Pontifícia Universidade Católica
do Rio de Janeiro

INF 1771 – Inteligência Artificial

Aula 22 – Waypoints e Pathfinding

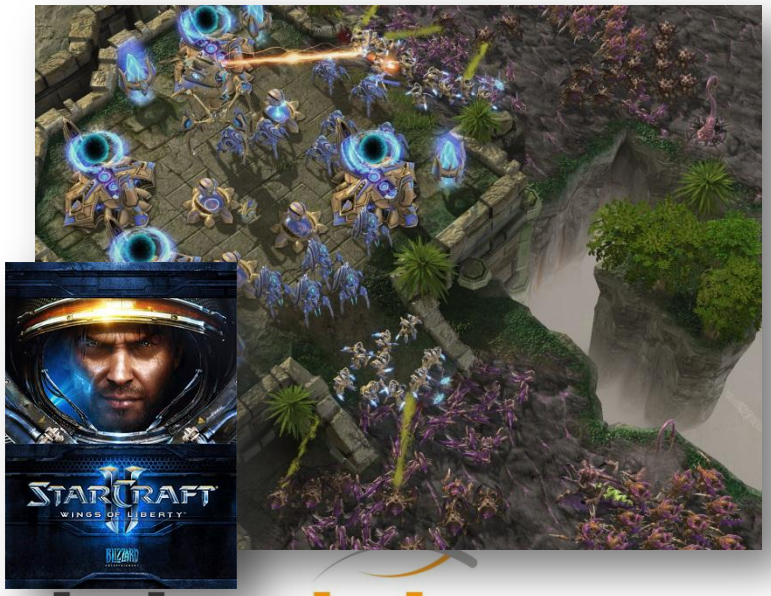
Edirlei Soares de Lima
<elima@inf.puc-rio.br>

visionlab
visualization laboratory

Introdução

- **Locomover-se no espaço do jogo** é uma ação fundamental dos NPCs em qualquer gênero de jogo.

- A busca de caminhos deve ser implementada de maneira muito eficiente, pois **será executada muitas vezes** por vários personagens durante o jogo



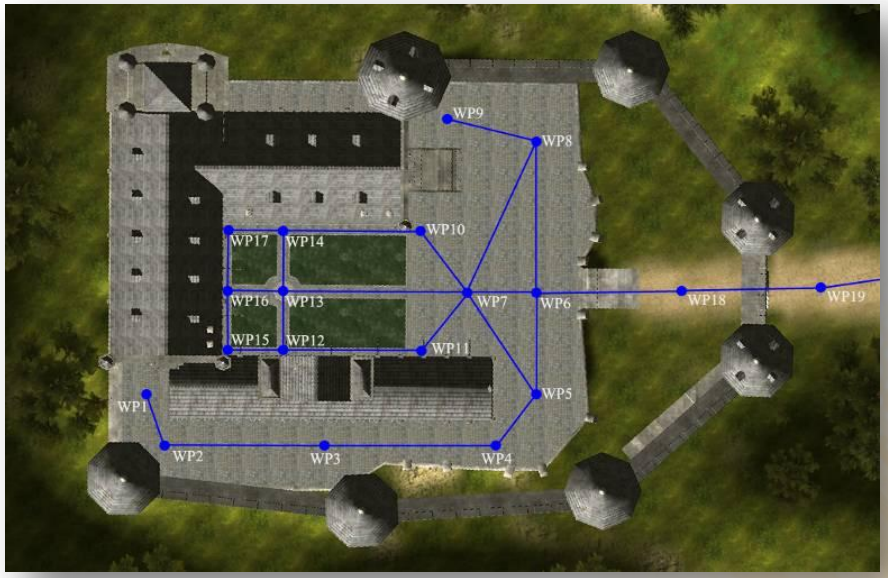
visionlab
visualization laboratory

Waypoints

- **Waypoints** são uma representação aproximada do terreno (amostragem).

- Fornecem **representações mais econômicas** do que as malhas poligonais dos cenários.

- **Estrutura de grafos**.



visionlab
visualization laboratory

Construção de Grafos de Navegação

- Um **grafo de navegação** é uma estrutura de grafo formada por um conjunto de waypoints.

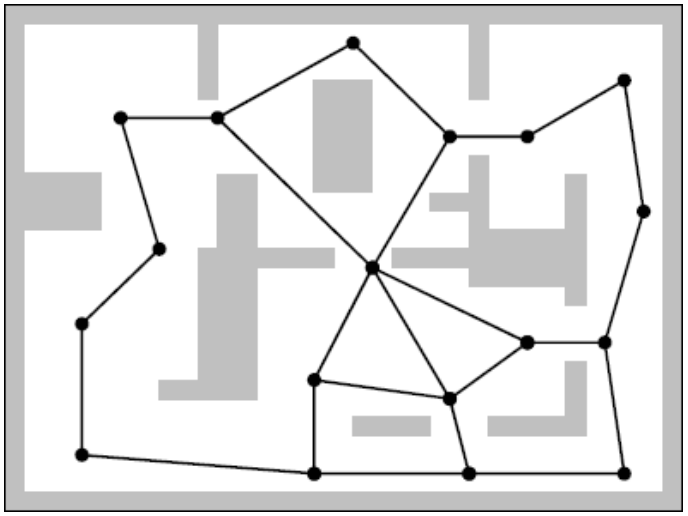
- Existem várias formas de representar a geometria do ambiente do jogo, da mesma forma existem várias estratégias para **converter a informação espacial em uma estrutura de grafo**:

  - Pontos de Visibilidade
  - Tiles (Grid)
  - Geometria Expandida
  - NavMesh

Pontos de Visibilidade

- A criação de um grafo de navegação baseado em **Pontos de Visibilidade** consiste na adição de nós em pontos importantes do ambiente.

- Os pontos do grafo devem ter pelo menos uma **linha reta de visão** para algum outro ponto.



Pontos de Visibilidade

- Geralmente o processo de adição dos pontos de visibilidade é feito manualmente pelo **game designer**.

- Se o jogo **restringe o movimento** dos agentes somente **sobre as arestas do grafo**, como ocorre no Pac-man, esta solução é a escolha perfeita.

- Entretanto, se o grafo é projetado para um jogo onde os agentes têm **maior liberdade** de movimentos é necessário realizar mais algumas tarefas.

Pontos de Visibilidade

1) Procura o nó visível A mais próximo da posição do NPC;

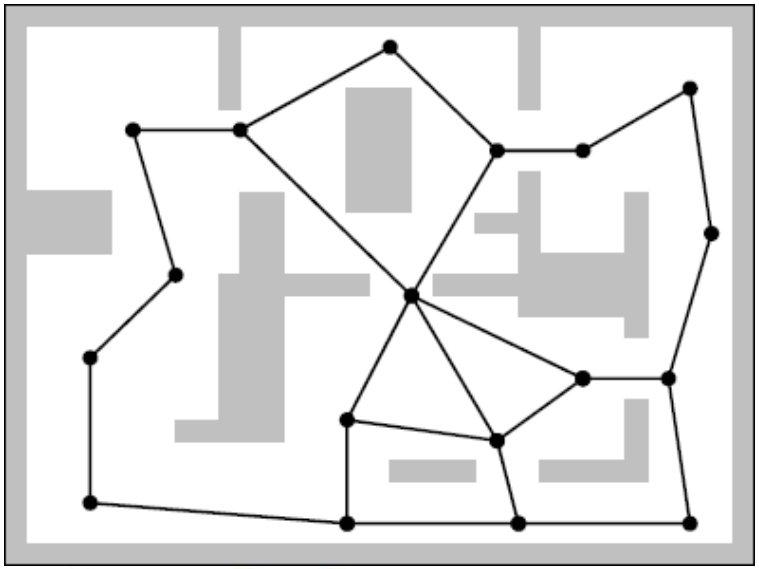
2) Procura o nó visível B mais próximo da posição destino;

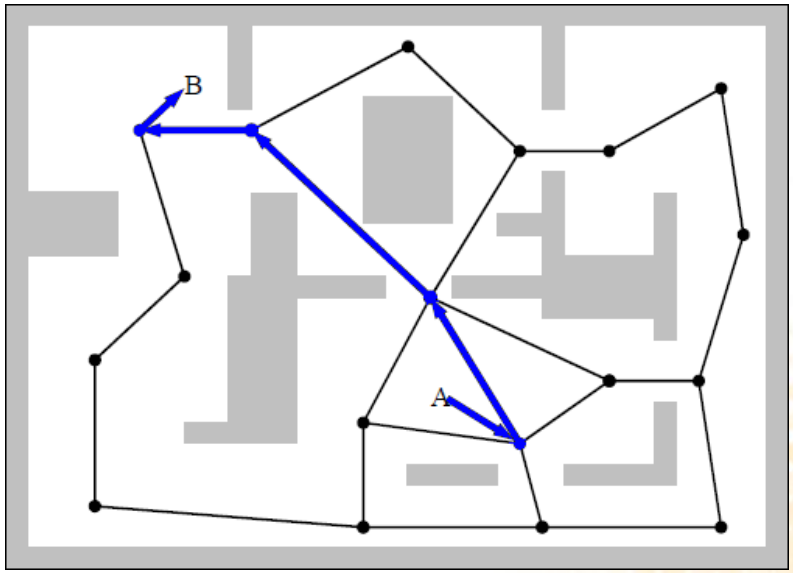
3) Usa um algoritmo para encontrar o caminho de menor custo entre A e B;

4) Move o NPC para o nó A;

5) Move o NPC sob o caminho calculado no passo 3;

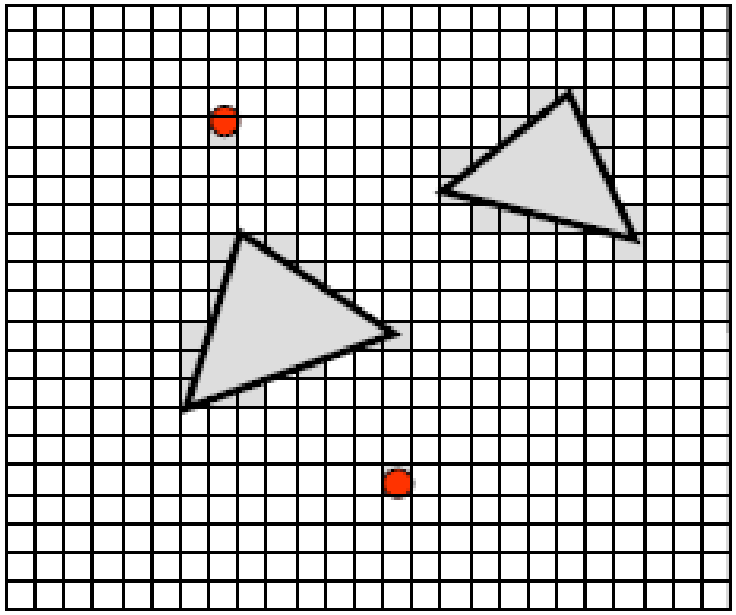
6) Move o NPC do nó B até a posição destino.

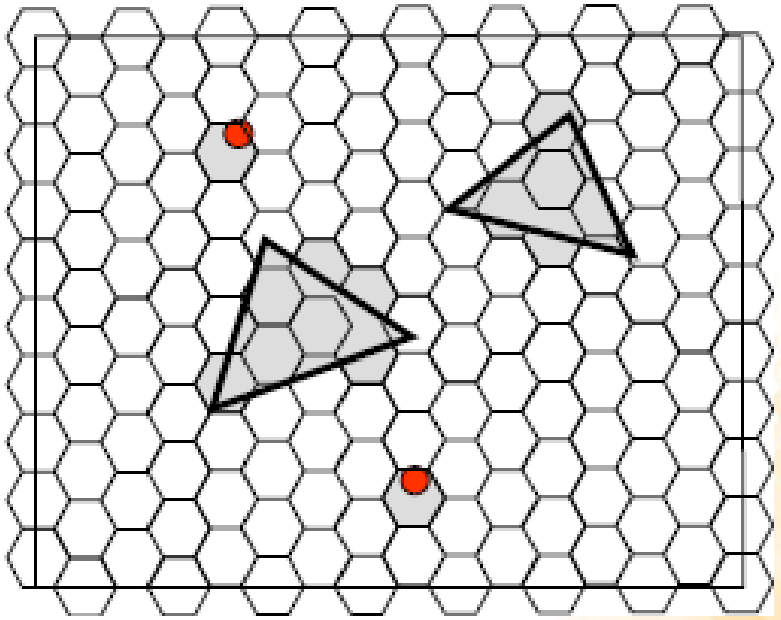




Tiles (Grid)

- Este tipo de abordagem é muito comum em jogos de estratégia em tempo real (RTS).
- Geralmente são grafos grandes e complexos, organizados por meio de quadrados ou hexágonos.
- Cada nó do grafo representa o centro de cada célula e as arestas representam a vizinhança de cada célula.

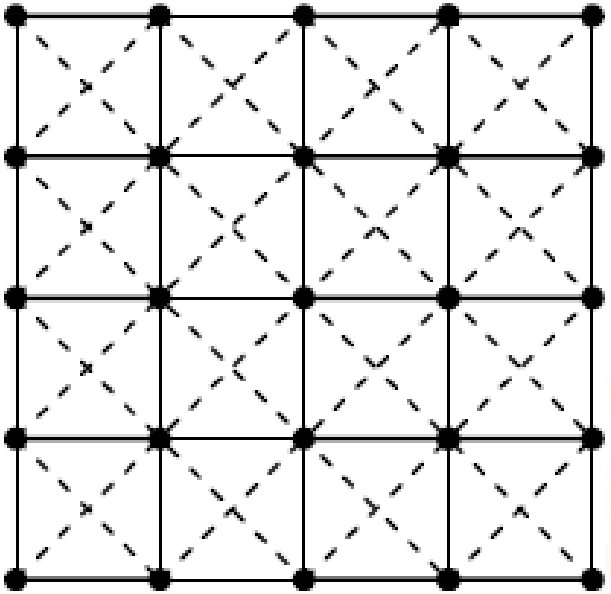




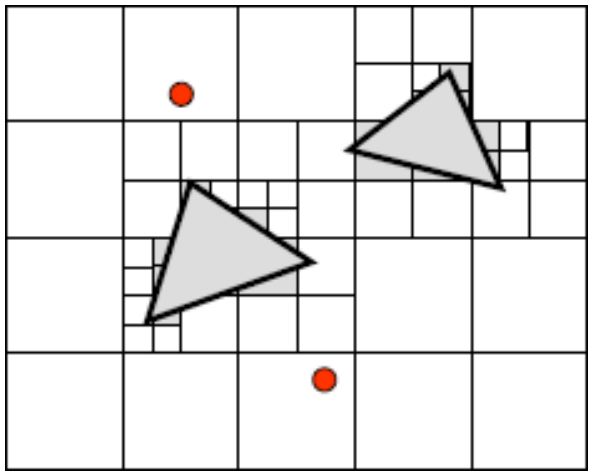
Tiles (Grid)

- O grande problema desta abordagem é que o número de vértices e arestas pode se tornar rapidamente muito elevados.

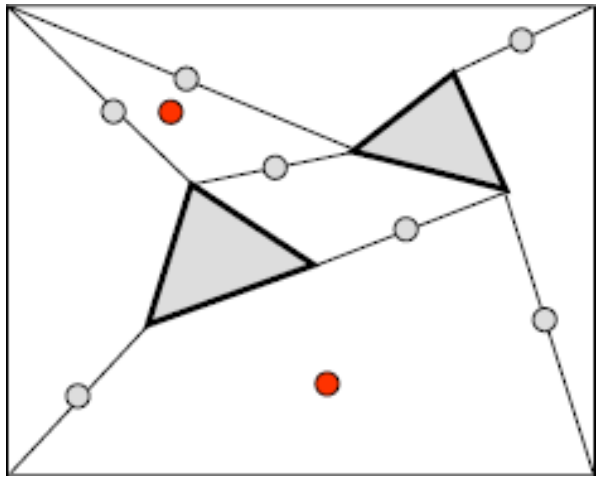
- Para um mapa com 100 x 100 células, tem-se 10.000 nós e 78.000 arestas.



Geração Automática de Waypoints



Quadtree



Polígonos Convexos

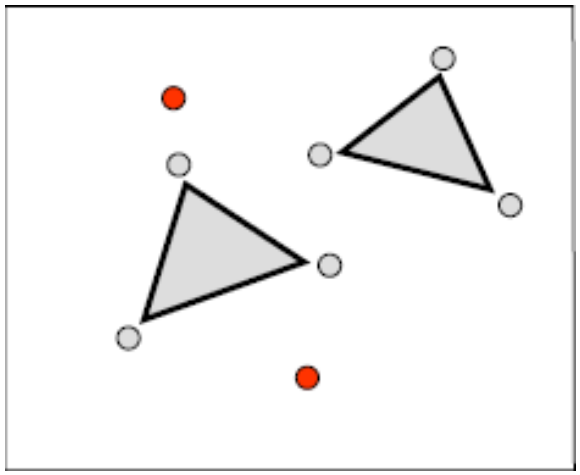
Waypoint no centro ou vértices

Waypoint de inicio ou objetivo

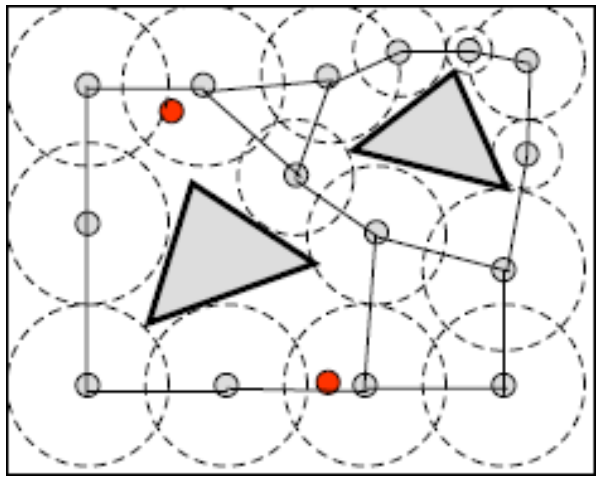
Waypoint

visionlab
visualization laboratory

Geração Automática de Waypoints



Pontos de Canto



Preenchimento de Espaço com Círculos ou Cilindros

- Waypoint no centro ou vértices
- Waypoint de inicio ou objetivo
- Waypoint

visionlab
visualization laboratory

Pathfinding

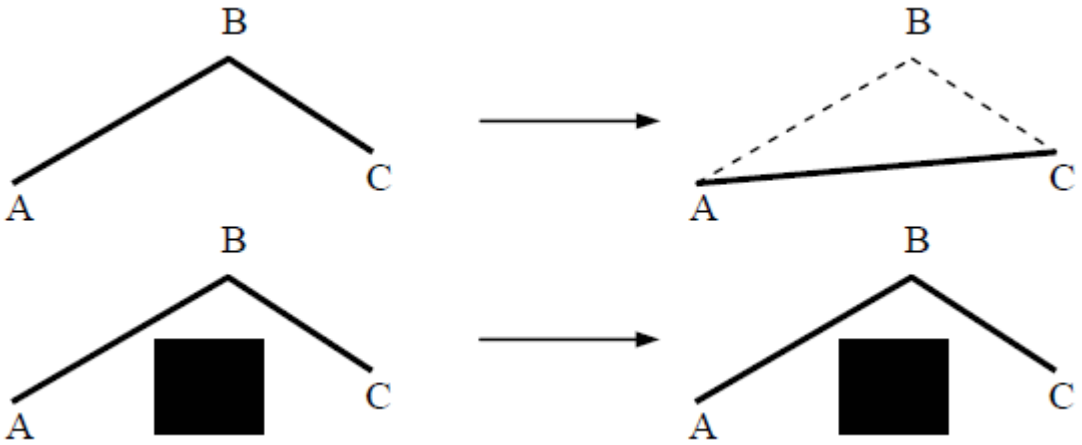
- Uma vez definido o grafo de navegação é possível utilizar técnicas de busca para encontrar caminhos entre dois pontos.

- O algoritmo de busca mais utilizado em jogos é o **A***.
  - Normalmente é possível calcular boas funções heurísticas.

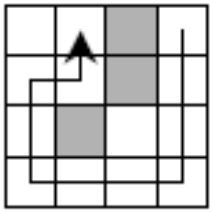
- O caminho gerado pelo A* é composto por uma **lista de vértices** por onde o agente deve passar para chegar ao destino.

Suavização de Caminhos

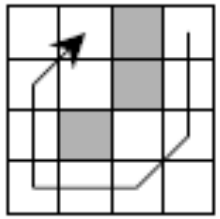
- Geralmente o caminho encontrado forma uma **linha sinuosa** (zigzag), visto que o espaço representado pelos waypoints é discretizado em pontos.
- Geralmente sempre existe uma forma de **suavizar o caminho** encontrado traçando-se um novo caminho que remove vértices não necessários.



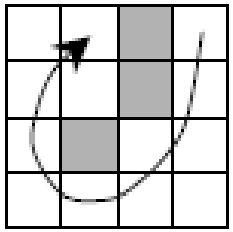
Suavização de Caminhos



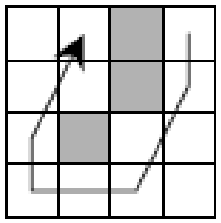
Manhattan



Diagonais



Spline
Catmull-Rom



String-pulling
(LOS – Line of Sight)

visionlab
visualization laboratory

Otimizações

- Em algumas situação do jogo, pode necessário que várias unidades realizem **buscas por caminhos simultaneamente**.

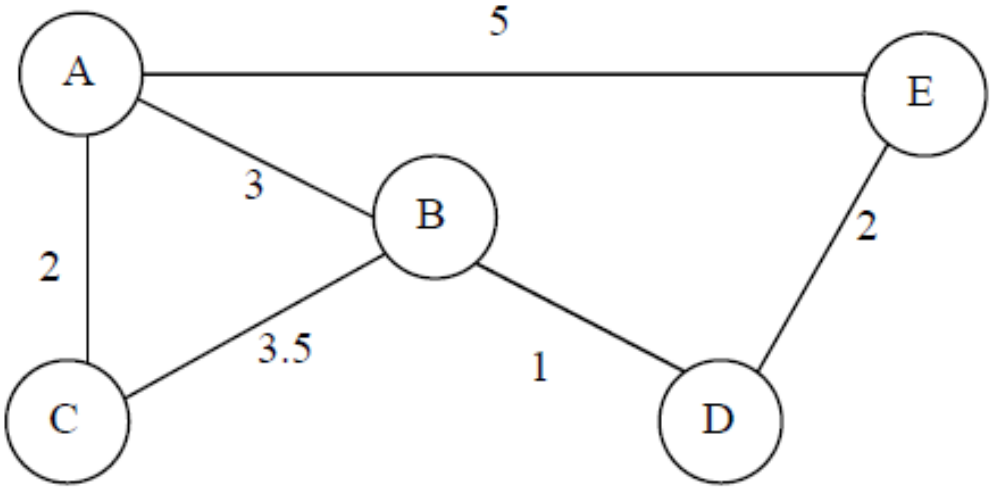
- Isso pode gerar um pico de carga da CPU que pode resultar em interrupções momentâneas do fluxo do jogo.

- **Técnicas de Otimização:**
  - Caminhos pré-calculados
  - Custos pré-calculados
  - Busca de caminhos hierárquica.

Caminhos Pré-Calculados

- **Tabela pré-calculada** com os melhores caminhos do grafo de navegação.

- Armazena-se somente o próximo nó que deve ser seguindo do nó atual ao nó destino.



	A	B	C	D	E
A	A	B	C	B	E
B	A	B	C	D	D
C	A	B	C	B	**B**
D	B	B	B	D	E
E	A	D	D	D	E

Custos Pré-Calculados

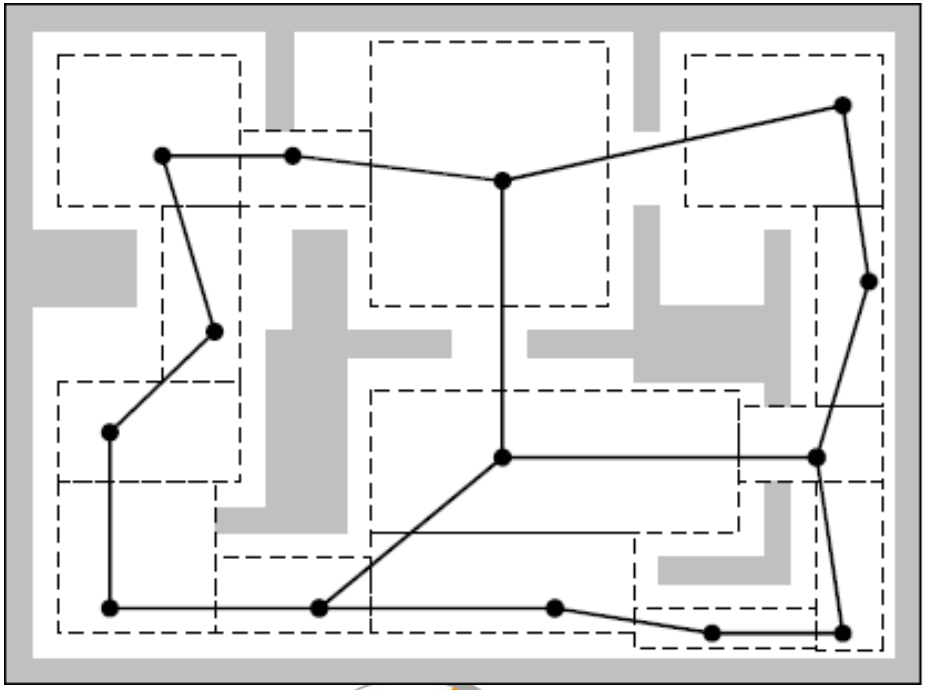
- Saber qual o melhor caminho entre dois nós somente é útil quando se sabe onde se deseja ir.

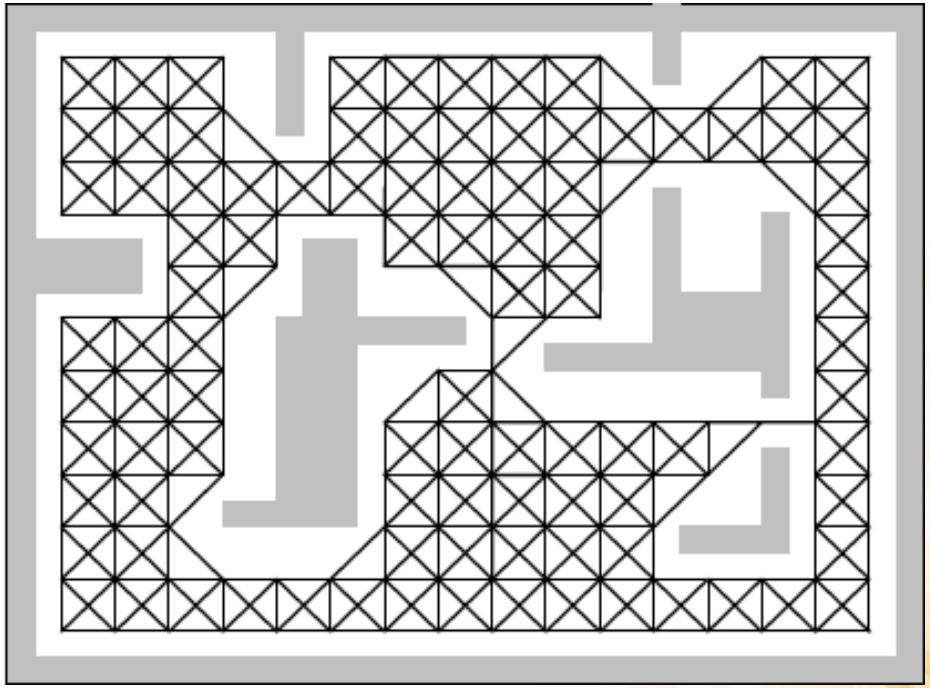
- Uma **tabela pré-calculada** com os **custos de locomoção** entre quaisquer dois nós também é uma informação muito util.

	A	B	C	D	E
A	0	3	2	4	5
B	3	0	3.5	1	3
C	2	3.5	0	4.5	6.5
D	4	1	4.5	0	2
E	5	3	6.5	2	0

Busca de Caminhos Hierárquica

- Esta estratégia consiste em achar o caminho de modo gradativo, partindo-se de uma solução grosseira até uma solução mais refinada.





Leitura Complementar

- Millington, I.; Funge, J.: **Artificial Intelligence for Games**, 2nd Ed., Morgan Kaufmann, 2009.