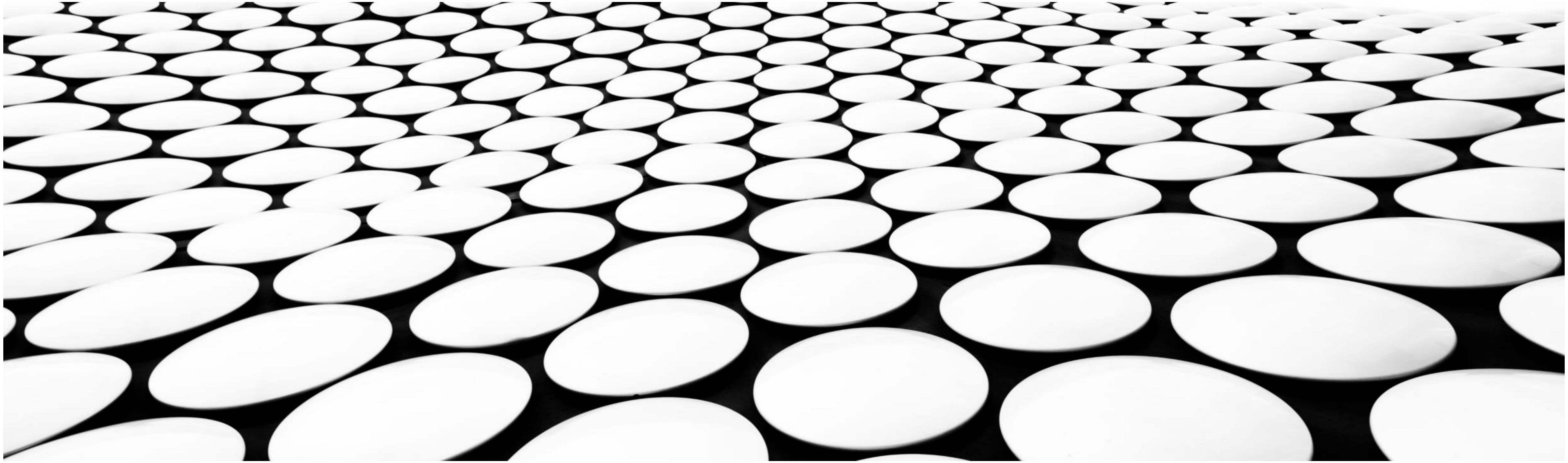
SOLUTION SPACES

BLIND AND HEURISTIC SEARCHES

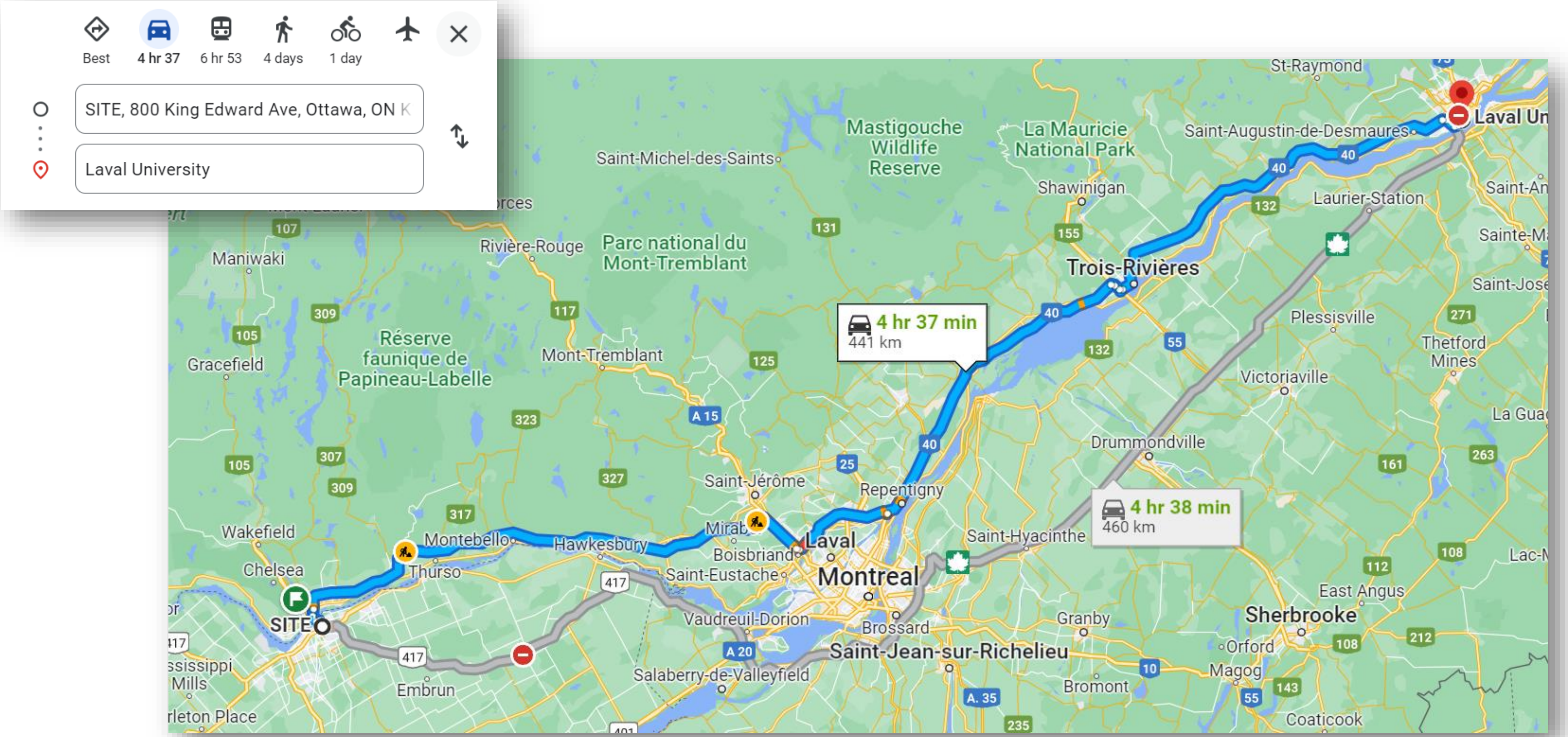


SOLUTION SPACES

- Part 1 – Examples of search problems
- Part 2 – Blind searches (review)
- Part 3 – Heuristic searches

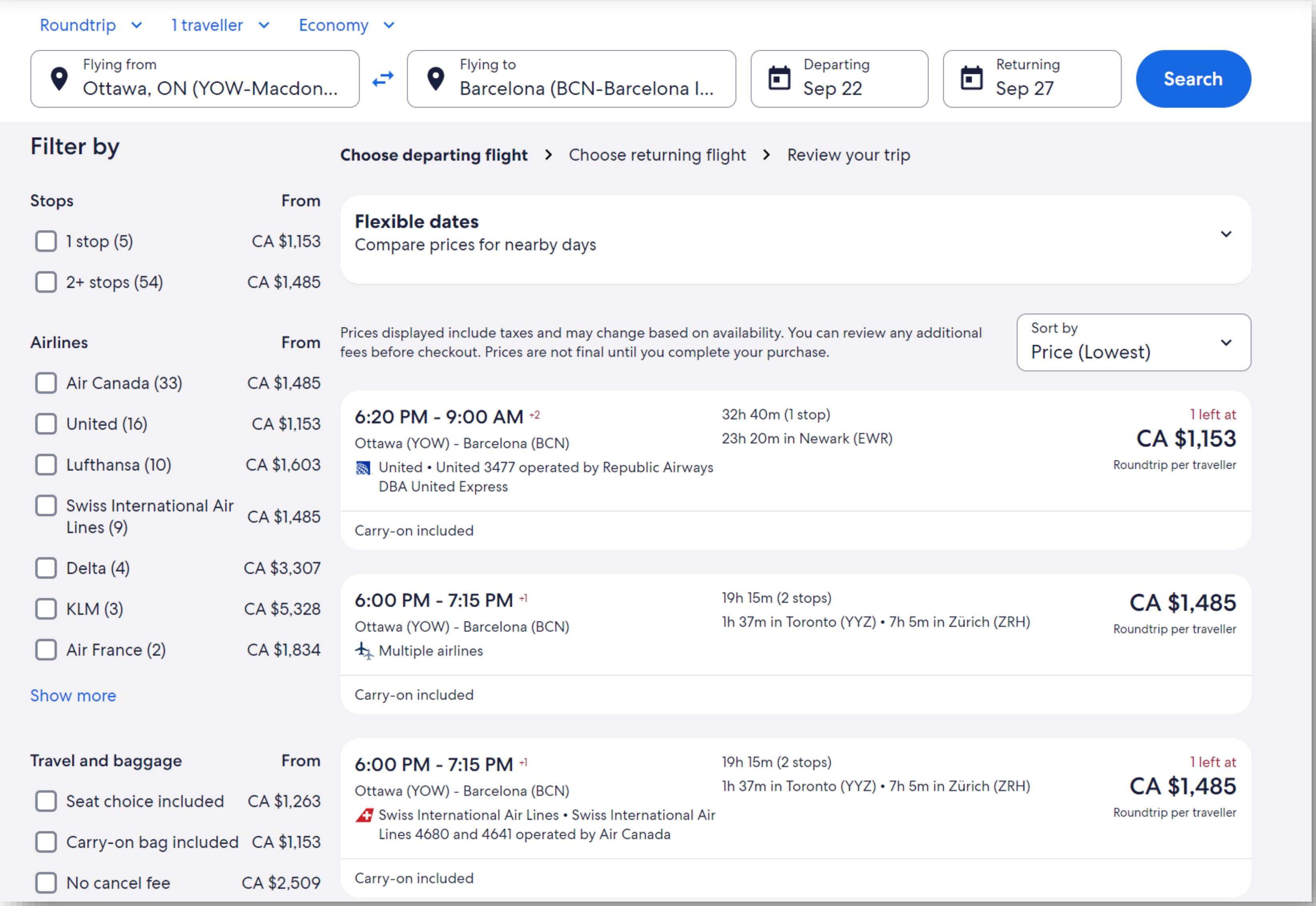
Part 1
Types of problems

SECTOR : LOGISTICS AND TRANSPORTATION

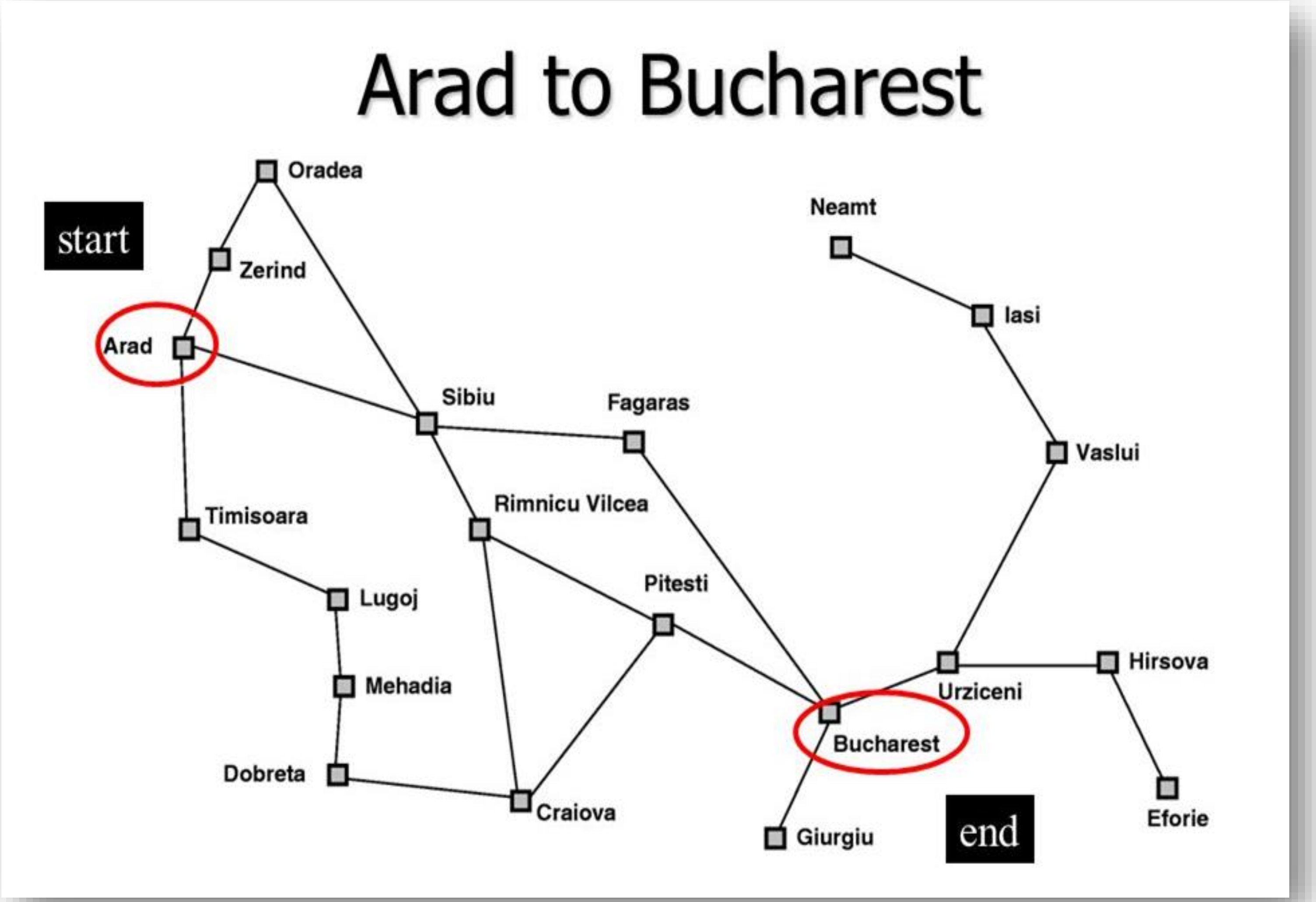


SECTOR : LOGISTICS AND TRANSPORTATION

Expedia



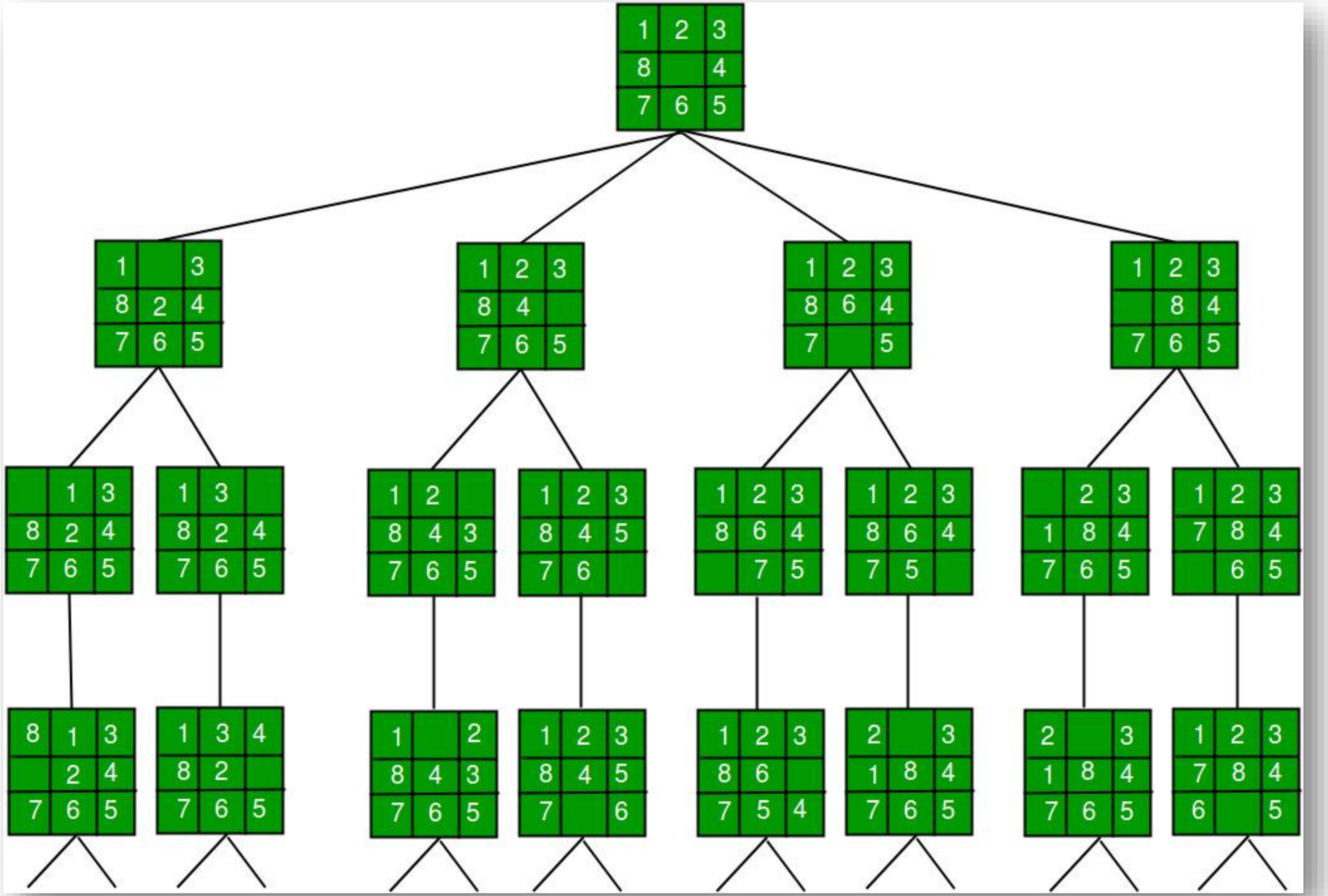
AI textbook typical path search problem



Solutions:

- A-Z-O-S-F-B
- A-Z-O-S-R-P-B
- A-Z-O-S-R-C-P-B
- A-T-L-M-D-C-P-B
- A-T-L-M-D-C-R-P-B
- …

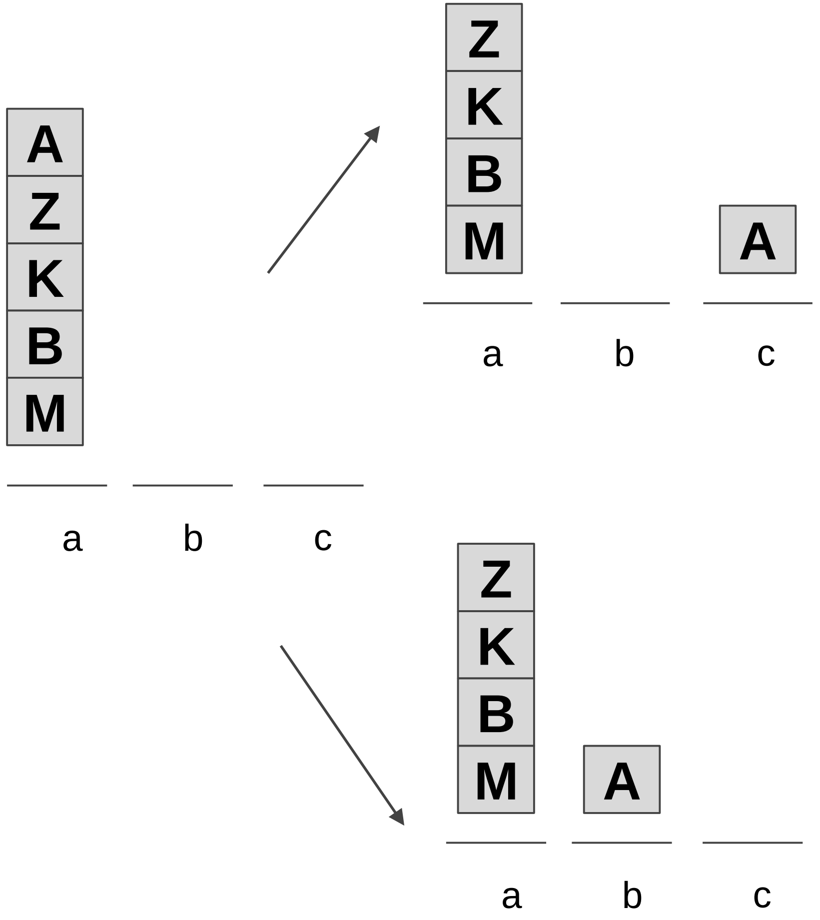
Game – 8 puzzle



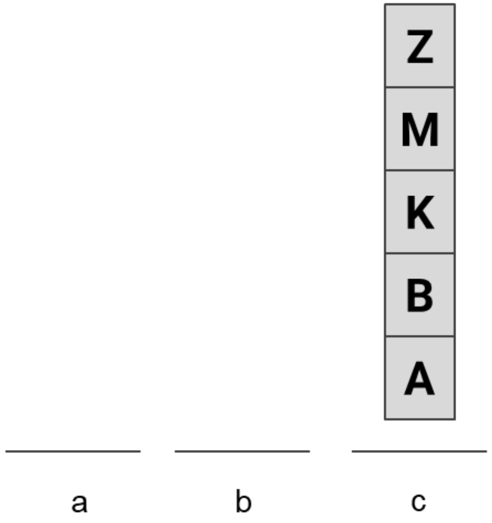
Source

Simple robotics (bloc movement) problem

Search space for actions from an initial to a final state



Final state



Transforming a statement into a search problem

Sometimes, the most difficult is expressing a description as a search problem

Example: Water Jug Problem

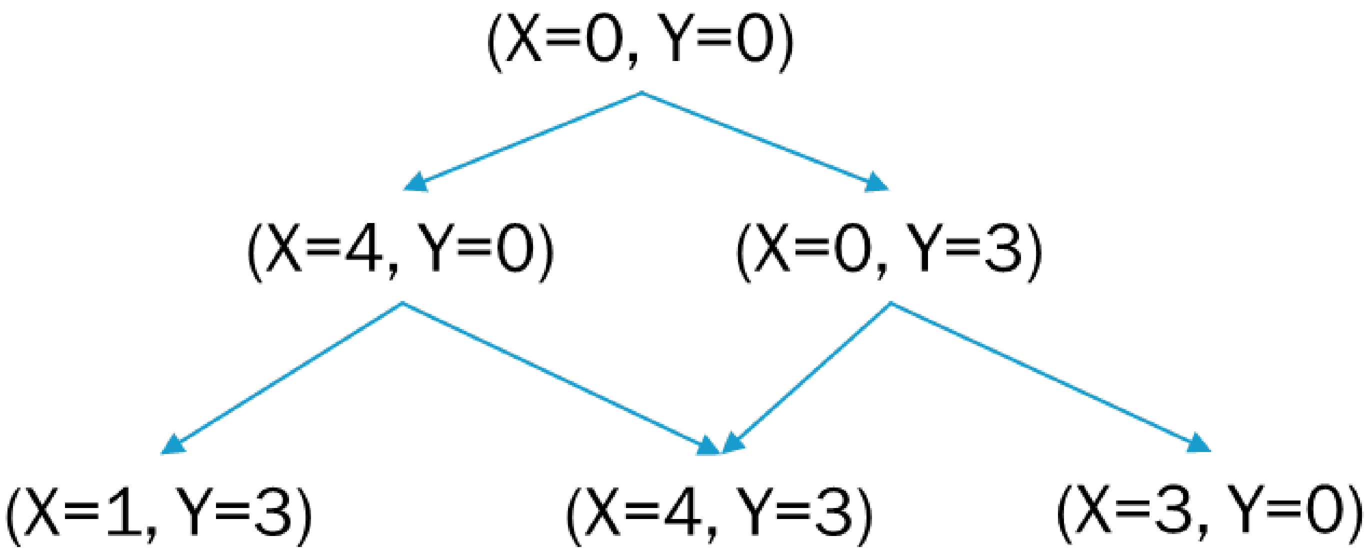
Consider the following problem:

> A Water Jug Problem: You are given two jugs, a 4-gallon one and a 3-gallon one, a pump which has unlimited water which you can use to fill the jug, and the ground on which water may be poured. Neither jug has any measuring markings on it. How can you get exactly 2 gallons of water in the 4-gallon jug?

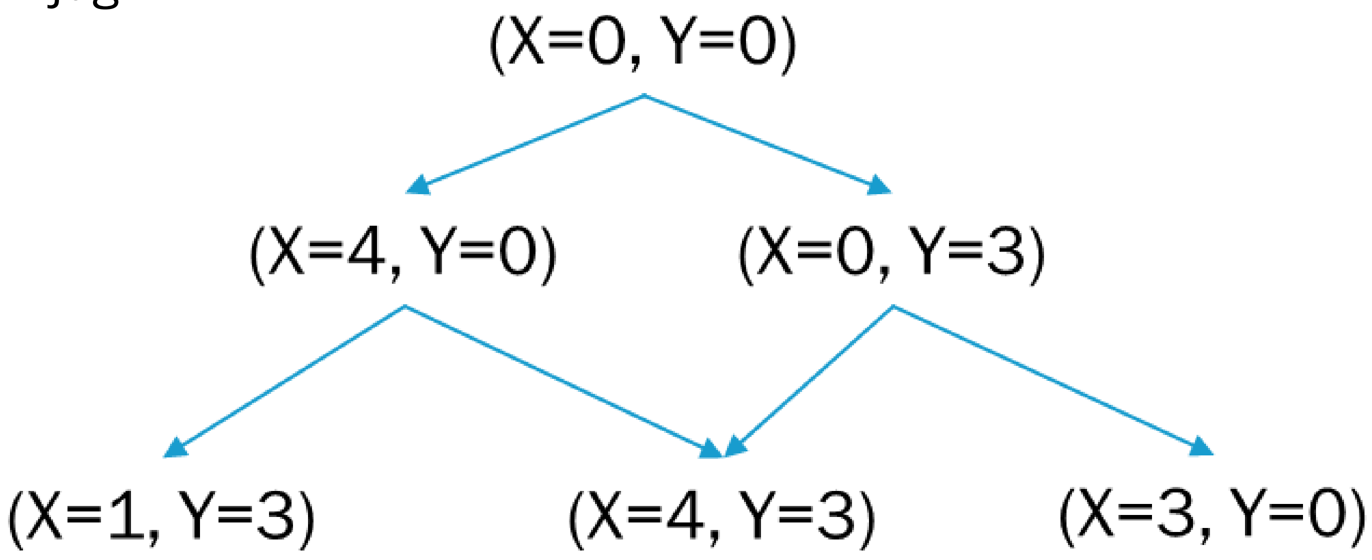
Source

X = 4-gallon jug
Y = 3-gallon jug

Nodes are states such that (X=1, Y=2) expresses the content of jugs X and Y
Edges are actions from one state to the other.



X = 4-gallon jug
Y = 3-gallon jug

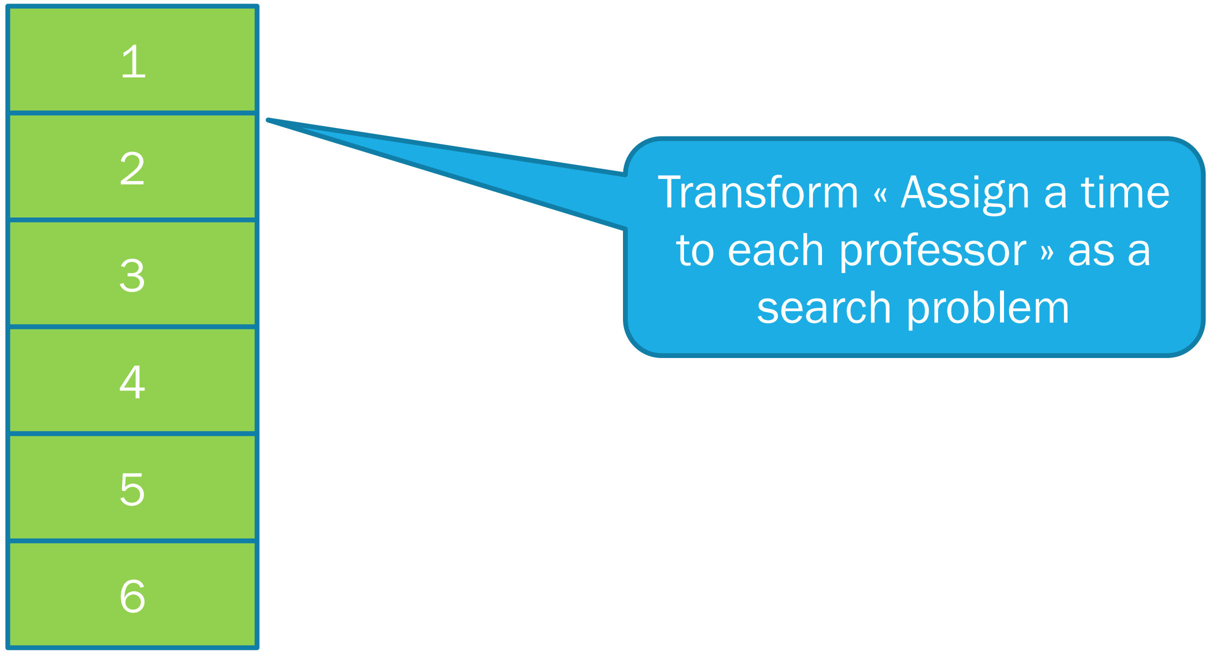


How would you express the end goal?

Scheduling problem

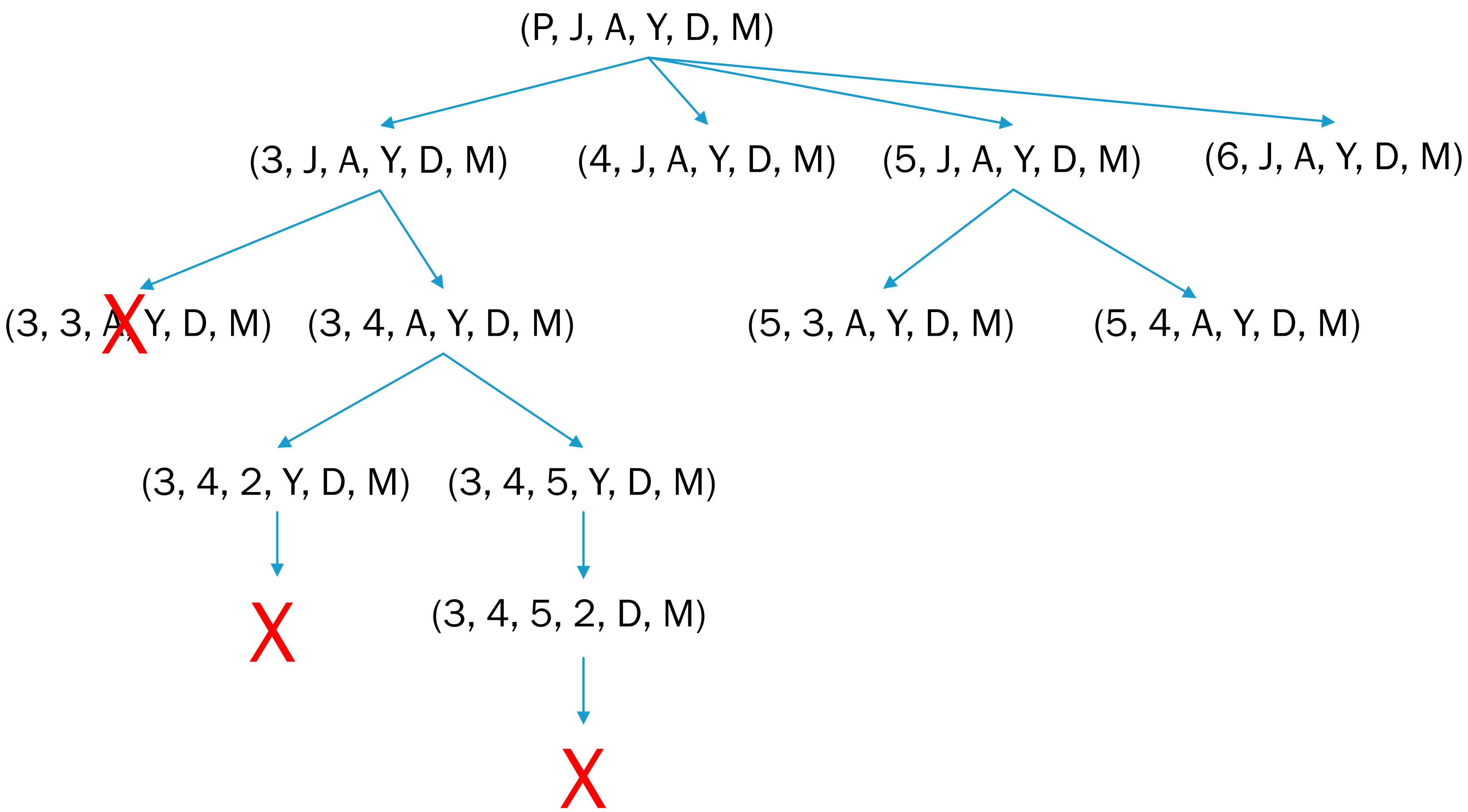
teacher	min	max
Peter	3	6
Jane	3	4
Anne	2	5
Yan	2	4
Dave	3	4
Mary	1	6

Each professor must teach one hour.
A single room is available.
Each professor has restrictions on when they can teach.



Scheduling problem

teacher	min	max
Peter	3	6
Jane	3	4
Anne	2	5
Yan	2	4
Dave	3	4
Mary	1	6



Scheduling problem

teacher	min	max
Peter	3	6
Jane	3	4
Anne	2	5
Yan	2	4
Dave	3	4
Mary	1	6

Search space is sometimes restricted by hard constraints, we can use them to speed up the search

(P, J, A, Y, D, M)

Solutions:

(6,3,5,2,4,1) ou (6,4,5,2,3,1)

Goal of algorithms :

- Searching for any solution

- Searching for a « good-enough » solution
  - Limited by processing time
  - Limited by number of actions in solution
  - Provided with a threshold (e.g. maximum time for an itinerary)

- Searching for an optimal solution:
  - Find the best solution.

- Search for all valid solutions.

- Detect where there is no solution.

What would be the goal for each problem presented?

- Suggested itineraries (Google Maps or other services)
- Suggested flight itineraries
- 8-puzzle game
- Manipulating a robot arm to move blocks
- Set of operations for filling a vase
- Assigning teachers to a class

IN SUMMARY

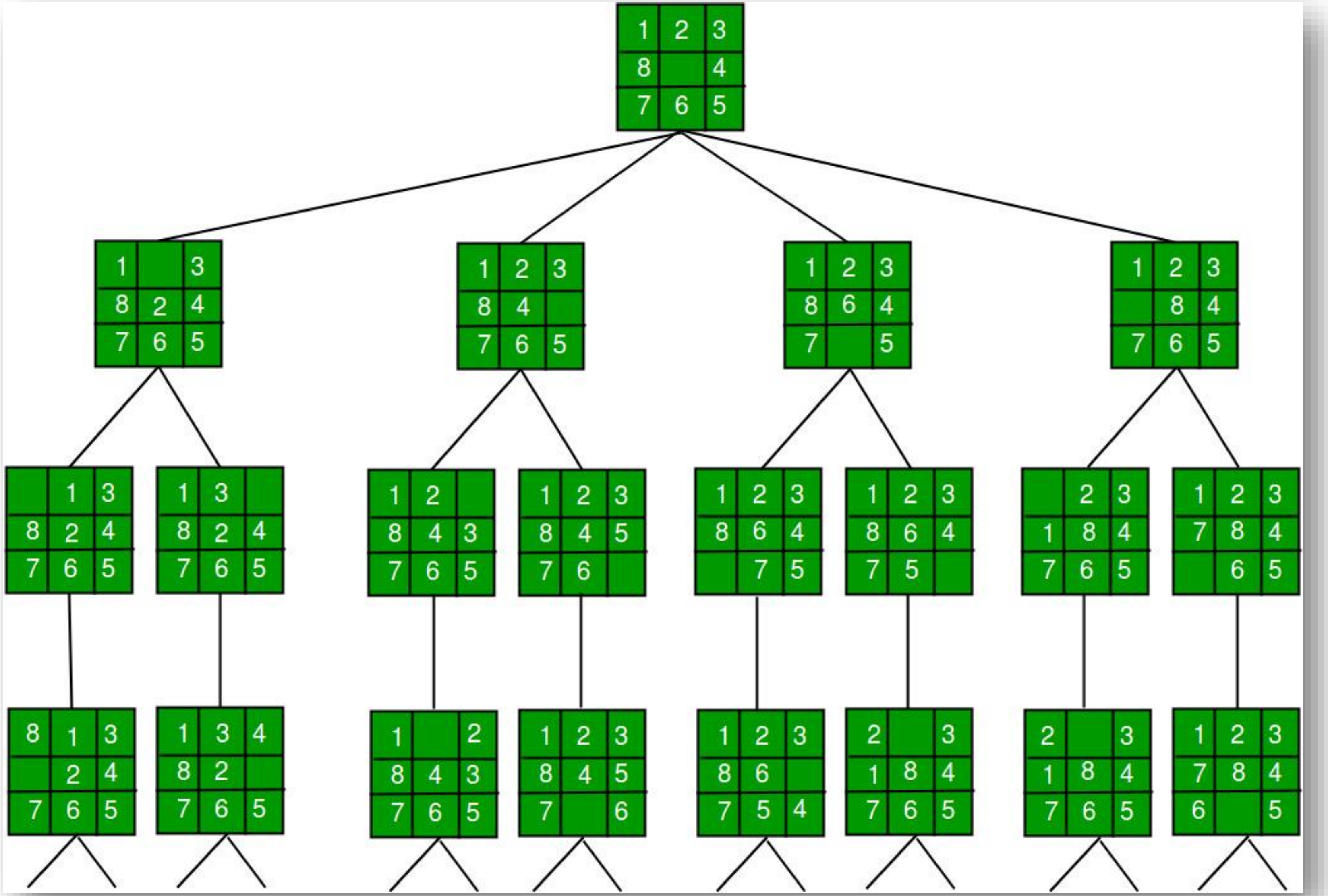
- Examples of search problems
- Characteristics of search problem solutions

Part 2
Blind searches

What characterizes a blind search?

- Representation of the space in the form of a directed graph (nodes/states + edges).
- No knowledge of the domain.
- No forward view, except for the list of neighboring nodes
- Memory of visited states/nodes.
- Memory of path used to visit the states.
- Ability to recognize the end state.

Game – 8 puzzle



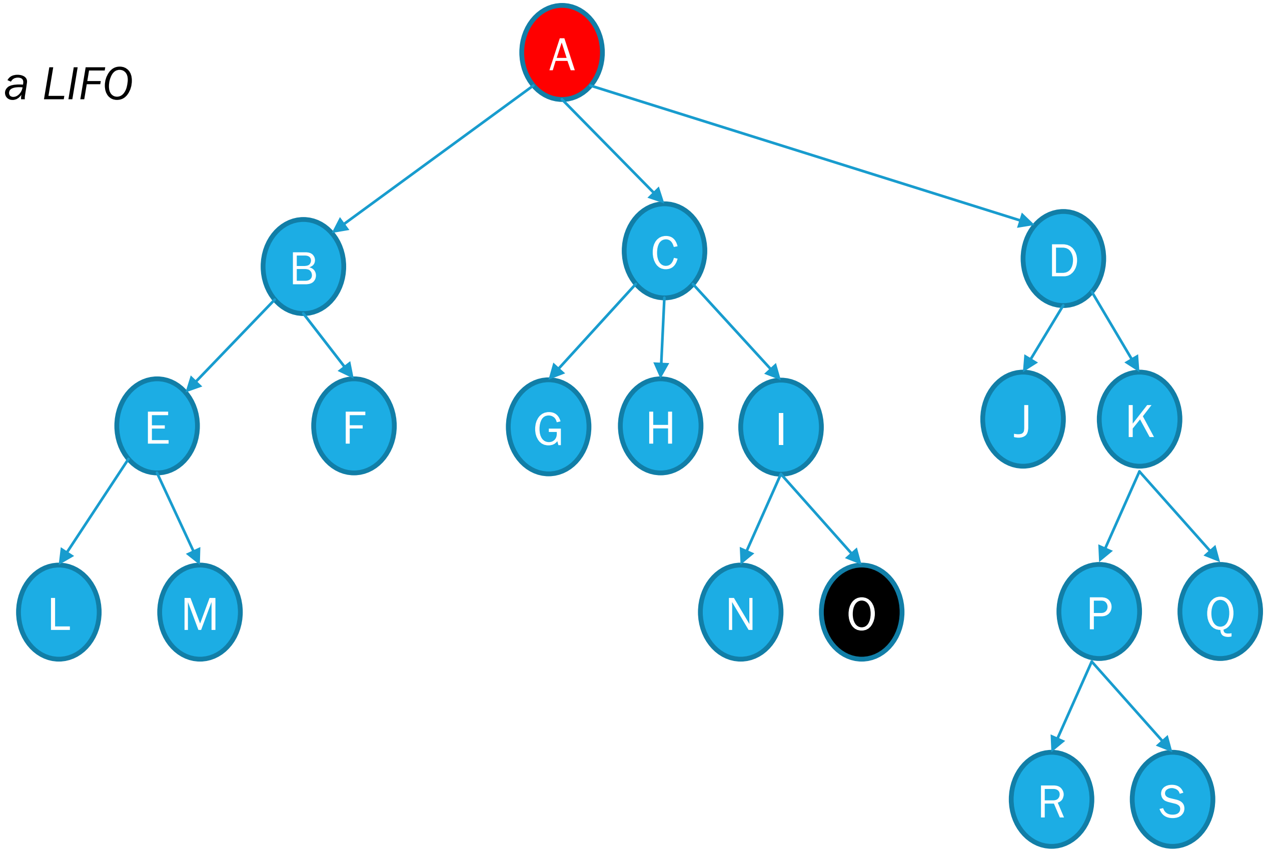
Source

Algorithms

- Depth-first search
- Breadth-first search
- Lowest-cost first search

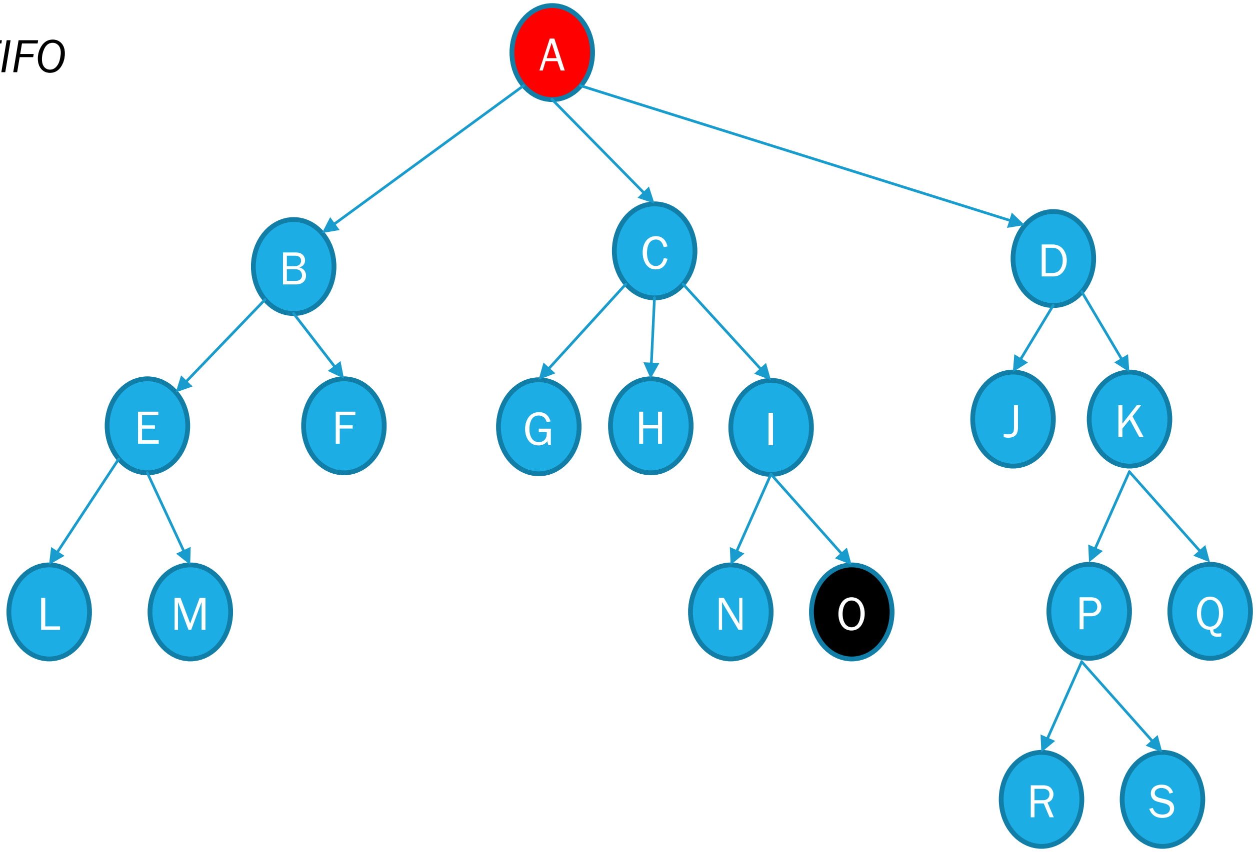
Depth-first Search

Implemented as a LIFO



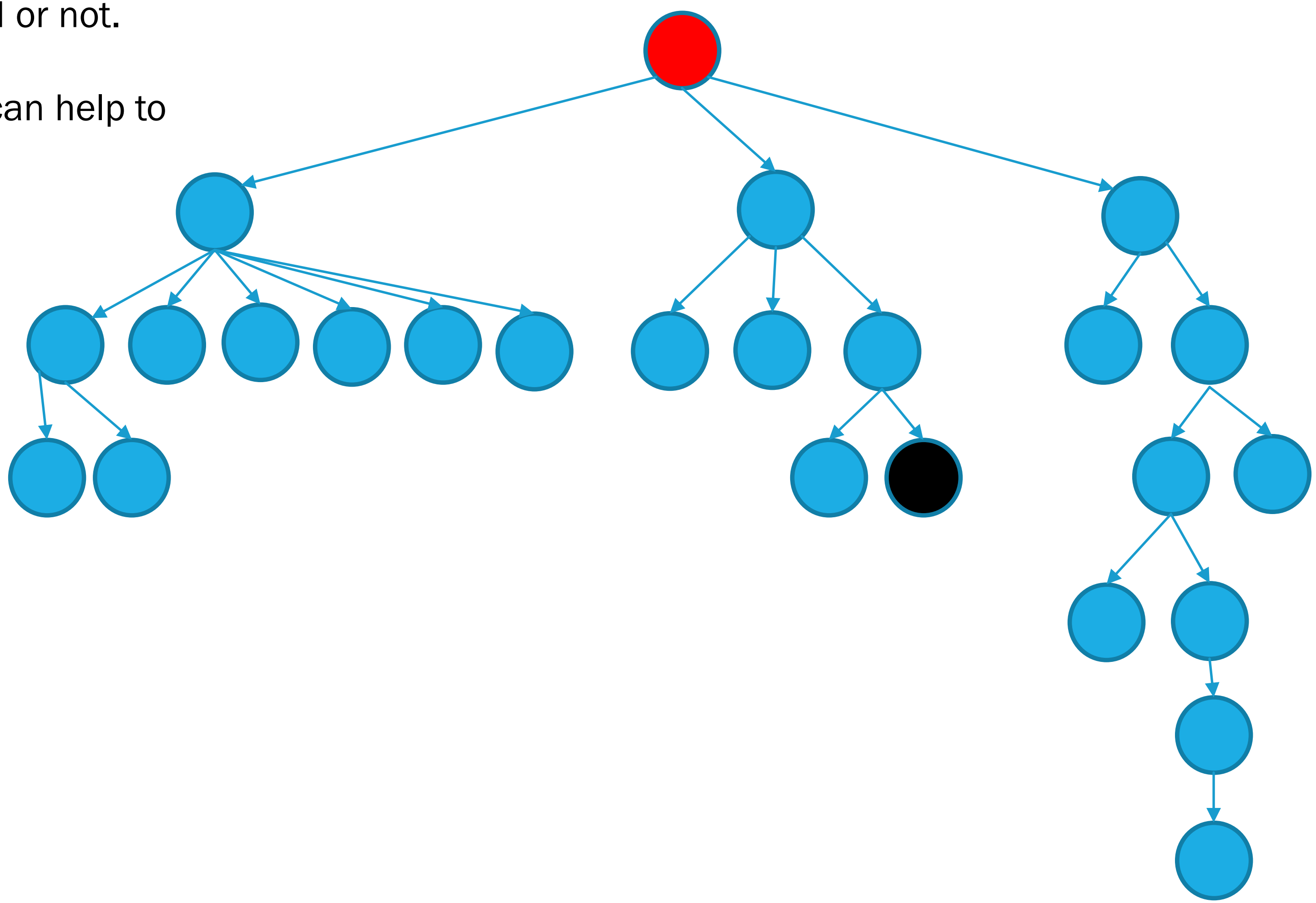
Breadth-first search

Implemented as a FIFO



Which to choose?

- Each approach can be problematic depending on whether the tree (or graph) is balanced or not.
- A minimum knowledge of the domain can help to know the topology of the graph.

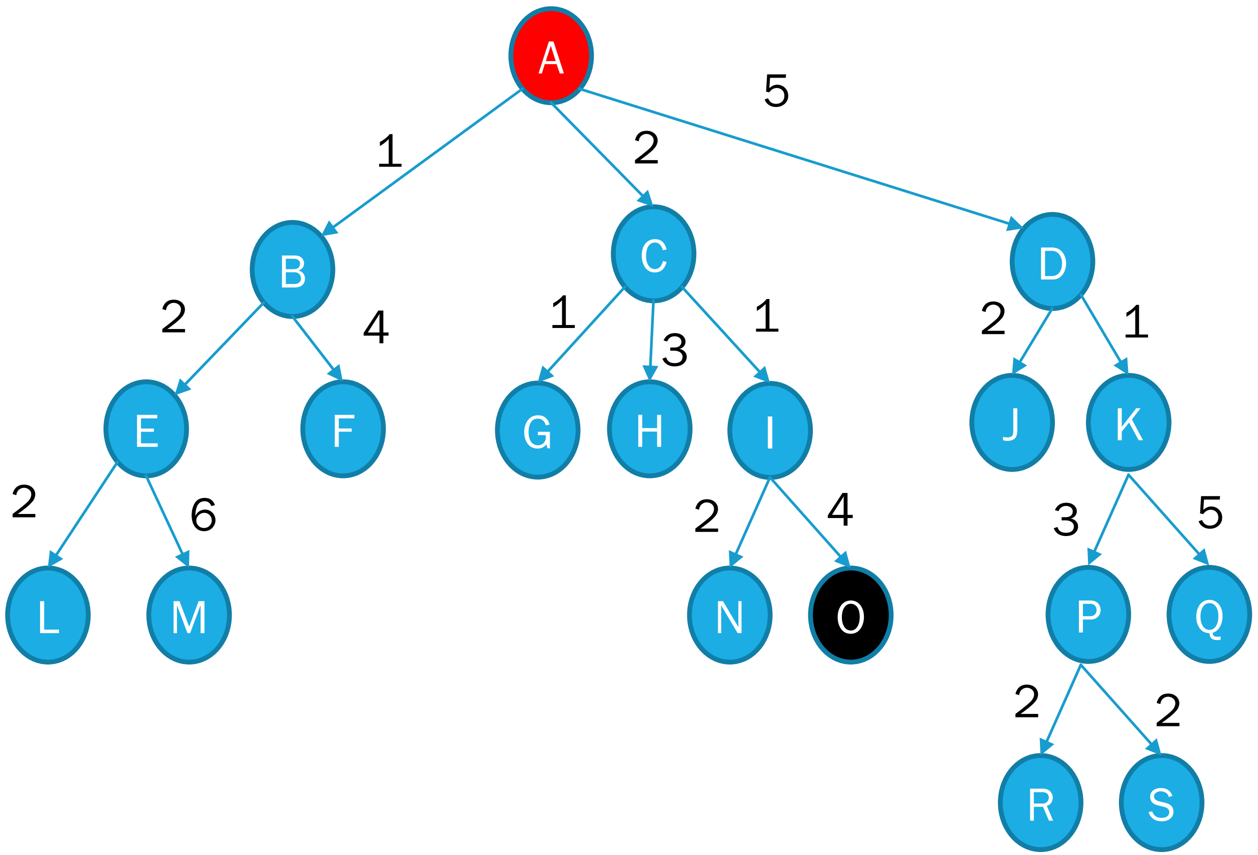


Lowest-cost first search

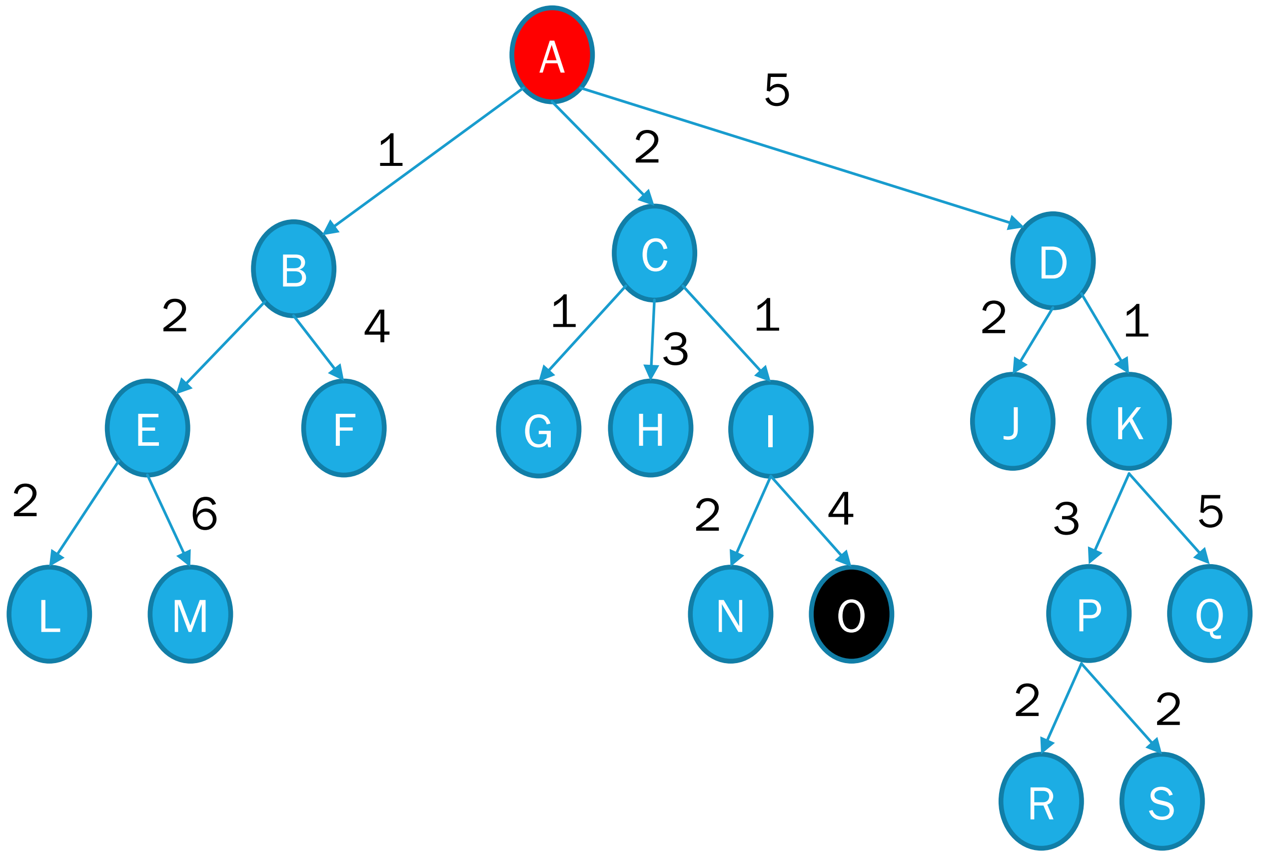
Contrarily to depth-first or breadth-first, the edges are not of uniform cost.

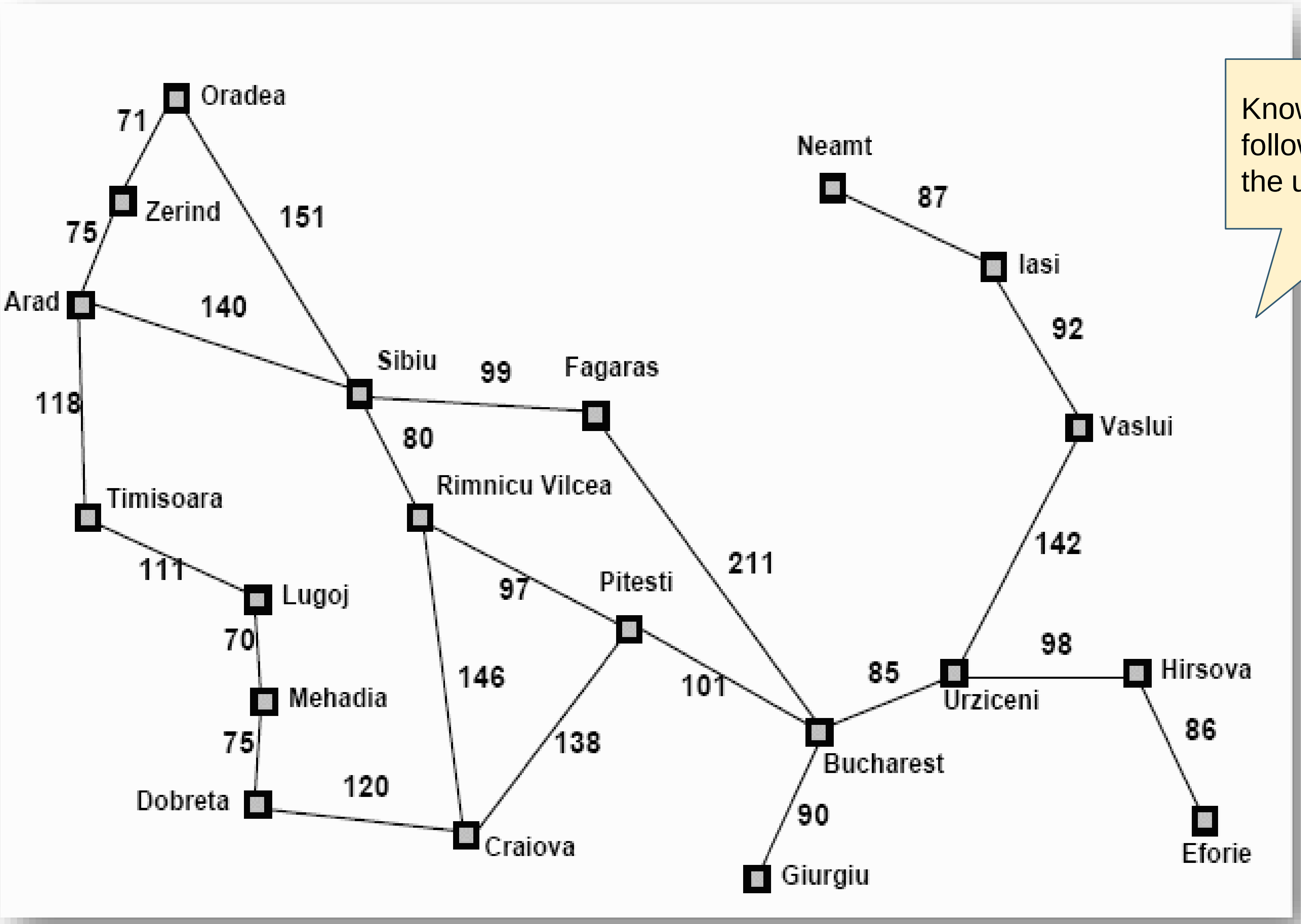
Search for OPTIMAL solution.

ATTENTION: The "lowest-cost" used is the cost of the path done so far.

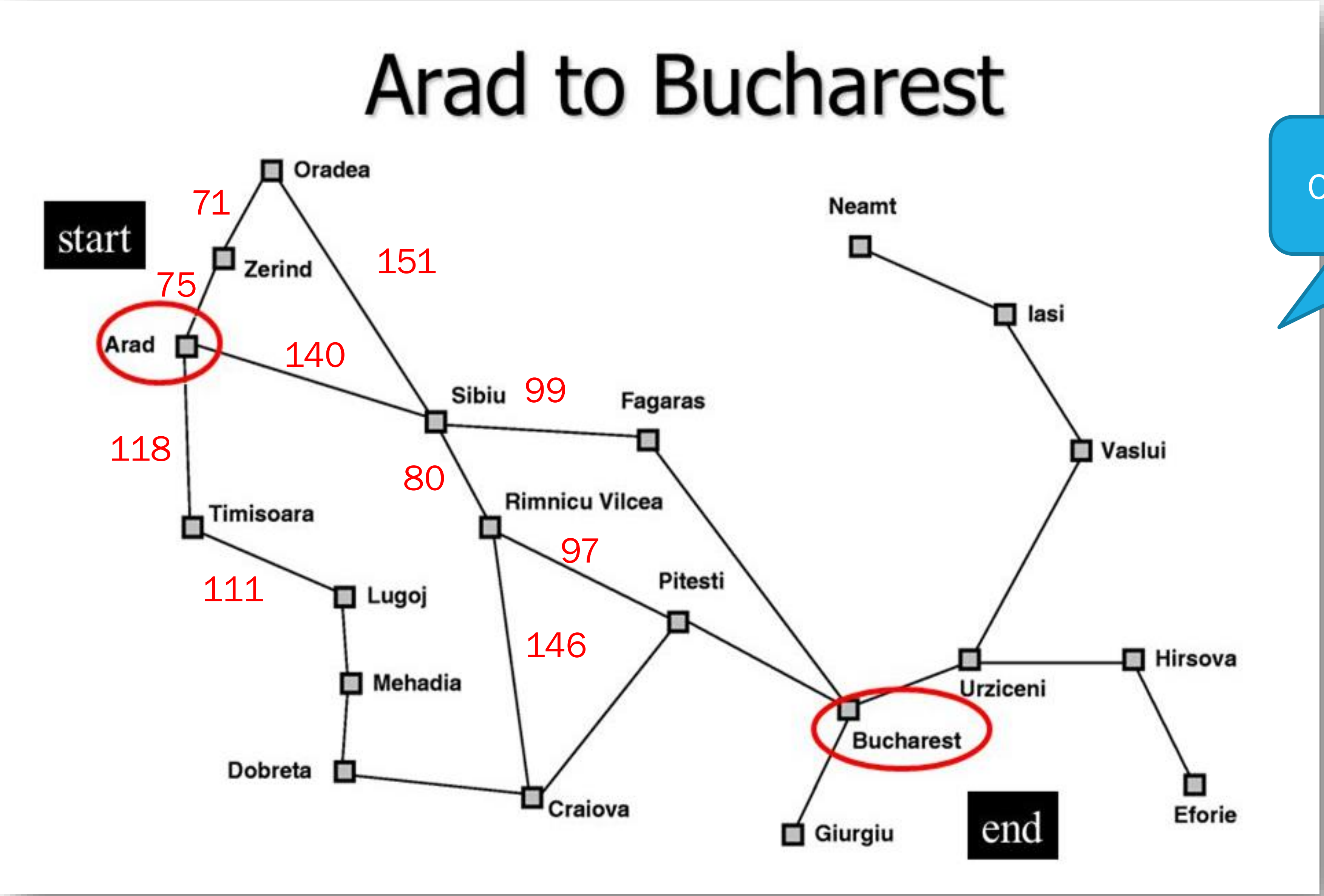


Lowest-cost first search





Knowledge of path followed so far (not the upcoming one).



Continue the search

IN SUMMARY

- Definition of blind search
- Three algorithms:
  - Depth-first search
  - Breadth-first search
  - Lowest-cost-first search

Part 3
Heuristic Searches

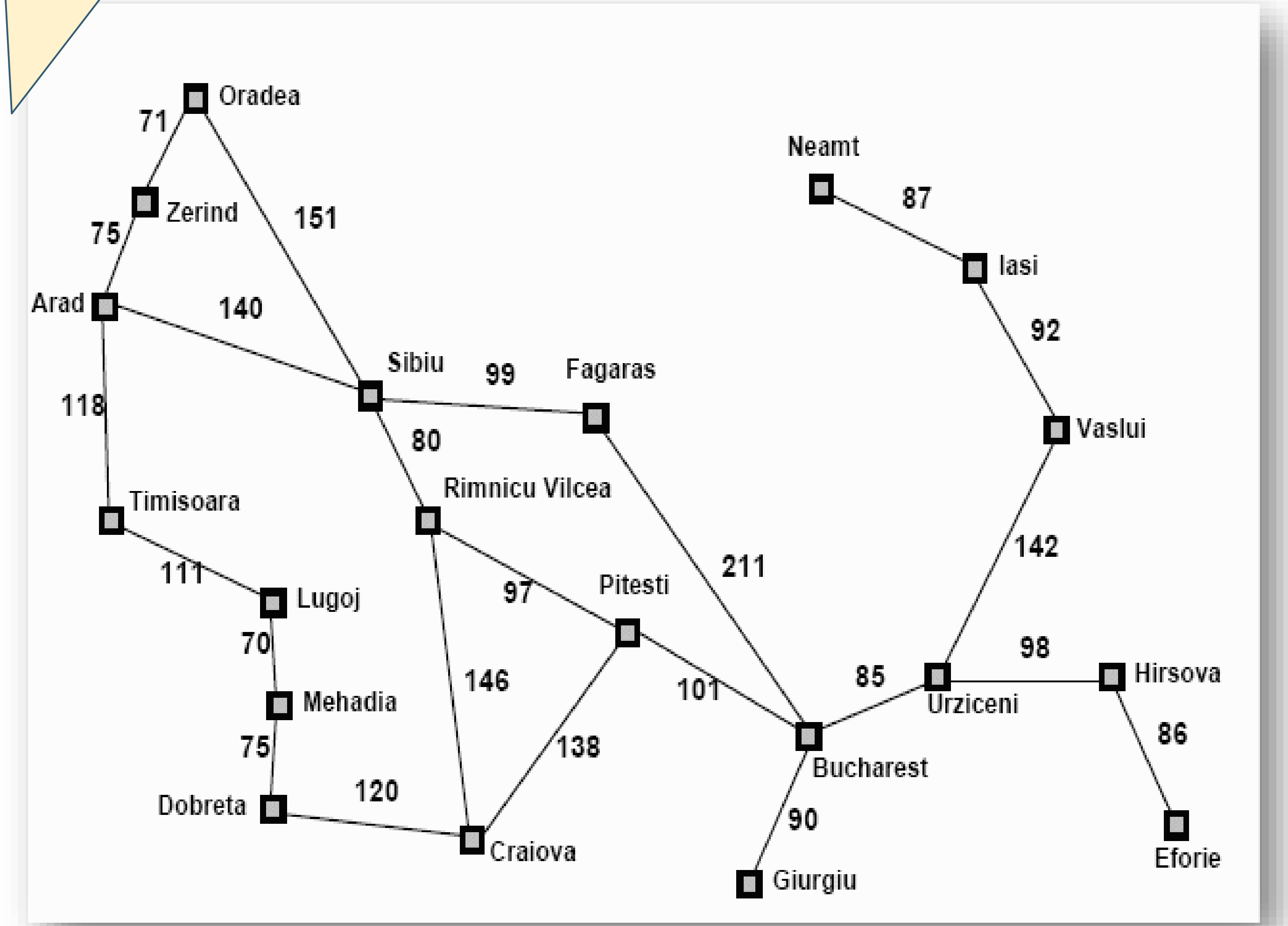
Chapter 3, Searching for Solutions, **Section 3.6 Heuristic Search,** from Artificial Intelligence, Foundations of Computational Agents

A heuristic search considers **knowledge of the problem** to explore some (not all) successors of the current state. This means pruning the state space, gaining speed, but perhaps missing the solution!

As opposed to blind search, a heuristic search has a "look-ahead" method it can use to **estimate how close a state is to the goal**.

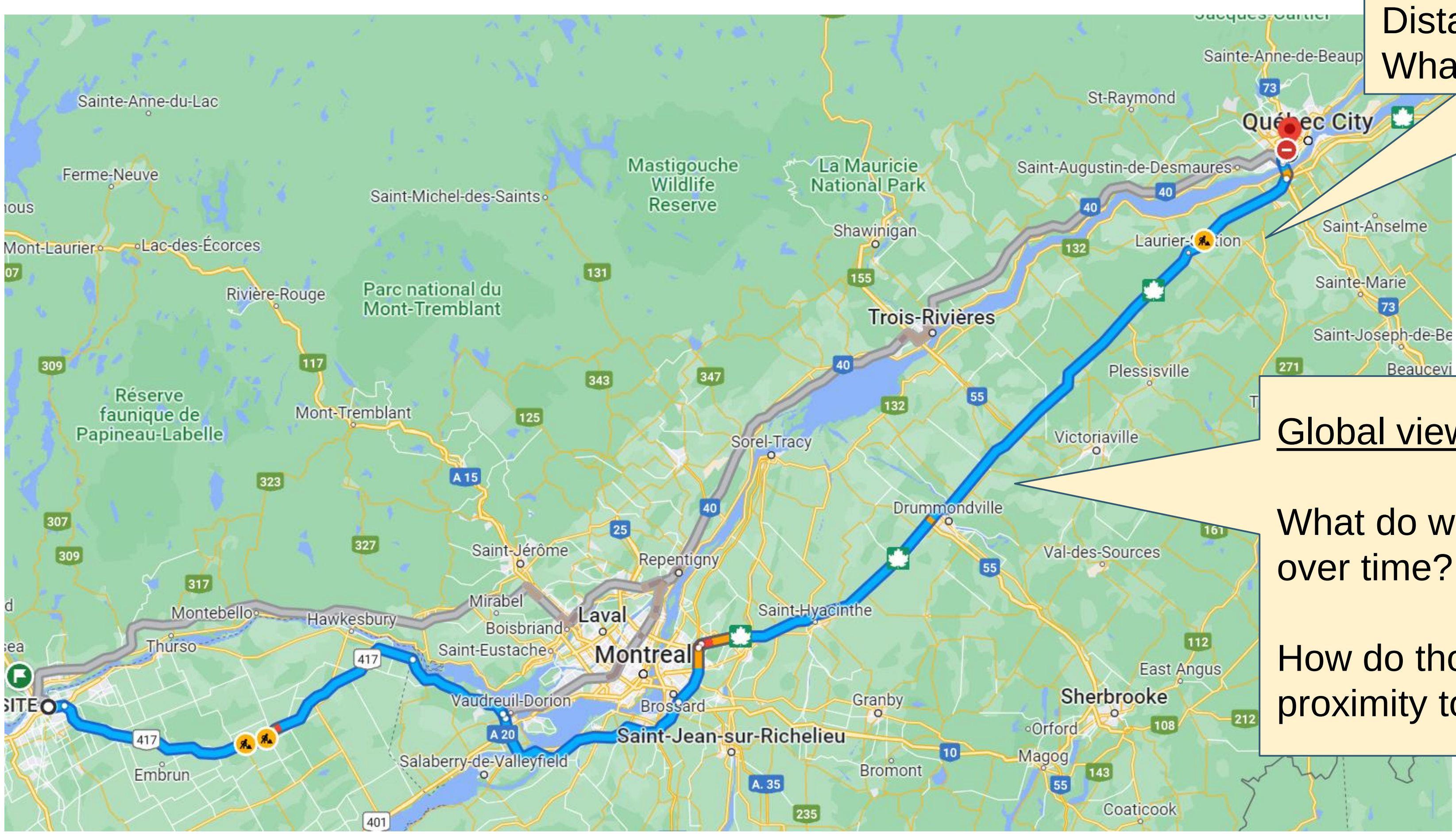
Local view:
Distance to next city.

Global view:
Underestimate. Straight line distance.



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



<u>Local view:</u>

Distance to next city.
What else??

<u>Global view:</u>

What do we know from statistics gathered over time?

How do those help provide an estimate of the proximity to the goal?

Important factor in route planning.
What does the delivery robot sees and/or knows?

Local: its position and the surrounding rooms

Global: full floor plan

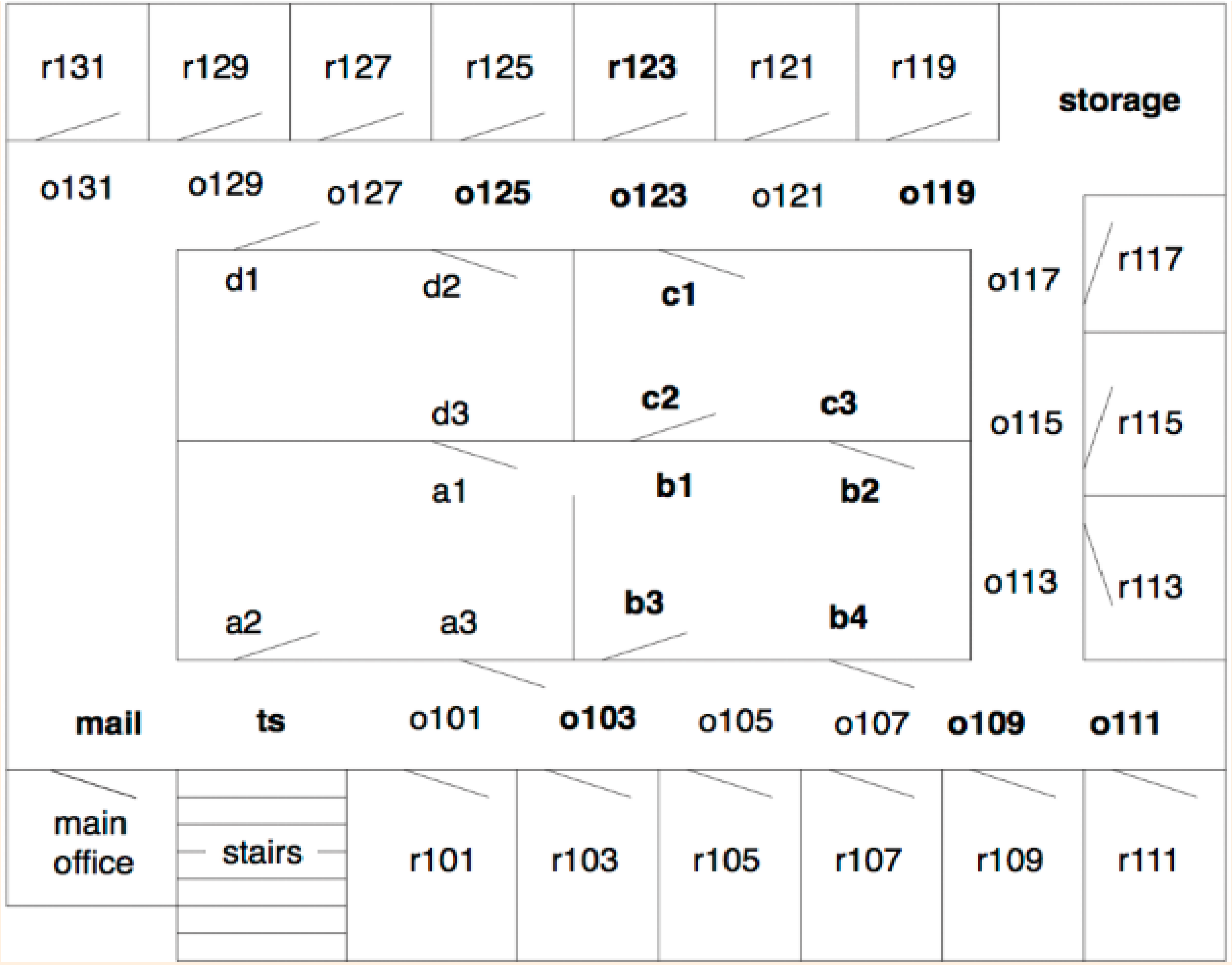
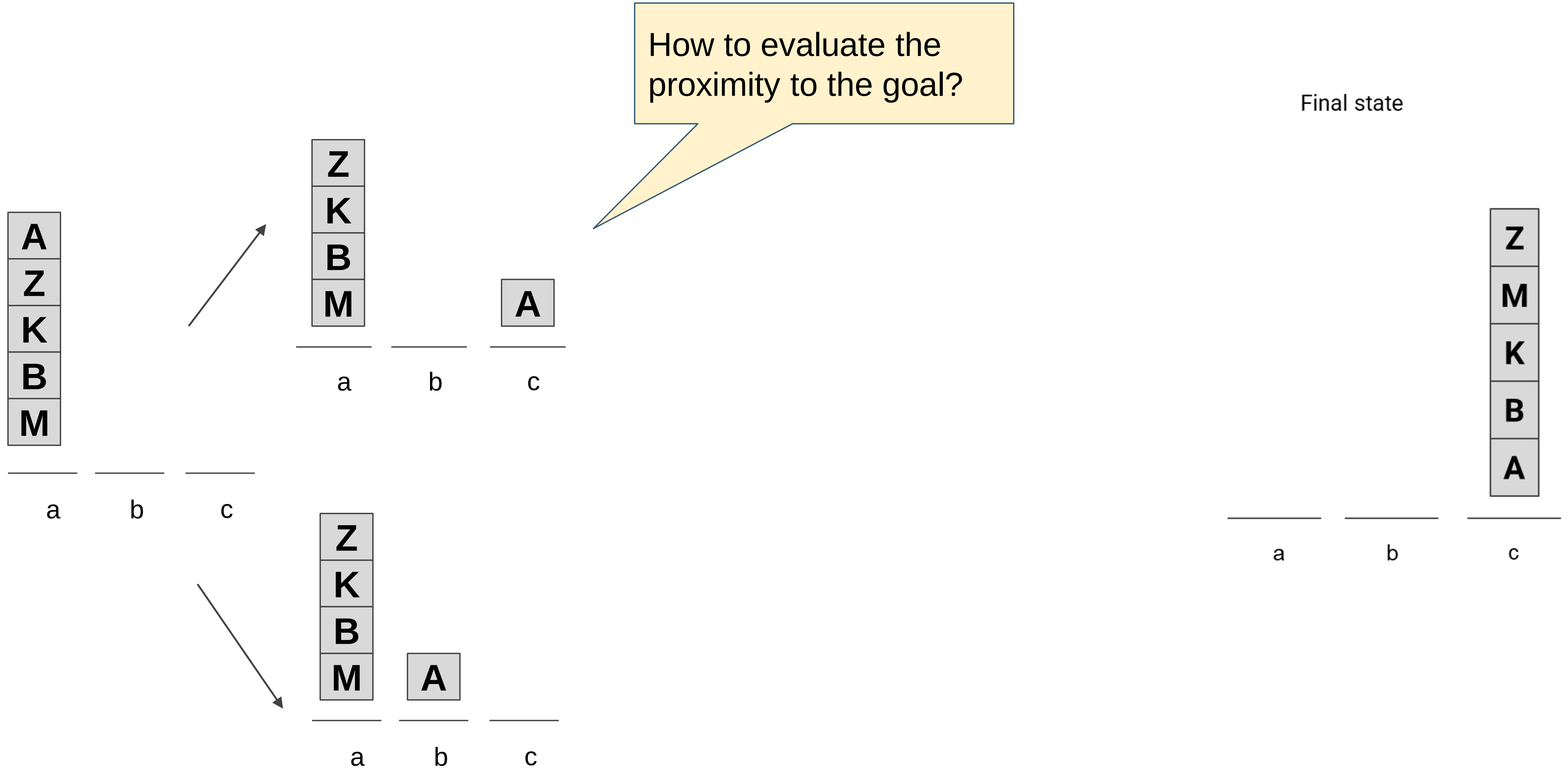
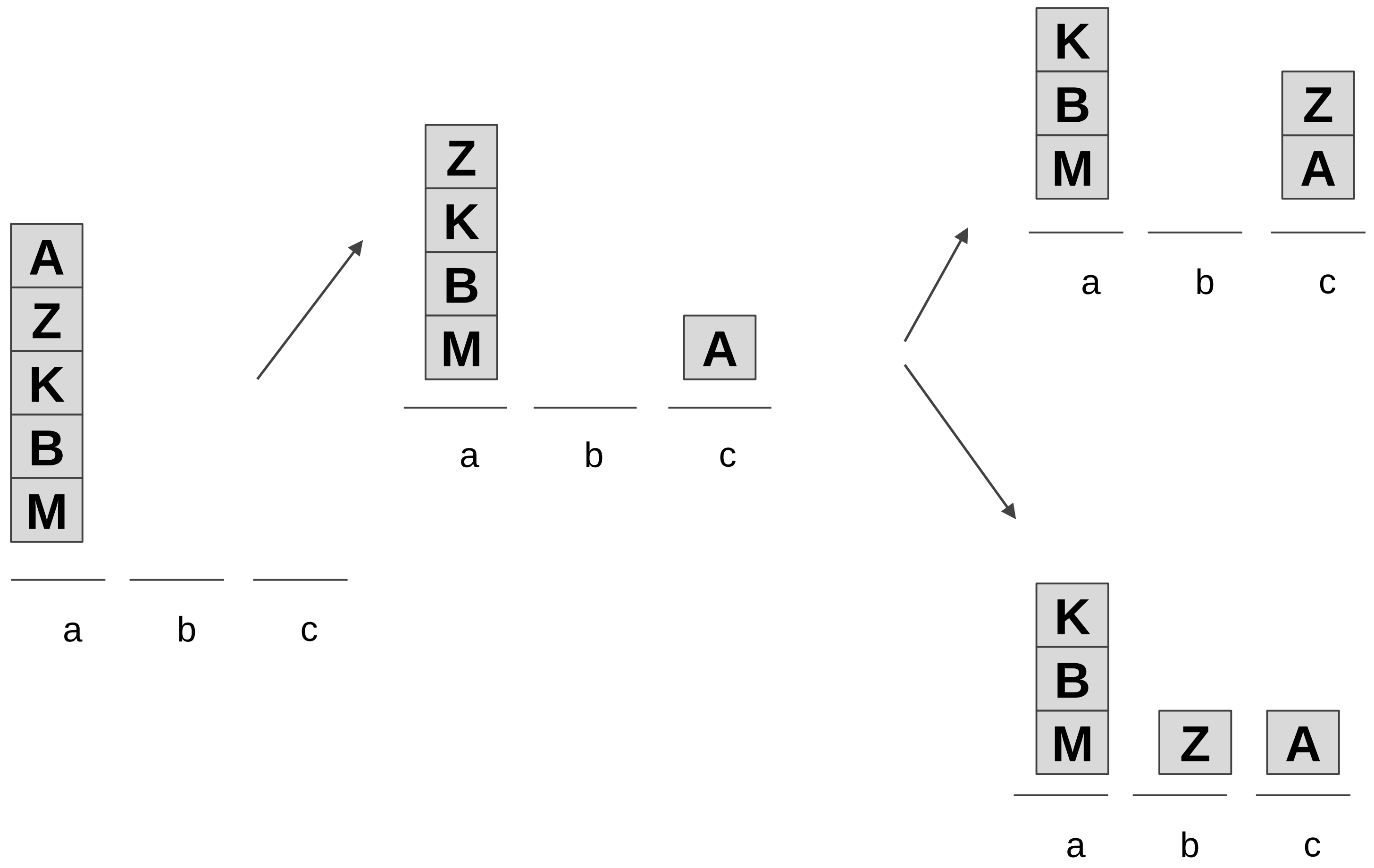
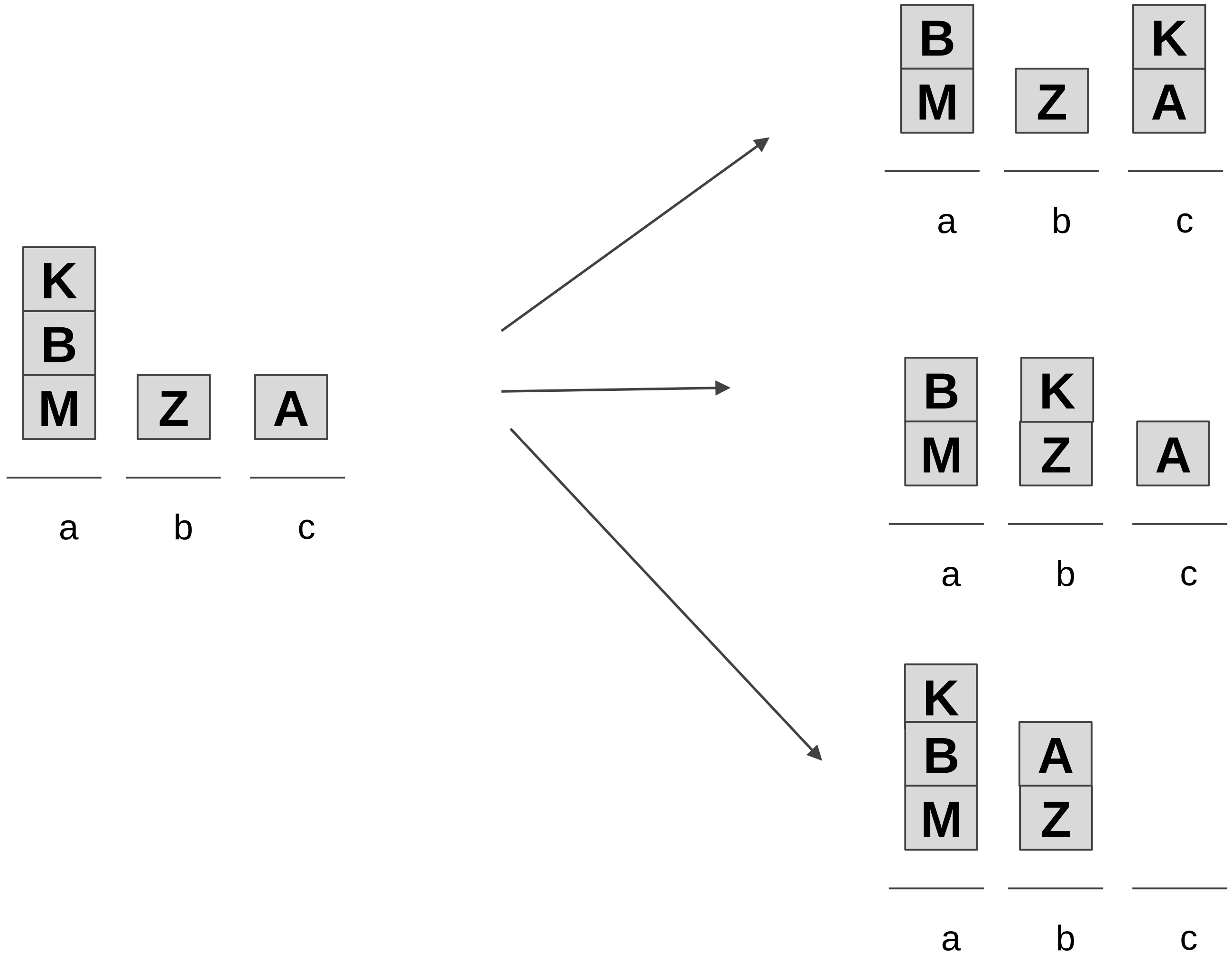


Figure 3.1: The delivery robot domain with interesting locations labeled


How to evaluate the proximity to the goal?
Final state
A
Z
K
B
M
a
b
c
Z
K
B
M
A
a
b
c
Z
K
B
M
A
a
b
c
Z
M
K
B
A
a
b
c



A good “heuristic” would provide a proximity to the goal.

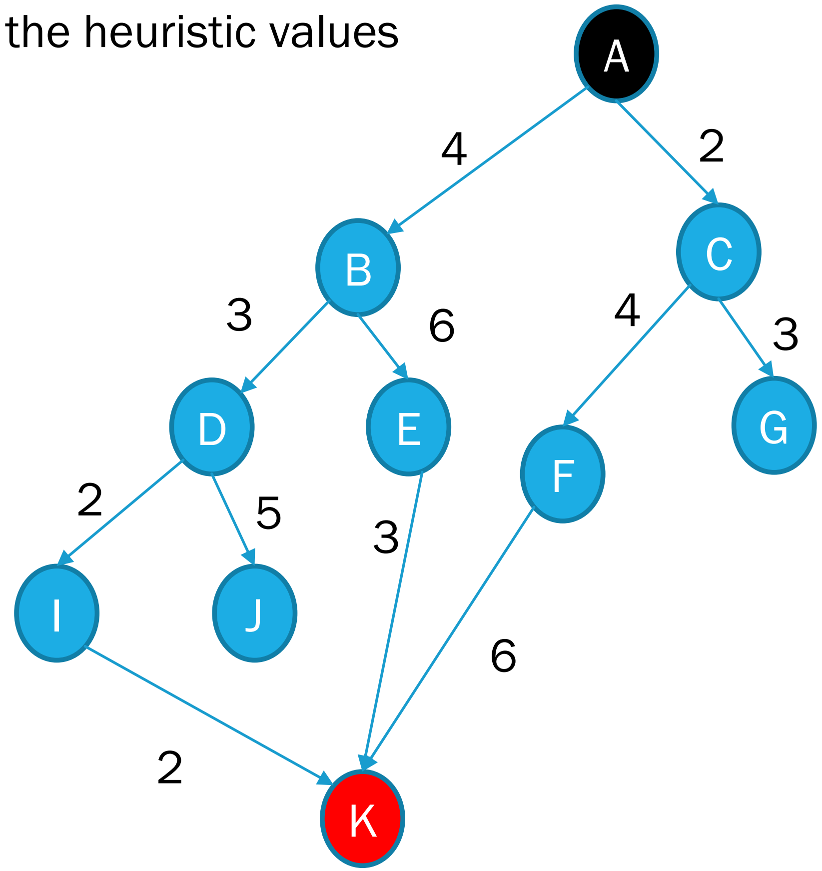

K
B
M
Z
A
a
b
c
B
M
Z
K
A
a
b
c
B
M
K
Z
A
a
b
c
K
B
M
A
Z
a
b
c

Algorithms

- Best-first search
- A*

Best-First Search

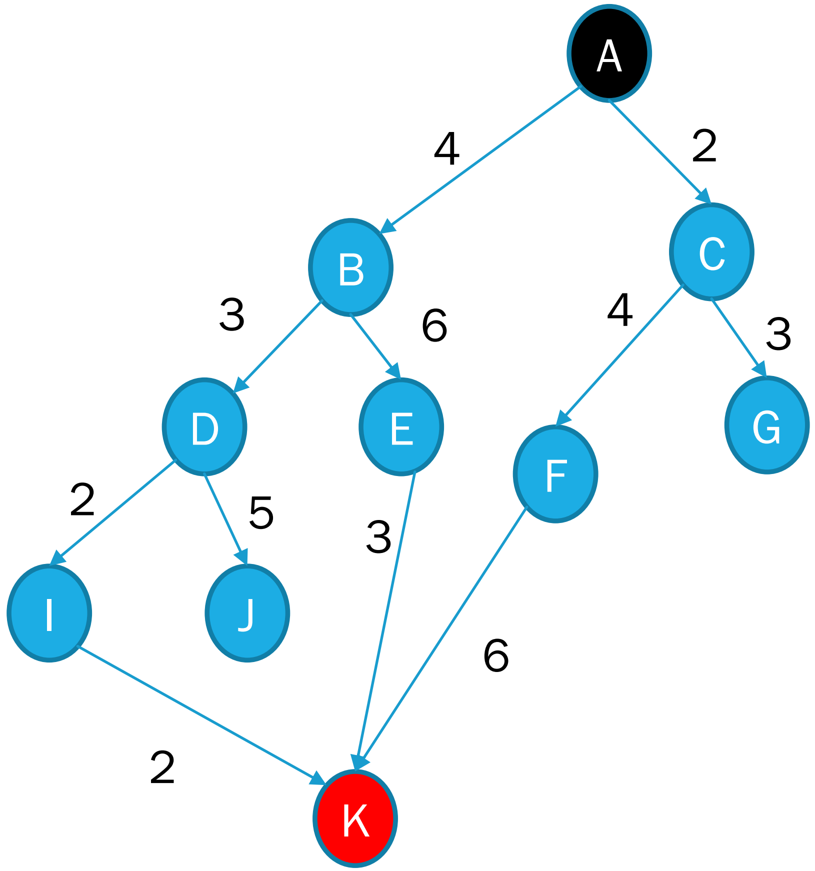
- Non-optimal.
- This algorithm completely trusts the heuristic values



Estimated distance to K

H(A) = 10
H(B) = 7
H(C) = 8
H(D) = 4
H(E) = 3
H(F) = 6
H(G) = 20
H(I) = 2
H(J) = 20

Best-First Search



Estimated distance to K

H(A) = 10
H(B) = 7
H(C) = 8
H(D) = 4
H(E) = 3
H(F) = 6
H(G) = 20
H(I) = 2
H(J) = 20

Opening order :

A --- B(7), C(8)
B -- D(4), E(3)
E -- K
K

A* algorithm

Optimal solution.

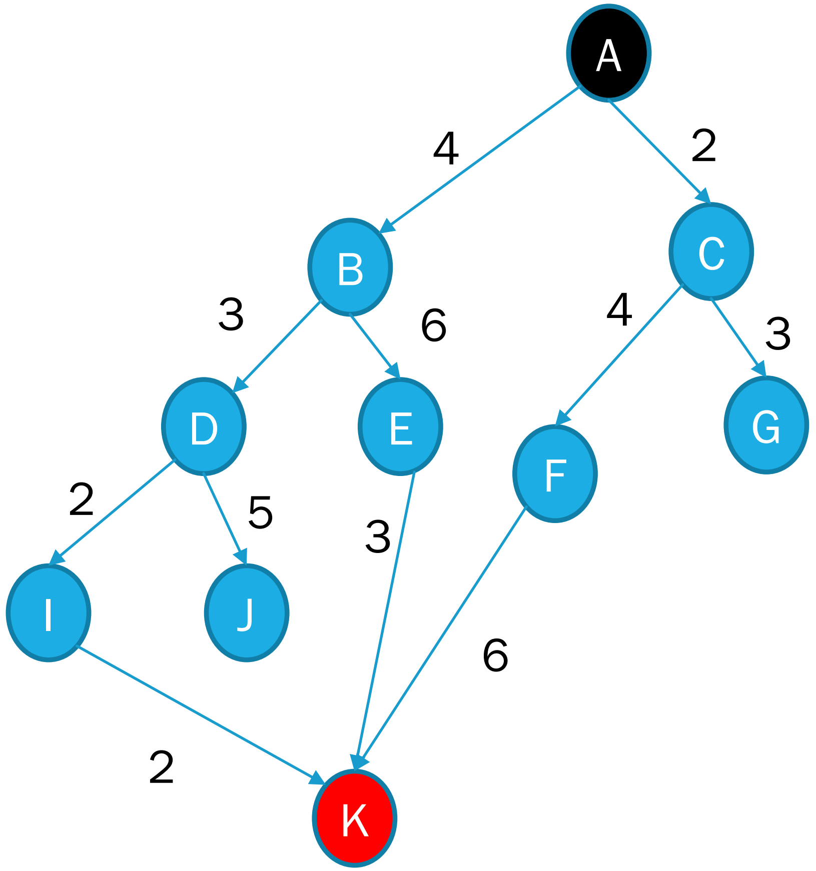
A* search combines the lowest-cost search and the best-first search. It looks behind and in front.

The evaluation function for the choice of the path to pursue on the frontier becomes:

$$\underbrace{\underbrace{start \xrightarrow{\text{path } p} n}_{cost(p)} \underbrace{\xrightarrow{\text{estimate}} goal}_{h(p)}}_{f(p)}$$

For A to lead to an optimal solution,*
h(p) must be an underestimate.

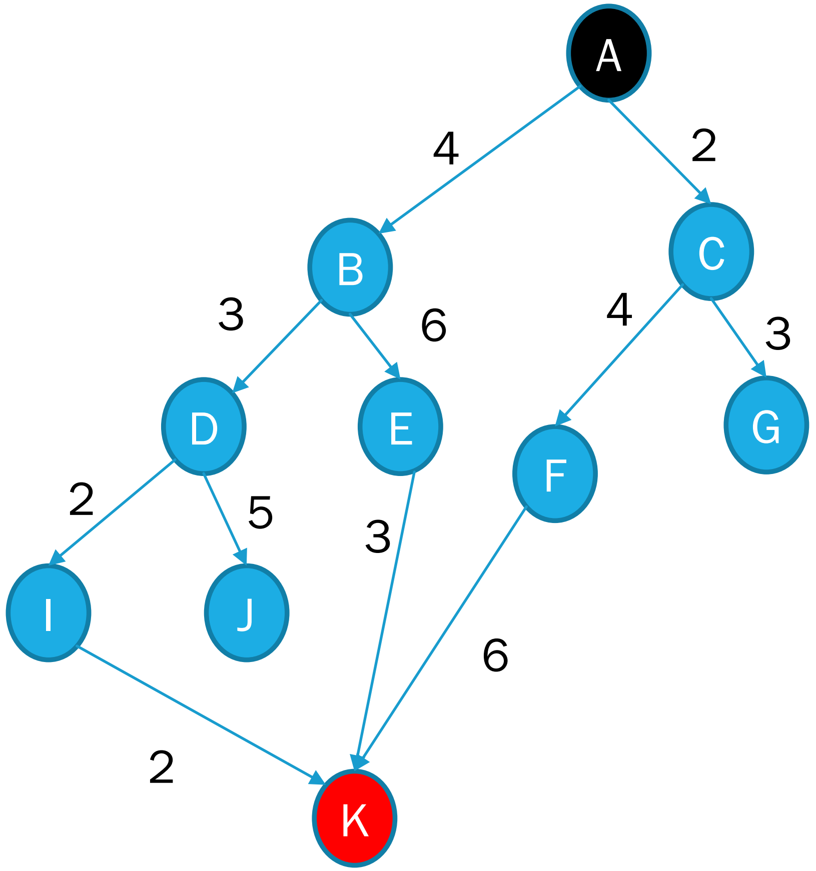
A* Search



Estimated distance to K

H(A) = 10
H(B) = 7
H(C) = 8
H(D) = 4
H(E) = 3
H(F) = 6
H(G) = 20
H(I) = 2
H(J) = 20

A* Search



Estimated distance to K

H(A) = 10
H(B) = 7
H(C) = 8
H(D) = 4
H(E) = 3
H(F) = 6
H(G) = 20
H(I) = 2
H(J) = 20

Opening order:

A --- B(4+7), C(2+8)
C -- F(2+4+6), G(2+3+20)
B -- D(4+3+4), E(4+6+3)
D – I (4+3+2+2), J(4+3+5+20)
I - K (4+3+2+2)

Path: A-B-D-I-K

IN SUMMARY

- Definition of heuristic search
- Presentation of two algorithms:
  - Best-first Search
  - A*

SOLUTION SPACES

- Part 1 – Examples of search problems
- Part 2 – Blind searches (review)
- Part 3 – Heuristic searches