Description générale du projet

*Linear algorithm for solution n-Queens Completion Problem*

Grigoryan E.

Le but de ce projet est la recherche et le développement d'un algorithme efficace pour résoudre le ***n-Queens Completion Problem***s. L'étude (avec différentes intensités et interruptions) a été menée au cours des années 2018-2019 et a duré environ un an et demi ane. Cette étude n'a pas reçu de soutien financier. Les résultats de l'étude sont publiés dans ***arxiv.org*** dans l'article: *Grigoryan E., Linear algorithm for solution n-Queens Completion problem,* [*https://arxiv.org/abs/1912.05935*](https://arxiv.org/abs/1912.05935).. La version russe de l'article est publiée sur le site Web de la communauté des programmeurs ***habr.com***: <https://habr.com/ru/post/483036/>

La tâche a une formulation mathématique simple. Il existe une composition de ***k*** reines qui sont régulièrement réparties sur un échiquier ***n × n***. Il est nécessaire de prouver que cette composition peut être complétée en une solution complète, et de donner au moins une solution, ou de prouver qu'une telle solution n'existe pas. Une composition de ***k*** reines est considérée comme cohérente si les trois conditions suivantes du problème sont remplies: dans chaque ligne, chaque colonne, ainsi que sur les diagonales gauche et droite passant par les cellules où se trouvent les reines, pas plus d'une reine n'est situé. Dans la formulation algorithmique de ce problème, au lieu de la phrase «il est nécessaire de prouver que ...», il sera écrit «il est nécessaire de développer un algorithme qui ...»

En 2017, dans un grand article: *C. Gent, I.-P. Jefferson and P. Nightingale. Complexity of n-queens completion. Journal of Artificial Intelligence Research, 59, 815–848, 2017*, <https://jair.org/index.php/jair/article/view/11079/26262>, il a été prouvé que ce problème appartient à l'ensemble ***NP-Complete*** . Il est supposé que la solution à ce problème peut ouvrir la voie à la résolution d'autres problèmes à partir de l'ensemble ***NP-Complete***.

**Vérification de l'exactitude du programme**

Tout programme (algorithme) a une propriété simple - il fonctionne ou non. Pour tester cela, il suffit de l'exécuter pour exécution. Pour vérifier cet algorithme, il est nécessaire de sélectionner la taille de l'échiquier (***n***), de former une composition aléatoire de taille ***k (1 <k <n)***, et de démarrer le programme d'achèvement. Par conséquent, le programme doit terminer la composition jusqu'à une solution complète, ou décider que la composition ne peut pas être terminée. Cependant, un arrangement de reines ne suffit pas. Afin de vérifier si la complexité temporelle du programme d'achèvement est linéaire, il est nécessaire:

- former un large échantillon de compositions aléatoires pour différentes valeurs de ***n***,

- exécuter le programme d'achèvement pour toutes les compositions de chaque échantillon,

- déterminer pour chaque échantillon le temps moyen d'achèvement.

Si nous divisons le temps moyen par n, nous obtenons le temps réduit, c'est-à-dire le temps moyen nécessaire pour positionner correctement la reine dans une position. Après avoir tracé la dépendance du temps réduit sur la taille de l'échiquier, nous pouvons établir la complexité temporelle de cet algorithme. Évidemment, la complexité temporelle de l'algorithme sera linéaire si le graphique est présenté comme une droite parallèle à l'axe des abscisses.

Pour tester l'algorithme, vous n'avez pas besoin d'écrire de code. En plus du programme principal:

*Completion\_k\_Queens\_Composition.m* – achèvement d'une composition arbitraire à la solution complète, ou décision que cette composition n'a pas de décision,

deux autres programmes ont été préparés:

*Solution\_n\_Queens\_Completion\_Problem.m* – - générer une composition aléatoire de taille ***k*** pour un échiquier arbitraire de taille ***n x n***,

*Validation\_n\_Queens\_Solution.m* - vérifier l'exactitude de la solution du ***n-Queens Problem***, ou l'exactitude de la composition de ***k*** reines.

Ils travaillent très vite. Par exemple, pour un échiquier de taille ***1000 x 1000*** cellules, le temps total, qui est en moyenne nécessaire pour:

- génération d'une composition arbitraire (***0,0015 s***.),

- compléter cette composition formée (***0,0622 s.***),

- et vérification de l'exactitude de la solution (***0,0003 s***.),

ne dépasse pas ***0,1*** seconde (à l'exclusion du temps nécessaire pour enregistrer les résultats ou télécharger les données).

Le code source des programmes est présenté avec des commentaires très détaillés et est conçu pour un large éventail d'utilisateurs. Si vous lirez l'article dans ***arxiv.org*** et les commentaires dans le code source du programme, et que cela n'a pas aidé à comprendre une partie du code, ou la logique de la procédure exécutable, écrivez-moi ([ericgrig@gmail.com](mailto:ericgrig@gmail.com)), je trouverai certainement l'occasion de répondre. Pour certaines questions qui peuvent survenir, j'ai préparé des réponses, et elles sont présentées dans le *Q & A\_**n-Queens Completion Problem.docx* . fichier. Toutes les nouvelles questions et réponses seront également publiées dans ce fichier.

L'algorithme ne dépend pas de la langue dans laquelle il est écrit. Le choix du langage de script ***Matlab*** est principalement lié à la commodité de la modélisation. Si vous traduisez de façon synchrone ce programme en ***C*** ou ***C ++***, cela fonctionnera plus rapidement. Je serai heureux si quelqu'un prend l'initiative et traduit ce programme dans d'autres langues. Pour ma part, je peux trouver une opportunité de conseiller.

La première version du programme a été écrite correctement sur le plan pédagogique, avec l'attribution de fonctions, où il y a des sections de code répétitives et logiquement connectées. Ensuite, l'algorithme a été amélioré et certaines parties du code ont changé, et en conséquence les parties du code qui lui étaient associées ont changé. Au stade initial, cela s'est répété assez souvent et, au fil du temps, ces fonctions ont «disparu» dans le programme. L'algorithme présenté ici est la cinquième version du programme et ne contient aucune fonction. Cependant, dans le texte à l'aspect «non armé», des sections de codes répétitifs sont visibles, ce qui, j'espère, sera correctement pris en compte lors de la traduction d'un programme dans un autre langage de programmation.

Je serai heureux si des sections du code, ou l'ensemble du programme dans son ensemble, seront utilisées à des fins scientifiques ou à des fins éducatives. Dans le même temps, je vous serais reconnaissant de bien vouloir vous référer à ma publication. C'est un élément de la culture et un signe de respect mutuel.

Pour la commodité des lecteurs, j'ai préparé toutes les informations connexes en trois langues: anglais, français, russe. Je serais profondément reconnaissant à toute personne capable de traduire correctement les informations et commentaires associés dans le code source dans d'autres langues.

Grigoryan Eros

  Octobre 2020, Marseille, France, [ericgrig@gmail.com](mailto:ericgrig@gmail.com)