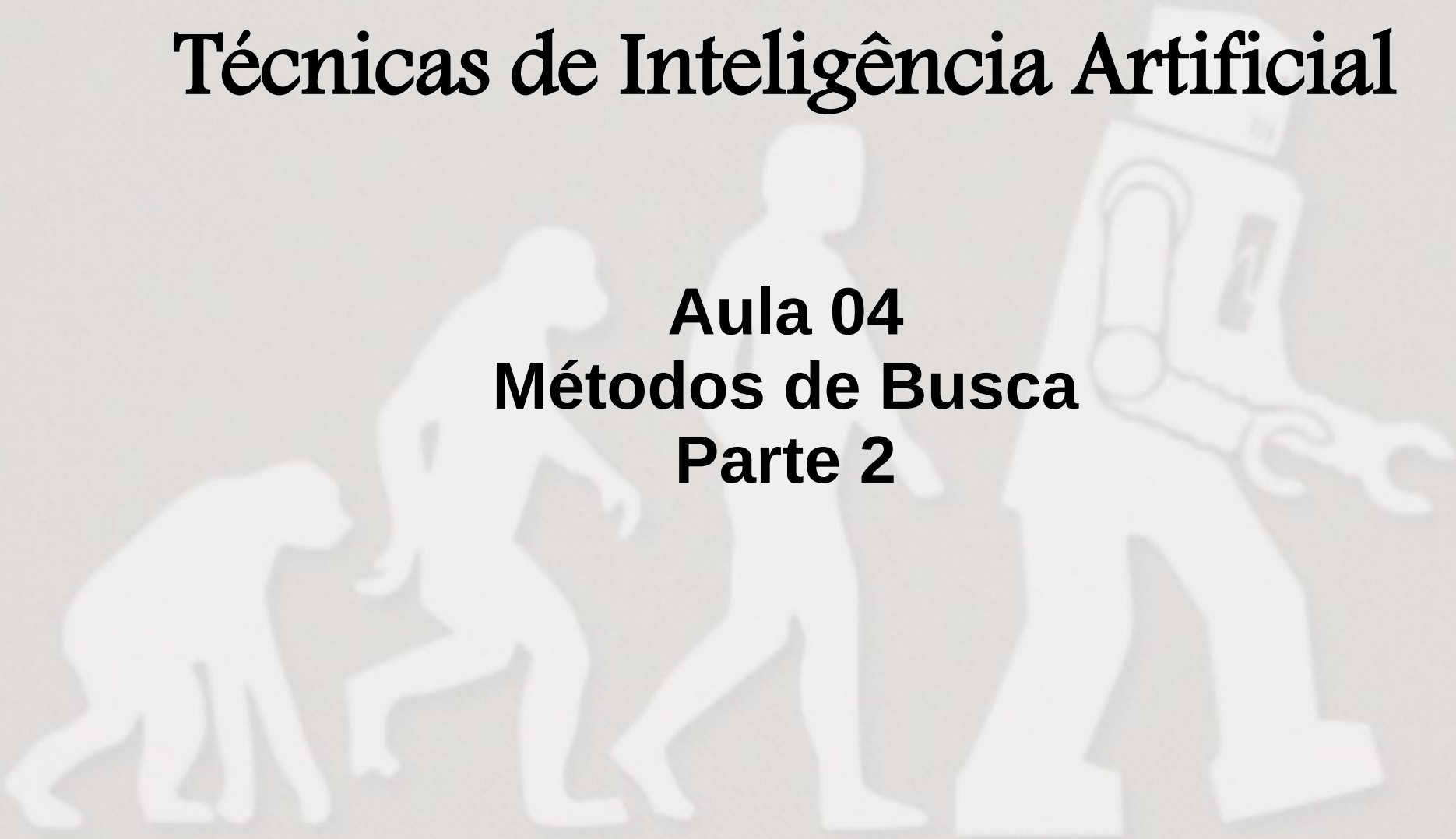
Universidade do Sul de Santa Catarina - UNISUL
Ciência da Computação

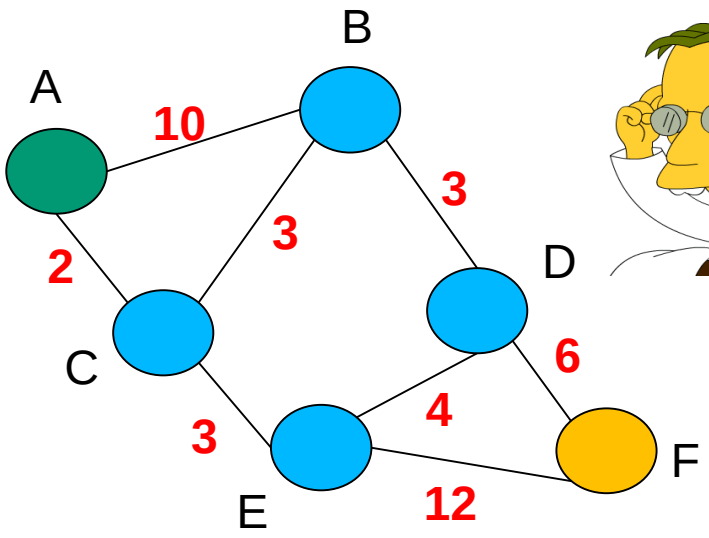
Técnicas de Inteligência Artificial



Aula 04
Métodos de Busca
Parte 2

Prof. Max Pereira

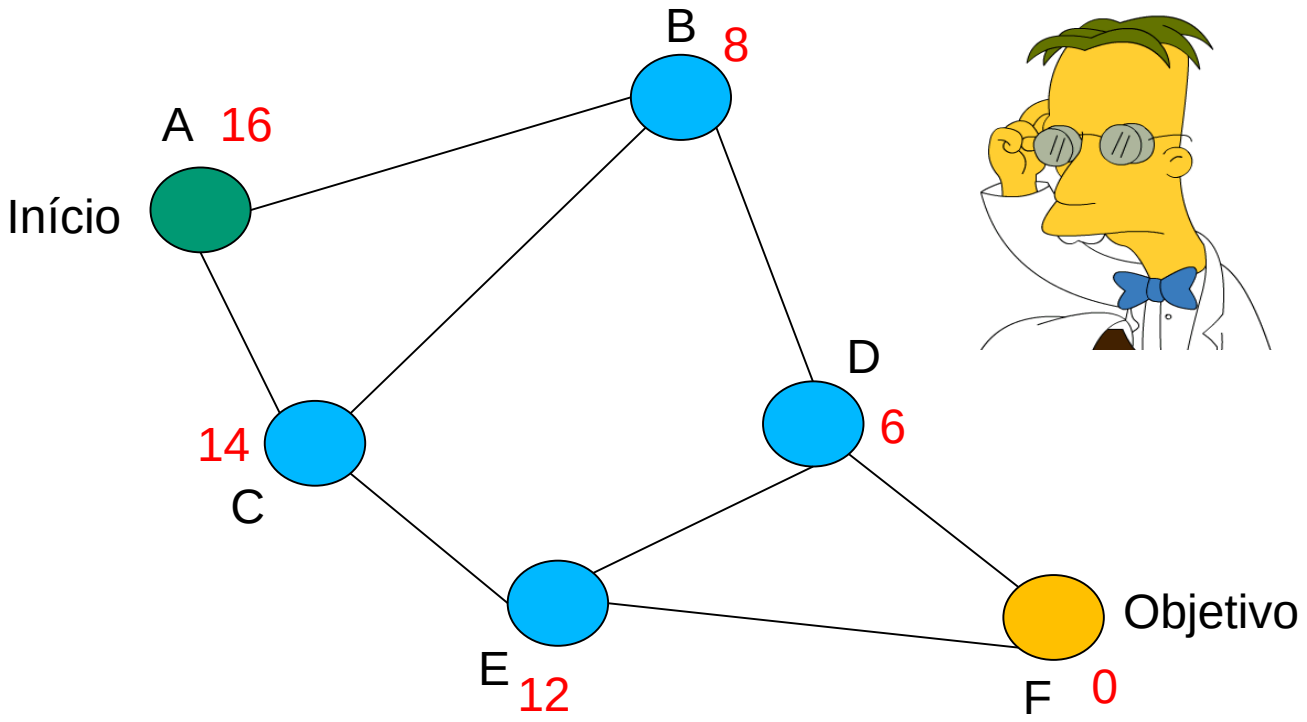
Relembrando...



Representação do espaço de busca como uma árvore de busca

Relembrando...

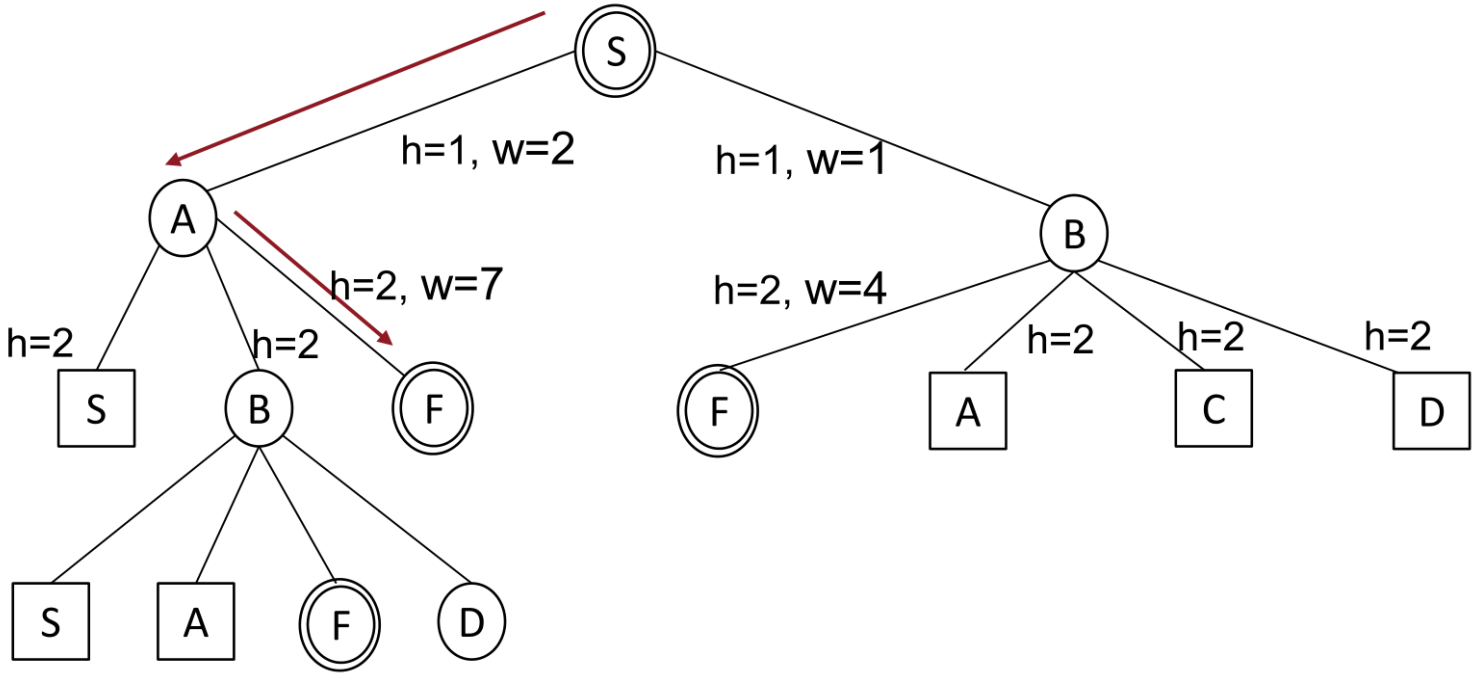
Aplicamos uma **heurística** à arvore de busca, que é a **distância em linha reta** de cada cidade à cidade objetivo.



Busca pelo Melhor Primeiro

Emprega uma **heurística** semelhante à **Subida da Colina**. A diferença é que, na busca pelo melhor primeiro, a **fila inteira é ordenada** após receber a inserção de novos caminhos, em vez de inserir um conjunto de caminhos ordenados.

Em termos práticos, isto significa que a busca pelo melhor primeiro **segue o melhor caminho disponível na árvore atual**, em vez de sempre seguir uma abordagem do tipo em profundidade.



Busca pelo Melhor Primeiro

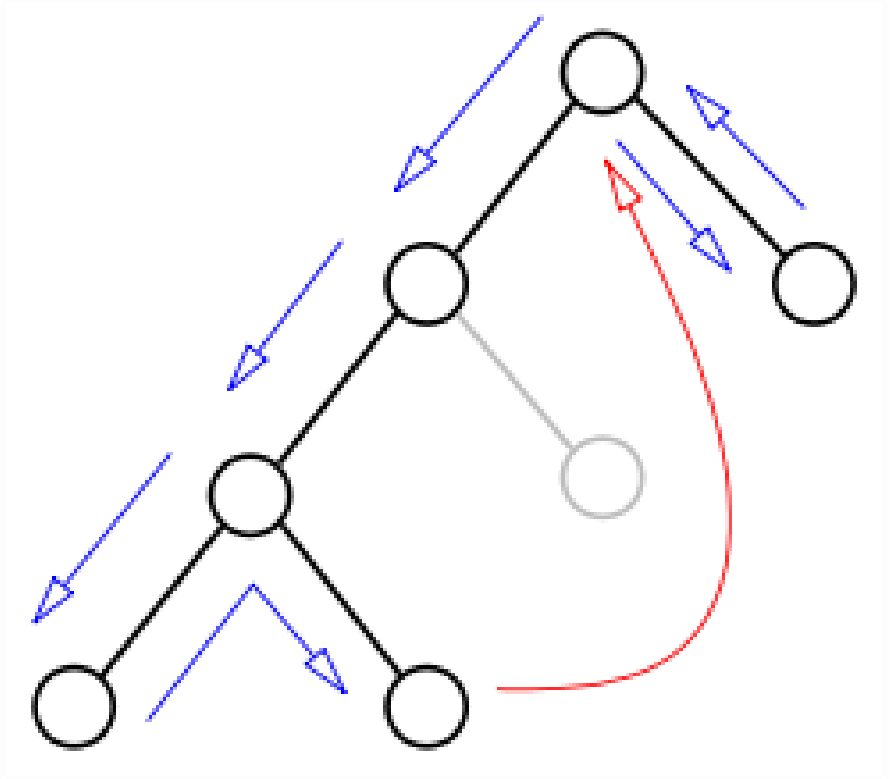
```
Function melhor()
{
      fila = [ ]; //inicializa uma fila vazia
      estado = no_raiz; //inicializa nó inicial
      while (true)
       {
            if eh_objetivo (estado)
               then return SUCESSO
            else
               inserir_na_frente_da_fila (sucessores (estado));
               ordenar (fila);
            if fila == [ ]
               then return FALHA;
            estado = fila [0]; //estado  = primeiro item na fila
            remover_primeiro_item_da (fila);
       }
}
```

Melhor caminho na nossa árvore de busca: A,B,D,F

Não é o caminho ótimo!

Identificando Caminhos Ótimos

O caminho ótimo é aquele que tem o menor custo ou envolve percorrer a distância mais curta do nó inicial ao nó objetivo. As técnicas descritas anteriormente podem encontrar o caminho ótimo por acidente, mas nenhuma delas garante encontrá-lo.



Algoritmos A*

Esses algoritmos são **similares** à **busca pelo melhor primeiro**, mas utilizam uma heurística um pouco mais complexa para selecionar um caminho na árvore.

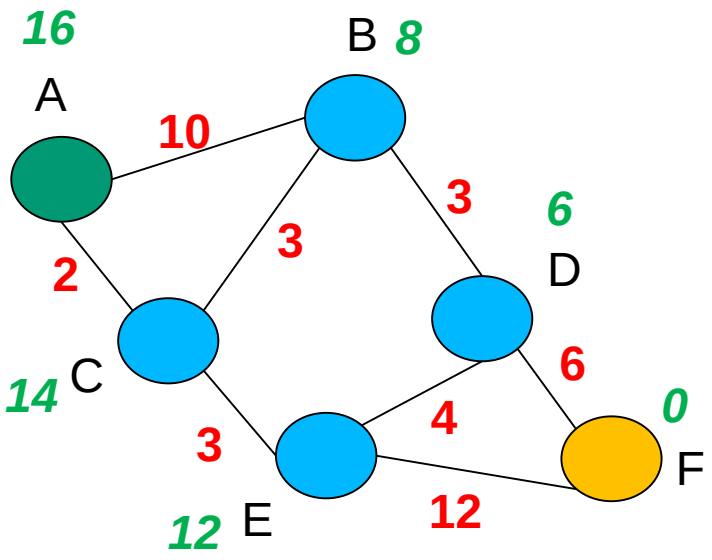
O algoritmo melhor primeiro sempre expande caminhos que envolvam ir para o nó que parece ser o mais perto do objetivo, mas sem considerar o **custo do caminho** até aquele nó.

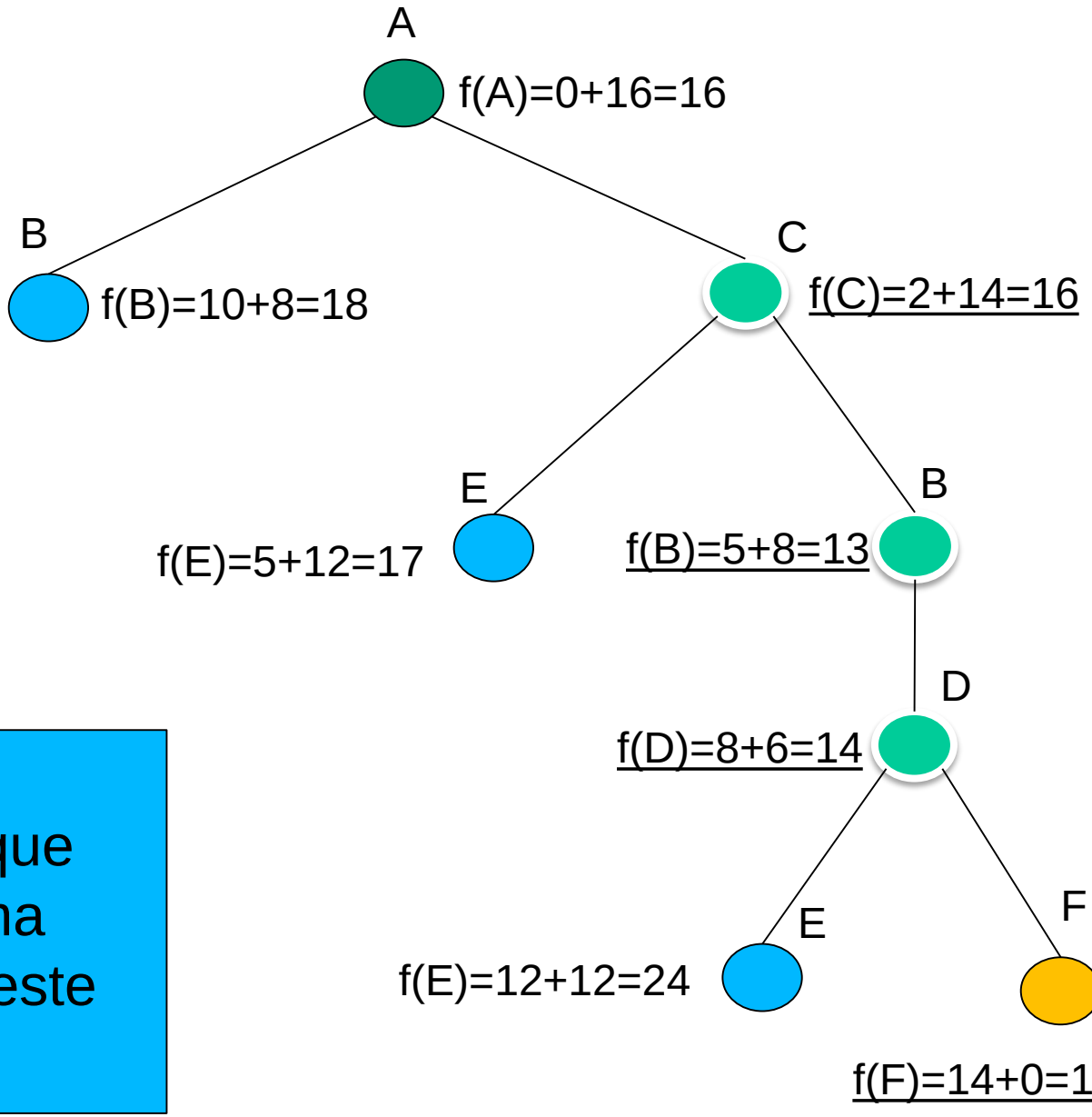
O algoritmo A* opera da mesma maneira que a busca pelo melhor primeiro, mas utiliza a seguinte função para avaliar os nós:

$f(nó) = g(nó) + h(nó)$

g(nó) é o custo do caminho que leva ao nó atual e *h(nó)* é uma subestimativa da distância deste nó até um estado objetivo.

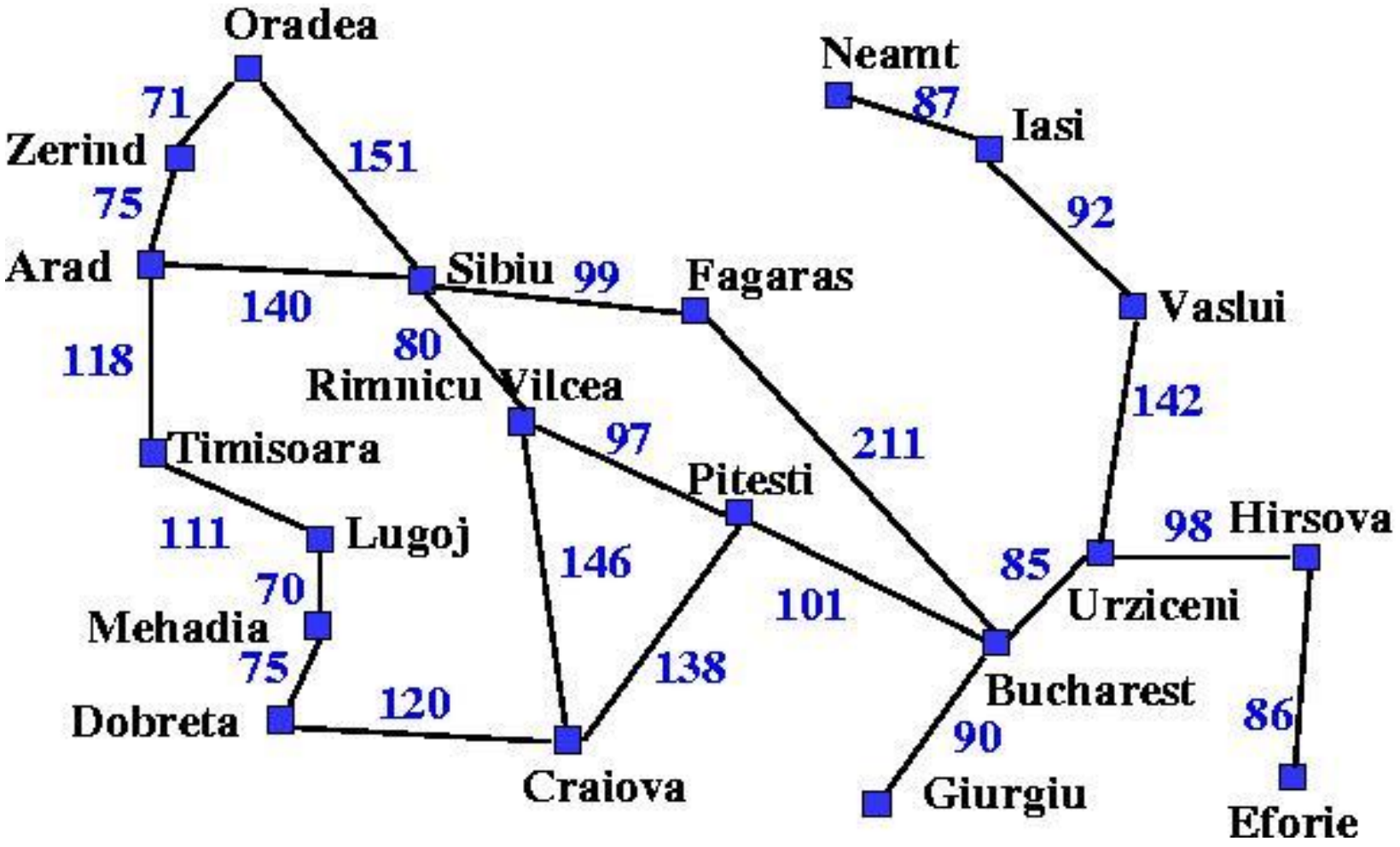
Algoritmo A*



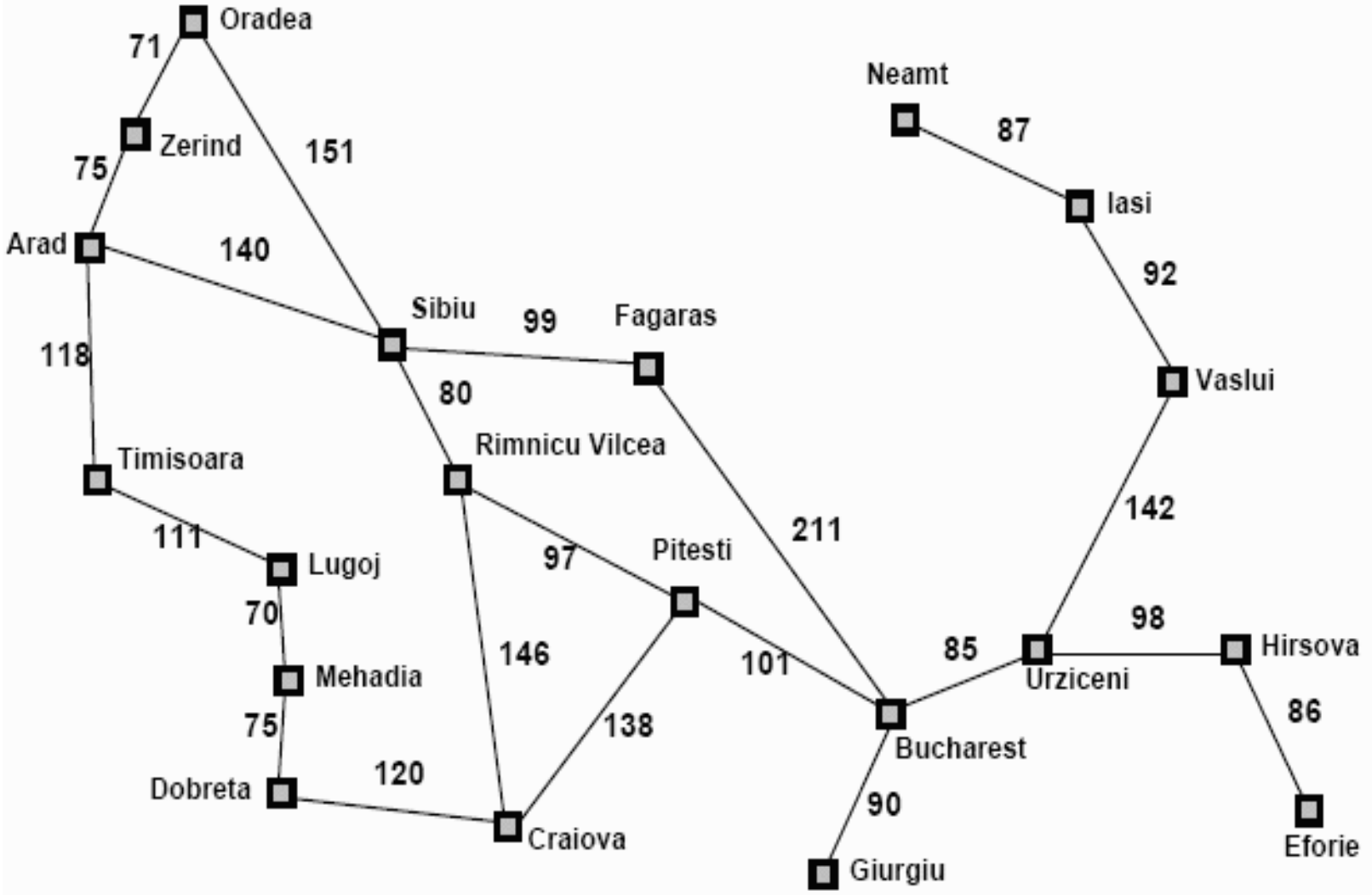


f(nó) = g(nó) + h(nó)
g(nó) é o custo do caminho que leva ao nó atual e *h(nó)* é uma subestimativa da distância deste nó até um estado objetivo.

Construir a árvore de busca e aplicar os algoritmos de *busca em profundidade*, *busca em largura*, *subida da colina*, *busca pelo melhor primeiro* e *A*.*



Mapa rodoviário simplificado de parte da Romênia. (Russel, Stuart J. e Norvig, Peter. **Inteligência Artificial,** 2ª ed, 2004. Elsevier, página 65).



Straight−line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374