Planning and Optimization

X2. Hands-On and Repetition

Gabriele Röger and Thomas Keller

Universität Basel

September 26, 2018

X2.1 Heuristics

X2.2 A* Search Algorithm

X2.3 Hands-On

Hands-On: Overview

Chapter overview: hands-on

- 1. The Planning Domain Definition Language (PDDL)
- 2. Getting to Know a Planner
- 3. Heuristics
- 4. A* search algorithm

X2.1 Heuristics

Heuristics

Definition (heuristic)

Let $\mathcal{S}$ be a state space with set of states S.
A heuristic function or heuristic for $\mathcal{S}$ is a function

$$h : S \to \mathbb{R}_0^+ \cup \{\infty\},$$

mapping each state to a non-negative number (or ∞).

Heuristics: Intuition

idea: $h(s)$ estimates cost of cheapest path from s to closest goal state

- heuristics can be arbitrary functions
- intuition: the closer h is to true cost to goal, the more efficient the search using h

X2.2 A* Search Algorithm

A* Search Algorithm

A* search algorithm

- based on heuristic h, define evaluation function f for node n: $f(n) := g(n) + h(n.\text{state})$
- trade-off between path cost and estimated proximity to goal
- intuition: $f(n)$ estimates costs of cheapest solution from initial state through n.state to goal

A* Search Algorithm: Pseudo-Code

A* search algorithm (with re-opening)

```
open := new priority queue, ordered by ⟨f, h⟩
if h(init-state) < ∞:
    open.insert(make_root_node())
distances := new HashTable
while not open.empty():
    n = open.pop-min()
    if (not distances.contains(n.state)) or (g(n) < distances[n.state]):
        distances[n.state] := g(n)
        if is-goal(n.state):
            return extract-solution(n)
        for each successor ⟨a, s'⟩ of n.state:
            if h(s') < ∞:
                n' := make_node(n, a, s')
                open.insert(n')
return unsolvable
```


A* Search Algorithm

Most important property

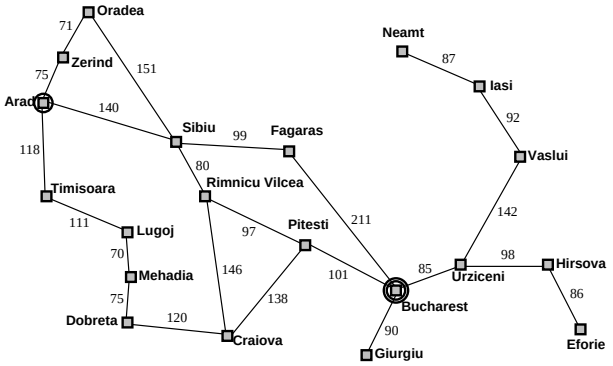
- A* is optimal if the applied heuristic is admissible.

For more details on best-first search and A^*, see chapter 15–19 of the AI course last semester.

(`https://dmi.unibas.ch/de/studium/computer-science-informatik/fs18/lecture-foundations-of-artificial-intelligence/`)

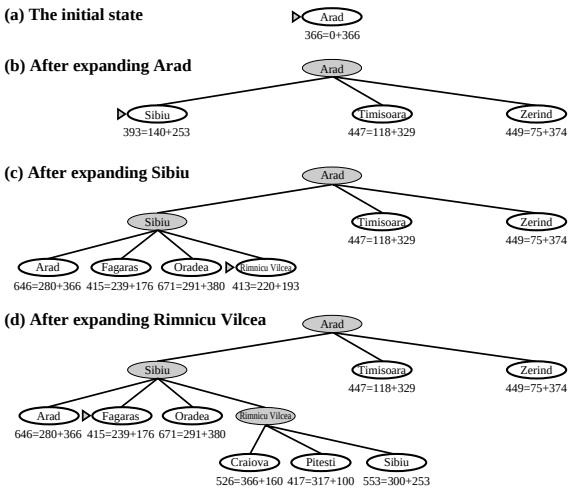
Example: A* for Route Planning

Example heuristic: straight-line distance to Bucharest

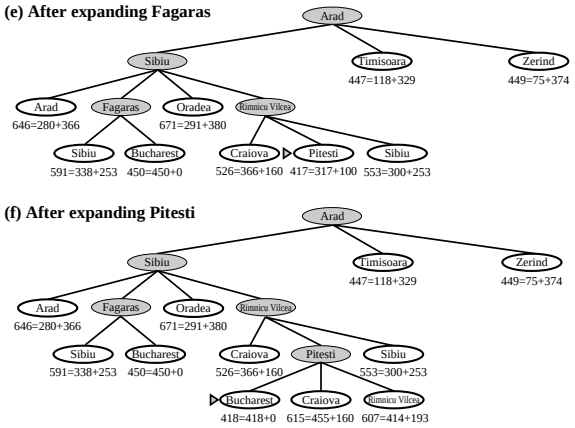


Arad	366
Bucharest	0
Craiova	160
Drobeta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	100
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Example: A* for Route Planning



Example: A* for Route Planning



X2.3 Hands-On

Hands-On

cloned course repository

```
https://bitbucket.org/aibasel/planopt-hs18
```

update the course repository

```
cd planopt-hs18
hg pull -u
```

compile the planner

```
cd classical/hands-on-2/fast-downward
./build.py
```

work on the hands-on exercises

- implement A* search
- compare to built-in implementation