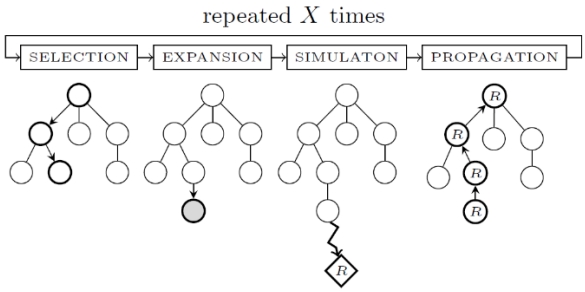
IA et Jeux – TP N°3 : MCTS

Objectif du TP

L'objectif de ce TP est de mettre en œuvre l'utilisation de l'algorithme MCTS pour aider à choisir le prochain mouvement de Ms. PacMan.



Algorithm 2 The UCT algorithm.

function UCTSEARCH(s_0)
  create root node v_0 with state s_0
  while within computational budget **do**
    $v_l \leftarrow$ TREEPOLICY(v_0)
    $\Delta \leftarrow$ DEFAULTPOLICY($s(v_l)$)
    BACKUP(v_l, Δ)
  return a(BESTCHILD($v_0, 0$))

function TREEPOLICY(v)
  while v is nonterminal **do**
    if v not fully expanded **then**
      return EXPAND(v)
    else
      $v \leftarrow$ BESTCHILD(v, Cp)
  return v

function EXPAND(v)
  choose $a \in$ untried actions from $A(s(v))$
  add a new child v' to v
    with $s(v') = f(s(v), a)$
    and $a(v') = a$
  return v'

function BESTCHILD(v, c)
  return $\underset{v' \in \text{children of } v}{\arg\max} \frac{Q(v')}{N(v')} + c\sqrt{\frac{2 \ln N(v)}{N(v')}}$

function DEFAULTPOLICY(s)
  while s is non-terminal **do**
    choose $a \in A(s)$ uniformly at random
    $s \leftarrow f(s, a)$
  return reward for state s

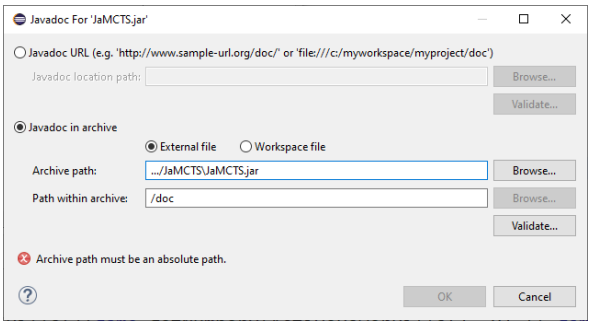
function BACKUP(v, Δ)
  while v is not null **do**
    $N(v) \leftarrow N(v) + 1$
    $Q(v) \leftarrow Q(v) + \Delta(v, p)$
    $v \leftarrow$ parent of v

Rédiger un Compte-Rendu

1. A la fin de ce TP, vous devrez rédiger un compte-rendu. Pour savoir ce qu'il doit/devrait contenir, lisez *attentivement* ce texte https://guillaume.piolle.fr/doc/tp.pdf.

Utilisation de la librairie JaMCTS

2. Heureusement pour vous, l'algorithme MCTS est déjà mis en œuvre en Java dans la librairie JaMCTS qui vous est fournie. Intégrez le fichier jar à votre projet Eclipse en modifiant le `Java Build Path` de votre projet.



3. Vous devriez également faire pointer la documentation de cette librairie vers le dossier `/doc` de cette même librairie. Utilisez le bouton `validate` pour vous assurer que vous pointer bien vers une documentation.

Monte Carlo Tree Search sur Tic-Tac-Toe

4. Vous trouverez ci-contre le programme principal de la précédente librairie. Exécuter la librairie pour vérifier qu'elle fonctionne correctement. Saisissez-le et modifiez-le pour utiliser d'autres stratégies de recherche, d'autre condition d'arrêt de recherche…

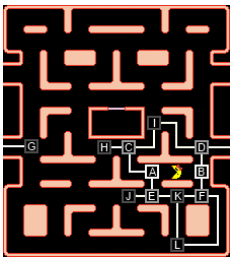
```
public class TTTMain {
    public static void main(String[] args) {
        State morpion = new TTTState(Player.p1) ;
        morpion = morpion.takeAction(new TTTAction(Player.p1, 0, 0)) ;
        morpion = morpion.takeAction(new TTTAction(Player.p2, 1, 1)) ;
        morpion = morpion.takeAction(new TTTAction(Player.p1, 1, 0)) ;
        morpion = morpion.takeAction(new TTTAction(Player.p2, 2, 0)) ;
        Random rand = new Random(1024) ;
        MCTS<StateInfo> mcts = new MCTS<StateInfo>(rand, new StopAfterMaxIterations(100),
                                new UpperConfidenceBoundPolicy(rand,MCTS.DEFAULT_EXPLORATION_CONST),
                                new RandomPolicy(rand),
                                StateInfo.class,
                                DEBUG.NO) ;
        System.out.println("Monte Carlo Tree Search Library v0.2\n");
        System.out.println("Test on Tic-Tac-Toe initial state:\n"+morpion);
        System.out.println("Next player to play: "+morpion.getCurrentPlayer());
        Action action = mcts.search(morpion) ;
        System.out.println("MCTS action: "+action);
        mcts.tree.showInFrame();
    }
```

MCTS simple pour Ms. PacMan

5. En vous inspirant de ce qui a été fait pour Tic Tac Toe, créez un nouveau contrôleur pour Pacman basé sur MCTS et nommez-le `MCTS<VotreNom>.java` (exemple : `MCTSPallez.java`). Le constructeur aura un seul paramètre qui est un contrôleur de fantômes qui n'est pas contenu dans la variable `game`. Par exemple, l'`executor` du framework PacMan pourra uniquement utiliser l'instruction suivante : `new MCTSPallez(new RandomGhosts())`.
6. Il est évident que vous aurez besoin de créer de nouvelles classes spécifiques aux actions et aux états de ce jeu. Pour cela, créer une nouvelle classe `PMSimpleAction` qui hérite de `Action`. Posez vous la question de ce qu'est une action pour PacMan ? Surchargez les méthodes `toString`, `equals`, et `hashCode`.
7. Créez une autre classe `PMSimpleState` qui hérite de `MCTS.Base.State`. Qu'est-ce qu'un état du jeu dans PacMan ? Définissez le constructeur de cette classe. *Indice* : vous pourriez avoir besoin du contrôleur des fantômes pour créer de nouveau états au fur et à mesure que vous avancez dans la simulation du jeu.
8. Définissez les méthodes abstraites héritées de classe mère `State`. Pensez aussi à surcharger la méthode `toString` qui est utilisée pour afficher l'arbre.
9. Testez votre programme avec différents contrôleurs de fantômes (aléatoires, agressifs…) et différentes stratégies de recherche pour MCTS. Faites des comparaisons statistiques…

MCTS Amélioré pour Ms. PacMan

10. Plutôt que d'enregistrer un état pour chaque position de pacman, (Yannakakis et al. 2018) suggère de stocker dans l'arbre de recherche que les états correspondants aux nœuds de jonctions de pacman (cf. TP N°1 – question 19).



Sources pour ce TP :
gameaibook.org

Yannakakis, Georgios N., and Julian Togelius. *Artificial Intelligence and Games*. Springer, 2018.