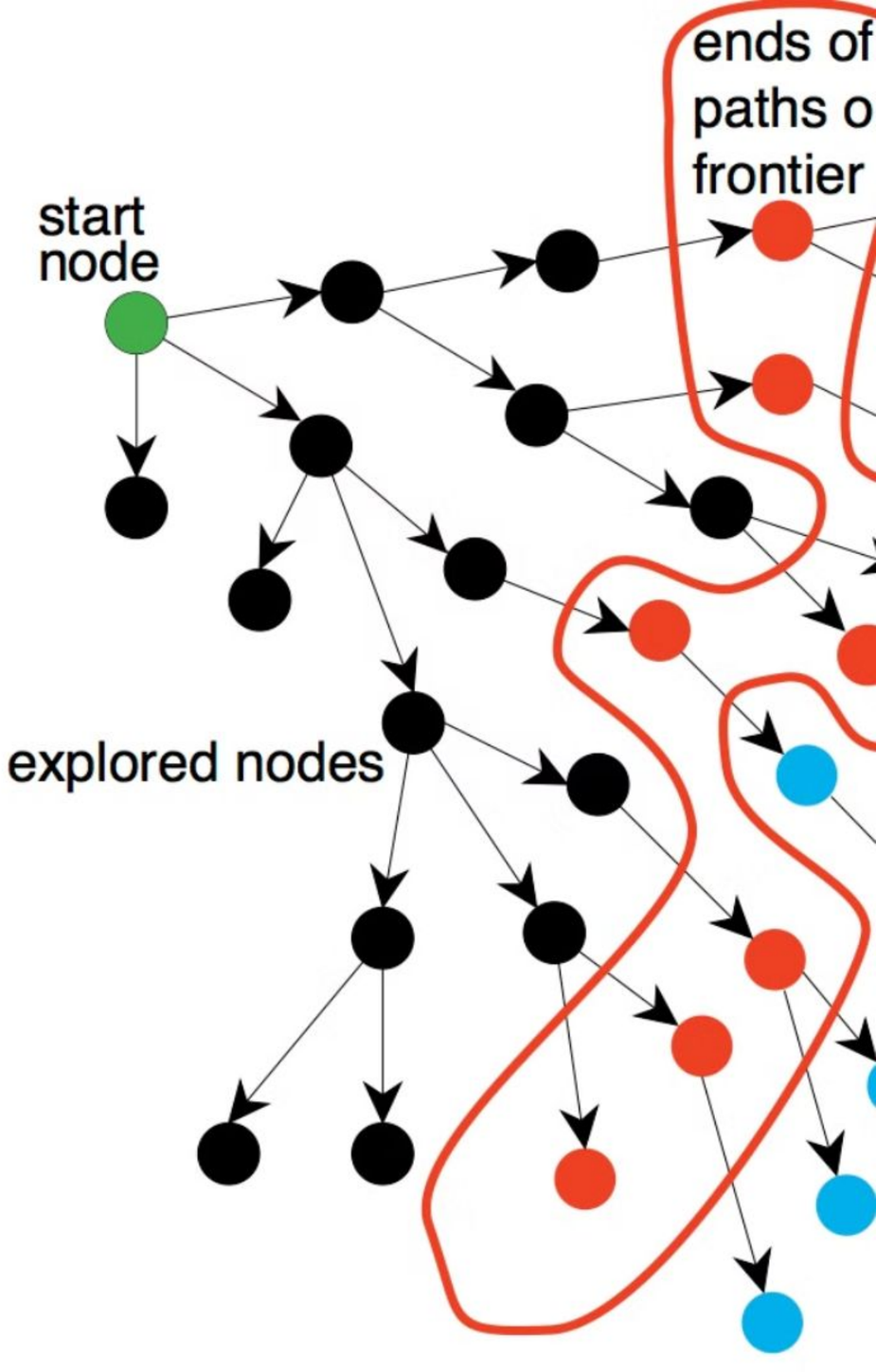
Introdução aos Algoritmos de Busca de Caminho

Um tópico fundamental em inteligência artificial que encontra aplicação em áreas como planejamento de rotas, jogos, robótica, entre outras. Dois algoritmos importantes: a Busca Gulosa (Greedy) e o A-Estrela (A*).

Prof. ME. Pablo De Chiaro Rosa

insta @chiarorosa



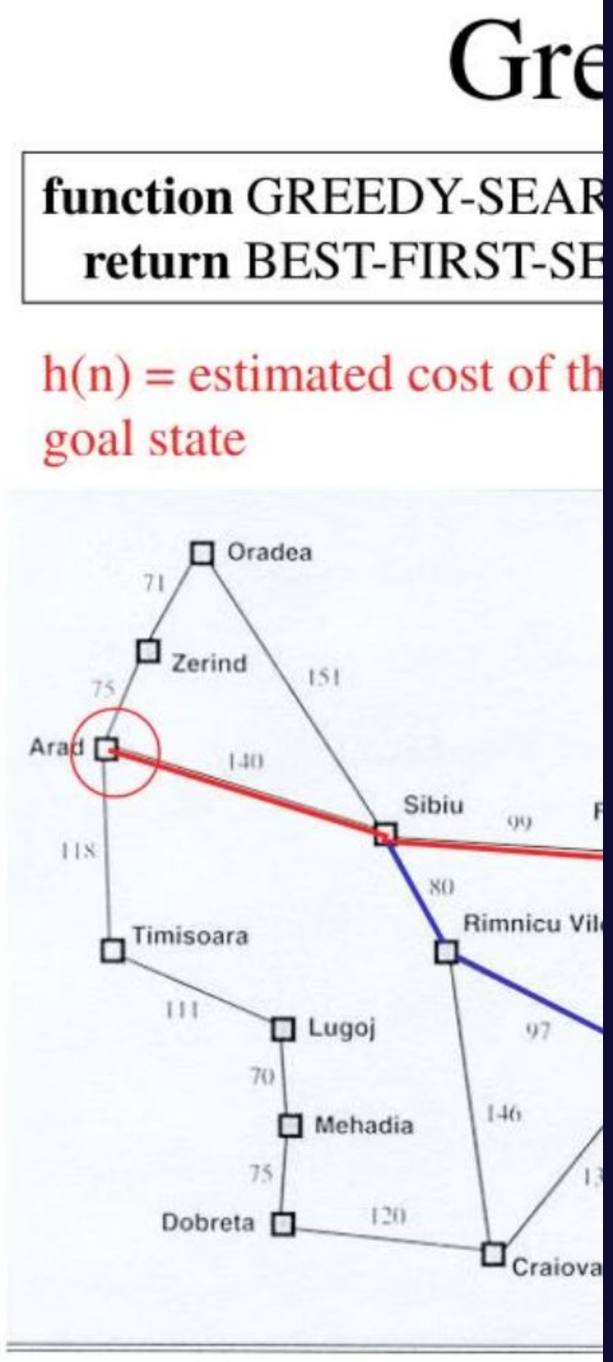
Fundamentos dos Algoritmos de Busca de Caminho

Problema a Resolver

Algoritmos de busca de caminho são projetados para resolver o problema de encontrar o caminho mais curto ou ótimo entre dois pontos. Eles diferem na forma como avaliam os caminhos, decidem qual caminho seguir e determinam a ordem de exploração dos nós (ou estados).

Princípio Fundamental

O princípio fundamental desses algoritmos baseia-se na exploração do espaço de estados, onde cada estado representa uma possível configuração do problema, e as transições entre estados representam movimentos ou ações.



Busca Gulosa (Greedy)

1 Princípio da Busca

A Busca Gulosa é um método de busca que faz escolhas que parecem ser as melhores no momento, ou seja, seleciona o nó que parece estar mais próximo do objetivo, sem considerar o custo do caminho percorrido até aquele ponto.

2 Vantagens e Desvantagens

Vantagens: Simplicidade e velocidade em alguns casos.

Desvantagens: Não garante a solução ótima, pois pode ficar preso em mínimos locais.

A-Estrela (A*)

1

Melhoria da Busca Gulosa

O A-Estrela é uma melhoria da Busca Gulosa, considerando tanto o custo do caminho percorrido até o momento quanto a estimativa heurística para o objetivo.

2

Vantagens e Desvantagens

Vantagens: Eficiência e garantia de encontrar a solução ótima sob certas condições. Desvantagens: Pode exigir muita memória se o espaço de busca for grande.

Exemplos Práticos

1

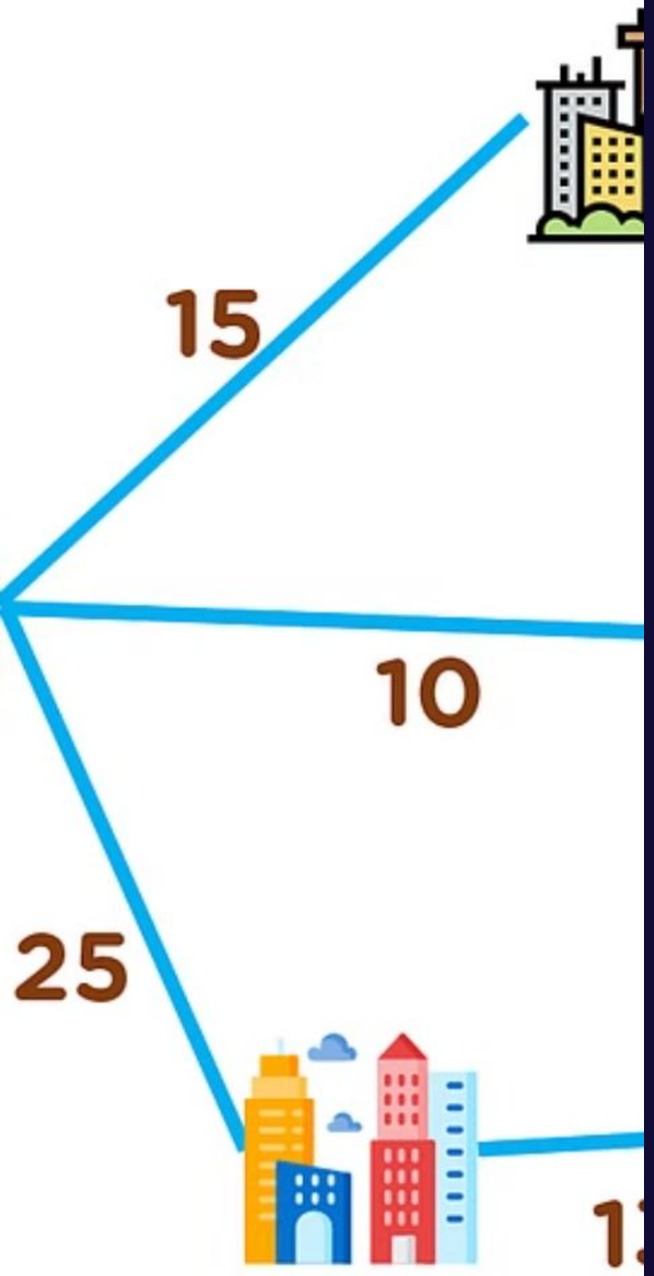
2

Exemplo de Busca Gulosa

Problema: Imagine que você está em uma cidade A e quer viajar para a cidade B. Há várias rotas possíveis, passando por diferentes cidades.

Exemplo de A-Estrela (A*)

Problema: Usando o mesmo cenário da viagem de A para B, mas aplicando o A-Estrela.



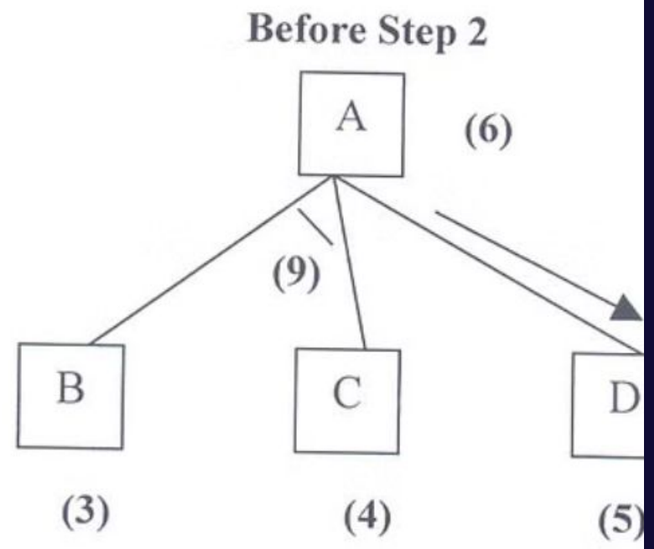
Implementação do Algoritmo de Busca Gulosa

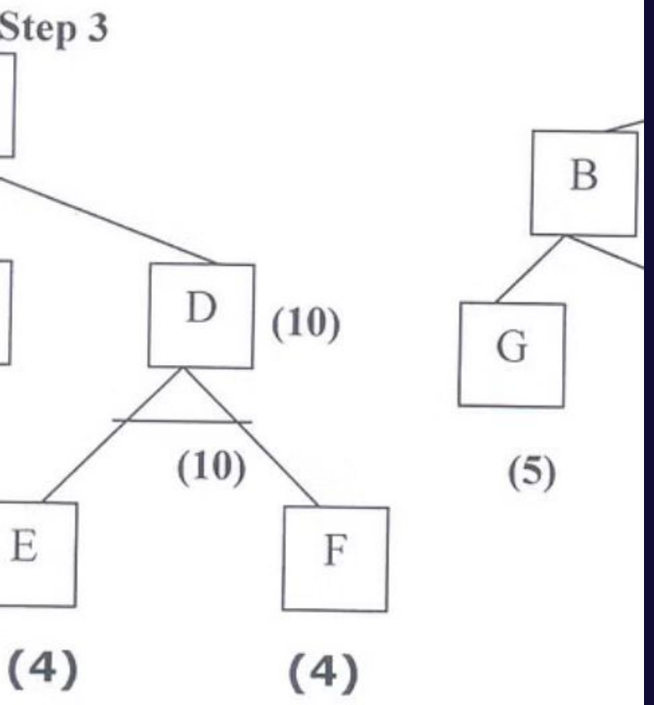
Representação do Problema

Cidades, Conexões entre cidades, Heurística.

Algoritmo em Python

Implementação do algoritmo de busca gulosa em Python para o problema de encontrar um caminho de uma cidade a outra, considerando um mapa com 10 cidades.





Implementação do Algoritmo A-Estrela (A*)

Implementação do A-Estrela

Supondo que as conexões e a heurística são as mesmas definidas anteriormente.

Explicações

Este algoritmo considera tanto o custo já percorrido quanto a estimativa até o destino, escolhendo o caminho que minimiza ambos.

HANDS ON

Busca do melhor caminho

Representação do Problema

- Cidades: Serão representadas por letras (A, B, C, ..., J).
- Conexões entre cidades: Um dicionário em Python onde cada chave é uma cidade, e o valor é uma lista de tuplas representando as cidades conectadas e a distância até elas.
- Heurística: Um dicionário onde cada chave é uma cidade, e o valor é a estimativa heurística da distância até a cidade destino (B, neste caso).

Comparação dos Resultados

Algoritmo	Caminho Escolhido	Custo Total
Busca Gulosa	A -> C -> F -> B	11
A-Estrela (A*)	A -> D -> E -> B	9

Detalhes dos Custos

Busca Gulosa

Caminho: A -> C -> F -> B. Detalhes dos

Custos: A -> C: 2, C -> F: 6, F -> B: 3. Custo

Total: 11.

A-Estrela (A*)

Caminho: A -> D -> E -> B. Detalhes dos

Custos: A -> D: 3, D -> E: 1, E -> B: 5. Custo

Total: 9.

Conclusão

A comparação ilustra a importância de considerar ambos os aspectos (custo percorrido e estimativa até o destino) ao buscar o caminho mais eficiente, especialmente em problemas complexos de roteamento e planejamento.