AVIGNON UNIVERSITÉ

Rapport Projet - Tetravex

Étudiant
Michel Marie LAMAH

4 décembre 2022

Master Informatique
Intelligence Artificielle

UE Conception Logicielle
ECUE Programmation Parallèle

Responsable
Mikael ROUVIER

UFR
SCIENCES
TECHNOLOGIES
SANTÉ

AVIGNON UNIVERSITÉ
CERI

CENTRE
D'**ENSEIGNEMENT**
ET DE **RECHERCHE**
EN **INFORMATIQUE**
ceri.univ-avignon.fr

Sommaire

1 Manuel d'utilisation

Ce programme dispose de 2 modes d'exécution :

1. **Avec menu** : vous donnant l'opportunité de choisir le format à exécuter ainsi que l'algorithme à utiliser ;
2. **Sans menu** : dans ce cas vous ne verrez que le résultat en fonction de la commande utilisée ;

En plus, ce programme utilise des caractères spéciaux qui n'affichent pas dans une console normale, du coup pour l'afficher dans votre console, il va falloir modifier le **codepage** dans cette derniere, à chaque nouvelle ouverture de la console à travers la commande 2, ci-dessous :

```
make set_code_page
```

Console 1. Modification du codepage

Si vous ne disposez pas du fichier **.exe**, pour le générer, il suffit de faire :

```
make compile
```

Console 2. Générer fichier exe

1.1 Sans Menu

Pour exécuter ce programme sans passer par le menu, il suffit de se placer dans le repertoire du projet et d'exécuter la commande 3 ci-dessous :

```
make algo_format
```

Console 3. Compilation et exéctuion Automatique

> **Remarque :** Les valeurs des `algo` sont présenté dans le tableau 1a et les `format` sont présenté dans le tableau 1b

Algorithme
iterative
recursive
thread_pool
thread_vector

(a) Code des algorithmes

Format
2x2
3x3
4x4
5x5
6x6
7x7
8x8

(b) Code des format

Table 1. Les codes des algorithmes et formats

Pour exécuter par exemple le format **7x7** avec l'algorithme **threadpool**, on fait comme suit :

```
make thread_pool_7x7
```

Console 4. Format 7x7 avec l'algo ThreadPool

1.2 Avec Menu

Pour pouvoir utiliser ce programme, il faudrait tout d'abord se placer dans le répertoire du projet, ouvrir la console et exécuter la commande 5 ci-dessous.

```
make menu
```

Console 5. Compilation et exéctuion

Ensuite, vous aurez ce menu, vous demandant de choisir le format à utiliser. Votre choix devra être compris entre **2** et **8**. Par la suite appuyez sur Entrée pour choisir l'algorithme

```
============FORMAT=============
[2]->Format 2x2
[3]->Format 3x3
[4]->Format 4x4
[5]->Format 5x5
[6]->Format 6x6
[7]->Format 7x7
[8]->Format 8x8
================================
Veuillez choisir le format [2-8]: 8
Vous avez choisi le format 8x8
Appuyez une touche pour continuer pour choisir l'algorithme a utiliser
```

Figure 1. Format

Dans le menu qui s'affiche on choisit l'algorithme à utiliser, le choix étant compris entre **1** et **4**

```
============ALGORITHME=============
[1]->Recursive (Sequentielle)
[2]->Iterative (Sequentielle)
[3]->ThreadPool (Multithread)
[4]->ThreadVector (Multithread)
================================
Veuillez choisir l'algo [1-4]: 3
Vous voulez bien resoudre le format 8x8 avec l'algo ThreadPool (Multithread)
```

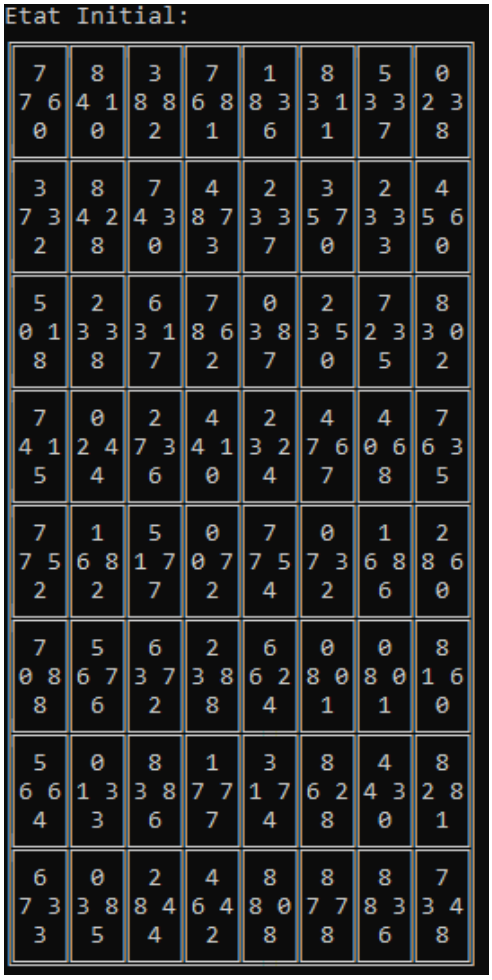
Figure 2. Algorithme

Il ne suffit d'attendre que la fin du traitement et on obtient alors les résultats suivant :

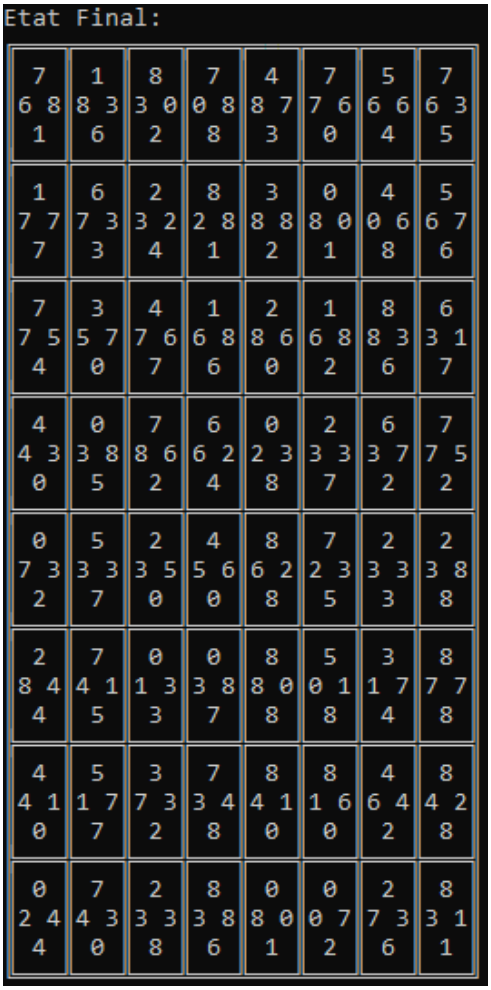
```
═══════════════════════════════════════════════════
ThreadPool (Multithread)
═══════════════════════════════════════════════════
m = minutes
s = seconde
us = milliseconde
us = microseconde
ns = nanoseconde
Temps d'execution: 279.869757200 s ==> 4 m 39 s 869 ms 757 us 199 ns

Format: 8 x 8
```

Figure 3. Statistiques



(a) Plateau d'entrée



(b) Plateau de Sortie

2 Explication des algorithmes

2.1 Modélisation

Ci-dessous, les **classes de base** utilisées pour la réalisation de ce projet

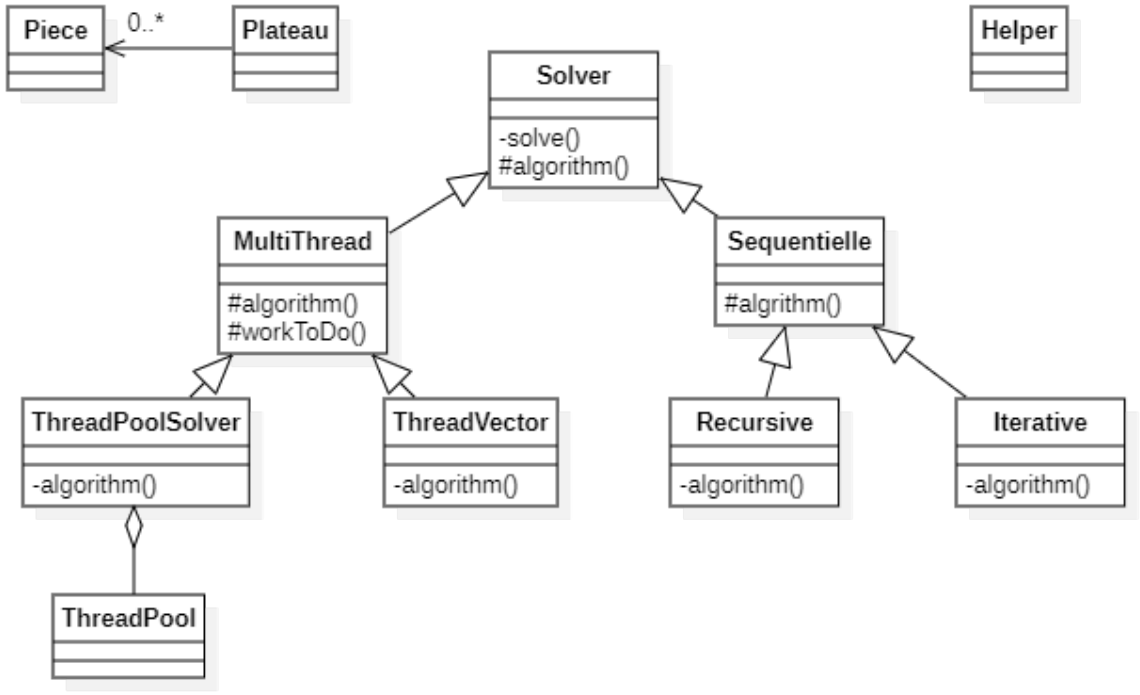


Figure 5. Les classes de base

Ensuite j'ai utilisé le pattern **Factory Method** pour créer nos **Solver**

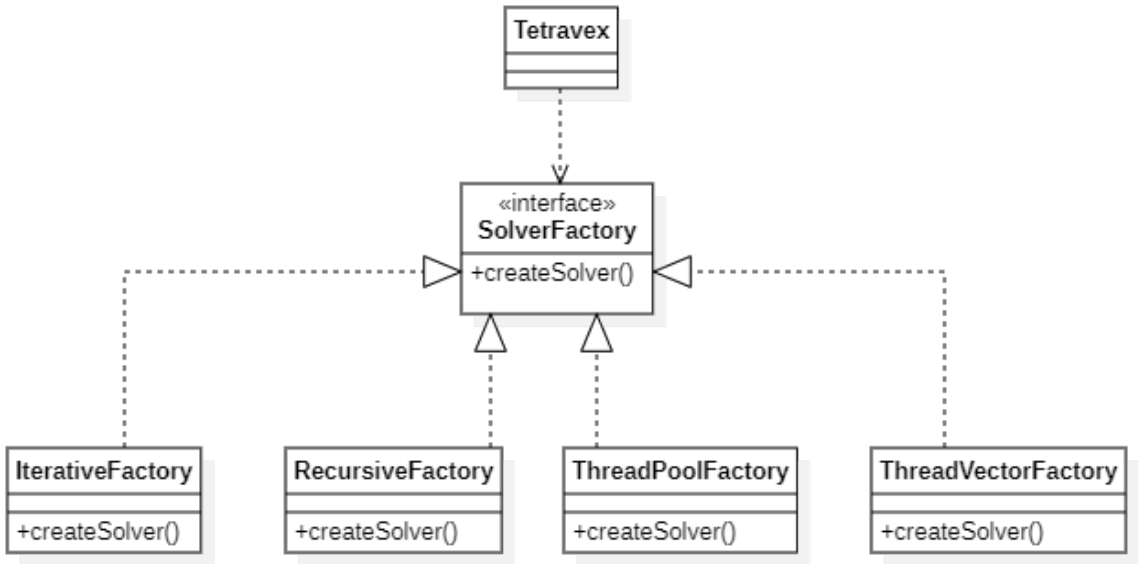


Figure 6. Design Pattern Factory Method

2.2 Explication

La méthode **itérative** utilise juste une simple boucle et parcours pièce par pièce jusqu'à ce que toutes les pièces soient placés et à chaque tour, dès qu'une pièce n'est pas le bon annule le déplacement

La méthode **récursive**, ici en lieu et place d'utiliser une simple boucle, on utilise une récursivité, jusqu'à ce qu'on trouve la solution.

La méthode avec le **vecteur de threads** utilise un thread associé à chaque pièce l'ajoute dans un vecteur puis les exécutent.

La méthode avec le **threadPool** utilise aussi un vecteur de threads avec une queue de tâche et à chaque q'un thread est dispo on lui fait passer une tâche à exécuter.

3 Courbes de comparaison

Le tableau 2 ci-dessous, montre les temps d'exécution moyen en seconde obtenu après **10** exécution de chaque algorithme.

Format	Iterative	Recursive	ThreadVector	ThreadPool
2x2	0	0	0	0,00020165
3x3	0	0	0,00166462	0,00156292
4x4	0	0	0,00357298	0,0005112
5x5	0	0,00176398	0,00035073	0,00010156
6x6	0,03044088	0,031106	0,01208871	0,01314513
7x7	7,791253	8,442841	1,9921956	0,865351
8x8	490,6894	524,4767	261,2228	262,9638

Table 2. Les durées d'exécution en fonction de l'algorithme utilisé

On obtient alors les diagrammes ci-dessous :

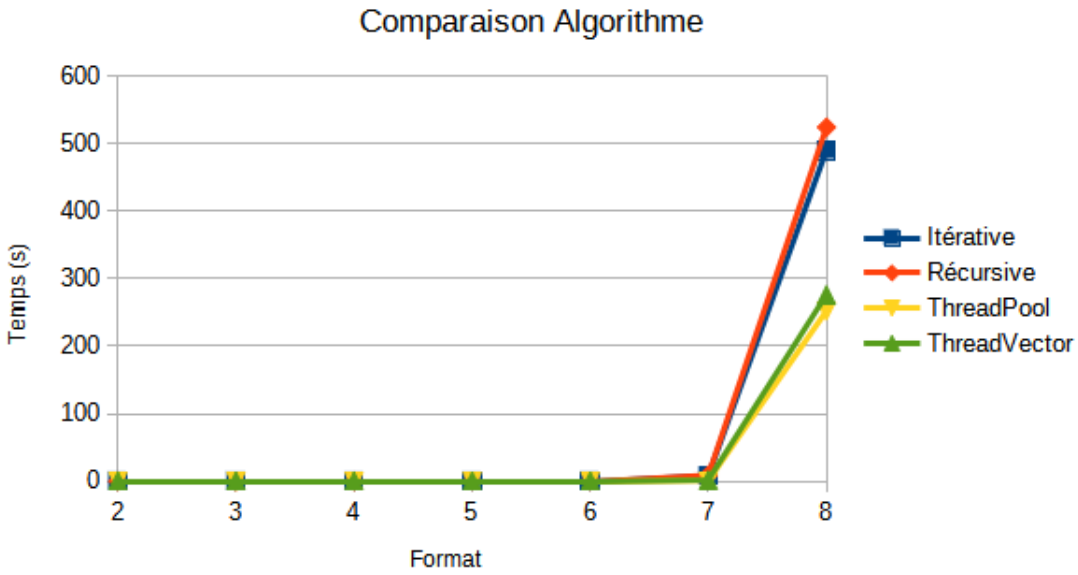


Figure 7. Courbe de comparaison

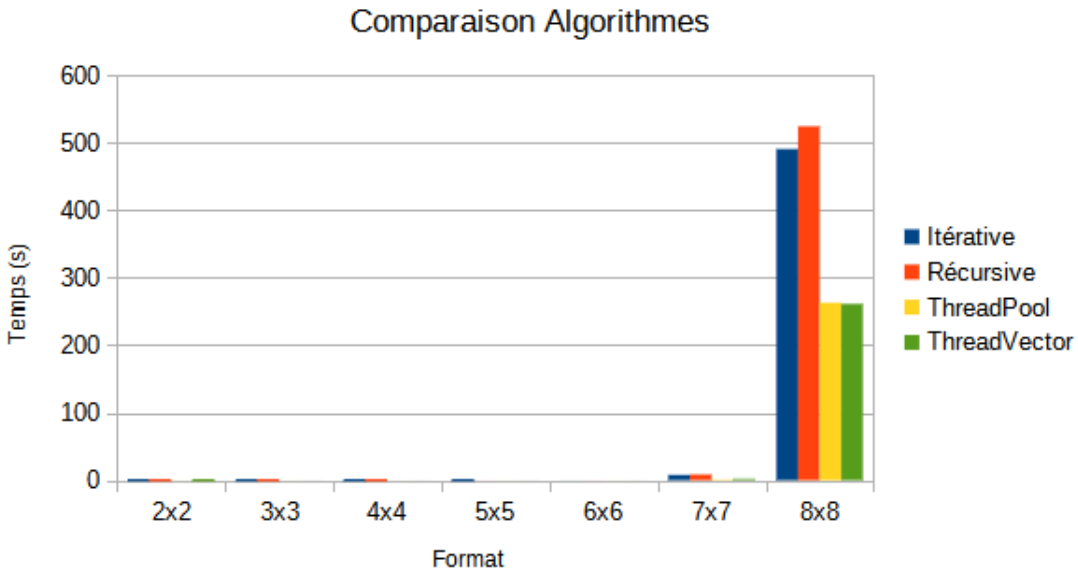


Figure 8. Diagramme à barre de comparaison

4 Interprétation des résultats obtenus

Les temps d'exécution présent dans le tableau 2 constituent la moyenne après 10 exécution de chaque algorithme.

On remarque que plus le format augmente, plus les méthodes utilisant notamment les threads trouvent rapidement la solution ce qui est d'ailleurs logique compte tenu de leur capacité à exécuter en parallèle les tâches.

De plus vu que la création des threads nécessite du temps on remarque que pour les petits formats, les méthodes multi-thread mettent souvent un peu de temps pour créer les temps ce qui justifie le fait que leur temps n'est quasiment pas nuls pour les petits formats.

En plus ces temps d'exécution varient d'un fichier à un autre, c'est-à-dire que plusieurs fichiers avec le même format peuvent aboutir à tes temps différents.

Pour finir, je tiens à rappeler que ces temps peuvent aussi varier d'un ordinateur à un autre c'est-à-dire que plus votre ordinateur est performant et plus les temps seront petits.