

# AI Game Programming : Awalé, où comment jouer automatiquement

Arno Lesage (no. 22202985)<sup>a,b</sup> and Jean-Jacques Viale (no. 22202859)<sup>a,c</sup>

<sup>a</sup>Université Côte d'Azur EUR - DS4H; <sup>b</sup>Master 1 Informatique, parcours IA; <sup>c</sup>Master 1 Informatique, parcours Informatique

## L'Awalé...

Awalé | MinMax | Coupe Alpha Beta | Fonction d'évaluation

# L'Awalé...

## 1. Présentation et règles du jeu

L'Awalé est un jeu de plateau et de stratégie tour-à-tour à deux joueurs d'origine Africaine datant du VIII<sup>ème</sup> siècle [Raabe \(2006\)](#). Le jeu se compose de deux composant :

- Un plateau constitué de trous,
- Des graines.

Au delà des règles traditionnelles variants de régions en régions, nous détaillerons dans la suite des règles adaptées, ajoutées et/ou simplifiées à des fins d'automatisations.

**Plateau et graines.** Dans cette version modifiée de l'Awalé, nous utiliserons un plateau composé de 16 trous numérotés de 1 à 16. Les trous impairs appartiennent au premier joueur [J1] et les trous pairs au deuxième joueur [J2]. Les graines, au nombre de 96, sont réparties de manière égale entre trois couleurs : rouge, bleu et transparent.

**Début de la partie.** Au début de la partie, deux graines de chaque couleur sont attribuées dans chaque trous du plateau. [J1] commence.

**Déroulé de la partie.** Au tour de [J1], [J1] choisi un trou lui appartenant et peut exécuté l'une des actions suivantes (sous réserve que les graines concernées soient dans le trou sélectionné) :

- Jouer les graines rouges [r],
- Jouer les graines bleues [b],
- Jouer les graines transparentes en tant que graines rouges [tr],
- Jouer les graines transparentes en tant que graines bleues [tb].

Si [J1] ne peut pas jouer à partir de ces trous à son tour, alors nous sommes dans une situation de famine, le tour de [J1] est passé et [J2] récupère l'ensemble des graines du plateau. Une fois son coup joué, [J2] joue de la même manière et rends la main à [J1].

*Remarque 1.1.* Un coup est noté Nc où N est le numéro du trou joué et c  $\in \{r, b, tr, tb\}$ .

**Essaimage.** L'essaimage correspond au mécanisme à travers lequel les graines sont jouées. Ainsi, une fois jouées, les graines rouges sont supprimés du trou sélectionné et sont réparties une à une dans les trous suivants en ordre croissant. Les graines bleues suivent un mécanisme similaire, mais ne répartissent les graines que dans les trous ne lui appartenant pas. Enfin, les graines transparentes peuvent être jouées soit comme les graines rouges soit comme les graines bleues.

*Remarque 1.2.* Le trou suivant le trou numéro 16 est le trou numéro 1.

*Remarque 1.3.* Si un graine transparente est joué, alors les graines de la couleur sélectionnée sont aussi jouées. Ainsi  $(\text{Play}[tr] \Rightarrow \text{Play}[r]) \wedge (\text{Play}[tb] \Rightarrow \text{Play}[b])$ . Si les graines de couleur correspondante ne se trouvent pas dans le trou sélectionné, alors les graines transparentes sont jouées sans impacts sur les autres couleurs.

*Remarque 1.4.* Si l'essaimage parvient à revenir sur le trou sélectionné, alors celui-ci est sauté et l'essaimage continue sur le trou suivant.

**Capture des graines.** À la fin de l'essaimage, le trou ayant reçu la dernière graine est vérifié. Si ce trou contient deux ou trois graines, alors la récolte commence et le joueur ayant fait l'essaimage récolte les graines contenu dans le trou et répète l'action pour tous les trous précédents (en ordre décroissant) jusqu'à ce que la condition ne soit plus vérifiée.

**Fin de la partie.** La fin de la partie peut arriver de plusieurs manières :

- **Limite de coup :** Si 400 coups ont été exécutés (200 coups par joueur), alors la partie s'arrête et le joueur ayant le plus de graine remporte la partie.
- **Limite de graine :** S'il y a moins de dix graines sur le plateau, alors la partie s'arrête et le joueur ayant le plus de graine remporte la partie.
- **Victoire :** Si un joueur a récolté au moins 49 graines, alors le joueur gagne la partie.
- **Égalité :** Dans le cas où la limite de coup ou la limite de graine est dépassée, si aucun joueurs ne domine, alors il y a égalité.

## 2. Implémentation des MinMax

MinMax, AlphaBeta, AlphaBeta avec profondeur adaptative

## 3. Implémentation des fonctions d'évaluation

Dans le cadre des fonctions d'évaluation, l'objectif est de déterminer les facteurs déterminants contribuant à la victoire d'un joueur si la profondeur d'un arbre MinMax (ou dérivé) n'est pas suffisante pour connaître l'entièreté du jeu.

À termes, une évaluation entre -1 (avantage [J2]) et 1 (avantage [J1]) est émise pour chaque coup afin d'estimer le mouvement le plus intéressant pour chaque joueur.

Template  $\LaTeX$  utilisé : iLCS Working Paper Template par Ernesto Calvo et Tiago Ventura sous licence Creative Commons CC BY 4.0.

**A. Une première tentative orienté attaque *raw*.** Dans cette première tentative, nous avons d'abord essayé d'identifier des caractéristiques "évidentes" pouvant déterminer la victoire d'un joueur :

- **Nombre de graine récoltées par joueur :** Plus un joueur a récolté de graine, mieux il est positionné pour la victoire. Ainsi, l'évaluation doit favoriser les positions maximisant le nombre de graine récolté par un joueur. Plus précisément, nous nous intéressons aux positions maximisant la différence de graine récoltées par les joueurs.  $[R_1]$
- **Nombre de graine par joueur :** Plus un joueur a de graines dans ces trous, plus il a de marge de manœuvre et donc plus élevée est la probabilité d'avoir des coups intéressants plus tard. Ainsi, nous voulons maximiser le nombre de graine dans les trous d'un joueur. Par soucis d'uniformisation, nous raisonnons comme précédemment sur la différence graine.  $[R_2]$
- **Nombre de graine transparentes par joueur :** Comme pour le nombre de graine dans les trous d'un joueur, plus un joueur a de graines transparentes, plus élevée est la marge de manœuvre en raison de la nature même de ces graines. Nous voulons donc maximiser cette valeur et raisonnerons sur la différence comme précédemment.  $[R_3]$
- **Potentielles captures :** Sûrement la variable la plus importante, il s'agit ici de savoir combien de graines peuvent être capturés pour chaque position évalué. Bien sûr, nous cherchons à maximiser ce nombre de potentielles captures.  $[R_4]$

Il ne nous reste maintenant qu'à gérer l'agrégation des valeurs ci-dessus mentionnées. Pour cela, nous appliquons déjà une étape de normalisation où chaque valeur est transposée entre -1 et 1. Ensuite, nous appliquons une simple somme pondérée afin d'obtenir un résultat entre -1 et 1.

Les poids de la sommes ont été ajustés à la main et n'ont pas été optimisés parfaitement en raison de contraintes de temps. Voici la somme utilisé dans le code :

$$raw(pos) = 0.4 [R_1] + 0.2 [R_2] + 0.1 [R_3] + 0.3 [R_4] \quad [1]$$

En raison d'une mauvaise implémentation de la logique de différence, cette fonction d'évaluation a tendance à faire des erreurs en présentant à  $[J1]$  des positions favorables pour  $[J2]$  comme étant favorable pour  $[J1]$ . Grâce aux poids, cet impact néfaste s'est avéré mitigé, mais à aussi rendu la détection du problème beaucoup plus chronophage lors des tests, ce qui a rendu ça correction tardive. À date de remise du rapport, une version corrigée existe et est sobrement nommé *corrected*.

**B. Deuxième tentative : ALL-IN en défense *defence*.** Dans une deuxième fonction d'évaluation, nous cherchons à résoudre un point faible de la fonction d'évaluation précédente : l'attaque est trop privilégié par rapport à la défense, ce qui rend certaines positions plus vulnérables à une attaque ennemi.

Pour résoudre ce problème, ou plutôt l'atténuer, nous faisons l'observation simple que moins un joueur possède de trous avec deux ou trois graines, moins il a de chance de se faire capturer ces graines ou de voir naître une chaîne de récolte trop longue profitant à l'adversaire.

Ainsi, nous cherchons à maximiser le nombre de trous ayant une graine ou plus de trois graines.  $[R_5]$

Une fois normalisé entre -1 et 1, nous avons l'agrégation suivante :

$$defence(pos) = 0.25 [R_1] + 0.06 [R_2] + 0.04 [R_3] + 0.3 [R_4] + 0.25 [R_5] \quad [2]$$

Lorem mettre si c'est bien...

## Conclusion

**REMERCIEMENTS.** Pas de remerciements particuliers

## References

Raabe, J. (2006). *Le Jeu de l'Awalé*. Editions L'Harmattan.