

A*

Bruno Rafael Severo, Marcelo Lima e Thiago Miranda

https://github.com/m4rcelolima/P2_Huffman

Motivação

- Pathfinding.
- Expandir Dijkstra.
 - Heurística!
- Qual o melhor caminho entre duas cidades?

Definições

- Lista aberta e fechada.
- Peso nas arestas.
- Valor heurístico.
- $f(n) = g(n) + h(n)$
 - $f(n)$ = custo estimado da solução de custo mais baixo passando por “n”.
 - $g(n)$ = custo do caminho desde o nó inicial até o nó “n”.
 - $h(n)$ = custo estimado do caminho de custo mais baixo desde “n” até o objetivo.

Pseudo-Código

ListaAberta; // Lista de nós que serão analisados

ListaFechada; // Lista de nós que já foram analisados

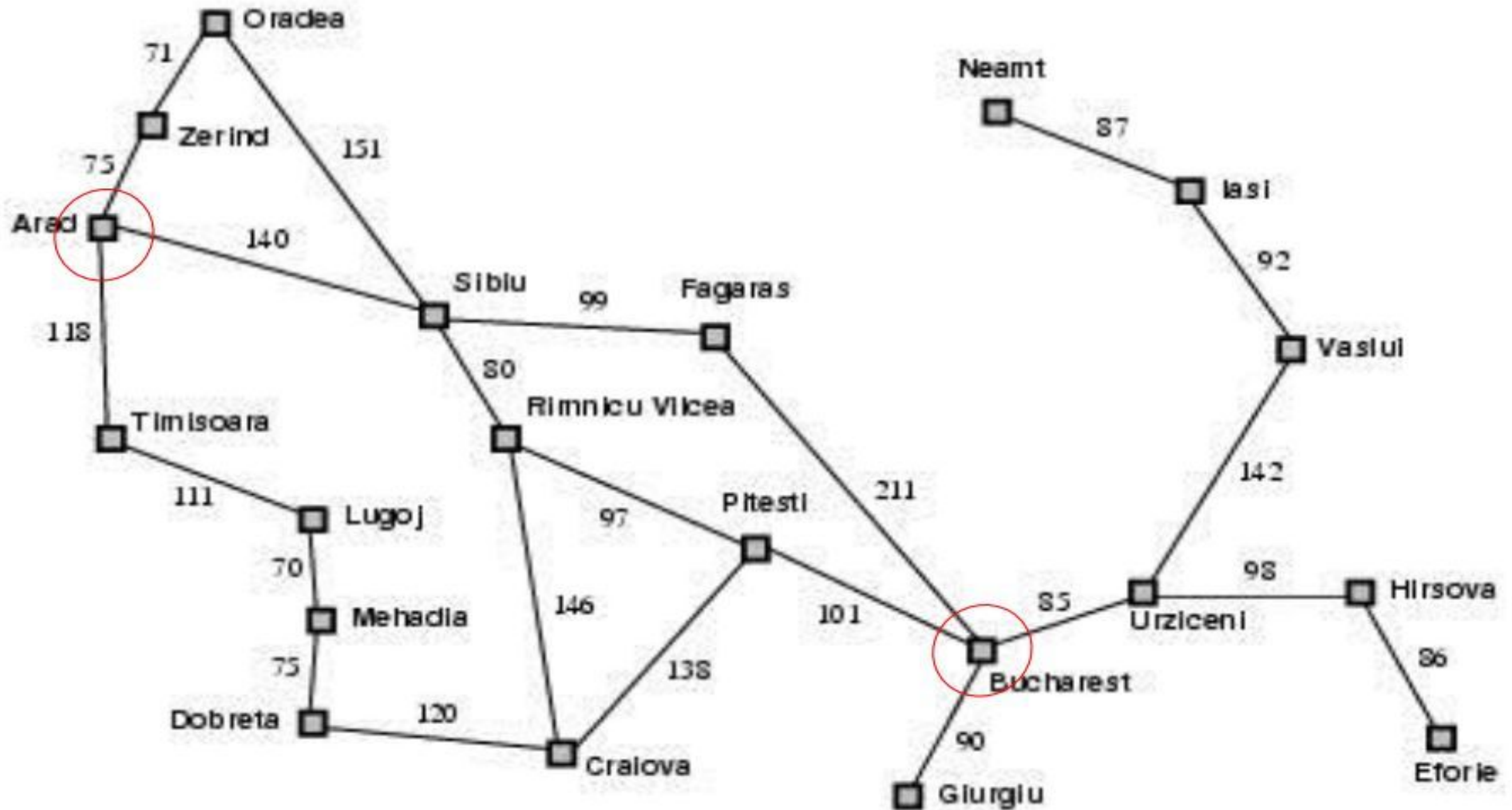
// Adicione o nó **INÍCIO** na **ListaAberta**;

add(**ListaAberta**, **INÍCIO**);

```
while(1){  
    ATUAL = getMin(ListaAberta); // nó da ListaAberta de menor F;  
    remove(ListaAberta, ATUAL); // remova ATUAL da  
    ListaAberta;  
    insere(ListaFechada, ATUAL); insere ATUAL na ListaFechada;  
  
    if(ATUAL == FIM) // Encontramos um caminho  
        printf("Caminho encontrado");  
        break;  
  
    if(isEmpty(ListaAberta)) // Caminho não encontrado  
        printf("Caminho não encontrado");  
        break;
```

```
for( i = 0; i < vizinhos; i++){  
    if(isListaFechada(VIZINHO) || isBlock(VIZINHO))  
        break;  
    if(!isListaAberta(VIZINHO)){  
        VIZINHO->parent = ATUAL;  
        calcule(G, H, F, VIZINHO);  
        insere(ListaAberta, VIZINHO);  
    }else if(isListaAberta(VIZINHO) &&  
        pastToMin(ATUAL,VIZINHO)){  
        calcule(G, 0, F, VIZINHO);  
        VIZINHO->parent = ATUAL;  
    }  
}  
} // Fim do While
```

Exemplo: Distância entre as cidades



Exemplo: Distância entre as cidades

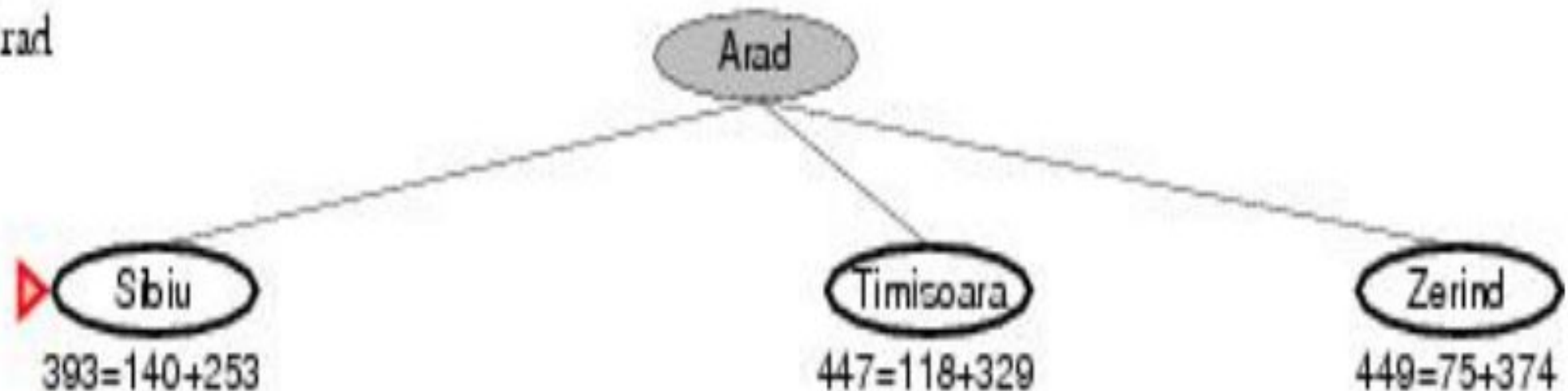
(a) The initial state



- Encontrar Bucareste partindo de Arad:
 - $f(\text{arad}) = 0 + h(\text{arad}) = 0 + 366 = 366$

Exemplo: Distância entre as cidades

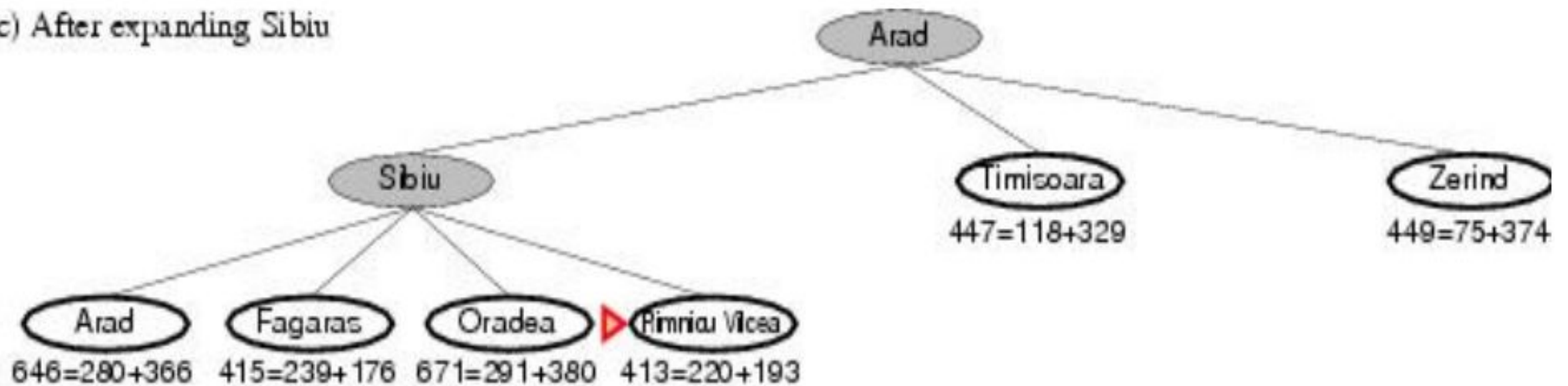
After expanding Arad



- Expandindo Arad e determinando $f(n)$ para cada nó:
 - $f(\text{Sibiu}) = c(\text{Arad}, \text{Sibiu}) + h(\text{sibiu}) = 140 + 253 = 393$
 - $f(\text{Timisoara}) = c(\text{Arad}, \text{Timisoara}) + h(\text{Timisoara}) = 118 + 329 = 447$
 - $f(\text{Zerind}) = c(\text{Arad}, \text{Zerind}) + h(\text{Zerind}) = 75 + 374 = 449$
- Melhor escolha é Sibiu

Exemplo: Distância entre as cidades

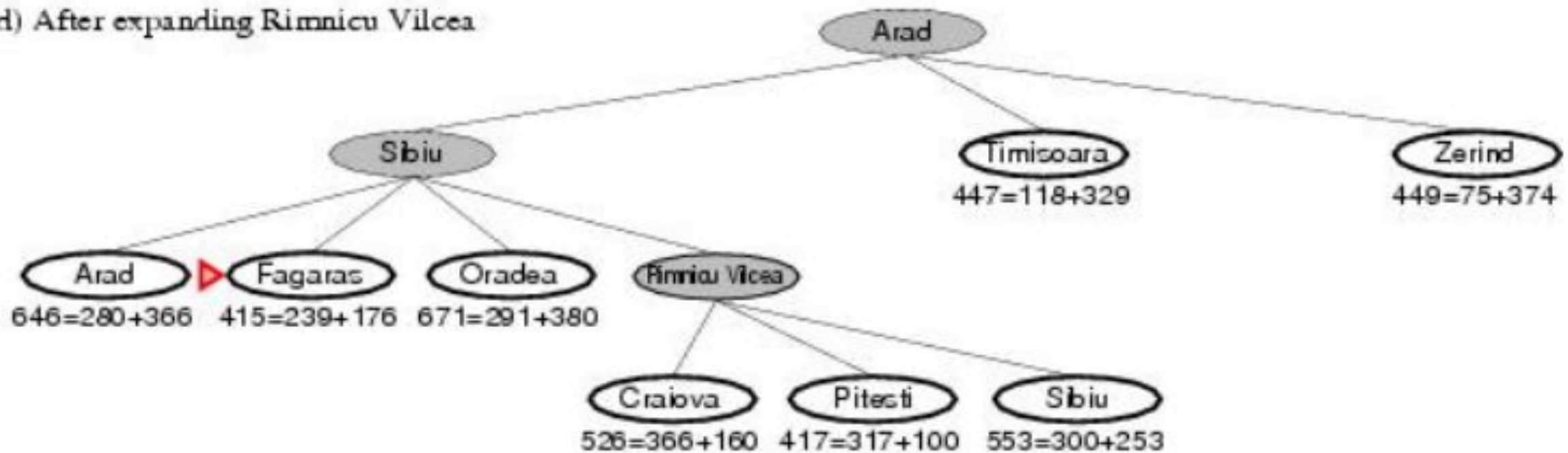
(c) After expanding Sibiu



- Expandindo Sibiu e determinado $f(n)$ para cada nodo:
 - $f(\text{Arad}) = g(\text{Arad}) + h(\text{Arad}) = 280 + 366 = 646$
 - $f(\text{Fagaras}) = g(\text{Fagaras}) + h(\text{Fagaras}) = 239 + 176 = 415$
 - $f(\text{Oradea}) = g(\text{Oradea}) + h(\text{Oradea}) = 291 + 380 = 671$
 - $f(\text{Rimnicu Vilcea}) = g(\text{Rimnicu Vilcea}) + h(\text{Rimnicu Vilcea}) = 220 + 193 = 413$
- Melhor escolha é Rimnicu Vilcea

Exemplo: Distância entre as cidades

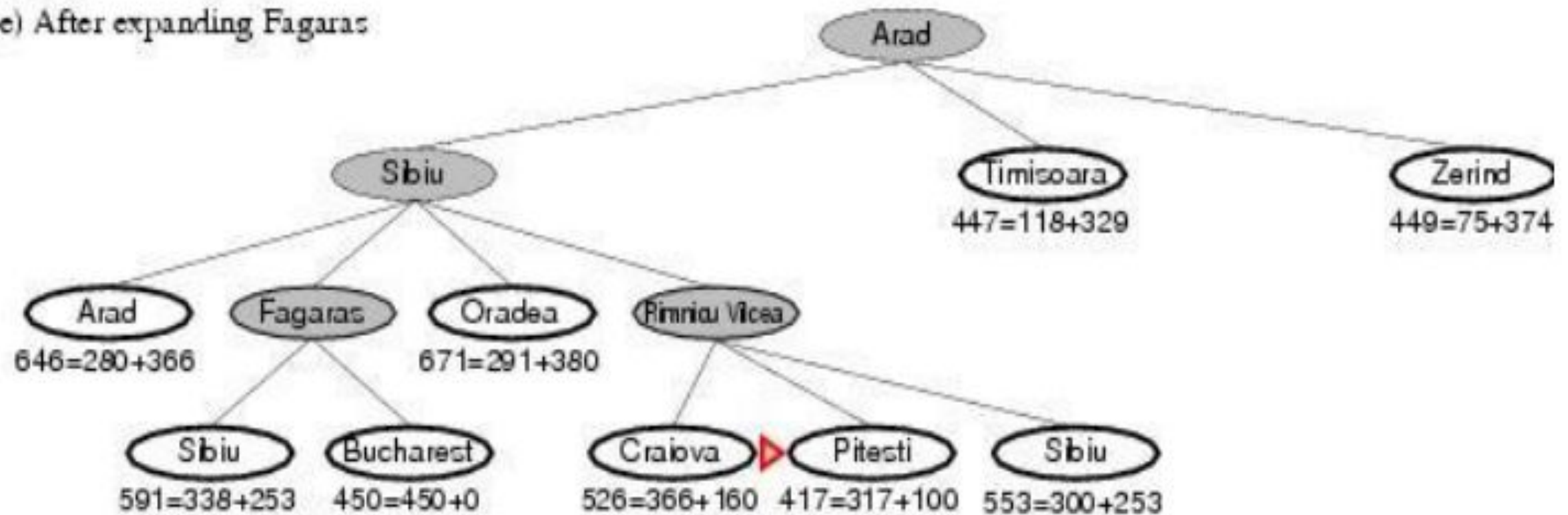
(d) After expanding Rimnicu Vilcea



- Expandindo Rimnicu Vilcea:
 - $f(\text{Craiova}) = g(\text{Craiova}) + h(\text{Craiova}) = 366 + 160 = 526$
 - $f(\text{Pitesti}) = g(\text{Pitesti}) + h(\text{Pitesti}) = 317 + 100 = 417$
 - $f(\text{Sibiu}) = g(\text{Sibiu}) + h(\text{Sibiu}) = 300 + 253 = 553$
- Melhor escolha é Fagaras.

Exemplo: Distância entre as cidades

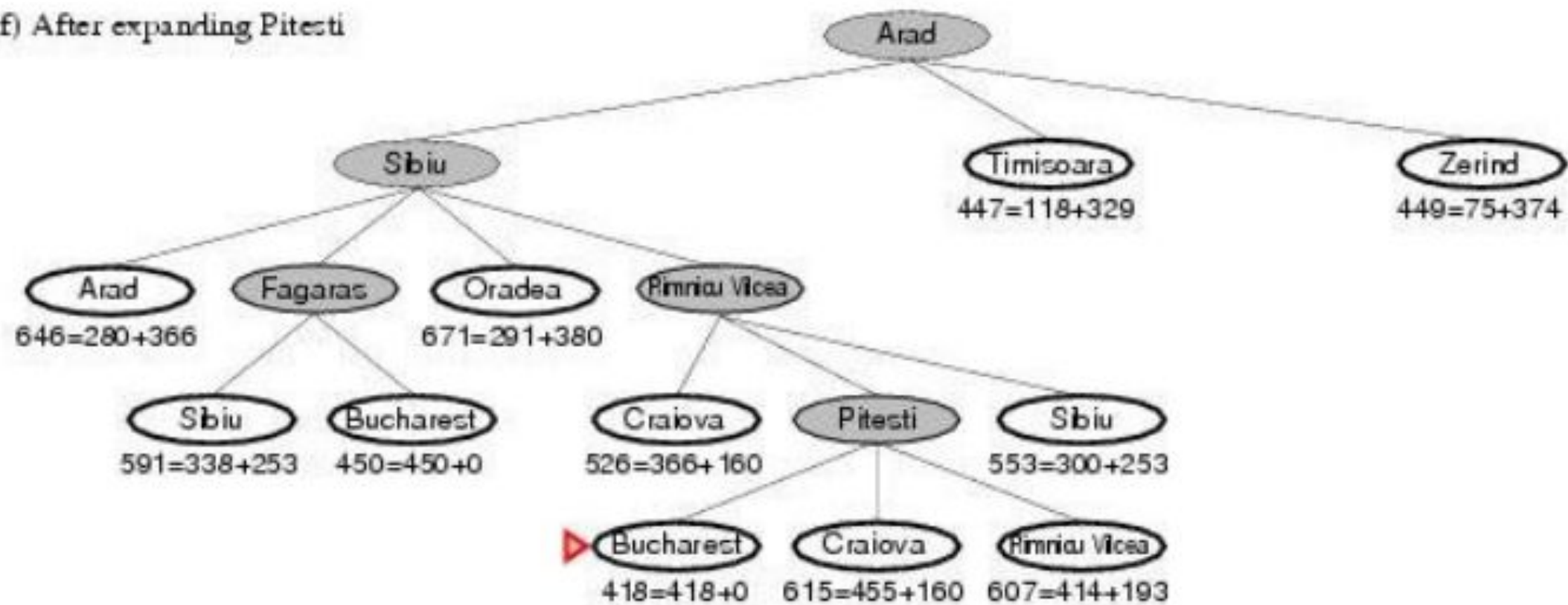
(e) After expanding Fagaras



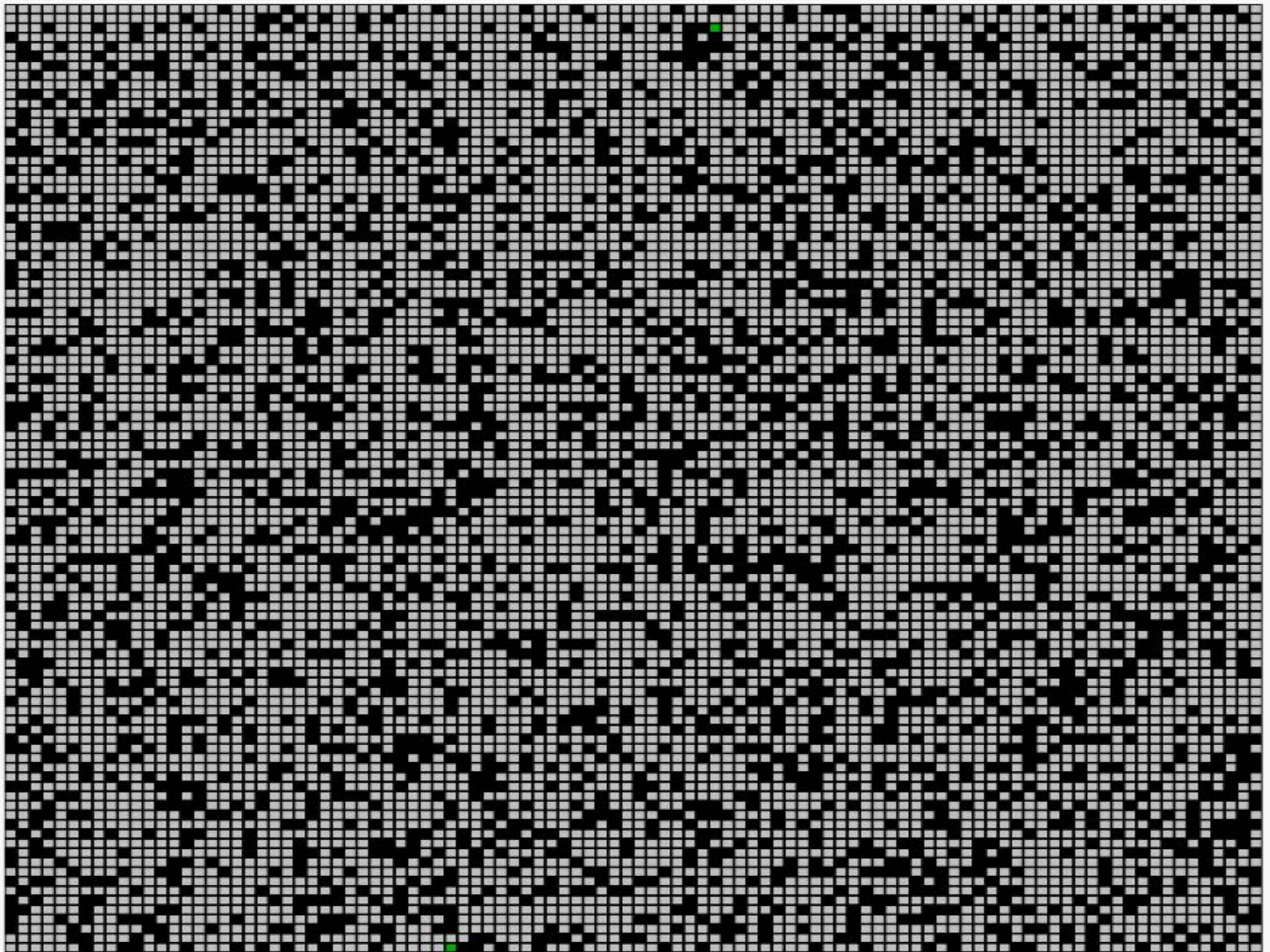
- Expandindo Fagaras:
 - $f(\text{Sibiu}) = g(\text{Sibiu}) + h(\text{Sibiu}) = 338 + 253 = 591$
 - $f(\text{Bucareste}) = g(\text{Bucareste}) + h(\text{Bucareste}) = 450 + 0 = 450$
- Melhor escolha é Pitesti

Exemplo: Distância entre as cidades

(f) After expanding Pitesti



- Expandindo Pitesti:
 - $f(\text{Bucareste}) = g(\text{Bucareste}) + h(\text{Bucareste}) = 418 + 0 = 418$
- Melhor escolha é Bucareste.
 - Solução ótima(dado que $h(n)$ é admissível).



Complexidade

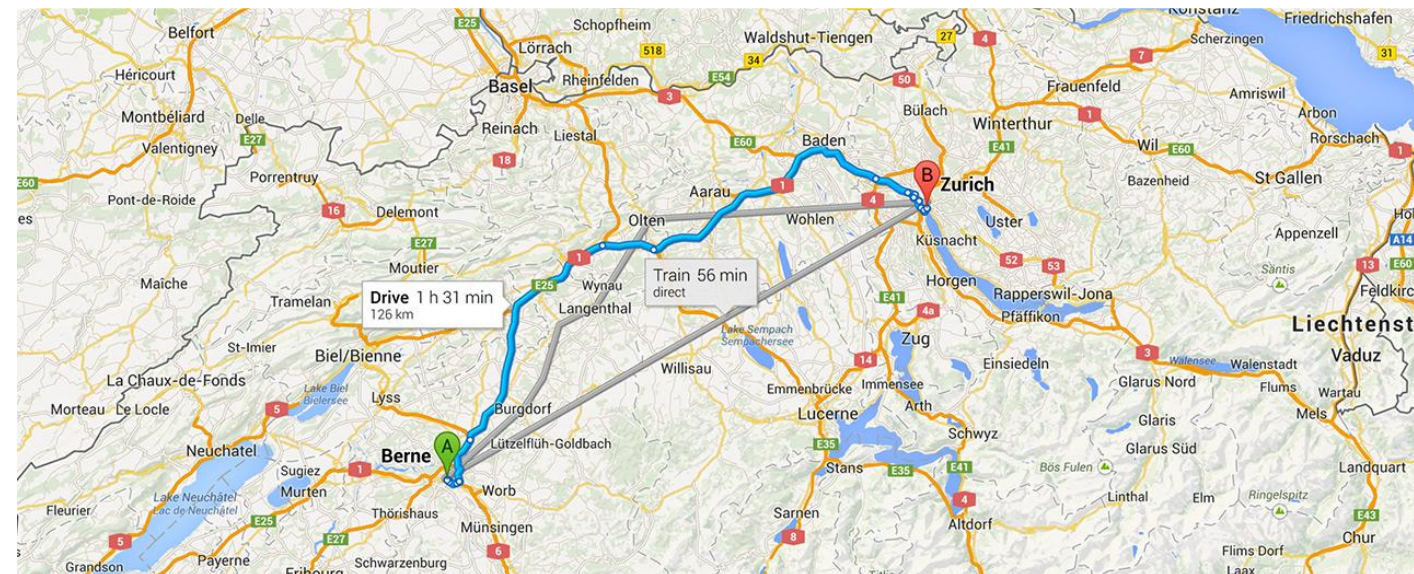
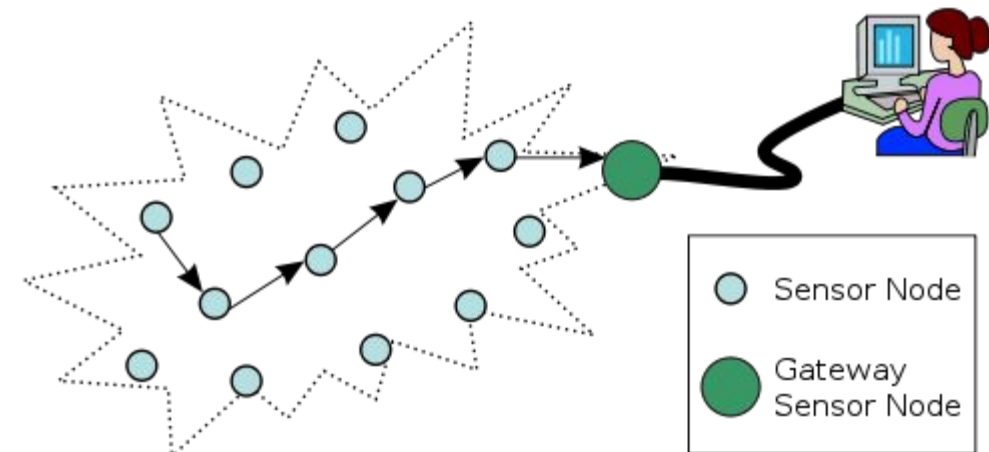
- Depende da Heurística.
- $O(b^n)$
 - b = fator de ramificação.
 - n = número de nós expandidos.
 - Exponencial!
 - Melhor a heurística, menor b .
- $|h(x) - h^*(x)| = O(\log(h^*(x)))$
 - Espaço de procura é uma árvore.
 - Polinomial!

De volta à Motivação...

- O algoritmo encontrou uma solução admissível, pois $h(n) \geq h^*(n)$.
- Heurística adicionada a Dijkstra
- Solução:
 - Completa.
 - Ótima.

Aplicações

- <https://www.youtube.com/watch?v=hmx1sx6ezQA>
- Many others!



Limitações

- Depende do cálculo da heurística.
- Não é prática para problemas de grande escala.
- Em geral, esgota o espaço antes de esgotar o tempo.