République Algérienne Démocratique et Populaire
الجمهوريـــة الجـزائريـة الديمـقراطيـة الشـعبية
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
وزارة التـعليم العـالي و البـحث العلـمي

المدرسة الوطنية العليا للإعلام الآلي
(المعهد الوطني للتكوين في الإعلام الآلي سابقا)
Ecole nationale Supérieure d'Informatique
ex. INI (Institut National de formation en Informatique)

Rapport TP5 -TPGO-

Tic-Tac-Toe

Réaliser par :
LAKHAL Mohamed Ilyes
MENACER Mohamed Amine

Proposer par:
Mr: HIDOUCI Walid Khaled

Groupe: B1 'SIQ'

Binôme: 02

2013/2014

Introduction

Les algorithmes élaborés pour résoudre des problèmes concernant les objets de la théorie de graphe ont de nombreuses applications dans tous les domaines liés à la notion de réseau (réseau social, réseau informatique, télécommunications, etc.) et dans bien d'autres domaines, par exemple les jeux.

Dans la théorie des jeux, de nombreux algorithme sont exploités afin de modéliser et analyser les situations optimales dans une configuration donnée, de déterminer une stratégie optimale pour chacun des joueurs, de prédire l'équilibre du jeu et de trouver comment aboutir à une situation optimale.

Ce TP a pour but de découvrir l'un des algorithmes utilisés dans la théorie des jeux, soit l'algorithme de MinMax qui est très utilisé dans les jeux à deux joueurs à somme nulle (et à information complète), il consiste à minimiser la perte maximum (c'est-à-dire dans le pire des cas).

I. Situation du problème

L'objectif principal du TP est de réaliser le jeu Tic-Tac-Toe tout en respectant les contraintes suivantes :

- La taille de la grille est supérieure ou égal à 4.
- Un alignement est compté lorsque le joueur aligne 4 symboles (par ligne, colonne ou diagonale).
- L'intersection entre les alignements est vide.
- Un joueur gagne lorsqu'il réalise « k » alignements ($k \geq 1$).

Juste pour rappeler, Le Tic-tac-toe est un jeu de réflexion se pratiquant à deux joueurs au tour par tour et dont le but est de créer le premier un alignement.

Tic-Tac-Toe est basé sur l'algorithme de MinMax et sur une fonction d'estimation qu'on détaillera dans la section suivante.

II. Solution

Notre solution est basée sur l'utilisation de l'algorithme MinMax et sa variante MinMax Alpha-béta.

1. MinMax Alpha-béta

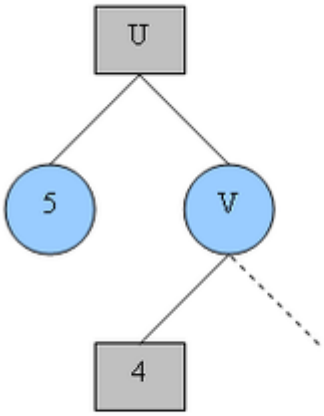
On prend α et β appartenant au domaine d'arrivée de la fonction d'évaluation tel que $\alpha < \beta$. On définit la fonction AlphaBeta ainsi :

- AlphaBeta(P, α, β) = g(P) si P est une feuille de l'arbre et g la fonction d'évaluation du nœud

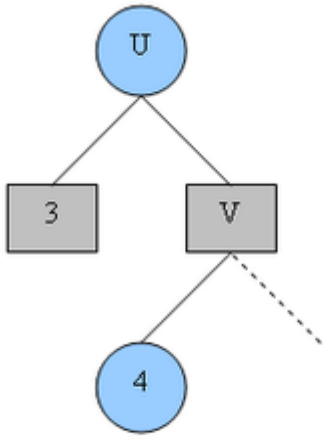
- AlphaBeta(P, α, β) = min(β, max(-AlphaBeta(, -β, -α))) où les sont les fils du nœud P

On appelle fenêtre αβ le couple (α, β) où α et β sont les deux paramètres d'appel de la fonction. Les nœuds élagués sont ceux qui seraient appelés avec une fenêtre tel que α ≥ β. Il existe 3 types de nœuds ne pouvant donc pas être élagués :

- Nœud de type 1 : fenêtre d'appel : (-∞, +∞)
- Nœud de type 2 : fenêtre d'appel : (-∞, β) avec β ≠ +∞
- Nœud de type 3 : fenêtre d'appel : (α, +∞) avec α ≠ -∞



Coupure Alpha



Coupure Beta

Le schéma ci-dessus présente les deux types de coupures possibles. Les nœuds Min sont représentés par un rond bleu et les nœuds Max par un carré gris. Rappel : les nœuds Min prennent la valeur minimum de leurs fils (et respectivement maximum pour les nœuds Max).

Coupure Alpha : le premier fils du nœud Min V vaut 4 donc V vaudra au plus 4. Le nœud Max U prendra donc la valeur 5 (maximum entre 5 et une valeur inférieure ou égale à 4).

Coupure Beta : le premier fils du nœud Max V vaut 4 donc V vaudra au minimum 4. Le nœud Min U prendra donc la valeur 3 (minimum entre 3 et une valeur supérieure ou égale à 4).

```
fonction ALPHABETA(P, alpha, beta)
 /* alpha est toujours inférieur à beta */
   si P est une feuille alors
       retourner la valeur de P
   sinon
       si P est un nœud Min alors
           Val = infini
           pour tout fils Pi de P faire
               Val = Min(Val, ALPHABETA(Pi, alpha, beta))
               si alpha ≥ Val alors  /* coupure alpha */
```

```
                retourner Val
            beta = Min(beta, Val)
        finpour
    sinon
        Val = -infini
        pour tout fils Pi de P faire
            Val = Max(Val, ALPHABETA(Pi, alpha, beta))
            si Val ≥ beta alors /* coupure beta */
                retourner Val
            alpha = Max(alpha, Val)
        finpour
    finsi
  retourner Val
  finsi
fin
```

2. Estimation

On représente la table du jeu par une matrice, soit x le score obtenu par les X et o le score obtenu par les O, la fonction ci-dessous retourne un entier pour une configuration donnée soit par ligne ou colonne ou diagonal ou diagonal inverse, donc pour obtenir un score global on doit faire la somme de toutes les configurations possibles.

```
int score = 0;
        int x = 0, o = 0;
        for (int i = 0; i < COLS; i++) {
            si (X){
                si (x == 0) {
                    score += Math.pow(10, x);
                } sinon {
                    score -= Math.pow(10, x - 1);
                    score += Math.pow(10, x);
                }
                x++;
                o = 0;
            }
            si (O) {
                si (o == 0) {
                    score -= Math.pow(10, o);
                } sinon {
                    score += Math.pow(10, o - 1);
                    score -= Math.pow(10, o);
                }
                o++;
                x = 0;
            }
```