POLYCHESS

Gestion de projets : Proj531

BOURABI Kaoutar

RANDRIAMAHEFA Christelle

HASSANI Yawoumihani

NDRI NDRI Kouadio Ange Richard J.

ABDOU MAHAMOUDOU Zakari Yaou.

Table des matières

I. Présentation du projet

1. Conception d'un jeu d'échecs sur python

Dans le cadre des enseignements de Gestion de projets (Proj531), nous avons dû concevoir un jeu d'échecs sur python. Ce projet devait nous permettre d'acquérir des compétences nécessaires à tout futur ingénieur tel que le travail en équipe ou encore accroître notre efficacité en termes d'organisation.

2. Règles

Nous avons rapporté ici un bref rappel général des règles du jeu des échecs.

Au début de la partie, chaque joueur dispose de 16 pièces : un **Roi**, une **Dame**, deux **Tours**, deux **Fous**, deux **Cavaliers** et huit **pions**.

Sur un échiquier, il y a 8 **colonnes** (de a à h) et 8 **rangées** (de 1 à 8).

Les deux joueurs jouent à tour de rôle en déplaçant une seule de leurs pièces. Si une pièce se déplace sur une case occupée par une pièce adverse, celle-ci est prise et enlevée de l'échiquier. Une pièce ne peut pas se placer sur une case occupée par une pièce de son propre camp. Seul le Cavalier peut sauter au-dessus des autres pièces.

3. Objectifs

L'objectif principal était de concevoir un programme pouvant dérouler une partie de jeu en prenant en compte toutes les règles de jeu et l'énoncé de notre sujet initial.

Notre programme devait être apte à dérouler une partie entre deux joueurs humains, une partie entre une machine et un joueur humain mais aussi une partie entre deux machines.

II. Résultats attendus

1. Jeu

Le programme final devrait nous permettre de faire une partie de jeu sur Python.

Pour cela, notre jeu doit être capable d'initialiser l'échiquier dans un premier temps. Pour chacun des joueurs, il doit pouvoir proposer des positions statistiquement gagnantes

grâce au livre Polyglot que l'on peut lire à l'aide de fonctions provenant de la bibliothèque « chess » de python.

Dans le cas où la position n'est pas dans « chess.polyglot », le programme proposera des positions possibles selon l'état actuel de l'échiquier avec leur pondération afin de laisser le choix à l'utilisateur dans le cas d'une partie entre machine et joueur humain ou entre deux joueurs humains.

Dans le cas général, le jeu présentera ou jouera le meilleur coup (lorsque c'est au tour de la machine de jouer) avec l'algorithme Min-Max (ou Alpha-Beta).

Ainsi se déroule une partie de jeu que l'on continue tant qu'il n'y a pas « échec et mat » sinon match nul.

Enfin, cette partie devra être sauvegardée au format PGN.

III. Gestion organisationnelle du projet

1. Organisation et outils

De prime abord, nous avons fait des choix d'ordre organisationnels en termes d'outils pour mener à bien notre projet.

Bien qu'imposé, l'utilisation de GitHub pour la gestion du code est choix plus qu'intéressant. En effet, il nous permet de travailler en local puis de faire basculer notre travail en distant afin que nos codes et/ ou modifications soient visibles pour chaque membre du groupe.

Nous avons décidé de créer une branche pour chacun d'entre nous afin de déposer notre travail au fur et à mesure de l'avancement du projet. La branche principale (master branch) est celle qui accueille le code assemblé, bien agencé et fonctionnel. Ainsi nous avons pu tester et forker une version complète et sans bogue à chaque étape d'avancement.

Pour réaliser ce projet, un outil de communication s'est avéré crucial. Nous avons fait le choix de créer un groupe Teams ainsi qu'un groupe WhatsApp.

Teams nous a permis de communiquer lors des séances communes de travail ou lors des meetings afin de se tenir informé de nos avancées ou difficultés. Le partage d'écran que propose cette plateforme se révèle être précieux. C'est aussi un outil très utile lorsque nous sommes plusieurs assignés à la même tâche.

WhatsApp a démontré sa praticité durant des moments d'interrogation nécessitant des réponses immédiates ou encore pour peaufiner certains détails comme fixer les horaires des meetings.

Afin de s'assurer de l'avancement d'un projet, il faut une bonne planification, c'est pourquoi nous avons choisi d'utiliser Trello. C'est un bon outil de planification

qui est complet et très intuitif. Ainsi nous avons pu voir sur quelle tâche chacun d'entre nous était assigné mais aussi avoir une description un peu plus détaillée de la tâche que nous pouvons modifier. Nous avons la possibilité d'ajouter des checklists pour décomposer la tâche que l'on a à effectuer. De plus, cet outil nous a permis d'avoir conscience des délais de temps imposés pour effectuer certaines tâches.

2. Organisation et tâches

Notre équipe est constitué de cinq membres dont un chef de projet. A chaque séance de travail, un programme était établi par le chef d'équipe selon les tâches, les avancées, les accords déjà vus en amont. Ainsi, chacune de nos séances étaient organisées de la manière qui suit.

Tout d'abord, une partie introductive où chacun d'entre nous explique ce qu'il a fait, ce qu'il lui reste à faire et les difficultés rencontrées. Ainsi, nous pouvions résoudre des problèmes en proposant des solutions dans le cas où des difficultés avaient été rencontré.

Ensuite, selon les conclusions de la partie introductive et la planification, chacun de nous pouvait alors commencer le travail en autonomie. Etant tous connectés sur Teams, si la moindre difficulté nous empêchait d'avancer, la personne la plus à l'aise de pouvoir apporter son aide a pu le faire sans problème.

En fin de séance, il s'agissait de conclure par une mise en commun des résultats obtenus lors de la séance de travail. Tout comme pour la partie introductive, chacun de nous expliquait où il en était en termes d'avancement et les difficultés rencontrées. Ainsi, des directives furent attribuées à chacun. Certaines de ces directives furent modifiées dans le cas de certaines tâches où par exemple nous n'avions pas compris le sens de ce que nous devions réaliser etc.

Durant chacune de ses séances, nous avons toujours fait appel au moins une fois à l'un des deux professeurs soit pour des questions d'ordre technique ou d'incompréhension mais aussi pour avoir un avis sur les directives à prendre comme par exemple l'affectation de plusieurs personnes aux tâches estimées comme étant les plus lourdes.

IV. Gestion technique du projet

Avant de commencer à coder, il nous a fallu analyser les différentes solutions possibles pour réaliser notre programme.

Dans un premier temps, nous avions décidé de créer nous même les différentes composantes d'un jeu d'échecs. Nous avions donc créé différentes classes : Joueur,

Pièce, Echiquier, Jeu. Chacune de ses classes devaient regrouper des attributs par exemple une pièce est caractérisé par son nom, sa pondération, sa couleur etc.

Seulement lors de la seconde séance de travail, l'un des professeurs nous a confirmé qu'il fallait utiliser la bibliothèque « chess » déjà intégré dans python pour réaliser le projet. Ainsi nous sommes repartis de zéro mais avec une solution de résolution plus simple étant donné que toutes les fonctions relatives au jeu d'échecs sont intégrées dans « chess ». Néanmoins, une grande phase de documentation a été nécessaire pour maîtriser les techniques de manipulation de cette bibliothèque.

1. Modules

Nous avons fait le choix d'une programmation modulaire et architectural.

En effet, notre programme est composé de 4 modules qui correspondent aux quatre fichiers :

- ***Main1.py*** qui contient la méthode principale permettant de dérouler une partie entre deux joueurs humains.
- ***Main2.py*** qui contient la méthode principale permettant de dérouler une partie entre une machine et un joueur humain
- ***Main3.py*** qui contient la méthode principale permettant de dérouler une partie entre un joueur et une machine.
- ***__init__.py*** est constitué de toutes les méthodes appelées dans les fonctions principales.

2. Choix techniques

Afin de mener à bien ce projet, il nous a fallu faire des choix d'ordre technique (paramétrage, erreurs etc…) afin de s'assurer que notre programme est bien fonctionnel.

a. Polyglot – pondération

Polygot(…) est la méthode permettant de récupérer les positions statistiquement gagnantes ainsi que leur pondération selon l'état courant. Cet état correspond à notre échiquier : le « board ».

Pour des questions de praticité, nous avons décidé d'insérer les mouvements et pondérations associées récupérés du livre de « chess.polyglot » dans une liste. Ainsi, cela a pu nous faciliter l'utilisation de ses mouvements par la suite, en manipulant la liste retournée par la fonction ***polyglot (…).***

Nous avons tout de même créer une méthode permettant de retourner les pondérations dans le cas où les positions ne sont plus dans « chess.polyglot » afin de continuer le jeu. En effet, ***ponderation (…)*** prend en compte la valeur associée à chaque pièce présente sur l'échiquier. L'ajout d'une condition dans la boucle a été plus que décisif afin de gérer les pondérations des cases vides qui généraient des erreurs dans notre programme.

b. Algorithme de jeu : Minmax, Alpha-beta

Afin de jouer ou de proposer le meilleur coup, l'utilisation de Minmax et Alpha-beta était obligatoire. Bien que ces algorithmes soient souvent utilisés dans des jeux, nous avons tout de même dû les repenser afin de les adapter à ce que nous souhaitions faire.

De ce fait, certains choix ont été établi : l'ajout d'une profondeur en paramètre qui correspond au « depth » dans ***alphabeta (…)*** et dans ***minmax(…).*** Ainsi, la gestion des données a pu être facilité. Il est important de noter que la complexité de ces algorithmes dans notre cas est n^p avec n : le nombre de mouvements possibles et p : la profondeur. Afin de réduire cette complexité et permettre un gain de temps lors de l'exécution de notre programme, nous avons décidé de fixer la profondeur à 3.

D'autres choix ont été fait afin d'adapter au mieux ces algorithmes à notre programme.

Effectivement, dans ***minmax (…)***, il s'est avéré prudent et crucial de créer une copie du « board » sur laquelle sont effectués les mouvements afin d'éviter par la suite, la réinitialisation du « board » de la partie en cours (à l'issue des mouvements présents dans « chess.polyglot »).

Alphabeta également a dû être adapté à ce que nous souhaitions. Tout d'abord, il est important de remarquer que nous avons décomposé cet algorithme en 3 méthodes : alpha (…), beta (…), alphabeta (…). Ce choix s'est révélé être très utile pour la simplification de la structuration du code mais plus spécialement pour le gain de temps dans la gestion d'erreur dans le code. Effectivement, c'est ce qui nous a permis de détecter des problèmes rapidement comme celui de la génération de mouvements pseudo-légaux. En effet, nous avons pu immédiatement gérer cette erreur de type « assertion error » afin de pallier les problèmes de génération de mouvements non légaux, par exemple.

c. Méthodes principales

Pour les méthodes principales, les principaux choix techniques importants que nous pouvons relever sont les suivants.

L'utilisation d'un compteur pour la gestion des tours, le « try … except » pour gérer les erreurs au niveau de la saisie des déplacements et permettre à l'utilisateur de ressaisir son mouvement si celui-ci n'a pas le bon format.

Notre programme laisse également la possibilité à l'utilisateur de nommer la partie qu'il va jouer et ce nom correspondra au nom du fichier PGN où sera sauvegardé la partie.

V. Bilan du projet

Afin de conclure, ce projet a été une expérience enrichissante pour chacun d'entre nous. Nous avons gagné en compétences relationnelles, organisationnelles en équipe comme en autonomie mais également des compétences techniques.

En effet, notre cohésion d'équipe et nos compétences relationnelles se sont développées simultanément. Effectivement, plus nous étions à l'aise les uns avec les autres, plus la cohésion de groupe grandissait notamment grâce à la communication. Celle-ci a pris une grande place au sein de notre équipe. C'est sur cette forte base de communication que nous avons pu développer nos compétences organisationnelles.

La première des difficultés rencontrées a été le manque de communication des problèmes rencontrées en autonomie au début du projet. Ce qui faisait obstacle à notre organisation. Néanmoins, notre chef d'équipe a su nous rappeler que chaque membre du groupe était une clé pour la réalisation du projet mais que nous étions avant tout une équipe. Autrement dit se soutenir pour avancer ensemble est évident. Cette prise de conscience a dissipé la gêne et nous a permis de pouvoir avancer en respectant les planifications.

L'autre difficulté rencontrée au cours du projet était la différence de niveau technique. En effet, nous n'avions pas tous les mêmes compétences néanmoins nous avons eu la chance d'avoir des compétences complémentaires. C'est ce qui a permis de régler ce problème très rapidement.

Effectivement, l'utilisation de GitHub, de ses commandes ou l'utilisation du Trello ou encore le codage sur python n'était pas nécessairement intuitif pour chacun d'entre nous. Nous avions chacun des compétences différentes : meilleures dans certaines choses et moins bonnes dans d'autres. Mais durant les séances ou les meetings nous avons pu pallier ce problème en communiquant ou en travaillant par paire volontairement afin de résoudre les problèmes rencontrés. C'est ainsi que nous avons tous gagné en compétences techniques en partageant les nôtres. Cependant, il est important de souligner que cela n'a pu être possible que grâce à l'implication de chacun d'entre nous dans les phases de documentation.