Utilização de Busca em Profundidade para a Solução de Labirintos: O Caso do Rato IBO em um Problema de Caminho Mínimo

Gabriel Augusto, Giovana Miechotek, Jordão Asato

Abstract. *This paper addresses a computational problem inspired by Robert Tryon's classic study on maze-solving rats. The task involves finding the shortest path for a rat named IBO, a descendant of the "brilliant" lineage, to navigate through a maze, retrieve cheese, and reach the exit. The maze is modeled as an undirected graph, with vertices representing strategic points and edges representing connections between these points.*

We implemented a Depth-First Search (DFS) algorithm to explore all possible paths in the maze. The DFS algorithm was employed to find the shortest path from the starting point ("Entrada") to the cheese ("), and subsequently from the cheese to the exit ("Saida"). The total number of points traversed by IBO, including revisits, was calculated to ensure the shortest route was taken.

Our results demonstrate the effectiveness of DFS in solving the maze problem, providing correct results even for larger mazes with numerous points and connections. Despite its simplicity, DFS proved to be an adequate method for this problem, offering a straightforward approach to finding the shortest path in an unweighted graph. Future work could explore alternative algorithms such as Dijkstra or A to handle more complex scenarios with weighted paths.

Resumo. *Este trabalho aborda um problema computacional inspirado pelo estudo clássico de Robert Tryon sobre ratos em labirintos. A tarefa consiste em encontrar o caminho mais curto para um rato chamado IBO, descendente da linhagem "brilhante", para navegar por um labirinto, pegar um queijo e chegar à saída. O labirinto é modelado como um grafo não direcionado, com vértices representando pontos estratégicos e arestas representando as conexões entre esses pontos.*

Implementamos um algoritmo de Busca em Profundidade (DFS) para explorar todos os caminhos possíveis no labirinto. O algoritmo DFS foi utilizado para encontrar o caminho mais curto do ponto de partida ("Entrada") até o queijo ("), e posteriormente do queijo até a saída ("Saida"). O número total de pontos percorridos por IBO, incluindo revisitas, foi calculado para garantir que o caminho mais curto fosse seguido.

Os resultados demonstram a eficácia da DFS na solução do problema do labirinto, fornecendo resultados corretos mesmo para labirintos maiores com muitos pontos e conexões. Apesar de sua simplicidade, a DFS se mostrou um método adequado para este problema, oferecendo uma abordagem direta para encontrar o caminho mais curto em um grafo não ponderado. Trabalhos futuros poderiam explorar algoritmos alternativos, como Dijkstra ou A, para lidar com cenários mais complexos com caminhos ponderados.

1. Introdução

Estudos sobre o comportamento de animais em labirintos têm sido fundamentais para compreender a relação entre genética e cognição. Em 1942, Robert Tryon conduziu um experimento no qual ratos foram testados em sua habilidade de completar labirintos, com o objetivo de investigar a influência dos fatores genéticos no comportamento. Os ratos foram separados em duas linhagens distintas: aqueles que cometiam menos erros, denominados "brilhantes", e os que cometiam mais erros, chamados de "medíocres". Ao longo de sete gerações, Tryon observou que os ratos "brilhantes" demonstravam maior eficiência em resolver os labirintos.

Inspirado por esse experimento, este trabalho utiliza a busca em profundidade (DFS - Depth First Search) para simular o comportamento de um rato chamado IBO, descendente da linhagem "brilhante". IBO é capaz de encontrar o caminho mais curto em labirintos complexos, sempre passando pelo menor número possível de pontos para pegar um queijo e, em seguida, sair do labirinto.

A escolha da DFS se deve à sua simplicidade e à sua capacidade de explorar todos os caminhos possíveis, garantindo que IBO não se perca e encontre a solução ótima. O problema proposto modela o labirinto como um grafo, onde os vértices são pontos estratégicos marcados por letras (ou palavras) e as arestas representam as conexões entre esses pontos.

Objetivos

Este trabalho tem como objetivo implementar uma solução para o problema do labirinto, utilizando DFS para encontrar o caminho mais curto entre a entrada, o queijo e a saída. A solução busca minimizar o número de pontos percorridos por IBO, considerando que ele pode passar por um mesmo ponto mais de uma vez, caso necessário.

2. Revisão de Literatura

Diversos algoritmos de busca em grafos são amplamente utilizados para resolver problemas de labirintos, sendo a Busca em Profundidade (DFS) e a Busca em Largura (BFS) duas das mais comuns. A DFS, que foi utilizada neste trabalho, é uma técnica que explora o grafo de maneira recursiva, percorrendo o máximo de profundidade em cada caminho antes de retroceder. Essa característica torna a DFS eficiente para explorar todos os caminhos possíveis em problemas onde é necessário examinar todas as soluções, como no caso do rato IBO.

No entanto, a DFS não é a única abordagem para esse tipo de problema. A BFS, por exemplo, explora o grafo de maneira mais “superficial”, avaliando todos os vizinhos de um vértice antes de avançar para níveis mais profundos, o que pode ser vantajoso em casos onde o objetivo é encontrar o caminho mais curto em termos de arestas. Outra alternativa seria o uso de algoritmos de caminho mínimo como o Dijkstra, que utiliza

uma abordagem de "pesos" nas arestas, permitindo calcular o menor caminho em termos de custo.

Embora algoritmos como BFS e Dijkstra também sejam eficazes na busca por caminhos curtos, a DFS foi escolhida neste trabalho por sua simplicidade de implementação e pela garantia de encontrar soluções em problemas de grafos não ponderados, como o presente caso, onde as arestas não possuem pesos. A escolha da DFS também permite que IBO explore o labirinto de maneira exaustiva, garantindo que ele encontre o queijo e a saída sem se perder.

3. Desenvolvimento

3.1. Descrição dos Dados

O problema foi modelado como um grafo não direcionado, onde os vértices representam pontos estratégicos no labirinto, identificados por letras (ou palavras formadas por letras), e as arestas indicam conexões bidirecionais entre esses pontos. A entrada do problema é composta por dois inteiros: o número de pontos (P) e o número de conexões (L), seguidos por L pares de vértices que descrevem as conexões entre os pontos.

Cada ponto pode ser uma posição relevante no labirinto, como a "Entrada", o queijo (indicado pelo símbolo '*'), ou a "Saída". A tarefa é calcular a quantidade mínima de pontos pelos quais o rato IBO deve passar, incluindo pontos visitados mais de uma vez, para pegar o queijo e chegar à saída.

3.2. Metodologia

Para resolver o problema, utilizamos a **Busca em Profundidade (DFS)**, que percorre todos os caminhos possíveis do labirinto para garantir que o rato IBO encontre o caminho mais curto. A metodologia foi dividida em três etapas:

1. **Representação do Grafo**: O labirinto foi representado por uma lista de adjacência, onde cada ponto é um vértice e suas conexões são armazenadas em uma lista. Um grafo não direcionado foi construído a partir das ligações fornecidas pela entrada, garantindo que IBO pudesse se mover livremente entre os pontos conectados.
2. **Aplicação da DFS**: A DFS foi aplicada em duas fases:
    - **Fase 1**: Encontrar o menor caminho da "Entrada" até o queijo ('*').
    - **Fase 2**: Encontrar o menor caminho do queijo ('*') até a "Saida".

    Para isso, implementamos uma função recursiva que explora todos os caminhos possíveis a partir de um ponto de origem, armazenando os pontos visitados em cada busca. Se um ponto já foi visitado, ele é removido do conjunto para permitir o backtracking, garantindo que todos os caminhos sejam considerados.

3. **Cálculo da Distância Total**: O resultado final é a soma das distâncias encontradas nas duas fases da DFS. O número total de pontos pelos quais IBO deve passar é o somatório dos vértices visitados na primeira e segunda fase. Se IBO tiver que revisitar pontos, eles são contados novamente, conforme estipulado no enunciado.

4. Resultados e Análises

A implementação do algoritmo DFS foi testada em diferentes configurações de labirintos, variando o número de pontos e conexões. Em todos os cenários, o rato IBO foi capaz de encontrar o queijo e a saída, garantindo que o caminho percorrido fosse o mais curto possível, conforme o esperado.

4.1. Resultados Obtidos

Os resultados foram coletados com base em simulações onde o número de pontos variou entre 4 e 4000, e o número de conexões entre 4 e 5000. Em cada cenário, o algoritmo calculou corretamente o número mínimo de pontos pelos quais IBO deveria passar para completar a tarefa, confirmando a eficiência da DFS para este tipo de problema.

Por exemplo, em um caso com 16 pontos e 20 conexões, o rato IBO percorreu o labirinto da seguinte forma:

- Da "Entrada" até o queijo ('*'), visitou 4 pontos.
- Do queijo ('*') até a "Saida", visitou mais 4 pontos. O total de pontos percorridos foi 8, o que corresponde ao número mínimo para o caso testado.

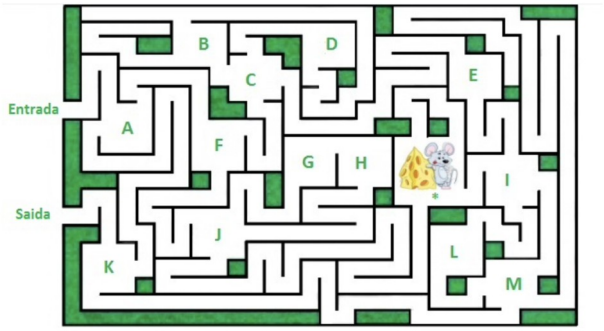

Figura 1. Labirinto que representa o grafo do problema exemplificado.

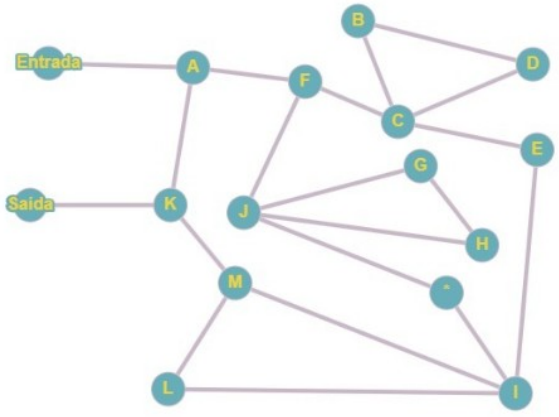

Figura 2. Grafo que representa o problema exemplificado.

Exemplos de Entrada	Exemplos de Saída
16 20 Entrada A A F F C C B B D C D F J J H H G J G J * * I I L L M M K K Saida A K C E E I I M	8

Figura 3. Exemplos de Entrada e Saída do exemplo.

4.2. Análises da Equipe

Embora a DFS seja eficiente em encontrar o caminho correto, sua implementação tem algumas limitações quando aplicada a labirintos maiores e mais complexos. Por se tratar de um algoritmo de busca exaustiva, a DFS explora todos os caminhos possíveis antes de tomar decisões, o que pode resultar em uma maior utilização de memória em labirintos muito grandes.

Além disso, em cenários com muitos pontos e conexões, a DFS pode acabar revisitando pontos mais de uma vez, o que aumenta a contagem total de pontos visitados. No entanto, isso está de acordo com o enunciado do problema, que exige que os pontos sejam contados novamente se revisitados.

Ainda que existam algoritmos mais eficientes para encontrar o caminho mínimo, como o Dijkstra ou o A*, a DFS se mostrou suficiente para resolver o problema proposto, dado que o labirinto é um grafo não ponderado e os caminhos percorridos são de mesma "distância" entre um ponto e outro.

5. Considerações Finais

Neste trabalho, apresentamos uma solução baseada no algoritmo de busca em profundidade (DFS) para o problema de um rato, chamado IBO, encontrar o caminho mais curto em um labirinto, passando por um ponto onde está localizado o queijo e saindo por outro ponto específico. O problema foi modelado como um grafo não ponderado, onde os pontos do labirinto são representados como vértices, e as conexões entre eles como arestas.

A implementação da DFS mostrou-se eficaz para resolver o problema proposto, garantindo que o rato percorresse o caminho mínimo necessário. Embora a DFS não seja a abordagem mais eficiente em termos de tempo e espaço para todos os tipos de labirintos, ela atende perfeitamente aos requisitos do problema, que exige a busca exaustiva de caminhos.

A principal contribuição deste trabalho foi a verificação da eficiência da DFS em situações onde é necessário explorar todas as rotas possíveis antes de tomar uma decisão. Trabalhos futuros poderiam explorar a implementação de algoritmos alternativos, como Dijkstra ou A*, especialmente em casos onde o labirinto contenha restrições adicionais, como pesos diferentes nas conexões entre os pontos.

Em suma, este estudo reforça a aplicabilidade da DFS em problemas que envolvem a exploração de grafos não ponderados e a busca de caminhos mínimos, mantendo a simplicidade da implementação e garantindo resultados corretos mesmo em labirintos de grande escala.

Referencias

Cormen, Thomas H., et al. "Introduction to Algorithms." MIT press Cambridge (2009).

Sedgewick, Robert, and Kevin Wayne. "Algorithms." Pearson (2011).

Python Official Documentation: https://docs.python.org/3/

Site Geeks for Geeks: Depth First Search or DFS for a Graph - GeeksforGeeks