SCC 503 - ALGORITMOS E ESTRUTURAS DE DADOS II

Grafos - Pathfinding com A*

Prof.: Leonardo Tórtoro Pereira
leonardop@usp.br

*Material baseado em aulas dos professores: Elaine Parros Machado de Souza, Gustavo Batista, Robson Cordeiro, Moacir Ponti Jr., Maria Cristina Oliveira e Cristina Ciferri.

Pathfinding

Pathfinding

- ➔ Planejamento de movimento entre pontos distantes
- ➔ “Roteador”
- ➔ Rota deve ser o mais sensata e curta possível
- ➔ Muito usado para jogos, robótica, simulações de modo geral

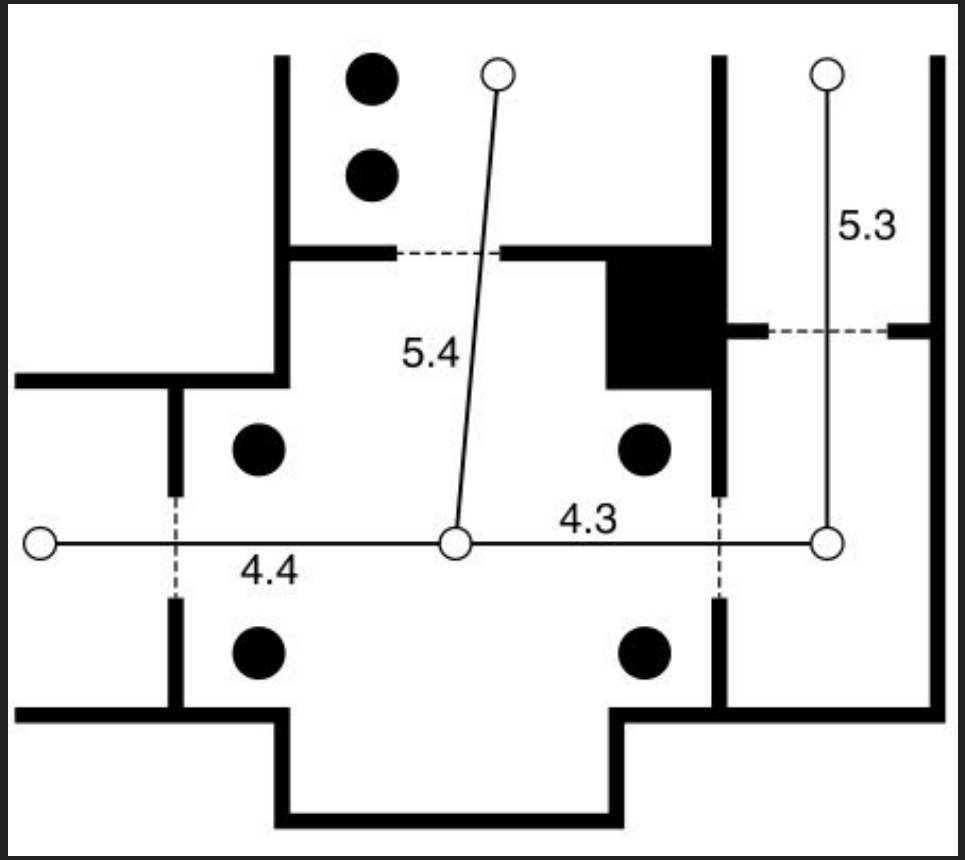
Pathfinding

- A maioria das aplicações usa o A*
  - Eficiente e fácil de implementar
  - Não trabalha diretamente com os dados de nível do problema
  - Requer que a aplicação seja representada como grafo direcionado com pesos não negativos
  - “Primo” do algoritmo de Dijkstra

Pathfinding

- Tanto Dijkstra quanto A* (e outros grafos) usam uma simplificação do contexto da aplicação
  - Representada em grafo
  - Se bem simplificada
    - O plano retornado pelo algoritmo será útil quando traduzido em termos de jogos/robótica/etc
  - Se simplificação descartar informações relevantes
    - Plano pode ser ruim

Grafo ponderado sobreposto à geometria de um nível



Fonte: Artificial Intelligence for Games

*Pathfinding - A**

Pathfinding - A*

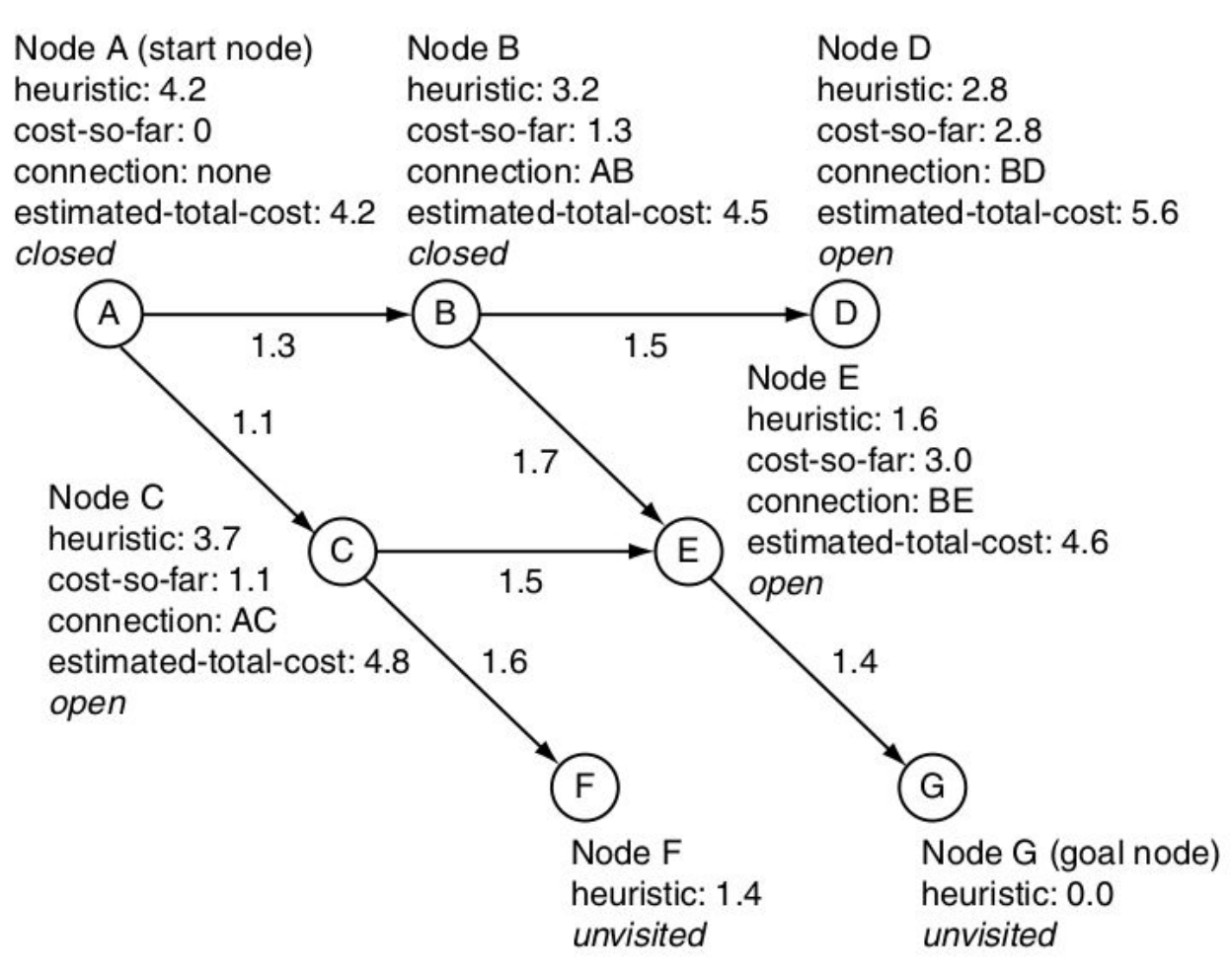
- Simples de implementar
- Muito eficiente
- Grande espaço para otimização
- Encontra caminho ponto-a-ponto
  - E não caminho mínimo
- Ao invés de considerar o nó visitado com o valor mais baixo até o momento, considera aquele que provavelmente levará ao melhor caminho (heurística)

Pathfinding - A*

- Depende muito da heurística escolhida
- Trabalha por iterações
  - A cada iteração considera 1 nó do grafo e segue suas conexões de saída
  - O nó (chamado de nó atual) é escolhido através de um algoritmo de seleção similar ao de Dijkstra, mas com a diferença da heurística

Pathfinding - A*

- Em cada iteração, para cada conexão de saída do nó é encontrado o nó final, e o valor total do custo do caminho até agora é armazenado, juntamente com a conexão com a qual o algoritmo chegou até ele.
- Além disso, o custo total estimado para o caminho do nó inicial até este nó, e deste nó até o objetivo.
  - Soma do custo do caminho até agora e o quão longe está do objetivo (heurística)



Fonte: Artificial Intelligence for Games

Pathfinding - A*

- O algoritmo mantém uma lista de nós abertos já visitados mas não processados, e nós fechados (já processados)
- Nós são movidos para a lista de abertos conforme são encontrados no fim das conexões
- Nós são movidos para a lista de fechados conforme são processados em sua iteração

Pathfinding - A*

- O nó da lista de abertos com o menor custo total estimado é selecionado a cada iteração
  - Quase sempre é diferente do nó com o menor custo até o momento
- Permite ao algoritmo examinar nós que são mais promissores primeiro

Pathfinding - A*

- Se a heurística for boa, os nós mais pertos do objetivo são considerados primeiro, estreitando a busca para a área mais promissora
- Ao chegar em um nó aberto ou fechado, é necessário revisar seus valores
- Calcula-se o custo até agora normalmente
  - Se for mais baixo que o valor existente no nó, é necessário atualizá-lo (não usar custo estimado)

Pathfinding - A*

- Diferente de Dijkstra, o A* pode achar rotas melhores para nós que estão na lista de fechados.
- Se uma estimativa prévia for muito otimista, o nó pode ter sido processado pensando-se que era a melhor escolha quando, de fato, não o era
  - Isso pode causar um problema em cadeia

Pathfinding - A*

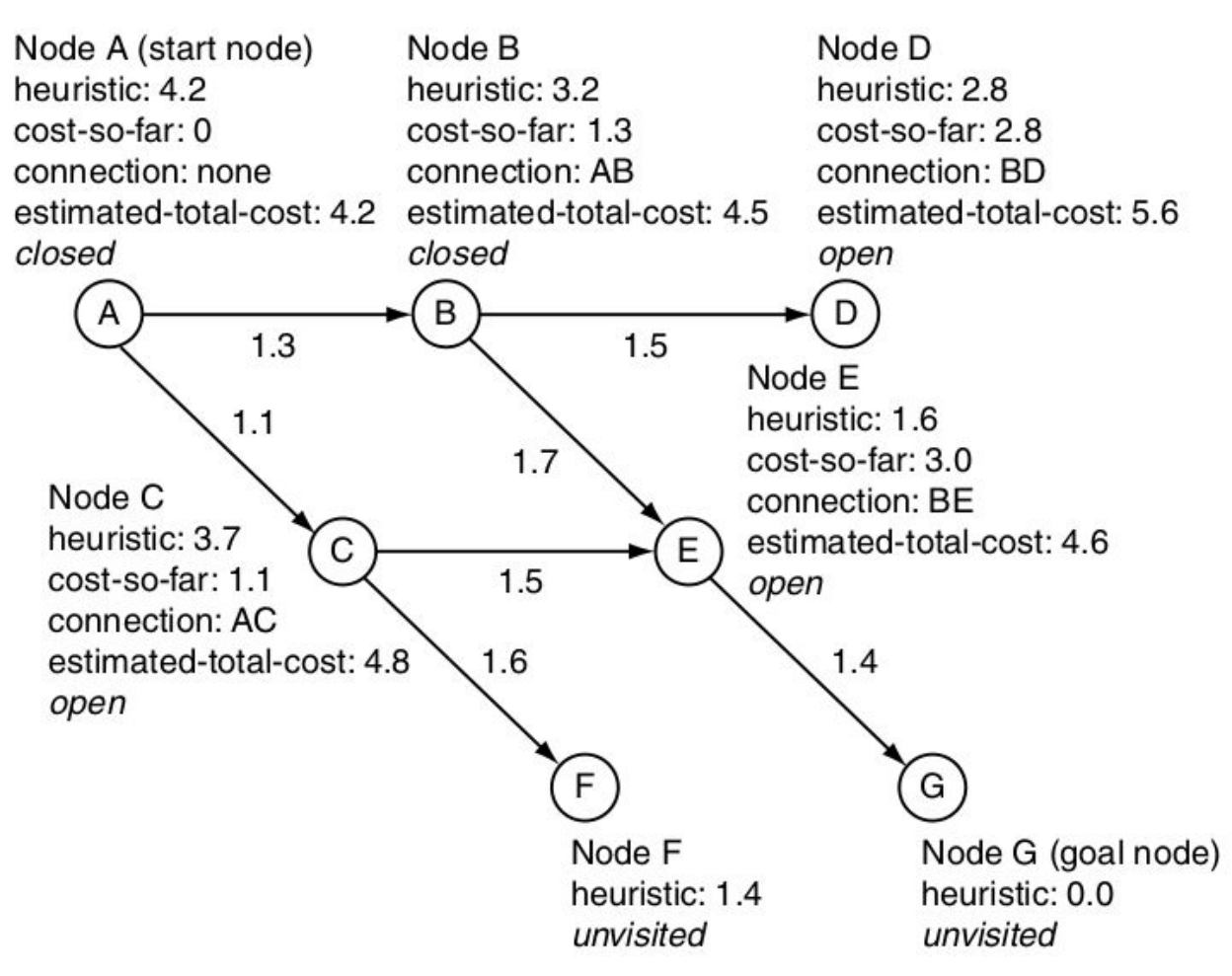
- Se um nó teve uma estimativa errada e colocado na lista de fechados, todas as suas conexões foram consideradas
- Pode ser que um conjunto de nós tiveram seus custos até agora baseado em um custo até agora de um nó errado
- Portanto, atualizar o valor do nó errado não é o suficiente
  - Todas suas conexões devem ser checadas novamente para propagar o valor

Pathfinding - A*

- ➔ No caso de revisar um nó na lista de abertos, isso não é necessário, uma vez que sabemos que conexões de um nó na lista não foram processadas ainda
- ➔ Existe um jeito simples de forçar o cálculo e propagação do valor
- ➔ O nó pode ser removido da lista de fechados e colocado de volta na de abertos

Pathfinding - A*

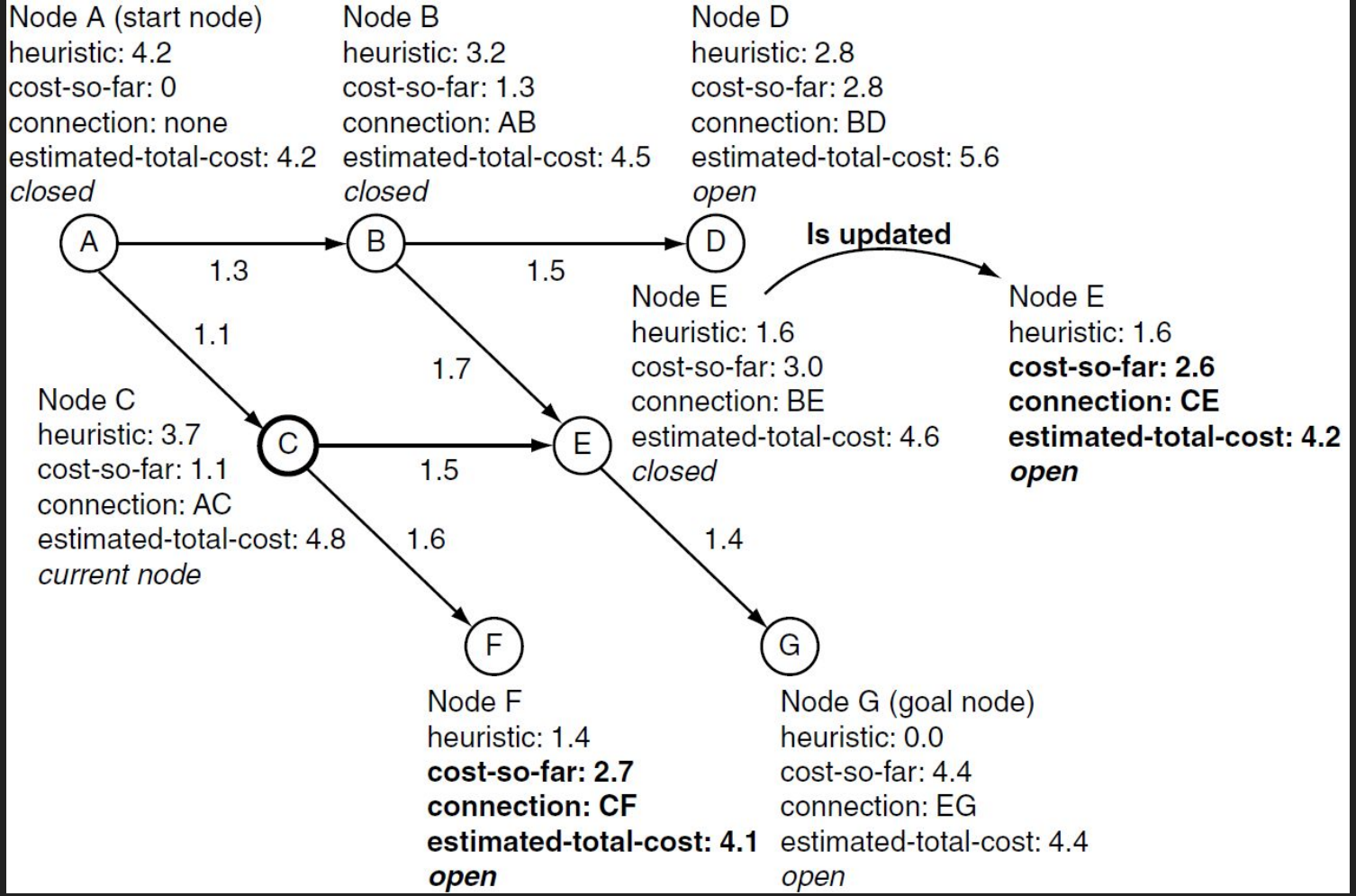
- ➔ Então esperará até ser fechado e ter suas conexões reconsideradas
- ➔ Quaisquer nós que dependam de seu valor serão eventualmente processados mais uma vez
- ➔ Lembra do grafo anterior?



Fonte: Artificial Intelligence for Games

Pathfinding - A*

- ➔ Este aqui é ele, 2 iterações depois
- ➔ Mostra a atualização de um nó fechado no grafo
- ➔ A nova rota E, via C, é mais rápida



Fonte: Artificial Intelligence for Games

A*

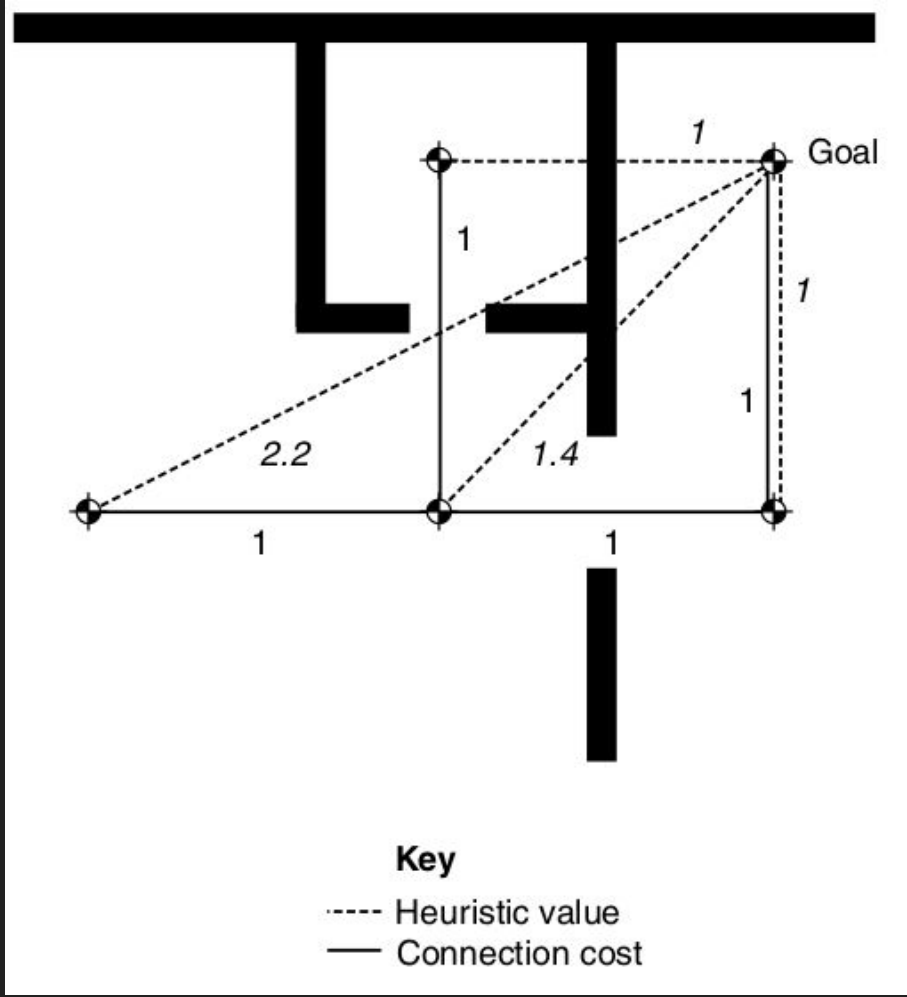
http://www.policyalmanac.org/games/aStarTutorial.htm

https://qiao.github.io/PathFinding.js/visual/

Pathfinding - A*

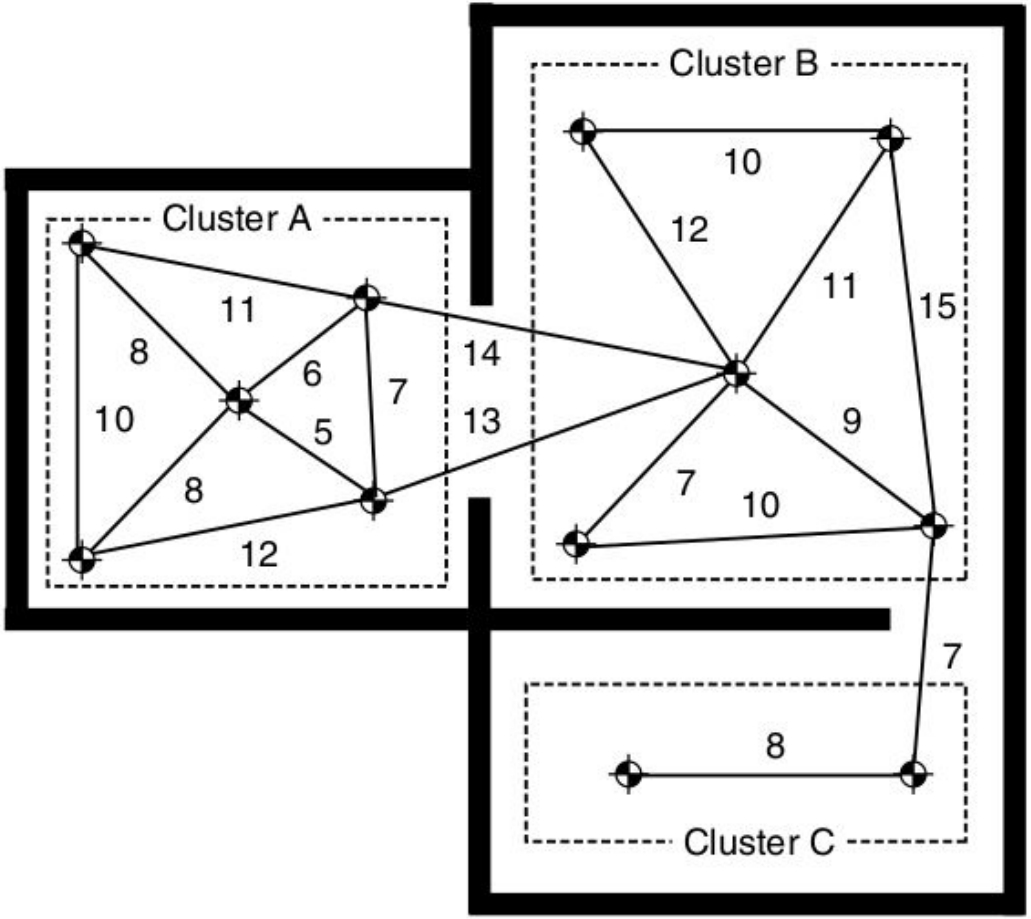
➔ Um pouco sobre heurísticas

Euclidiana



Fonte: Artificial Intelligence for Games

Cluster



	A	B	C
A	x	13	29
B	13	x	7
C	29	7	x

Lookup table

Fonte: Artificial Intelligence for Games

Representação de Mundo

Representação de Mundo

- ➔ É preciso mudar a representação do mundo para os nós e conexões do grafo e a função de custo que dá seus valores
- ➔ Os diferentes modos de dividir o nível em regiões conectadas são chamadas de esquemas de divisão

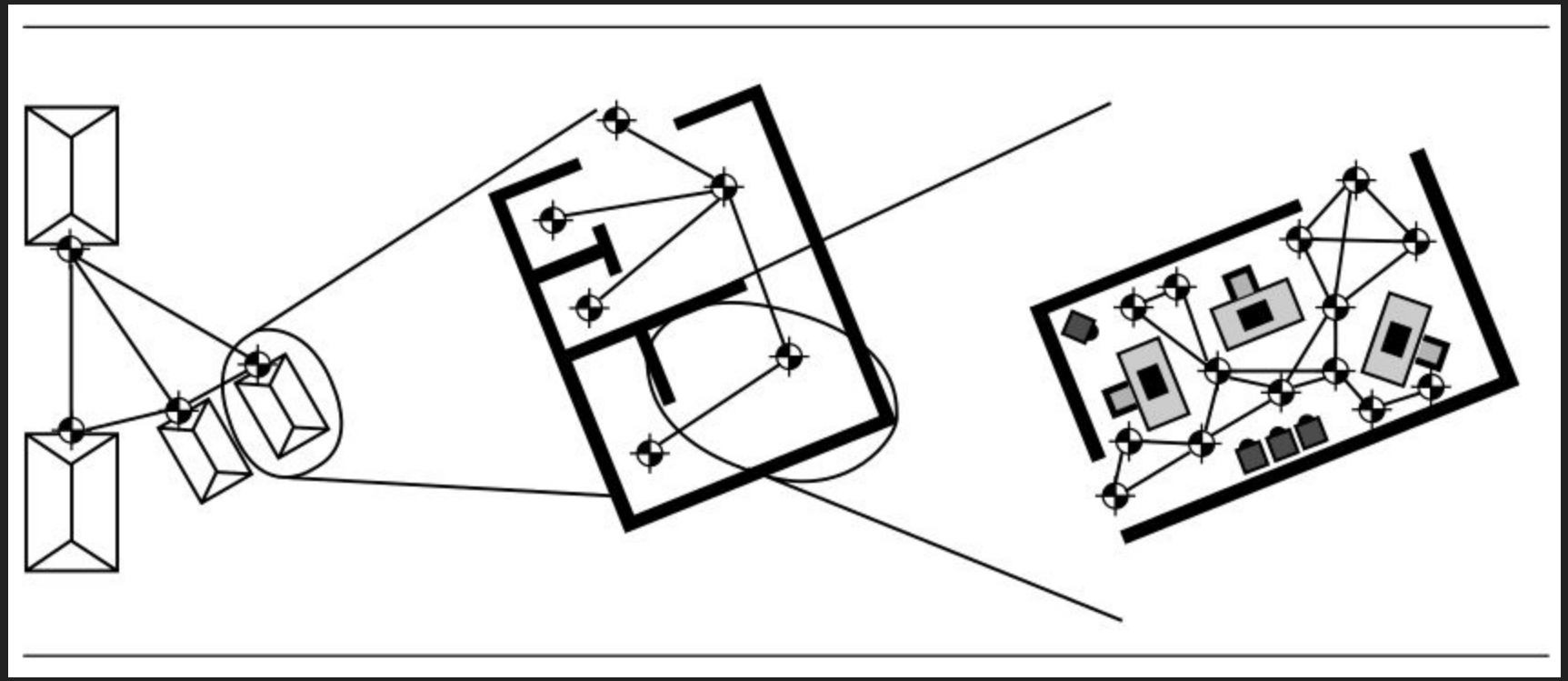
Representação de Mundo

- Cada esquema de divisão tem 3 propriedades importantes
  - Quantização/Localização
  - Geração
  - Validade
- Existem diversos tipos de esquema com suas vantagens e desvantagens
- Não serão abordados nesta aula

Grafo de *Pathfinding* Hierárquico

Grafo de Pathfinding Hierárquico

- ➔ Formação de *clusters* de lugares
- ➔ Processo similar ao de colisão (*Quad-trees* e *Octa-trees*)



Fonte: Artificial Intelligence for Games

Pathfinding - mais ideias

Pathfinding - mais ideias

- ➔ *Pathfinders* podem ser dinâmicos (consideram mudanças no ambiente)
- ➔ E até contínuos (de difícil implementação)

Referências

- ARTIFICIAL INTELLIGENCE FOR GAMES. Millington, I; Funge, J. Burlington, MA : Elsevier Morgan Kaufmann, 2009. 2ª ed.
- http://www.policyalmanac.org/games/aStarTutorial.htm
- WIRTH,N. Algorithms and Data Structures, Englewood Cliffs, Prentice-Hall, 1986.
- CORMEN, H.T.; LEISERSON, C.E.; RIVEST, R.L. Introduction to Algorithms, MIT Press, McGraw-Hill, 1999.
- ZIVIANI, N. Projeto de Algoritmos, Thomson, 2a. Edição, 2004.
- SZWARCFITER,J.L. Grafos e Algoritmos Computacionais. Editora Campus, 1983.
- Van Steen, Maarten. "Graph theory and complex networks." An introduction 144 (2010).
- Gross, Jonathan L., and Jay Yellen. Graph theory and its applications. CRC press, 2005.
- Barabási, A.-L., Pósfai, M. (2016). Network science. Cambridge: Cambridge University Press. ISBN: 9781107076266 1107076269