NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

Análise e Desenho de Algoritmos

Projeto 2

Can Rob Escape?

Pedro Afonso – 63281

Manuel Gouveia – 64922

Maio 2024

Índice

Resolução do problema:

Para a resolução do problema “Can Rob Escape?”, adotámos uma estratégia algorítmica para garantir que Rob possa navegar seguramente através de um caminho repleto de lasers. A resolução para o problema é desenvolvida de forma a considerar tanto a posição dos raios laser como o tamaho de Rob, utilizando uma abordagem algorítmica para verificar se há caminho viável pelo corredor sem acontecerem colisões.

O código Java foi estruturado de forma modular, dividindo a lógica em duas classes que facilitam a manutenção e a compreensão dos componentes individuais do problema. A classe principal ‘**Main’** é responsável pela leitura dos dados de entrada (‘inputs’) e iniciar o processo de verificação utilizando a class **‘Solver’**. Esta executa a lógica central, verificando se Rob pode passar pelas coordenadas sem colidir com os lasers.

A classe **‘Solver’**, peça central desta solução, implementa os métodos para adicionar as coordenadas dos lasers e para verificar se é possível para Rob atravessar om corredor seguramente. Utiliza uma combinação de estruturas de dados como arrays e uma estrutura union-find para gerenciar e agrupar os lasers.

Lógica de Implementação:

- **Inicialização:** Ao iniciar a classe **‘Solver’**, esta configura os parâmetros básicos como a largura do corredor, o diâmetro de Rob e o número de lasers.
- **Adição de Lasers:** Cada laser adicionado é processado para determinar se está suficientemente próximo de outro, agrupando-os se necessário para facilitar a verificação de caminho livre. Esta etapa é crucial para a resolução do problema pois lasers próximos podem criar um conflito em relação à passagem de Rob, isto se a distância entre si for mais curta que o diâmetro de Rob.
- **Verificação de Caminho:** Após todos os lasers serem adicionados, o método **‘IsPossible()’** é chamado para verificar se existe uma faixa contínua livre de lasers que seja mais larga que o diâmetro de Rob, permitindo a sua passagem.

Após a conclusão da verificação de caminho, o programa responde imprimindo uma simples linha através da classe **‘Main’** que pode apenas corresponder a **‘Rob manages to escape!’**, caso tenho sido encontrado um caminho seguro para Rob passar entre os lasers, ou **‘Impossible to escape'**, caso contrário.

Complexidade Temporal:

A complexidade temporal do algoritmo é determinada principalmente pelo número de verificações necessárias para avaliar a possibilidade de Rob passar pelos lasers sem colidir. Cada laser adicionado ao sistema é comparado com todos os outros já existentes para determinar se estão próximos o suficiente para formar um grupo. Devido a esta necessidadde, este processo tem uma complexidade temporal aproximada de **O(B^2)**, onde B é o número de lasers.

Álem disso, a utilização da estrutura Union-Find para gerenciar os grupos de lasers, apesar de eficiente, opera com um tempo de execução quase constante, por operação **O(α(B))**, onde α é a função inversa de Ackermann. Portanto, as operações de união e busca não dominam a complexidade temporal, que é majoritariamente definida pela comparação par a par dos lasers, sendo assim **O(B^2)**.

Complexidade Espacial:

A complexidade espacial do algoritmo e influenciada principalmente pelo armazenamento de dados relacionados aos lasers e seus grupos. Cada laser tem as suas coordenadas armazenadas e, para além disso, estruturas são necessárias para o agrupamento e gerenciamento dos grupos de lasers.

Assim, para cada laser, armazena-se as suas coordenadas e o índice do seu grupo em uma estrutura de lista ligada, o que implica uma complexidade espacial de **O(B)**. Além disso, utilizamos a estrutur Union-Find para gerenciar os grupos, como esta é eficiente em manter grupos minimizados através da compressão de caminho, o espaço adicional permanece proporiconal ao número de lasers, **O(B)**

Conclusão:

O algortimo para o projeto “Can Rob Escape?” provou ser uma solução eficaz para determinar ser Rob consegue seguramente atravessar os lasers, utilizando uma estrutura Union-find e verificações sistemáticas para determinar rotas seguras. Este método demonstrou uma boa eficiência tanto em termos de complexidade temporal, com O(B^2), como espacial, com O(B), tornando-o adequado para diferentes cenários.

O equilíbrio entre a complexidade computacional e a eficácia operacional ressalta a durabilidade e praticidade do algoritmo, adequado para situações onde precisão e eficiência são cruciais. A estrutura bem fundamentada e a abordagem sistemática oferecem uma base sólida para futuras inovações e ajustes, podendo aprimorar ainda mais o desempenho e a aplicabilidade em cenários mais desafiadores.