Sorbonne Université
MASTER ANDROIDE M2 – MADI

le 13 Octobre 2022

Projet MADI

Le cavalier à marche aléatoire

Introduction

On s'intéresse aux déplacements d'un cavalier dans une grille donnée dont certaines cases sont interdites (cases colorées en noir). Le cavalier est initialement placé sur une case non-interdite de la grille (typiquement le coin supérieur gauche) et on cherche à atteindre une case but donnée (typiquement le coin inférieur droit) en un minimum de mouvements. En chaque case, le cavalier choisit une action parmi l'un des huit coups légaux autorisés aux échecs (chaque action correspond à une case cible distincte atteignable en un saut de cavalier et qui ne doit pas être hors de la grille et ne doit pas être une case interdite). Ces actions sont notées R, T, Y, U, J, H, G, F respectivement et sont représentées dans la figure ci-dessous (le rond jaune représente la position du cavalier et les lettres représentent les différentes cases cibles possibles correspondant aux 8 actions).

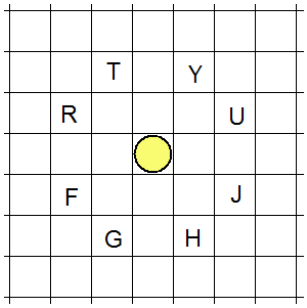

FIGURE 1 – Actions possibles

Les effets de ces différentes actions ne sont pas nécessairement déterministes. Sauf mention contraire stipulant qu'on est dans le cas déterministe, on supposera généralement que, lors d'un déplacement élémentaire de cavalier, la case cible n'est pas atteinte avec certitude et que l'une des 8 cases périphériques peut être atteinte (pourvu qu'elle ne soit pas interdite) au lieu de toucher la cible. Si q est le nombre de cases périphériques non-interdites autour d'une case cible donnée ($q \leq 8$), la probabilité d'atteindre la case cible est $1 - \frac{q}{16}$ et la probabilité de chaque case périphérique est $\frac{1}{16}$. Ainsi dans l'exemple ci-dessous, si l'on déclenche l'action J, les 8 cases périphériques autour de J sont celles inscrites dans le rectangle rouge. Ici on observe que 2 d'entre elles sont interdites donc ici $q = 6$. La probabilité de tomber sur J est donc de 0.625 et la probabilité de chaque case périphérique est 0.0625.

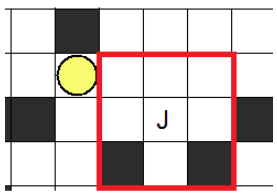

FIGURE 2 – Effet aléatoire d'une action

L'objet de ce projet est de modéliser différents problèmes de planification dans cette grille comme des processus décisionnels Markoviens, puis d'implanter des algorithmes de résolution et de tester les politiques optimales de déplacement dans cette grille à l'horizon infini. Certaines cases de la grille sont

marquées d'un point de couleur pour signifier qu'elles engendrent une conséquence spécifique lorsqu'on passe dessus. Cette conséquence est codifiée par une échelle de couleur (vert, bleu, rouge, noir). Selon les cas, les couleurs pourront désigner différéents niveaux de risque encourus ou de coûts, ou encore différents types de récompenses dans d'autres problèmes. Un exemple de telle grille est donné ci-dessous, le disque jaune représentant la position du cavalier, et les cases noires les cases interdites.

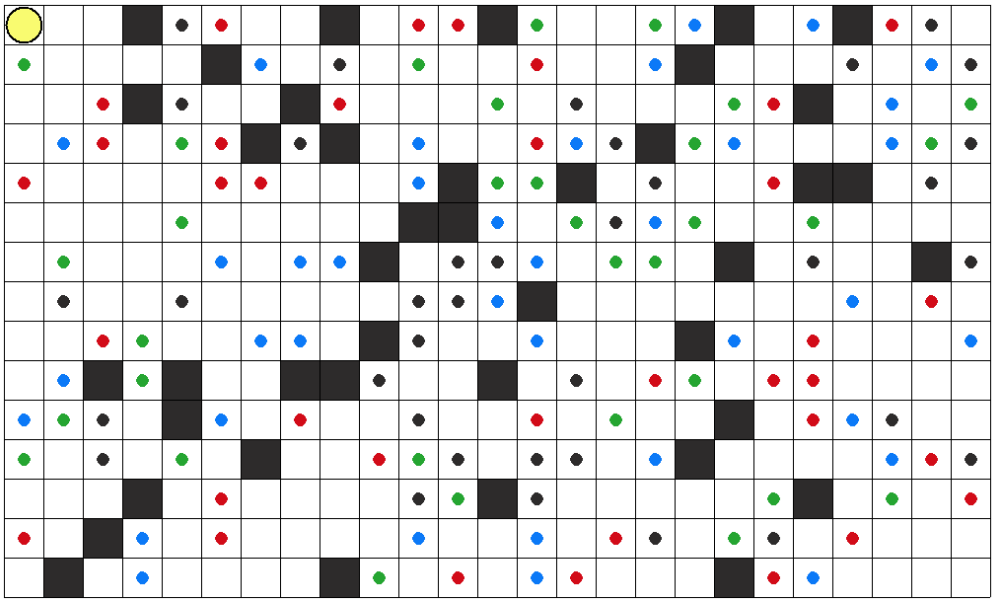

FIGURE 3 – Un exemple de grille tiré aléatoirement

Pour engendrer de telles grilles et simuler les déplacements d'un cavalier suivant une politique donnée on pourra utiliser le programme python mdpmadi22.py qui accompagne l'énoncé. Ce programme n'est donné qu'à titre indicatif, on pourra éventuellement le modifier et l'enrichir ou s'en inspirer pour faire une autre visualisation graphique. Le programme donné permet de guider manuellement le cavalier en indiquant les actions à effectuer au clavier. Il pourrait permettre aussi, en appuyant sur la touche 'espace', d'exécuter une politique préalablement calculée (par défaut un déplacement choisi aléatoirement s'execute dans la version initiale qui est donnée).

1) Recherche d'une trajectoire prudente

Dans un premier temps, on considère que les couleurs représentent le niveau de risque encouru en traversant chaque case, avec le code couleur suivant (vert : risque très faible, bleu : risque faible, rouge : risque moyen, noir : risque élevé). Les cases blanches (sans point de couleur) correspondent à un risque nul. Pour les autres on suppose que ces niveaux de risques sont codés par une fonction coût (additive) donnée par le tableau suivant :

x	vert	bleu	rouge	noir
$c(x)$	2	4	8	16

On suppose ici que chaque déplacement élémentaire de cavalier (un saut) à partir d'une case de couleur x à un coût immédiat de $1+ c(x)$. On cherche alors à déterminer la politique d'espérance de coût actualisé minimum pour atteindre la case cible.

a) Modéliser le problème comme un processus décisionnel Markovien en donnant une récompense significative lorsqu'on atteint la case but, par exemple un bonus (réduction de coût) de 1000 points mais on pourra essayer d'autres valeurs en fonction de la dimension de la grille choisie. Ecrire les équations de Bellman définissant la valeurn v_{ij} d'être dans une case (i, j) pour une politique stationnaire optimale et un facteur d'actualisation γ.

b) Dans un premier temps on se place dans le cas déterministe (la case cible est atteinte avec une probabilité 1 à chaque saut). Déterminer par itération de la valeur un chemin optimal de la case initiale vers la case but et tester sur une grille de taille puis 5×10, 10×15, 15×20 (on essaiera avec $\gamma = 1$ et $\gamma = 0.5$). On donnera les grilles en visualisant les trajectoires optimales et on donnera les temps de calcul.

c) On se place maintenant dans le cas de transitions aléatoires définies comme expliqué ci-dessus. On souhaite déterminer une politique optimale par itération de la valeur et effectuer des essais numériques de résolution de grilles de différentes tailles. Pour chaque taille on tirera 10 instances de grilles aléatoirement et on donnera dans un tableau le temps moyen de résolution et le nombre moyen d'itérations. On pourra également étudier l'impact de γ en faisant varier $\gamma = 0.9, \gamma = 0.7, \gamma = 0.5$.

d) Pour une grille donnée on essayera de piloter à la main quelques trajectoires menant à la case but et on comparera leur coût au coût moyen d'une politique mixte optimale observé sur 20 executions successives de cette politique sur la même grille. On pourra aussi comparer les coûts moyens observés avec la valeur espérée de la politique mixte optimale testée.

2) Recherche d'une politique équilibrée

Dans cette partie on considère des grilles contenant environ 15% de cases interdites, ainsi que 20% de cases rouges et 20% de cases bleues, les couleurs vertes et noires n'étant plus présentes.

a) On cherche une politique optimale équilibrée entre cases rouges et cases bleues pour atteindre l'état but à partir de l'état initial. Pour cela on utilise des coûts immédiats à deux composantes (c_1, c_2), de telle sorte que faire un saut depuis une case blanche engendre un coût immédiat de $(1, 1)$, faire un saut depuis une case bleue engendre un coût immédiat $(2, 0)$ et faire un saut depuis une case rouge engendre un coût immédiat de $(0, 2)$. Par exemple, une trajectoire qui fait 22 sauts de cavaliers dont 10 depuis une case blanche, 8 depuis une case bleue et 4 depuis une case rouge aura pour vecteur coût $(10 + 6 \times 2, 10 + 4 \times 2) = (22, 18)$, auquel il faut retirer le bonus final d'atteindre la case but sur chacune des deux composantes. Une trajectoire traversant autant de cases bleues que de cases rouges aura un vecteur de coûts cumulés équilibré. Pour évaluer une politique π, on utilise le vecteur $(c_1(\pi), c_2(\pi))$ qui représentent les coûts espérés actualisés de la politique π. On souhaite trouver une politique qui optimise le critère suivant :

$$\min_{\pi} \max\{c_1(\pi), c_2(\pi)\} \tag{1}$$

Expliquer pourquoi on ne peut utiliser l'itération de la valeur pour la résolution de ce problème. Proposer alors une approche de résolution reposant sur la formulation d'un MDP bi-objectif et sa résolution par programmation mathématique pour déterminer une politique mixte optimale.

b) Tester la résolution pratique du problème de recherche d'une trajectoire équilibrée entre le bleu et le rouge sur des grilles de différentes tailles avec l'approche proposée à la question précédente. On donnera ici encore les temps moyens de résolution. Simuler les politiques optimales obtenues en les éxécutant 20 fois sur la même instance. Calculer le vecteur moyen obtenu (c_1, c_2) qui représente l'espérance de coût empiriquement observée de la politique optimale selon les deux critères considérés.

c) Comment modifier l'approche proposée en a) pour déterminer une politique *pure* optimale au sens du critère (1) ? Tester cette approche et comparer ses performances à celles trouvées en b) sur les mêmes instances de problème.

Accès au solveur Gurobi et implémentation en python

Le projet sera implanté de préférence en python, langage qui permet de développer facilement les interfaces utiles pour visualiser la grille (voir exemple donné) mais aussi pour dialoguer facilement avec gurobi solver quand il s'agit de résoudre des programmes linéaires. Pour résoudre les programmes linéaires, le solveur gurobi (http ://www.gurobi.com/) est installé dans les salles en libre-service permanent mais peut aussi être installé sur des machines personnelles en téléchargeant depuis une adresse de Sorbonne Université.

Modalités de travail et délivrables

Le travail est à effectuer en binôme. Le binôme constitué devra être déclaré par mail à partrice.perny@lip6.fr (objet du mail : binome MADI) avant le jeudi 27 Octobre 2022. Les projets seront déposés au plus tard le vendredi 6 janvier 2023 à minuit sur le site moodle de MADI. Votre livraison sera constituée d'une archive zip nommée nom1_nom2.zip qui comportera les sources du programme, un fichier README détaillant comment exécuter le programme, et un rapport rédigé (un fichier au format pdf nommé nom1_nom2.pdf) qui présentera le travail effectué et les réponses aux différentes questions. Le plan du rapport suivra le plan du sujet. Il est fortement recommandé de rédiger son rapport avec LaTeX.