CS 3011: Artificial Intelligence

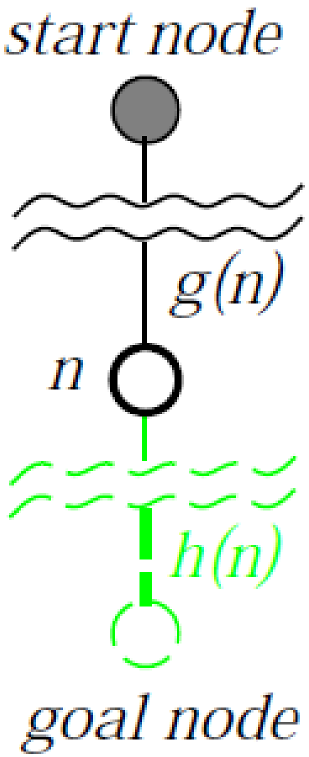
Solving Problems by Searching

Instructors: Dr. Durgesh Singh

CSE Discipline, PDPM IIITDM, Jabalpur -482005

A* Search

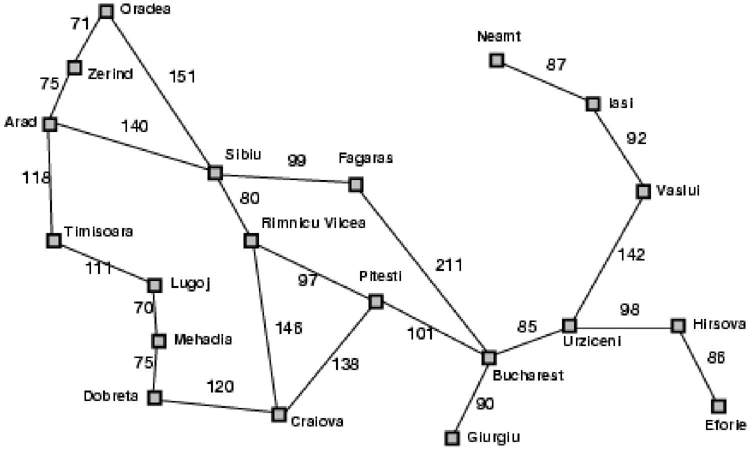
- Idea: Avoid expanding paths that are already expensive
- Evaluation function $f(n) = g(n) + h(n)$
  - $g(n)$ = exact cost so far to reach n
  - $h(n)$ = estimated cost to goal from n
  - $f(n)$ = estimated total cost of cheapest path through n to goal
- A* search uses an admissible heuristic:
  - $h(n) \leq h^*(n)$ where $h^*(n)$ is the true cost from n
  - Also $h(n) \geq 0$, and $h(G)=0$ for any goal G
  - E.g., $h_{SLD}(n)$ is an admissible heuristic because it doesn't overestimate the actual road distance.



A* Search

- If we are trying to find the cheapest solution, a reasonable thing to try first is the node with the lowest value of $g(n) + h(n)$
- This strategy is more than just reasonable
  - Provided that $h(n)$ satisfies certain conditions, A* using TREE search is both complete and optimal.

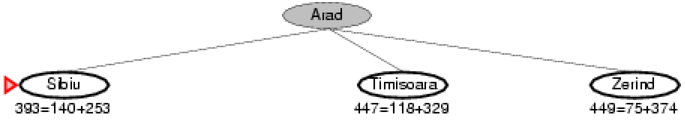
A* search example

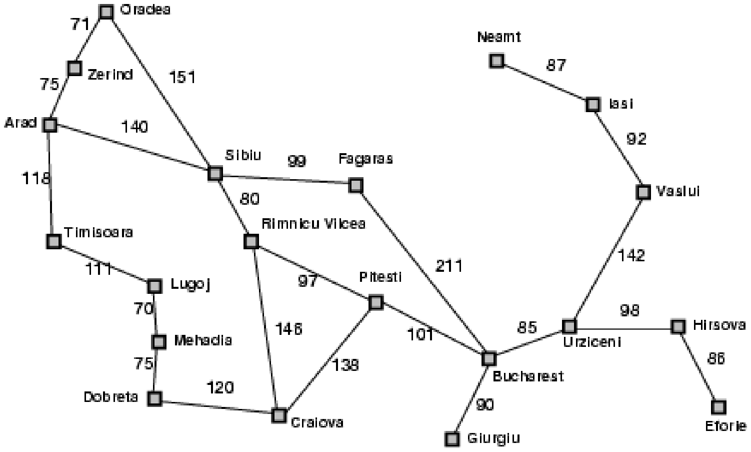


Straight–line distance to Bucharest

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	10
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

A* Tree search example

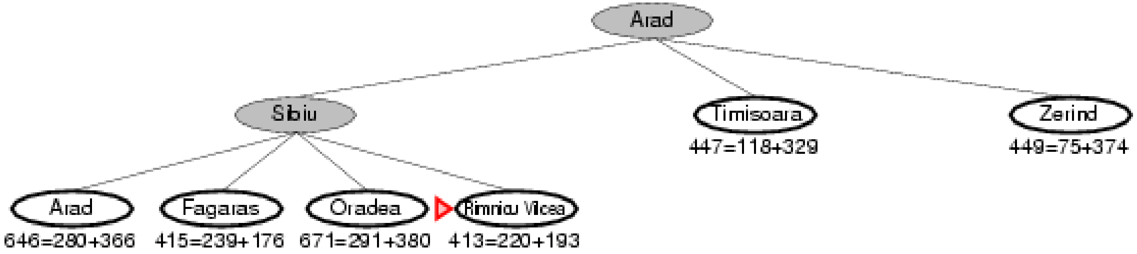


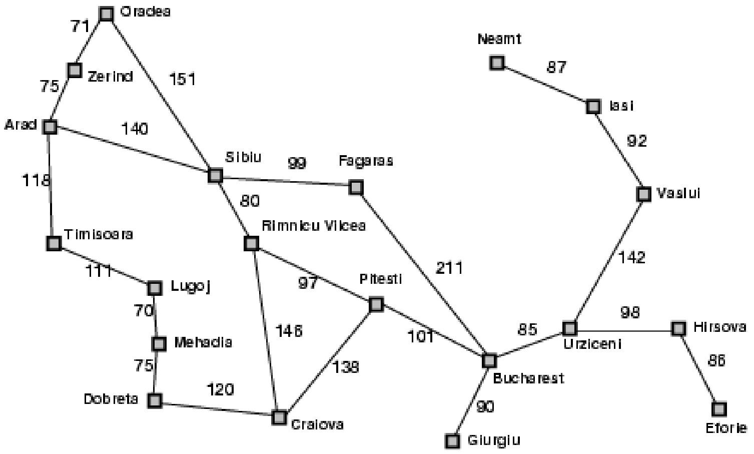


Straight–line distance to Bucharest

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	10
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

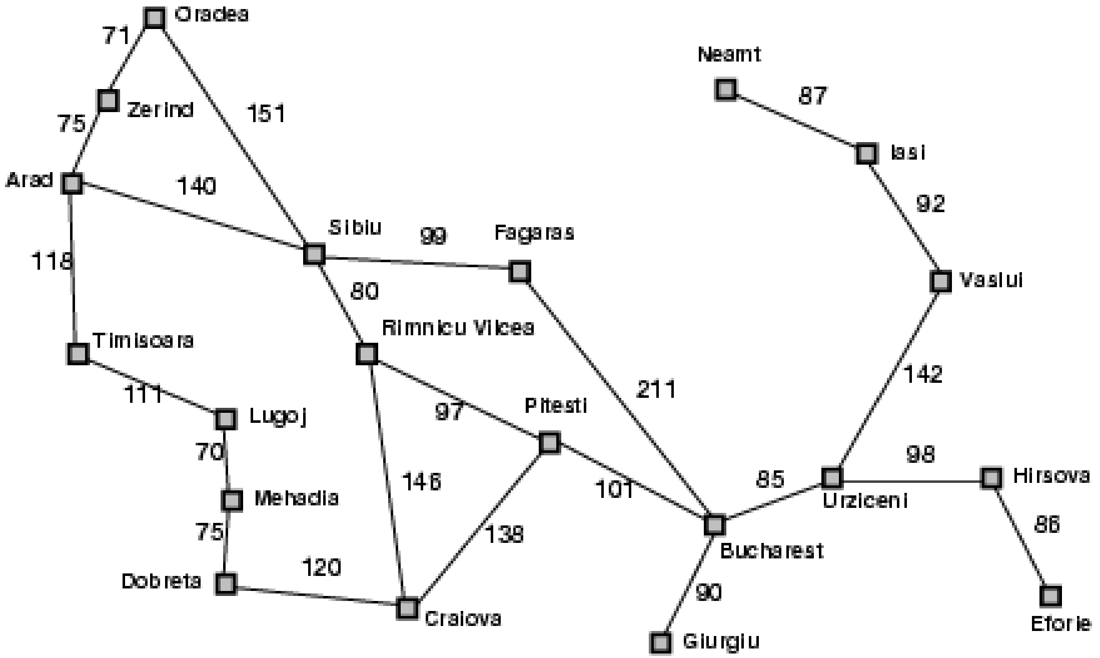
A* Tree search example





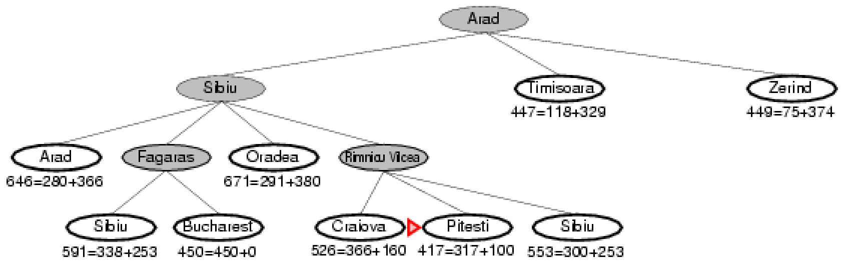
Straight–line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	10
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

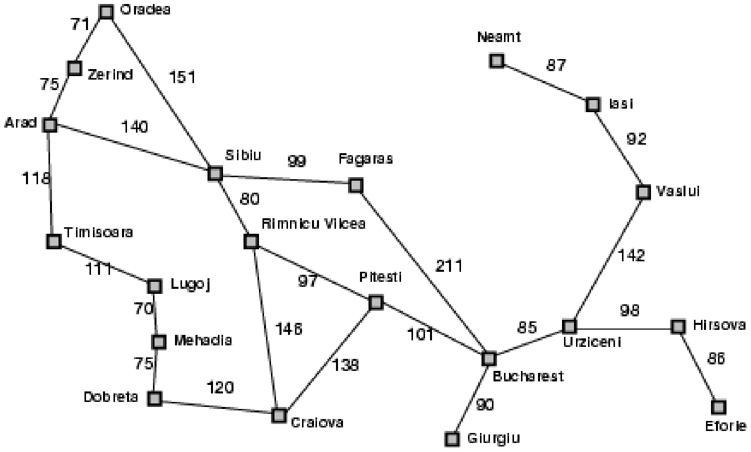
A* Tree search example



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	10
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

A* Tree search example

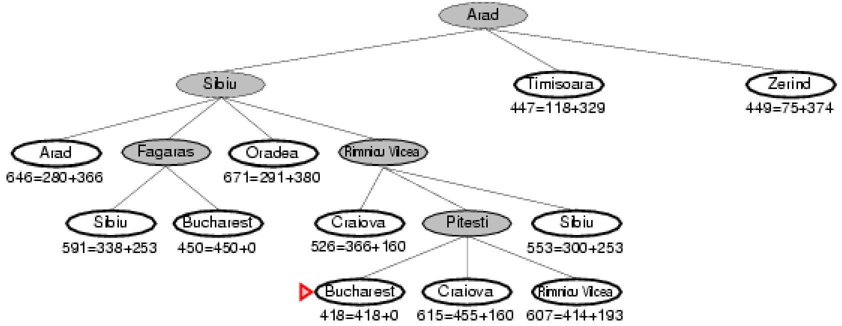


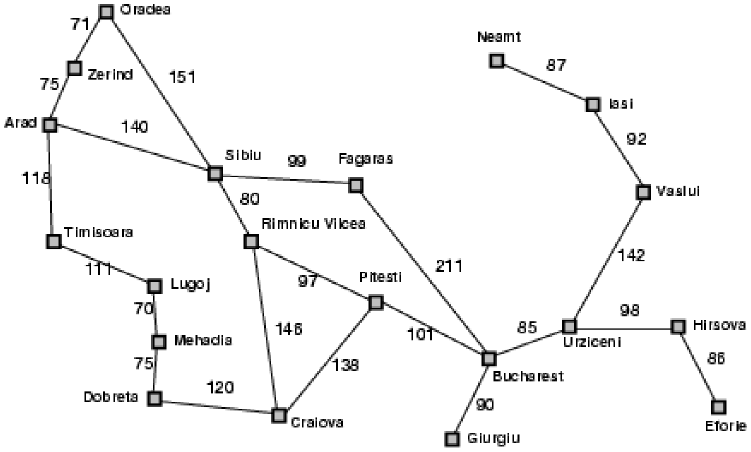


Straight–line distance to Bucharest

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	10
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

A* Tree search example

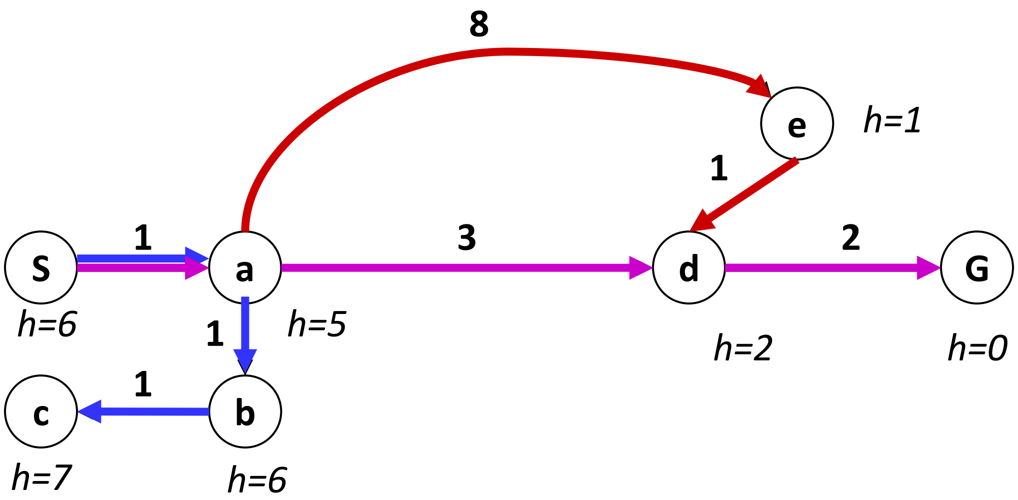


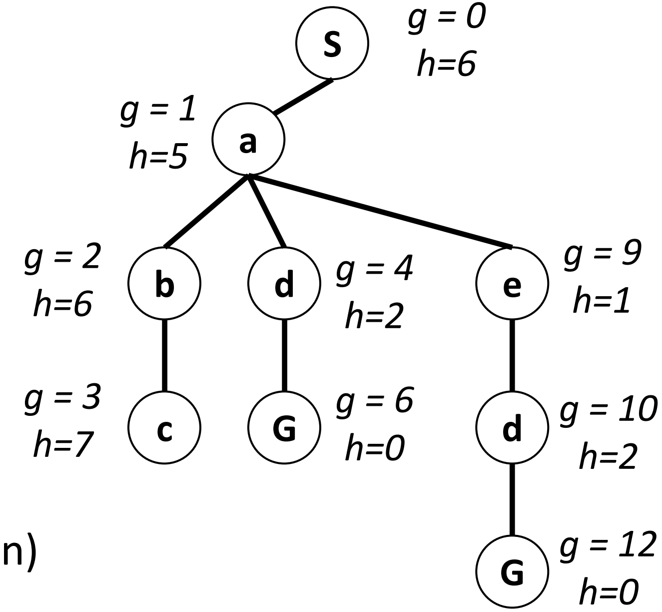


Straight–line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	10
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Combining UCS and Greedy

- Uniform-cost orders by path cost, g(n)
- Greedy orders by heuristic value, h(n)

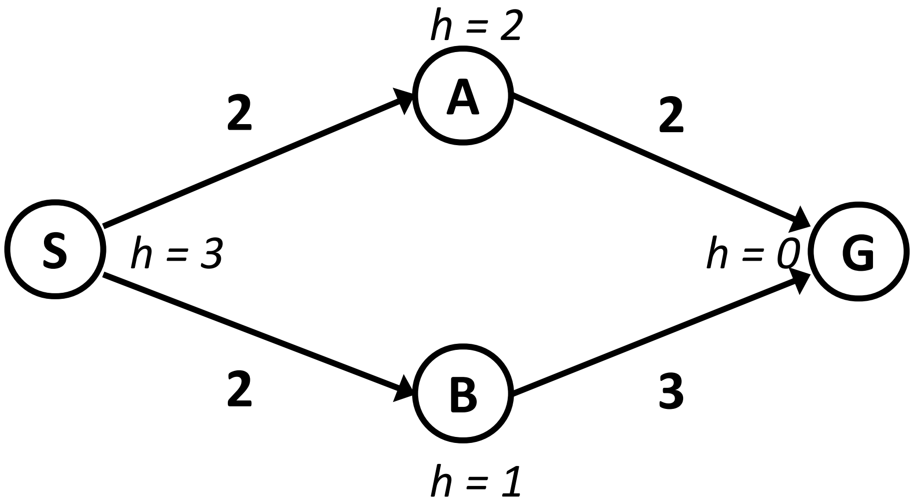




- A* Search orders by the sum: f(n) = g(n) + h(n)

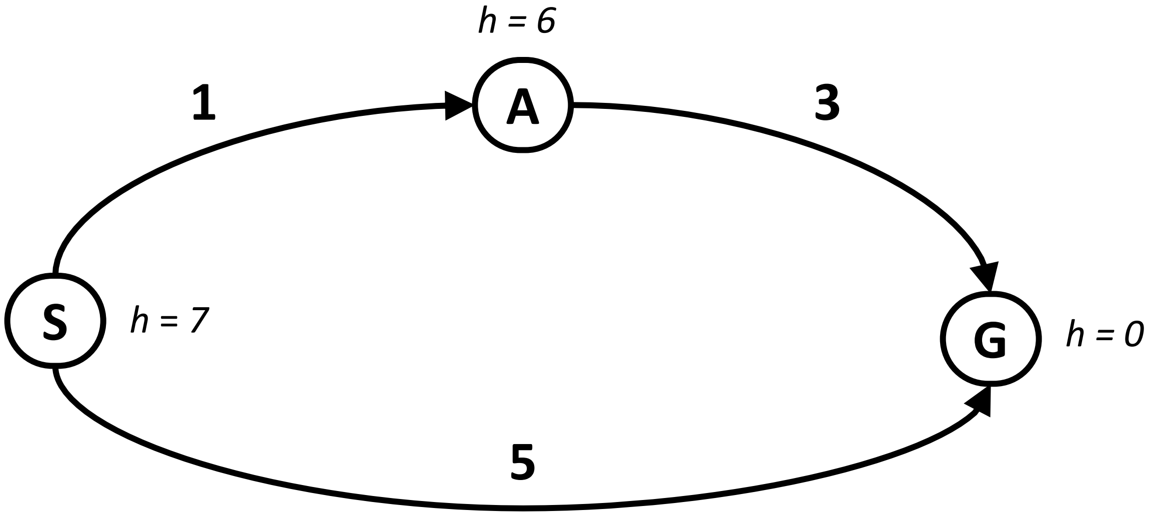
When should A* terminate?

- Should we stop when we enqueue a goal?



- No: only stop when we dequeue a goal

Is A* Optimal?



- What went wrong?
- We need estimates to be less than actual costs!

Admissible Heuristics

- A heuristic *h* is *admissible* (optimistic) if:

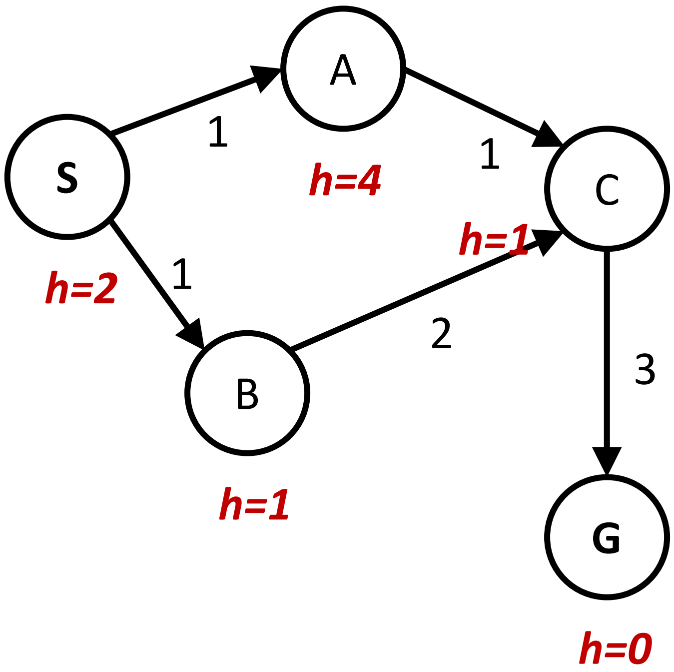
$$0 \leq h(n) \leq h^*(n)$$

  where $h^*(n)$ is the true cost to a nearest goal

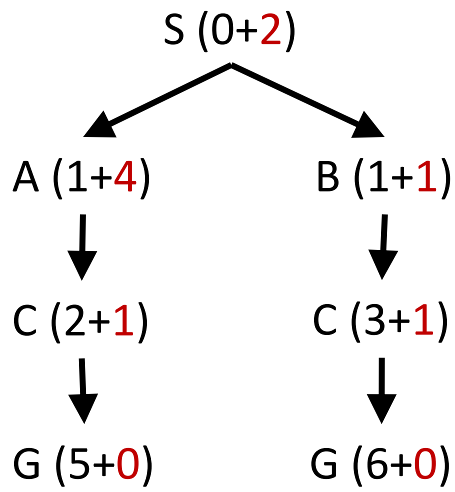
- Theorem: If *h(n)* is admissible, A* using `TREE-SEARCH` is optimal

A* Graph Search Gone Wrong?

State space graph

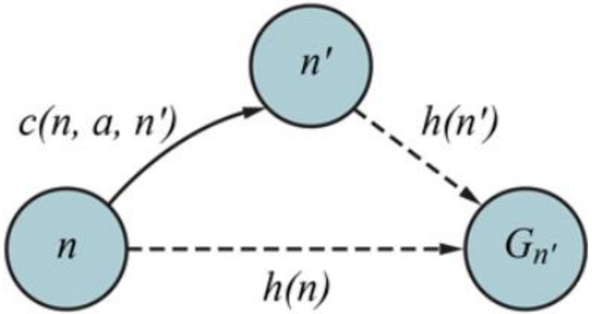


Search tree



Consistent heuristics

- A heuristic is consistent if for every node n, every successor n' of n generated by any action a,
- $h(n) \le c(n,a,n') + h(n')$
- `If h is consistent, we have
- $f(n') = g(n') + h(n')$
- $= g(n) + c(n,a,n') + h(n')$
- $\ge g(n) + h(n)$
- $= f(n)$
- i.e., f(n) is non-decreasing along any path.
- Theorem: If h(n) is consistent, A* using GRAPH-SEARCH is optimal



Dominance

Definition: If $h_2(n) \geq h_1(n)$ for all n (both admissible) then h_2 dominates h_1

- **Essentially, domination translates directly into efficiency:**
- h2 is better for search.
- A* using h2 will never expand more nodes than A* using h1 .

Properties of A*

- Complete? Yes
- Time? Exponential: O(b^m)
- Space? O(b^m):Keeps all nodes in memory
- Optimal? Yes