Rapport de projet Jarnac

Auteurs

- VÕ Alexis – alexis.vo@universite-paris-saclay.fr, LDD IM 2
- ĐOÀN Thanh Phát – thanh-phat.doan@universite-paris-saclay.fr, LDD IM 2 (portail MI)

Résumé du travail effectué

Pour le projet de fin de semestre d'*Info 111*, nous avons choisi le projet *Jarnac*. Il nous semblait être le plus adapté puisque nous avions déjà une certaine expérience en programmation et ce projet nous permet davantage d'être créatif. Ainsi, nous avons décidé de suivre notre propre découpage du code.

Le projet a débuté par une maquette globale de nos fonctions et de nos fichiers : choix pertinents des noms de variables et de fonctions. Nous avons ensuite déterminé un planning avec des dates limites et une répartition du travail à réaliser. Bien que nous n'avancions pas au même rythme, nous conservions – grâce au planning mis en place – le même tempo. Les délais ont été tenus et cela nous a permis d'avancer sereinement.

Nous avons hésité avant d'implémenter une interface graphique. Finalement, nous y avons renoncé car il nous aurait fallu plus de temps. Nous préférons avoir un code de qualité et pertinent sans interface graphique plutôt qu'une interface graphique avec un mauvais code. Ainsi, cela nous a amené à perfectionner notre programme (ajout de fonctions), d'étudier sa complexité (méthode de recherche dans le dictionnaire) et d'ajouter une Intelligence Artificielle (IA). Nous avons décidé de ne pas supprimer les mots impossibles à former dans les dictionnaires (exemple : le mot « anxieux » possède 2 lettres « x » mais la pioche n'en possède qu'une seul) ; ceci afin d'offrir une plus grande flexibilité dans le développement du jeu (ajout de lettres dans la pioche par exemple) au détriment de place en mémoire (minime), certes.

Pour l'Intelligence Artificielle, nous l'avions mis en œuvre de façon naïve. Plus précisément, pour déterminer quels sont les meilleurs mots possibles à former, compte tenu des mots de la réserve et sur le plateau, notre IA passe en revue un dictionnaire. Cependant, quelques observations cruciales nous ont permis d'effectuer de nombreuses optimisations :

- La réduction de la taille du dictionnaire pour l'IA peut être un excellent moyen d'ajuster sa difficulté et de réduire le nombre de mots que l'IA doit parcourir.

- Lorsqu'un mot est placé sur le plateau, ses lettres peuvent être réarrangées pour former un nouveau mot, mais ne peuvent pas être enlevées. Ainsi, à partir d'un mot initial, nous pouvons créer un ensemble de mots composé des lettres du mot initial, grâce au dictionnaire de l'IA. Cet ensemble est considérablement plus petit que le dictionnaire, ce qui accélère considérablement la recherche par l'IA.

Nous sommes satisfaits de l'implémentation de notre IA, mais il y a quelques inconvénients et limitations à noter :

- Comme le dictionnaire utilisé par l'IA est pondéré en fonction de la longueur des mots et de la difficulté de l'IA (choisie par le joueur), même une IA difficile peut manquer quelques mots longs apparemment faciles et évidents, tandis qu'une IA facile peut repérer quelques mots courts vraiment peu communs.
- Cette implémentation de l'IA est assez gourmande en mémoire. En effet, chaque mot de chaque tableau possède son propre ensemble de mots possibles répondant à la condition.
- La vitesse de l'IA dépend fortement de la taille du dictionnaire (nous remarquons que pour un petit dictionnaire (< 100 000 mots), l'IA a une complexité négligeable devant un grand dictionnaire (> 300 000 mots)

Finalement, la réalisation de l'IA est un challenge qui nous a beaucoup inspiré.

D'autres extensions ont été réalisées. Le joueur peut choisir :

- La langue du dictionnaire (Français, Anglais)
- Le nombre de lignes du plateau
- La longueur maximale (ex: 6,15,... au lieu de 9) et minimale (ex: 4,5,... au lieu de 3) d'un mot
- La possibilité d'annoncer Jarnac! (2 fois par défaut)
- L'aide de l'IA (à consommer avec modération ! Conseil : on peut l'utiliser au début pour la prise en main)
- De jouer contre l'IA
- De jouer à plusieurs (multijoueur)

Démonstration

Nous montrerons que la prise en main du jeu est aisée. Nous lancerons ensuite une partie IA VS IA. Pendant l'exécution, nous détaillerons plusieurs aspects de notre code. Lors des questions, les éléments les plus intéressants à évoquer seront les détails de l'implémentation de l'IA, notamment le cœur de la fonction *misAJourTabIAPourUneLigne*. Un autre point intéressant est une analyse au cœur du moteur du jeu. Enfin, la justification des tests peut également être pertinente.

Organisation du travail

Chaque semaine, nous nous retrouvions le vendredi pour mettre en commun notre avancée. Cela nous a permis d'une part de vérifier la cohérence de nos fonctions et d'autre part de faire de la programmation côte à côte.

Au total, cela représente 10 heures de travail en binôme et 10 heures de travail par binôme. Le plus long ayant été la correction des bugs, l'assemblage des fonctions dans la branche principale sur *GitLab*, et le traitement des exceptions.

Alexis a travaillé sur le moteur, la documentation / tests et les dictionnaires. Thanh Phát a travaillé sur le contrôleur, la vue en mode texte et l'Intelligence Artificielle. Nous avons tous les deux corrigés les bogues de l'un et l'autre. La collaboration s'est effectuée via GitLab. Sa prise en main était assez intuitive. Nous avons demandé à nos proches d'être les bêta-testeurs, rôle qu'ils ont apprécié et découvert.

Prise de recul

Initialement, nous compilions notre code avec g++ mais cela est rapidement devenu fastidieux aux vues des nombreux fichiers. Nous avons donc introduit CMake (ouvrant plus de compatibilité entre systèmes d'exploitation que Make) qui compile les fichiers d'une seule exécution. D'ailleurs nous avons eu quelques problèmes lors de l'exécution du fichier build.sh sur nos ordinateurs (Linux Pop! OS et MacOS) que nous avons résolu en synchronisant nos versions de compilation C++. Par ailleurs, nous n'avons pas réussis à collaborer via JupyterHub car le partage du dossier a été défaillant. Donc Thanh Phát s'est chargé des soumissions et Alexis des envois de son code, le tout étant fait en accord l'un avec l'autre.

Si nous avions à refaire un tel projet, nous pensons échanger davantage sur notre avancée personnelle durant la semaine pour ne pas être surpris lors de la mise en commun. Nous pensons également exploiter davantage Git, évitant ainsi de tout télécharger à chaque fois au format zip et d'économiser du temps, éviter des erreurs, et avoir plusieurs versions de notre programme. De plus, nous aurions beaucoup plus codé ce programme dans l'esprit de la programmation objet.

En conclusion, ce projet nous a beaucoup plu. Nous aurions aimé implanter l'interface graphique pour la soutenance, mais nous le ferons ultérieurement. Quoiqu'il en soit, il nous a été très enrichissant.