Tufts

Lecture 04: Heuristic Search (A*)

Artificial Intelligence (CS 131)

1

Review: Best-First and Uniform-Cost Search

Image source: Russell & Norvig (2021)

```
function BEST-FIRST-SEARCH(problem, f) returns a solution node or failure
  node ← NODE(STATE=problem.INITIAL)
  frontier ← a priority queue ordered by f, with node as an element
  reached ← a lookup table, with one entry with key problem.INITIAL and value node
  while not IS-EMPTY(frontier) do
    node ← POP(frontier)
    if problem.IS-GOAL(node.STATE) then return node
    for each child in EXPAND(problem, node) do
      s ← child.STATE
      if s is not in reached or child.PATH-COST < reached[s].PATH-COST then
        reached[s] ← child
        add child to frontier
  return failure
```

```
function UNIFORM-COST-SEARCH(problem) returns a solution node, or failure
  return BEST-FIRST-SEARCH(problem, PATH-COST)
```


2

Improving on Uninformed Search

- Uniform-Cost Search always checks the node in the frontier with the least path-cost so far
- This is node evaluation: ranks each node, chooses *best*
  - We can create other forms of this sort of best-first search, by changing *how* we do the evaluation
- We can do better (sometimes) if we have a heuristic
  - An *estimate* $h(n)$ of remaining cost from node n to the goal
  - Node evaluation can now be based also upon *estimated cost after* n, instead of only *known cost before*, $g(n)$
  - These heuristic values affect how we order our priority queue, and therefore the order of expanding frontier

3

Using Heuristics

- Heuristic: problem-specific knowledge, used to reduce *expected* search effort (*actual* performance may vary)
  - Evaluate relative desirability of expanding a node
  - Heuristics estimate the "distance" to a goal
- Examples depend on the domain:
  - Travel planning:
    - Euclidean (straight-line) distance
  - 8-puzzle
    - Manhattan distance
    - Number of misplaced tiles
  - Game-play:
    - Maximum possible amount won
    - Worst possible opponent play

4

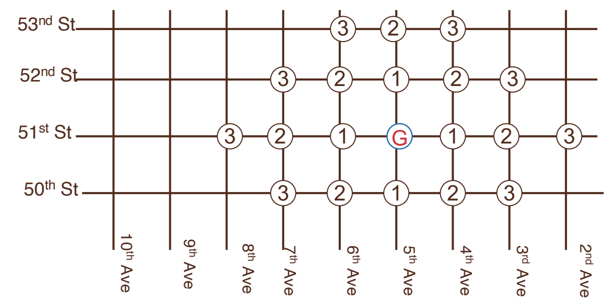
Greedy Search

- A simple form of heuristic search
  - Order fringe nodes in terms of heuristic value *h(n)*
  - Always expand node with least heuristic estimate
- For example: suppose we are searching for a path from point A to point B in a graph and:
  - We know size of graph, $|V| = N$
  - We know nothing else about it
  - Each edge-move has uniform cost = 1 unit per step
- What heuristics might we use? What is the result?

Performance of greedy search is dependent upon *how good* the heuristic is: in some cases, just the same as doing a simple (and exponential) breadth/depth-first search!

5

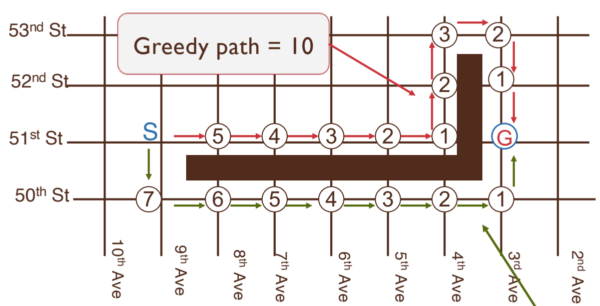
The Manhattan Distance Heuristic



A common heuristic for grid-based search problems: how many moves do we need to make horizontally or vertically to get to the goal G?

6

Greedy Search Using the MD



Optimal path = 8

Greedy search is complete, but it is *not* always optimal

7

A*: Optimal Heuristic Search

- Similar to Best-First search except that the evaluation is based on *total estimated path* (solution) cost, and we order our priority queue fringe by combined measure:

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

where we have:

$g(n)$ = cost of actual path so far from start state to n

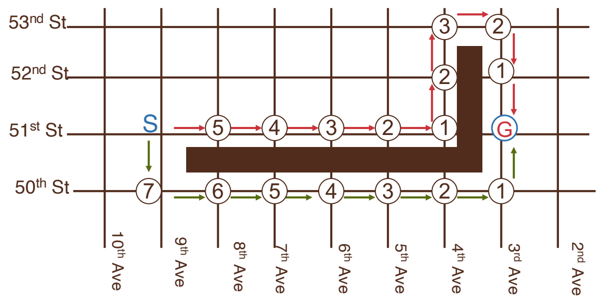
$h(n)$ = heuristic estimate of remaining cost from n to goal

(By convention, for any goal-state G, $h(G) = 0$)

- Combines the greedy approach with the knowledge about actual path-costs that we have gained during search process

8

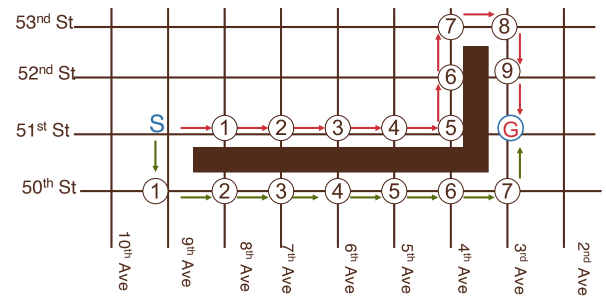
Heuristic (h) Values



We can use the *same* heuristic as the Greedy search method

9

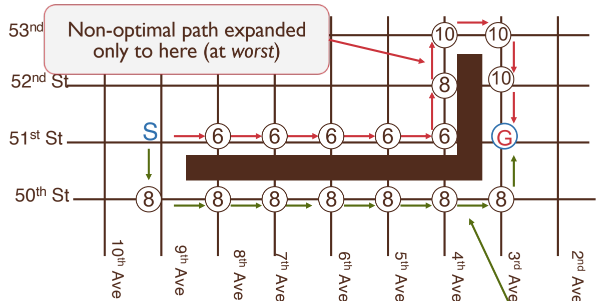
Actual Cost (g) Values



If each action has same cost (1), $g(n)$ is just path-length

10

Final A* values: $f = g + h$



Non-optimal path expanded only to here (at *worst*)

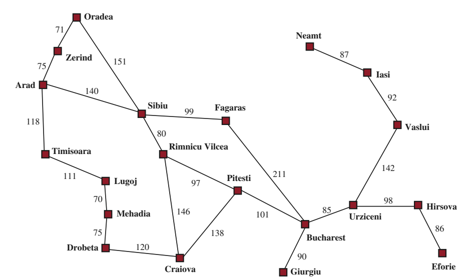
Optimal path will be expanded after that point, all the way to the goal

When heuristic is *correct* $f(n)$ is also correct

11

Getting to Bucharest Using A*

Image source: Russell & Norvig (2021)

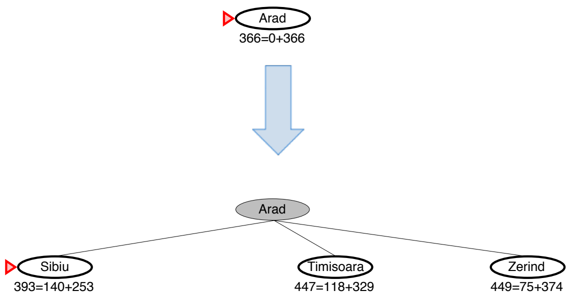


Heuristic (actual road distance may be greater)

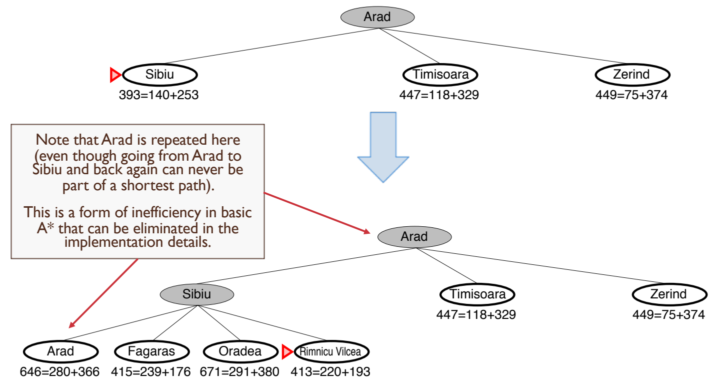
Arad	366	**Mehadia**	241
Bucharest	0	**Neamt**	234
Craiova	160	**Oradea**	380
Drobeta	242	**Pitesti**	100
Eforie	161	**Rimnicu Vilcea**	193
Fagaras	176	**Sibiu**	253
Giurgiu	77	**Timisoara**	329
Hirsova	151	**Urziceni**	80
Iasi	226	**Vaslui**	199
Lugoj	244	**Zerind**	374

12

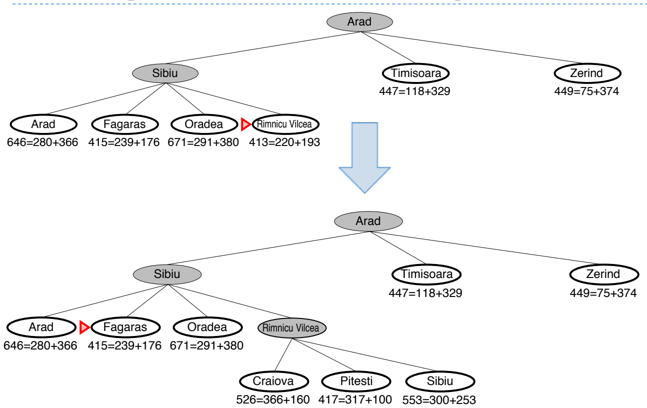
Getting to Bucharest Using A*



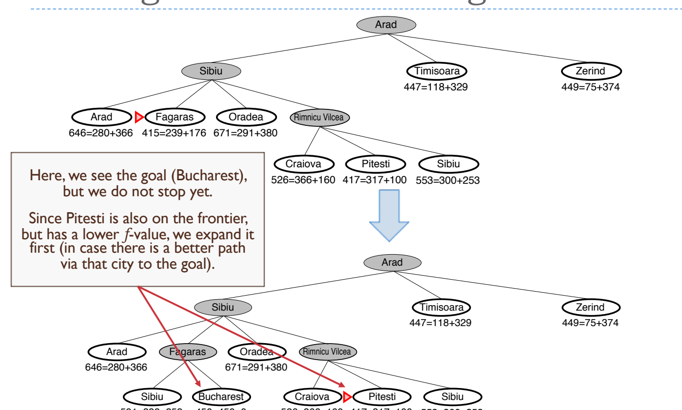
Getting to Bucharest Using A*



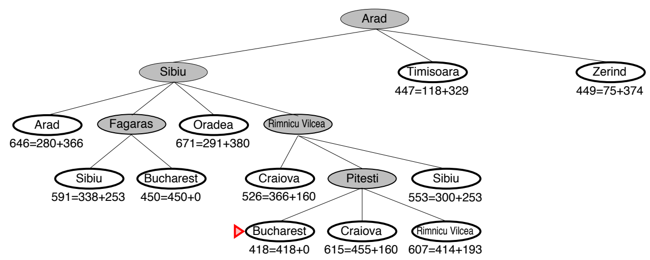
Getting to Bucharest Using A*



Getting to Bucharest Using A*



Getting to Bucharest Using A*



Finally, when the goal is found on the frontier and expanded, since it now has the lowest f-value, we are done, and have found the best possible path to that goal.

Admissibility and Consistency

- Admissible heuristic: *never overestimates* actual cost to reach the goal (so it never "discourages" us from expanding a node during search)
- For any node n, let $C^*(n)$ be the optimal cost from n to goal
$$\forall n,\ h(n) \leq C^\star(n)$$
- For any nodes n and n', let $a^*(n, n')$ be the optimal action that leads from n to n', which is to say:
$$\forall a,\ c(n, a^\star(n, n'), n') \leq c(n, a, n')$$
- This implies that optimal costs respect optimal paths/actions, i.e., if we have some optimal path ($Start,\ \ldots,\ n_k,\ n_{k+1},\ \ldots,\ Goal$):
$$C^\star(n_k) = c(n_k, a^\star(n_k, n_{k+1}), n_{k+1}) + C^\star(n_{k+1})$$
- Which in turn means that admissible heuristics respect optimal costs:
$$\forall n, n',\ h(n) \leq c(n, a^\star(n, n'), n') + C^\star(n')$$

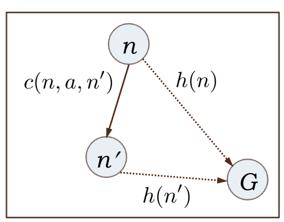
Admissibility and Consistency

$$\forall n,\ \forall a,\ \forall n',\ h(n) \leq c(n, a, n') + h(n')$$

Heuristic estimate for node n	*Actual cost* to get from n to n'	Heuristic estimate for *next node* n'

- Consistent heuristic: a form of "triangle inequality"
  - Consistent heuristics do not overestimate *total costs* **and** do not overestimate costs of *single actions*

Theorem: If heuristic h *is* consistent, then h is admissible



Consistency Implies Admissibility

- *Theorem*: If heuristic h *is* consistent, then h is admissible
  - Proven by induction on number of actions along path from some node to goal
- Base case: suppose node n is 1 step from the goal, G; by the definition of consistency:
$$h(n) \leq c(n, a^\star(n, G), G) + h(G)$$
- Since G is a goal-state, we have that $h(G) = 0$, and so:
$$h(n) \leq c(n, a^\star(n, G), G) = C^\star(n)$$

- Inductive hypothesis: Assume result holds for any nodes at distances $1,\ \ldots,\ k$ steps from the goal
- Inductive step: Show result holds nodes $(k + 1)$ steps from the goal

Consistency Implies Admissibility

- *Theorem*: If heuristic h is consistent, then h is admissible
- Inductive hypothesis: Assume result holds for any nodes at distances $1, \ldots, k$ steps from the goal
- Inductive step: Show result holds for nodes $(k + 1)$ steps from the goal
- Suppose we have nodes n, n', such that n is $(k + 1)$ steps from the goal on a path: $(Start, \ldots, n, n', \ldots, Goal)$
- By the definition of consistency:

$$h(n) \leq c(n, a^\star(n, n'), n') + h(n')$$

- But since n' is only k steps from the goal, the inductive hypothesis tells us that $h(n')$ doesn't over-estimate, i.e., $h(n') \leq C^\star(n')$
- Therefore, we have our result:

$$h(n) \leq c(n, a^\star(n, n'), n') + C^\star(n') = C^\star(n) \quad \blacksquare$$

21

Other Important Results about Consistent and/or Admissible Heuristics

- *Theorem*: If heuristic h is *consistent*, then path costs are monotonic (never decrease)
- *Theorem*: If heuristic h is *admissible*, then A* search is optimal, and always finds best path to a goal
- *Theorem*: If heuristic h is *consistent* and *admissible*, then A* is maximally efficient; that is, no algorithm will *always* expand fewer nodes than A*

22