"Exploração de Algoritmos de Busca na Resolução de Labirintos: Análise Comparativa e Implementação Interativa"

Resumo

Este trabalho aborda a resolução de labirintos, um dos problemas clássicos da ciência da computação, que envolve desafios relacionados à busca, otimização e navegação. Por meio da implementação de três abordagens algorítmicas distintas, o projeto analisa diferentes estratégias para solucionar o problema, avaliando seus resultados sob os critérios de eficiência e complexidade. O desenvolvimento foi realizado utilizando a linguagem Python, com suporte do framework Pygame para visualização, o que proporcionou uma experiência interativa e didática tanto para os desenvolvedores quanto para os usuários.

No projeto, o labirinto é representado como uma matriz bidimensional, onde cada célula possui um estado específico. Essas células podem ser paredes, que impedem o movimento, ou caminhos transitáveis, que permitem a progressão do agente. Essa estrutura facilita a modelagem de problemas reais de navegação, como o planejamento de rotas em mapas urbanos ou o controle de movimentação em sistemas robóticos. Para resolver o labirinto, o agente inicia em uma posição específica e tem como objetivo alcançar outro ponto previamente definido. O papel do programa é encontrar um trajeto válido entre os dois pontos, utilizando diferentes algoritmos de busca.

Motivação e Objetivo

A principal motivação para a realização deste projeto foi a aplicação prática e visual de algoritmos clássicos de busca, permitindo uma compreensão mais clara e intuitiva de seus comportamentos e características. Através de uma interface gráfica desenvolvida com Pygame, o funcionamento de cada algoritmo pode ser observado em tempo real, desde o percurso do agente até os nós visitados e o trajeto final. Esse aspecto visual transforma o projeto em uma ferramenta educacional eficaz e uma base sólida para estudos avançados em inteligência artificial e robótica.

O objetivo central do trabalho é implementar e comparar o desempenho de três algoritmos de busca amplamente utilizados: a **Busca em Largura (BFS)**, a **Busca em Profundidade (DFS)** e a **Busca Heurística A***. Esses algoritmos foram

escolhidos devido à sua relevância em problemas de navegação e pela diversidade de abordagens que representam. Além disso, o projeto explora o impacto do uso de heurísticas na eficiência das buscas, destacando a importância de estratégias inteligentes em cenários mais complexos.

Descrição Detalhada dos Métodos

Os três algoritmos de busca implementados apresentam características distintas que influenciam diretamente a maneira como o agente navega e resolve o labirinto.

- **Busca em Largura (BFS)**
  A BFS é um algoritmo sistemático que explora todos os caminhos em um nível antes de avançar para o próximo. Essa abordagem garante que o primeiro trajeto encontrado seja sempre o mais curto em termos de número de passos. Para isso, utiliza-se uma fila para gerenciar os nós a serem visitados, o que assegura uma exploração ordenada e metódica.
  Apesar de sua eficácia em encontrar trajetos curtos, a BFS apresenta como limitação um alto consumo de memória em labirintos maiores, já que é necessário armazenar todos os nós de um nível simultaneamente. No contexto do projeto, o algoritmo demonstrou bom desempenho em encontrar soluções curtas, mas frequentemente explorou áreas do labirinto que não contribuíram diretamente para o trajeto final, evidenciado pelo grande número de células visitadas antes de alcançar o objetivo.
- **Busca em Profundidade (DFS)**
  Diferentemente da BFS, o algoritmo DFS prioriza a profundidade sobre a largura. Usando uma pilha para armazenar os nós, ele tenta avançar ao máximo por um único caminho antes de retroceder para explorar alternativas. Essa abordagem é mais rápida em certos cenários, especialmente quando o objetivo está localizado em trajetos mais profundos, mas não garante encontrar o caminho mais curto.
  Nos testes realizados, a DFS apresentou resultados variados. Em labirintos simples, foi capaz de encontrar soluções rapidamente. Contudo, em cenários mais complexos, percorreu trajetos desnecessários, o que resultou em caminhos mais longos e menos eficientes. A visualização gráfica foi

fundamental para evidenciar esse comportamento, mostrando o agente navegando por trechos extensos antes de retornar ao caminho correto.

- **Busca Heurística A***
  O algoritmo A* combina o custo acumulado de um trajeto com uma estimativa da distância restante até o objetivo, utilizando uma função heurística para orientar a busca. No projeto, a heurística utilizada foi baseada na distância mínima calculada pelo BFS, o que ajudou a equilibrar a exploração com a eficiência, priorizando caminhos mais promissores.
  Entre os algoritmos implementados, o A* foi o mais eficiente tanto em termos de células visitadas quanto de trajeto final encontrado. Sua implementação utilizou uma fila de prioridade para organizar os nós a serem explorados, classificando-os de acordo com o custo total (soma do custo acumulado e da heurística). Os resultados confirmaram a superioridade do A* em cenários complexos, onde foi capaz de encontrar soluções rápidas e precisas.

Funcionalidades Extras e Interatividade

Além da implementação básica dos algoritmos, o projeto incorporou funcionalidades adicionais para enriquecer a experiência do usuário e aprimorar a precisão dos resultados. Uma das principais funcionalidades foi a heurística dinâmica, que calcula em tempo real a distância mínima entre duas células, otimizando a eficiência do A*, especialmente em labirintos maiores.

Outro destaque foi a interatividade proporcionada pelo programa. Durante a execução, o usuário pode alternar entre algoritmos pressionando teclas específicas(1 - BFS, 2 - DFS e 3 - A*), observando em tempo real as diferenças no comportamento do agente. Cores distintas são utilizadas para indicar paredes, caminhos, células visitadas e o trajeto final, tornando o processo mais intuitivo e visualmente acessível.

Análise de Desempenho e Comparação

A análise dos resultados revelou diferenças fundamentais entre os algoritmos. O BFS destacou-se pela precisão ao encontrar trajetos curtos, mas sua exploração ampla de células reduziu sua eficiência em labirintos grandes. A DFS, por ser uma abordagem mais simples, mostrou-se útil para encontrar soluções válidas rapidamente, mas frequentemente percorreu caminhos mais longos e ineficientes. O

A*, por sua vez, combinou rapidez e precisão, consumindo menos recursos e entregando resultados superiores, especialmente em cenários mais complexos.

Essa comparação evidencia a importância de escolher o algoritmo adequado para cada contexto. Em aplicações reais, como navegação de robôs ou planejamento de rotas urbanas, a seleção do método pode impactar significativamente o desempenho geral do sistema.

Considerações Finais

Este projeto demonstrou a aplicação prática de algoritmos clássicos de busca em um ambiente visual e interativo, contribuindo para o estudo e análise dessas estratégias. A implementação dos algoritmos BFS, DFS e A* possibilitou uma exploração detalhada de diferentes abordagens, evidenciando suas vantagens e limitações.

Além disso, o projeto abre caminhos para futuras extensões, como a inclusão de algoritmos mais avançados, como o de Dijkstra ou a busca em profundidade iterativa, além da geração procedural de labirintos e métricas mais detalhadas de desempenho. Em resumo, a iniciativa reforça a importância de estratégias inteligentes na solução de problemas práticos e destaca o papel dos algoritmos na tomada de decisões em cenários complexos.

Murilo Russo - 8161