Aula 16: Grafos – Problema do Menor Caminho

(Algoritmo: Dijkstra)

DCC405-Estrutura de Dados II

Prof. Me. Acauan C. Ribeiro

Fazer exemplo google maps

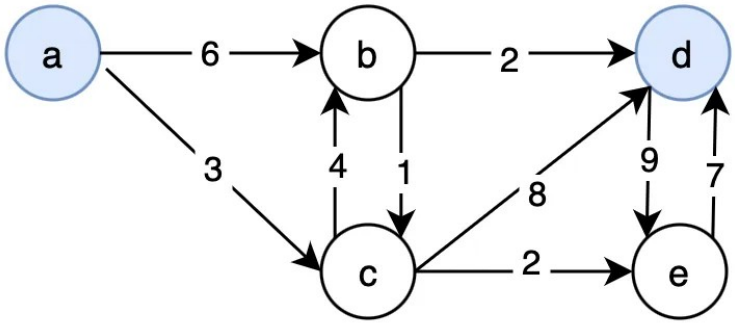
- Menor caminho entre dois pontos é uma reta?
- Roteiro:
  - Ver vídeo: How Dijkstra's Algorithm Works.mp4
  - Abrir: https://medium.com/programadores-ajudando-programadores/os-grafos-e-os-algoritmos-697c1fd4a416
  - Implementar: Antes ver: https://realpython.com/iterate-through-dictionary-python/#iterating-through-keys-directly
  -

Definição

- Um algoritmo de busca é um algoritmo para recuperar informações armazenadas dentro de alguma estrutura de dados, ou calculadas no espaço de busca de um domínio de problema [1]. Diferentemente da pesquisa em profundidade (DFS) e da pesquisa em largura (BFS), o **algoritmo de Dijkstra** e a **pesquisa de custo uniforme (UCS)** consideram o custo do caminho até o objetivo.
- Por exemplo, em uma rede rodoviária, o custo do caminho pode ser a distância total percorrida ou o tempo total gasto. O algoritmo de Dijkstra é um **algoritmo que encontra o caminho mais curto de um nó para todos os outros nós no grafo**, enquanto o **UCS encontra o caminho mais curto entre 2 nós**. Agora, vamos explicar o algoritmo UCS, uma variante do algoritmo de Dijkstra, com mais detalhes.

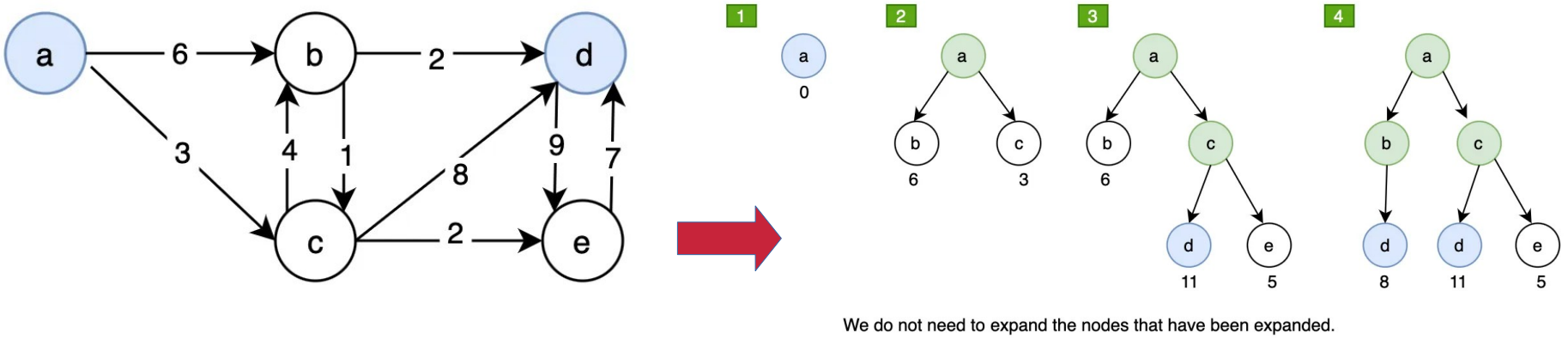
Uniform-Cost Search

- O UCS **expande o nó com menor custo de caminho g até o nó de inicio**. UCS é a modificação do BFS. Em vez de usar a fila First-In-First-Out, ele usa uma fila prioritária com custo de caminho g(n) para ordenar os nós. [3] Vejamos um exemplo com o seguinte Grafo direcionado, o nó inicial **a** e o nó final **d** estão sendo destacados em azul:



Uniform-Cost Search

- O UCS **expande o nó com menor custo de caminho g até o nó de inicio**. UCS é a modificação do BFS. Em vez de usar a fila First-In-First-Out, ele usa uma fila prioritária com custo de caminho g(n) para ordenar os nós. [3] Vejamos um exemplo com o seguinte Grafo direcionado, o nó inicial **a** e o nó final **d** estão sendo destacados em azul:



Uniform-Cost Search

- A partir da expansão passo a passo, pudemos ver que o custo do caminho está sendo levado em consideração e ele expande o nó com o menor custo do caminho. Por exemplo, da etapa 2 para a etapa 3, ele expande o nó c que possui o menor custo de caminho até o momento. Além disso, na etapa 3, mesmo que tenha alcançado o nó de destino d com um custo total de 11 após a expansão, como o custo atual do nó b é $6 < 11$, ele continuará expandindo o nó b. Depois de expandir o nó b na etapa 4, ele também atinge o nó de destino d com um custo total de 8 e não há mais nós para expandir agora. Então, o caminho mais curto de a até d $= [a \rightarrow b \rightarrow d]$ com custo total $= 8$.

Uniform-Cost Search

Depois de entender como o UCS funciona, o algoritmo de **Dijkstra** é apenas um algoritmo que encontra o caminho mais curto para cada ponto em vez de um único ponto.

Agora, vamos avaliar este algoritmo:

1. Complexidade de tempo: quanto tempo leva para encontrar uma solução?

Número de nós com custo de caminho g = custo da solução ótima

- Equivalente ao número de nós que saem da fila de prioridade

2. Complexidade do espaço: número máximo de nós na memória

Número de nós com custo de caminho g = custo da solução ótima

3. Completude: sempre encontra uma solução, se houver?

Sim

4.Otimalidade: sempre encontra a melhor solução (de menor custo)?

Sim