Travaux Dirigés Intelligence Artificielle n°4

Master 1

——

Algorithme de recherche locale, Algorithme génétique et algorithme $\alpha - \beta$

▸ **Exercice 1.** Résoudre le problème des 6 reines par l'algorithme de recherche locale (le problème consiste à placer 6 reines sur un échiquier 6×6 sans que deux d'entre elles ne se menacent mutuellement), en tenant compte du fait qu'il y a exactement une reine par ligne. Vous pouvez vous aidez de l'échiquier fourni à la figure 1, en vous fabriquant des jetons. On commence avec les reines placées comme indiqué. On utilise la même fonction d'utilité qu'en cours. Dans la situation de départ il y a 9 paires de reines qui s'attaquent mutuellement. On choisit de deplacer une reine vers la case qui permet de réduire le plus possible ce nombre.

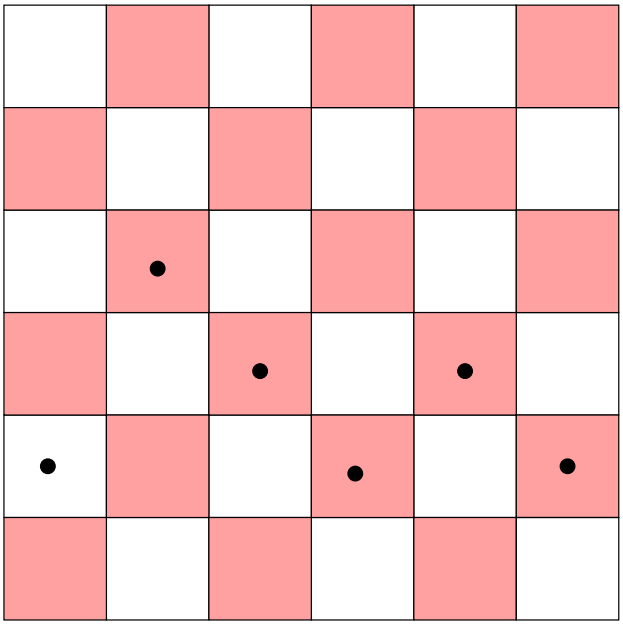

Figure 1: échiquier

▸ **Exercice 2.** On considère le problème du voyageur de commerce pour un graphe complet. On suppose que les villes sont numérotées de 1 à n.

Les codages suivants peuvent-ils permettrent d'appliquer l'algorithme génétique, sinon quels sont les problèmes posés et peut-on les résoudre facilement et efficacement :

- On écrit un "tour" comme une suite des n villes. On combine deux tours en les coupant de manière aléatoire, au même endroit, et en recollant les morceaux.

exemple avec $n = 5$:

parent 1	4	5	1 ⋮	2	3	$\longrightarrow$	enfant 1	4	5	1 ⋮	3	5
parent 2	4	2	1 ⋮	3	5		enfant 2	4	2	1 ⋮	2	3

Les mutations sont obtenues en inversant deux villes dans le tour.

- On représente un tour par sa matrice d'adjacence (bidirectionnelle). Ainsi le tour 4-5-1-2-3 (puis retour à 4) est-il représenté par la matrice

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix}$$

  Pour combiner les deux parents, on commence par garder les adjacences communes (c'est-à-dire faire un et bit à bit des deux matrices d'adjacence), puis on alterne les adjacences de chaque parent, enfin en cas de conflit, on fait appel au hasard pour garder un parcours correct (chaque ville apparaissant une unique fois).

▸ **Exercice 3.** Appliquer l'algorithme α-β sur l'arbre de jeu donné par la figure 2, la racine étant un nœud maximisant et l'évaluation des feuilles étant donnée.

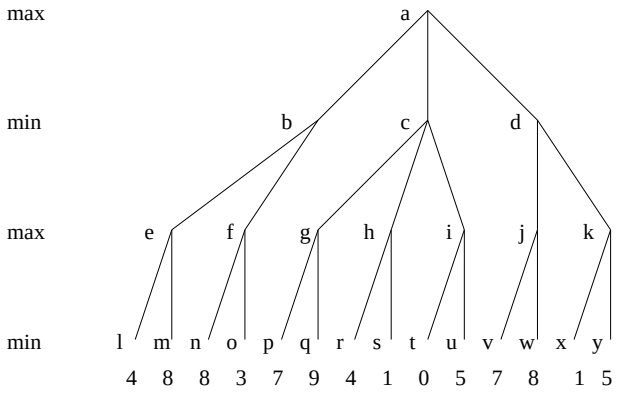

Figure 2: arbres de jeu

▸ **Exercice 4.** On considère un jeu dont le déroulement ressemble au jeu du solitaire, excepté que le plateau de jeu est différent (voir figure 3) et qu'il y a 2 joueurs : min qui tente de minimiser le nombre de pions restant à la fin et max qui tente de maximiser ce nombre. C'est min qui commence : dérouler l'algorithme $\alpha - \beta$ pour connaître le meilleur score que min peut espérer faire si les joueurs jouent le mieux possible.

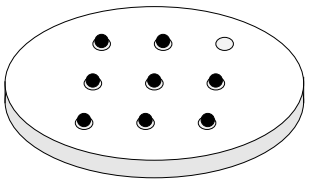

Figure 3: plateau de jeu