Informované prohledávání stavového prostoru

Neinformované prohledávání:

- DFS, BFS a varianty
- nemá (téměř) žádné informace o pozici cíle – slepé prohledávání
- zná pouze:
  - počáteční/cílový stav
  - přechodovou funkci

Informované prohledávání stavového prostoru

Neinformované prohledávání:

- DFS, BFS a varianty
- nemá (téměř) žádné informace o pozici cíle – slepé prohledávání
- zná pouze:
  - počáteční/cílový stav
  - přechodovou funkci

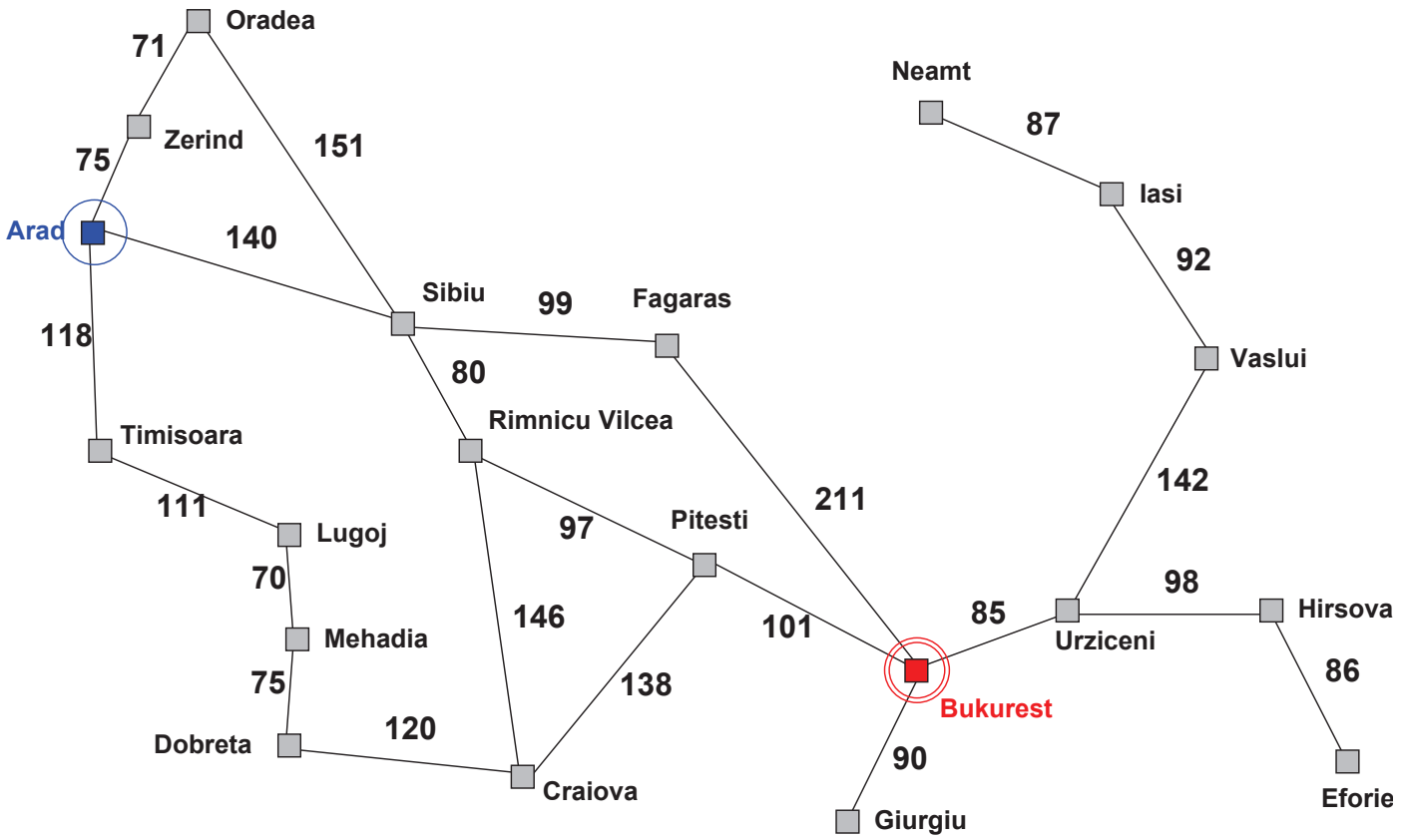
Informované prohledávání:

má navíc informaci o (odhadu) blízkosti stavu k cílovému stavu – heuristická funkce (heuristika)

Heuristické hledání nejlepší cesty

- Best-first Search
- použití ohodnocovací funkce $f(n)$ pro každý uzel – výpočet přínosu daného uzlu
- udržujeme seznam uzlů uspořádaný (vzestupně) vzhledem k $f(n)$
- použití heuristické funkce $h(n)$ pro každý uzel – odhad vzdálenosti daného uzlu od cíle
- čím *menší* $h(n)$, tím blíže k cíli, $h(\mathrm{Goal}) = 0$.
- nejjednodušší varianta – hladové heuristické hledání, *Greedy best-first search*
  $f(n) = h(n)$

Příklad – schéma rumunských měst



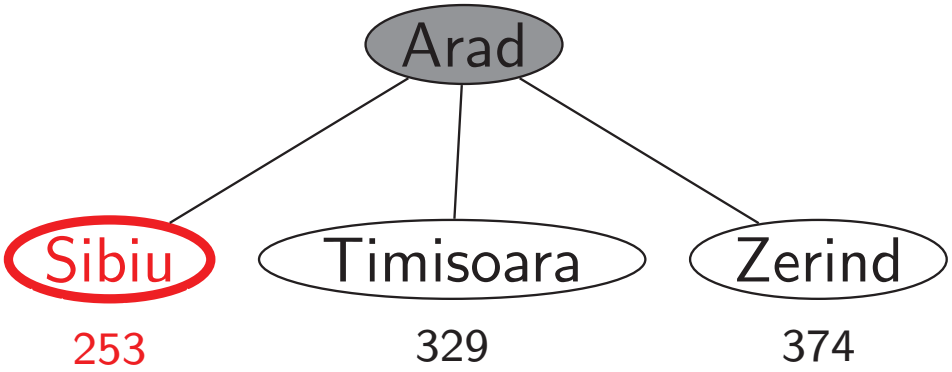
Arad	366
Bukurest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vilcea	199
Zerind	374

Hladové heuristické hledání – příklad

Hledání cesty z města *Arad* do města *Bukurest*
ohodnocovací funkce $f(n) = h(n) = h_{\text{vzd\_Buk}}(n)$, přímá vzdálenost z n do Bukuresti

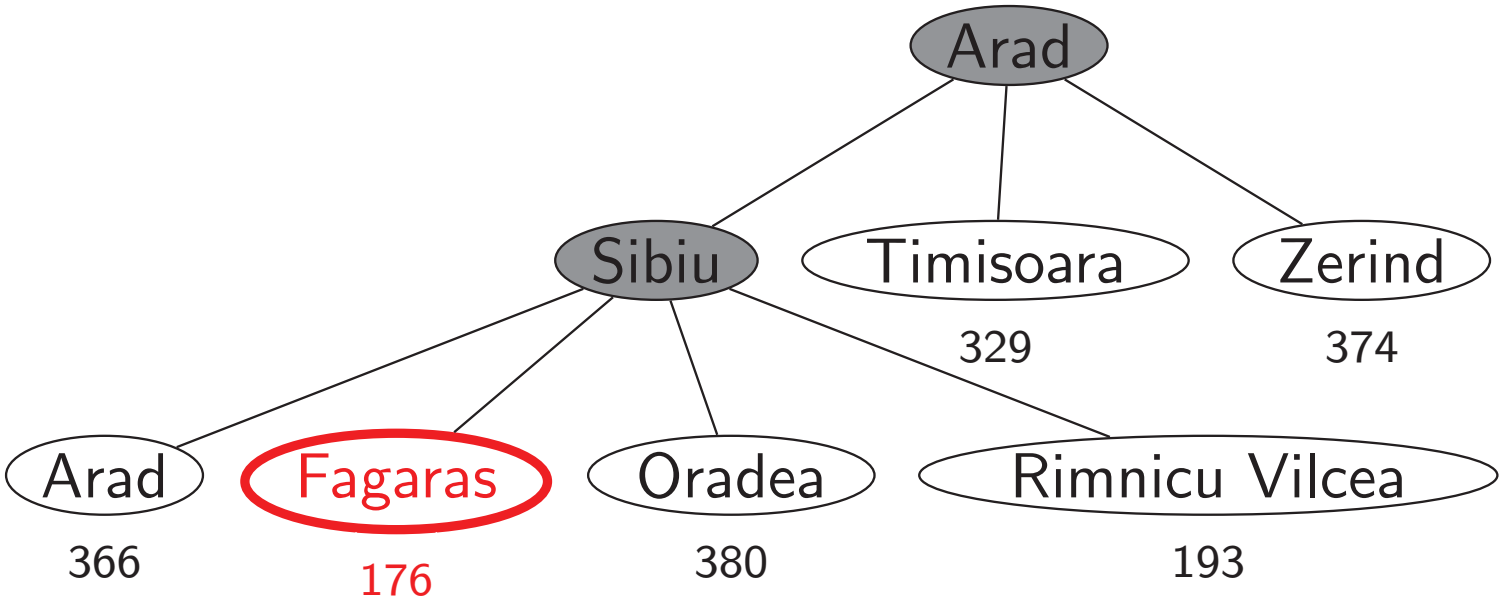
Hladové heuristické hledání – příklad

Hledání cesty z města *Arad* do města *Bukurest*
ohodnocovací funkce $f(n) = h(n) = h_{\text{vzd\_Buk}}(n)$, přímá vzdálenost z n do Bukuresti



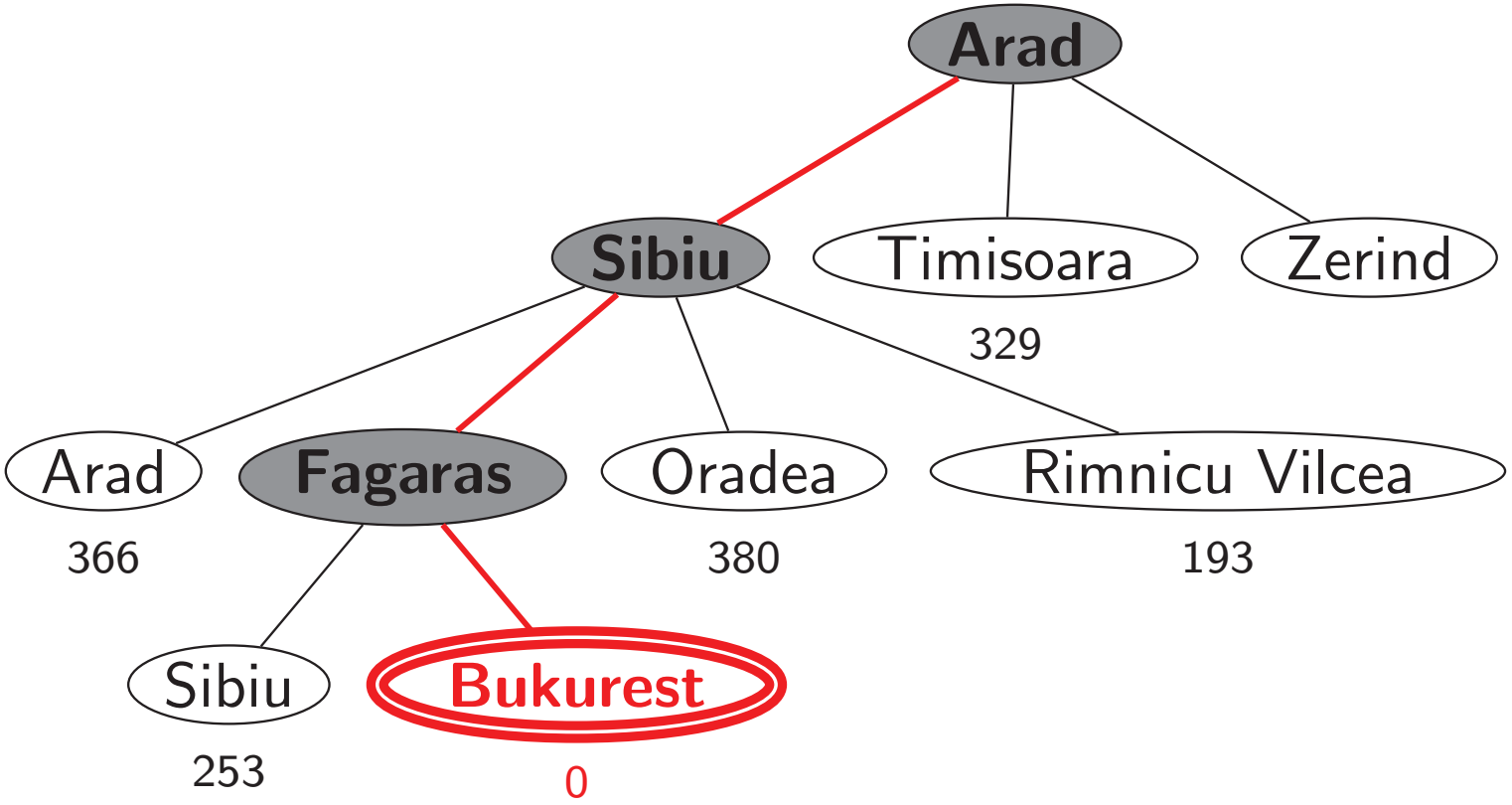
Hladové heuristické hledání – příklad

Hledání cesty z města *Arad* do města *Bukurest*
ohodnocovací funkce $f(n) = h(n) = h_{\text{vzd\_Buk}}(n)$, přímá vzdálenost z n do Bukuresti



Hladové heuristické hledání – příklad

Hledání cesty z města *Arad* do města *Bukurest*
ohodnocovací funkce $f(n) = h(n) = h_{\text{vzd\_Buk}}(n)$, přímá vzdálenost z n do Bukuresti



Hladové heuristické hledání – vlastnosti

- expanduje vždy uzel, který se zdá nejblíže k cíli
- cesta nalezená v příkladu ($g(\text{Arad} \to \text{Sibiu} \to \text{Fagaras} \to \text{Bukurest}) = 450$) je sice úspěšná, ale není optimální ($g(\text{Arad} \to \text{Sibiu} \to \text{RimnicuVilcea} \to \text{Pitesti} \to \text{Bukurest}) = 418$)
- *úplnost*
  optimálnost
  časová složitost
  prostorová složitost

Hladové heuristické hledání – vlastnosti

- expanduje vždy uzel, který se zdá nejblíže k cíli
- cesta nalezená v příkladu ($g(\text{Arad} \to \text{Sibiu} \to \text{Fagaras} \to \text{Bukurest}) = 450$) je sice úspěšná, ale není optimální ($g(\text{Arad} \to \text{Sibiu} \to \text{RimnicuVilcea} \to \text{Pitesti} \to \text{Bukurest}) = 418$)
- *úplnost* — obecně není úplný (nekonečný prostor, cykly)
  optimálnost
  časová složitost
  prostorová složitost

Hladové heuristické hledání – vlastnosti

- expanduje vždy uzel, který se zdá nejblíže k cíli
- cesta nalezená v příkladu ($g(\text{Arad} \to \text{Sibiu} \to \text{Fagaras} \to \text{Bukurest}) = 450$) je sice úspěšná, ale není optimální ($g(\text{Arad} \to \text{Sibiu} \to \text{RimnicuVilcea} \to \text{Pitesti} \to \text{Bukurest}) = 418$)
- *úplnost* — obecně není úplný (nekonečný prostor, cykly)
  optimálnost — není optimální
  časová složitost
  prostorová složitost

Hladové heuristické hledání – vlastnosti

- expanduje vždy uzel, který se zdá nejblíže k cíli
- cesta nalezená v příkladu ($g(\text{Arad} \to \text{Sibiu} \to \text{Fagaras} \to \text{Bukurest}) = 450$) je sice úspěšná, ale není optimální
  ($g(\text{Arad} \to \text{Sibiu} \to \text{RimnicuVilcea} \to \text{Pitesti} \to \text{Bukurest}) = 418$)
úplnost	obecně není úplný (nekonečný prostor, cykly)
optimálnost	není optimální
časová složitost	$O(b^m)$, hodně záleží na h
prostorová složitost	

Hladové heuristické hledání – vlastnosti

- expanduje vždy uzel, který se zdá nejblíže k cíli
- cesta nalezená v příkladu ($g(\text{Arad} \to \text{Sibiu} \to \text{Fagaras} \to \text{Bukurest}) = 450$) je sice úspěšná, ale není optimální
  ($g(\text{Arad} \to \text{Sibiu} \to \text{RimnicuVilcea} \to \text{Pitesti} \to \text{Bukurest}) = 418$)
úplnost	obecně není úplný (nekonečný prostor, cykly)
optimálnost	není optimální
časová složitost	$O(b^m)$, hodně záleží na h
prostorová složitost	$O(b^m)$, každý uzel v paměti

Hledání nejlepší cesty – algoritmus A*

- některé zdroje označují tuto variantu jako Best-first Search
- ohodnocovací funkce – kombinace $g(n)$ a $h(n)$:

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

$g(n)$	je cena cesty do n
$h(n)$	je odhad ceny cesty z n do cíle
$f(n)$	je odhad ceny nejlevnější cesty, která vede přes n

Hledání nejlepší cesty – algoritmus A*

- některé zdroje označují tuto variantu jako Best-first Search
- ohodnocovací funkce – kombinace $g(n)$ a $h(n)$:

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

$g(n)$	je cena cesty do n
$h(n)$	je odhad ceny cesty z n do cíle
$f(n)$	je odhad ceny nejlevnější cesty, která vede přes n

- A* algoritmus vyžaduje tzv. přípustnou (*admissible*) heuristiku:

$$0 \leq h(n) \leq h^*(n), \text{ kde } h^*(n) \text{ je skutečná cena cesty z } n \text{ do cíle}$$

tj. odhad se volí vždycky kratší nebo roven ceně libovolné možné cesty do cíle
Např. přímá vzdálenost $h_{\text{vzd\_Buk}}$ nikdy není delší než (jakákoliv) cesta

Heuristické hledání A* – příklad

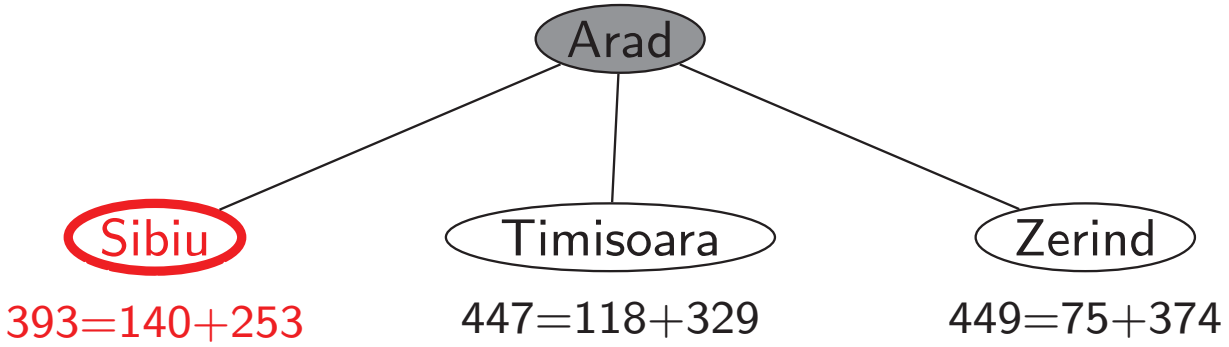
Hledání cesty z města *Arad* do města *Bukurest*
ohodnocovací funkce $f(n) = g(n) + h(n) = g(n) + h_{\mathrm{vzd\_Buk}}(n)$

Arad

366=0+366

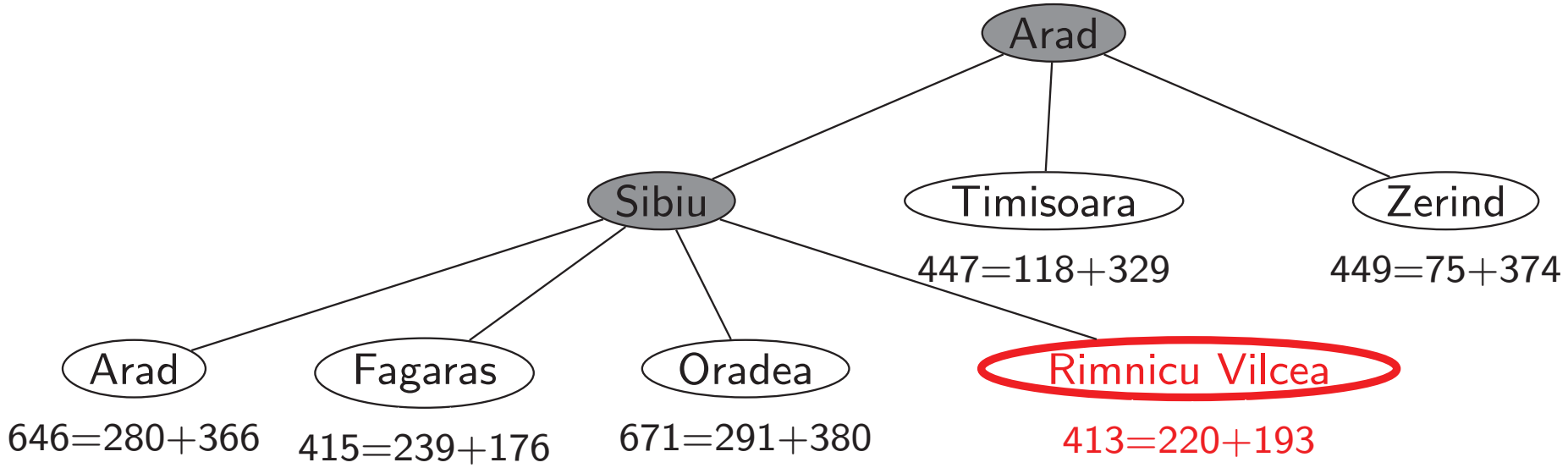
Heuristické hledání A* – příklad

Hledání cesty z města *Arad* do města *Bukurest*
ohodnocovací funkce $f(n) = g(n) + h(n) = g(n) + h_{\text{vzd\_Buk}}(n)$



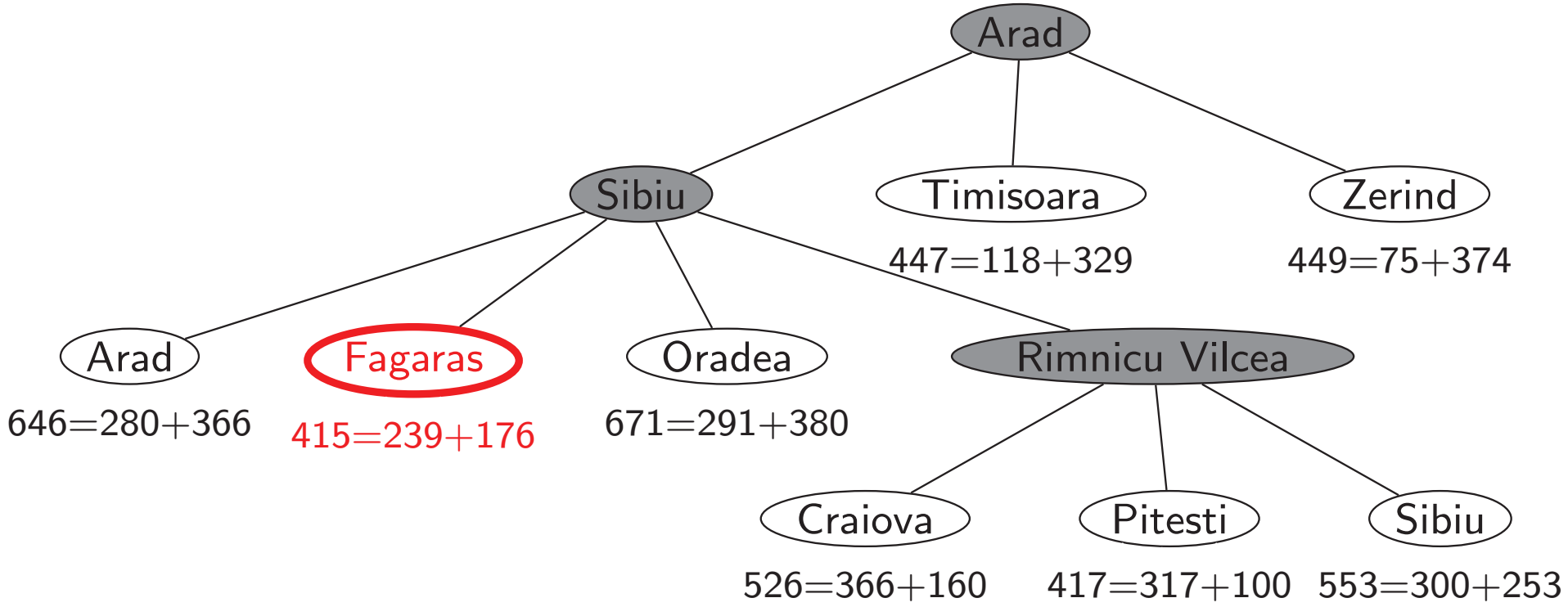
Heuristické hledání A* – příklad

Hledání cesty z města *Arad* do města *Bukurest*
ohodnocovací funkce $f(n) = g(n) + h(n) = g(n) + h_{\text{vzd\_Buk}}(n)$



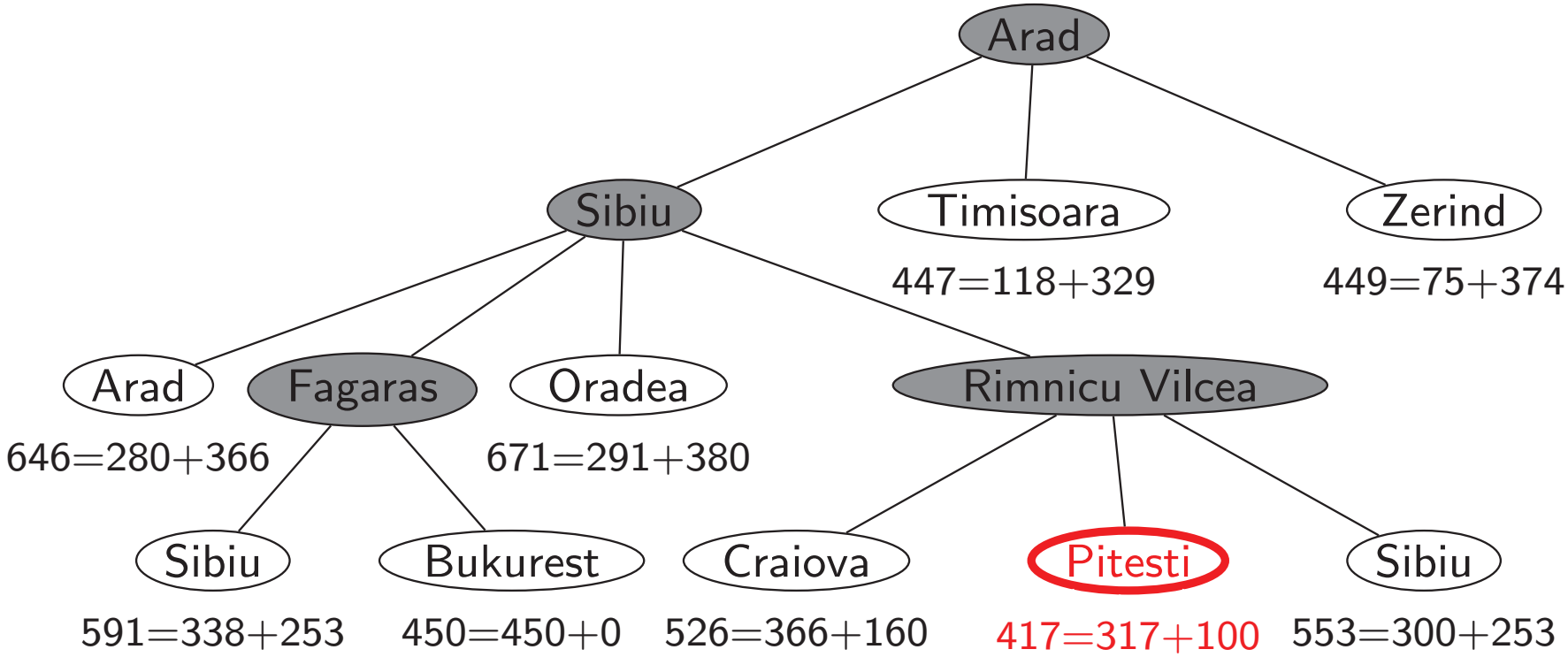
Heuristické hledání A* – příklad

Hledání cesty z města *Arad* do města *Bukurest*
ohodnocovací funkce $f(n) = g(n) + h(n) = g(n) + h_{\text{vzd\_Buk}}(n)$



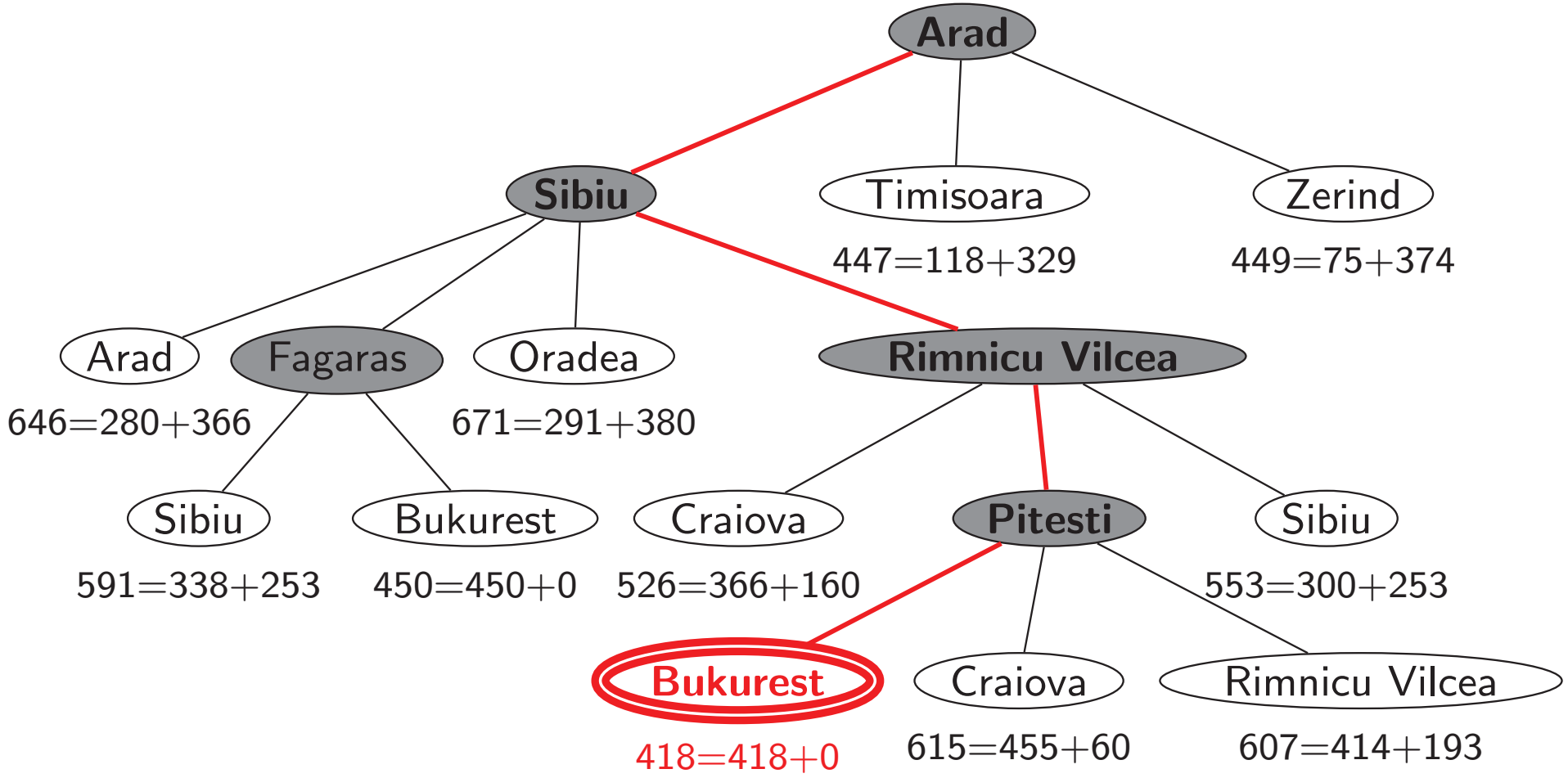
Heuristické hledání A* – příklad

Hledání cesty z města *Arad* do města *Bukurest*
ohodnocovací funkce $f(n) = g(n) + h(n) = g(n) + h_{\text{vzd\_Buk}}(n)$



Heuristické hledání A* – příklad

Hledání cesty z města *Arad* do města *Bukurest*
ohodnocovací funkce $f(n) = g(n) + h(n) = g(n) + h_{\text{vzd\_Buk}}(n)$



Hledání nejlepší cesty A* – vlastnosti

- expanduje uzly podle $f(n) = g(n) + h(n)$

  A* expanduje všechny uzly s $f(n) < C^*$
  A* expanduje některé uzly s $f(n) = C^*$
  A* neexpanduje žádné uzly s $f(n) > C^*$

- *úplnost*

  optimálnost

  časová složitost

  prostorová složitost

Hledání nejlepší cesty A* – vlastnosti

- expanduje uzly podle $f(n) = g(n) + h(n)$

  A* expanduje všechny uzly s $f(n) < C^*$
  A* expanduje některé uzly s $f(n) = C^*$
  A* neexpanduje žádné uzly s $f(n) > C^*$

- *úplnost* je úplný (pokud [počet uzlů s $f < C^*$] $\neq \infty$)

  optimálnost

  časová složitost

  prostorová složitost

Hledání nejlepší cesty A* – vlastnosti

- expanduje uzly podle $f(n) = g(n) + h(n)$

  A* expanduje všechny uzly s $f(n) < C^*$
  A* expanduje některé uzly s $f(n) = C^*$
  A* neexpanduje žádné uzly s $f(n) > C^*$

- *úplnost* je úplný (pokud [počet uzlů s $f < C^*$] $\neq \infty$)

  optimálnost je optimální

  časová složitost

  prostorová složitost

Hledání nejlepší cesty A* – vlastnosti

- expanduje uzly podle $f(n) = g(n) + h(n)$

  A* expanduje všechny uzly s $f(n) < C^*$
  A* expanduje některé uzly s $f(n) = C^*$
  A* neexpanduje žádné uzly s $f(n) > C^*$

- *úplnost* je úplný (pokud [počet uzlů s $f < C^*$] $\neq \infty$)

  optimálnost je optimální

  časová složitost $O\left((b^*)^d\right)$, exponenciální v délce řešení d

  b^* ... tzv. *efektivní faktor větvení*, viz dále

  prostorová složitost

Hledání nejlepší cesty A* – vlastnosti

- expanduje uzly podle $f(n) = g(n) + h(n)$

  A* expanduje všechny uzly s $f(n) < C^*$
  A* expanduje některé uzly s $f(n) = C^*$
  A* neexpanduje žádné uzly s $f(n) > C^*$

- *úplnost* je úplný (pokud [počet uzlů s $f < C^*$] $\neq \infty$)

  optimálnost je optimální

  časová složitost $O\left((b^*)^d\right)$, exponenciální v délce řešení d

  b^* ... tzv. *efektivní faktor větvení*, viz dále

  prostorová složitost $O\left((b^*)^d\right)$, každý uzel v paměti

Hledání nejlepší cesty A* – vlastnosti

- expanduje uzly podle $f(n) = g(n) + h(n)$

  A* expanduje všechny uzly s $f(n) < C^*$
  A* expanduje některé uzly s $f(n) = C^*$
  A* neexpanduje žádné uzly s $f(n) > C^*$

- *úplnost* je úplný (pokud [počet uzlů s $f < C^*$] $\neq \infty$)

  optimálnost je optimální

  časová složitost $O\left((b^*)^d\right)$, exponenciální v délce řešení d

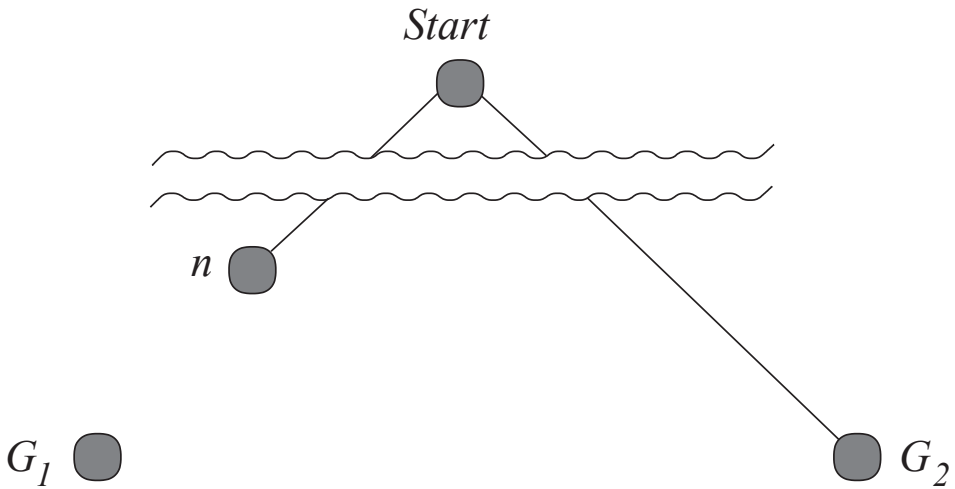
  b^* ... tzv. *efektivní faktor větvení*, viz dále

  prostorová složitost $O\left((b^*)^d\right)$, každý uzel v paměti

Problém s prostorovou složitostí řeší algoritmy jako *IDA**, *RBFS*

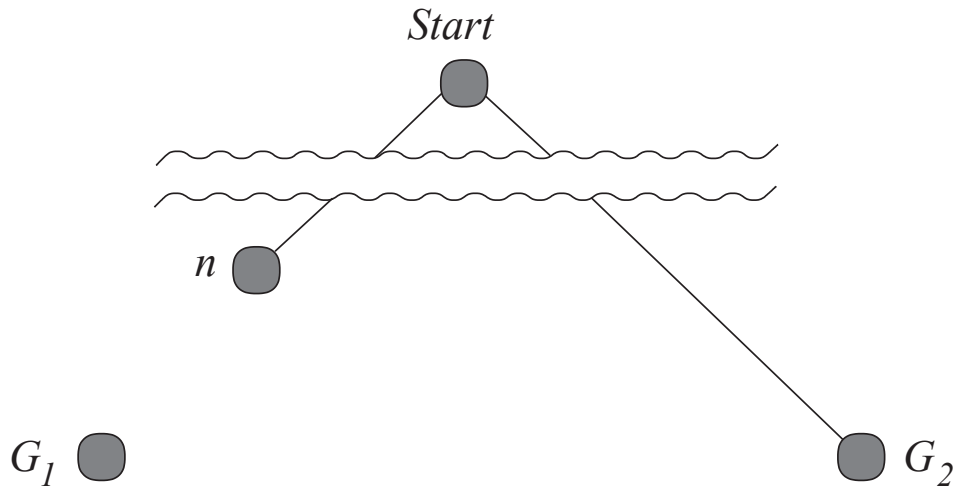
Důkaz optimálnosti algoritmu A*

- předpokládejme, že byl vygenerován nějaký suboptimální cíl G_2 a je uložen ve frontě.
- dále nechť n je neexpandovaný uzel na nejkratší cestě k optimálnímu cíli G_1 (tj. chybně neexpandovaný uzel ve správném řešení)



Důkaz optimálnosti algoritmu A*

- předpokládejme, že byl vygenerován nějaký suboptimální cíl G_2 a je uložen ve frontě.
- dále nechť n je neexpandovaný uzel na nejkratší cestě k optimálnímu cíli G_1 (tj. chybně neexpandovaný uzel ve správném řešení)

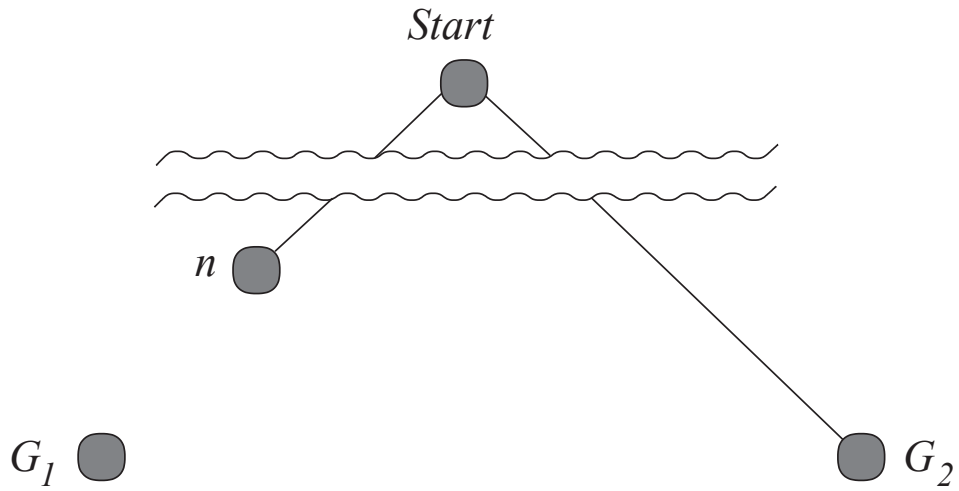


Pak

$$\begin{aligned} f(G_2) &= g(G_2) && \text{protože } h(G_2) = 0 \\ &> g(G_1) && \text{protože } G_2 \text{ je suboptimální} \\ &\geq f(n) && \text{protože } h \text{ je přípustná} \end{aligned}$$

Důkaz optimálnosti algoritmu A*

- předpokládejme, že byl vygenerován nějaký suboptimální cíl G_2 a je uložen ve frontě.
- dále nechť n je neexpandovaný uzel na nejkratší cestě k optimálnímu cíli G_1 (tj. chybně neexpandovaný uzel ve správném řešení)



Pak

$$\begin{aligned} f(G_2) &= g(G_2) && \text{protože } h(G_2) = 0 \\ &> g(G_1) && \text{protože } G_2 \text{ je suboptimální} \\ &\geq f(n) && \text{protože } h \text{ je přípustná} \end{aligned}$$

tedy $f(G_2) > f(n) \Rightarrow$ A* nikdy nevybere G_2 pro expanzi dřív než expanduje $n \rightarrow$ spor s předpokladem, že n je *neexpandovaný uzel* □