

Rapport MADI

Mathieu Berthelley - Élodie Cayez

8 janvier 2016

Résumé

On s'intéresse aux déplacements d'un cavalier dans une grille donnée dont certaines cases sont interdites. On considère que le cavalier est initialement dans le coin supérieur gauche de la grille et l'on veut atteindre une case cible 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

Introduction

Ce rapport présentera différentes approches du problème. En première partie on s'intéressera à une approche prudente, le cavalier devra se rendre d'un bout à l'autre de la grille en minimisant les pénalités induites par certaines cases colorées. Pour ce faire on utilisera l'algorithme d'itération de la valeur. En seconde partie le cavalier devra se rendre d'un bout à l'autre de la grille en équilibrant le nombre de fois qu'il passe sur les cases bleues et rouges. Pour ce faire on utilisera la programmation mathématique pour résoudre un PDM bi objectifs.

1 Recherche d'une trajectoire prudente

1.1 Modélisation

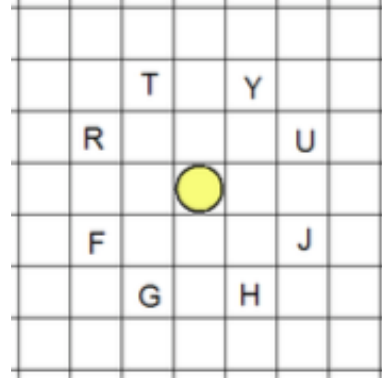
PDM = <S,A,T,R>

S Pour une matrice de taille $n \times m$ contenant b cases noires, il y a $n \times m - b$ états du monde. On notera S_i^j l'état du monde où le cavalier est sur la case (i, j) .

A Actions possibles : R, T, Y, U, J, H, G, F chaque élément correspond à un mouvement décrit sur cette figure.

Ainsi les voisins de l'état S_i^j sont les états :

- S_{i-2}^{j+1} si mouvement R
- S_{i-1}^{j+2} si mouvement T
- S_{i+1}^{j+2} si mouvement Y
- S_{i+2}^{j+1} si mouvement U
- S_{i+2}^{j-1} si mouvement J
- S_{i+1}^{j-2} si mouvement H
- S_{i-1}^{j-2} si mouvement G
- S_{i-2}^{j-1} si mouvement F



T Soit s l'état actuel du monde, a l'action effectuée et s' l'état attendu de l'action a . Soit $A(x)$ l'ensemble des adjacents de x , $N(x)$ l'ensemble des voisins de x au sens précédemment définie. On a alors :

$$T(s, a, s') = 1 - \frac{|N(s)|}{16} \quad (1a)$$

$$T(s, a, s'') = \frac{|N(s)|}{16}, \forall s'' \in N \quad (1b)$$

R Soit :

- V l'ensemble des cases marquées verte
- B l'ensemble des cases marquées bleue
- R l'ensemble des cases marquées rouge
- N l'ensemble des cases marquées noire

On a alors :

$$R(x) = c(blanc) - 1 = -1, \forall x \in Bl \quad (2a)$$

$$R(x) = c(vert) - 1 = -11, \forall x \in V \quad (2b)$$

$$R(x) = c(bleu) - 1 = -21, \forall x \in B \quad (2c)$$

$$R(x) = c(rouge) - 1 = -31, \forall x \in R \quad (2d)$$

$$R(x) = c(noir) - 1 = -41, \forall x \in N \quad (2e)$$

$$R(S_n^m) = 999 \quad (2f)$$

1.2 Équations de Bellman associées

Les équations de Bellman associées à un tel processus décisionel Markovien sont l'ensemble des équations de la forme :

$$V^*(s) = \max_a \left[-1 + \gamma \left(\frac{V^*(s'(a)) \cdot (1 - |N|)}{16} + \frac{|N|}{16} \cdot \sum_{s'' \in N(s'(a))} V^*(s'') \right) \right], \forall s \in Bl \quad (3a)$$

$$V^*(s) = \max_a \left[-11 + \gamma \left(\frac{V^*(s'(a)) \cdot (1 - |N|)}{16} + \frac{|N|}{16} \cdot \sum_{s'' \in N(s'(a))} V^*(s'') \right) \right], \forall s \in V \quad (3b)$$

$$V^*(s) = \max_a \left[-21 + \gamma \left(\frac{V^*(s'(a)) \cdot (1 - |N|)}{16} + \frac{|N|}{16} \cdot \sum_{s'' \in N(s'(a))} V^*(s'') \right) \right], \forall s \in B \quad (3c)$$

$$V^*(s) = \max_a \left[-31 + \gamma \left(\frac{V^*(s'(a)) \cdot (1 - |N|)}{16} + \frac{|N|}{16} \cdot \sum_{s'' \in N(s'(a))} V^*(s'') \right) \right], \forall s \in R \quad (3d)$$

$$V^*(s) = \max_a \left[-41 + \gamma \left(\frac{V^*(s'(a)) \cdot (1 - |N|)}{16} + \frac{|N|}{16} \cdot \sum_{s'' \in N(s'(a))} V^*(s'') \right) \right], \forall s \in N \quad (3e)$$

1.3 Itération de la valeur - transitions déterministes

Data: S, A, T, R , la description du PDM. ϵ , la précision voulue. γ , le facteur d'actualisation

Result: D^* , l'ensemble des politiques optimales.

$t \leftarrow 0$;

$V_0(s) \leftarrow 0, \forall s \in S$;

repeat

$t \leftarrow t + 1$;

foreach $s \in S$ **do**

foreach $a \in A$ **do**

$Q_t^a(s) \leftarrow R(s, a) + \gamma \sum_{s' \in S} T(s, a, s') \cdot V_{t-1}(s')$;

end

$V_t(s) \leftarrow \max_{a \in A} \{Q_t^a(s)\}$;

end

until $\max_{s \in S} \{|V_t(s) - V_{t-1}(s)|\} < \epsilon$;

foreach $s \in S$ **do**

$d(s) \leftarrow \operatorname{argmax}_{a \in A} \{Q_t^a(s)\}$;

end

return $d(s), \forall s \in S$;

Algorithm 1: algorithme d'itération de la valeur

1.4 Itération de la valeur - transitions aléatoires

1.5 Calibration de gamma

1.5.1 Théorie

Le choix d'un γ adéquat est d'une importance cruciale pour maximiser l'efficacité de la politique qui en dépend. Si il est trop petit, l'agent ne se préoccupe pas assez du futur, il y a peu de chance qu'il arrive jusqu'au but. Au contraire si il est trop grand, l'agent regarde trop dans le futur et tous ses voisins semble attirants. Nous avons réfléchi à un moyen de calculer le γ le plus approprié pour une grille donnée. Notons W , H respectivement la longueur et la hauteur d'une telle grille. La distance de Manhattan pour aller d'un bout à l'autre est $D = W + H$ (la diagonale). Dans ce problème le cavalier peut se déplacer de 4 cases d'un coup, on peut donc approximer le nombre de nécessaire de coups qu'il lui faudrait pour se rendre dans le coin inférieur gauche depuis le coin supérieur droit par $D_c = D/4$ (la diagonale pour le cavalier). On peut calculer la proportion de la récompense final qui arrivera une case x donnée avec la suite suivante :

$$U(x) = U(x - 1)^\gamma$$

ou plus précisément la proportion de la récompense final qui arrive à la case départ :

$$U(D_c) = U(D_c - 1)^\gamma$$

Par la suite nous posons $B = [\text{start}, \text{end}]$ la fenêtre dans laquelle nous souhaitons que la récompense perçue depuis la case départ se trouve. En résolvant par recherche dichotomique :

$$U(D) \in B$$

nous obtenons une valeur de γ de façon computationnelle. Une amélioration possible de cette méthode est de trouver une meilleure valeur pour D_c par un algorithme du plus court chemin par exemple. Vous trouverez ci-dessous l'algorithme que nous avons implémenté qui prend en entrée une grille et renvoie le gamma adéquat.

1.5.2 Algorithmie

Data: g , un grille, \min , \max la fenêtre voulue
Result: γ , le facteur d'actualisation le plus approprié pour g
 $B_\gamma \leftarrow \{0, 1\};$
 $\gamma \leftarrow \text{mean}(B_\gamma);$
 $D_c \leftarrow (g.\text{width} + g.\text{height})/4;$
repeat
 $\text{tmp} \leftarrow 1000;$
 foreach i **in** $\text{range}(D_c)$ **do**
 $\text{tmp} \leftarrow \text{pow}(\text{tmp}, \gamma);$
 end
 if $\text{tmp} \leq \min$ **then**
 $B_\gamma \leftarrow \{B_\gamma[0], \gamma\};$
 end
 if $\text{tmp} \geq \max$ **then**
 $B_\gamma \leftarrow \{\gamma, B_\gamma[1]\};$
 end
 $\gamma \leftarrow \text{mean}(B_\gamma);$
until $\text{tmp} \geq \min$ **and** $\text{tmp} \leq \max;$
return γ
Algorithm 2: algorithme de recherche dichotomique de gamma

1.5.3 Résultats

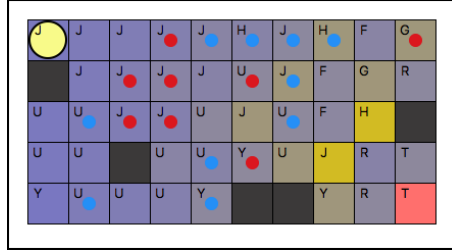


FIGURE 1 – $\gamma = 0.625$

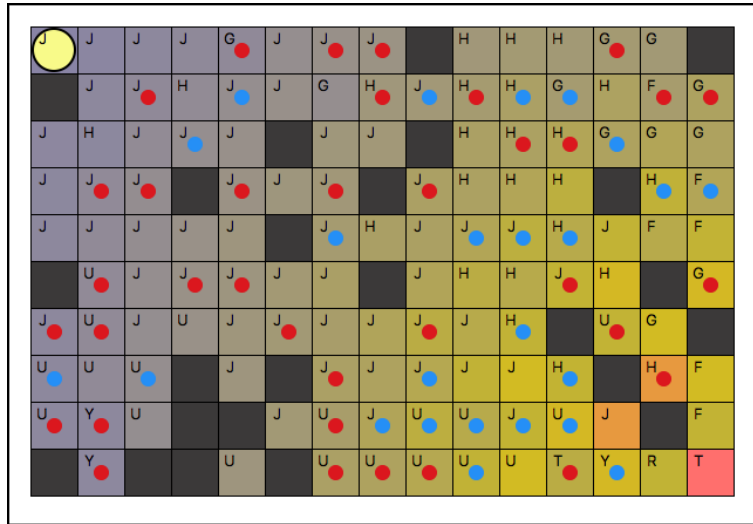


FIGURE 2 – $\gamma = 0.8$

Vous trouverez ci-dessus deux grilles auxquelles on a superposé à chaque état s une couleur correspondant à la valeur $V(s)$ (heatmap du bleu/jaune/rouge). Le gamma a été calculé avec la méthode décrite ci-dessus, vous remarquerez que dans les deux cas le dégradé s'étire approximativement de l'état de départ à l'état d'arrivée.

2 Recherche d'une politique équilibrée

2.1 Modélisation

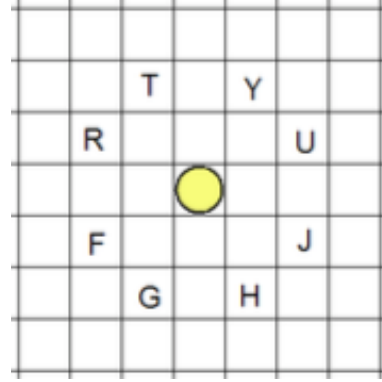
PDM = <S,A,T,R>

S Pour une matrice de taille $n \times m$ contenant b cases noires, il y a $n \times m - b$ états du monde. On notera S_i^j l'état du monde où le cavalier est sur la case (i, j) .

A Actions possibles : R, T, Y, U, J, H, G, F chaque élément correspond à un mouvement décrit sur cette figure.

Ainsi les voisins de l'état S_i^j sont les états :

- S_{i-2}^{j+1} si mouvement R
- S_{i-1}^{j+2} si mouvement T
- S_{i+1}^{j+2} si mouvement Y
- S_{i+2}^{j+1} si mouvement U
- S_{i+2}^{j-1} si mouvement J
- S_{i+1}^{j-2} si mouvement H
- S_{i-1}^{j-2} si mouvement G
- S_{i-2}^{j-1} si mouvement F



T Soit s l'état actuel du monde, a l'action effectuée et s' l'état attendu de l'action a . Soit $A(x)$ l'ensemble des adjacents de x , $N(x)$ l'ensemble des voisins de x au sens précédemment définie. On a alors :

$$T(s, a, s') = 1 - \frac{|N(s)|}{16} \quad (4a)$$

$$T(s, a, s'') = \frac{|N(s)|}{16}, \forall s'' \in N \quad (4b)$$

R Soit :

- Bl l'ensemble des cases non marquées
- B l'ensemble des cases marquées bleue
- R l'ensemble des cases marquées rouge

On a alors :

$$R(x) = -2, \forall x \in Bl \quad (5a)$$

$$R(x) = c(bleu) - 2 = -1, \forall x \in B \quad (5b)$$

$$R(x) = c(rouge) - 2 = -1, \forall x \in R \quad (5c)$$

$$R(S_n^m) = 1000 - 2 = 998 \quad (5d)$$

2.2 Résolution par programmation mathématiques

2.2.1 Algorithmie

$$\max \sum_{s \in S} V_s - z \quad (6a)$$

$$sc : z \leq \sum_{s \in Red} V_s \quad (6b)$$

$$z \leq \sum_{s \in Blue} V(s) \quad (6c)$$

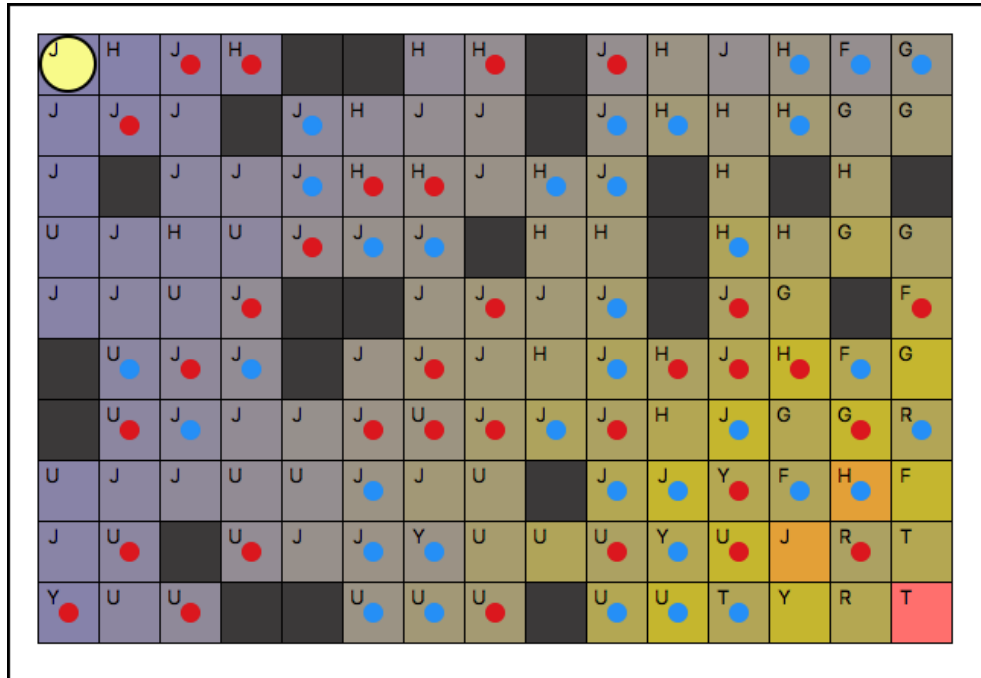
$$V_s \geq R(s') + \gamma \left[\left(1 - \frac{|A(s')|}{16} \right) \cdot V_{s'} + \sum_{s'' \in A(s')} \frac{1}{16} \cdot V_{s''} \right] \quad (6d)$$

$$V(s) \in \mathbb{R}^+, \forall s \in S \quad (6e)$$

avec :

- S, ensemble des états
- Blue, Red, ensemble des états de S tel quel la case est marquée respectivement bleu ou rouge
- End, l'ensemble des états finaux, ici End = (width, height)
- V(s), ensemble des états voisins de s
- A(s), ensemble des états adjacents de s
- (6b), (6c) on minimise la plus grande espérance entre $C_{Red}C_B$
- (6d) $\forall s \in S, \forall s' \in N(s)$
- (6e), V_s , la valeur de la récompense espérée pour l'état S

2.2.2 Résultats



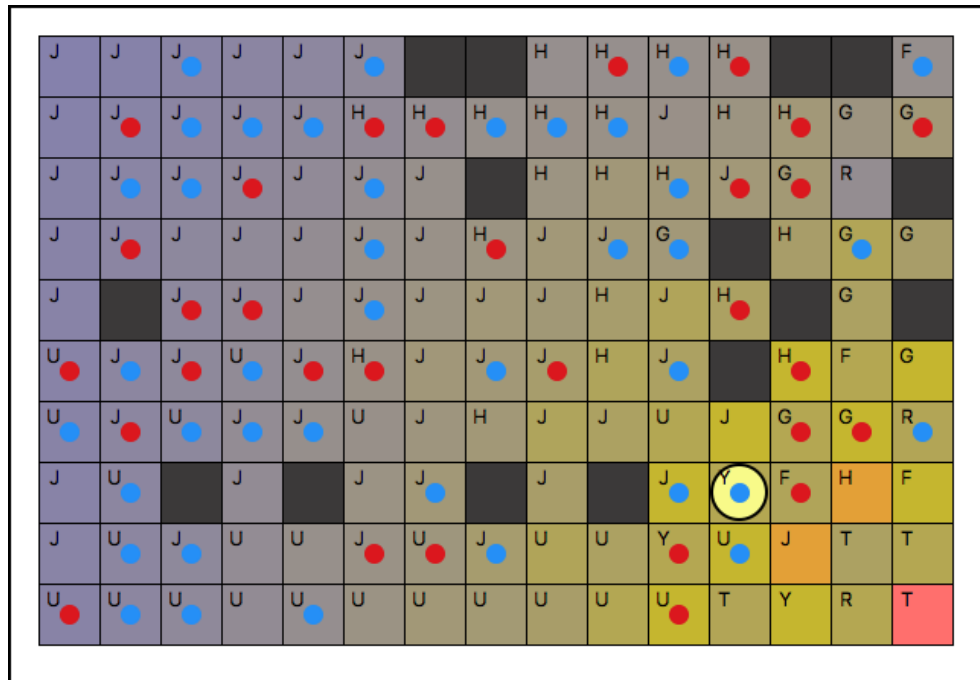
Ci-dessus une capture d'écran de la grille une fois la politique calculée. Chaque case est coloriée selon son espérance d'utilité, de plus on trouve dans le coin supérieur gauche de chaque case le coup à jouer pour suivre la politique.

```
##### PL #####
BLUE:  4.1
RED:   4.6
Coups: 15.7
Score: 1011.7
Computing: 0.16
#####
```

FIGURE 3 – résultat moyen sur 1000 répétitions

2.3 Résolution par itération de la valeur

2.3.1 Résultats

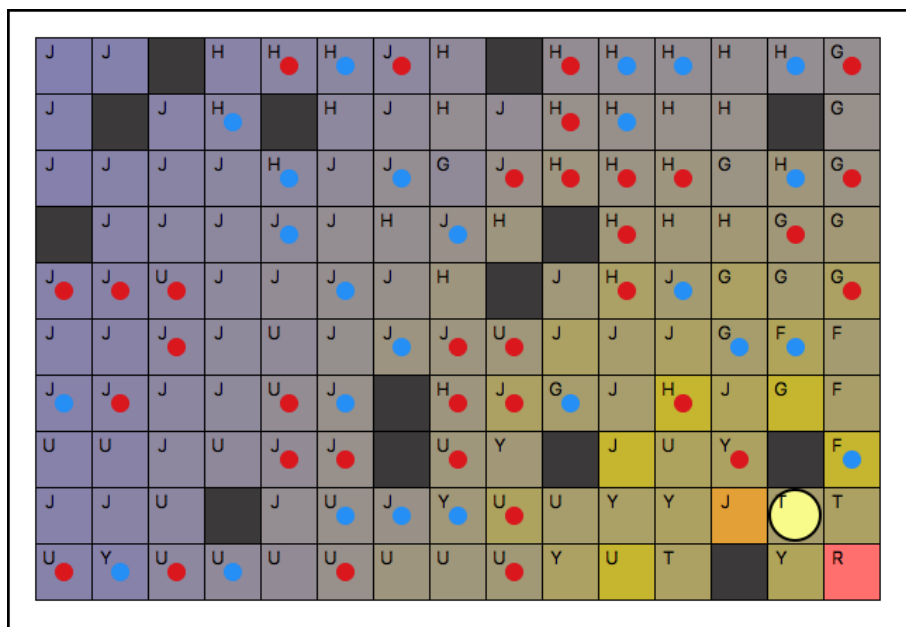


Ci-dessus une capture d'écran du la grille une fois la politique calculée. Chaque case est coloriée selon son espérance d'utilité, de plus on trouve dans le coin superieur gauche de chaque case le coup à jouer pour suivre la politique.

```
#### Iteration ####
BLUE:  3.7
RED:   1.9
Coups:  22.3
Score:  995.4
Computing: 4.38
#####
```

FIGURE 4 – résultat moyen sur 1000 répétitions

2.4 Comparaison



```

##### PL ##### Iteration ####
BLUE:  2.9          BLUE:  1.1
RED:    2.1          RED:   3.4
Coups:  13.1         Coups:  17.9
Score:  1013.2        Score:  968.7
Computing: 0.15       Computing: 4.78
#####

```

FIGURE 5 – résultat moyen sur 1000 répétitions