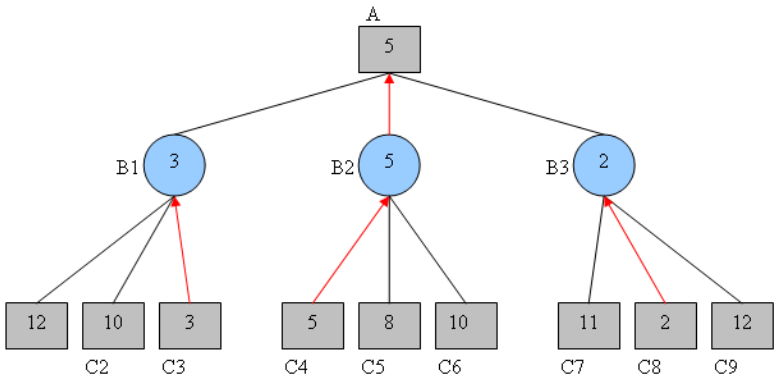
Institut national des sciences appliquées de Rouen

INSA de Rouen

Projet Info GM4 - Vague 1

Algorithme Min-Max



Auteurs :
Thibaut André-Gallis
thibaut.andregallis@insa-rouen.fr
Kévin Gatel
kevin.gatel@insa- rouen.fr
Chloé Mechelinck
chloe.mechelinck@insa-rouen.fr

Enseignants :
Nathalie Chaignaud
nathalie.chaignaud@insa-rouen.fr

07 Décembre 2021

Table des matières

Introduction

But du projet :

Afin de mettre en $\frac{1}{2}$uvre les compétences acquises dans le codage en langage objet, nous avons choisi le projet consistant à implémenter l'algorithme Min-Max dans le cas d'un jeu de dames.

Nous avons donc réaliser des recherches pour comprendre ces concepts et pouvoir les implémenter. Cela nous a aussi permis de travailler en équipe avec des personnes différentes de nos précédents projets.

1. L'algorithme

1.1 Explication de l'algorithme du Minimax

L'algorithme minimax ou minmax est un algorithme s'appliquant dans le cas d'un jeu (donc dans le cadre de la théorie des jeu) à deux joueurs lorsqu'il s'agit d'un jeu à somme nulle. Son objectif est de minimiser la perte maximum. Un jeu est à somme nulle si la somme des gains et des pertes de tous les joueurs est égale à 0, c'est à dire la perte de l'un est le gain de l'autre. Ce type de jeu répond à plusieurs caractéristiques, démontrées notamment par le théorème du minimax de Von Neumann, dès 1926 (présence de configurations d'équilibre, existance de l'algorithme...).

Le principe est globalement assez simple. L'ordinateur passe en revue tous les possibilités pour chaque pièce sur un nombre limité de coup, créant ainsi un arbre des possibilités (dont nous parlerons d'avantage dans la 4ème partie dans nos exemples). Ensuite chaque noeud se voit affecter une valeur en fonction des bénéfices du joueur et de l'adversaire. Le choix retenu sera la branche partant d'une feuille de cette arbre jusqu'à la racine, indiquant ainsi le coup qui doit être réalisé.

Un problème de mémoire se pose alors car chaque pièce peut se déplacer à gauche ou à droite (sauf si une pièce la bloque ou si le plateau de jeu ne continue pas) et cela sur un seul coup. Si on réalise plusieurs coups (ce qui est nécessaire), l'arbre devient rapidement très vaste. En pratique, on explore souvent, lorsque l'algorithme est optimisé une partie seulement de cet arbre à l'aide de méthodes dites d'élagage.

1.2 Explication de la méthode alpha-béta et de l'élagage

L'élagage alpha-bêta (ou ...) est une méthode grâce à laquelle on va pouvoir réduire la taille de l'arbre en enlevant certaines branches à l'aide de conditions choisies. Ainsi, on réduit le nombre de noeuds évalués et donc le temps de calcul de la branche "à choisir". Il s'agit d'une optimisation du minimax sans perdre des informations.

Cet élagage repose sur le fait qu'il n'est pas nécessaire d'examiner les sous arbres dont la configuration et le résultat ne permettra pas une amélioration du gain. On évalue pas les noeuds (et leur sous-arbre) dont la qualité (le gain, l'intérêt) sera inférieur à un noeud déjà évalué. Pour cela on va d'ailleurs travailler sur l'arbre dans un sens fixé, ici de gauche à droite.

2. Le jeu de Dame

2.1 Implémentation du jeu de Dames

Nous avons tout d'abord fixé les règles que nous allions respecter, les variantes sur un jeu aussi populaire que les dames étant nombreuses.

Les pions ne peuvent pas se déplacer en arrière ni manger en arrière.

Il est obligatoire pour un pion de manger lorsqu'il peut.

Les dames peuvent manger en arrière et se déplacer dans les quatres diagonales sur l'espace de plusieurs cases.

Nous avons réalisé notre code en java, langage que nous maitrisions d'avantage. Nous allons d'abord donc brièvement présenter les classes qui concerne le jeu de dame principalement :

2.1.1 Case

La classe Case représente tout simplement une case du plateau du jeu de Dame. Elle possède donc des coordonnées, une couleur, une pièce qui peut être vide et plusieurs booléens qui ont un rôle dans l'implémentation et la partie graphique du jeu.

2.1.2 Coordonnées

La classe Coordonnées possède deux entiers x,y qui représentent juste des coordonnées sur le plateau. Nous avons crée cette classe pour ne pas avoir à transporter deux variables à chaque fois.

2.1.3 Couleur

Cette classe sert juste à différencier la couleur d'une pièce ou d'une case

2.1.4 Damier

La classe Damier est la classe principale du jeu de Dame. Elle représente le plateau du jeu mais pas seulement. Il contient également le tableau des pièces du jeu, le tableau des cases du jeu, la taille du plateau ainsi que plusieurs booléen qui jouent un rôle dans les règles du jeu et son déroulement. Elle possède également la procédure d'affichage principale du jeu.

2.1.5 Joueur

La classe Joueur représente un joueur du jeu évidemment. Elle se décline en deux classes qui héritent de joueur ordi et humain. Elles ne servent qu'à différencier si le joueur est un ordinateur ou un utilisateur sur sa machine. On retrouve les procédure AJoue qui implémente un coup dans le jeu, ainsi que APerdu qui retourne si le joueur a perdu ou gagné la partie.

2.1.6 Lanceur

Cette classe constitue le Main de notre programme, il est possible d'y modifier certains paramètre de la partie manuellement, sinon nous utilisons une autre classe que nous allons voir pour laisser le joueur décider des règles.

2.1.7 Menu

La classe menu permet de laisser l'utilisateur choisir ses règles pour la partie. Elle consiste simplement en un menu à choix multiples que l'on récupère et que l'on transmet au main pour lancer la partie.

2.1.8 Piece

La classe pièce correspond à une pièce du jeu elle se décompose en deux classes qui héritent de celle-ci : Pion et Reine qui sont les deux différents types de pièces qu'on retrouve dans le jeu de dame. Dans le fonctionnement de notre jeu c'est au niveau de la pièce que l'on vérifie si elle peut être mangée, si elle peut manger une autre pièce ainsi qu'afficher les déplacements possibles de la pièce.

2.1.9 Souris

La classe souris est sert à récupérer les actions de la souris pour permettre au joueur d'utiliser une souris pour jouer au lieu du clavier. Elle permet de récupérer les coordonnées de la case sur laquelle le joueur clique.

2.1.10 TableauPiece

La classe TableauPiece gère la liste des pièces du jeu d'un camp comme de l'autre.

2.2 Implémentation de la partie Algorithme Min Max et ses améliorations

Nous allons maintenant nous intéresser d'avantage à la partie IA du projet. Nous allons détailler les classes du programme qui joue un rôle dans l'IA basée sur un algorithme min-max ainsi que certaines procédures importantes.

2.2.1 Arbre

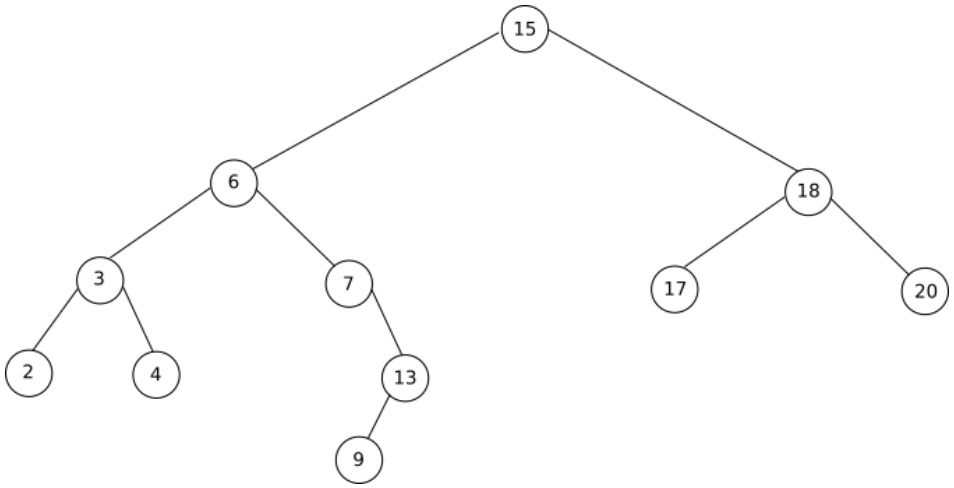


FIGURE 2.1 – Exemple d'un arbre quelconque

La classe Arbre représente l'arbre sur lequel se base notre algorithme min-max. D'un point de vue informatique notre classe arbre est en fait la racine de l'arbre et on accède au reste de l'arbre grâce à la liste de successeurs qu'elle possède. C'est à dire que la classe arbre est le nœud tout en haut, nous allons maintenant définir ce qu'est un nœud dans notre cas de figure.

2.2.2 NoeudDame

La classe NoeudDame représente un nœud pour le jeu de dame dans le cadre de l'algorithme min-max. Elle implémente la classe abstraite Nœud. Nous l'avons donc défini comme un damier du jeu de dame. Un NoeudDame possède également une liste de NoeudDame qui lui succède et une valeur de profondeur permettant de savoir à quel profondeur de l'arbre il se situe. Cette classe possède également une des procédures les plus importante pour notre IA. Nous explorer l'Heuristique plus en détail car elle est essentielle pour le fonctionnement de l'IA.

2.2.3 Coup

La classe Coup représente un coup pour une pièce du jeu. Elle possède deux pièces celle au départ du coup et celle à la fin. Évidemment ce qui va changer entre deux sont les coordonnées de la pièce.

2.2.4 Resultat_minMax

Cette classe a été crée pour exploiter le résultat retourné par notre méthode min-max. Elle n'a pas de signification particulière elle est juste une manière pratique de transporter les données dans ce cas. Pour résumer c'est une liste de coups à effectuer que nous renvoie notre algorithme min-max.

2.3 Application de la méthode à des exemples

Pour mieux comprendre cette méthode et les différents cas possibles, voici plusieurs exemples (simplifier car on a choisit de représenter deux pions sur plusieurs coups en ayant attribué des valeurs aux noeuds).

3. UML

3.1 diagramme de cas d'utilisation

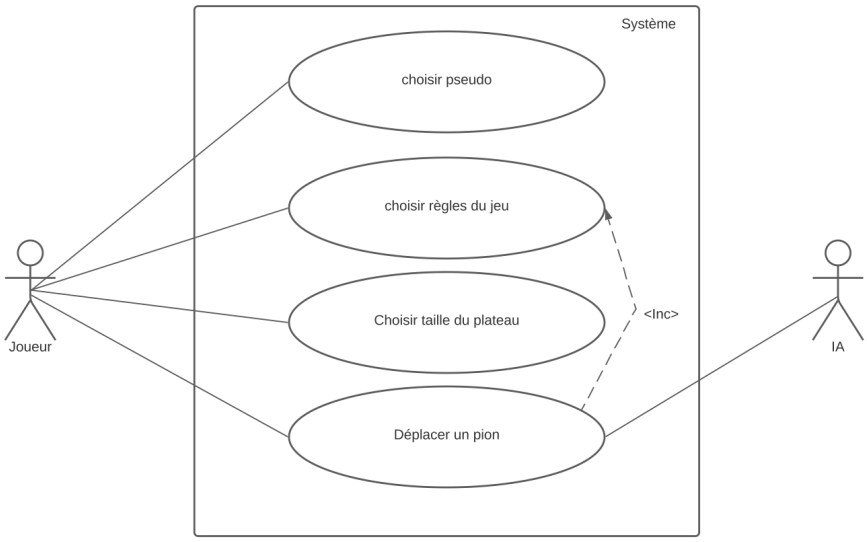


FIGURE 3.1 – Diagramme use case du jeu de dame

L'utilisateur peut ainsi choisir de jouer à 2, de jouer contre l'ordinateur ou afficher les règles, stockées dans un fichier texte extérieur.

3.2 diagramme de séquences

Nous avons réalisé plusieurs diagrammes de séquences pour détailler la partie algorithme Min-max. Les voici :

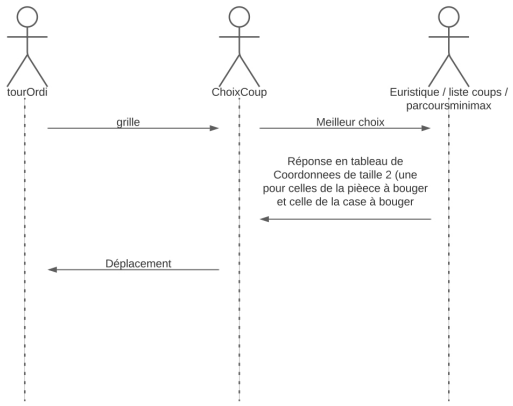


FIGURE 3.2 – Diagramme de séquence choix coup ordi

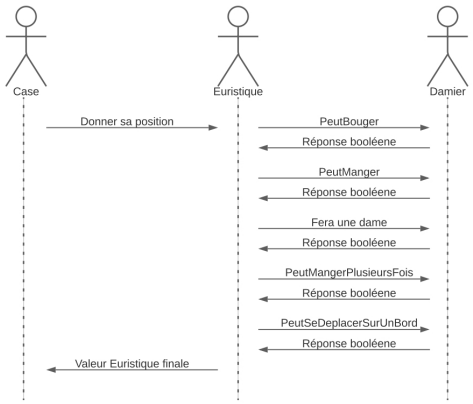


FIGURE 3.3 – Diagramme de séquence heuristique

3.3 diagramme de classes

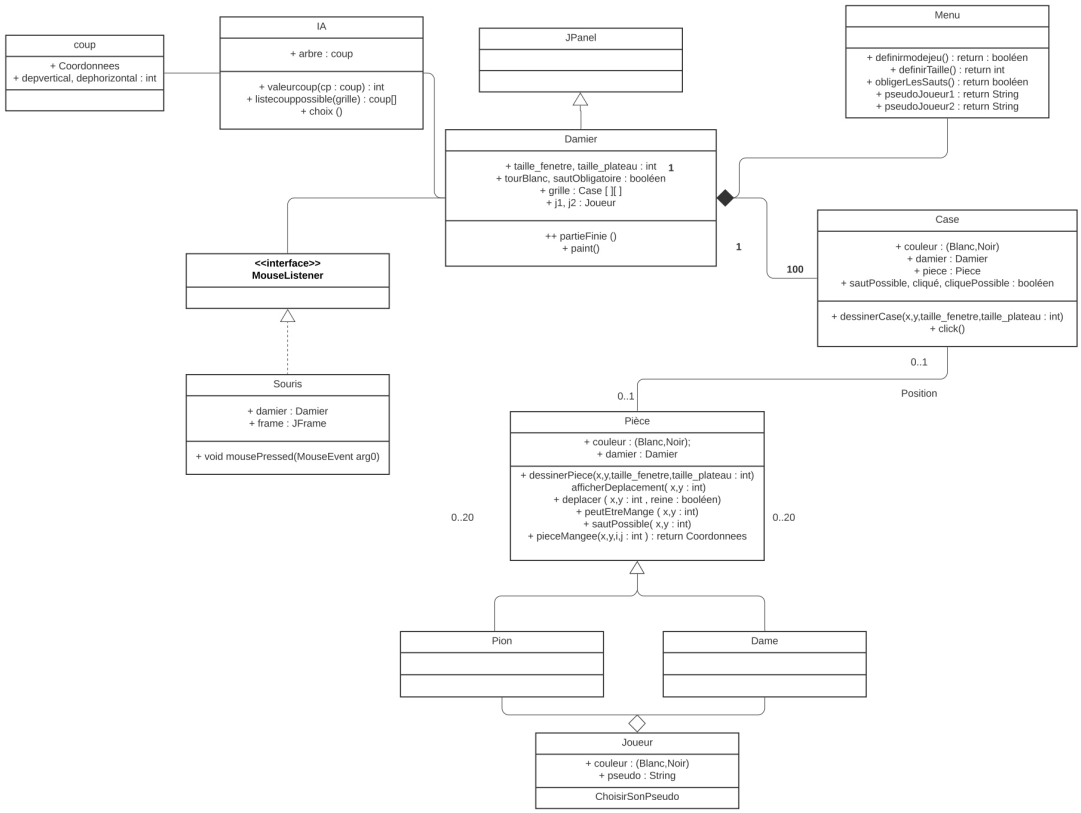


FIGURE 3.4 – Diagramme des classes

Voici le diagramme de classe de notre projet. Celui-ci est susceptible d'évoluer jusqu'à la fin de notre code, il se sépare en deux parties : la première pour l'implémentation du jeu de Dames la seconde pour celle de l'algorithme Minimax.

Un certain nombre de fonction seront détaillées dans les parties suivantes.

Conclusion :

Cette première partie nous a permis de nous renseigner sur la théorie des jeux, les spécificité du jeu de Dames (les différents types de positions des pions...). Nous avons aussi commencé à coder après avoir réalisé nos diagrammes en essayant de rendre notre code le plus portable possible, tout cela en travaillant en équipe à trois. Nous allons poursuivre le codage de notre programme, tout en réalisant certaines parties en pseudo-code d'abord car la fonction euristique est très importante mais aussi assez importante.

Pour aller plus loin :

Comme dit précédemment nous ne sommes qu'au milieu de notre projet et nous n'avons donc pas fini les fonctions de la partie implémentation du mininmax.

Nous aimerions réaliser la fonction élagage, ainsi que plusieurs euristiques que nous comparerions pour obtenir la meilleur (il s'agit de la difficulté à laquelle sont confrontées les personnes ayant travaillé sur cet algorithme etc : choisir la bonne euristique).

Si le temps qu'il nous reste nous le permet nous aimerions pouvoir afficher à la demande du joueur, l'arbre des possiblités de sa composition actuelle et la branche choisie.

Nous voulons aussi réaliser un menu pour notre jeu avant le début de la partie où l'on peut choisir de jouer à deux, seul contre l'ordinateur ou bien d'afficher les règles.

Annexe