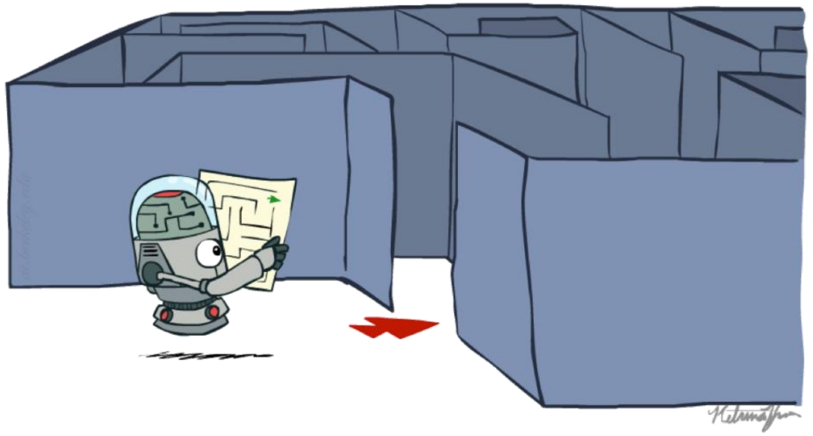
Problem Solving Using Search

Introduction to State Space Search



Instructor: Dr. Mohsin Ashraf

Slide credits: AI BERKELEY

Previous Lecture

- Intelligent Agent
- Performance measure
- Different types of Environments
- IA examples based on Environment
- Agent types
  - Simple reflex agents
  - Reflex agents with state/model
  - Goal-based agents
  - Utility-based agents

Introduction

- Suppose an agent can execute several actions immediately in a given state
- It doesn’t know the utility of these actions
- Then, for each action, it can execute a sequence of actions until it reaches the goal
- The immediate action which has the best sequence (according to the performance measure) is then the solution
- Finding this sequence of actions is called search, and the agent which does this is called the problem-solver.
- NB: Its possible that some sequence might fail, e.g., getting stuck in an infinite loop, or unable to find the goal at all.


T 69
LISDOONVARNA 10
LIOS DUIN BHEARNA
LISDOONVARNA 19
VIA COAST ROAD
51 KILLIMER CAR FERRY
CORA FINNE
COROFIN 15
TEA & GARDEN ROOMS
km GREGANS
5 CASTLE HOTEL
RUSHEEN LODGE
B & B
AILLWEE CAVE
BALLYVAUGHAN
OPEN TO VISITORS
Burren
Dolmen
West Clare
Cycleways
2 km NEWTOWN CASTLE & TRAIL

Problem-Solving Agent

- **Problem solving agent** is a **goal-based agent** that decides what to do by **systematically** finding sequences of actions that lead to desirable states (goal)

- Many problems can be represented as a **set of states** and a **set of rules** of how one state is transformed to another.

Examples

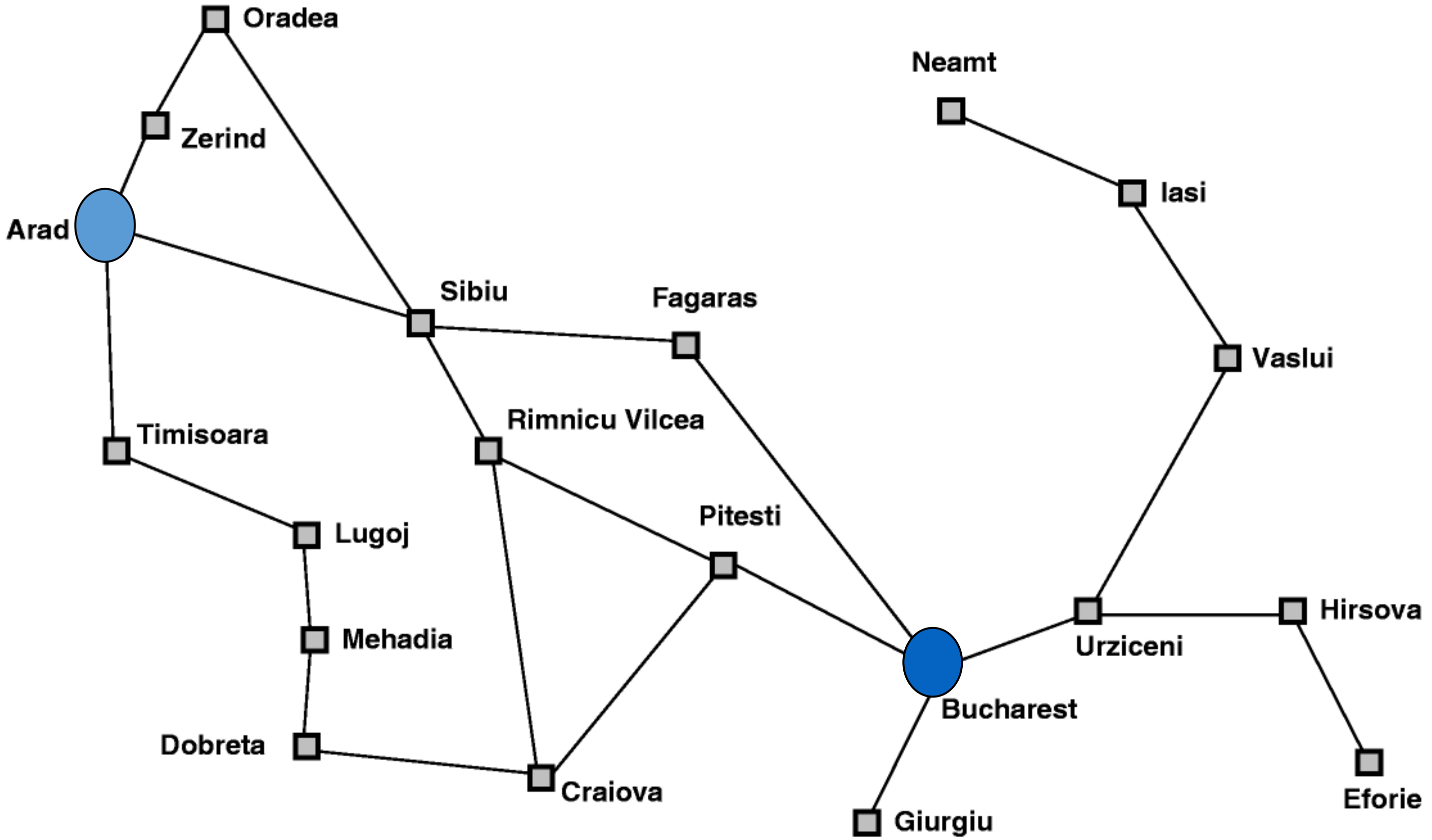
- Getting from home to UCP
    - **START**: home location
    - **GOAL**: UCP FIT Department.
    - **OPERATORS**: move one block, turn
- Loading a moving truck
    - **START**: apartment full of boxes and furniture
    - **GOAL**: empty apartment, all boxes and furniture in the truck
    - **OPERATORS**: select item, carry item from apartment to truck, load item
- Getting settled
    - **START**: items randomly distributed over the place
    - **GOAL**: satisfactory arrangement of items
    - **OPERATORS**: select item, move item

Example Problems

- **Toy Problems**
  - vacuum world
  - 8-puzzle
  - 8-queens
  - cryptarithmetic
  - vacuum agent
  - missionaries and cannibals
- **Real-world Problems**
  - route finding
  - touring problems
    - traveling salesperson
  - VLSI layout
  - robot navigation
  - assembly sequencing
  - Web search

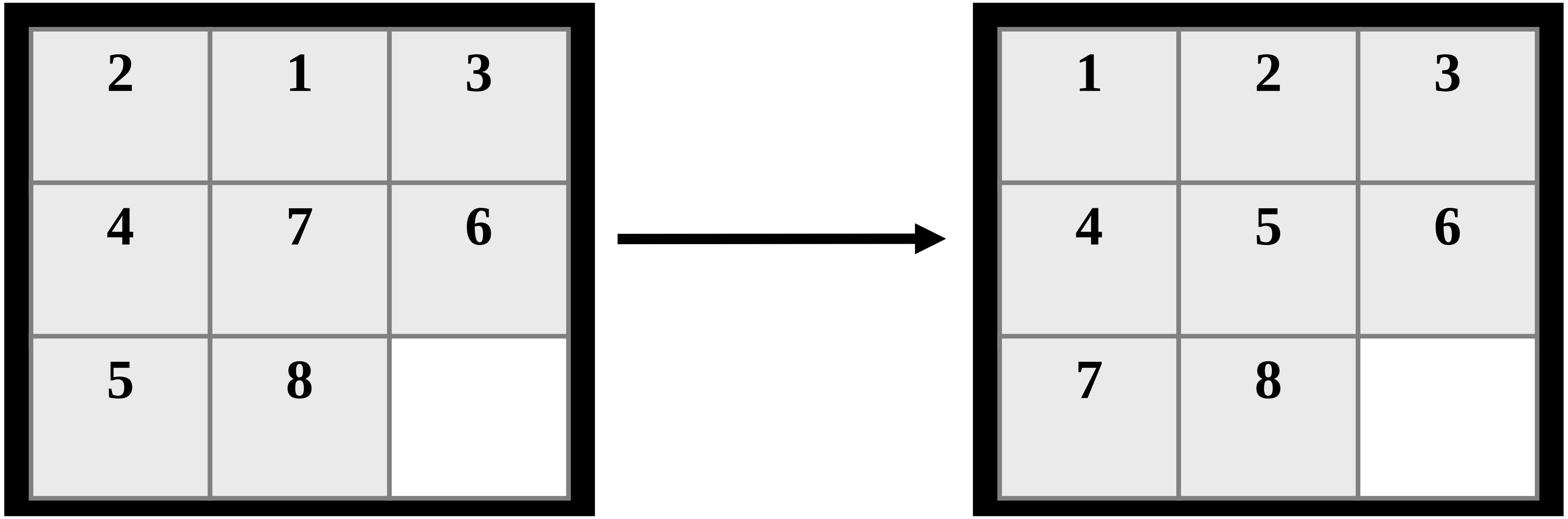
Classical AI Search Problem

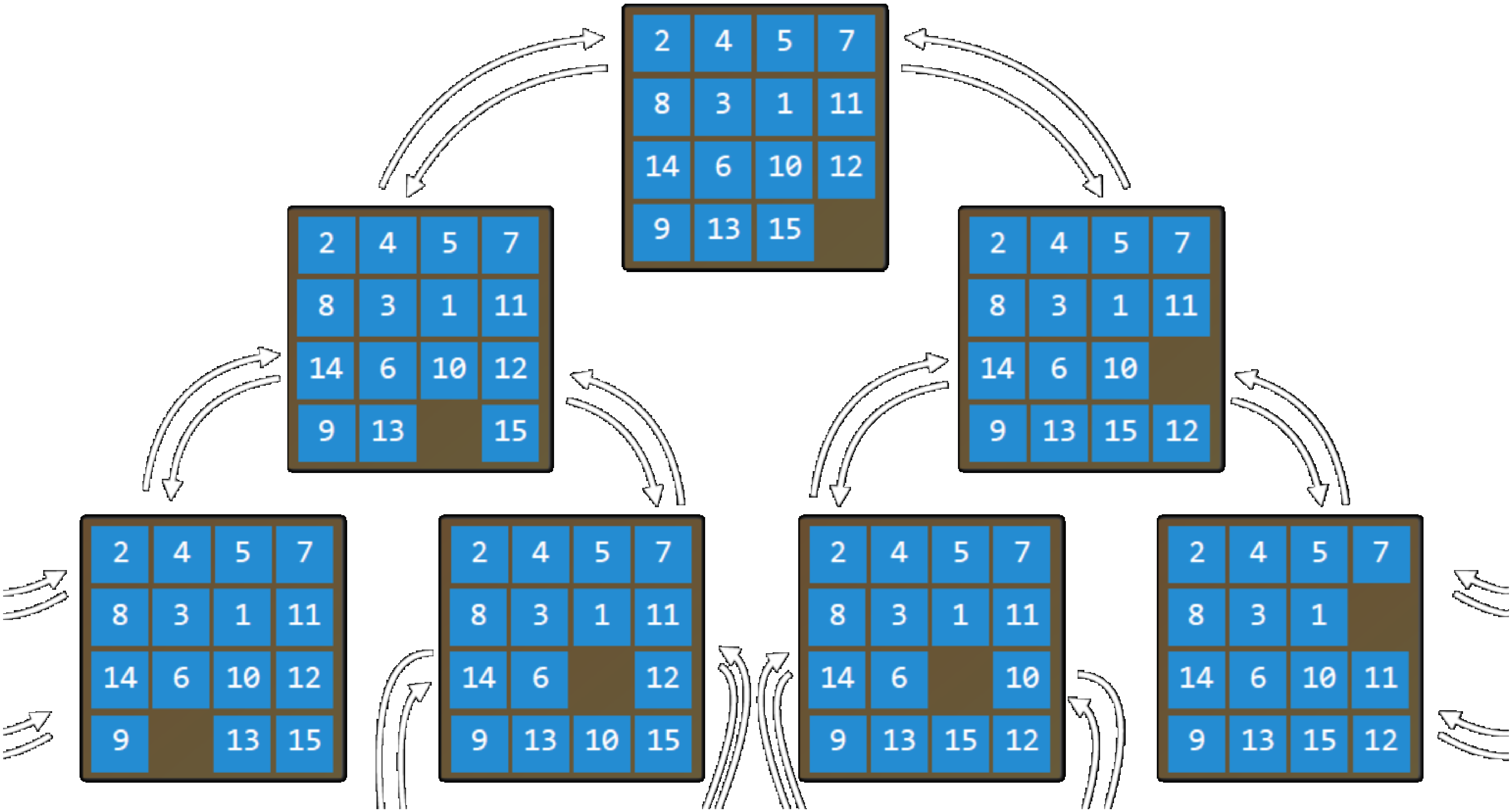
- Touring Agent Problem (Map Search / Navigation)

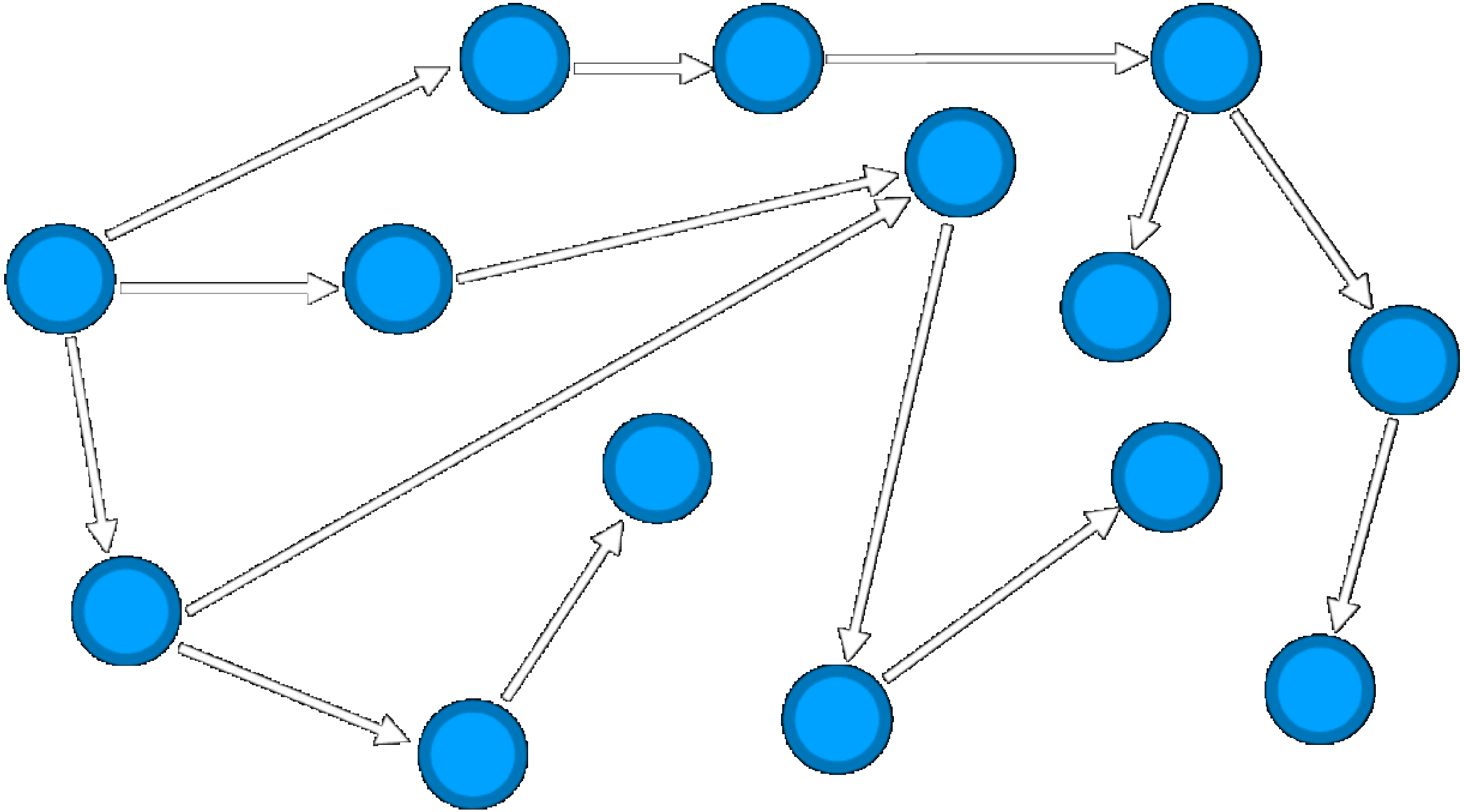


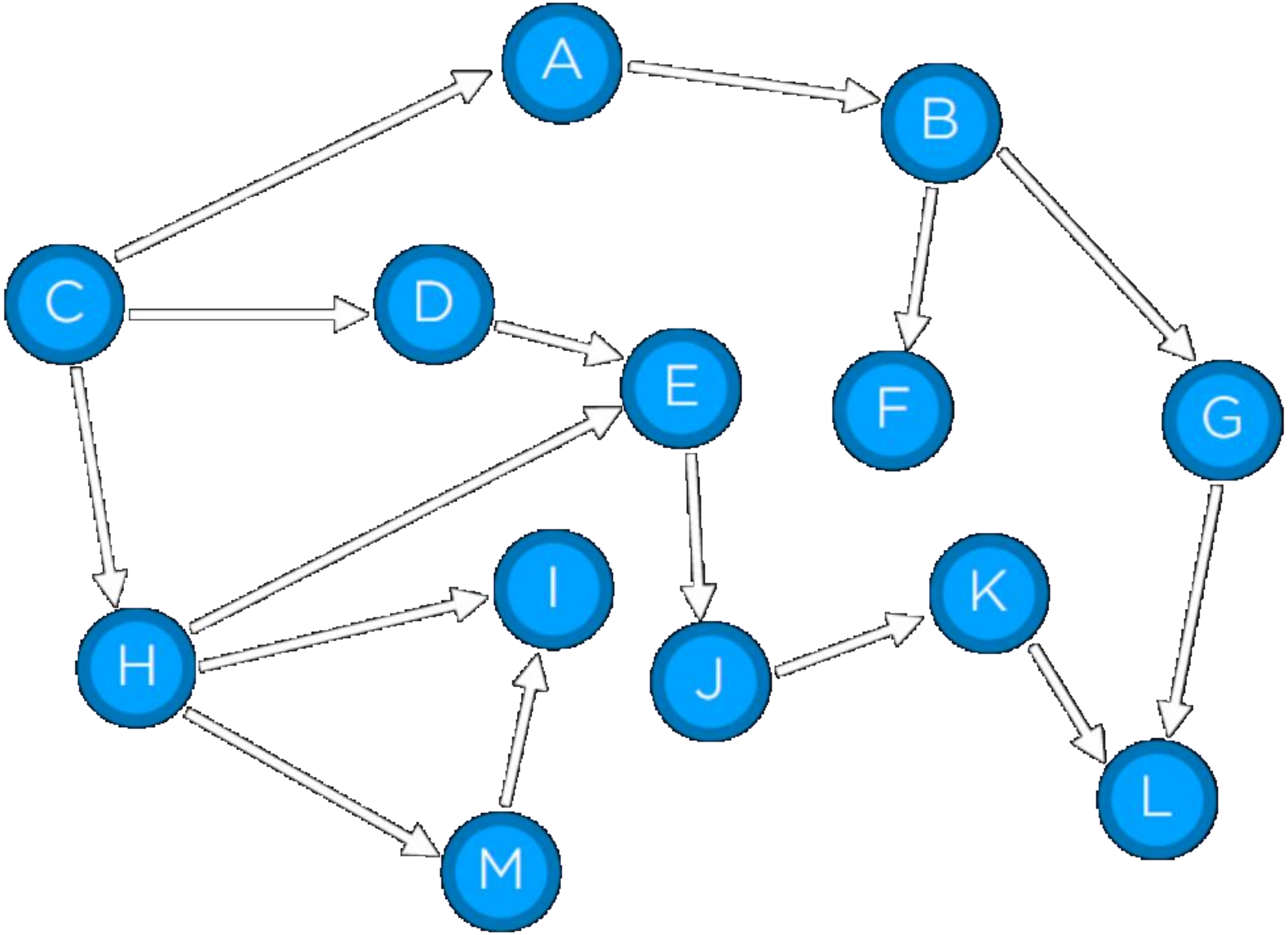
Classical AI Search Problem

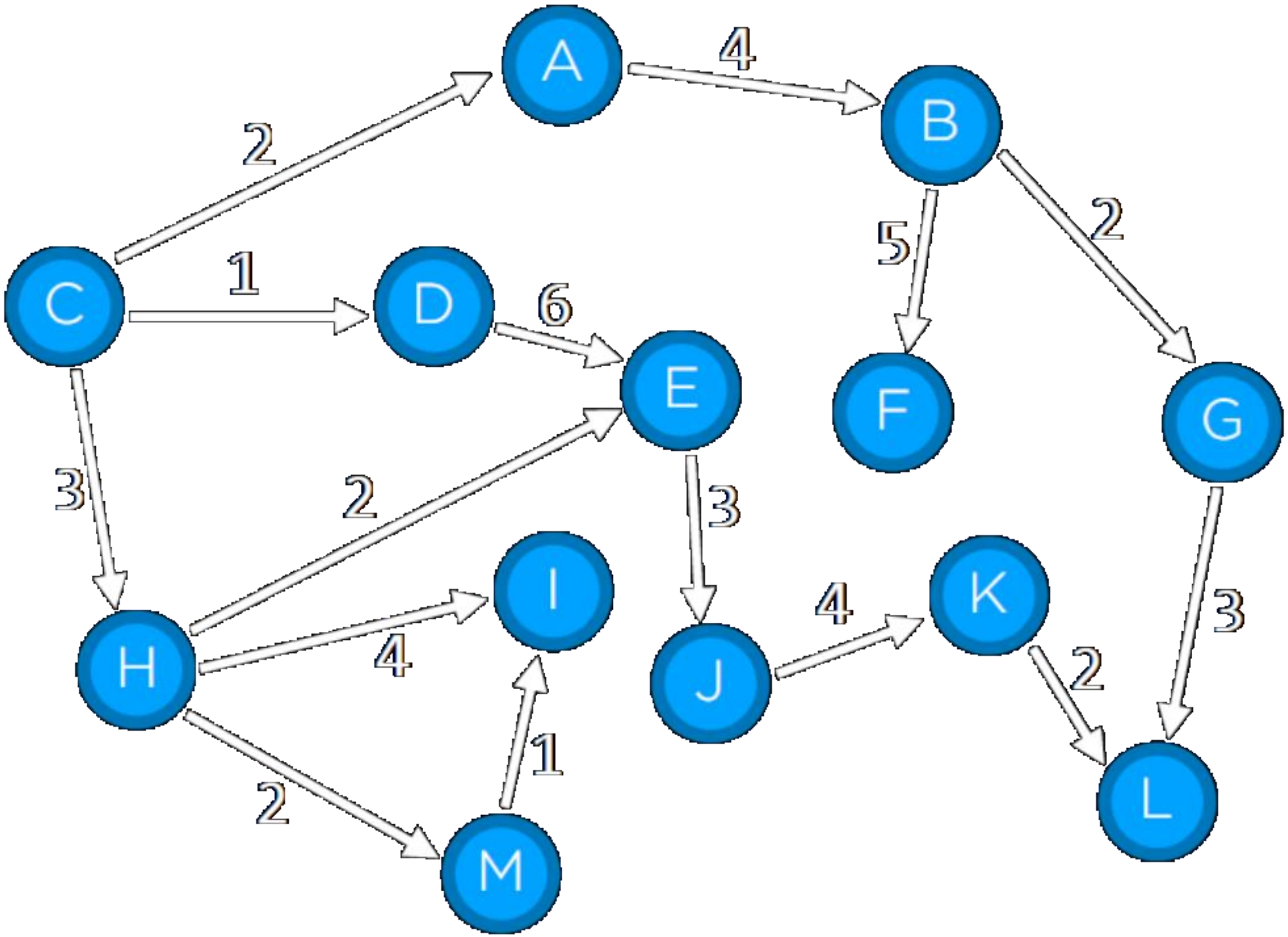
- 8-Puzzle




2 4 5 7
8 3 1 11
14 6 10 12
9 13 15
2 4 5 7
8 3 1 11
14 6 10 12
9 13 15
2 4 5 7
8 3 1 11
14 6 10
9 13 15 12
2 4 5 7
8 3 1 11
14 6 10 12
9 13 15
2 4 5 7
8 3 1 11
14 6 12
9 13 10 15
2 4 5 7
8 3 1 11
14 6 10
9 13 15 12
2 4 5 7
8 3 1
14 6 10 11
9 13 15 12




A
B
C
D
E
F
G
H
I
J
K
L
M

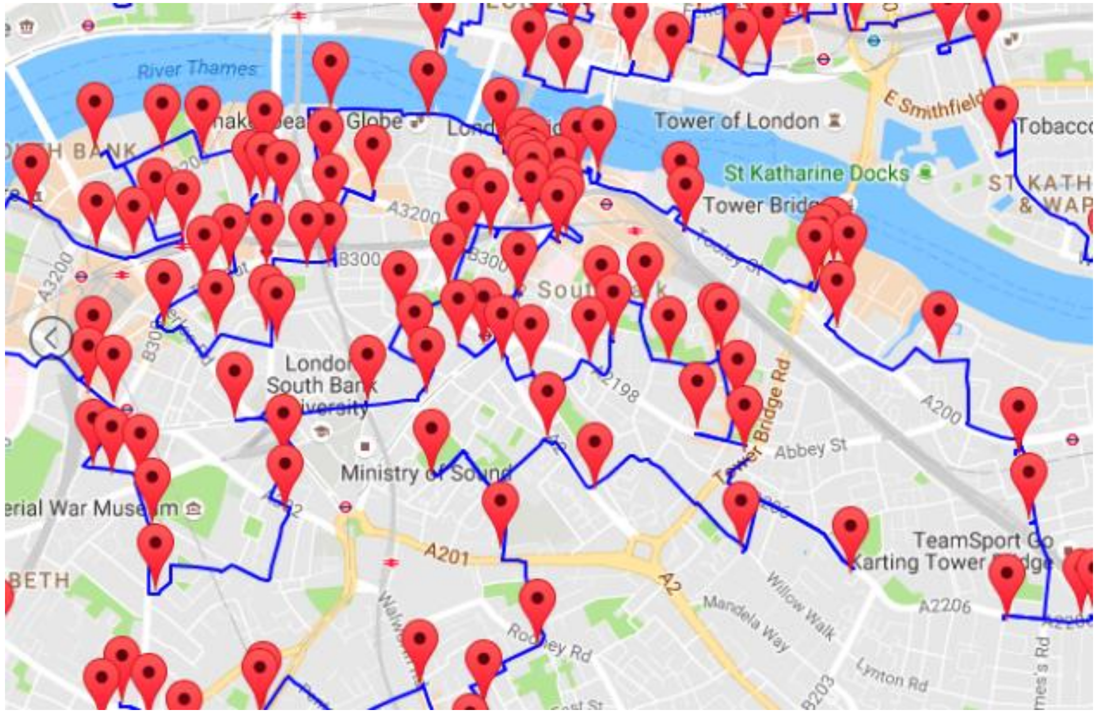

A
B
C
D
E
F
G
H
I
J
K
L
M
4
2
1
6
5
2
3
2
3
4
4
3
2
2
1

Classical AI Search Problem

- **Travelling Salesperson Problem (TSP)**

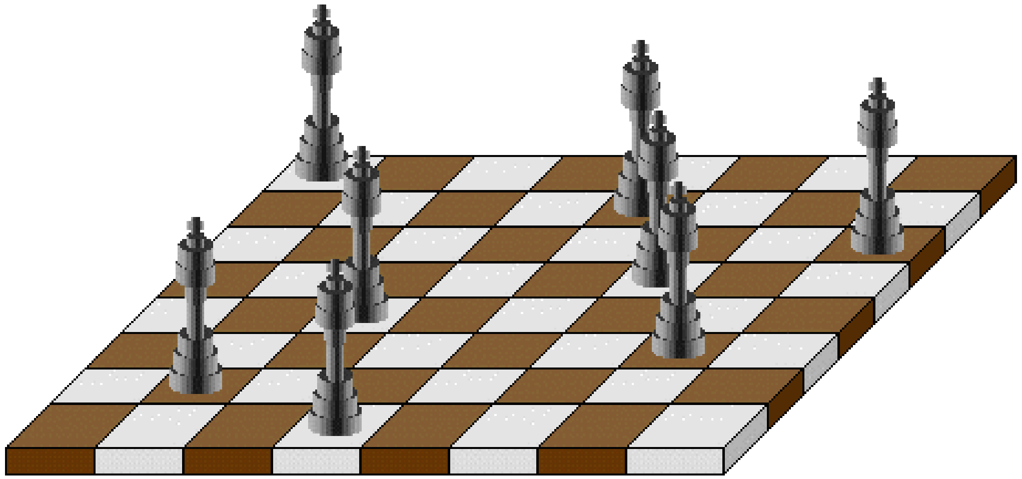
  - Suppose a salesman has five cities to visit and ten must return home.

  - The goal of the problem is to find the shortest path for salesman to travel, visiting each city, and then returning to the starting city.

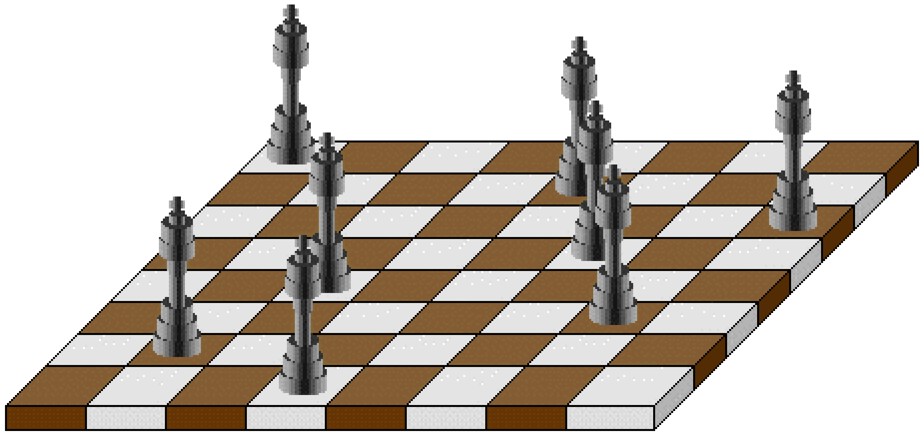


Classical AI Search Problem

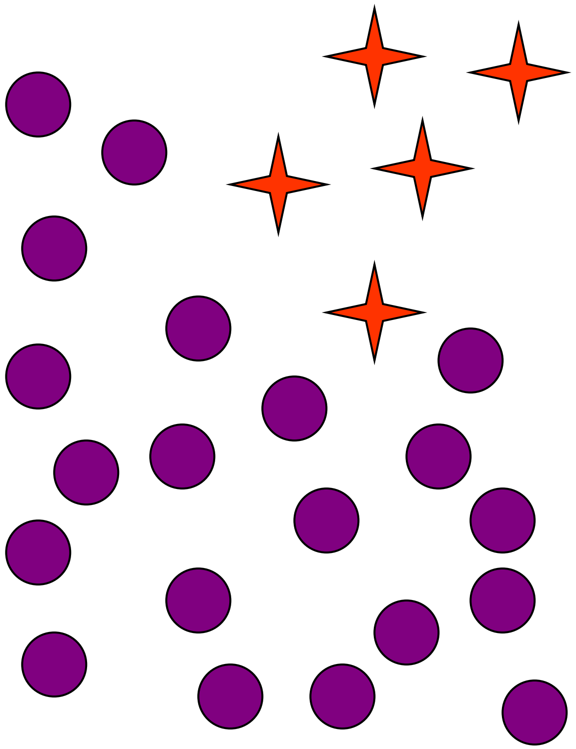
- N-Queens
  - Problem of placing n chess queens on an $n \times n$ chessboard so that no two queens attack each other
  - A solution requires that no two queens share the same row, column, or diagonal

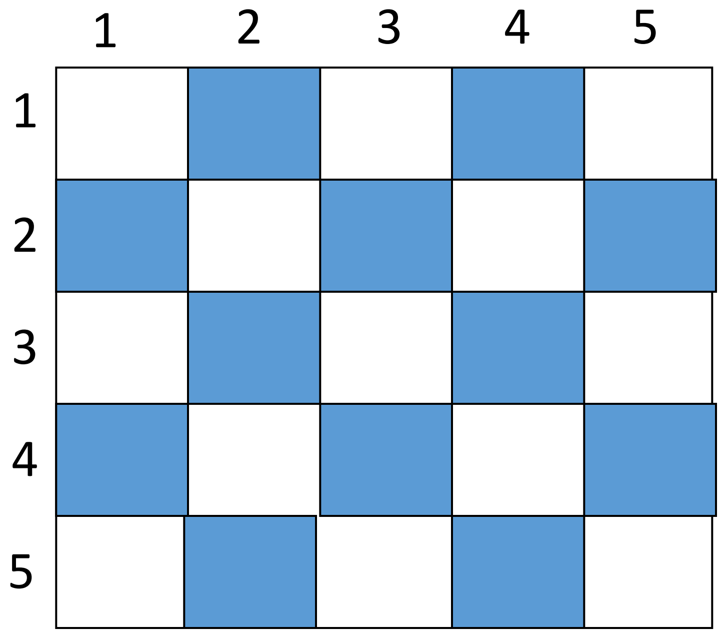


Classic AI Search Problems

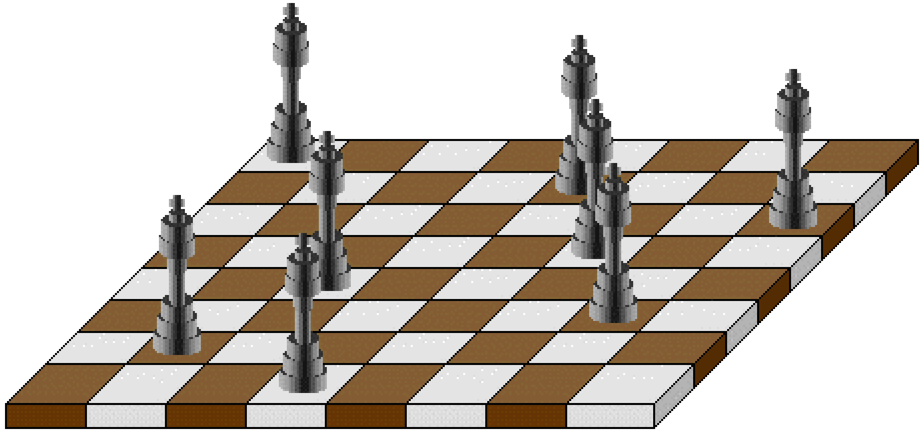


- 5-Queens:

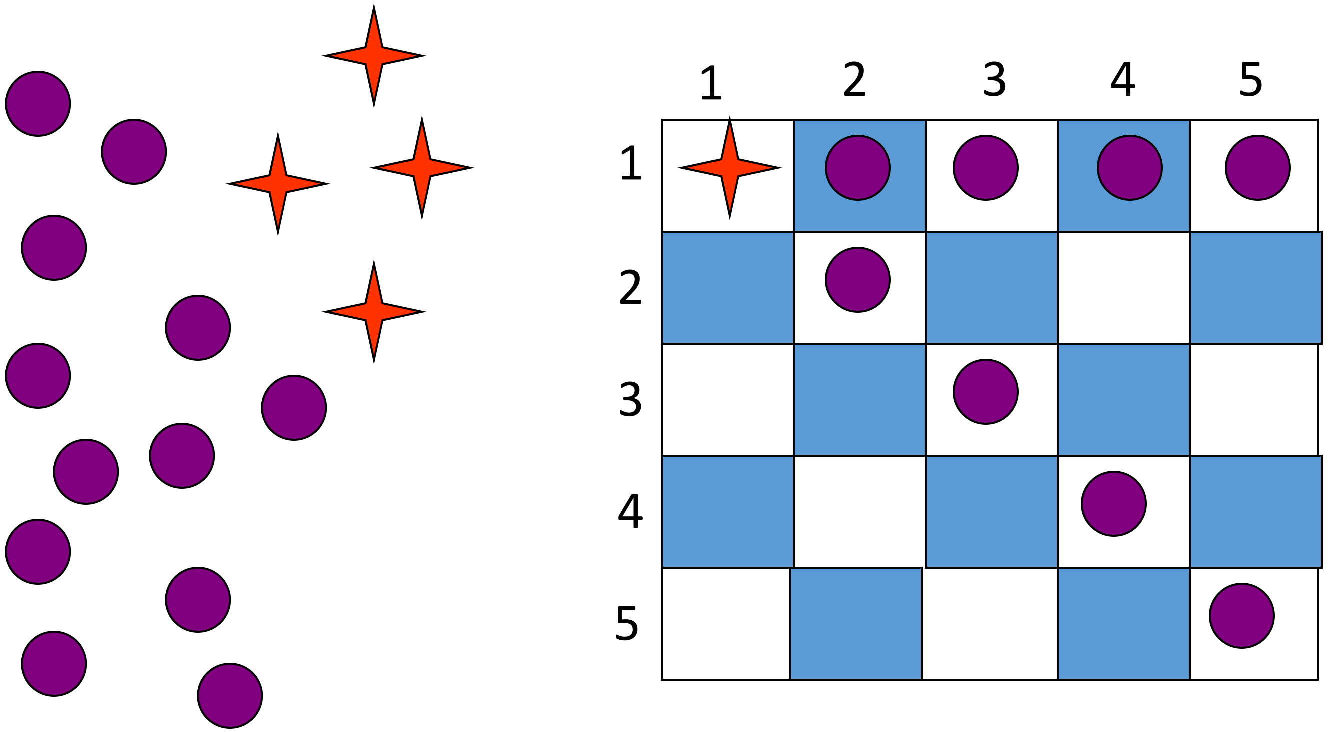




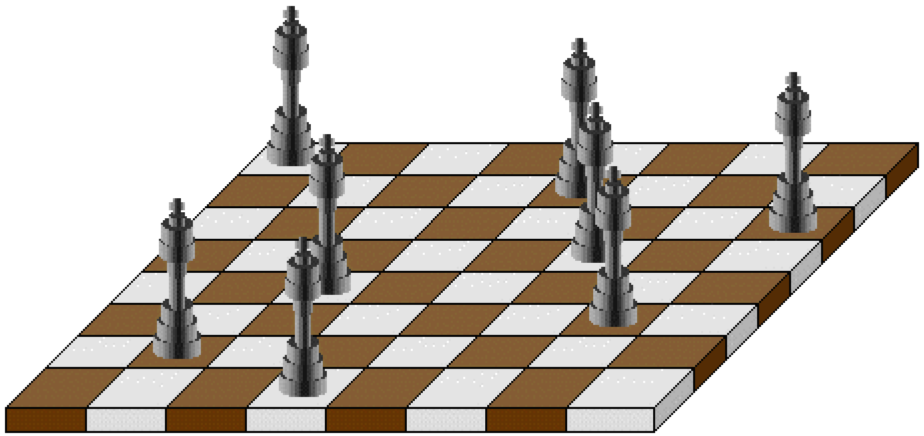
Classic AI Search Problems



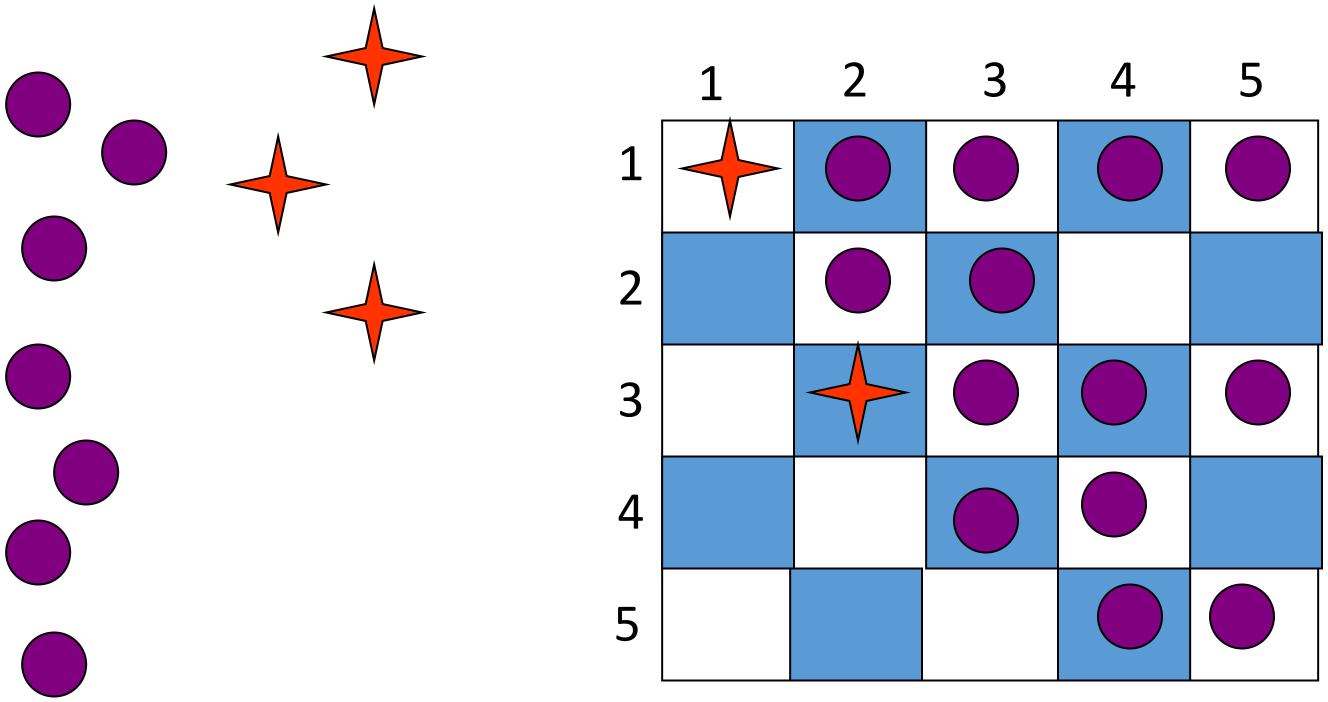
- 5-Queens:



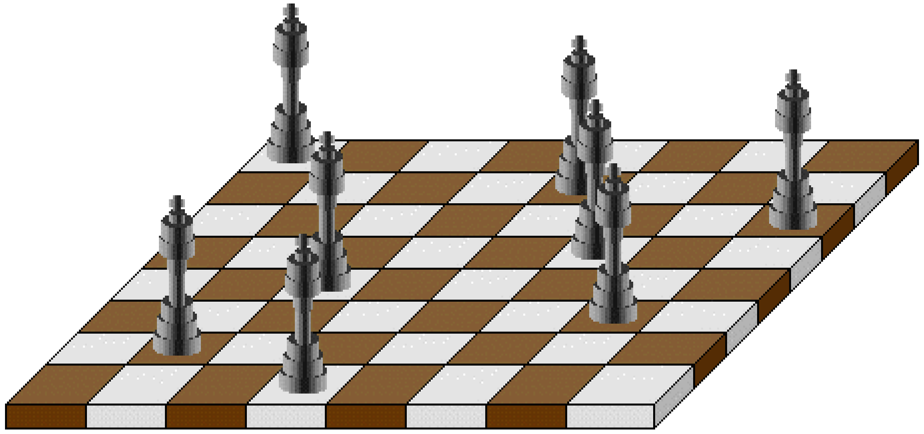
Classic AI Search Problems



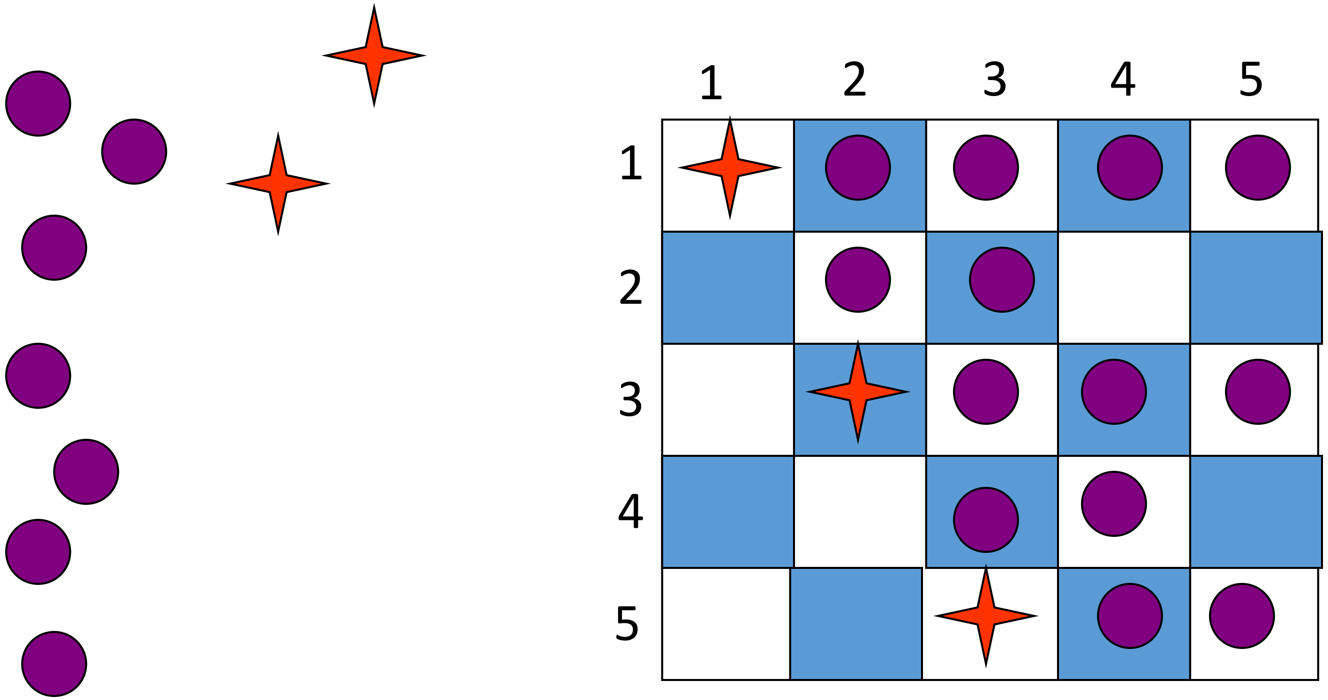
- 5-Queens:



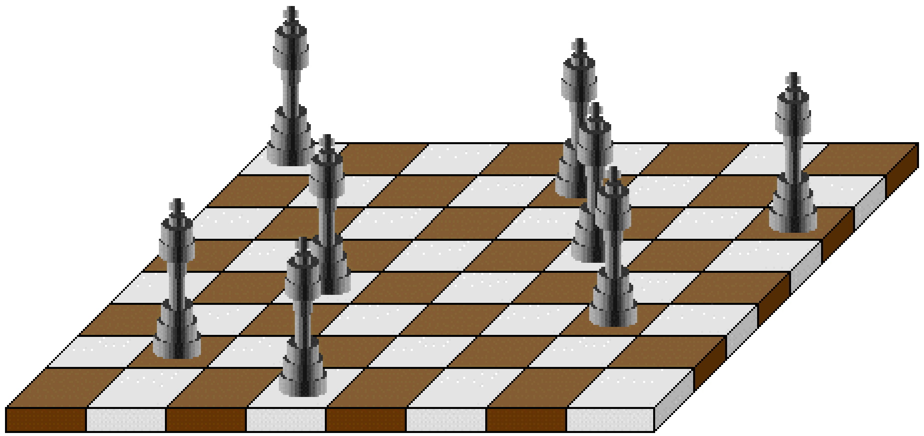
Classic AI Search Problems



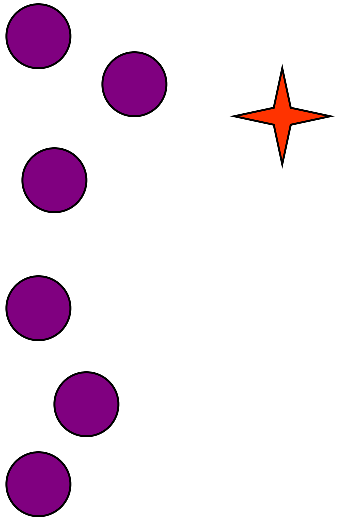
- 5-Queens:

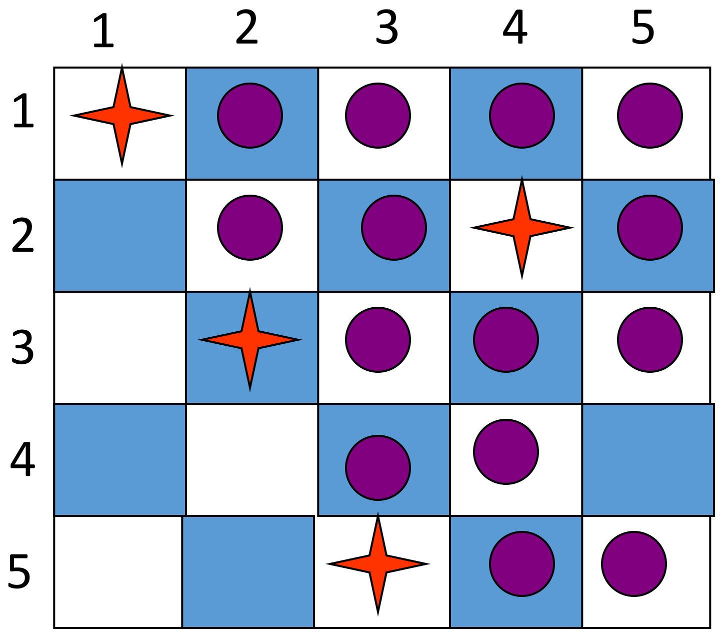


Classic AI Search Problems

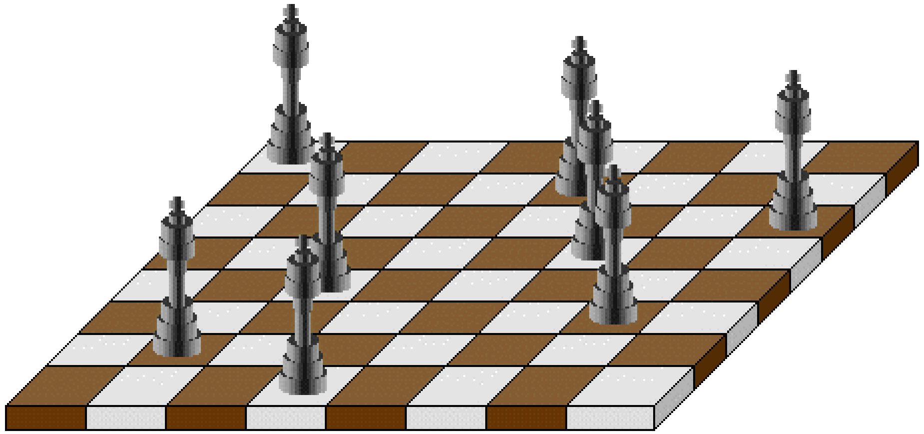


- 5-Queens:

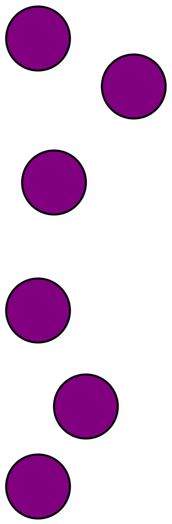




Classic AI Search Problems

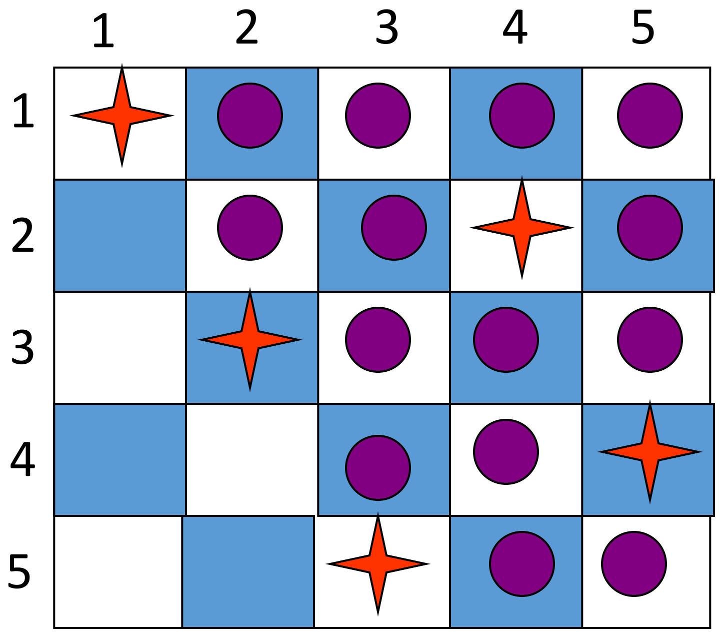


- 5-Queens:



Solution !!

No Queen is under Attack



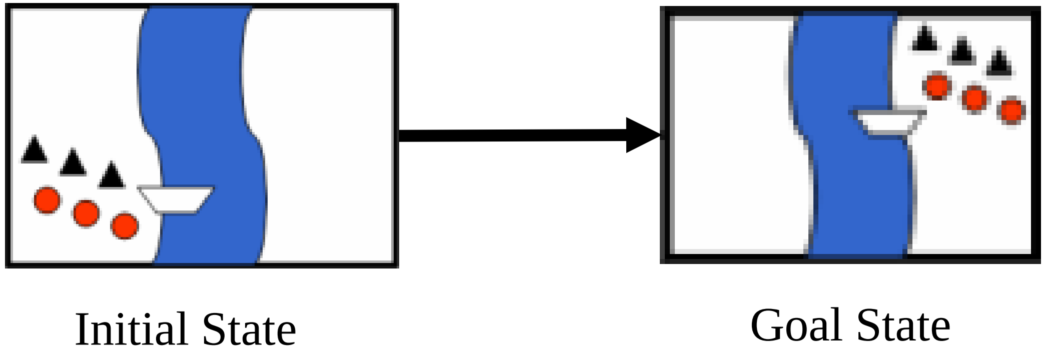
River Crossing Problem

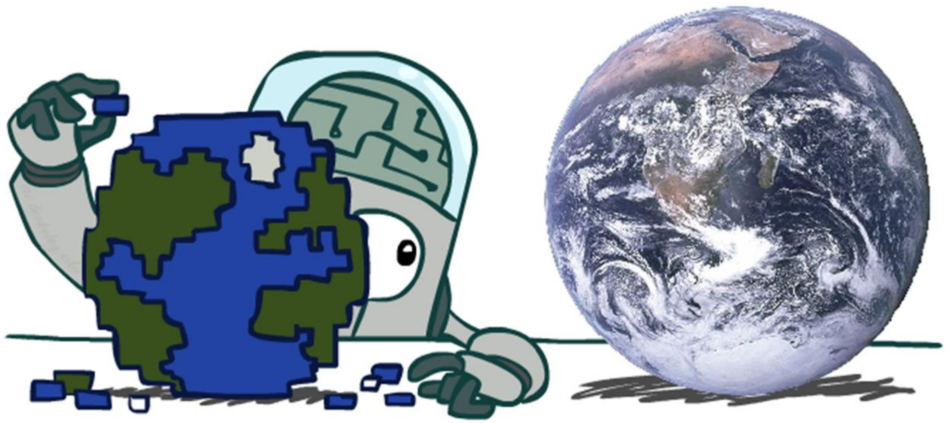
- A farmer wishes to carry a wolf, a duck and corn across a river, from the south to the north shore.
  - The farmer has a small rowing boat.
  - The boat can only carry at most the farmer and one other item.
  - If left unattended the wolf will eat the duck and the duck will eat the corn.

- How can the farmer safely transport the wolf, the duck and the corn to the opposite shore?

Missionaries and cannibals

- Three missionaries and three cannibals are on the left bank of a river.
- There is one boat which can hold one or two people.
- Find a way to get everyone to the right bank, without ever leaving a group of missionaries in one place outnumbered by cannibals in that place.





Problem Solving by Searching

Problem Formulation

Problem Formulation

A **Problem Space** consists of

- The current state of the world (**initial state**)
- A description of the actions we can take to transform one state of the world into another (**operators or successor function**).
- A description of the desired state of the world (**goal state**), this could be implicit or explicit.
- A **solution** consists of the goal state, or a path to the goal state.

Problem Formulation

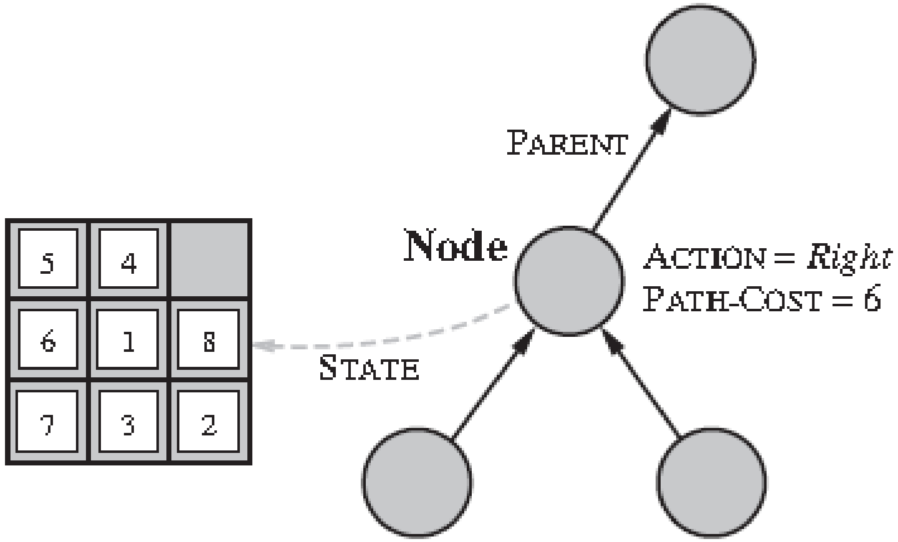
- Initial state
- Actions (operators, successor functions)
- State space
- Path
- Goal test
- Solution
- Path cost
- Total cost

Representation of a Search/State Space

- A convenient way of representing search spaces is as a graph

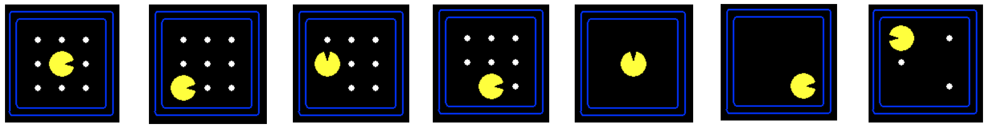
  - **States** are nodes
  - **Actions** are edges
  - **Initial state** is root
  - **Solution** is path from root to goal node

  - Edges sometimes have associated costs
  - States resulting from operator are children

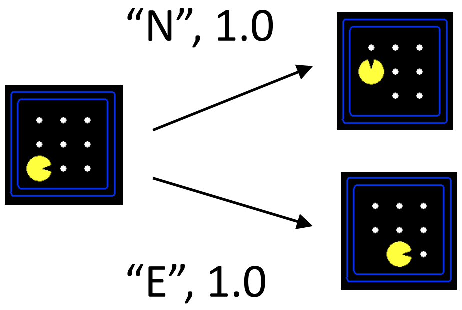


State Space Representation

- A search problem consists of:
  - A state space



  - A successor function (with actions, costs)



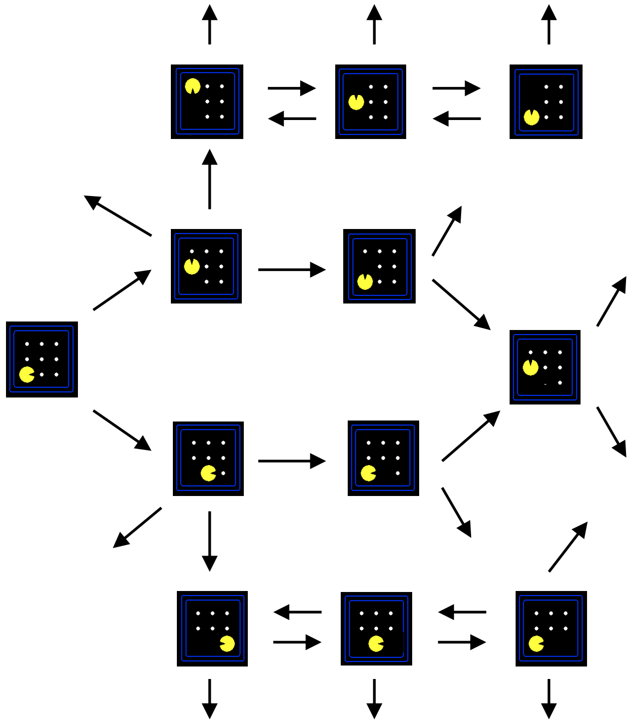
  - A start state and a goal test
- A solution is a sequence of actions (a plan) which transforms the start state to a goal state

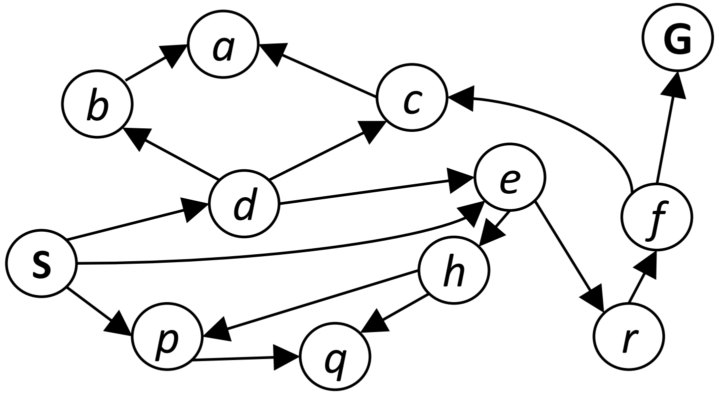
State Space Graphs

- State space graph: A mathematical representation of a search problem
  - Nodes are (abstracted) world configurations
  - Arcs represent successors (action results)
  - The goal test is a set of goal nodes (maybe only one)

- In a state space graph, each state occurs only once!

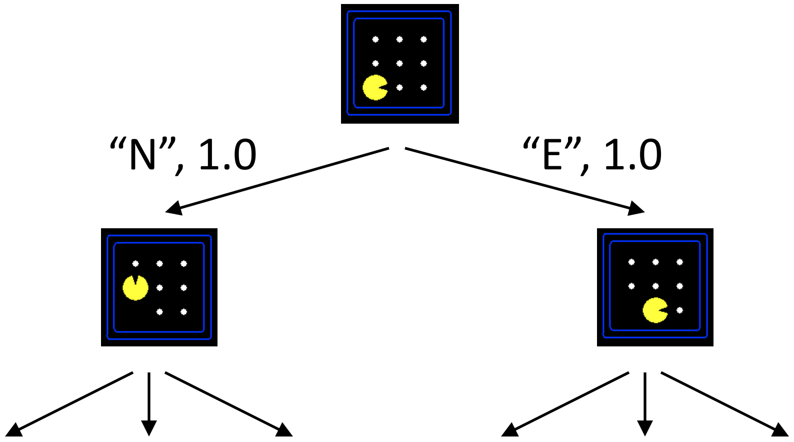
- We can rarely build this full graph in memory (it’s too big), but it’s a useful idea





Tiny search graph for a tiny search problem

Search Trees



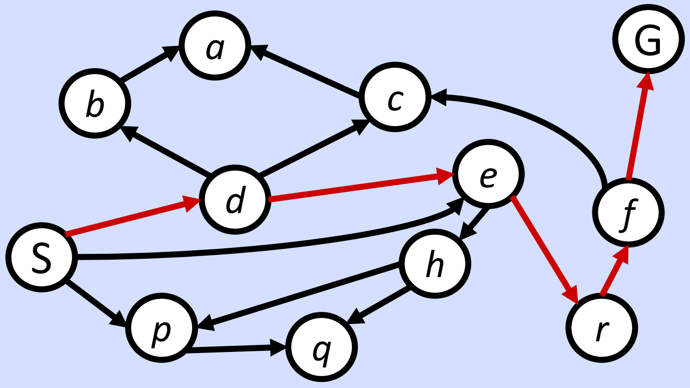
This is now / start

Possible futures

- A search tree:
  - A "what if" tree of plans and their outcomes
  - The start state is the root node
  - Children correspond to successors
  - Nodes show states, but correspond to PLANS that achieve those states
  - For most problems, we can never actually build the whole tree

State Space Graphs vs. Search Trees

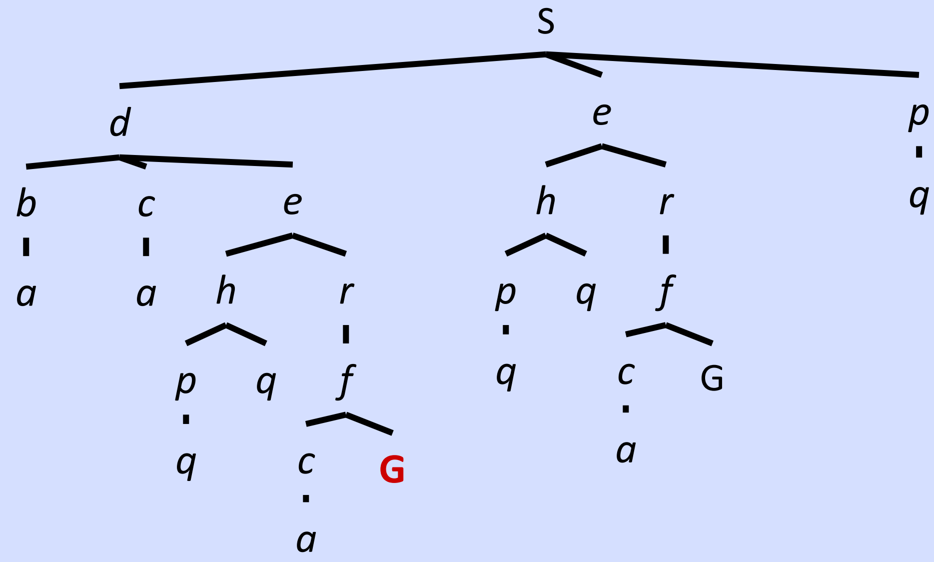
State Space Graph



Each NODE in the search tree is an entire PATH in the state space graph.

We construct both on demand – and we construct as little as possible.

Search Tree

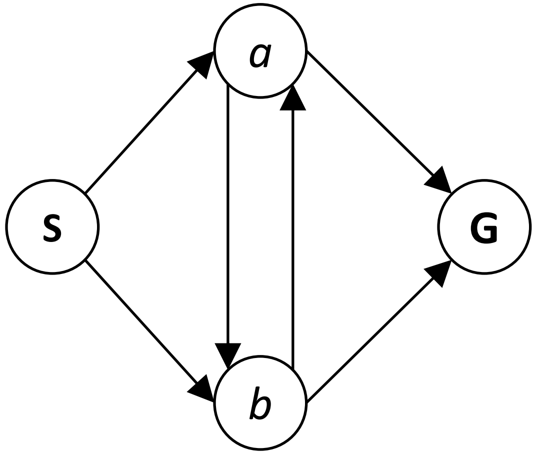


Approach

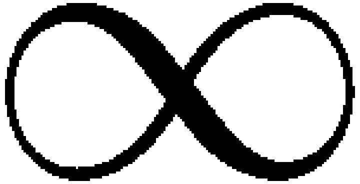
- Start with a **frontier** that contains the initial state.
- Repeat:
  - If the frontier is empty, then no solution.
  - Remove a node from the frontier.
  - If node contains goal state, return the solution.
  - **Expand** node, add resulting nodes to the frontier.

Quiz: State Space Graphs vs. Search Trees

Consider this 4-state graph:



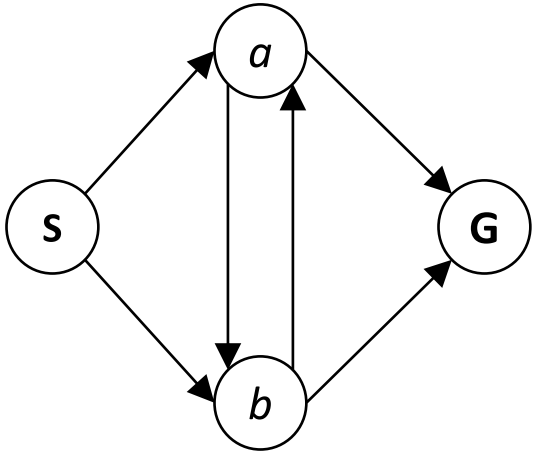
How big is its search tree (from S)?



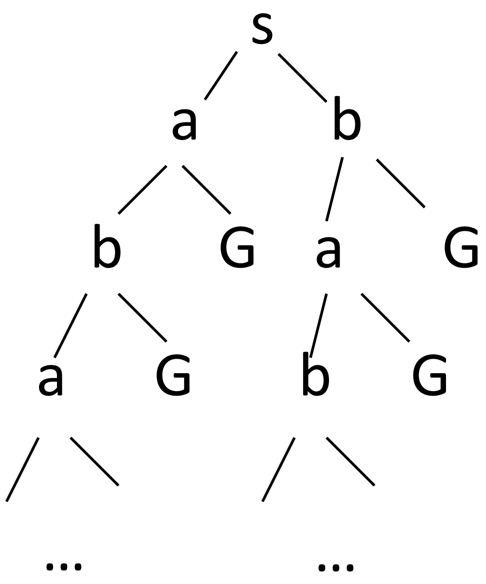
Important: Lots of repeated structure in the search tree!

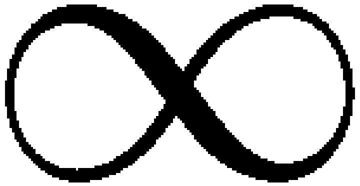
Quiz: State Space Graphs vs. Search Trees

Consider this 4-state graph:



How big is its search tree (from S)?





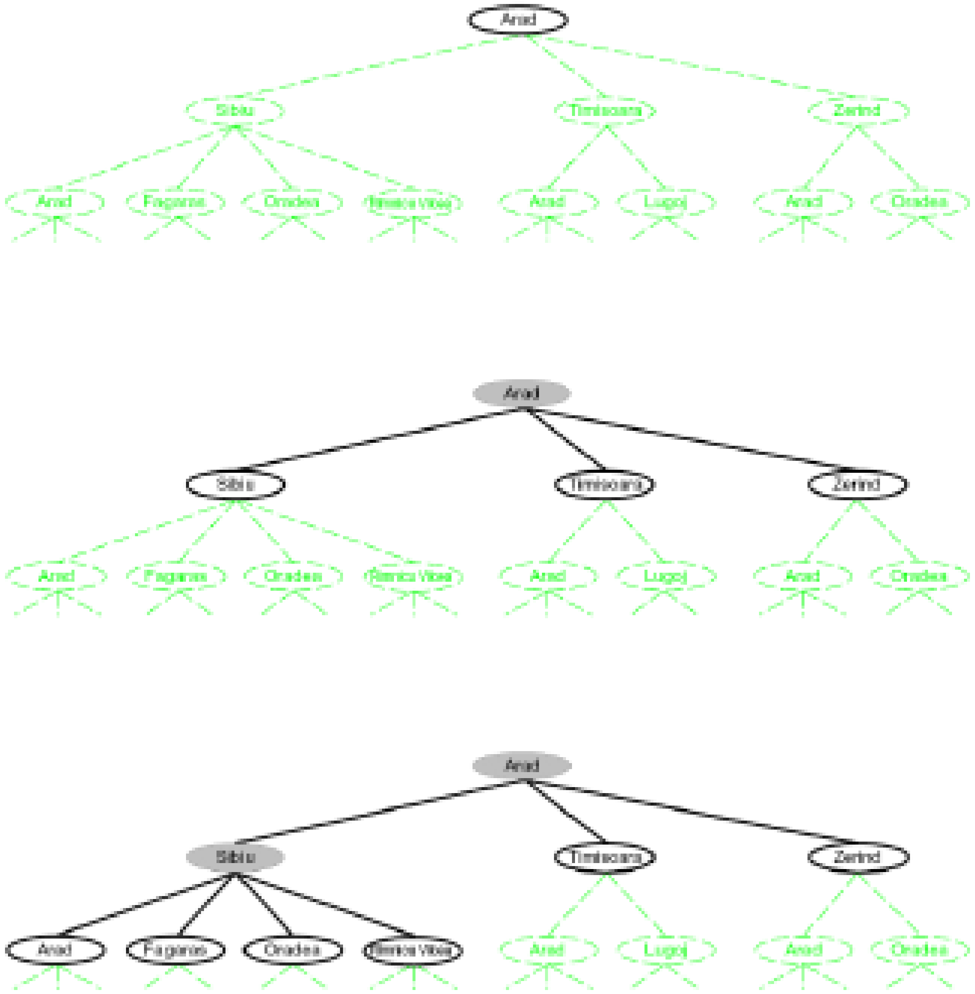
Important: Lots of repeated structure in the search tree!

Revised Approach

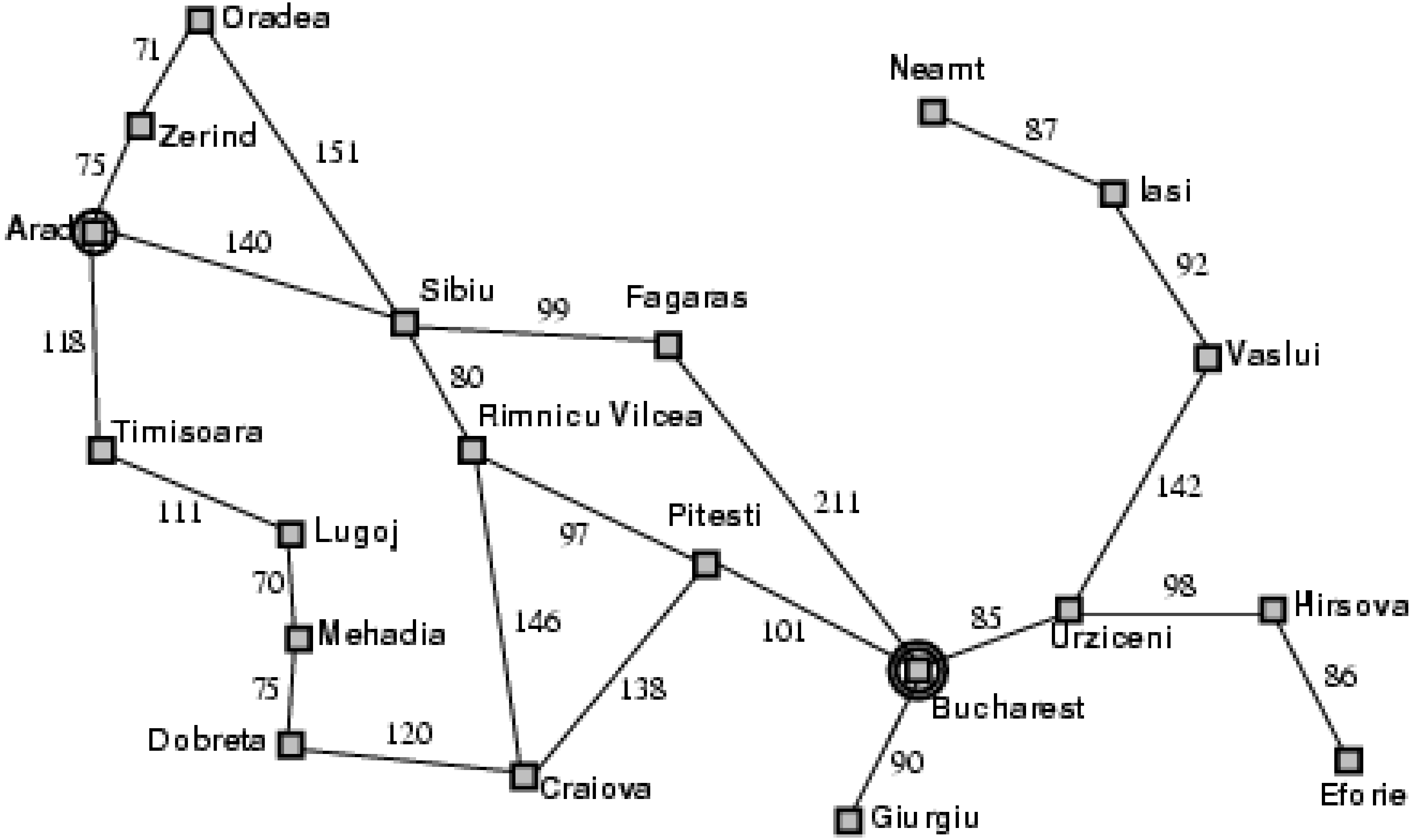
- Start with a **frontier** that contains the initial state.
- Start with an empty **explored set**.
- Repeat:
  - If the frontier is empty, then no solution.
  - Remove a node from the frontier.
  - If node contains goal state, return the solution.
  - Add the node to the explored set.
  - **Expand** node, add resulting nodes to the frontier if they aren't already in the frontier or the explored set.

Searching with a Search Tree

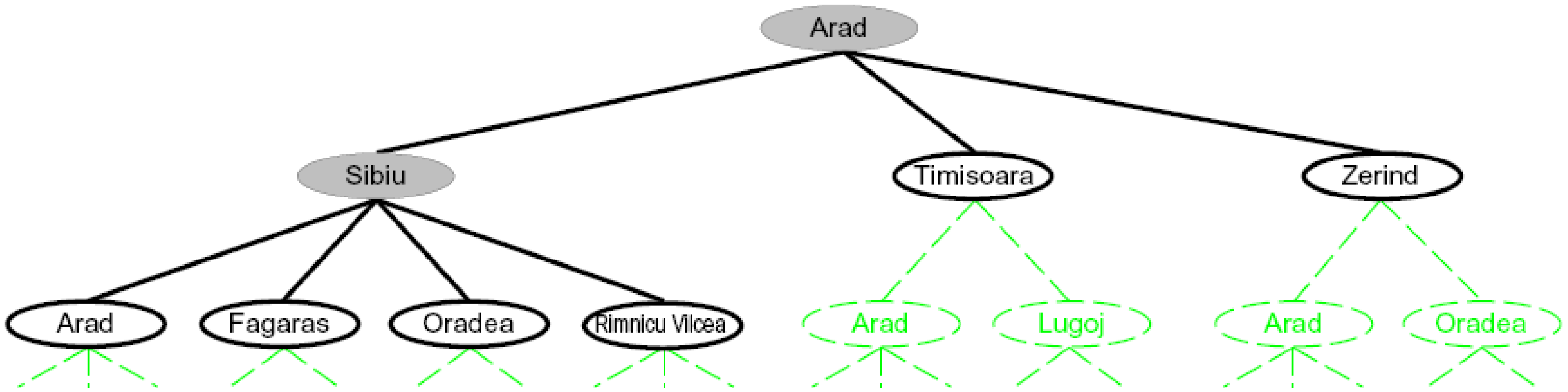
- Search:
- (1) Expand out potential plans (tree nodes)
- (2) Maintain a fringe of partial plans under consideration.
- (3) Try to expand as few tree nodes as possible



Example: Romania



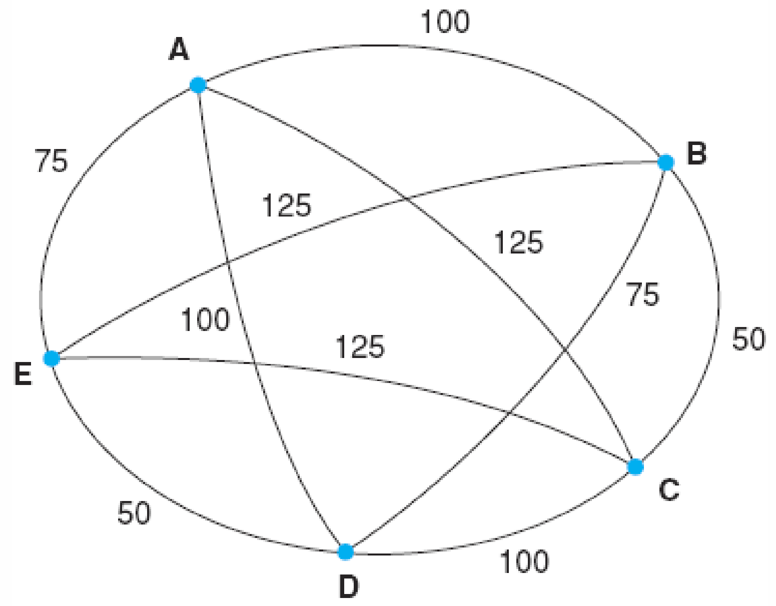
Searching with a Search Tree



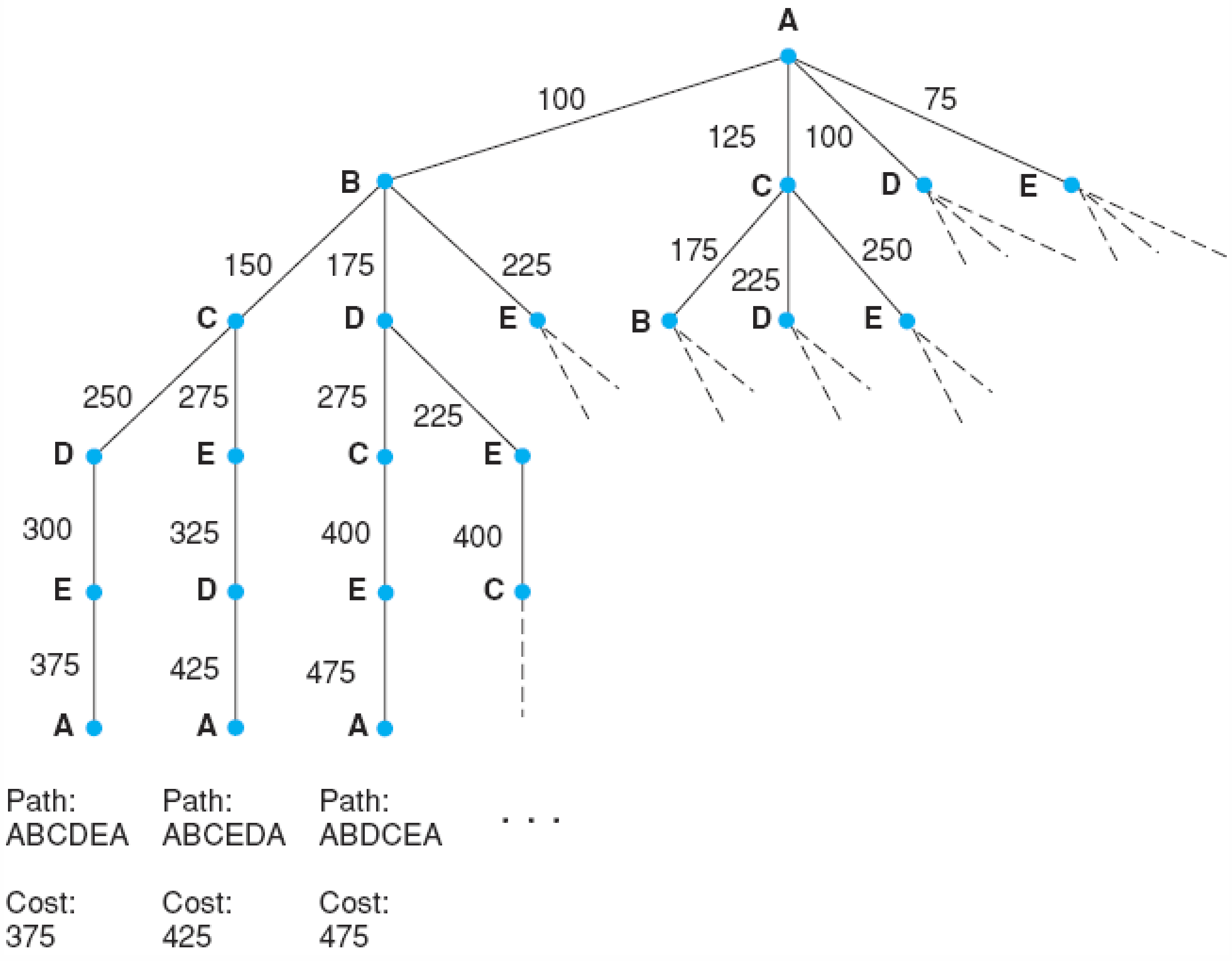
- Search:
  - Expand out potential plans (tree nodes)
  - Maintain a fringe of partial plans under consideration
  - Try to expand as few tree nodes as possible

Problem Formulation: TSP

- **Travelling Salesperson Problem (TSP)**
  - Suppose a salesman has five cities to visit and then must return home.
  - The goal of the problem is to find the shortest path for salesman to travel, visiting each city, and then returning to the starting city.



Search for the travelling salesperson problem. Each arc is marked with the total weight of all paths from the start node (A) to its endpoint.



Problem Formulation: 8-Puzzle Problem

Initial State	Operators	Goal State

2	1	3
4	7	6
5	8	

Slide blank square left.
Slide blank square right.
….

1	2	3
4	5	6
7	8	

Problem Formulation: 8-Puzzle Problem

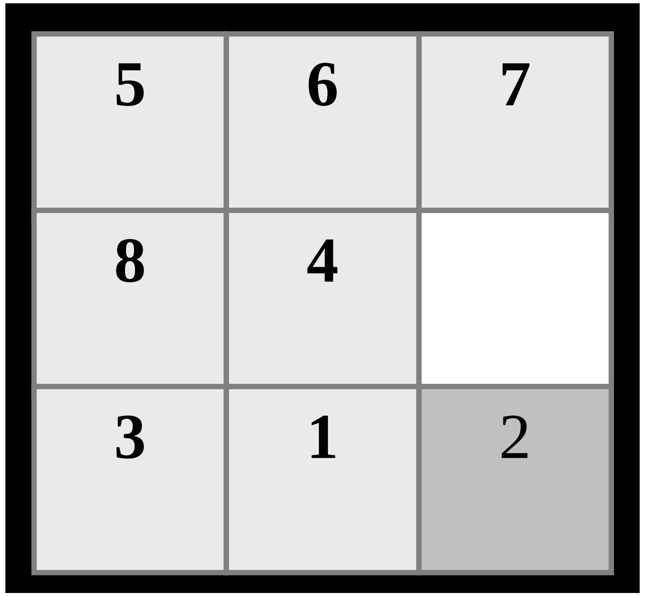
Representing states:

- **For the 8-puzzle**

- 3 by 3 array
    - 5, 6, 7
    - 8, 4, BLANK
    - 3, 1, 2

- A vector of length nine
    - 5, 6, 7, 8, 4, BLANK, 3, 1, 2

- A list of facts
    - Upper_left = 5
    - Upper_middle = 6
    - Upper_right = 7
    - Middle_left = 8

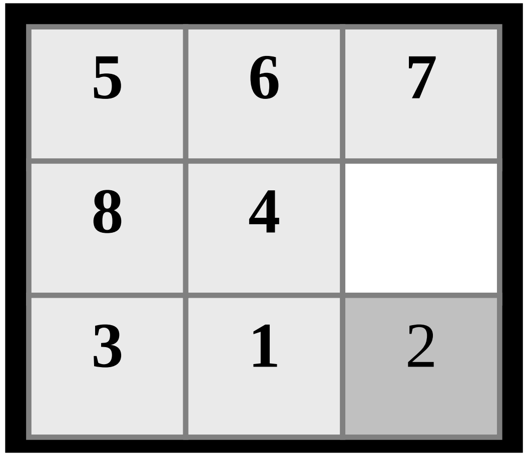


Problem Formulation: **8-Puzzle Problem**

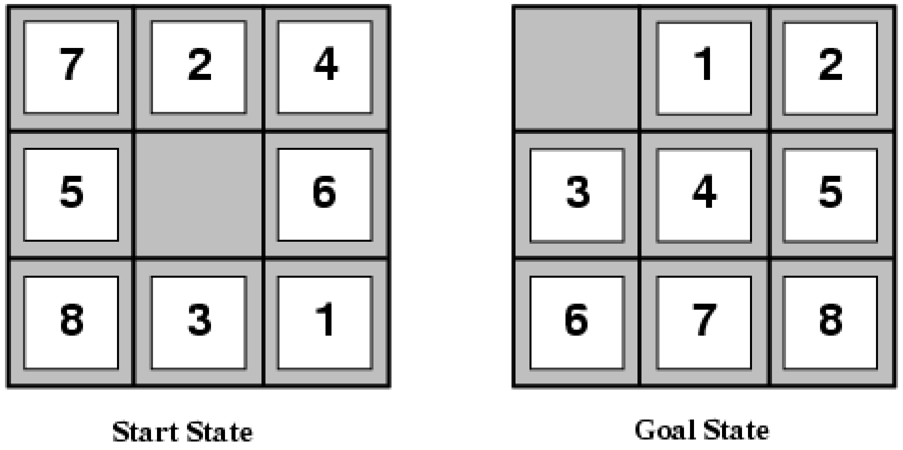
- Specifying operators
  - There are often many ways to specify the operators, some will be much easier to implement...

- **`Move 1 left`**
- **`Move 1 right`**
- **`Move 1 up`**
- **`Move 1 down`**
- **`Move 2 left`**
- **`Move 2 right`**
- **`Move 2 up`**
- **`Move 2 down`**
- **`Move 3 left`**
- **`Move 3 right`**
- **`Move 3 up`**
- **`Move 3 down`**
- **`Move 4 left`**
- …

- **`Move Blank left`**
- **`Move Blank right`**
- **`Move Blank up`**
- **`Move Blank down`**

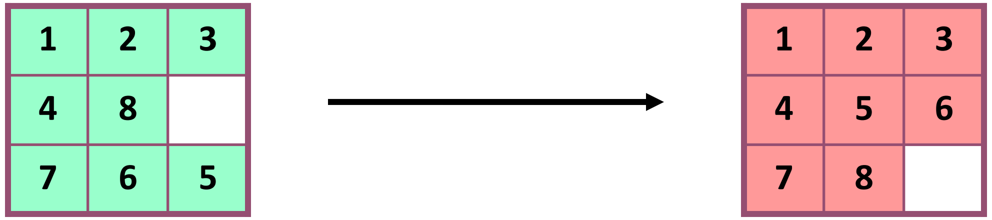


Problem Formulation: **8-Puzzle Problem**



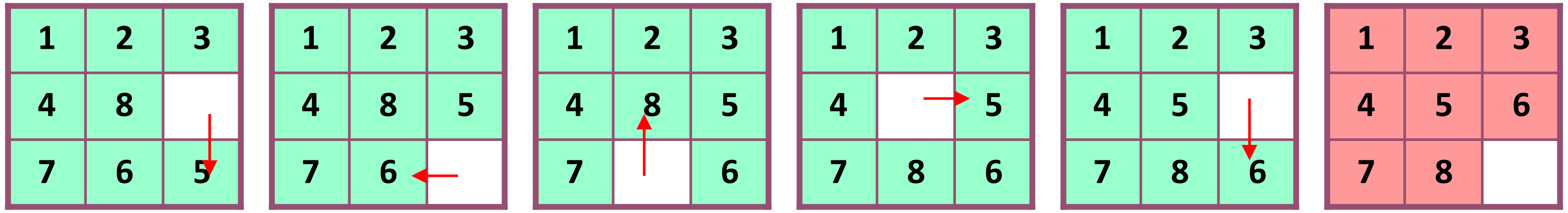
- states locations of tiles
- actions move blank left, right, up, down
- goal test goal state (given)
- path cost 1 per move

Problem Formulation: 8-Puzzle Problem



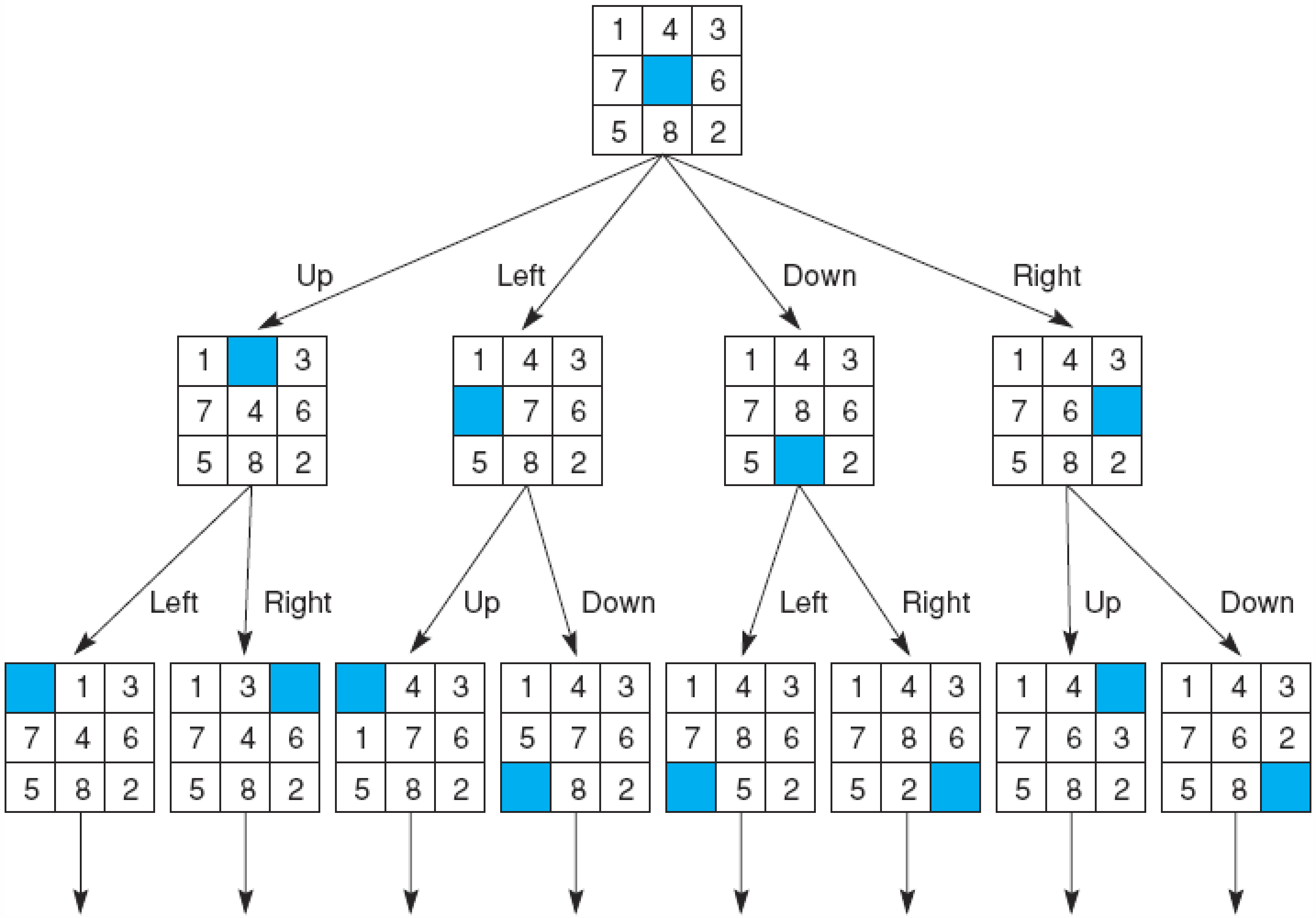
Initial state **Goal state**

- Operators: ***slide blank up, slide blank down, slide blank left, slide blank right***

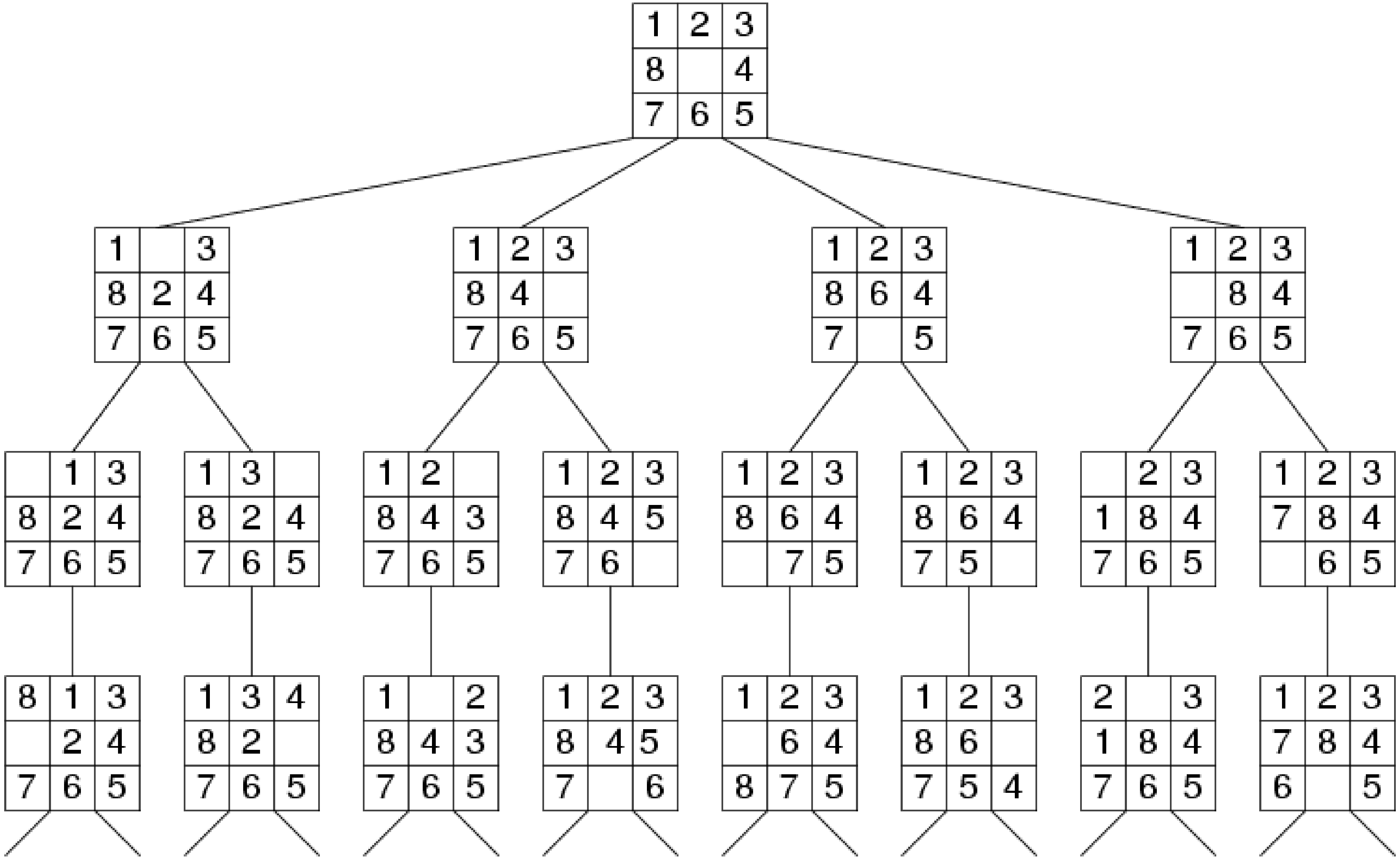


- Solution: ***sb-down, sb-left, sb-up,sb-right, sb-down***
- *Path cost:* ***5*** *steps to reach the goal*

State space of the 8-puzzle generated by "move blank" operations

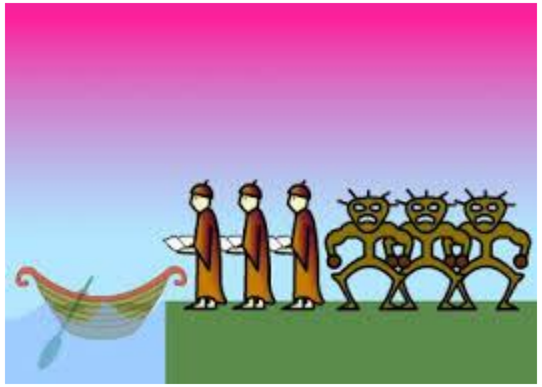


Fragment of 8-Puzzle Problem Space



Problem Formulation: Missionaries and Cannibals

- Three missionaries and three cannibals are on the left bank of a river.
- There is one boat which can hold one or two people.
- Find a way to get everyone to the right bank, without ever leaving a group of missionaries in one place outnumbered by cannibals in that place.



Missionaries and cannibals

- States: three numbers (i, j, k) representing the number of missionaries, cannibals, and boats on the left bank of the river.
- Initial state: (3, 3, 1)
- Operators: take one missionary, one cannibal, two missionaries, two cannibals, one missionary and one cannibal across the river in a given direction
- Goal Test: reached state (0, 0, 0)?
- Path Cost: Number of crossings.

Formalization of the M&C Problem

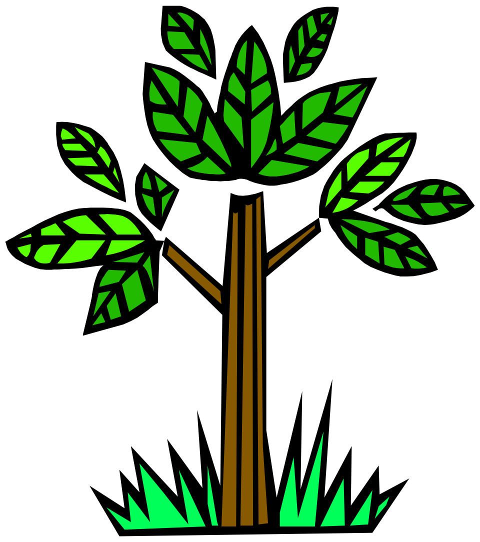
- **State space**: triple (x,y,z), where x,y, and z represent the number of missionaries, cannibals and boats currently on the original bank.
- **Initial State**: (3,3,1)
- **Successor function**: From each state, either bring one missionary, one cannibal, two missionaries, two cannibals, or one of each type to the other bank.
- **Note**: Not all states are attainable (e.g., (0,0,1)), and some are illegal.
- **Goal State**: (0,0,0)
- **Path Costs**: 1 unit per crossing

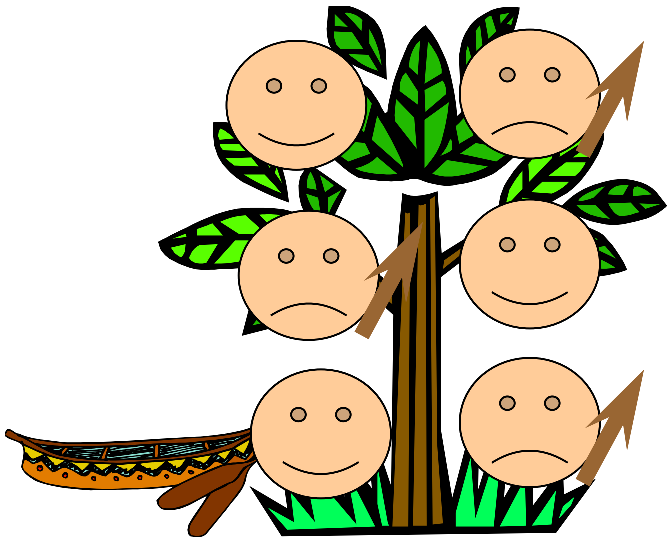
Missionaries and Cannibals

Solution = the sequence of actions within the path :

[(3,3,1)→ (2,2,0)→(3,2,1) →(3,0,0) →(3,1,1) →(1,1,0) →(2,2,1) →(0,2,0) →(0,3,1) →(0,1,0) → (0,2,1) →(0,0,0)]

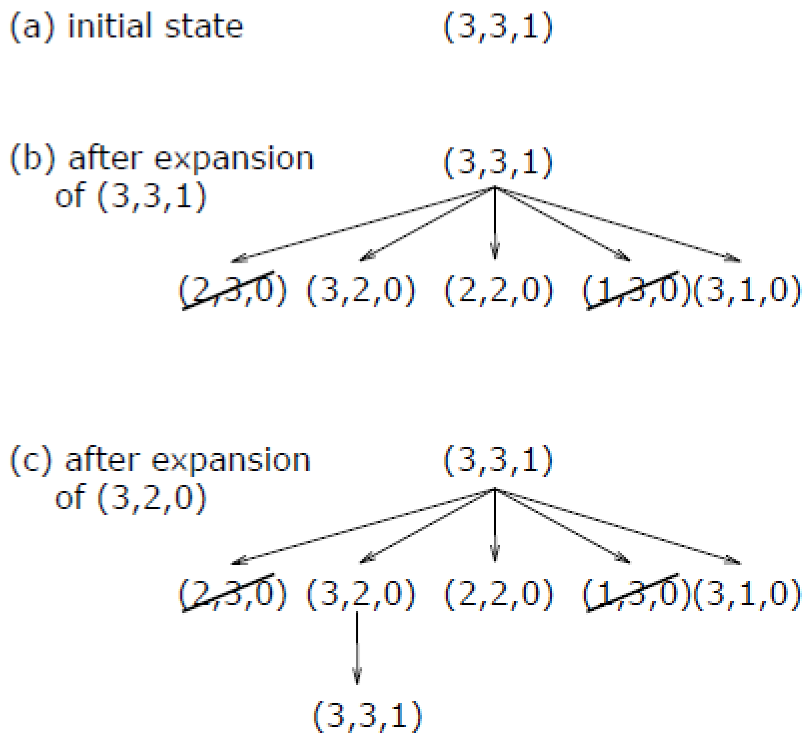
Cost = 11 crossings





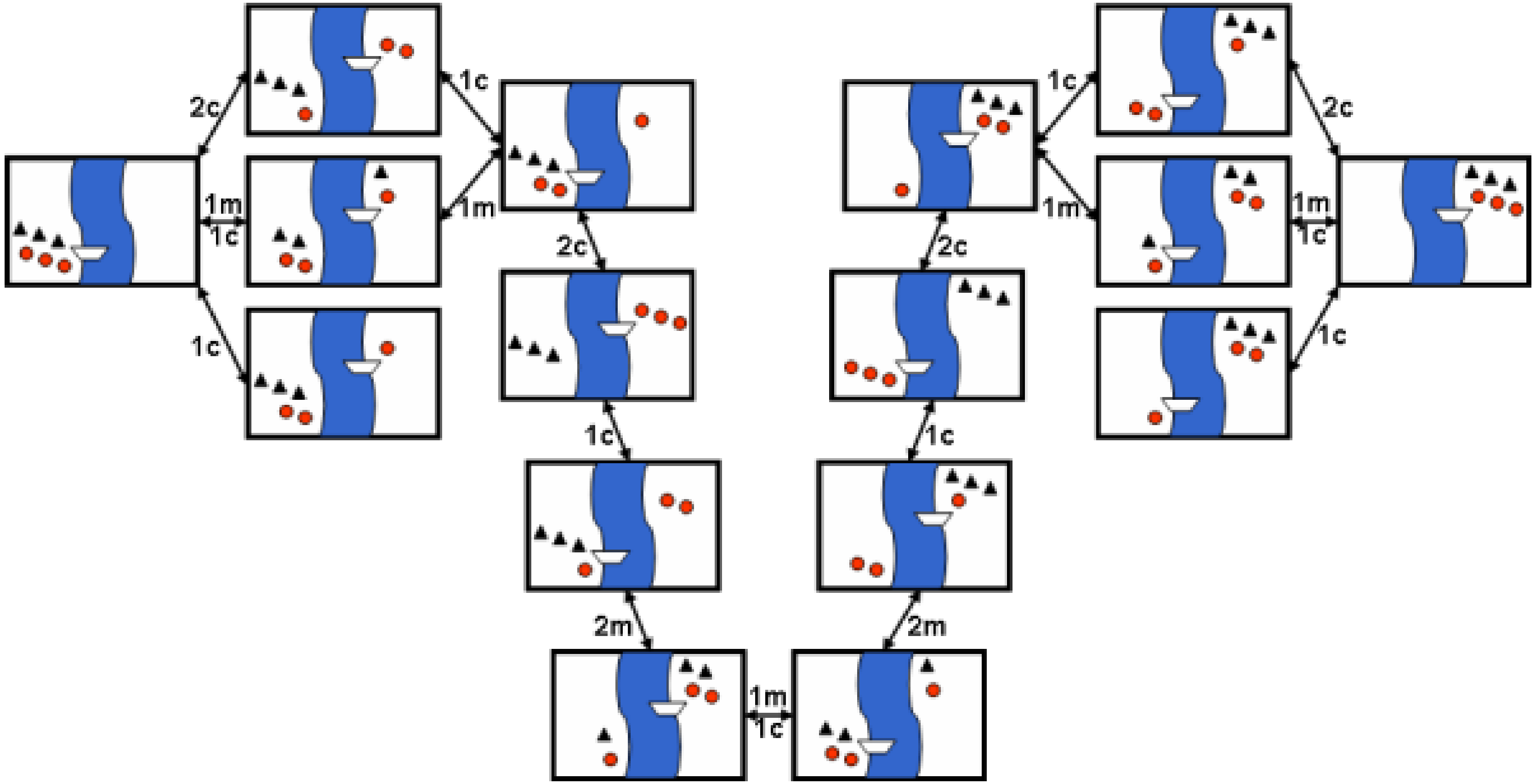
Missionaries and Cannibals

- From the initial state, produce all successive states step by step ⍰ search tree.



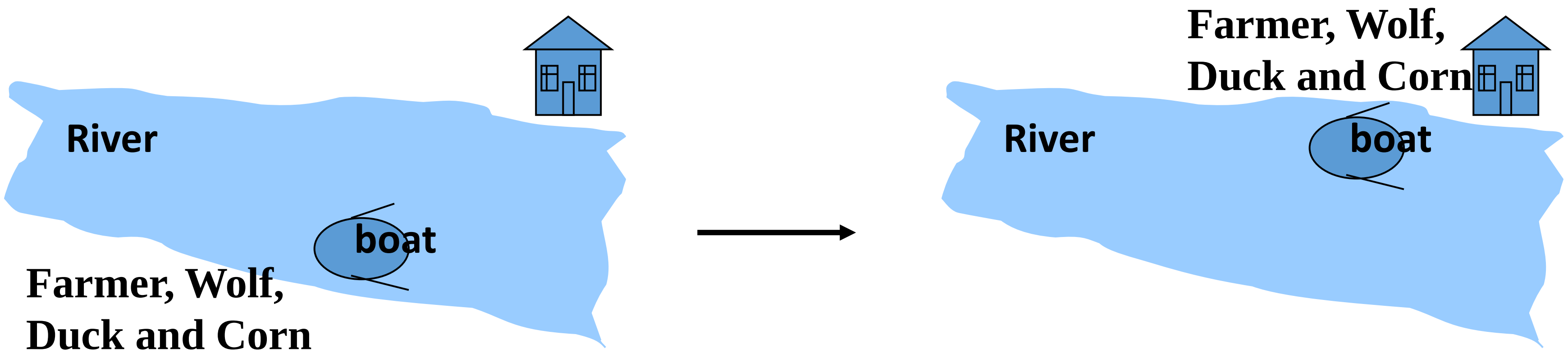
Missionaries and Cannibals

- *Search-space for the Missionaries and Cannibals problem*



The River Problem

- A farmer wishes to carry a wolf, a duck and corn across a river, from the south to the north shore. The farmer has a small rowing boat. The boat can only carry at most the farmer and one other item.
- If left unattended the wolf will eat the duck and the duck will eat the corn.

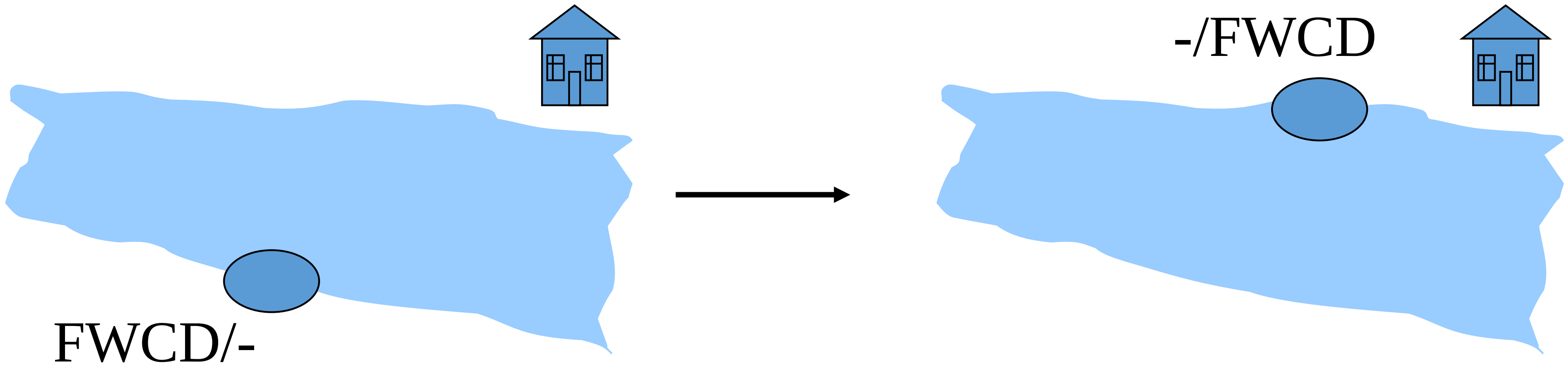


How can the farmer safely transport the wolf, the duck and the corn to the opposite shore?

The River Problem

- The River Problem:

F=Farmer W=Wolf D=Duck C=Corn /=River



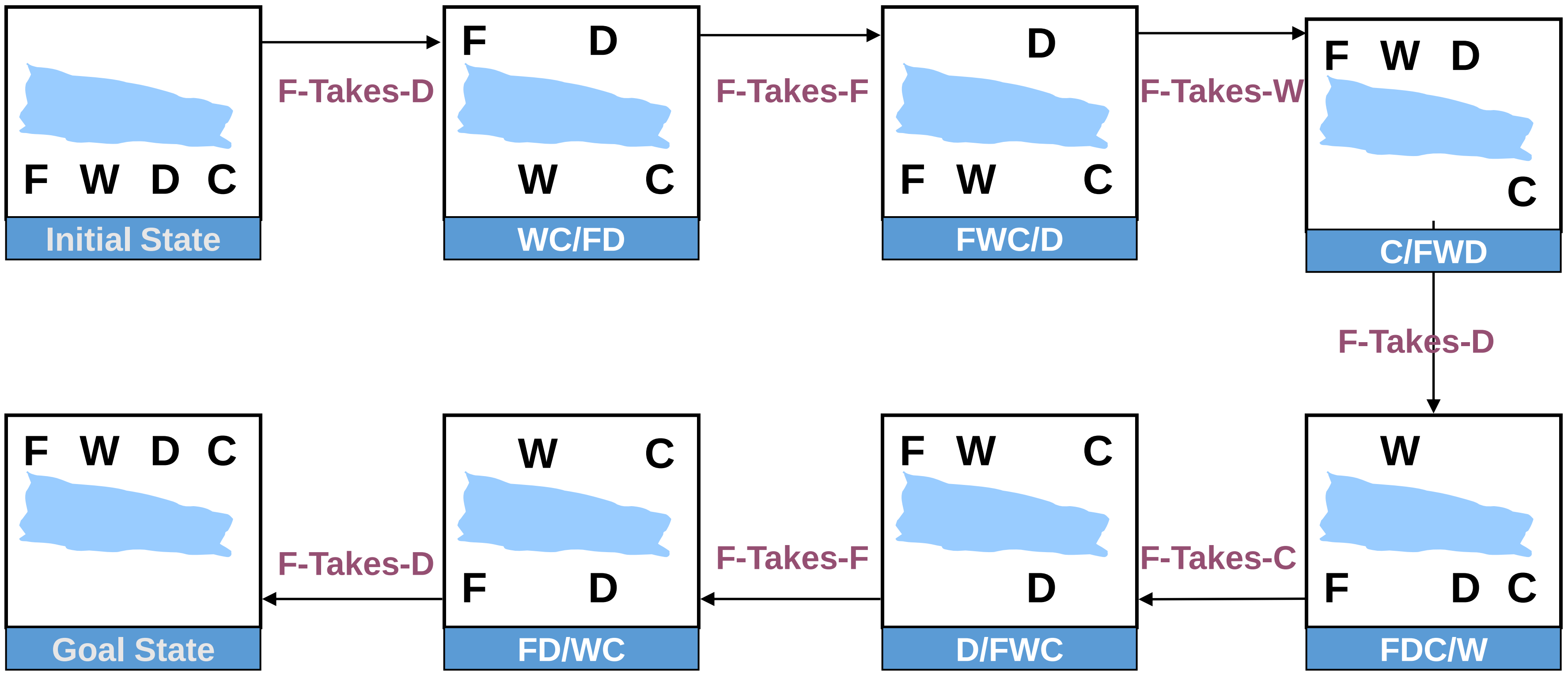
How can the farmer safely transport the wolf, the duck and the corn to the opposite shore?

The River Problem

- Problem formulation:
  - **State representation**: location of farmer and items in both sides of river
    [items in South shore / items in North shore] : (FWDC/-, FD/WC, C/FWD …)
  - **Initial State**: farmer, wolf, duck and corn in the south shore FWDC/-
  - **Goal State**: farmer, duck and corn in the north shore -/FWDC
  - **Operators**: the farmer takes in the boat at most one item from one side to the other side
    (F-Takes-W, F-Takes-D, F-Takes-C, F-Takes-Self [himself only])
  - **Path cost**: the number of crossings

The River Problem

- **Problem solution:** (path Cost = 7)
  - While there are other possibilities here is one **7** step solution to the river problem



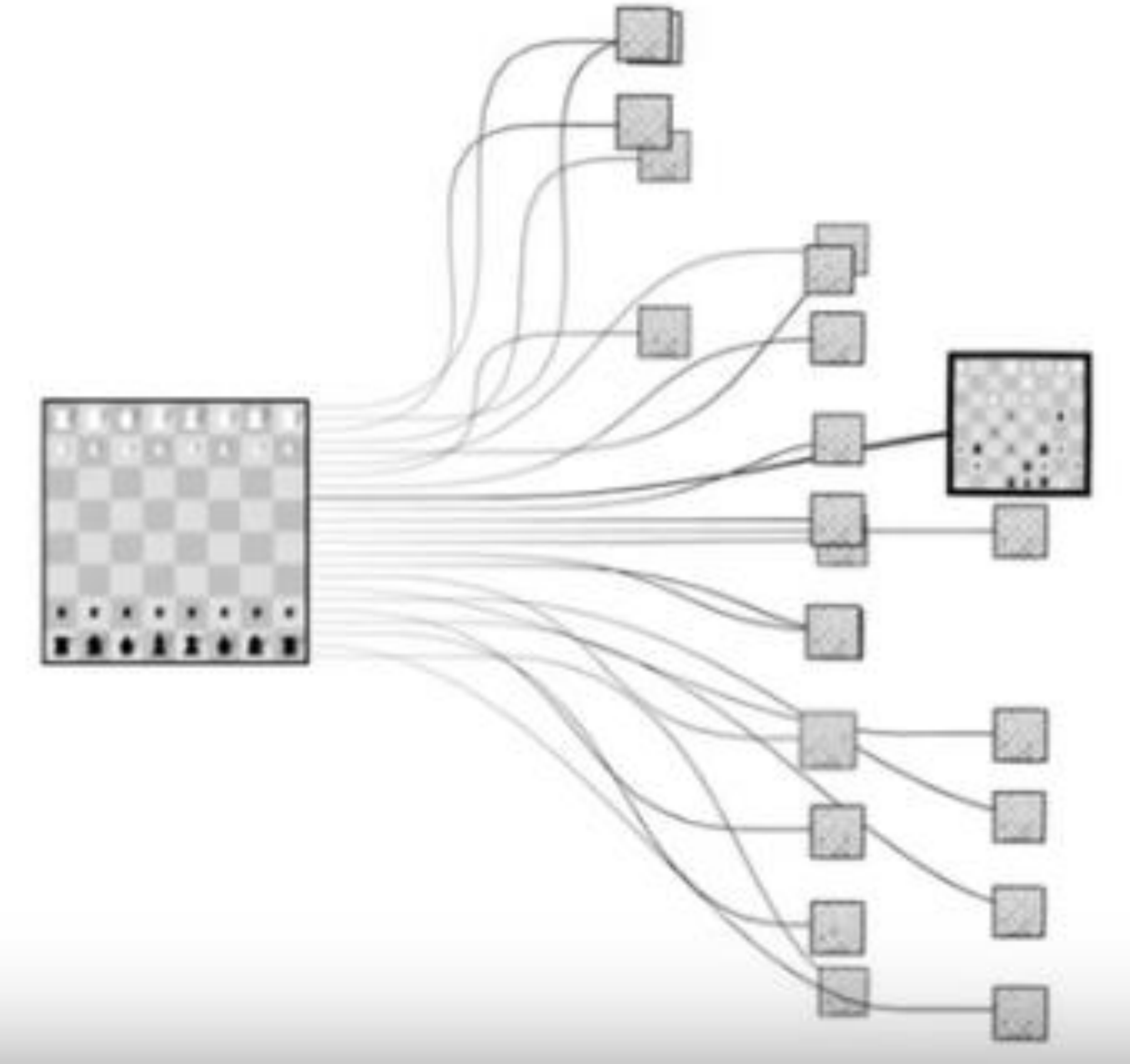
Summary

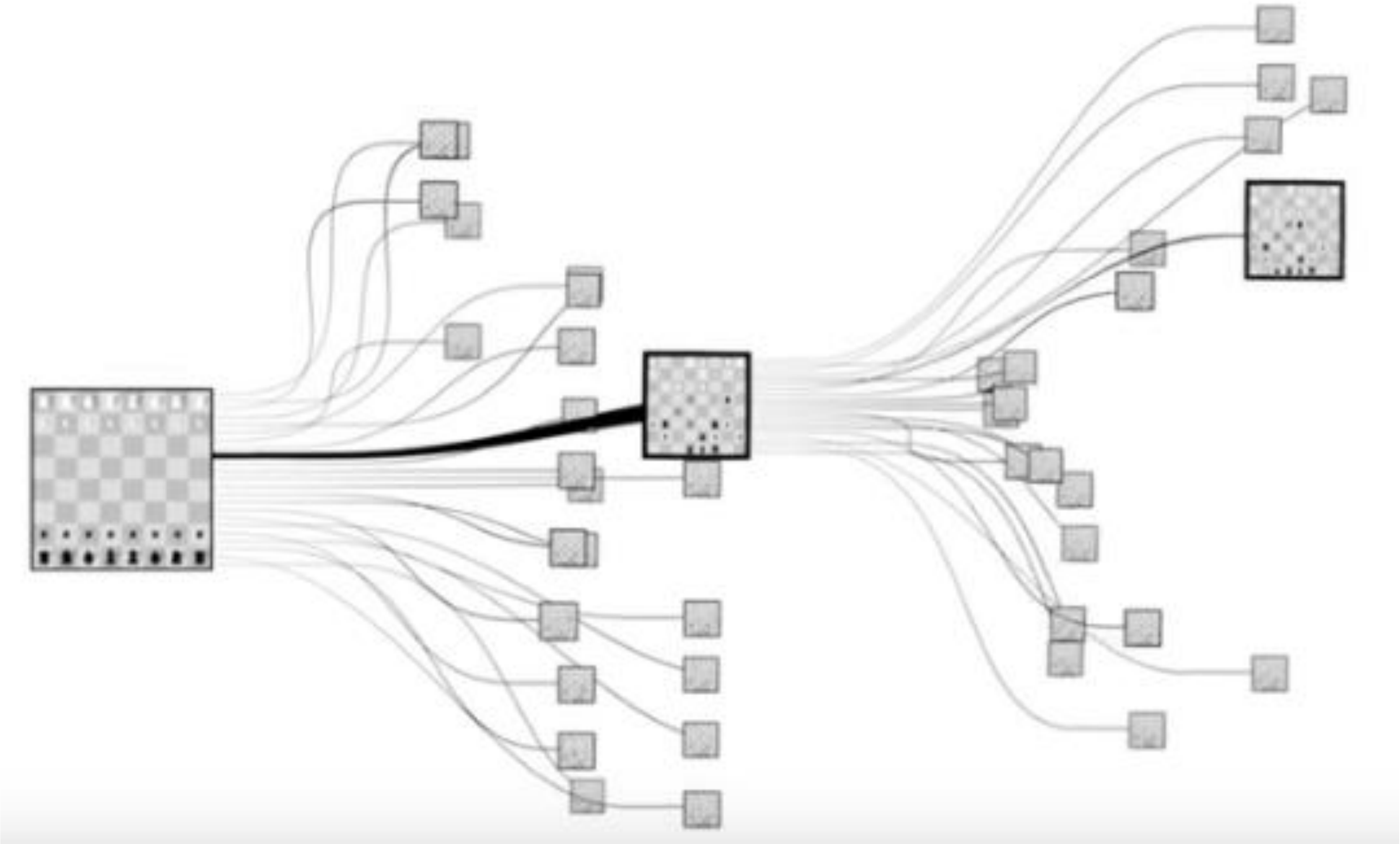
- **Search**: process of constructing sequences of actions that achieve a goal given a problem.
- It is assumed that the environment is **observable, deterministic, static and completely known**.
- **Goal formulation** is the first step in solving problems by searching. It facilitates problem formulation.
- **Formulating a problem** requires specifying five components:
  - State representation,
  - Initial state,
  - Goal state,
  - Operators (actions), and
  - Path cost function.

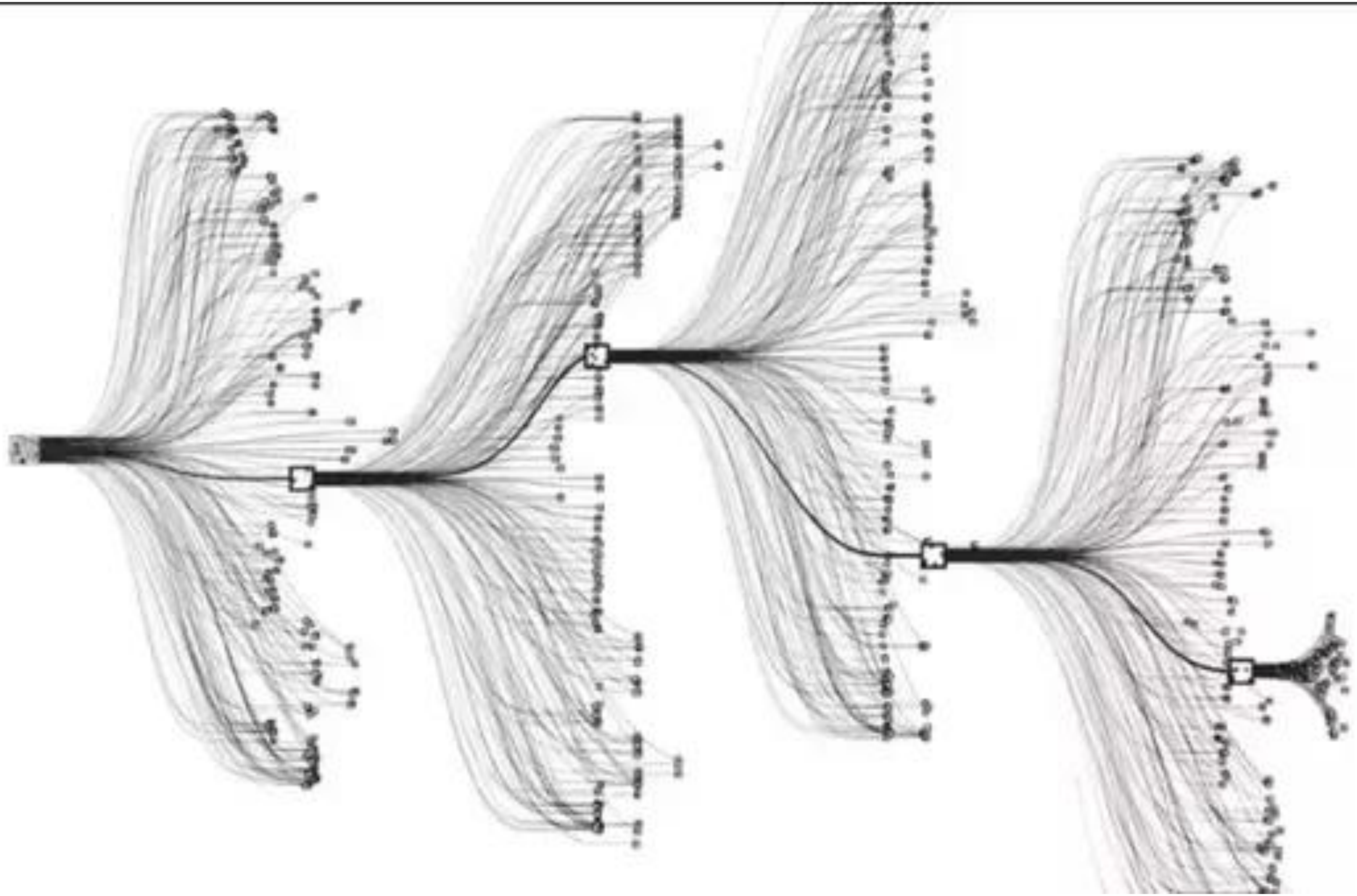
Search Strategies (Next Lecture)

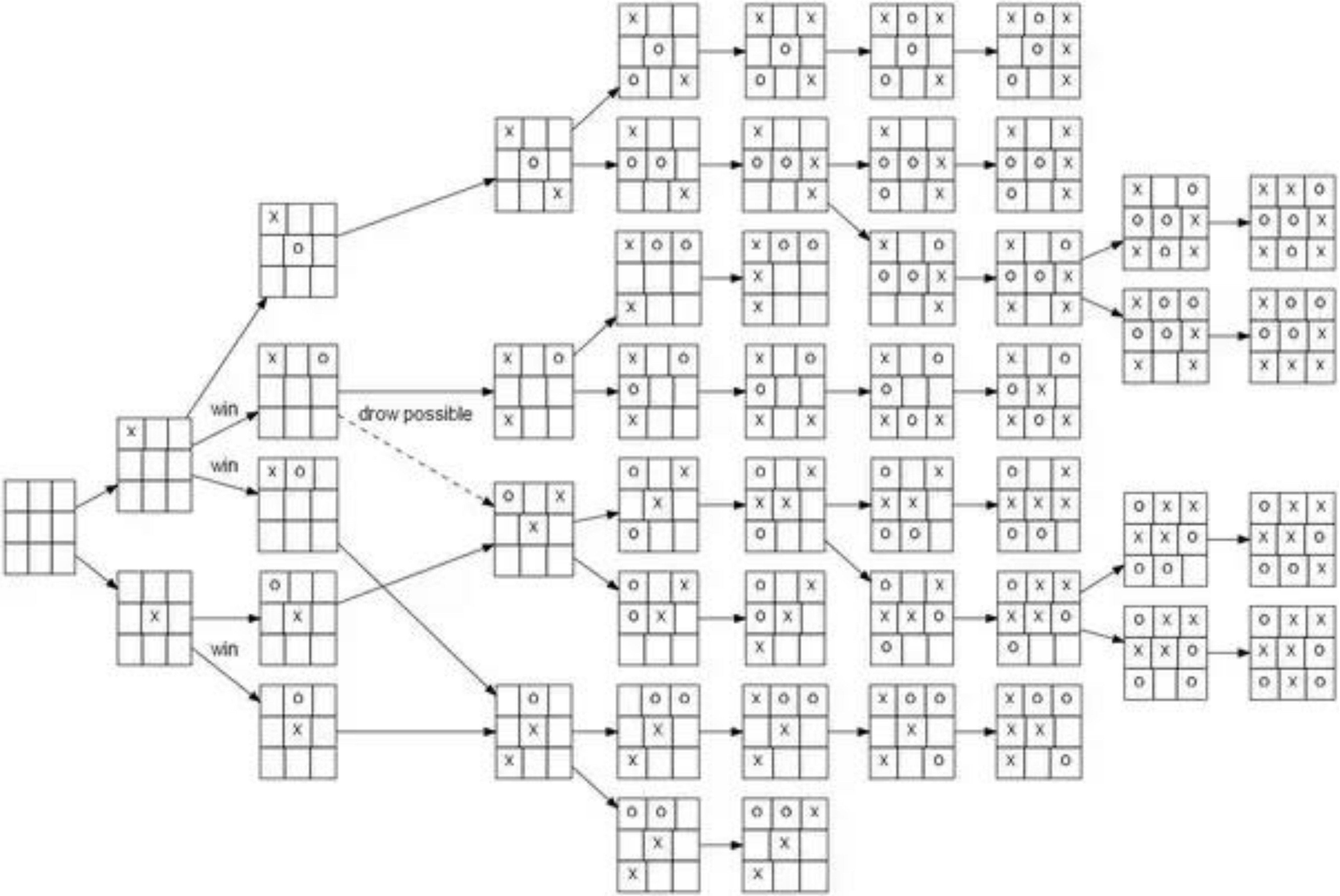
- Several general-purpose search algorithms that can be used to solve these problems.

1. Uninformed search algorithms
    - algorithms that are given no information about the problem other than its definition
    - Although some of these algorithms can solve any solvable problem, none of them can do so efficiently
2. Informed search algorithms
    - can do quite well given some guidance on where to look for solutions








win
win
win
drow possible