qu'il y a toujours un gagnant (quelque soit le sommet initial) et que le gagnant a (au moins) une stratégie gagnante sans mémoire. Par exemple, une stratégie gagnante existe pour un des joueurs dans le jeu d'échecs, mais elle n'est pas calculable dû à la taille du graphe qui est énorme : on estime à 10^{150} le nombre de positions aux échecs !

4 Résolution des jeux d'accessibilité

Dans un jeu d'accessibilité on définit une partie comme gagnée par A, si elle passe *au moins une fois* par F. On peut calculer les sommets initiaux à partir desquels A gagne le jeu d'accessibilité de manière itérative en calculant des attracteurs $\text{Attr}^0 \subseteq \text{Attr}^1 \subseteq \cdots \subseteq V$:

- $\text{Attr}^0 = F$
- $\text{Attr}^{k+1} = \text{Attr}^k \cup \{v \in V_A \mid \exists w \in \text{Attr}^k, (v, w) \in E\} \cup \{v \in V_B \mid \forall w : (v, w) \in E \Rightarrow w \in \text{Attr}^k\}$

L'ensemble Attr^k représente l'ensemble des sommets à partir desquels A peut atteindre F en au plus k coups, quels que soient les coups de B.

Comme V est fini, on a bien $\text{Attr}^{k_0} = \text{Attr}^{k_0+1}$ à partir d'un certain k_0. L'ensemble Attr^{k_0} décrit tous les sommets gagnants pour A. Le jeu décrit par G est gagnant pour A si $v_0 \in \text{Attr}^{k_0}$.

Ce calcul peut également servir à définir la (les) stratégie(s) sans mémoire gagnante(s) pour A. Ainsi, au pas $k + 1$ on définit $f(v) = w$ avec $(v, w) \in E$ et $w \in \text{Attr}^k$ si $f(v)$ n'est pas encore défini. On choisira parmi les stratégies possibles celle où $f(v_0)$ est définie et qui ne comporte pas de cycles sans un sommet de F.

5 Entrées et sorties du projet

Graphe de positions Dans une première étape, on vous demande de construire le graphe du jeu ChocoBar à partir des dimensions (hauteur M et largeur N) de la barre.

Chaque sommet de ce graphe $v \in V$ est étiqueté par A ou B pour indiquer à qui appartient ce sommet et par l'état de la barre de chocolat. La position initiale $v_0 \in V$ du jeu ainsi que les positions gagnantes de $F \subset V$ sont indiquées explicitement.

Pour tester vos algorithmes, on vous **demande** d'utiliser les formats vus en TP :

[2] Cette défintion correspond à une stratégie dite "sans mémoire". Il existe des stratégies plus générales qui dépendent du passé mais elle ne sont pas utiles pour ce projet.

– Dot pour l'affichage du graphe des positions ainsi construit,
– le format textuel pour l'affichage et la lecture des graphes de positions.

Pour la représentation du graphe *durant l'algorithme*, il faut utiliser une représentation adaptée, qui minimise la complexité.

Stratégie Une stratégie sans mémoire pour A est un tableau de taille $|V_A|$, représentant une fonction $f : V_A \to V_B$.

Pour tester vos algorithmes, on vous **demande** de décrire les statégies en entrée par un fichier texte ayant le format suivant :

– la première ligne contient un entier qui indique le nombre de lignes suivant cette ligne ; il doit être égal au nombre d'états correspondant au joueur pour lequel la stratégie est spécifiée.
– les lignes suivantes contiennent deux entiers séparés par un blanc ; le premier entier est le numéro de l'état du joueur propriétaire de la stratégie et le deuxième est le numéro de l'état vers lequel ce joueur doit jouer.

Par exemple, une stratégie pour un joueur B dans un graphe avec 4 états $\{0, 1, 2, 3\}$ dont les états 1 et 2 appartiennent à ce joueur serait donnée comme suit :

```
2
1 0
2 0
```

pour spécifier la stratégie suivante $f_B : \{1, 2\} \to \{0, 3\}$, $f_B(1) = 0$, $f_B(2) = 0$.

Pour lire un tel fichier, vous pouvez utiliser l'API `StreamTokenizer` comme dans la classe `DotGraph` fournie.

Partie Une partie est un chemin de G (liste de sommets) issu du sommet v_0.

Elle sera affichée comme une séquence dont chaque élément est un numéro d'état, la configuration de la barre et le joueur qui a joué.

6 Travail à réaliser

Le travail est à faire en groupe de DEUX étudiants.

On vous demande d'écrire en pseudo-langage, puis programmer en Java les algorithmes suivants :

1. Un algorithme `genChoco` qui prend en entrée la dimension de la barre de chocolat et génère le graphe de positions associé.
2. Un algorithme `partie` qui prend en entrée un graphe, une stratégie sans mémoire pour A et une stratégie sans mémoire pour B et indique le perdant, c'est-à-dire celui qui meurt empoisonné.
3. Un algorithme `strategie` qui prend en entrée un graphe et une stratégie pour A et indique si la stratégie est gagnante ou non pour A, c'est-à-dire que A force B à manger le carré empoisonné.
4. Un algorithme `strategieA` qui prend en entrée un graphe de jeu et calcule une stratégie gagnante pour A.

Pour chacun des algorithmes ci-dessus vous devez fournir :

– un fichier `doc.txt` qui spécifie l'algorithme en pseudo-langage (avec son auteur), la représentation du graphe utilisée et sa complexité,
– les sources Java du programme (avec son auteur), et
– un répertoire `tst` qui contient au minimum trois tests *intéressants* fait pour votre algorithme ainsi que les résultats obtenus.

Chaque élément ci-dessus sera noté de façon équivalente, un programme qui marche mais sans documentation, ni fichiers de test sera noté à 1/3 de la note finale.

7 Remise du projet

Le projet sera envoyé aux correcteurs en utilisant le programme `/ens/sighirea/Algo2/remise_projet` au plus tard le vendredi 17 décembre 2004 à 21h. Un fichier d'explication `remise_projet.txt` se trouve dans le même répertoire.

Ce qui doit être envoyé est un fichier d'archive compressée Unix `<login>.tar.gz` contenant les répertoires correspondant aux algorithmes ci-dessus. Pour créer ce fichier, dans le répertoire où se trouve votre projet (avec les sous-répertoires ci-dessus), taper :

```
> tar cvf <login>.tar genChoco partie strategie strategieA
> gzip <login>.tar
```