|  |  |
| --- | --- |
| SMA  Résolution du Taquin | Résumé  Dans ce rapport est exposé une méthode de résolution du Taquin à l’aide d’un système multi agents.  Jérémy Dollé, Etienne Debard  5A Informatique |

TABLE DES MATIERES

[Introduction 2](#_Toc500324096)

[Librairie MadKit 2](#_Toc500324097)

[Approche naïve 2](#_Toc500324098)

[Principe général 2](#_Toc500324099)

[Exemple 3](#_Toc500324100)

[Etape 1 : Déplacement simple 3](#_Toc500324101)

[Etape 2 : Déviation 3](#_Toc500324102)

[Etape 3 : Amélioration du déplacement 4](#_Toc500324103)

[Limite 4](#_Toc500324104)

[Approche multi-agent avec dialogue 4](#_Toc500324105)

[Principe général 4](#_Toc500324106)

[Système d’agression 4](#_Toc500324107)

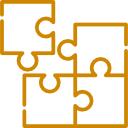
[Système de message 4](#_Toc500324108)

[Exemple 4](#_Toc500324109)

[Conclusion 4](#_Toc500324110)

CF8600

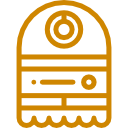
# Introduction

Dans ce rapport nous allons exposer les choix que nous avons pris pour la résolution du système du Takin. Dans un premier temps nous allons formuler une approche naïve en exposant le principe général ainsi que le résultat obtenu. Ensuite, nous allons expliquer une approche multi-agent en commençant par le principe général ainsi que le résultat obtenu.

# Librairie MadKit

Le but De ce TP étant l’implémentation d’un algorithme d’apprentissage, nous avons décidé de choisir une librairie pour tout ce qui est question d’affichage pour éviter de perdre du temps à concevoir une interface et aussi pour vérifier rapidement nos résultats. La librairie MadKit est très simple d’utilisation est colle parfaitement à nos besoins, c’est pour cette raison que nous avons opté pour cette dernière.

# Approche naïve

Pour commencer, nous avons implémenté un algorithme naïf que nous avons petit à petit amélioré. Dans la suite, nous avons commencé par expliquer le principe général avec les améliorations apporté petit à petit à notre solution. Ensuite des exemples afin d’illustrer notre propos et pour finir les limites à cette solution.

## Principe général

Premièrement, notre agent agissait de la manière suivante : l’agent se déplace sur la bonne colonne, s’il est sur la bonne colonne alors il se déplace sur la bonne ligne. De cette façon, si le nombre d’agents est élevé alors nos agents se bloquaient les uns les autres assez rapidement. Nous avons donc décidé d’implémenter de façon naïve une déviation. Pour ce faire, nous avons fait en sorte que lorsqu’un agent veux se déplacer sur une case déjà occupée alors il choisit une autre case paris celle disponibles. Afin d’éviter que nos agents fassent les mêmes déplacements les cases indisponibles sont celle qui sont occupées + la case précédemment occupée par l’agent. Afin d’améliorer l’efficacité de notre algorithme, nous avons changé la stratégie de déplacement des agents (Bonne colonne puis bonne ligne). Nous avons dit à notre agent de se déplacer sur la bonne ligne ou la bonne colonne en fonction de ce qui est le plus proche : si la colonne est plus proche que la ligne alors l’agent va d’abord se placer sur la colonne et inversement.

## Exemple

Dans les trois exemples qui vont suivre, « X » est un agent, la case rouge représente la case cible de l’agent X.

### Etape 1 : Déplacement simple

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| X |  |  |  |  | X |  |  |  |  | X |  |  |  |  | X |
|  |  |  |  |  |  |  |  | Comme expliqué précédemment, L’agent « X » se déplace d’abord sur la bonne colonne pour ensuite se déplacer sur la bonne ligne. |  |  |  |  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  | X |

### Etape 2 : Déviation

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| X |  | O |  |  | X | O |  |  |  | O |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | X  L’agent « X » se déplace de la même façon (colonne puis ligne). Par contre cette fois un autre agent se trouve sur son chemin. A l’étape 2 l’agent n’a que deux solutions car devant lui se trouve l’agent et derrière lui il se trouve la case d’où il vient et ne peut donc pas y retourner. Une fois la déviation faite alors l’algorithme normal reprend. |  |  |  |  | X |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  | X |

### Etape 3 : Amélioration du déplacement

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| X |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | X |  |  |  | Comme expliqué précédemment, Nous avons décidé d’améliorer le déplacement des agents en se déplaçant soit vers la bonne ligne soit vers la bonne colonne. Ici l’agent se trouve à une distance de 1 de la bonne ligne et 3 de la bonne colonne. L’agent va donc se déplacer sur la bonne ligne puis sur la bonne colonne. | X |  |  |  |  | X |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  | X |

## Limite

Après avoir implémenté cette approche naïve, nous pouvons relever quelques limites. En effet, les agents marchent indépendamment les uns des autres, il n’y a aucune communication entre eux. Ce qui pose un problème de taille : si un agent est bien placé il ne bougera plus ce qui fait que si un agent est entouré d’agents bien positionnés il sera coincé et ne pourra pas atteindre son objectif.

# Approche multi-agent avec dialogue

## Principe général

### Système d’agression

### Système de message

## Exemple

# Conclusion