Résoudre de manière optimale le Rubik's cube

Kevin CHEN, Yuxiang LI

Résumé
TODO

Mots clés
IDA* — Pattern Database — HashMap

*__Code source__: https://github.com/lyx-x/Rubik

Table des matières

1. Analyse du sujet

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

and some mathematics $\cos \pi = -1$ and α in the text[1].

2. Implémentation

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

$$\cos^3 \theta = \frac{1}{4}\cos\theta + \frac{3}{4}\cos 3\theta \tag{1}$$

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula

[1] And some mathematics $\cos \pi = -1$ and α in the text.

Figure 1. Wide Picture

libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

1. First item in a list
2. Second item in a list
3. Third item in a list

2.1 Organisation du code

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Paragraph Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

Paragraph Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Donec odio elit, dictum in, hendrerit sit amet, egestas sed, leo. Praesent feugiat sapien aliquet odio. Integer vitae justo. Aliquam vestibulum fringilla lorem. Sed neque lectus, consectetuer at, consectetuer sed, eleifend ac, lectus. Nulla facilisi. Pellentesque eget lectus. Proin eu metus. Sed porttitor. In hac habitasse platea dictumst. Suspendisse eu lectus. Ut mi mi, lacinia sit amet, placerat et, mollis vitae, dui. Sed ante tellus, tristique ut, iaculis eu, malesuada ac, dui. Mauris nibh leo, facilisis non, adipiscing quis, ultrices a, dui.

2.2 Algorithmes de recherche

Morbi luctus, wisi viverra faucibus pretium, nibh est placerat odio, nec commodo wisi enim eget quam. Quisque libero justo, consectetuer a, feugiat vitae, porttitor eu, libero. Suspendisse sed mauris vitae elit sollicitudin malesuada. Maecenas ultricies eros sit amet ante. Ut venenatis velit. Maecenas sed mi eget dui varius euismod. Phasellus aliquet volutpat odio. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Pellentesque sit amet pede ac sem eleifend consectetuer. Nullam elementum, urna vel imperdiet sodales, elit ipsum pharetra ligula, ac pretium ante justo a nulla. Curabitur tristique arcu eu metus. Vestibulum lectus. Proin mauris. Proin eu nunc eu urna hendrerit faucibus. Aliquam auctor, pede consequat laoreet varius, eros tellus scelerisque quam, pellentesque hendrerit ipsum dolor

sed augue. Nulla nec lacus.

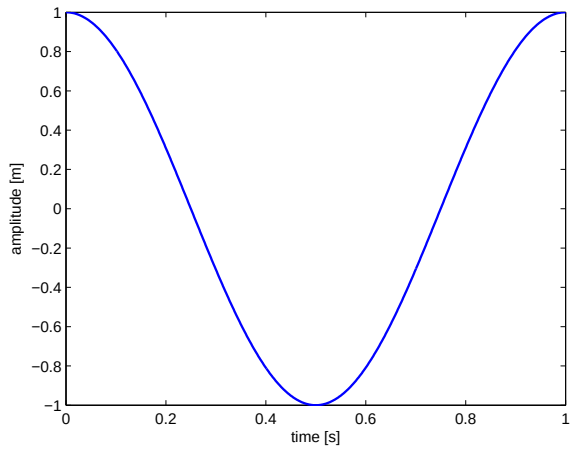

Figure 2. In-text Picture

Reference to Figure 2.

3. Résultats

Suspendisse vitae elit. Aliquam arcu neque, ornare in, ullamcorper quis, commodo eu, libero. Fusce sagittis erat at erat tristique mollis. Maecenas sapien libero, molestie et, lobortis in, sodales eget, dui. Morbi ultrices rutrum lorem. Nam elementum ullamcorper leo. Morbi dui. Aliquam sagittis. Nunc placerat. Pellentesque tristique sodales est. Maecenas imperdiet lacinia velit. Cras non urna. Morbi eros pede, suscipit ac, varius vel, egestas non, eros. Praesent malesuada, diam id pretium elementum, eros sem dictum tortor, vel consectetuer odio sem sed wisi.

3.1 Tests

Sed feugiat. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Ut pellentesque augue sed urna. Vestibulum diam eros, fringilla et, consectetuer eu, nonummy id, sapien. Nullam at lectus. In sagittis ultrices mauris. Curabitur malesuada erat sit amet massa. Fusce blandit. Aliquam erat volutpat. Aliquam euismod. Aenean vel lectus. Nunc imperdiet justo nec dolor.

3.2 Problèmes rencontrés

Sed feugiat. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Ut pellentesque augue sed urna. Vestibulum diam eros, fringilla et, consectetuer eu, nonummy id, sapien. Nullam at lectus. In sagittis ultrices mauris. Curabitur malesuada erat sit amet massa. Fusce blandit. Aliquam erat volutpat. Aliquam euismod. Aenean vel lectus. Nunc imperdiet justo nec dolor.

3.2.1 Estimateur de la distance

Etiam euismod. Fusce facilisis lacinia dui. Suspendisse potenti. In mi erat, cursus id, nonummy sed, ullamcorper eget, sapien. Praesent pretium, magna in eleifend egestas, pede

Tableau 1. Table of Grades

Name		
First name	Last Name	Grade
John	Doe	7.5
Richard	Miles	2

pede pretium lorem, quis consectetuer tortor sapien facilisis magna. Mauris quis magna varius nulla scelerisque imperdiet. Aliquam non quam. Aliquam porttitor quam a lacus. Praesent vel arcu ut tortor cursus volutpat. In vitae pede quis diam bibendum placerat. Fusce elementum convallis neque. Sed dolor orci, scelerisque ac, dapibus nec, ultricies ut, mi. Duis nec dui quis leo sagittis commodo.

3.2.2 HashCode

Etiam euismod. Fusce facilisis lacinia dui. Suspendisse potenti. In mi erat, cursus id, nonummy sed, ullamcorper eget, sapien. Praesent pretium, magna in eleifend egestas, pede pede pretium lorem, quis consectetuer tortor sapien facilisis magna. Mauris quis magna varius nulla scelerisque imperdiet. Aliquam non quam. Aliquam porttitor quam a lacus. Praesent vel arcu ut tortor cursus volutpat. In vitae pede quis diam bibendum placerat. Fusce elementum convallis neque. Sed dolor orci, scelerisque ac, dapibus nec, ultricies ut, mi. Duis nec dui quis leo sagittis commodo.

3.2.3 Opération du fichier

Etiam euismod. Fusce facilisis lacinia dui. Suspendisse potenti. In mi erat, cursus id, nonummy sed, ullamcorper eget, sapien. Praesent pretium, magna in eleifend egestas, pede pede pretium lorem, quis consectetuer tortor sapien facilisis magna. Mauris quis magna varius nulla scelerisque imperdiet. Aliquam non quam. Aliquam porttitor quam a lacus. Praesent vel arcu ut tortor cursus volutpat. In vitae pede quis diam bibendum placerat. Fusce elementum convallis neque. Sed dolor orci, scelerisque ac, dapibus nec, ultricies ut, mi. Duis nec dui quis leo sagittis commodo.

Word Definition

Concept Explanation

Idea Text

3.2.4 Pattern database

Aliquam lectus. Vivamus leo. Quisque ornare tellus ullamcorper nulla. Mauris porttitor pharetra tortor. Sed fringilla justo sed mauris. Mauris tellus. Sed non leo. Nullam elementum, magna in cursus sodales, augue est scelerisque sapien, venenatis congue nulla arcu et pede. Ut suscipit enim vel sapien. Donec congue. Maecenas urna mi, suscipit in, placerat ut, vestibulum ut, massa. Fusce ultrices nulla et nisl.

3.2.5 Taille de HashMap

Aliquam lectus. Vivamus leo. Quisque ornare tellus ullamcorper nulla. Mauris porttitor pharetra tortor. Sed fringilla justo

sed mauris. Mauris tellus. Sed non leo. Nullam elementum, magna in cursus sodales, augue est scelerisque sapien, venenatis congue nulla arcu et pede. Ut suscipit enim vel sapien. Donec congue. Maecenas urna mi, suscipit in, placerat ut, vestibulum ut, massa. Fusce ultrices nulla et nisl.

- First item in a list
- Second item in a list
- Third item in a list

3.3 Comparaison et conclusion

Plus un programme est lord, plus on sent l'importance des algorithmes et des structures de données. La résolution de Rubik's Cube, faisant partie de cette catégorie de programmes, nous a fait beaucoup réfléchi sur les algorithmes que nous utilisons afin d'aller plus loin dans la recherche de solution. Nous avons constaté au cours du temps qu'à chaque modification majeure du code, le programme devient plus rapide. Au début, le programme ne pouvait trouver que des solutions à 5 étapes avec une recherche naîve, puis le nombre d'étapes est passé à 7 même 8 en implémentant IDA*, à la fin ce record a été battu par une solution à 13 étapes en 42 secondes après avoir étudié plus de 900 000 cas possibles.

3.3.1 Algorithmes de recherche

Dans la section précédente, nous avons présenté les 3 algorithms testés. Pour se donner une idée de la performance de chaque algorithme, on pourra jeter un coup d'oeil sur le résultat d'un test de comparaison entre les différentes méthodes.

Profondeur	DFS[1] (PQ)[2]	DFS[1]	IDA*
1	0	0	0
2	0	0	0
3	0	1	0
4	17	7	0
5	387	246	0
6	(48865)[3]	(27989)[3]	0
7	-	-	0
8	-	-	2
9	-	-	7
10	-	-	87
11	-	-	1143
12	-	-	5270
13	-	-	(31488)[3]
14	-	-	-

1. DFS avec une profondeur limité
2. Ce parcours utilise une queue de priorité
3. Cette valeur est calculée avec moins de 5 essais

Tableau 2. Temps moyens de la recherche (algorithme)

On constate que l'utilisation de la queue de priorité n'a pas amélioré la vitesse de recherche, ce qui est dû principalement à la maintenance de la queue, car une recherche en profondeur avec la limite de profondeur qui s'incrément un par un ne se diffère pas vraiment d'un parcours en largeur. Dans ce cas-là, l'utilisation de la queue de priorité n'est pas rentable, parce qu'on finit toujours pas parcourir presque toutes les configurations. On pourrait penser à utiliser une queue de priorité afin d'implémenter un parcours A*, mais cette méthode ne donne pas un chemin minimal dans la plupart des cas. Comme une solution manuelle de Robik's Cube existe déjà, la seule question qui reste aujourd'hui est de savoir comment la résoudre d'une manière optimale.

Quant à l'algorithme IDA*, ce résultat n'est point surprenant. Pour une solution de moins de 6 étapes (ce qui correspond au niveau de notre Pattern Database, c'est-à-dire qu'il possède toutes les configurations d'une distance inférieure ou égale à 6), à chaque appel récursif, on est sûr de s'approcher de l'état final, ceci dit qu'on a toujours au moins un chemin pour faire diminuer la distance. Et comme la distance initiale est inférieure ou égale à 6, la solution sera trouvé immédiatement. Au-delà de 6 étapes, on constate qu'il y a une augmentation de temps comme dans les autres cas, cela vient du fait qu'on ne peut pas distinguer par exemple une distance 10 et une distance 7 à cause de la taille de notre base de données, cela brouille la piste de recherche et on est obligé d'étudier beaucoup de cas comme s'ils étaient de la même distance. On pourrait comprendre cet algorithme comme une translation (sur l'axe de profondeur) d'une distance imposée par le niveau de Pattern Database.

3.3.2 Estimateurs de distance

Pour utiliser l'algorithme IDA*, il nous faut une fonction heuristique ou un estimateur de distance. Les conditions nécessaires et suffisantes d'utilisation d'une telle fonction sont prouvées dans la section précédente. Ici, on va comparer de diverses fonctions et essayer de comprendre en quoi elles sont différentes.

Profondeur	Simple[1]	Manhattan[1]	Pattern[1]
3	0	3	0
4	2	5	0
5	20	26	0
6	223	135	0
7	1380	215	0
8	-	1118	1
9	-	-	13
10	-	-	79
11	-	-	1161
12	-	-	7309

1. Le calcul se trouve dans la section précédente

Tableau 3. Temps moyens de la recherche (estimateur)

3.3.3 Conclusion

Bien que le résultat précédent n'est pas suffisant pour résoudre un cube quelconque, il nous donne une piste de l'amélioration : agrandir Pattern Database et améliorer HashMap et HashCode. Plus la base de données est large, plus on choisit la bonne direction. Et puis le deuxième intervient lors de

Profondeur	Coins[1]	Arêtes 1[1]	Arêtes 2[1]
4	0	0	2
5	2	0	90
6	8	0	220
7	38	1	1256
8	123	13	-
9	220	144	-
10	1855	339	-
11	-	1108	-

1. Le calcul se trouve dans la section précédente

Tableau 4. Temps moyens de la recherche (Pattern)

l'éxécution du programme, car la taille de HashMap est limité par l'environnement de JVM et cette taille va déterminer la vitesse de la recherche. Rien n'interdit de résoudre complètement le Rubik's Cube, mais cela demandera une recherche encore plus poussée que les analyses dans ce rapport et surtout un ordinateur plus puissant.

Annexe

So long and thanks for all the fish [**?**].