Busca Heurística e Planejamento

Algoritmos

Os algoritmos de busca heurística utilizam funções heurísticas para guiar a exploração do espaço de estados e encontrar soluções de forma mais eficiente.

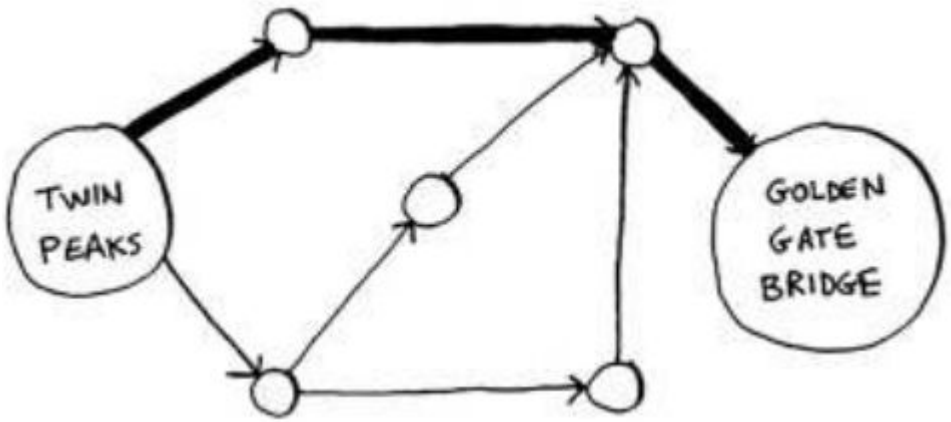
Heurística

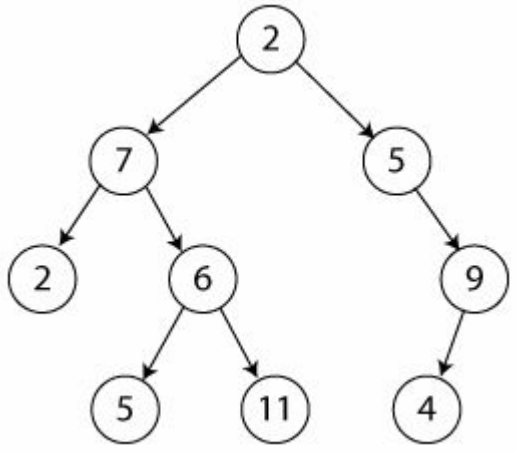
1. Arte de inventar ou descobrir.
2. Método que pretende levar a inventar, descobrir ou a resolver problemas.

Heurística é um conceito fundamental em Inteligência Artificial (IA) e ciência da computação, referindo-se a uma técnica ou método prático que ajuda a resolver problemas de forma mais eficiente, mesmo que não garanta a solução ótima. Em algoritmos de IA, heurísticas são usadas para reduzir o tempo de busca ou tomar decisões mais rápidas em ambientes complexos, onde uma abordagem exata seria computacionalmente inviável.

Espaço

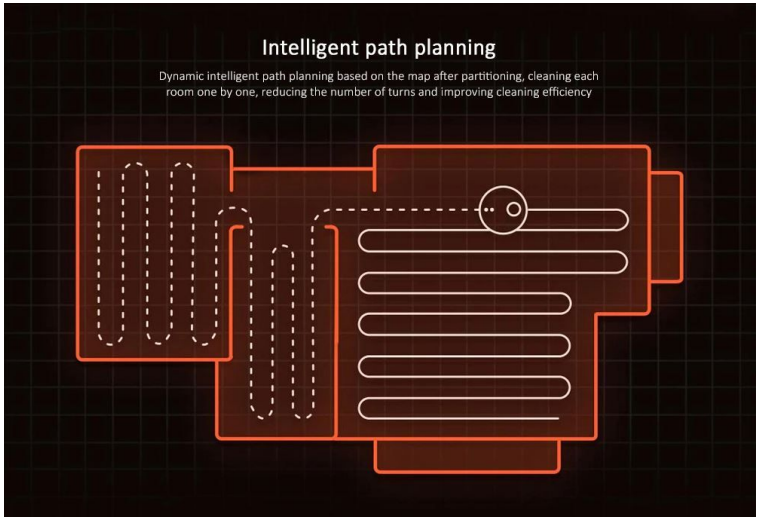
O espaço pode se referir a uma mapa, caminho, labirinto, entre outros. Muitas vezes o espaço poderá ser representado como uma matriz ou um grafo.





Aplicação

- Logística.
- Robótica.
- Games.



Algoritmos de busca em grafo

1. Algoritmos de Busca Não Informada (Cega)

Não utilizam conhecimento adicional sobre o problema, apenas a estrutura do grafo.

- Busca em Largura (BFS - Breadth-First Search)
- Busca em Profundidade (DFS - Depth-First Search)
- Busca em Profundidade Limitada (DLS - Depth-Limited Search)

2. Algoritmos de Busca Informada (Heurística)

Utilizam funções heurísticas para estimar o custo até o objetivo, direcionando a busca.

- Busca Gulosa (Greedy Best-First Search)
- Algoritmo A* (A Estrela - A-Star Search)

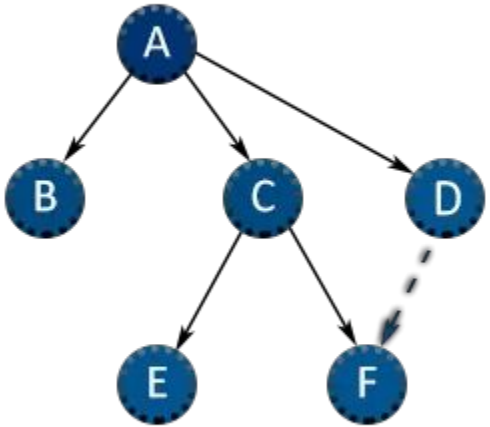
3. Algoritmos para Grafos com Restrições ou Dinâmicos

- **Dijkstra** (para grafos com pesos positivos)
- **Bellman-Ford** (para grafos com pesos negativos)

Métodos de Busca em Grafos

Busca em Largura
Breadth-First Search

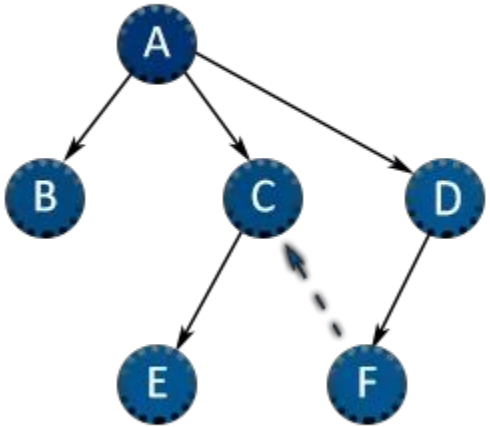
BFS



A B C D E F

Busca em Profundidade
Depth-First Search

DFS



A D F C E B

Busca Gulosa

1 - Expande o nó mais próximo do objetivo (usando heurística h(n)), sendo h(n) o menor caminho.

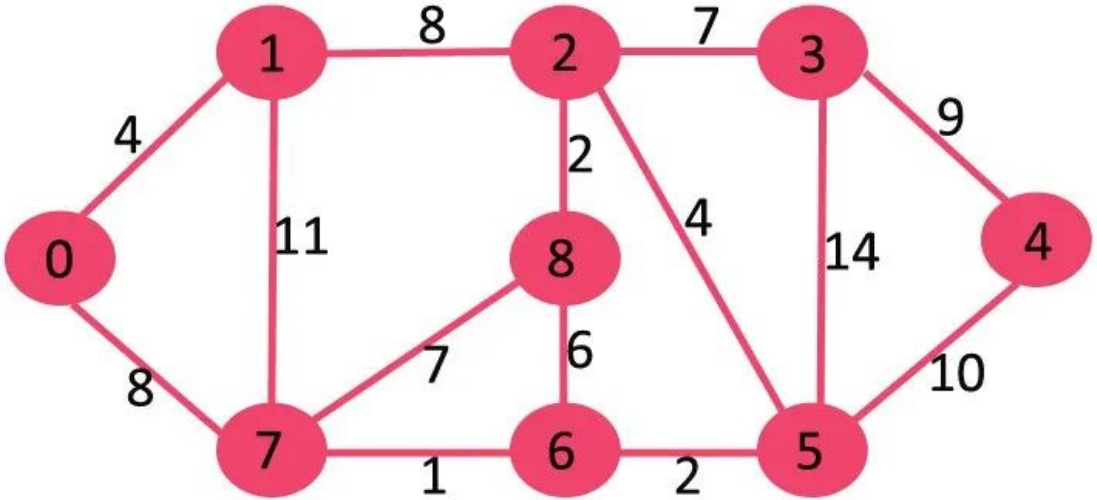
- Não garante otimalidade, mas é eficiente em muitos casos.
- Pode ficar preso em mínimos locais.

Caminho de 0 a 8:

- 0 →1 →2 →8
- (4+8+2=14).

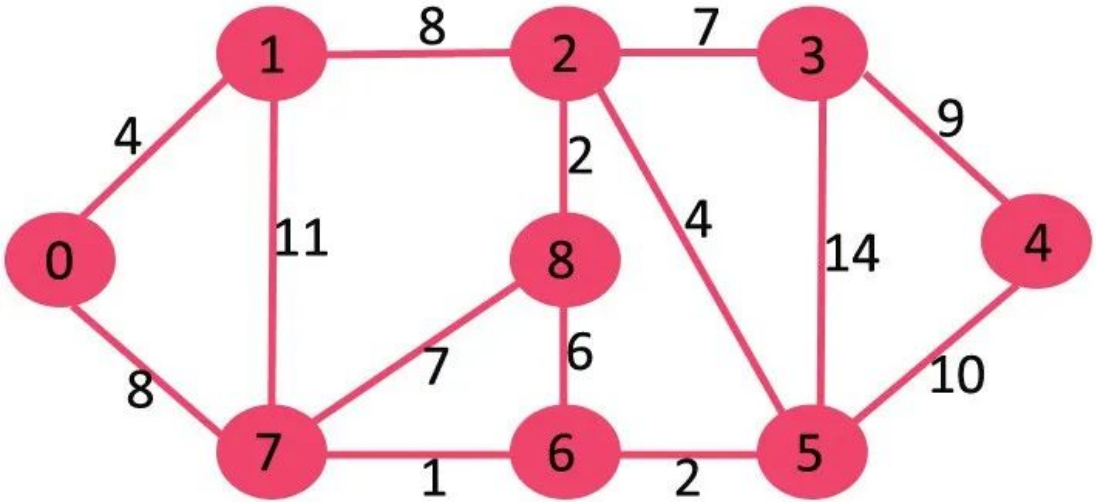
Caminho de 7 a 8:

- 7 →6 →5 →2 →8
- (1+2+4+2=9)



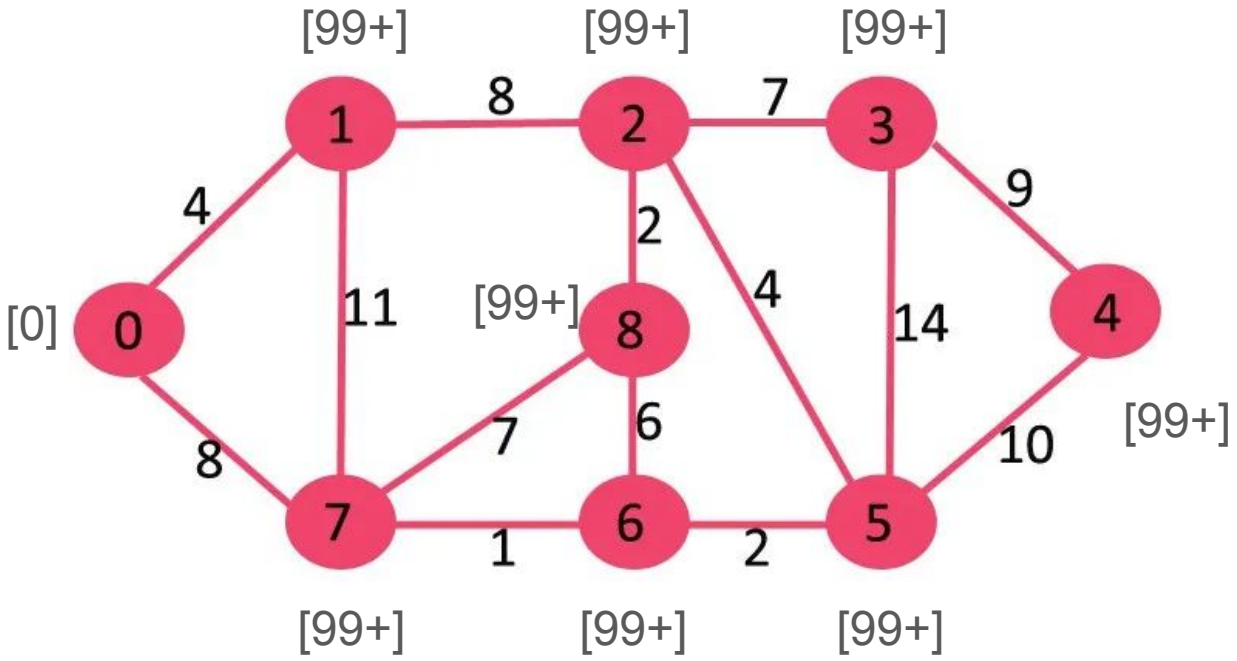
Algoritmo de Dijkstra

O algoritmo de Dijkstra foi criado pelo matemático holandês Edsger W. Dijkstra em 1956. Ele foi desenvolvido originalmente para resolver problemas de roteamento em redes de telecomunicações, mas rapidamente se tornou uma ferramenta importante em muitas outras áreas, como em sistemas de navegação, logística e em diversas outras áreas.



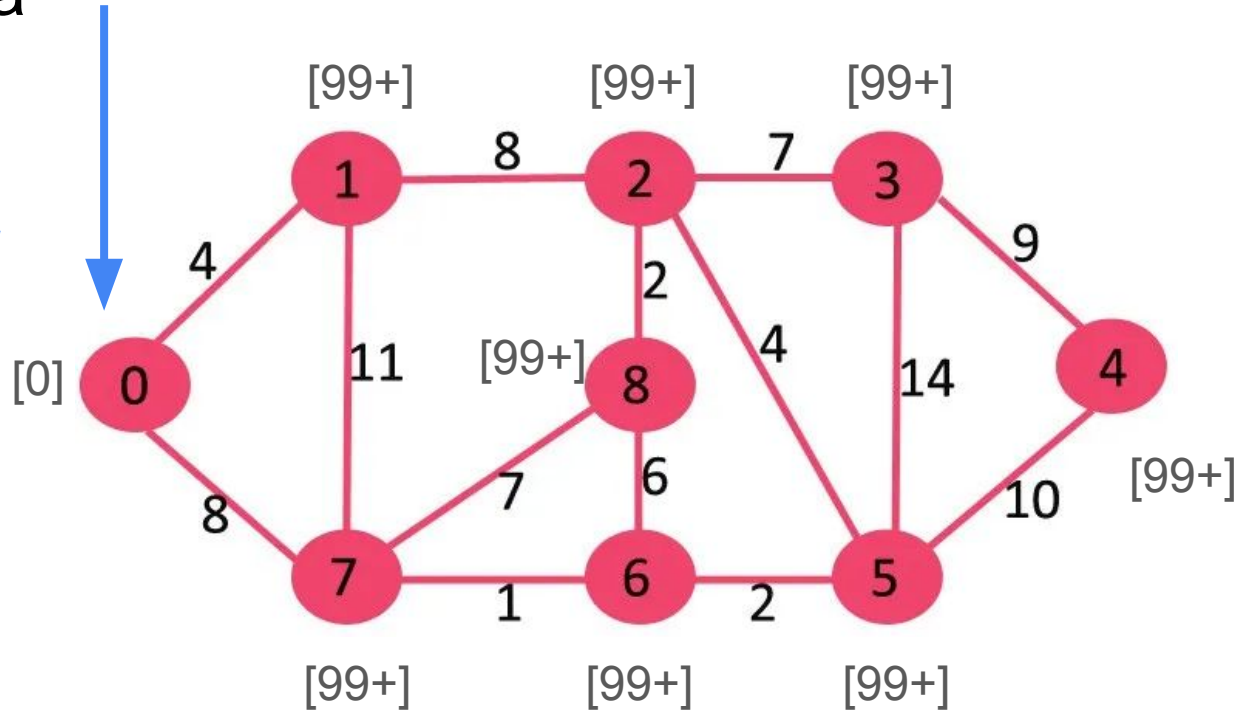
Algoritmo de Dijkstra

1. **Inicializar custos.**
2. Escolher o vértice não visitado com menor distância atual.
3. Atualizar a distância dos vizinhos.
4. Escolher um novo vértice não visitado para ser o vértice atual.
5. Repetir passos 2, 3 e 4 até não existirem vértices não visitados.



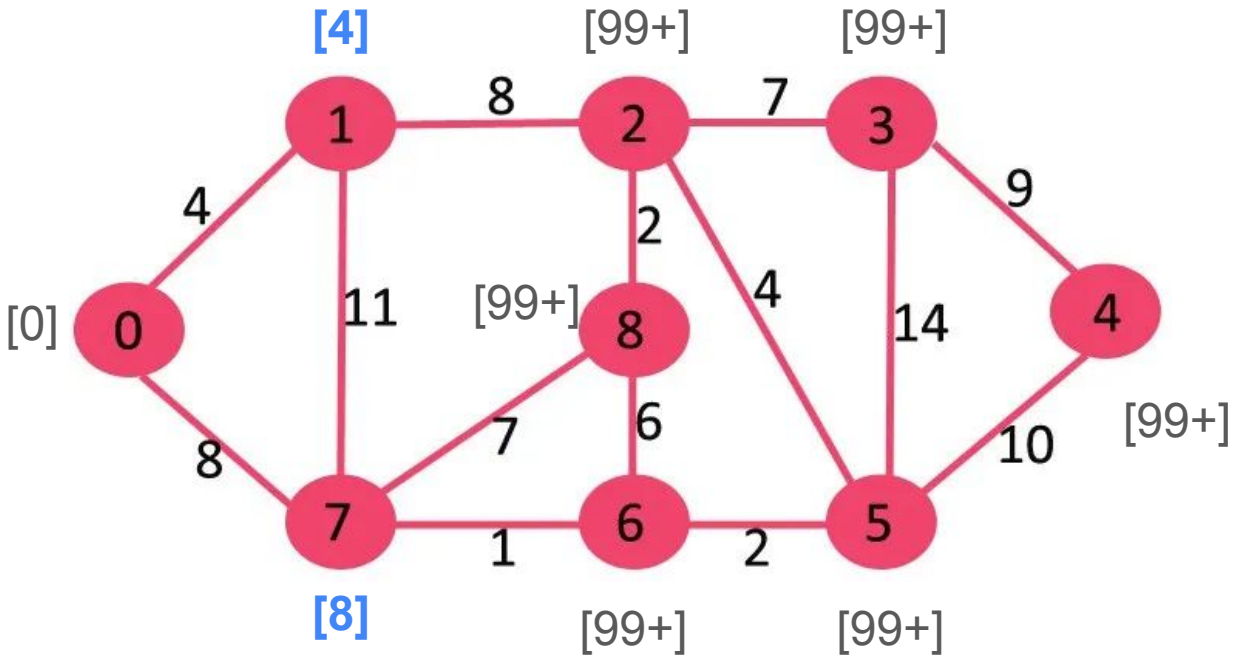
Algoritmo de Dijkstra

1. Inicializar custos.
2. **Escolher o vértice não visitado com menor distância atual.**
3. Atualizar a distância dos vizinhos.
4. Escolher um novo vértice não visitado para ser o vértice atual.
5. Repetir passos 2, 3 e 4 até não existirem vértices não visitados.



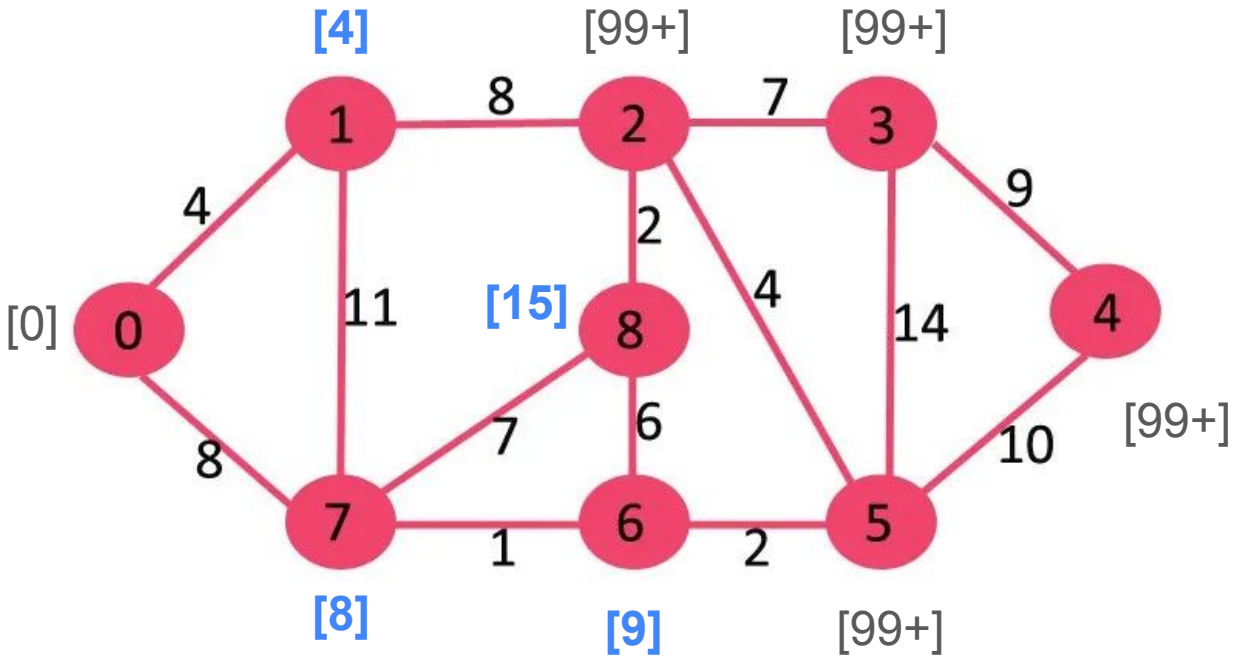
Algoritmo de Dijkstra

1. Inicializar custos.
2. Escolher o vértice não visitado com menor distância atual.
3. **Atualizar a distância dos vizinhos.**
4. Escolher um novo vértice não visitado para ser o vértice atual.
5. Repetir passos 2, 3 e 4 até não existirem vértices não visitados.



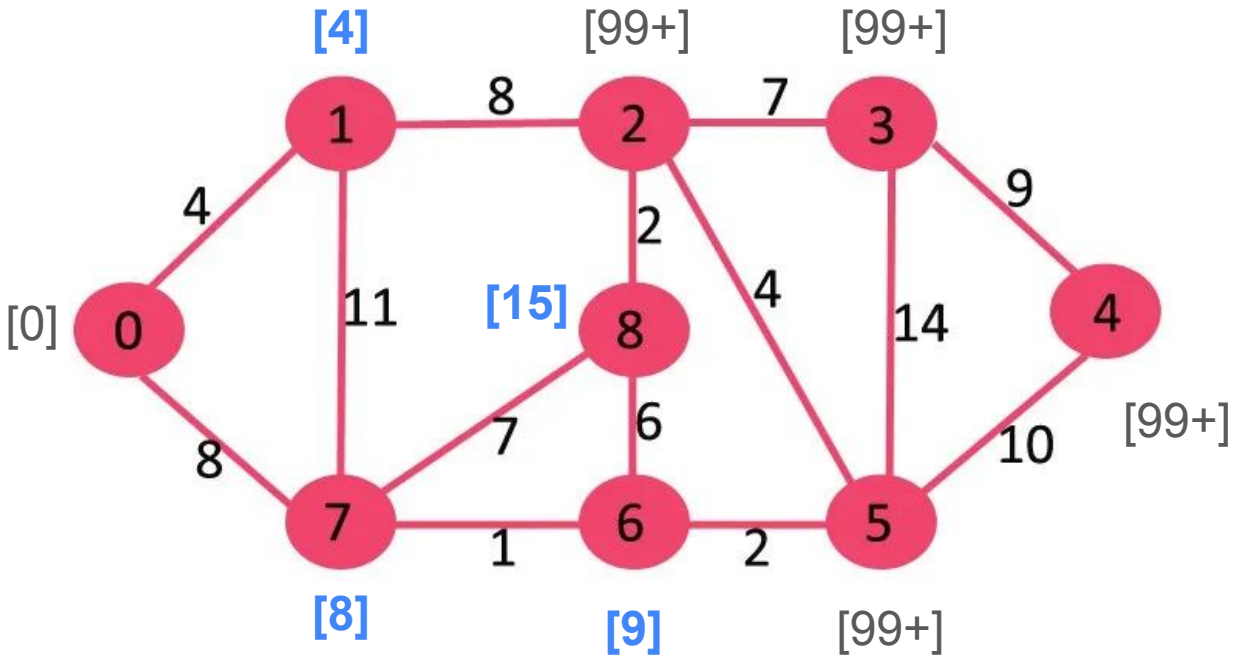
Algoritmo de Dijkstra

1. Inicializar custos.
2. Escolher o vértice não visitado com menor distância atual.
3. **Atualizar a distância dos vizinhos.**
4. Escolher um novo vértice não visitado para ser o vértice atual.
5. Repetir passos 2, 3 e 4 até não existirem vértices não visitados.



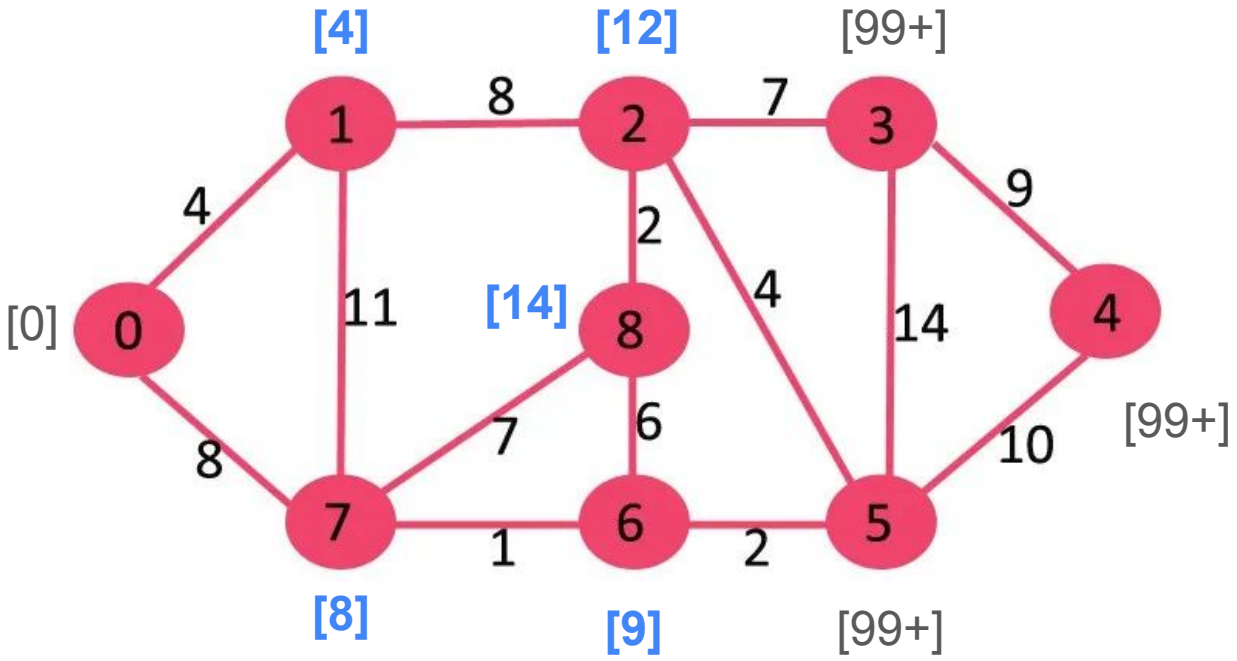
Algoritmo de Dijkstra

1. Inicializar custos.
2. Escolher o vértice não visitado com menor distância atual.
3. **Atualizar a distância dos vizinhos.**
4. Escolher um novo vértice não visitado para ser o vértice atual.
5. Repetir passos 2, 3 e 4 até não existirem vértices não visitados.



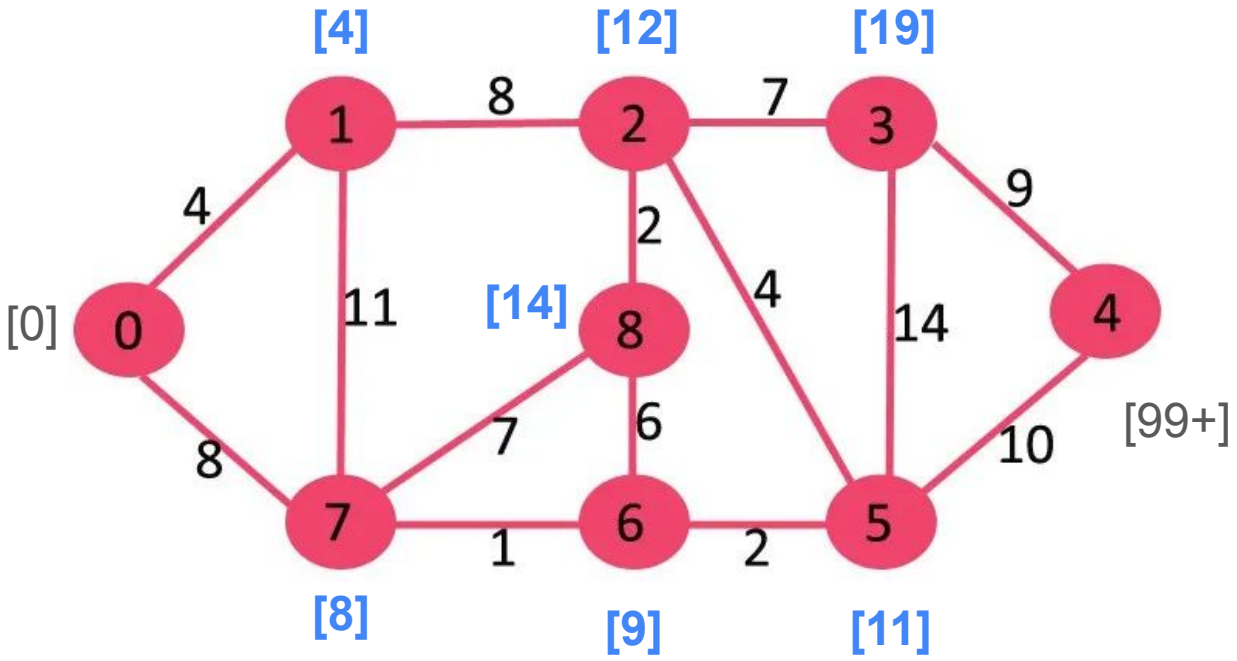
Algoritmo de Dijkstra

1. Inicializar custos.
2. Escolher o vértice não visitado com menor distância atual.
3. **Atualizar a distância dos vizinhos.**
4. Escolher um novo vértice não visitado para ser o vértice atual.
5. Repetir passos 2, 3 e 4 até não existirem vértices não visitados.



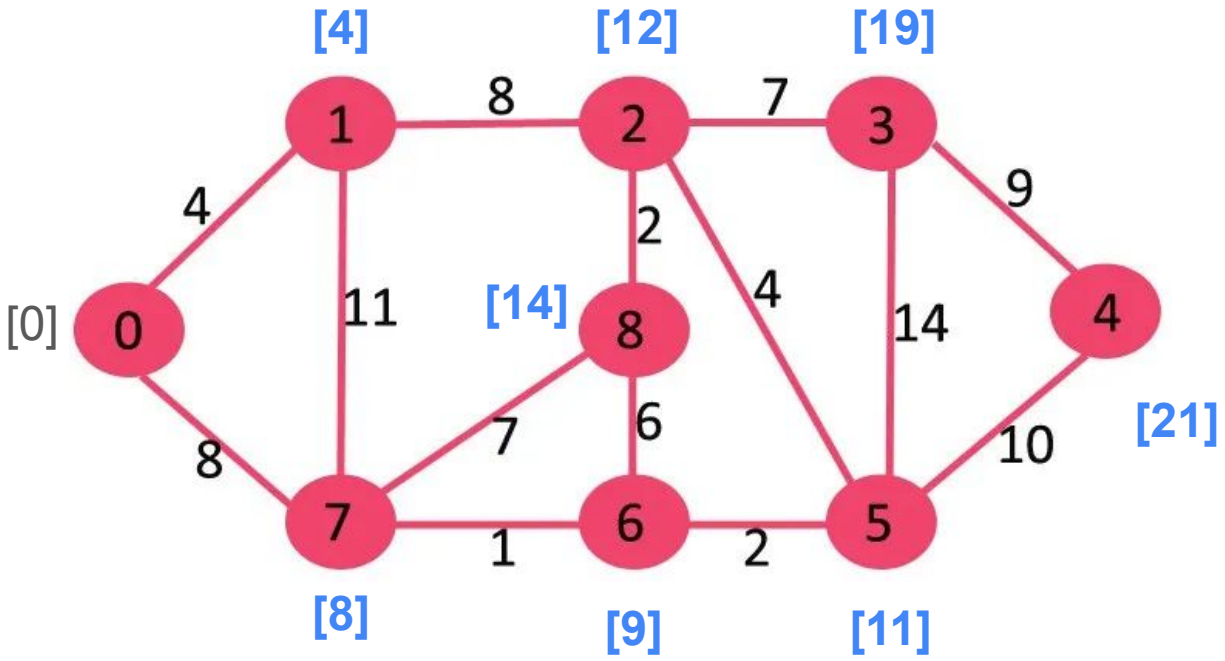
Algoritmo de Dijkstra

1. Inicializar custos.
2. Escolher o vértice não visitado com menor distância atual.
3. **Atualizar a distância dos vizinhos.**
4. Escolher um novo vértice não visitado para ser o vértice atual.
5. Repetir passos 2, 3 e 4 até não existirem vértices não visitados.



Algoritmo de Dijkstra

1. Inicializar custos.
2. Escolher o vértice não visitado com menor distância atual.
3. **Atualizar a distância dos vizinhos.**
4. Escolher um novo vértice não visitado para ser o vértice atual.
5. Repetir passos 2, 3 e 4 até não existirem vértices não visitados.



Algoritmo de A*

O A* surgiu como uma melhoria do algoritmo de Dijkstra (1956), combinando os conceitos de Busca de Custo Uniforme com Busca Heurística.

O diferencial do algoritmo A* foi adicionar uma função **heurística** para estimar a distância restante até o destino, tornando a busca mais eficiente.

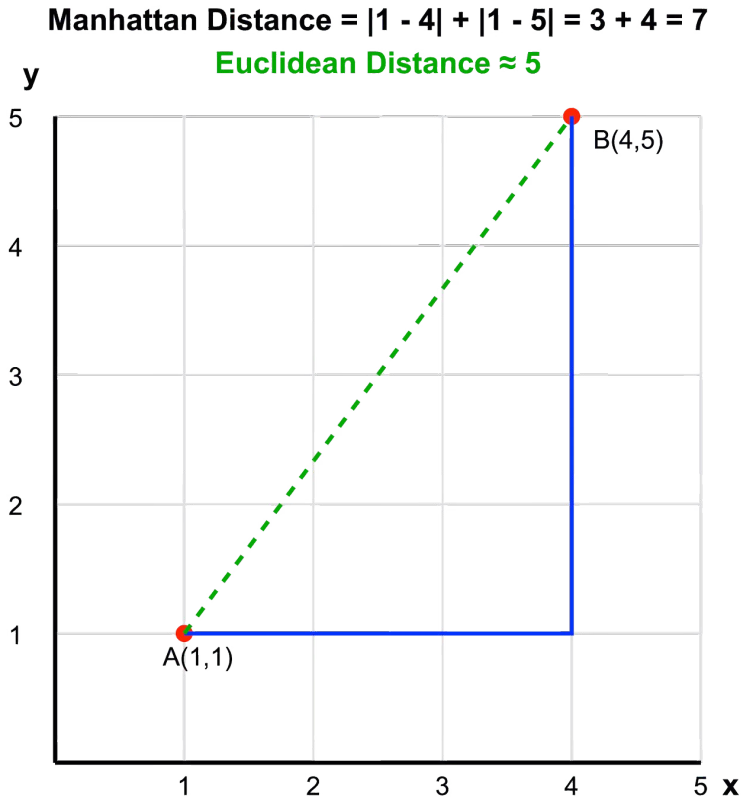
Heurísticas - 2D

Distância de Manhattan

$$d(Man) = |x_1 - x_2| + |y_1 - y_2|$$

Distância Euclidiana

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$



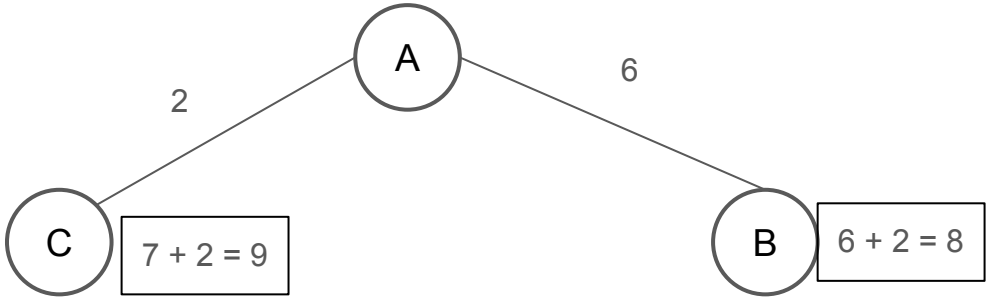
Heurísticas - Distância de Haversine - 3D

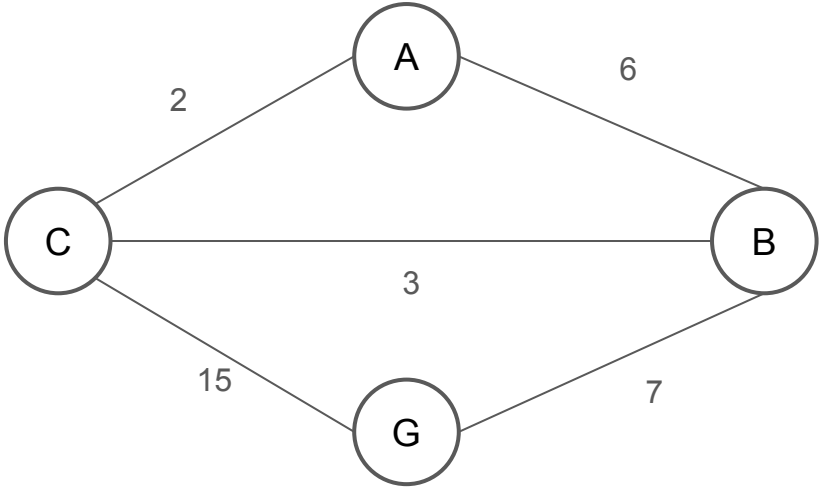
- $d = 2r\ arcsine(\sqrt{sin^2\left(\frac{\varphi_2-\varphi_1}{2}\right) + cos\varphi_1.cos\varphi_2.sin^2(\frac{\omega_2-\omega_1}{2})})$
- Where
  - φ_1, φ_2 are the latitude of point 1 and point 2 in radian form.
  - ω_1, ω_2 are the longitude of point 1 and point 2 in radian form.

https://community.esri.com/t5/coordinate-reference-systems-blog/distance-on-a-sphere-the-haversine-formula/ba-p/902128

A* - Algoritmo

Visitados: A

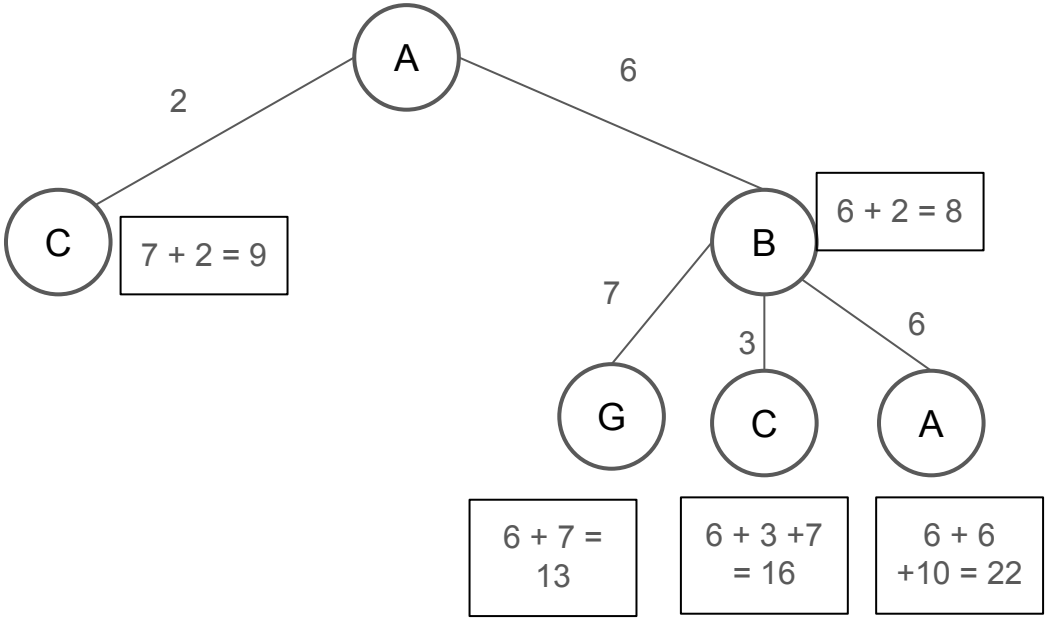


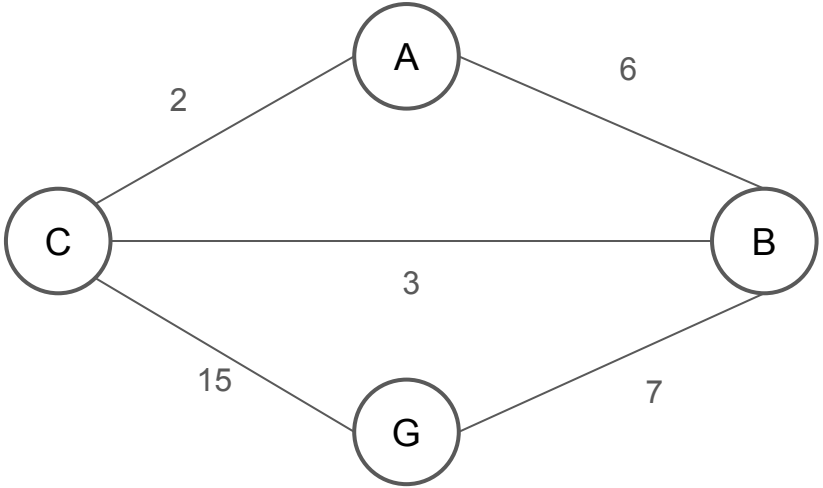


Heurística	
Nó	Distância Até Objetivo
A	10
B	2
C	7
G	0

A* - Algoritmo

Visitados: A,C,B

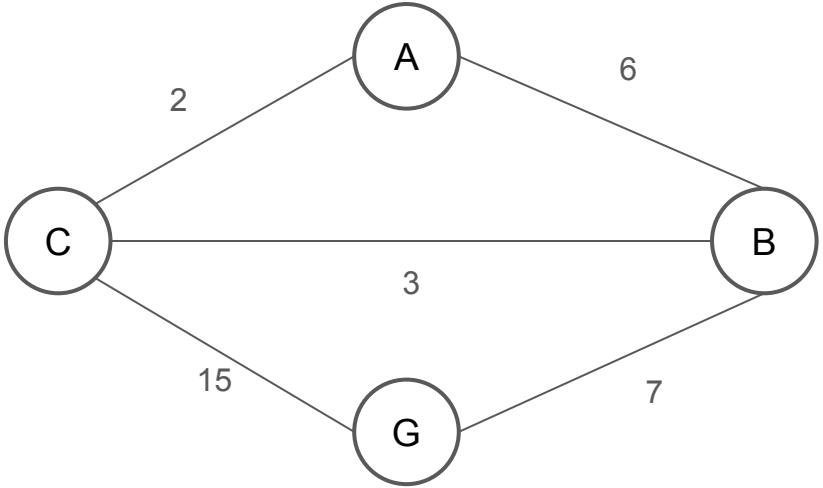


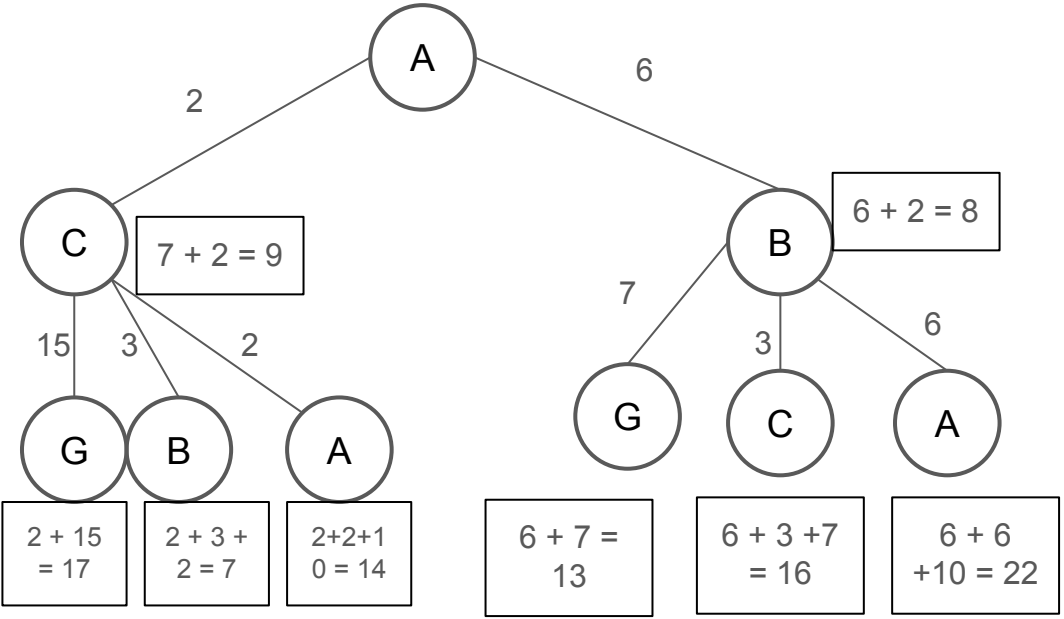


Heurística	
Nó	Distância Até Objetivo
A	10
B	2
C	7
G	0

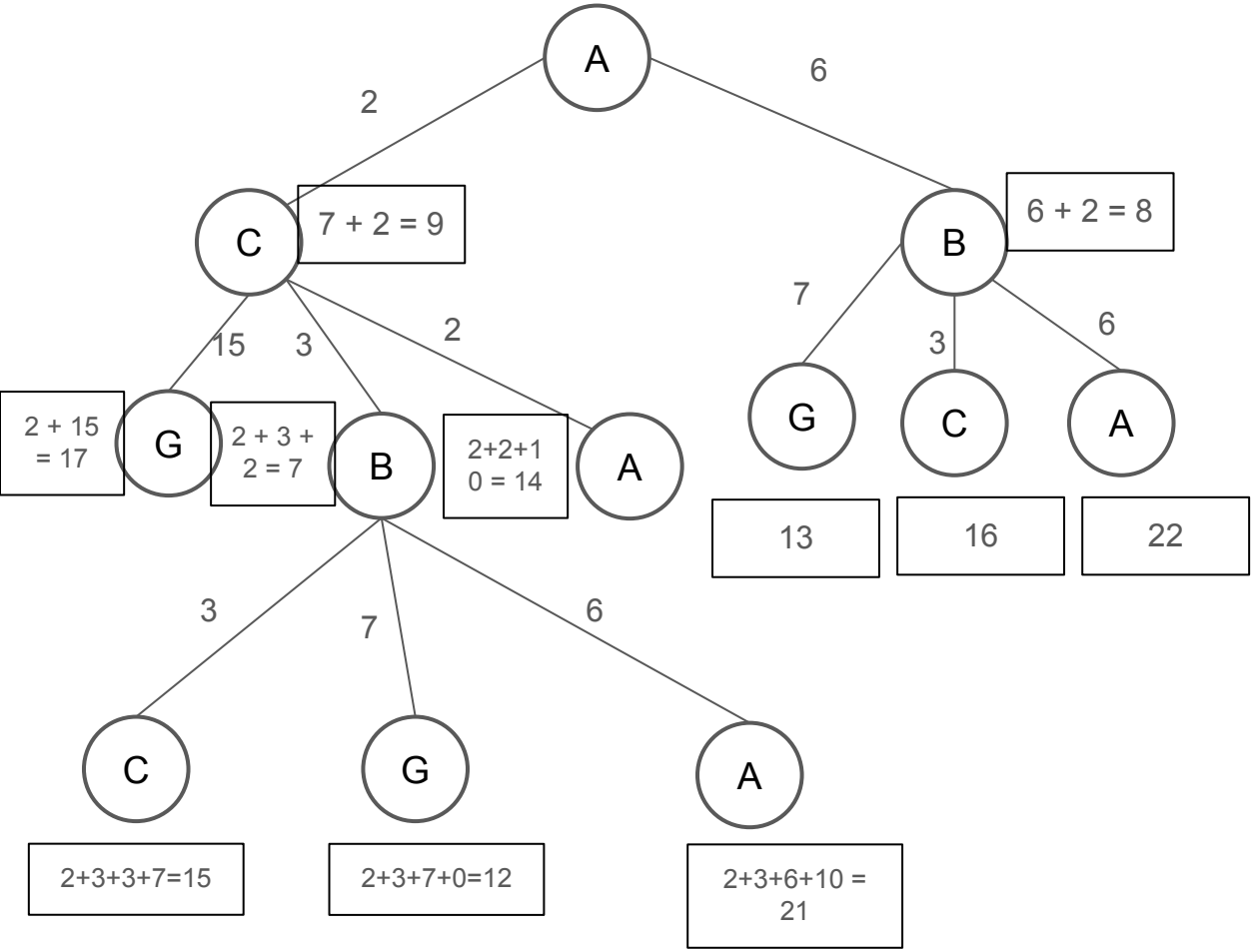
A* - Algoritmo

Visitados: A,C,B

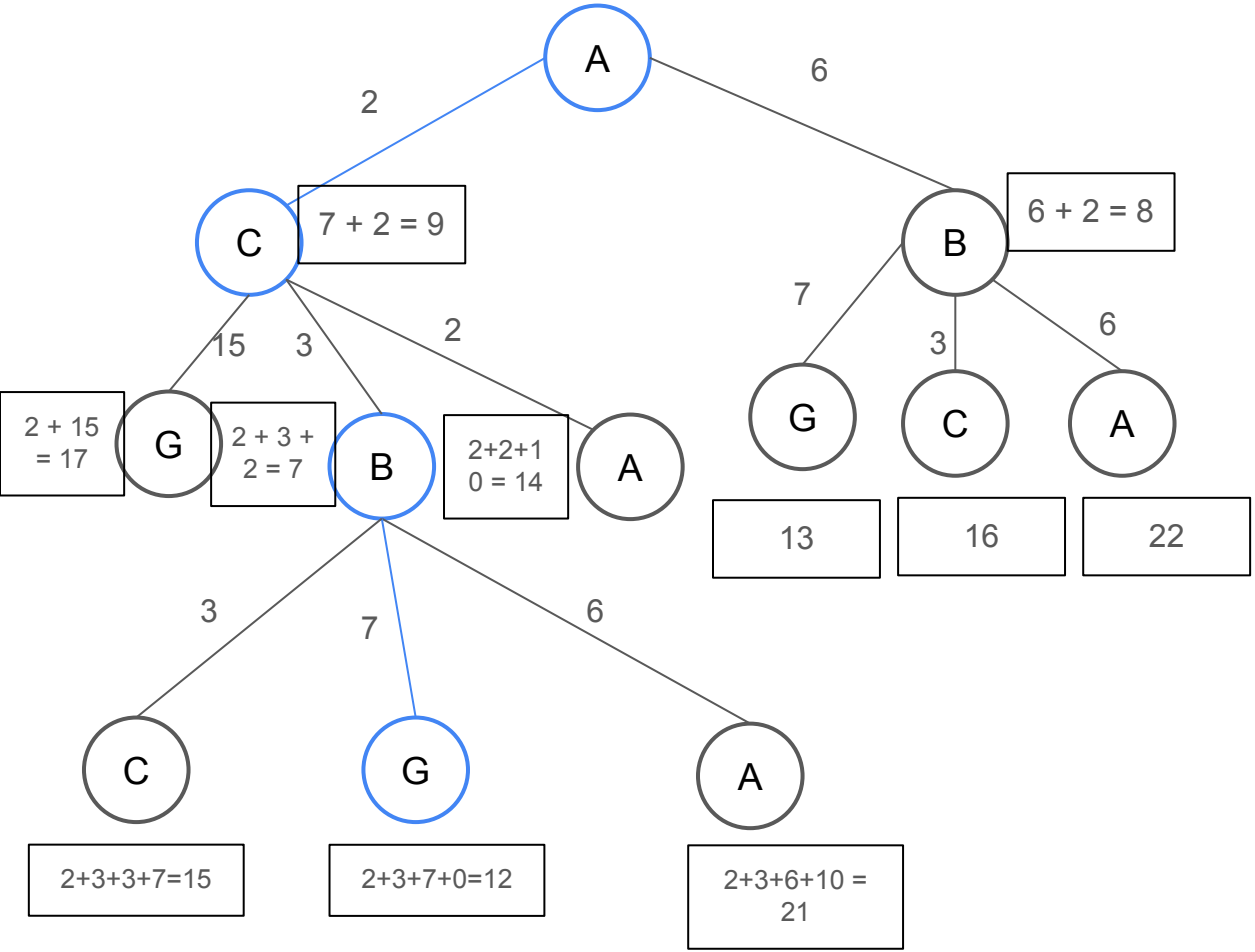




Heurística	
Nó	Distância Até Objetivo
A	10
B	2
C	7
G	0



Heurística	
Nó	Distância Até Objetivo
A	10
B	2
C	7
G	0



Heurística	
Nó	Distância Até Objetivo
A	10
B	2
C	7
G	0