Inteligência Artificial

Busca com informação

Prof. Fabio Augusto Faria

Material adaptado de Profa. Ana Carolina Lorena e livro
“Inteligência Artificial, S. Russell e P. Norving”

1º semestre 2021

Estratégias de busca sem informação

- Encontram soluções:
  - Gerando sistematicamente novos estados e
  - Comparando-os com o objetivo

- São muito ineficientes na maioria dos casos
  - Estratégias de busca com informação
    - Usam conhecimento específico do problema
    - Podem encontrar soluções de maneira mais eficiente

Busca com informação

- ***Busca heurística***
  - Utiliza conhecimento específico do problema
    - Além de definição do próprio problema

Tentativa de expandir os caminhos mais promissores primeiro

Heurística auxilia a encontrar os nós mais promissores a cada passo

  – Heurística é a função que estima distância ao objetivo

Busca com informação

- Uma abordagem geral: melhor escolha primeiro
  - Expande nós com base em função de avaliação f(n)
    - Mede distância até o objetivo, considerando heurística
    - Nó com *avaliação* ***mais baixa*** é selecionado para expansão
  - Vários algoritmos
    - Funções de avaliação diferentes

- Implementação: Introduz na fila de nós a serem expandidos de acordo com f(n) (fila de prioridades)

Busca com informação

- Algoritmo melhor escolha primeiro

fronteira ← Inserir (Nó (Estado-Inicial [problema]))

Repita

se fronteira está vazia **então retorna** falha

nó ← Remove-Primeiro (fronteira)

se Teste-Término [problema] aplicado a Estado [nó] tiver sucesso

então retorna nó

fronteira ← InserirDeAcordoF(fronteira, Expandir[problema, nó])

(Insere novos nós na fronteira ordenados pela função f)

fim

Busca com informação

- Componente fundamental: função heurística h(n)

  Estima custo do caminho de menor custo de n até um nó objetivo

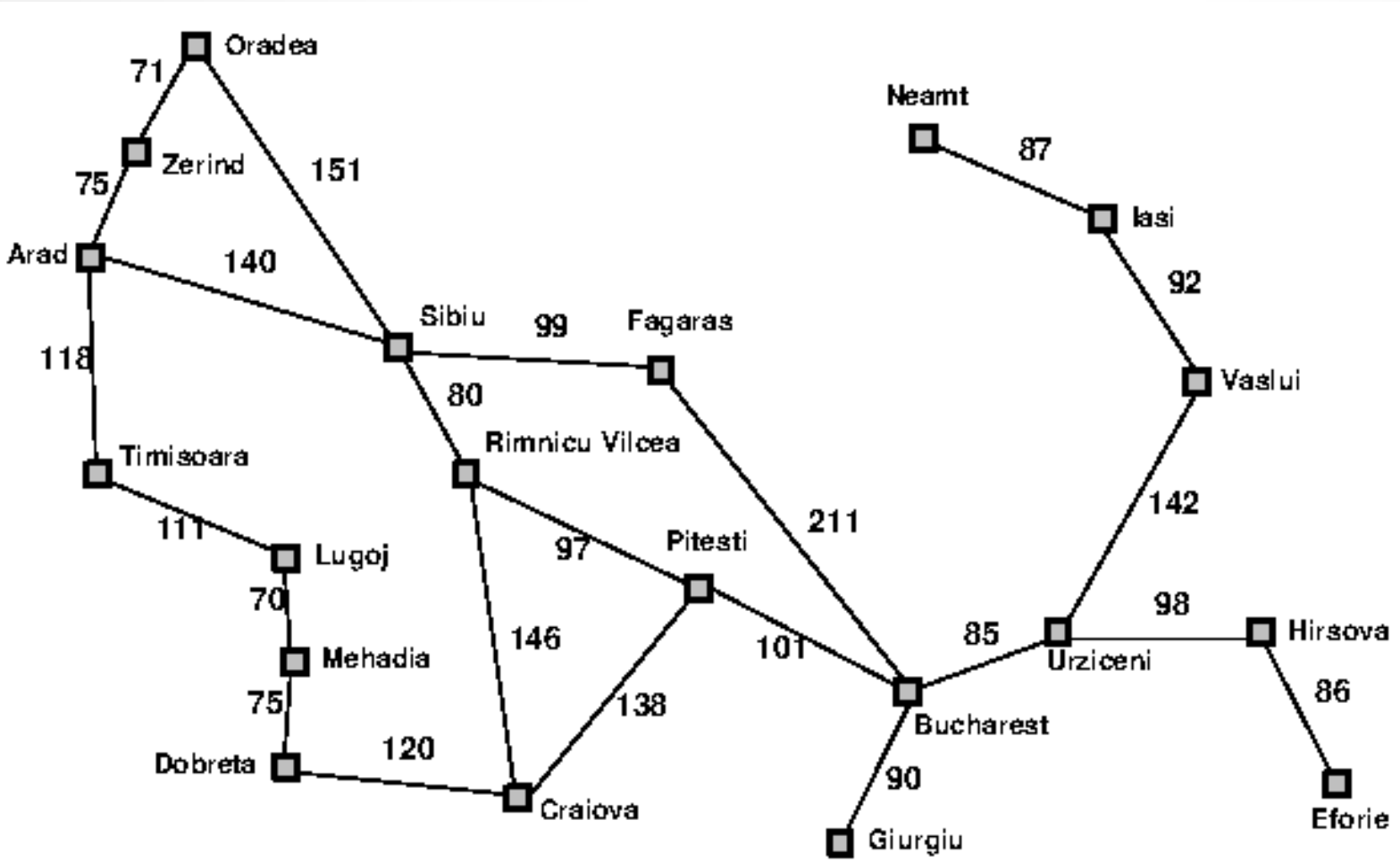
  - Exemplo: Arad a Bucareste
    - Distância em linha reta entre essas cidades
  - Forma mais comum de adicionar conhecimento do problema
  - Específica para cada problema
    - *Restrição*: se n é um nó objetivo, h(n) = 0

Busca gulosa

- Tenta expandir nó mais próximo à meta
  - Supondo que provavelmente levará a uma solução rápida
  - Avalia nós usando função heurística apenas
    - f(n) = h(n)

Exemplo

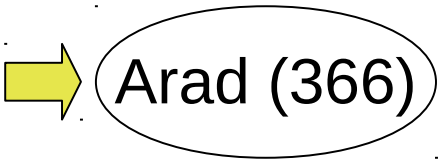
- Ir de Arad a Bucareste
  - Heurística de distância em linha reta h_{DLR}



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

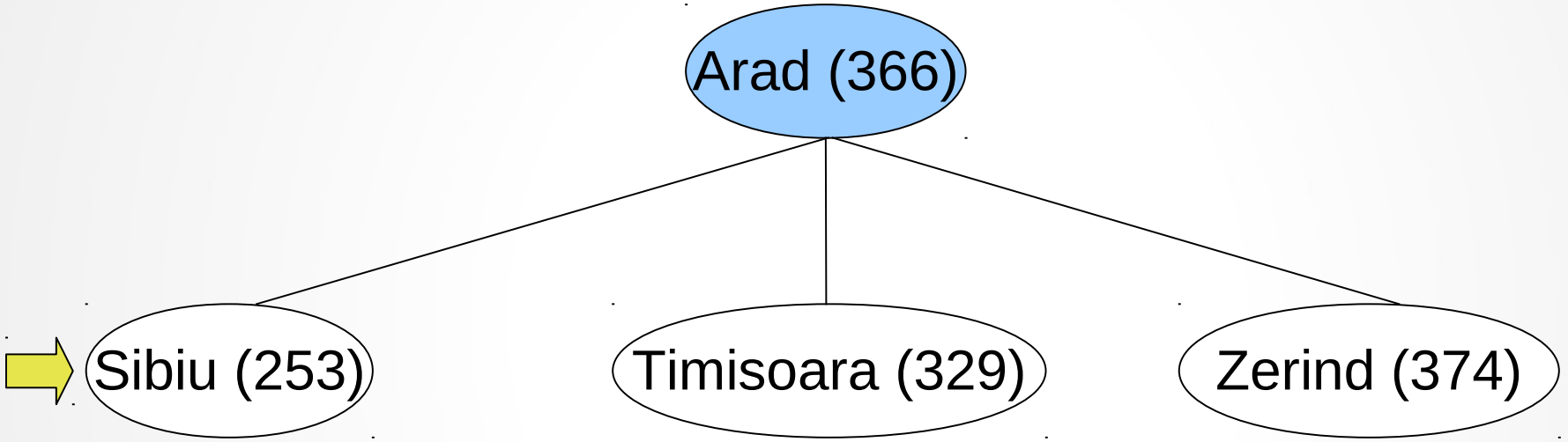
Exemplo

- ***(a) Estado inicial***



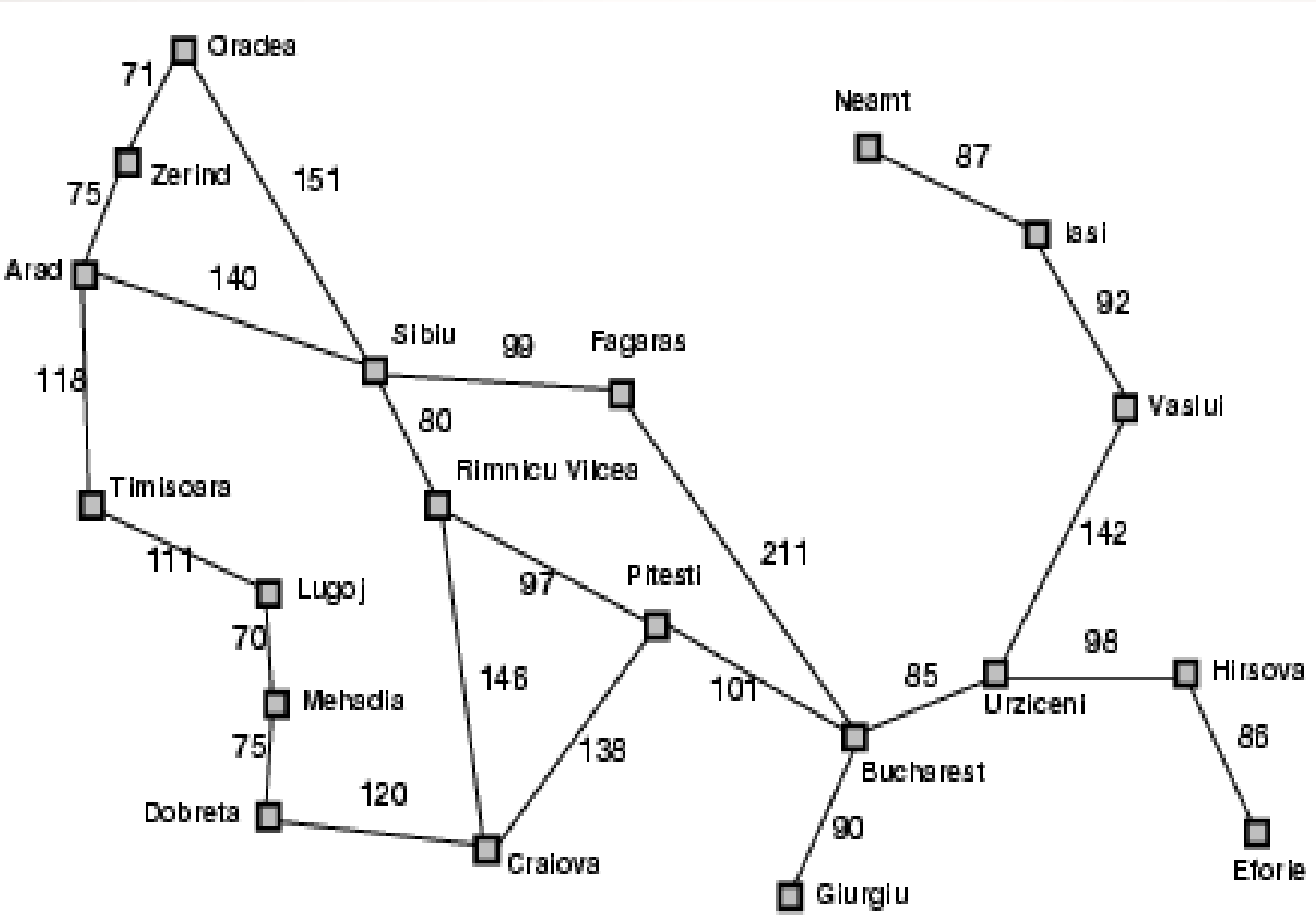
Exemplo

- ***(b) Expansão de Arad***



Exercício: continuar a aplicar a busca gulosa

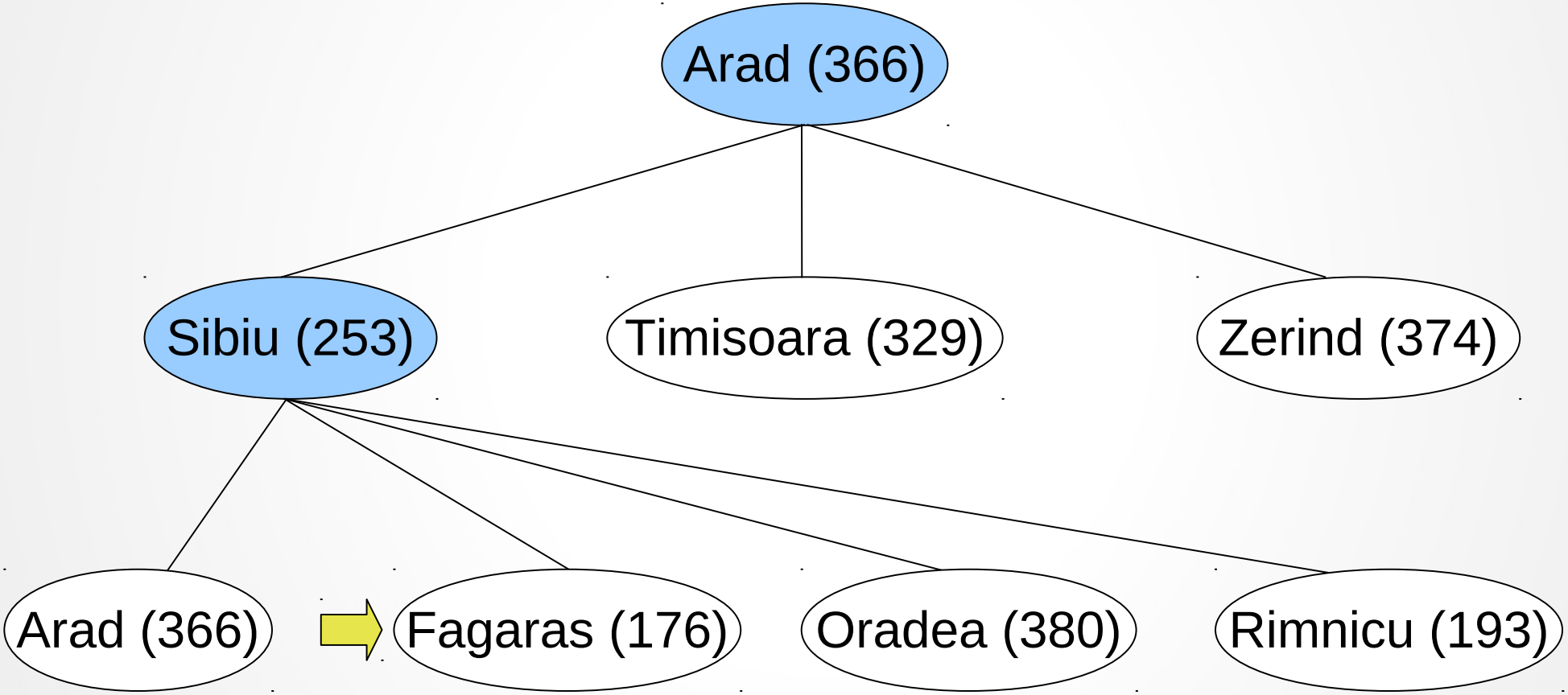
Exemplo



Straight–line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	100
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

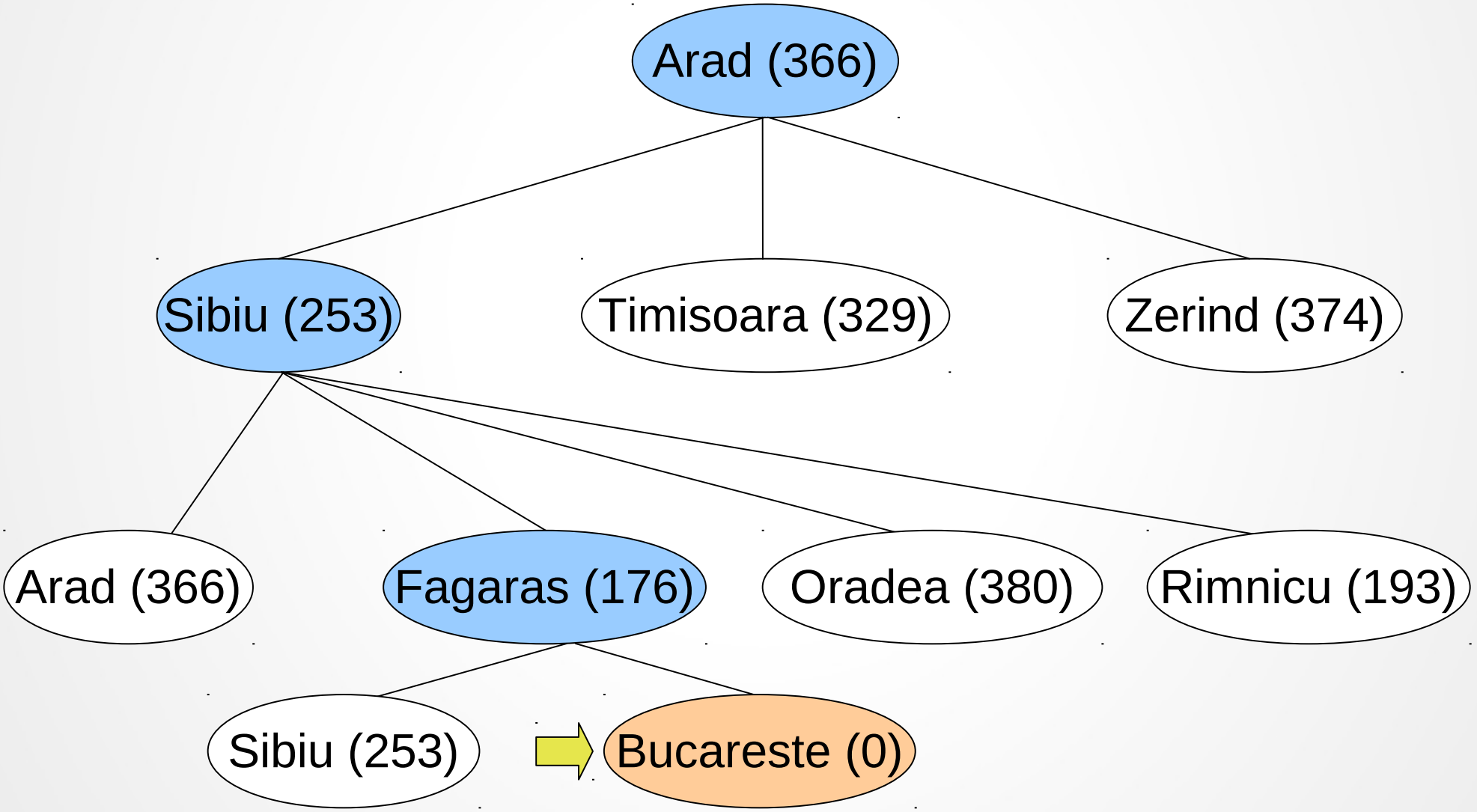
Exemplo

- ***(c) Expansão de Sibiu***



Exemplo

- ***(c) Expansão de Fagaras***



Exemplo

- Encontrou solução sem expandir nenhum nó que não estivesse no caminho da solução

- Contudo, solução **não é ótima**
  - 32 km mais longo do que por Rimniciu e Pitesti

- Nomenclatura guloso
  - Em cada passo, tenta chegar o mais perto possível do objetivo
  - Não é ótima, pois segue o melhor passo **considerando somente o momento atual**
  - pode haver um caminho melhor seguindo algumas opções piores em alguns pontos da árvore de busca

Busca gulosa

- Minimizar h(n) é suscetível a **falsos inícios**
  - Ex.: ir de Iasi a Fagaras
    - Busca gulosa com h_{DLR}:
      - Iasi ⇨ Neamt ⇨ Iasi ⇨ Neamt ⇨ …
    - Solução:

      - Iasi ⇨ Vaslui ⇨ Urziceni ⇨ Bucareste ⇨ Fagaras

        - Vaslui é mais distante que Neamt do objetivo de acordo com a heurística
        - Mas é o caminho que liga a Fagaras (por Neamt não tem caminho)

Busca gulosa

- Semelhante à busca em profundidade
  - Prefere seguir em um único caminho
- É incompleta
  - Pode entrar em caminho infinito
- Não é ótima
- Complexidade tempo no pior caso: $O(b^m)$
  - m é a profundidade máxima do espaço de busca
    - Com boa heurística pode ter redução substancial

Busca A*

- Minimizando custo total estimado da solução
  - Avalia nós combinando:
    - g(n): custo real do caminho para alcançar cada nó
      - Custo de nó inicial até o nó n (valor exato)
    - h(n): custo estimado para ir do nó até o objetivo
      - Custo estimado do caminho de n ao objetivo

$$f(n) = g(n) + h(n)$$

Custo estimado da solução "mais barata" passando por n

Ideia: evitar expandir caminhos que já ficaram caros

Busca A*

- Encontrar solução de custo mais baixo
  - Escolher primeiro nó com menor valor f(n)

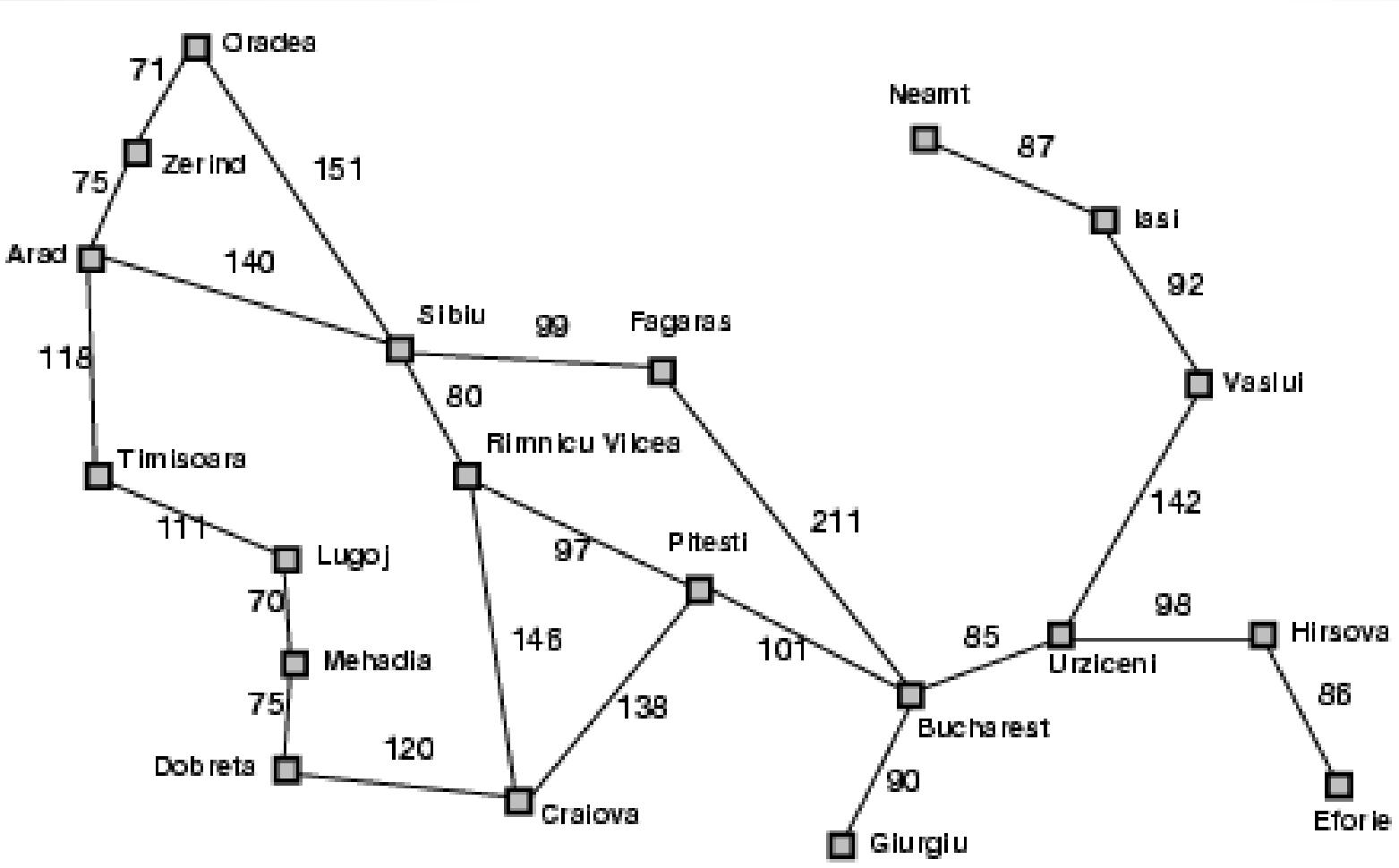
- *Se a função heurística h(n) satisfaz algumas condições, A* é completa e ótima*
  - Em busca em árvore, é *ótima* se h(n) for heurística admissível
    - Nunca superestima o custo para alcançar o objetivo
    - Heurística otimista: supõe que custo da resolução do problema é menor do que ele é na realidade
  - Assim, f(n) nunca irá superestimar o custo verdadeiro de uma solução, já que g(n) é o valor exato

Busca A*

- Ida de Arad a Bucareste
  - h_{DLR} é admissível
    - Caminho mais curto entre dois pontos quaisquer é uma linha reta
    - Usando:
      - g(n) a partir dos custos das estradas na figura
      - h(n) como h_{DLR}

Exemplo

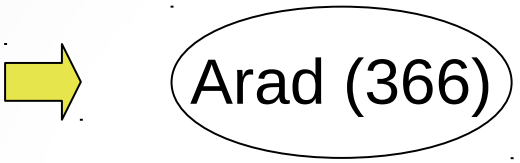
- Ir de Arad a Bucareste
  - Heurística de distância em linha reta h_{DLR}



Straight–line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	100
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Exemplo

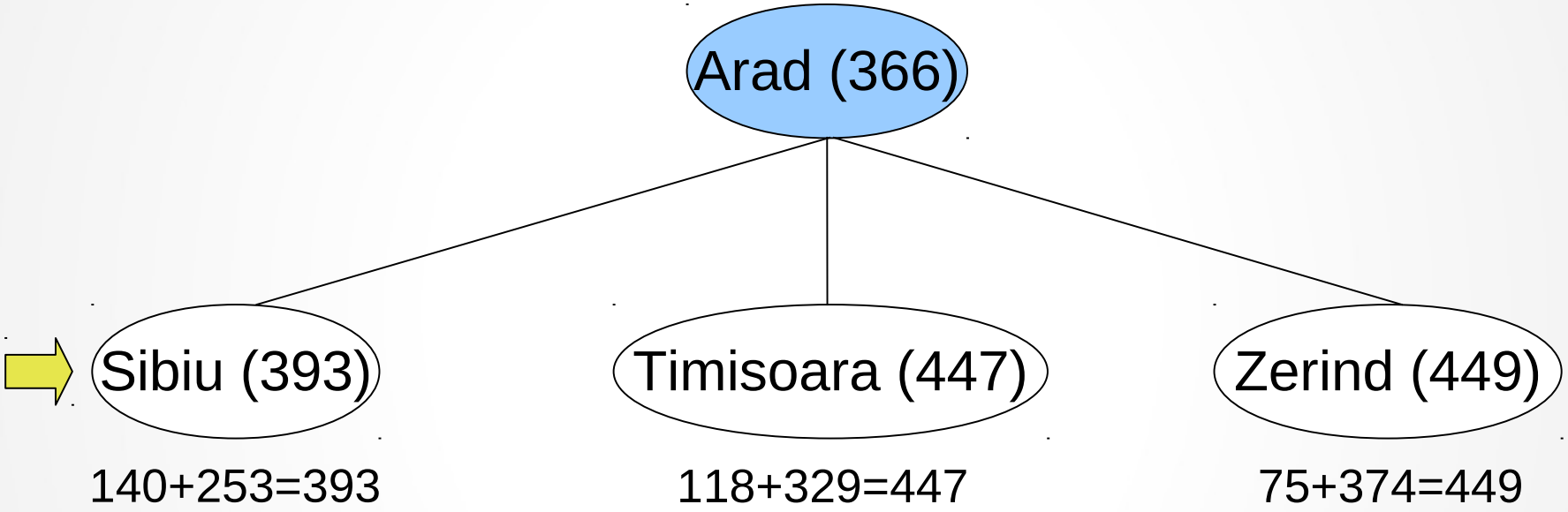
- ***(a) Estado inicial***



0+366=366

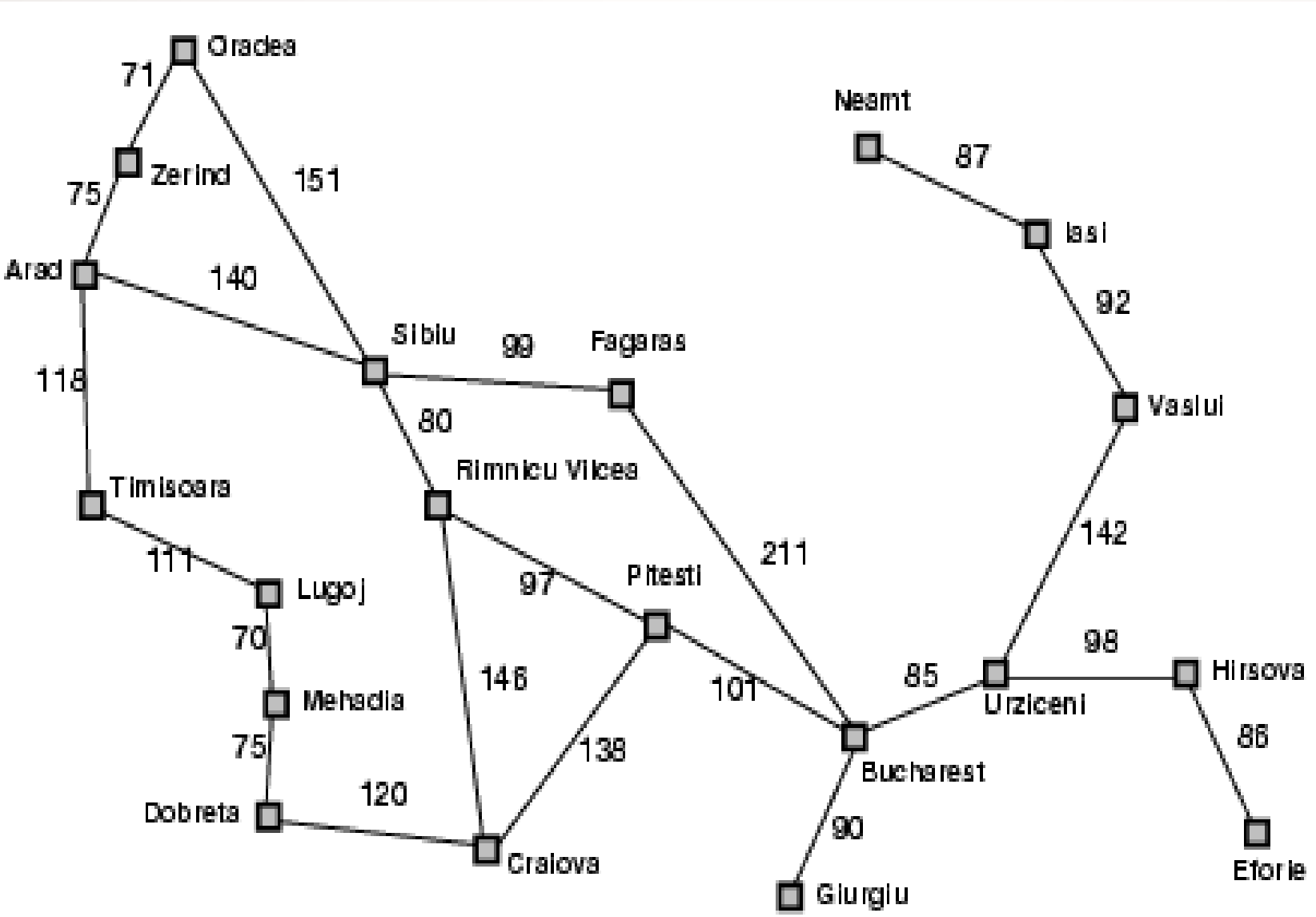
Exemplo

- ***(b) Expansão de Arad***



Exercício: continuar a aplicar a busca A*

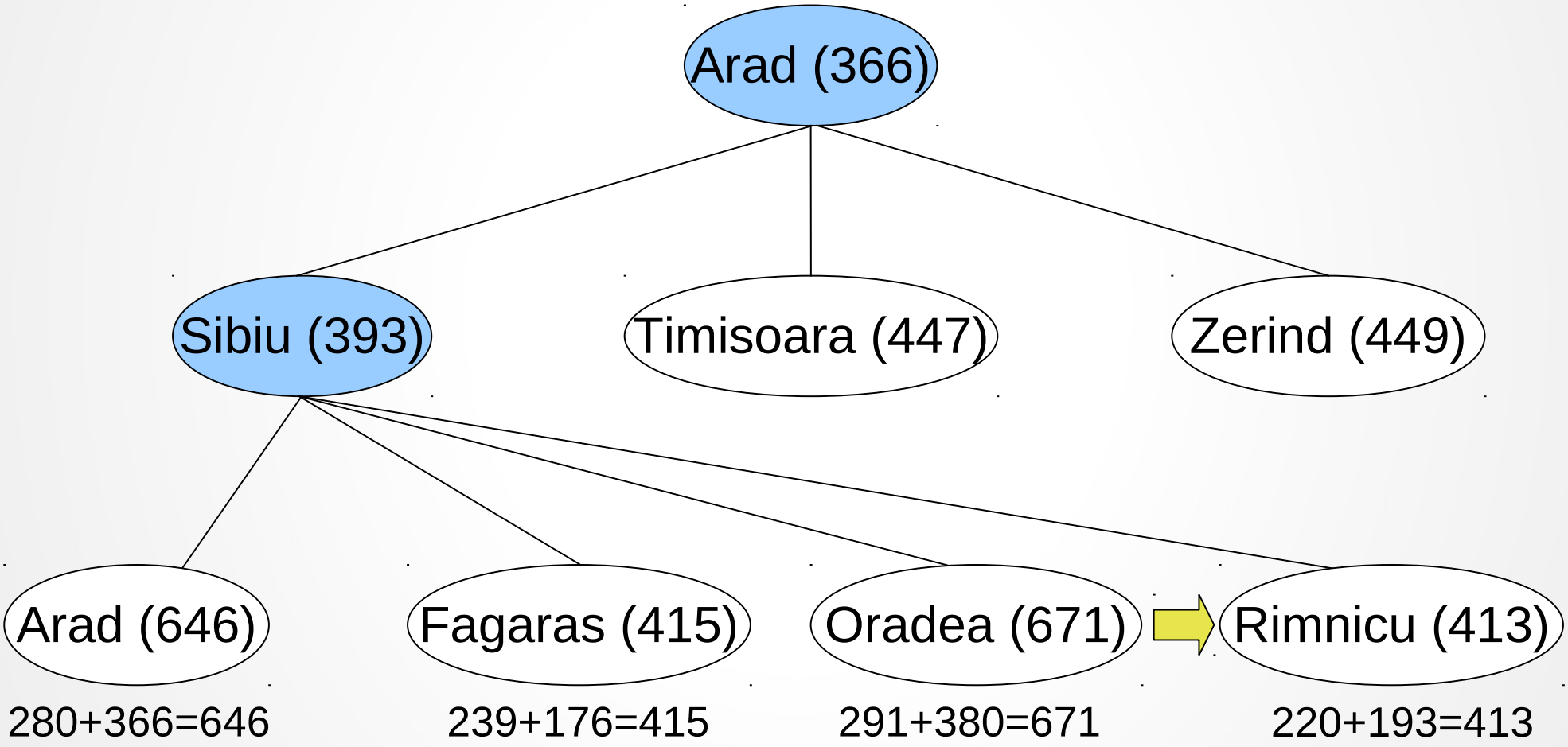
Exemplo



Straight–line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	100
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

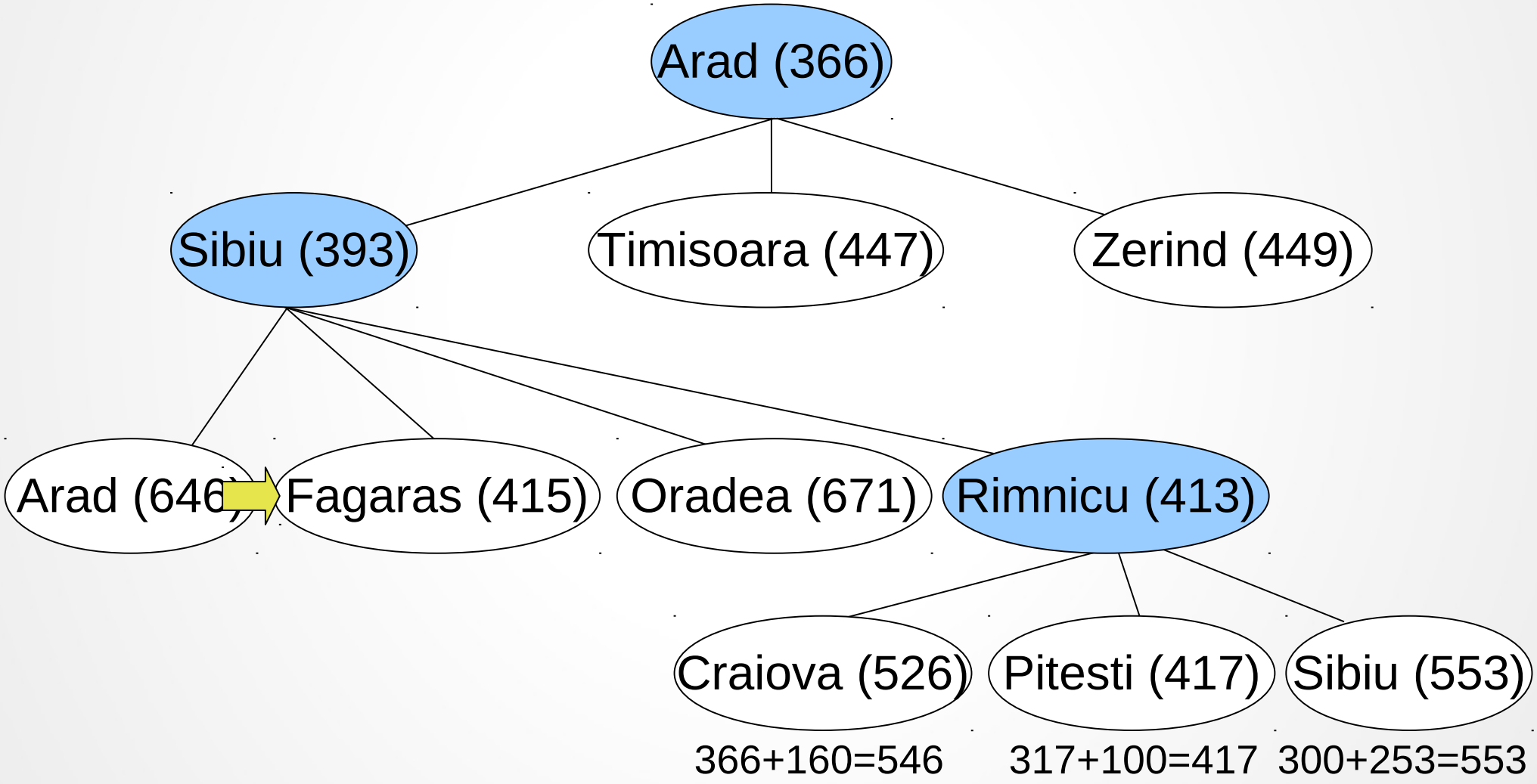
Exemplo

- ***(c) Expansão de Sibiu***



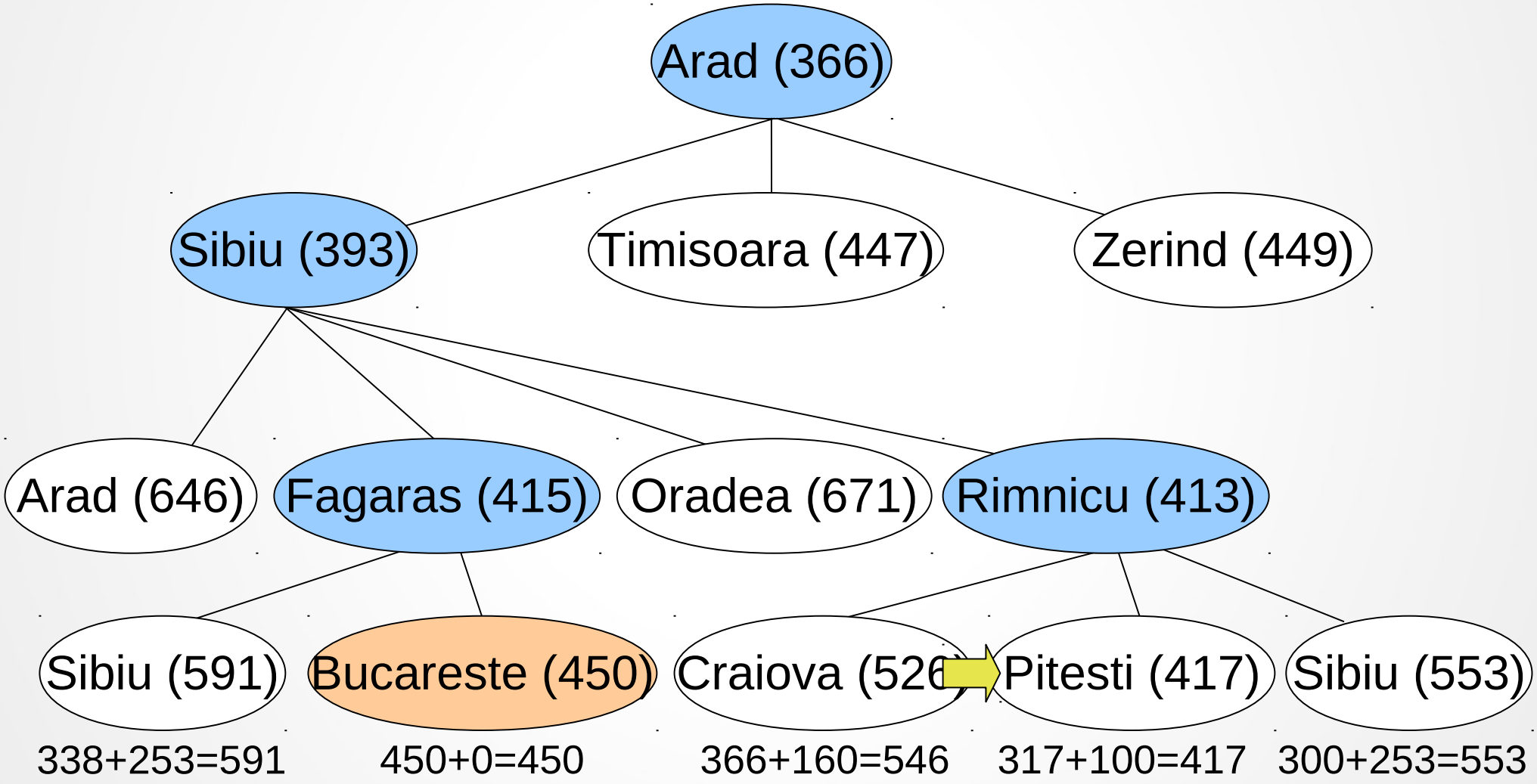
Exemplo

- ***(d) Expansão de Rimnicu***



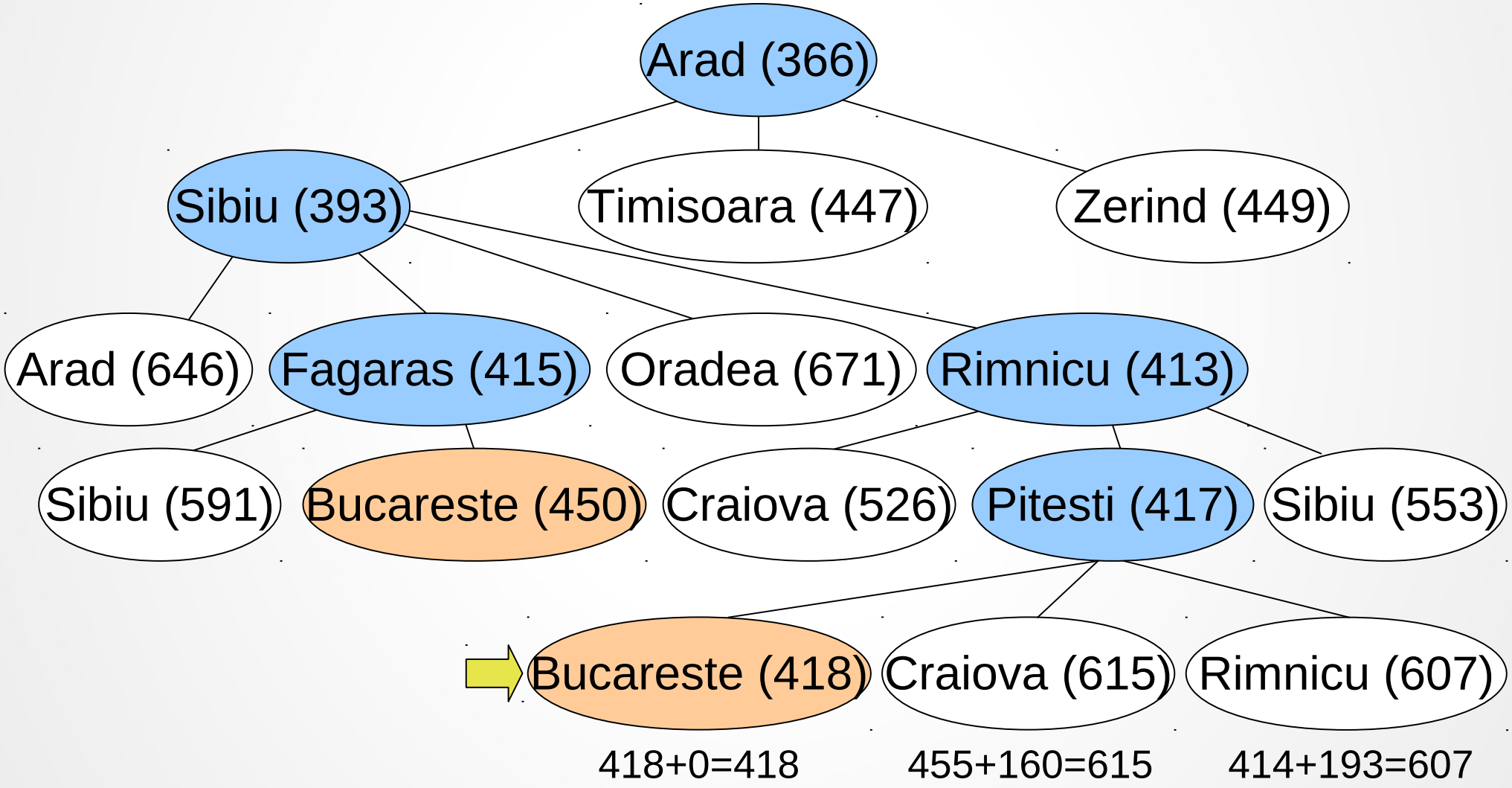
Exemplo

- ***(e) Expansão de Fagaras***



Exemplo

- ***(f) Expansão de Pitesti***



Busca A*

- Suponha que um nó objetivo não ótimo G2 apareça
  - Como no exemplo, em que Bucareste apareceu após expansão de Fagaras
    - Mas caminho era maior do que passando por Pitesti
  - Seja C* o custo da solução ótima
    - Como G2 não é ótimo e h(G2) = 0

$$f(G2) = g(G2) + h(G2) = g(G2) > C^*$$

Busca A*

- Considere G1 um nó de borda que está no caminho da solução ótima
  - No exemplo, Pitesti
  - Se h não superestima o custo de completar o caminho da solução, então:

$$f(G1) = g(G1) + h(G1) \leq C^*$$

Busca A*

- Combinando conclusões:

$$f(G1) \leq C^* < f(G2)$$

  - Então G2 não será expandido e
  - A* deve retornar uma solução ótima

Busca A*

- É **completa**
  - b deve ser finito
- É **otimamente eficiente** para qualquer função heurística
  - Nenhum outro algoritmo ótimo tem a garantia de expandir um número de nós menor que A*

Busca A*

- Complexidade de tempo ainda é exponencial na maioria dos casos
  - $O(b^d)$, em que d é o nível da solução mais barata mais rasa
- Complexidade de espaço é exponencial
  - Mantém todos os nós gerados na memória
  - Em geral A* esgota espaço bem antes de tempo

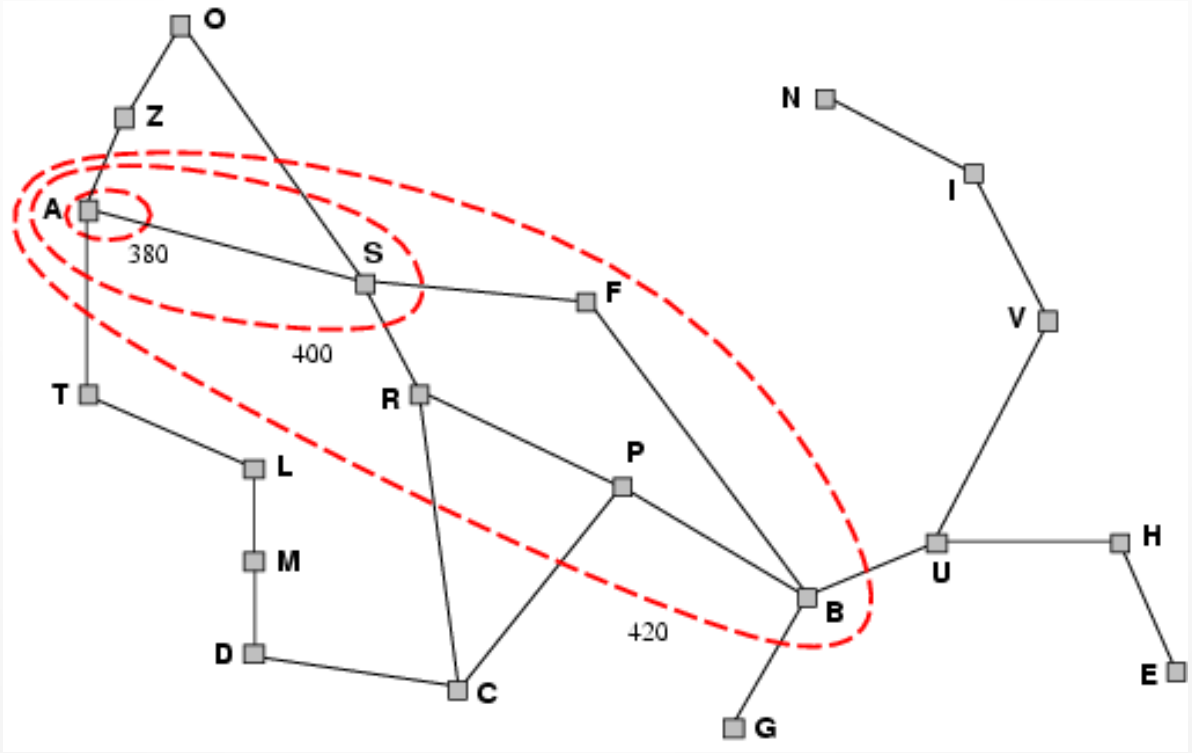
A* iterativo

Como custo de espaço de A* é grande, usar uma versão iterativa

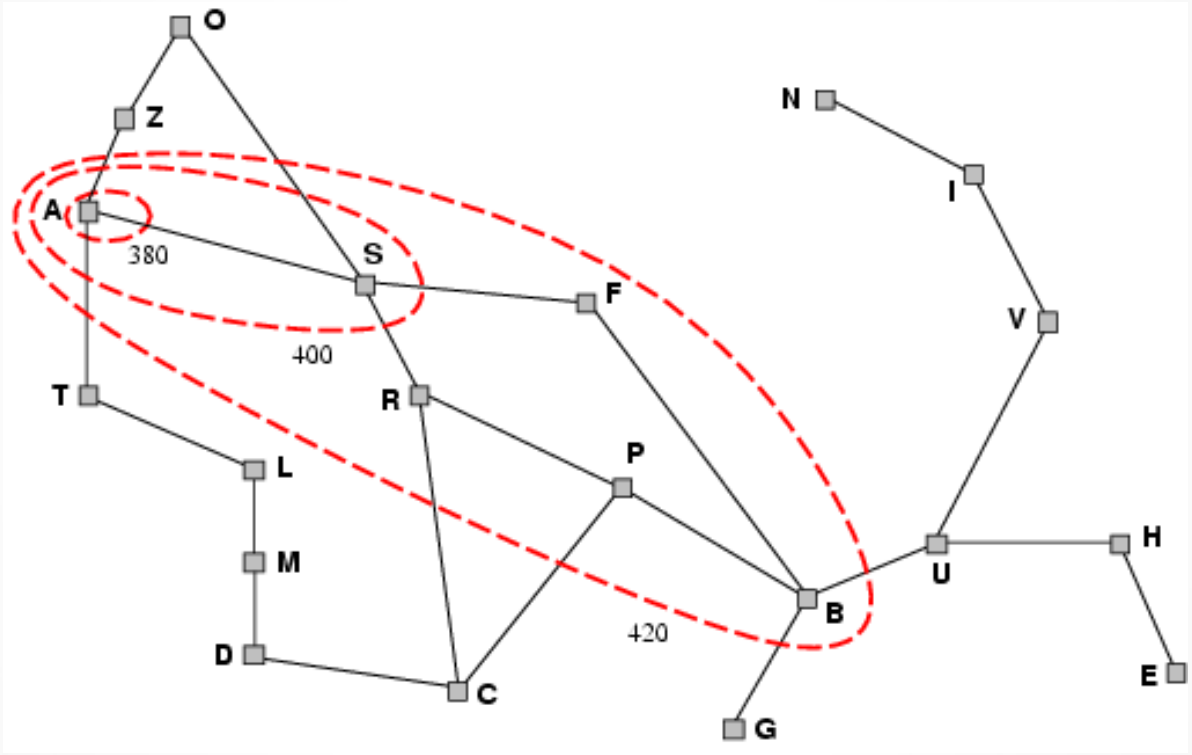
A cada passo, aumenta o valor máximo de f que pode expandir

Busca A*

- Contornos no espaço de estados
  - Contorno *i* tem todos os nós com $f=f_i$, em que $f_i < f_{i+1}$
  - Por outro lado, busca em largura adiciona camadas de acordo com nível do nó apenas



Busca A*



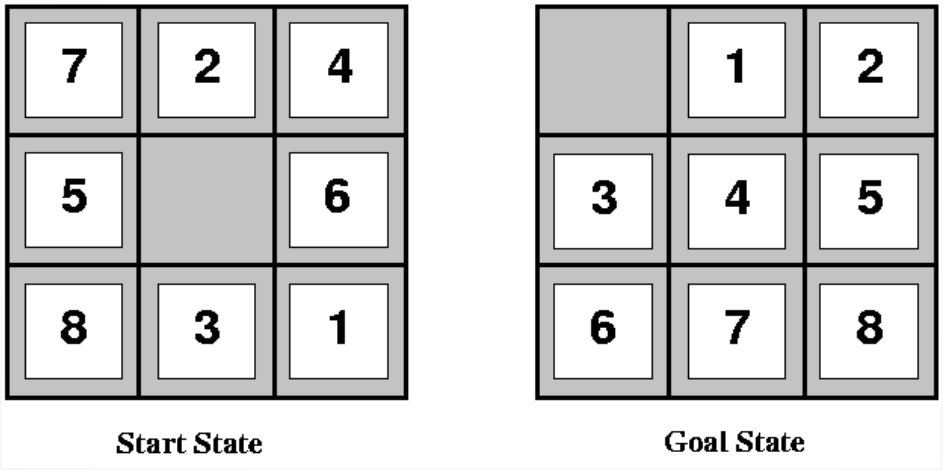
Com heurísticas mais precisas, as faixas se alongam em direção ao estado objetivo e se concentram mais em torno do caminho ótimo

Exemplos de heurísticas

- Jogo dos blocos deslizantes
  - h1 = número de blocos em posições erradas
    - Admissível, pois cada bloco deve ser movido ao menos uma vez
  - h2 = distância dos blocos de suas posições objetivo
    - Admissível, ao menos terá que deslocar isso

h1 = 8

h2 = 18



Exemplos de heurísticas

- Jogo dos blocos deslizantes: qual usar?

	Custo da busca		
d	BAI	$A^*(h_1)$	$A^*(h_2)$
2	10	6	6
4	112	13	12
6	680	20	18
8	6384	39	25
10	47127	93	39
12	3644035	227	73
14	–	539	113
16	–	1301	211
18	–	3056	363
20	–	7276	676
22	–	18094	1219
24	–	39135	1641

Heurística dominante

A* usando h2 é melhor que A* usando h1 e muito melhor que a busca por aprofundamento iterativo

h2 é sempre melhor que h1, pois

$$\forall n \; h2(n) \geq h1(n)$$

(chega mais próximo do valor real)

Isto é, h2 **domina** h1

Dominância = eficiência
A* com h2 nunca expandirá mais nós que A* com h1

Heurística dominante

Números de nós expandidos:

d = 14:

- BAI = 3473941 nós
- A*(h1) = 539 nós
- A*(h2) = 113 nós

d = 24:

- BAI = muitos nós
- A*(h1) = 39135 nós
- A*(h2) = 1641 nós

Referências

Capítulo 3 Russel e Norvig

Trabalho Individual – 18/07/2021

Resolver Quebra-cabeça de 8 peças

- Implementar 3 algoritmos de busca (larguna, profundidade e gulosa);

- Rodar 100 vezes;

- Escrever um relatório sobre os experimentos observados quanto à completeza (sim, não e passos para soluções), otimalidade, tempo e memória.

- Mostrar que h2 domina h1 (slide 36);

- Propor uma nova heurística e mostre s ela domina as outras duas.