Rapport du projet de Modélisations, Optimisation, Graphes, et Programmation Linéaire

Spécialité

Agents Distribués, Recherche Opérationnelle, Robotique, Interaction et décision

Sorbonne Université - Faculté des Sciences de l'ingénieur

Réalisé par

Milo Roucairol et Ouaguenouni Mohamed El Hachemi

1 Variante séquentielle

1.1 Stratégie Aveugle

Dans cette partie, nous répondons aux questions relatives a l'implémentation de la stratégie optimale :

Q1. *$Q(d,k)$ est la probabilité d'obtenir k points en lançant d dés sachant qu'aucun dés ne donne 1, pour $d > 1$.*
Il y a une chance sur 5 d'obtenir chaque face du dé (face 1 exclue), donc $Q(d,k))$ correspond à la somme des probabilités d'obtenir le score moins voulu moins la valeur d'une face pondéré par la chance d'obtenir cette face, elle se calcule donc par la récurrence suivante :

$$Q(d,k) = \sum_{j=2}^{6} Q(d-1, k-j)/5 \tag{1}$$

avec :
$Q(1,k) = 1/5$ *si* $k \in 2, .., 6$ 0 *Sinon.*

Q2. La méthode demandée pour trouver la stratégie aveugle est appelée "dStar", elle retourne min(D, 6), on lancera ce nombre de dés dans la stratégie aveugle.

Q3. Ce n'est pas toujours optimal de jouer le mouvement qui maximise l'espérance du nombre de points, car dans la situation où il manque seulement deux points pour atteindre N, on a intérêt à lancer un seul dé pour minimiser les chances d'avoir 1.

1.2 Programmation dynamique

Q4. La formule permettant de calculer en chaque état l'espérance de gain est donnée par :

$$EG(i,j) = \sum_{d=1}^{D} p_d^{i,j} * (\sum_{k=N-i}^{N+6*D} P(d,k) - \sum_{k=1}^{N-i-1} P(d,k) * EG(j, i+k)) \tag{2}$$

avec $p_d^{i,j}$ la probabilité de jouer un nombre "d" de dés en etant dans l'état (i,j).
Les cas d'initialisation sont, pour tout a et b tel que $a < N < b < N + 6 * D$:
$E(a,b) = 0$ et $E(b,a) = 1$
Il s'agit de la somme pondérée par les $p_d^{i,j}$ pour tout d de tous les scores k possibles en lançant d dés : si k + i >= N, le joueur dont c'est le tour a gagné et pas l'autre joueur alors on y associe la victoire (de valeur 1) sinon, on y associe l'opposé de l'espérance de gain de l'autre joueur dans l'état suivant.

Q5. Pour déterminer une stratégie optimale, il faut déterminer quel nombre d jouer en chaque état (i, j). Les deux joueurs suivent la stratégie optimale donc on peut utiliser la formule (2) pour calculer le nombre de dés à jouer. En trouvant le vecteur p maximisant EG(i, j), on trouve la stratégie optimale.
On peut ramener ce problème à une instance de programmation linéaire, mais en l'absence d'autre condition que : $\forall i, j \in \{1...N\}^2 \sum_{d=1}^{N} p_d^{i,j} = 1$ le problème se ramène a :

$$d = \arg\max_{d \in 1...D}((\sum_{k=N-i}^{N+6*D} P(d,k) - \sum_{k=1}^{N-i-1} P(d,k) * EG(j, i+k))) \tag{3}$$

avec d le nombre de jets a jeter en l'état (i,j) pour jouer optimalement.
On a une fonction générant tous les P(d, k) et EG est initialisé, de cette façon, on peut donc trouver quel coup est optimal en tout état (i, j).
Le nombre de calculs est grandement réduit en utilisant un memo (programmation dynamique).

Q6. Si l'on gagnait 0 au lieu de 1 à chaque lancé contenant au moins un dés à 1, le calcul de l'espérance de gain EG(i, j) ferait appel à EG(j, i), l'algorithme tomberait donc dans une récursion infinie (avant de faire un stack overflow).

Q8. Le gain est de 1 en cas de victoire, -1 en cas de défaite : G1(a,b) = probabilité que a gagne - probabilité que b gagne

Tableau du nombre des nombres de victoires des joueurs et de l'espérance de gain de J1 :
Conformément à ce à quoi l'on pouvait s'attendre, la stratégie optimale se révèle plus performant

	D=10 ; N=100	D = 5, N = 100	D = 20, N = 50	D = 20, N = 100
G1(optimale, aveugle)	J1 : 1376 J2 : 1018 EG = 358/2394	J1 : 1260 J2 : 1003 EG = 257/2263	J1 : 1557 J2 : 1007 EG = 550/2564	J1 : 1345 J2 : 1013 EG = 242/2358
G1(optimale, aléatoire)	J1 : 2059 J2 : 1006 EG = 1053/3065	J1 : 2605 J2 : 1000 EG = 1605/3605	J1 : 1839 J2 : 1004 EG = 835/2843	J1 : 2167 J2 : 1002 EG = 1165/3169
G1(aveugle, aléatoire)	J1 : 1869 J2 : 1000 EG = 869/2869	J1 : 2243 J2 : 1004 EG = 1239/3247	J1 : 1553 J2 : 1008 EG = 545/2561	J1 : 1769 J2 : 1004 EG = 765/2773

que la stratégie aveugle, en particulier lorsqu'on laisse plus de liberté dans le choix du nombre de dés à lancer, puisqu'elle exploite cette liberté et pas la stratégie aveugle. La stratégie aveugle est plus performante que la stratégie aléatoire, la stratégie optimale aussi.

2 Variante Simultanée

2.1 Variante simplifiée

Q10. Pour calculer l'espérance de gain du joueur 1 quand il joue d1 dés et que son adversaire en joue d2, il faut parcourir les différentes possibilités de points qu'ils peuvent obtenir (6*d1 par 6*d2 possibilités) et sommer d'une part les possibilités ou 1 gagne et d'autre part celles ou il perd, ce qui donne l'expression suivante :

$$EG_1(d_1, d_2) = \sum_{i=1}^{6*d_1} \sum_{j=1}^{i-1} P(d_1, i) * P(d_2, j) - \sum_{i=1}^{6*d_1} \sum_{j=i+1}^{6*d_2} P(d_1, i) * P(d_2, j) \tag{4}$$

La matrice pour D=3 est donnée par :

(d1,d2)	1	2	3
1	0	-0.375000	-2.268519e-01
2	0.375000	0	-1.988169e-01
3	0.226852	0.198817	0

Q11. Si le joueur 2 connaît le vecteur de probabilité du joueur 1, il peut calculer selon ce qu'il jouerais son espérance de gain et ainsi la minimiser en prenant le nombre de dés a jeter qui minimiserait cette espérance.
Le problème qu'il doit résoudre est donc :

$$d = \arg \min_{d \in 1...D} (\sum_{i=1}^{d_1} p_1(i) * EG_1(i, d)) \tag{5}$$

Q12. Le joueur 1 doit minimiser le maximum d'espérance du joueur 2 il doit par conséquent résoudre le système :

$$\begin{array}{llll} \min\limits_{x} & \alpha & & \\ \text{s.t.} & EG_1^t P^t & \leq & (\alpha ... \alpha) \\ & \sum\limits_{i=1}^{d_1} p_i & = & 1 \\ & p_i & \geq & 0 \qquad \forall i \in 1...d_1 \end{array} \tag{6}$$

L'expérimentation de la stratégie aveugle contre la stratégie calculée par la résolution de ce PL donne en moyenne un pourcentage de victoires de 55.9963 %

Il est important de noter que l'expérimentation ici n'a de sens que pour D=5 et D=6, car en dessous de D=5 les deux stratégies (aveugle et optimale) retournent des stratégies pures équivalentes consistant a jeter un nombre maximal de dés, et au dela de D=6 les stratégies dans les deux cas ne changent plus.

2.2 Cas Général

Q15. Soit $a < N$; $i \geq N$; $j \geq N$ on distingue les cas suivants :
$EG_1(i,a) = 1$
$EG_1(a,j) = -1$
$EG_1(i,j) = 1$ si i>j
$EG_1(i,j) = -1$ si i<j
$EG_1(i,j) = 0$ si i=j

Q16. La formule de l'espérance gain $E^{d_1,d_2}(i,j)$ se construit à partir d'un état (i, j) représentant les scores des joueurs, où le joueur 1 joue d_1 dés et où le joueur 2 joue d_2 dés, comme suit :

- La probabilité que le score du joueur 1 soit supérieur a l'objectif et plus grand que celui du joueur 2 auquel on associe un gain de 1.
- La probabilité que le score du joueur 2 soit plus grand que l'objectif et plus grand que celui du joueur 1 auquel on associe un gain de -1
- la probabilité qu'aucun des deux joueurs n'aient atteint l'objectif auquel on associe l'espérance au tour suivant.

ce qui donne la formule suivante :

$$\begin{aligned} E^{d_1,d_2}(i,j) = & \sum_{k=N-i}^{N+6*D} \sum_{l=1}^{k-1} P(d_1,k) * P(d_2,l) \\ & - \sum_{k=N-j}^{N+6*D} \sum_{l=1}^{k-1} P(d_1,l) * P(d_2,k) \\ + \sum_{k=1}^{N-i-1} \sum_{l=1}^{N-j-1} & P(d_1,k) * P(d_2,l) * EG_1(i+k,j+l) \end{aligned} \tag{7}$$

La condition nécessaire a ce calcul reste cependant de connaître les $EG_1(k,l)$ pour $k > i; j > l$ ce qui nous oriente vers la conception d'un algorithme de **programmation dynamique**.

Q17. La matrice $E_1^{i,j}$ est calculable a partir de la matrice $EG_1(k,l)$ pour $k > i$; $j > l$ via la formule $\forall d1 \in \{1, ..., D\}$ et $\forall d2 \in \{1, ..., D\}$

$$E_1(i,j)_{d1,d2} = E^{d_1,d_2}(i,j) \tag{8}$$

La structure de données que nous choisissons de représenter est un dictionnaire (hashmap) qui associe a chaque entrée (i,j) représentant un état la matrice de gain $E_1(i,j)$.

Q18. Le calcul de l'espérance de gain en (i,j) un état donné se fait en connaissant la matrice de gain $E_1(i,j)$ associée a travers la résolution du programme linéaire suivant :

$$\begin{array}{llll} \max\limits_{x} & EG_1(i,j) & & \\ \text{s.t.} & E_1(i,j)P^t & \geq & (EG_1(i,j), ..., EG_1(i,j)) \\ & \sum\limits_{i=1}^{d_1} p_i & = & 1 \\ & p_i & \geq & 0 \qquad \forall i \in 1...d_1 \end{array} \tag{9}$$

L'algorithme de programmation dynamique a mettre en place pour permettre le calcul de l'espérance de gain en chaque état consiste en chaque état (i,j) a calculer $EG_1(i+1,j+1)$, $EG_1(i,j+1)$ et $EG_1(i+1,j)$ avant de résoudre le PL.
Cet algorithme s'appuiera sur les conditions d'initialisation mise en évidences dans la question 15 et sur l'utilisation d'un mémo.

Q19. Pour adapter l'algorithme il faut enregistrer à chaque résolution de PL dans un dictionnaire la distribution multinomiale associée a la stratégie optimale calculée.

Q20. Expérimentation : La simulation que nous proposons a été réalisée pour N=40 et D=5 puis pour N=60 et D=3 afin que cela puisse être calculable dans un temps raisonnable.
Nous avons effectué durant cette simulation 100 parties de 1000 tours chacune.
Pour la première simulation, Nous observons que la stratégie optimale est meilleure que la stratégie aveugle en moyenne, car elle gagne dans 51% des cas, nous remarquons également un écart type important de 3,71 qui ne permet donc pas de trancher de façon claire concernant la meilleure stratégie.
Cependant, quand on fait s'affronter la stratégie aveugle optimale et la stratégie optimale nous remarquons une victoire de la stratégie optimale dans 56.23% des cas avec un écart type de 4.11 ce qui montre la supériorité de la stratégie optimale.
Pour la seconde simulation en revanche nous observons une moyenne de victoire plus nette pour la stratégie optimale contre la stratégie aveugle avec une moyenne de : 54.589% de victoires pour un écart type de 5.36, nous expliquons cette progression par le fait qu'en changeant N le nombre d'état par lesquels doivent transiter les deux joueurs avant que l'un d'eux gagne augmente, et c'est dans ces états la que la stratégie optimale a un avantage conséquent.