CAP 6635 – Artificial Intelligence

Lecture 8: Foundations of AI (Part 3)

Oge Marques, PhD

Professor

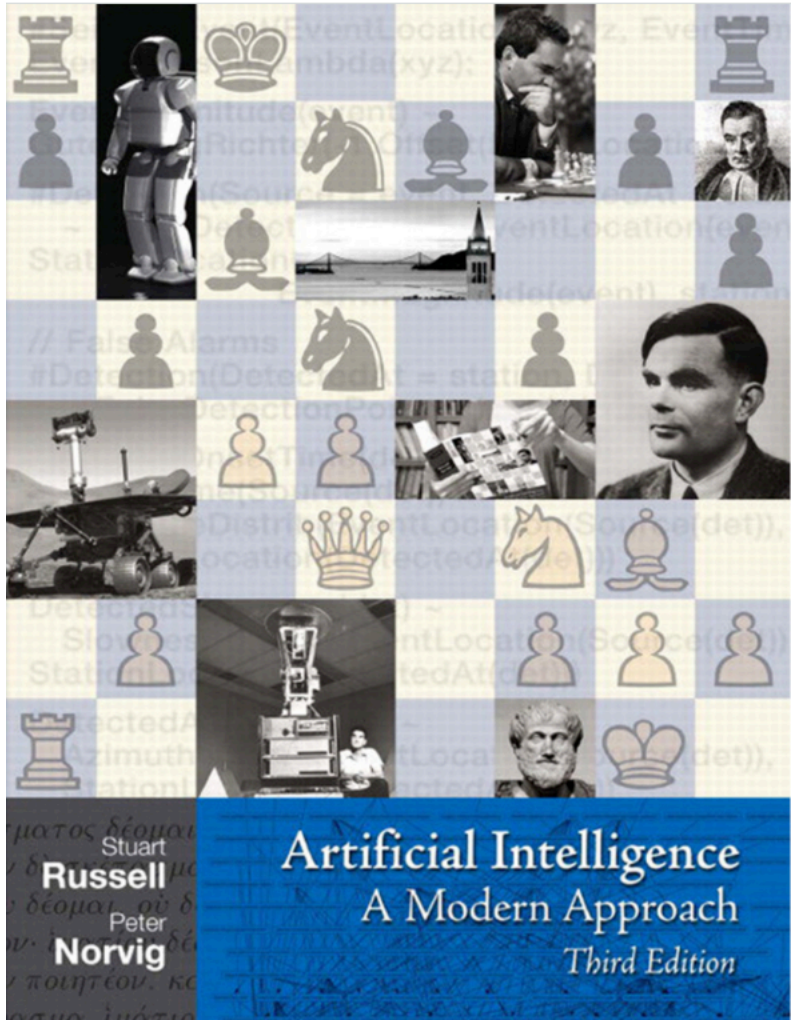
College of Engineering and Computer Science

College of Business

FAU FLORIDA ATLANTIC UNIVERSITY

@ProfessorOge

ProfessorOgeMarques



A whirlwind tour of Russell & Norvig 3/e

Outline

- Introduction
- Intelligent agents and their environments
- Solving problems by searching
- Adversarial search and games
- Constraint satisfaction problems (CSPs)
- Logical Agents
- Planning
- Knowledge representation
- Uncertain knowledge and reasoning
- Learning
- Communicating, perceiving, and acting

Introduction

Introduction

- What is AI?

Thinking Humanly	**Thinking Rationally**
"The exciting new effort to make computers think . . . *machines with minds*, in the full and literal sense." (Haugeland, 1985) "[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning . . ." (Bellman, 1978)	"The study of mental faculties through the use of computational models." (Charniak and McDermott, 1985) "The study of the computations that make it possible to perceive, reason, and act." (Winston, 1992)
Acting Humanly	**Acting Rationally**
"The art of creating machines that perform functions that require intelligence when performed by people." (Kurzweil, 1990) "The study of how to make computers do things at which, at the moment, people are better." (Rich and Knight, 1991)	"Computational Intelligence is the study of the design of intelligent agents." (Poole *et al.*, 1998) "AI . . . is concerned with intelligent behavior in artifacts." (Nilsson, 1998)

Figure 1.1 Some definitions of artificial intelligence, organized into four categories.

Turing Test: In order to pass it…

… the computer would need to possess the following capabilities:

- **natural language processing**: to enable it to communicate successfully in English;
- **knowledge representation**: to store what it knows or hears;
- **automated reasoning**: to use the stored information to answer questions and to draw new conclusions;
- **machine learning**: to adapt to new circumstances and to detect and extrapolate patterns.

Total Turing Test…

…would include a video signal so that the interrogator can test the subject's perceptual abilities, as well as the opportunity for the interrogator to pass physical objects "through the hatch."

To pass the total Turing Test, the computer will need:

- **computer vision** to perceive objects, and
- **robotics** to manipulate objects and move about.

Foundations of AI

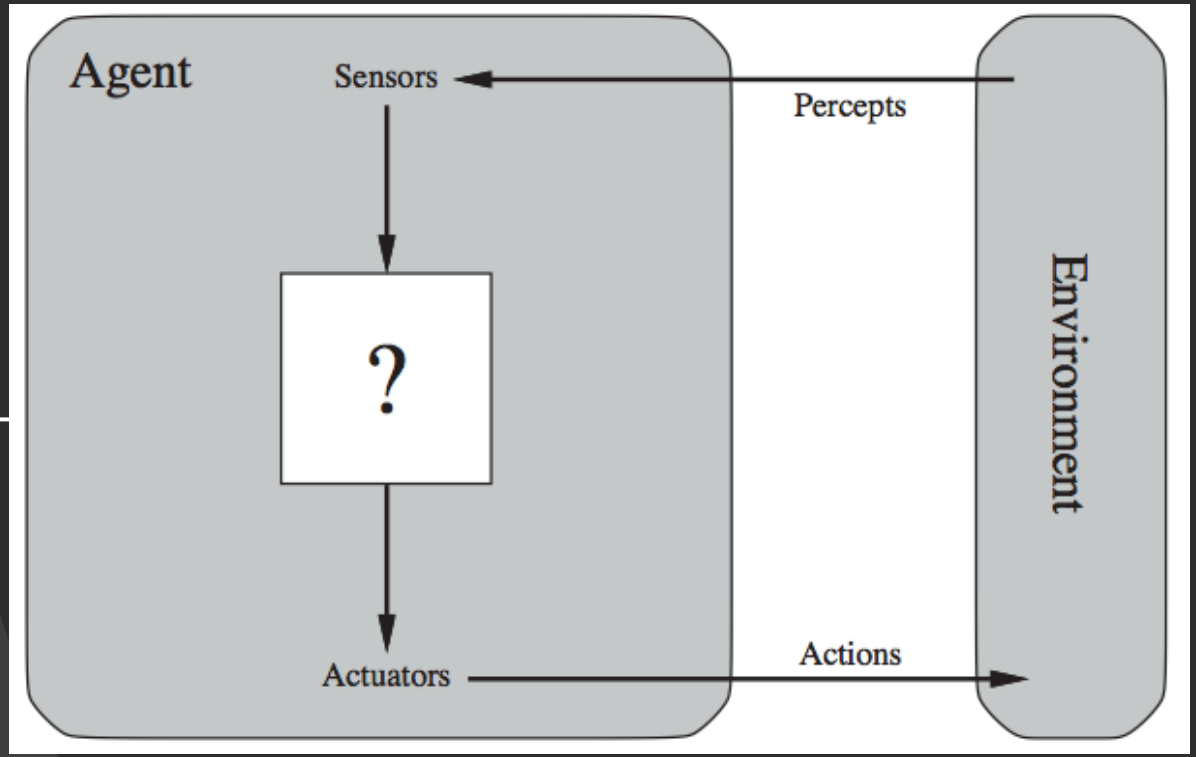
- Philosophy
- Mathematics
- Economics
- Neuroscience
- Psychology (esp. Cognitive Psychology)
- Computer Engineering
- Control Theory and Cybernetics
- Linguistics

Intelligent agents and their environments

In which we discuss the nature of agents, perfect or otherwise, the diversity of environments, and the resulting menagerie of agent types.

Agents

- An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators
- Human agent:
  - eyes, ears, and other organs for sensors;
  - hands, legs, mouth, and other body parts for actuators
- Robotic agent:
  - cameras and infrared range finders for sensors;
  - various motors for actuators



Rational agents

- For each possible percept sequence, a rational agent should:
  - select an action that is expected to maximize its performance measure,
  - based on the evidence provided by the percept sequence and
  - whatever built-in knowledge the agent has.

PEAS: Performance measure, Environment, Actuators, Sensors

Agent Type	Performance Measure	Environment	Actuators	Sensors
Medical diagnosis system	Healthy patient, reduced costs	Patient, hospital, staff	Display of questions, tests, diagnoses, treatments, referrals	Keyboard entry of symptoms, findings, patient's answers
Satellite image analysis system	Correct image categorization	Downlink from orbiting satellite	Display of scene categorization	Color pixel arrays
Part-picking robot	Percentage of parts in correct bins	Conveyor belt with parts; bins	Jointed arm and hand	Camera, joint angle sensors
Refinery controller	Purity, yield, safety	Refinery, operators	Valves, pumps, heaters, displays	Temperature, pressure, chemical sensors
Interactive English tutor	Student's score on test	Set of students, testing agency	Display of exercises, suggestions, corrections	Keyboard entry

Environment types

- **Fully observable** (vs. partially observable): An agent's sensors give it access to the complete state of the environment at each point in time.
- **Deterministic** (vs. stochastic): The next state of the environment is completely determined by the current state and the action executed by the agent.
- **Episodic** (vs. sequential): The agent's experience is divided into atomic "episodes" (each episode consists of the agent perceiving and then performing a single action), and the choice of action in each episode depends only on the episode itself.

Environment types

- **Static** (vs. dynamic): The environment is unchanged while an agent is deliberating. (The environment is semi-dynamic if the environment itself does not change with the passage of time but the agent's performance score does)
- **Discrete** (vs. continuous): A limited number of distinct, clearly defined percepts and actions.
- **Single agent** (vs. multiagent): An agent operating by itself in an environment.

Environment types

Task Environment	Observable	Agents	Deterministic	Episodic	Static	Discrete
Crossword puzzle	Fully	Single	Deterministic	Sequential	Static	Discrete
Chess with a clock	Fully	Multi	Deterministic	Sequential	Semi	Discrete
Poker	Partially	Multi	Stochastic	Sequential	Static	Discrete
Backgammon	Fully	Multi	Stochastic	Sequential	Static	Discrete
Taxi driving	Partially	Multi	Stochastic	Sequential	Dynamic	Continuous
Medical diagnosis	Partially	Single	Stochastic	Sequential	Dynamic	Continuous
Image analysis	Fully	Single	Deterministic	Episodic	Semi	Continuous
Part-picking robot	Partially	Single	Stochastic	Episodic	Dynamic	Continuous
Refinery controller	Partially	Single	Stochastic	Sequential	Dynamic	Continuous
Interactive English tutor	Partially	Multi	Stochastic	Sequential	Dynamic	Discrete

Agent types
(in order of increasing generality)

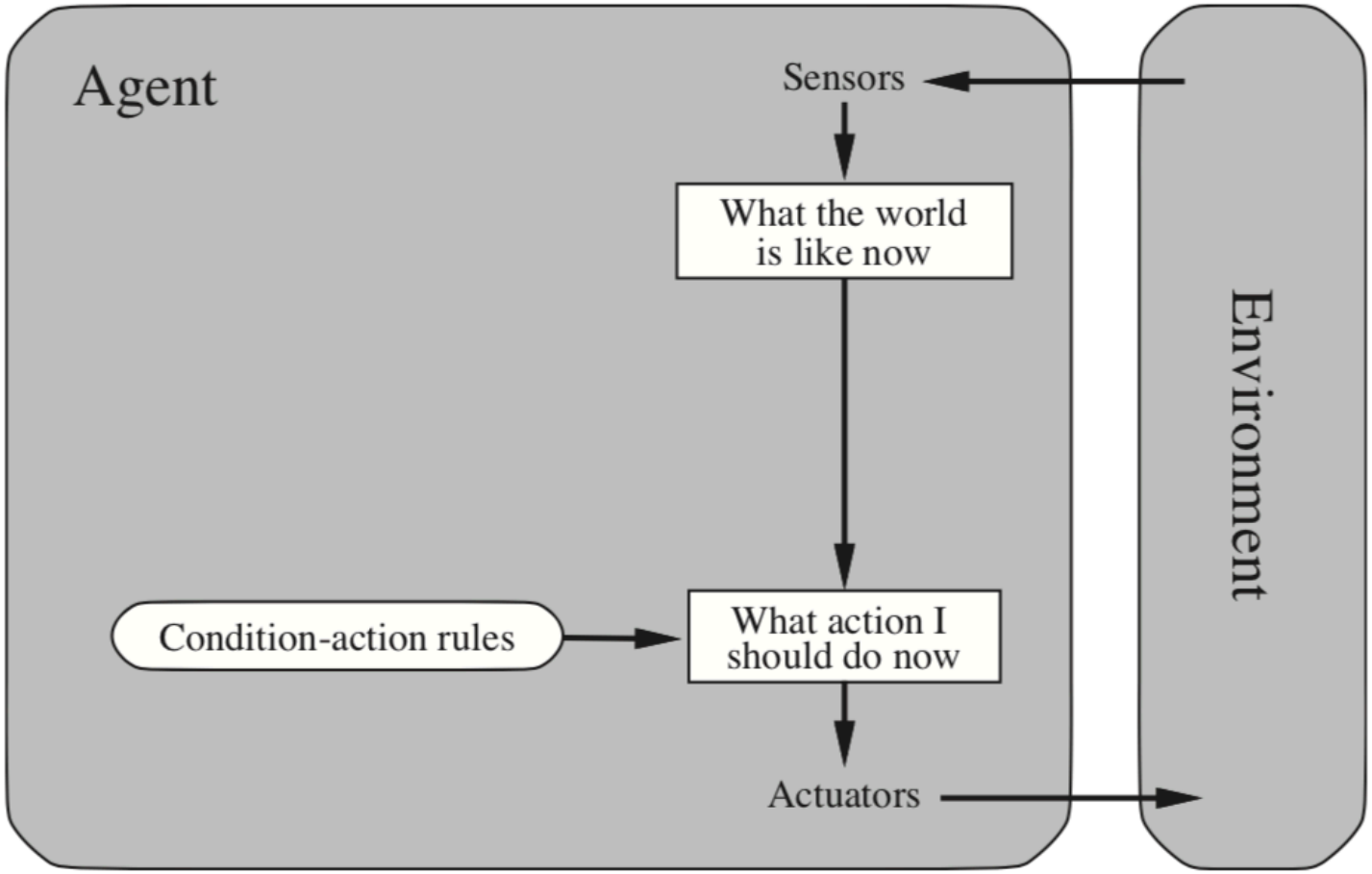
Simple reflex agents

Agents that keep track of the world

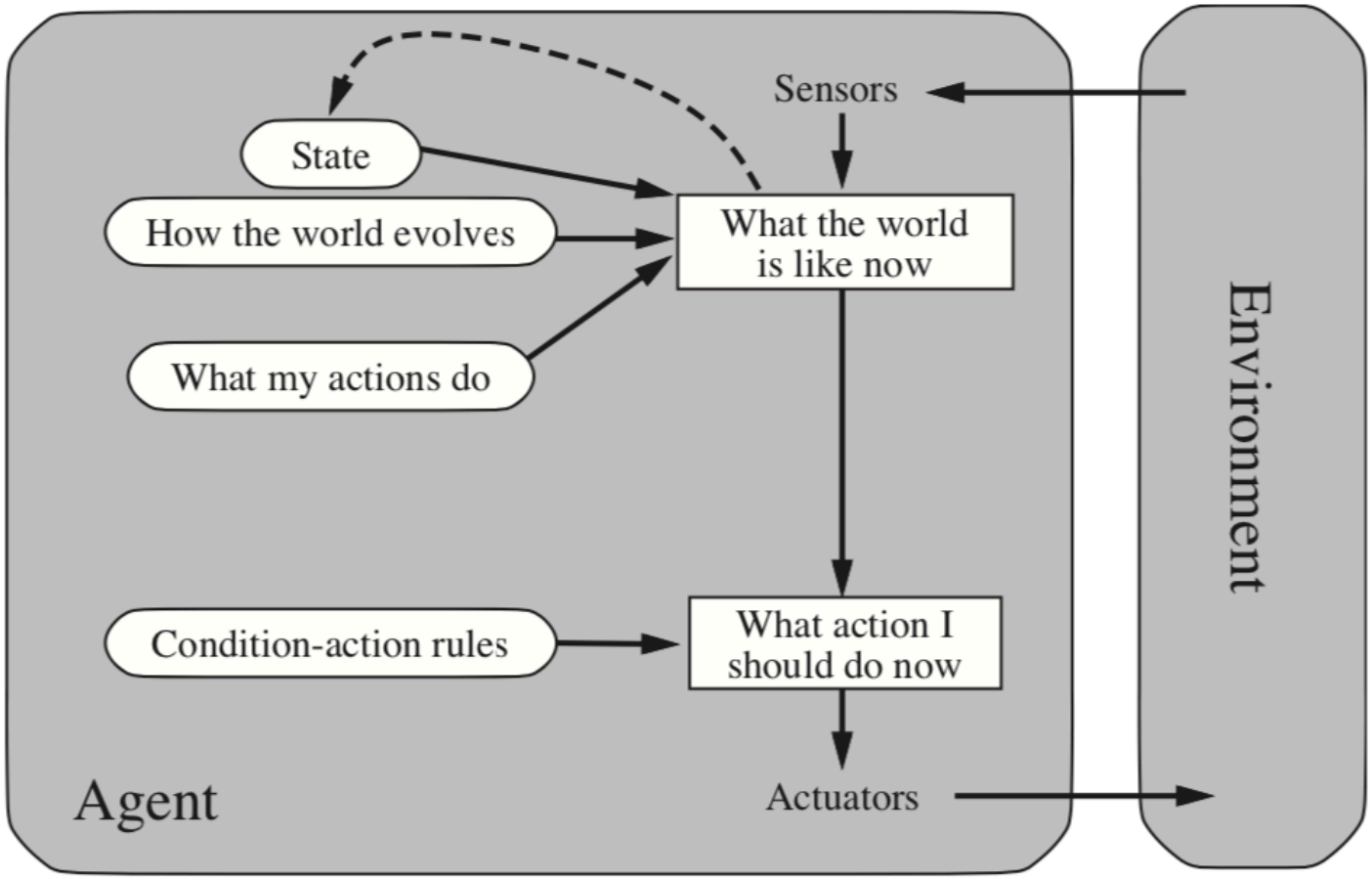
Goal-based agents

Utility-based agents

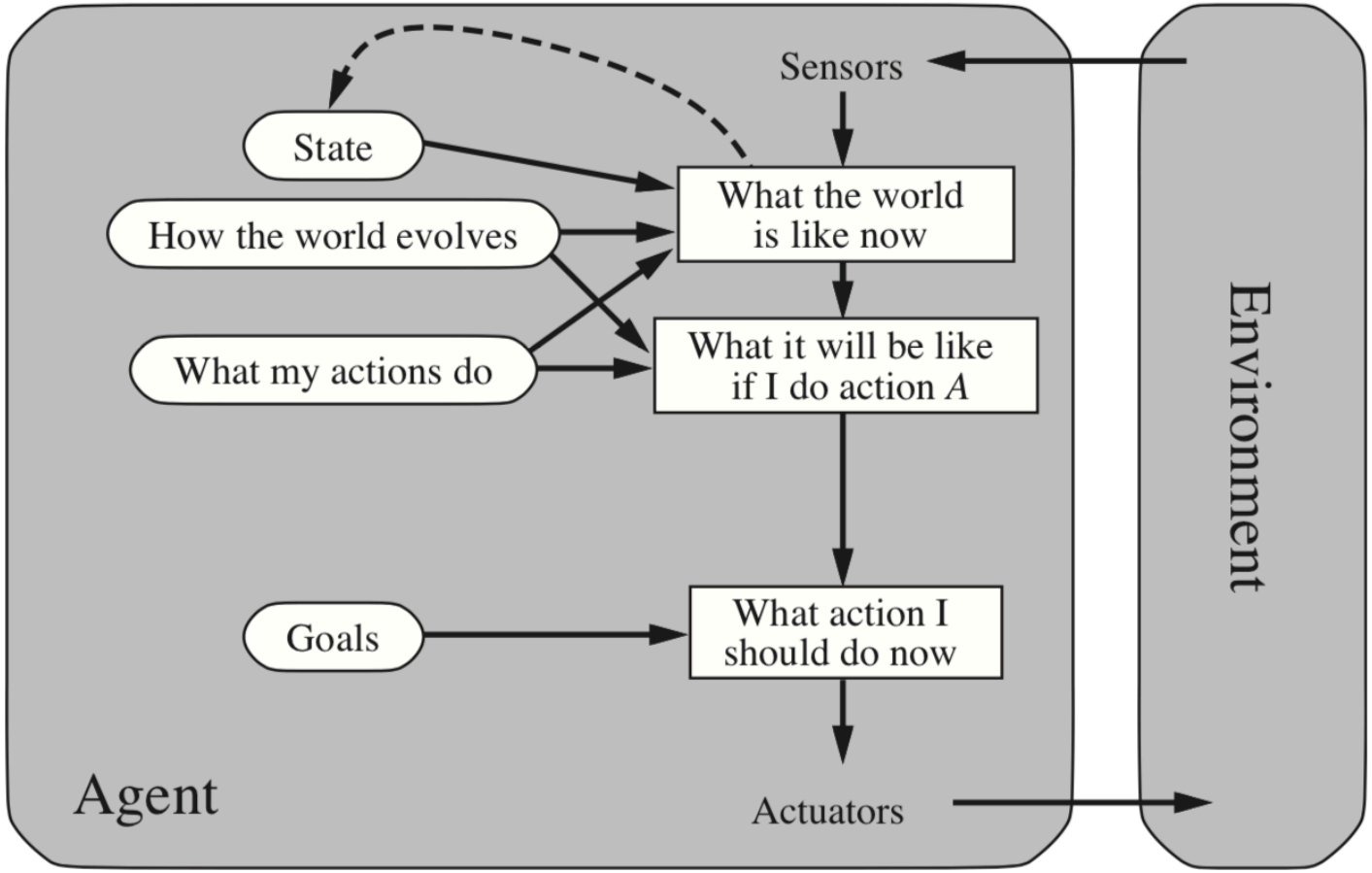
Simple reflex agents



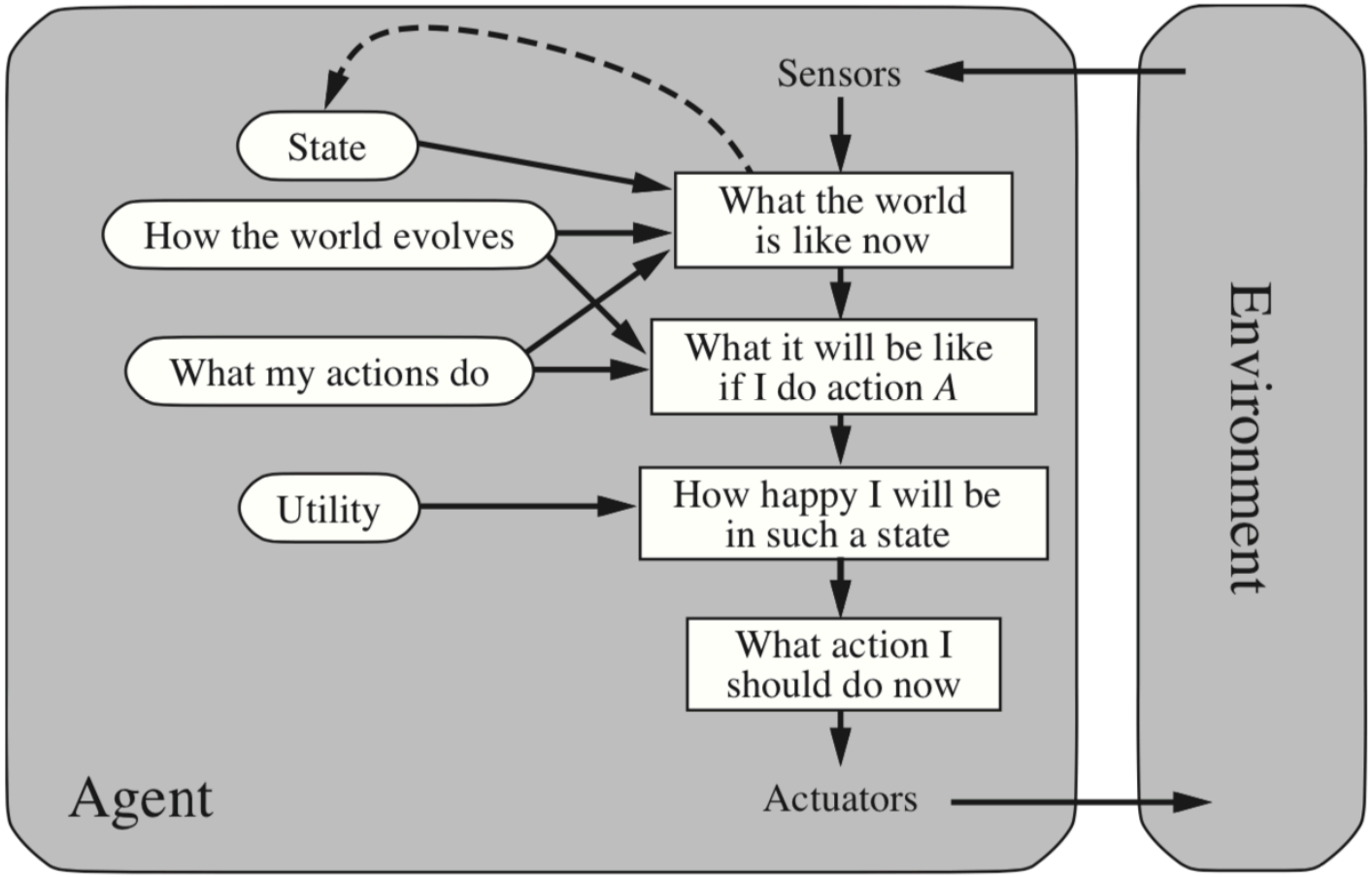
Agents that keep track of the world (Agents with internal states)



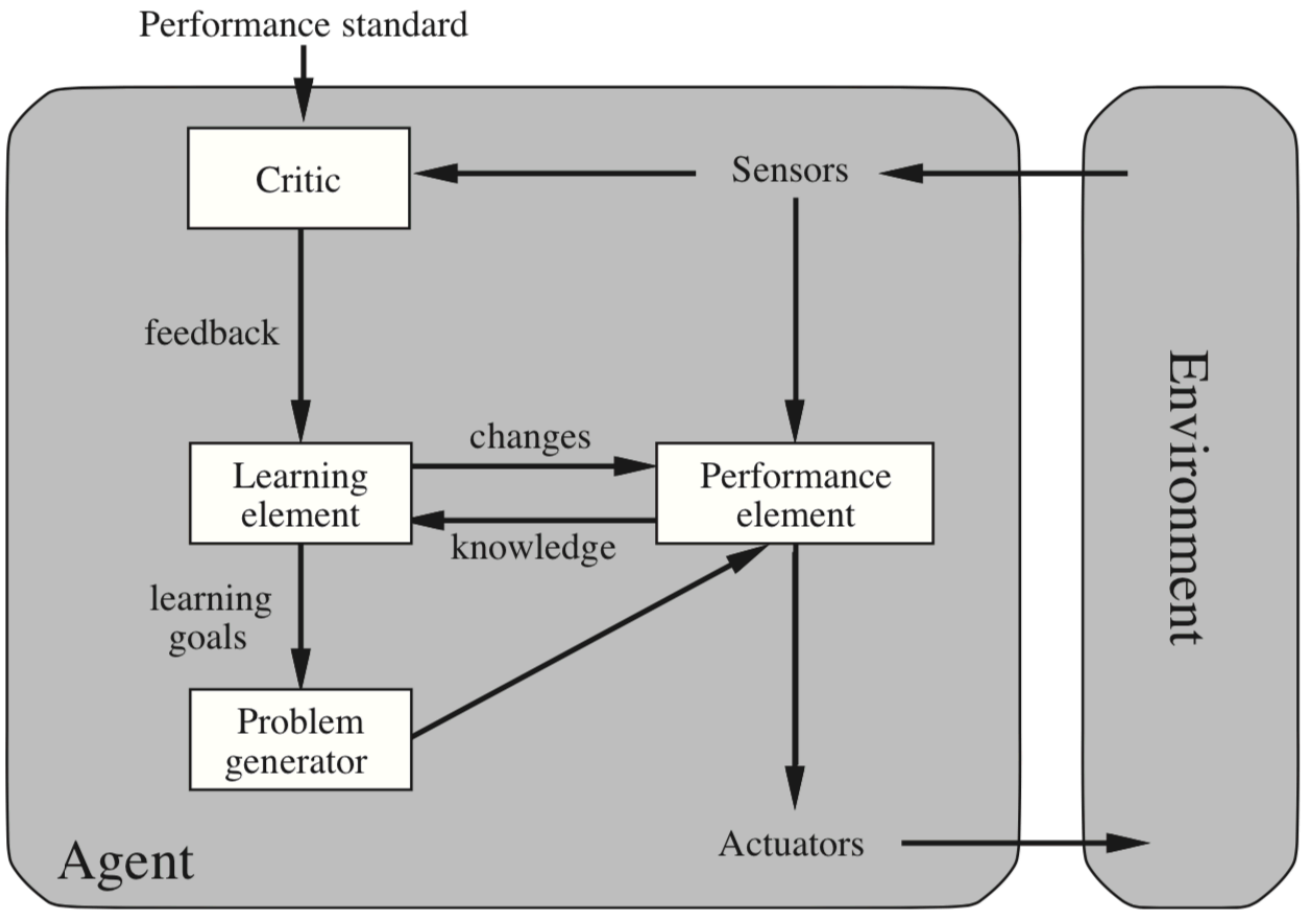
Model-based, goal-based agent



Model-based, utility-based agent



A general learning agent



Search strategies

In which we see how an agent can find a sequence of actions that achieves its goals when no single action will do.

Problem-solving agents

Restricted form of general agent:

```
function SIMPLE-PROBLEM-SOLVING-AGENT(percept) returns an action
    static: seq, an action sequence, initially empty
            state, some description of the current world state
            goal, a goal, initially null
            problem, a problem formulation

    state ← UPDATE-STATE(state, percept)
    if seq is empty then
        goal ← FORMULATE-GOAL(state)
        problem ← FORMULATE-PROBLEM(state, goal)
        seq ← SEARCH(problem)
    action ← RECOMMENDATION(seq, state)
    seq ← REMAINDER(seq, state)
    return action
```

Example: Romania

On holiday in Romania; currently in Arad.
Flight leaves tomorrow from Bucharest

Formulate goal:

be in Bucharest

Formulate problem:

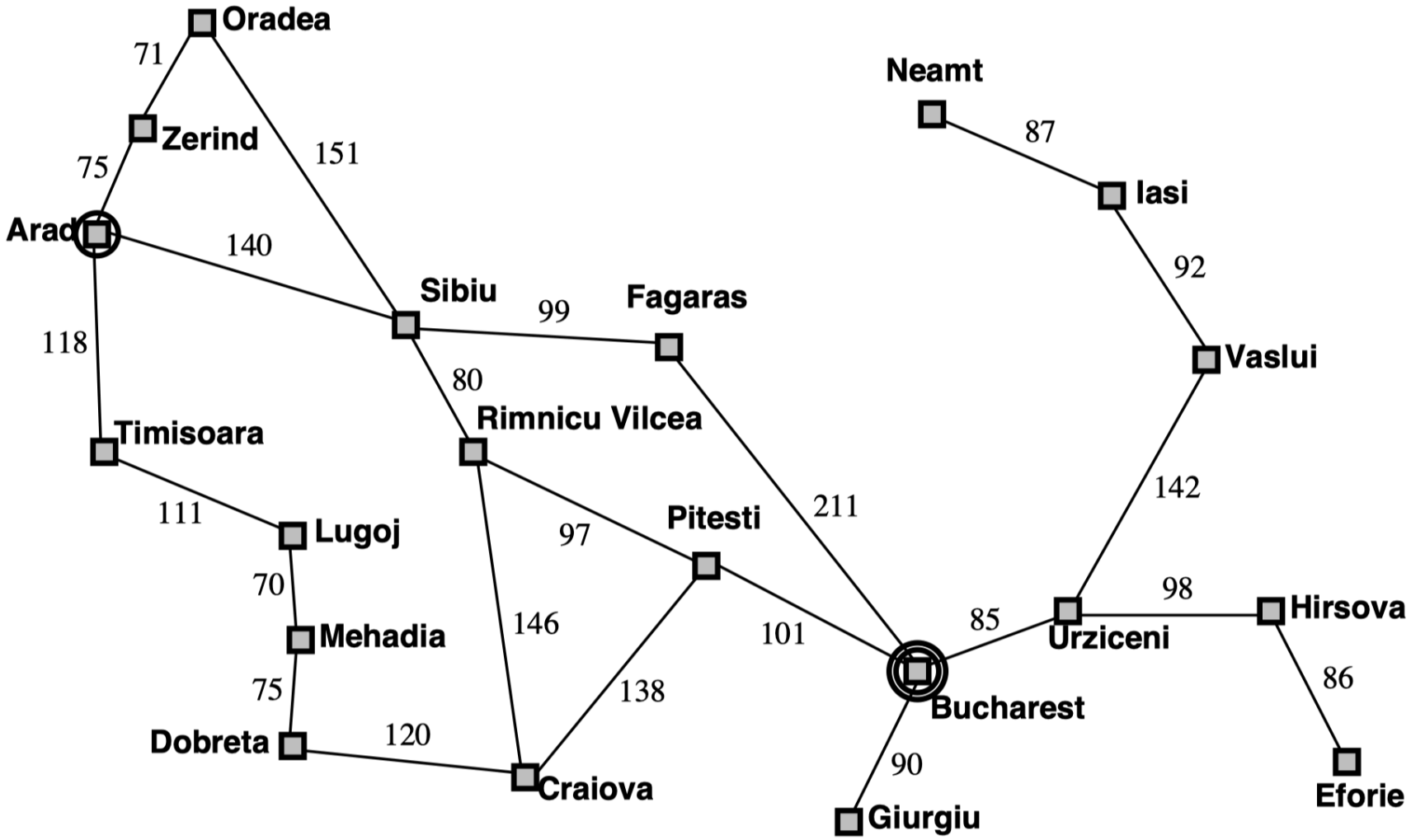
states: various cities

actions: drive between cities

Find solution:

sequence of cities, e.g., Arad, Sibiu, Fagaras, Bucharest

Example: Romania



Tree search algorithms

Basic idea:

offline, simulated exploration of state space

by generating successors of already-explored states

(a.k.a. expanding states)

```
function TREE-SEARCH( problem, strategy) returns a solution, or failure
  initialize the search tree using the initial state of problem
  loop do
    if there are no candidates for expansion then return failure
    choose a leaf node for expansion according to strategy
    if the node contains a goal state then return the corresponding solution
    else expand the node and add the resulting nodes to the search tree
  end
```

Search strategies

A strategy is defined by picking the **order of node expansion**

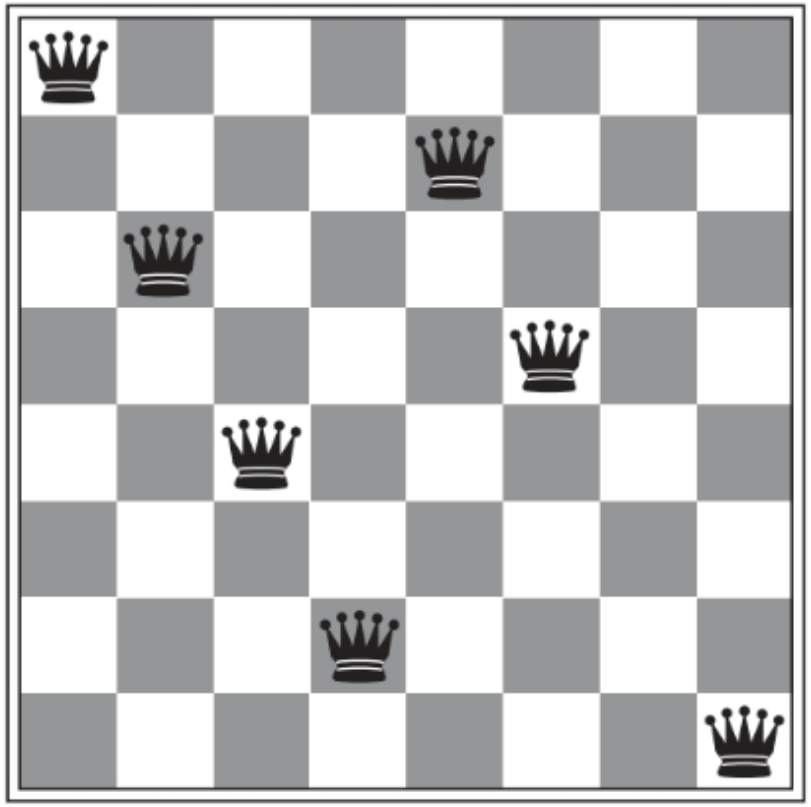
Strategies are evaluated along the following dimensions:

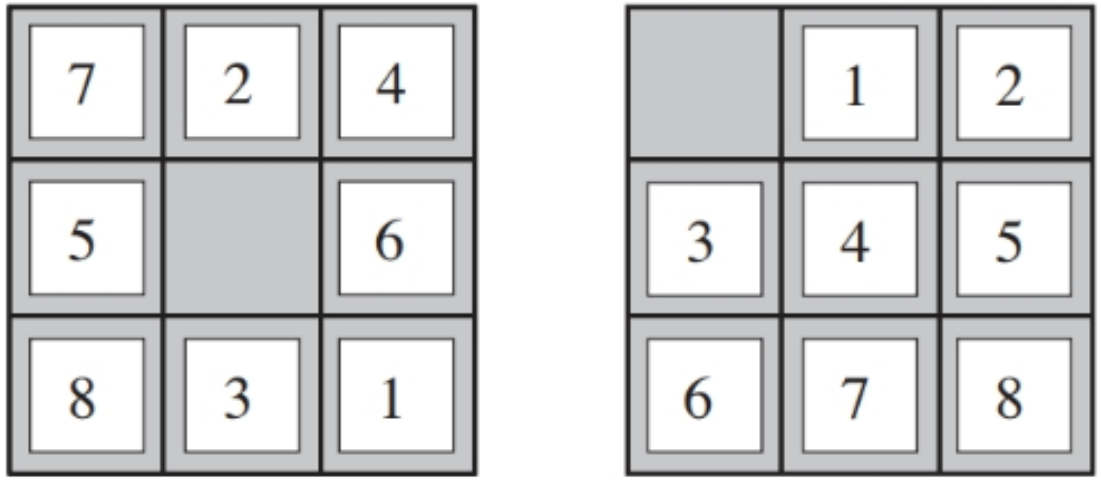
- completeness—does it always find a solution if one exists?
- time complexity—number of nodes generated/expanded
- space complexity—maximum number of nodes in memory
- optimality—does it always find a least-cost solution?

Time and space complexity are measured in terms of

- b—maximum branching factor of the search tree
- d—depth of the least-cost solution
- m—maximum depth of the state space (may be ∞)

Search strategies: toy problems





Start State

Goal State

Uninformed (“blind”) search strategies

- Breadth-first search
- Uniform-cost search
- Depth-first search
- Depth-limited search
- Iterative deepening search

Evaluation of tree-search strategies

Criterion	Breadth-First	Uniform-Cost	Depth-First	Depth-Limited	Iterative Deepening	Bidirectional (if applicable)
Complete?	Yes[a]	Yes[a,b]	No	No	Yes[a]	Yes[a,d]
Time	$O(b^d)$	$O(b^{1+\lfloor C^*/\epsilon \rfloor})$	$O(b^m)$	$O(b^\ell)$	$O(b^d)$	$O(b^{d/2})$
Space	$O(b^d)$	$O(b^{1+\lfloor C^*/\epsilon \rfloor})$	$O(bm)$	$O(b\ell)$	$O(bd)$	$O(b^{d/2})$
Optimal?	Yes[c]	Yes	No	No	Yes[c]	Yes[c,d]

Figure 3.21 Evaluation of tree-search strategies. b is the branching factor; d is the depth of the shallowest solution; m is the maximum depth of the search tree; l is the depth limit. Superscript caveats are as follows: [a] complete if b is finite; [b] complete if step costs $\geq \epsilon$ for positive ϵ; [c] optimal if step costs are all identical; [d] if both directions use breadth-first search.

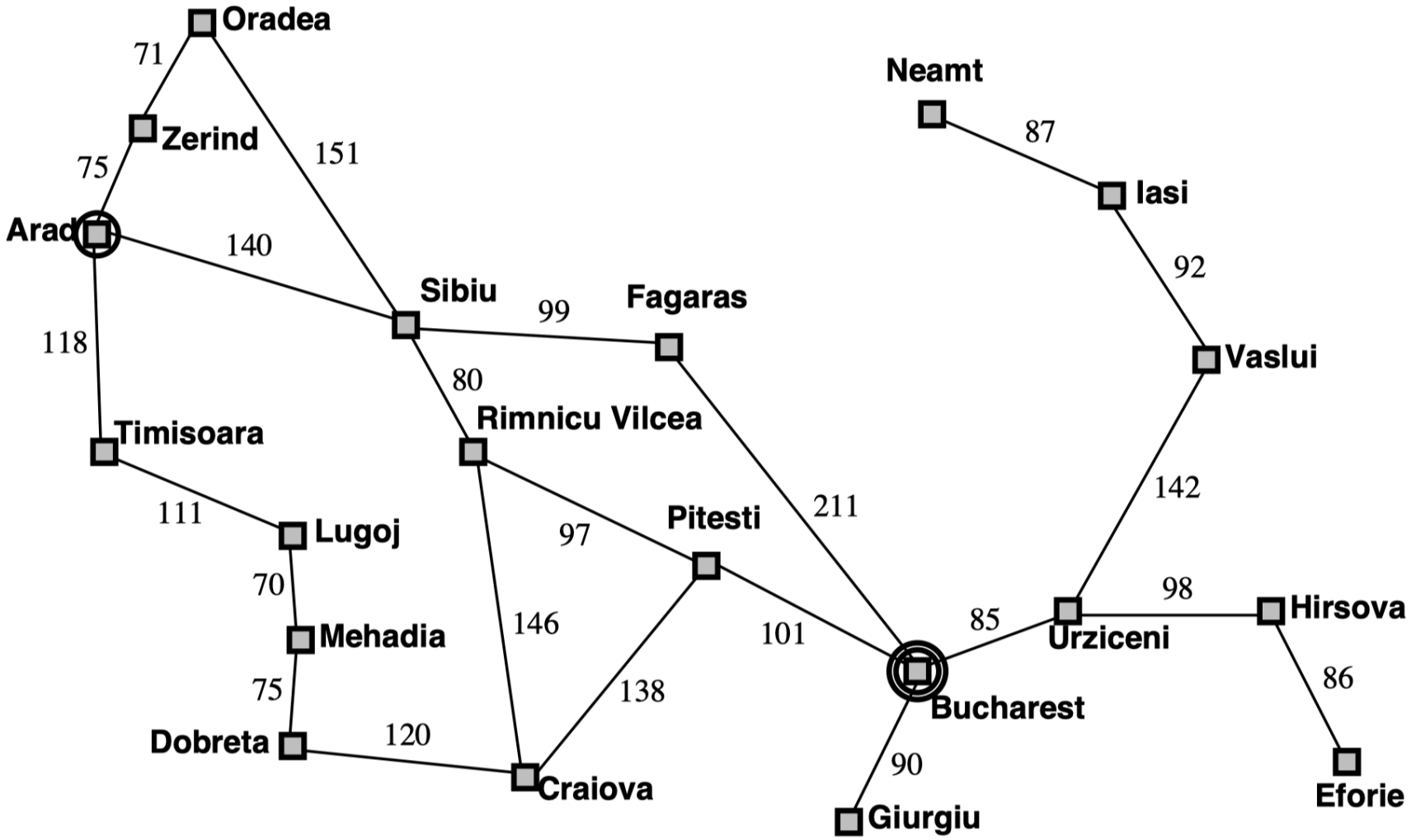
Informed (heuristic) search strategies

Greedy best-first search

A* search

Memory-bound heuristic search

Example: Romania



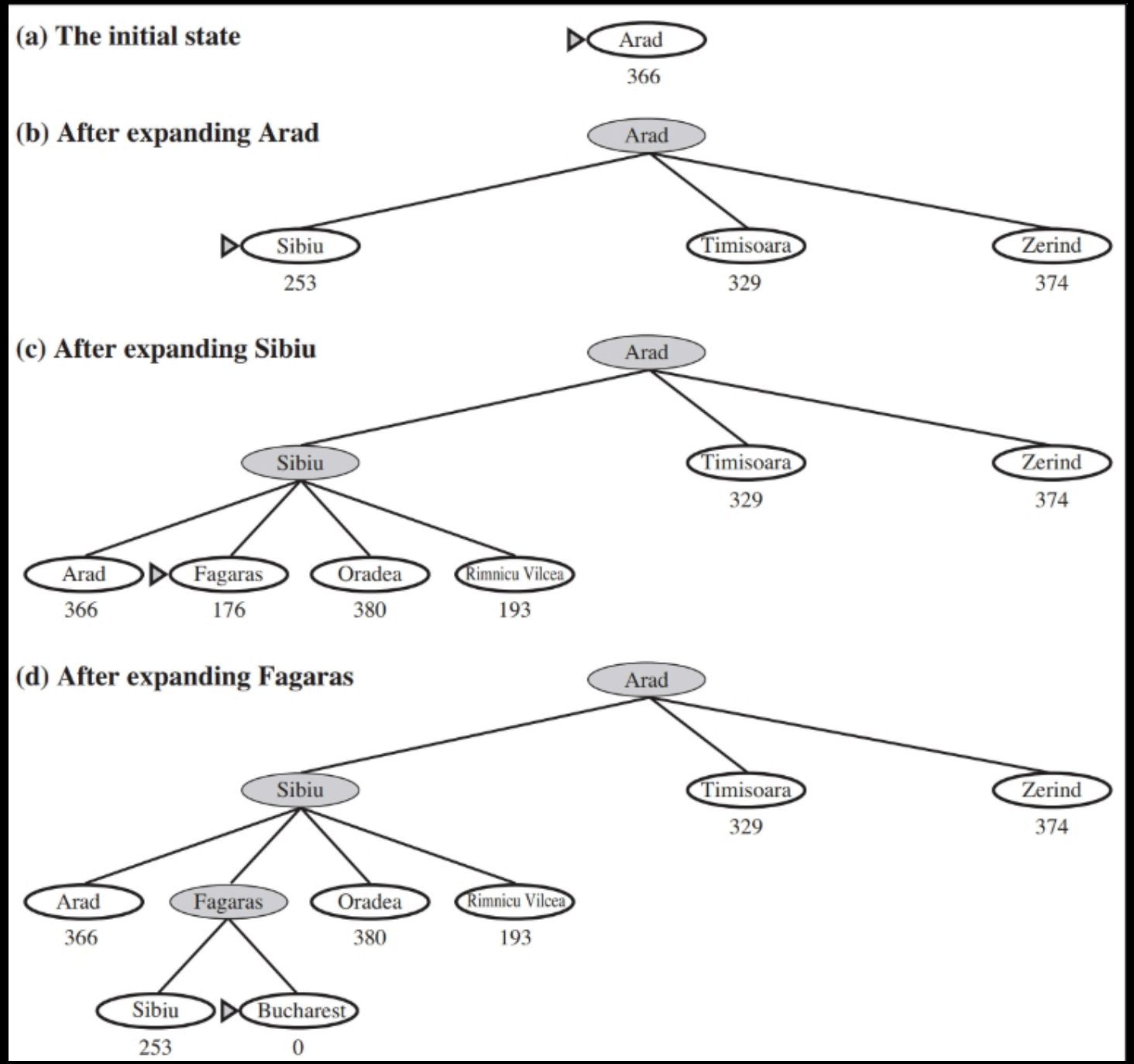
Greedy best-first search

- Tries to expand the node that is closest to the goal, i.e., evaluates nodes by using just a heuristic function.
- Example: Romania – **straight-line distance** as heuristic: h_{SLD}
  - Notice that the values of h_{SLD} cannot be computed from the problem description itself.
  - Moreover, it takes a certain amount of experience to know that h_{SLD} is correlated with actual road distances and is, therefore, a useful heuristic.

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

Greedy best-first search

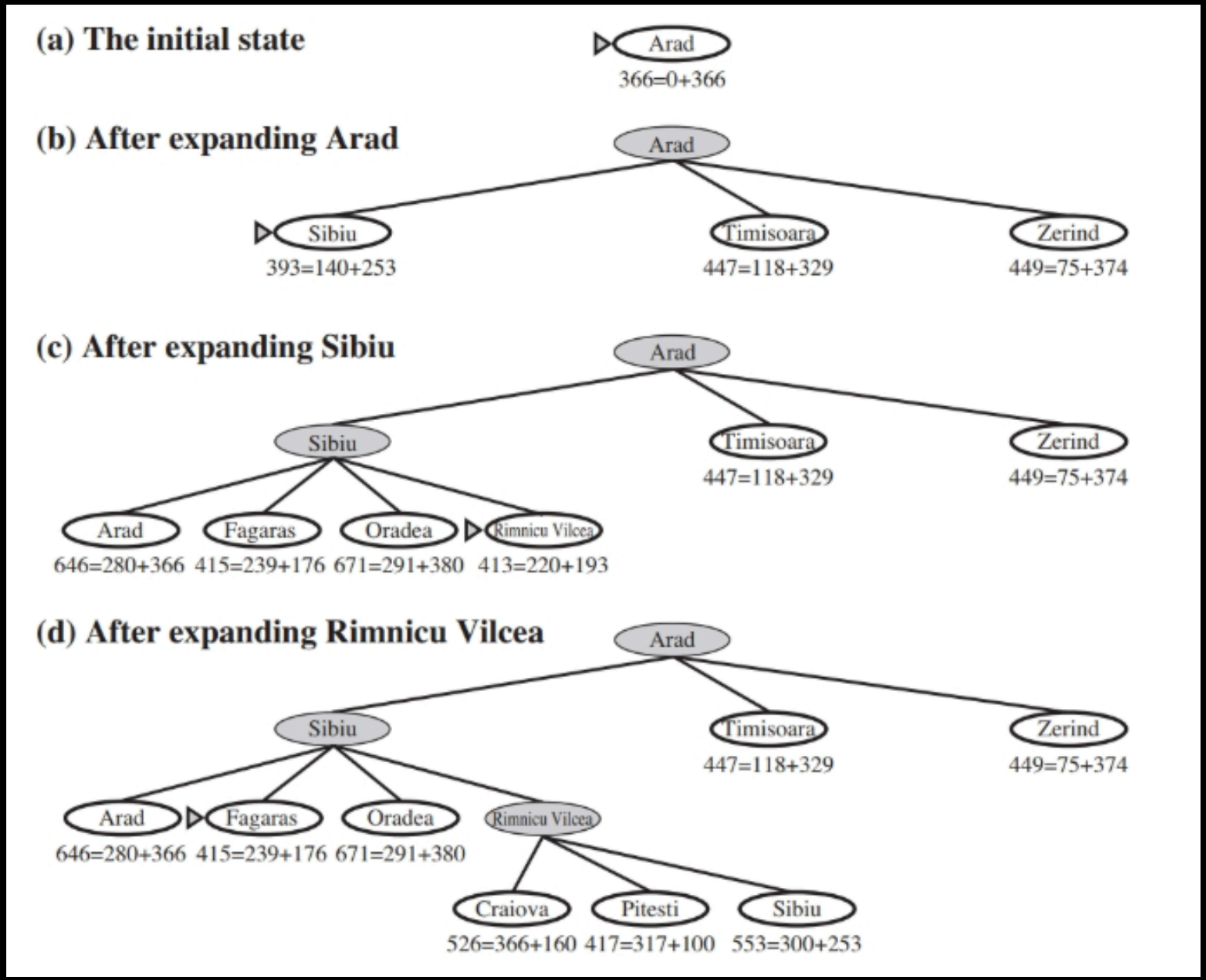
Total distance:
450 km



A* search

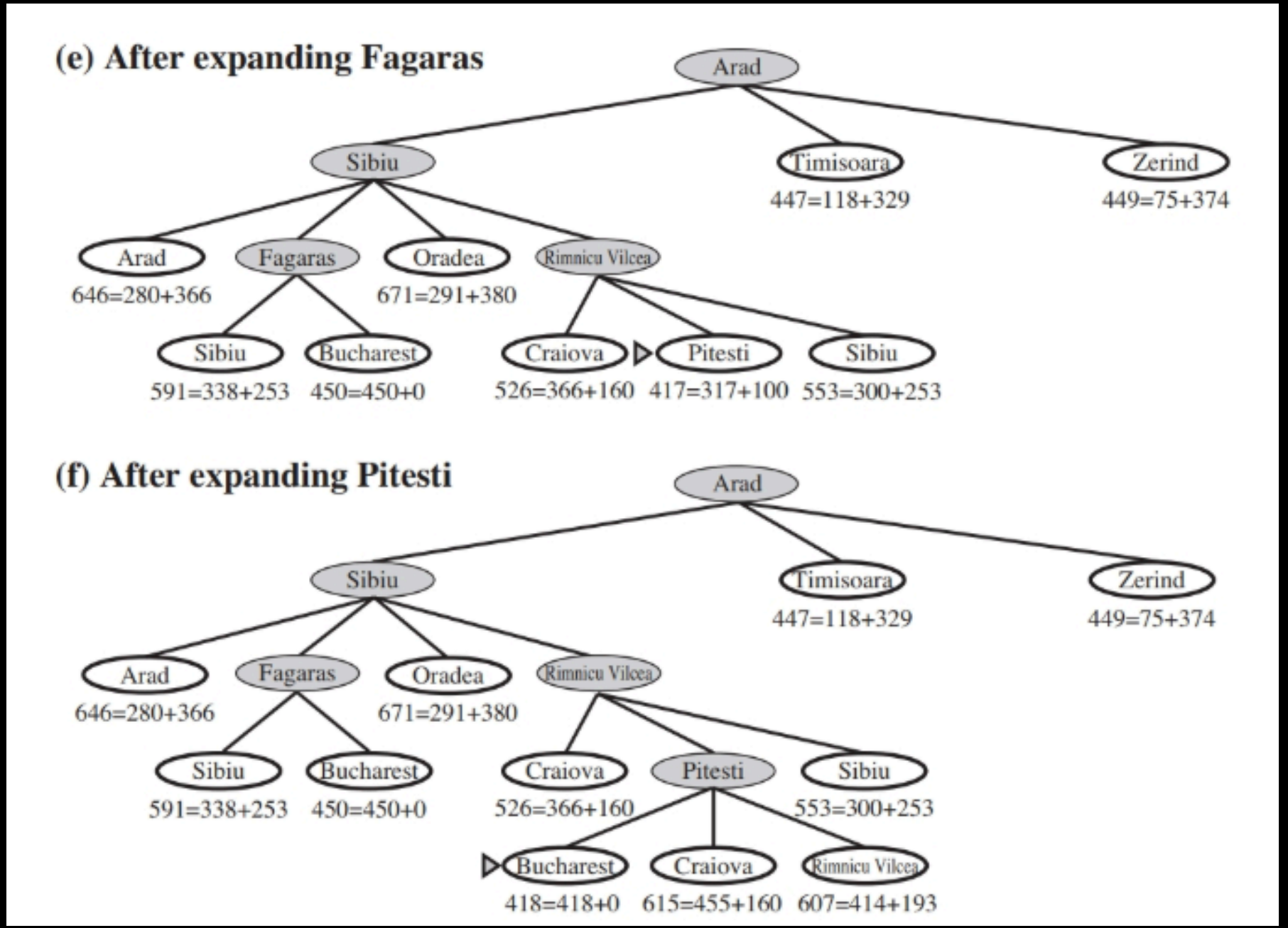
- Evaluates nodes by combining the cost to reach the node (uniform-cost search) with a heuristic function (greedy best-first search).
- Provided that the heuristic function h(n) satisfies certain conditions, **A∗ search is both complete and optimal**.

A* search

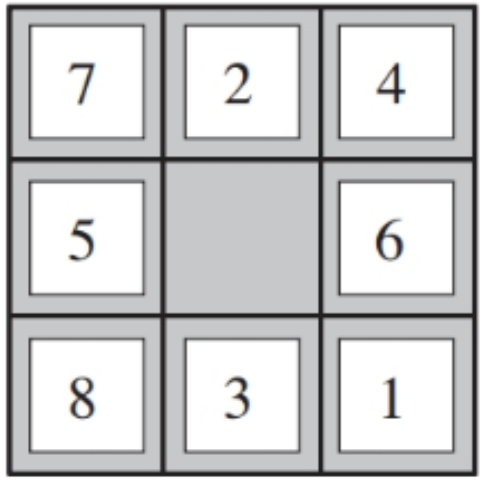

(a) The initial state
Arad
366=0+366
(b) After expanding Arad
Arad
Sibiu
Timisoara
Zerind
393=140+253
447=118+329
449=75+374
(c) After expanding Sibiu
Arad
Sibiu
Timisoara
Zerind
447=118+329
449=75+374
Arad
Fagaras
Oradea
Rimnicu Vilcea
646=280+366
415=239+176
671=291+380
413=220+193
(d) After expanding Rimnicu Vilcea
Arad
Sibiu
Timisoara
Zerind
447=118+329
449=75+374
Arad
Fagaras
Oradea
Rimnicu Vilcea
646=280+366
415=239+176
671=291+380
Craiova
Pitesti
Sibiu
526=366+160
417=317+100
553=300+253

A* search

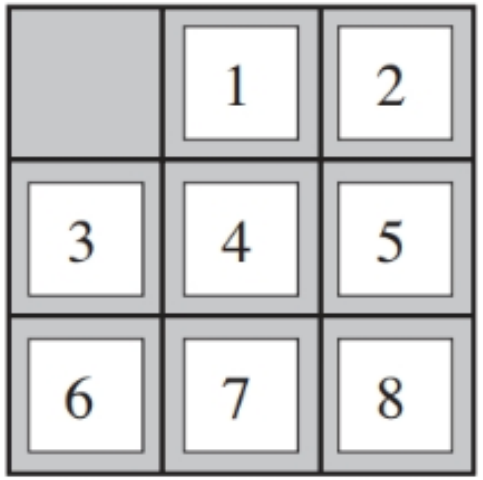
Total distance: 418 km



What makes a good heuristic?



Start State



Goal State

Heuristics are learned from experience!

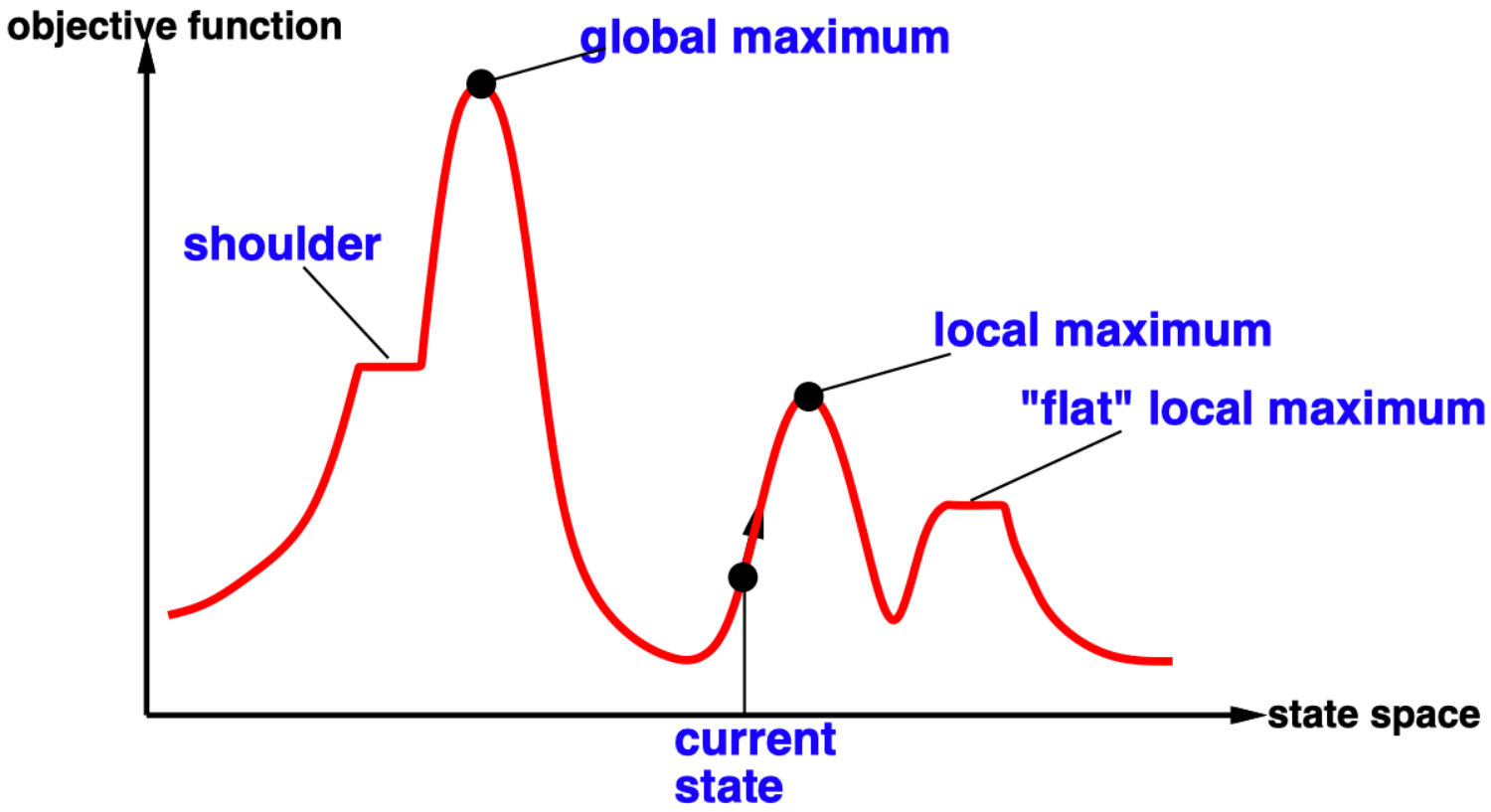
- Average solution cost: 22 steps
- Exhaustive tree search to depth 22 would look at 3^{22} (more than 30 billion) states!

- Possible heuristics:
  - h1 = the number of misplaced tiles: **h1 = 8**.
  - h2 = the sum of the Manhattan distances of the tiles from their goal positions: **h2 = 3+1+2+2+2+3+3+2 = 18**

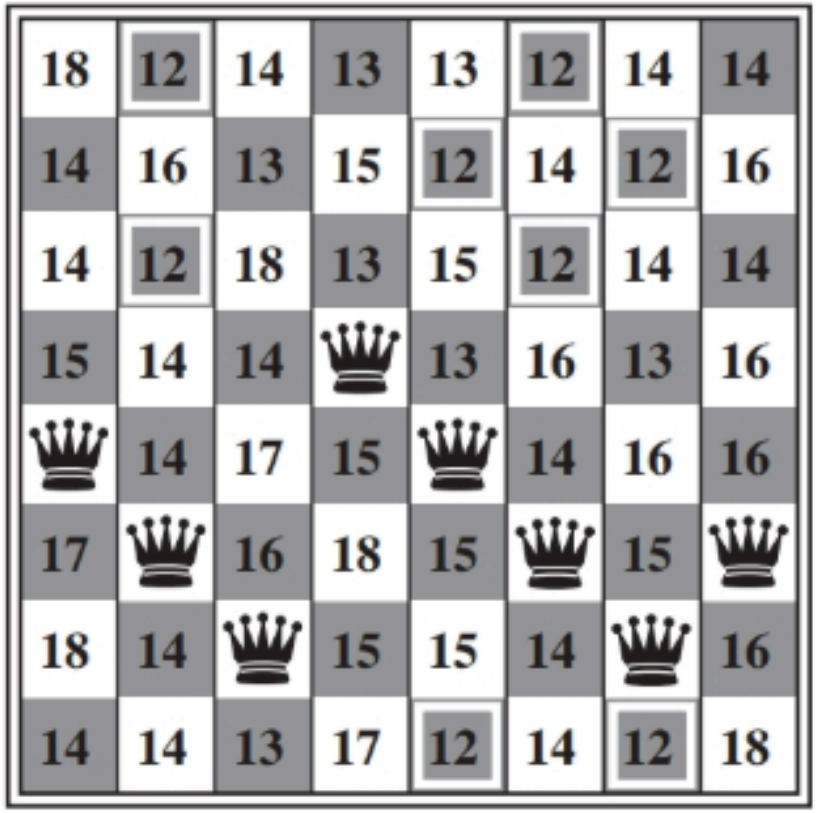
Local search and optimization problems

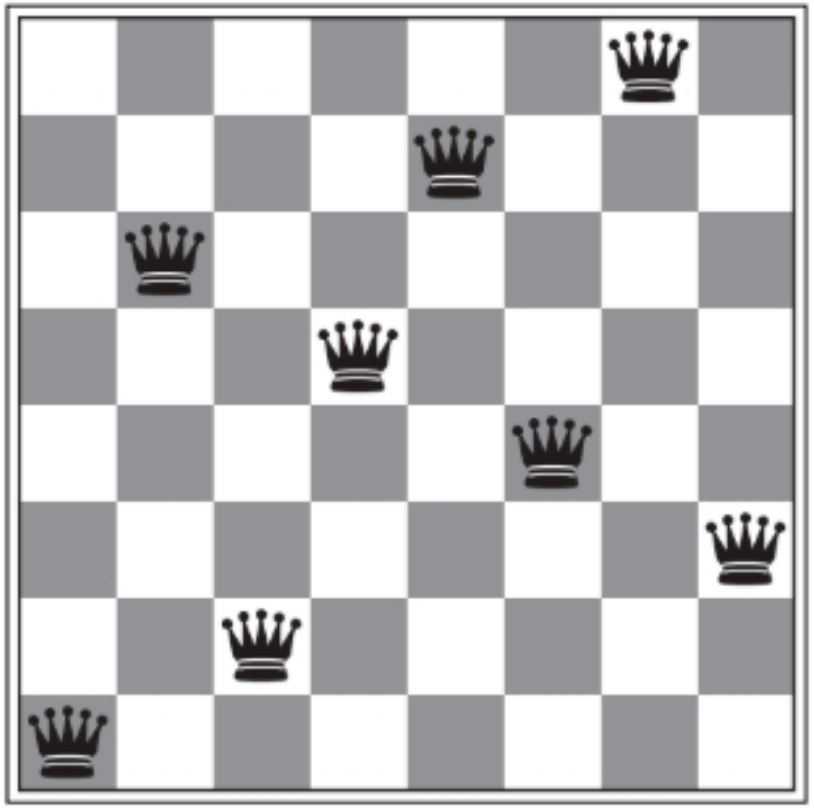
- In many optimization problems, path is irrelevant
  - The goal state itself is the solution
- Then state space = set of “complete” configurations
- In such cases, can use iterative improvement algorithms; keep a single “current” state, try to improve it

Hill climbing (gradient ascent/descent)



Greed hill-climbing search: example





- The **heuristic cost function h: the number of pairs of queens that are attacking each other, either directly or indirectly**.
- **The global minimum of this function is zero, which occurs only at perfect solutions.**
- **Left: a state with h = 17**. The figure also shows the values of all its successors, with the best successors having h=12.
- From the state on the left, it takes just five steps to reach **the state on the right, which has h = 1 and is *very nearly a solution*.**
- Unfortunately, hill climbing often gets stuck for the reasons shown in previous slide.

Local search solutions

- Key idea: add randomness
- Methods:
  - Stochastic hill-climbing
  - Random-restart hill-climbing
  - Simulated annealing (gradient descent)
  - Stochastic beam search
- Alternative approach: Genetic Algorithms (GAs)

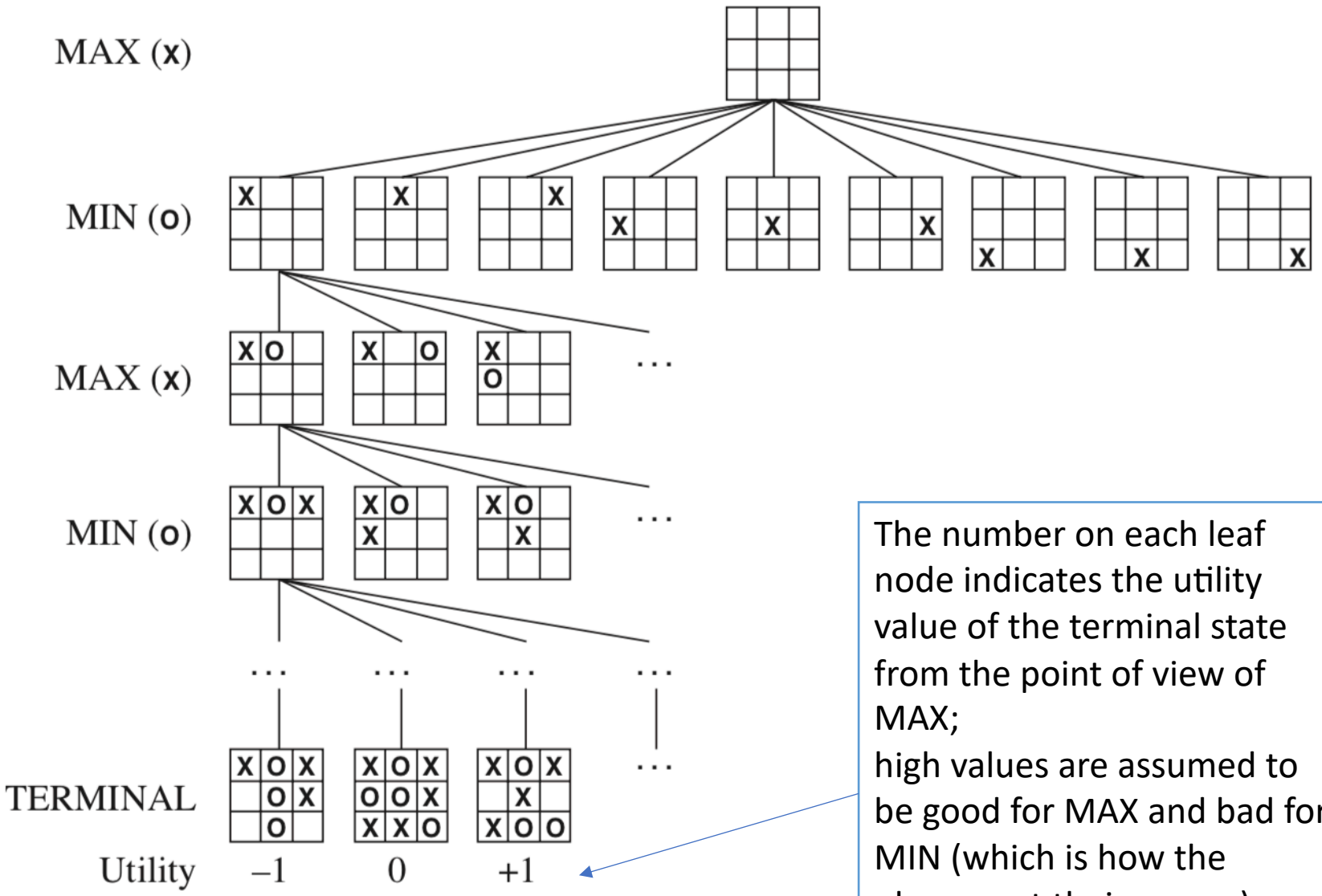
Search becomes harder when…

- … the environment becomes more complex:
  - Non-deterministic
  - Partially observable
  - Dynamic

Adversarial search and games

In which we examine the problems that arise when we try to plan ahead in a world where other agents are planning against us.

(Partial) game tree: example

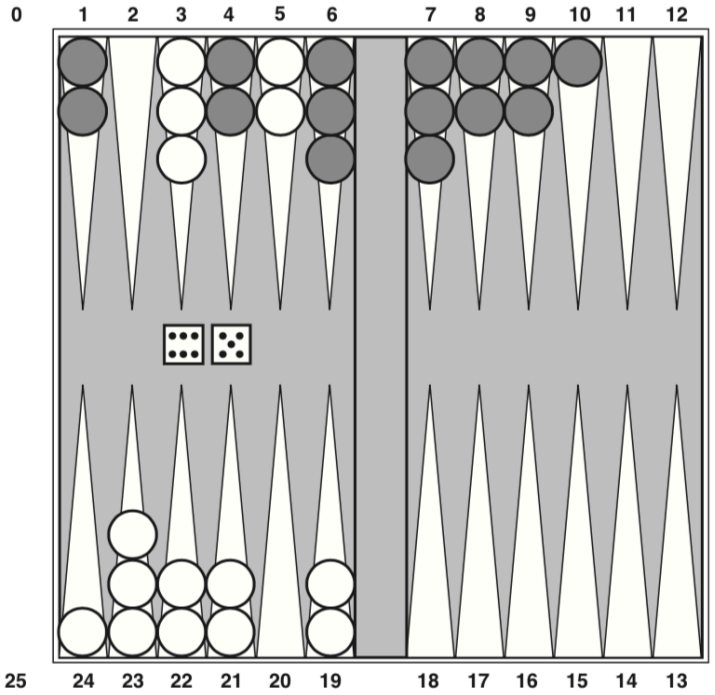


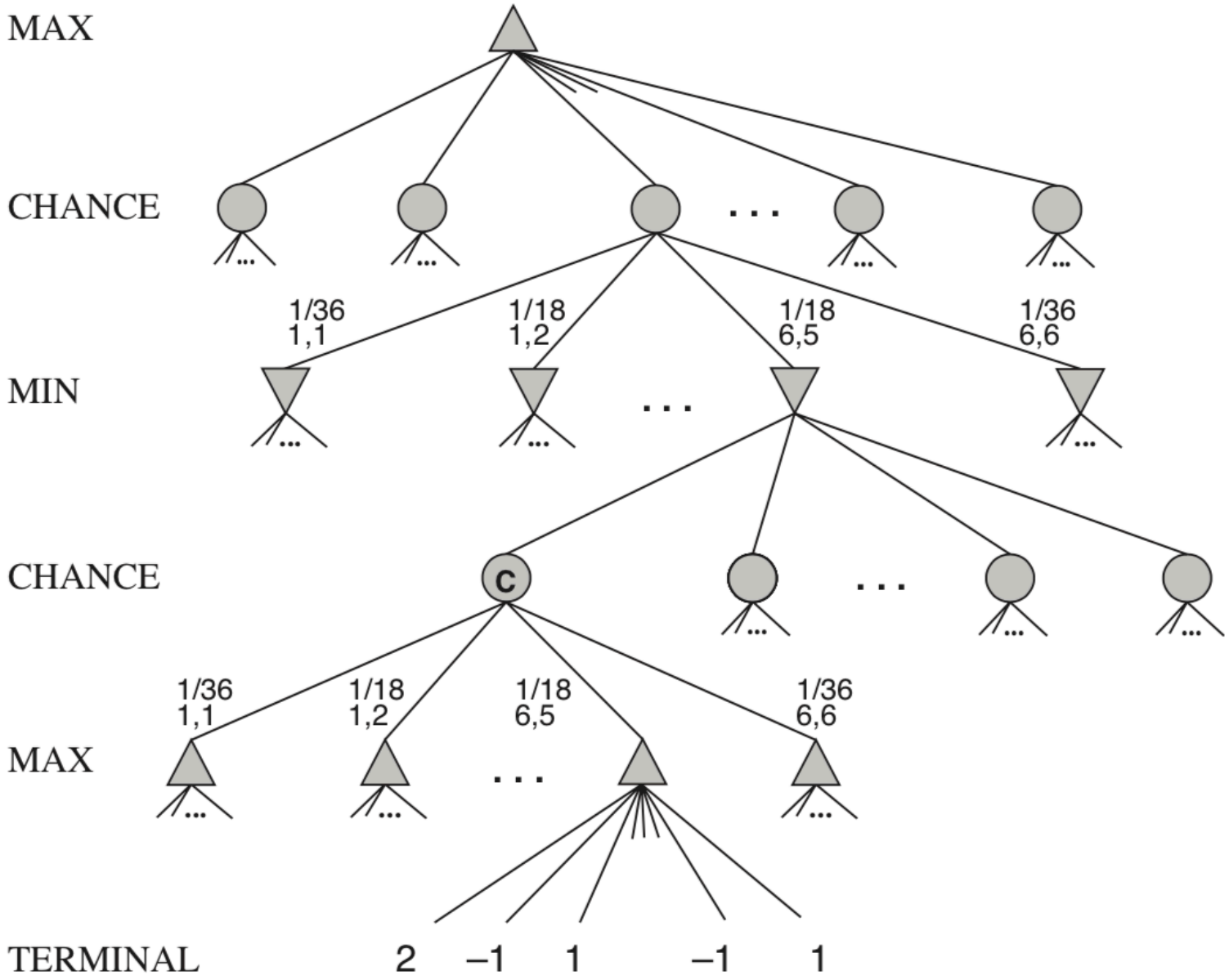
For tic-tac-toe the game tree is relatively small—fewer than 9! = 362,880 terminal nodes.

But for chess there are over 10^{40} nodes

The number on each leaf node indicates the utility value of the terminal state from the point of view of MAX;
high values are assumed to be good for MAX and bad for MIN (which is how the players get their names).

Stochastic games (e.g., backgammon)





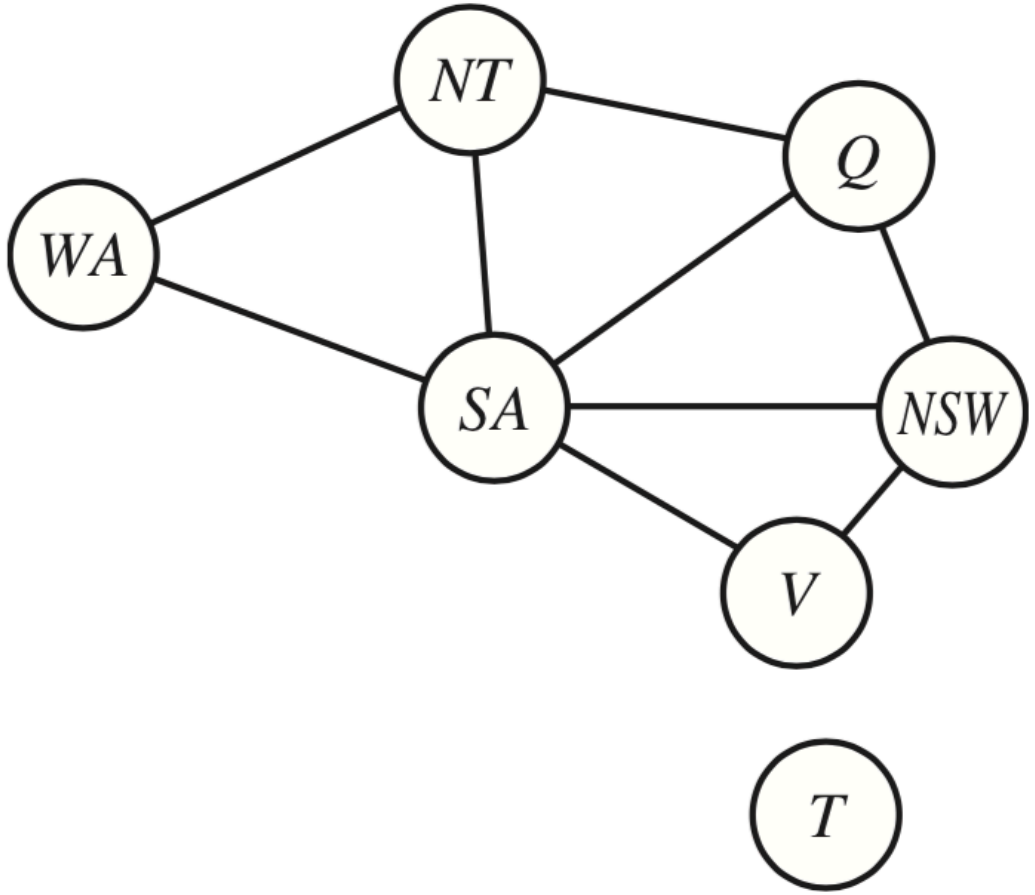
Sidebar: Why do AI researchers love games?

- Challenging problems, requiring non-trivial solutions
- Variable degrees of complexity
- Appealing to humans, recognizable by large audiences
- Brute-force is usually not the way… but having powerful computers help!
- Chance at fame, fortune, visibility
- Benchmarks: human experts and computer algorithms
- A clear chance at showing computers being better than humans.
  - How so, exactly?

Constraint satisfaction problems (CSPs)

In which we see how treating states as more than just little black boxes leads to the invention of a range of powerful new search methods and a deeper understanding of problem structure and complexity.

Example: map coloring

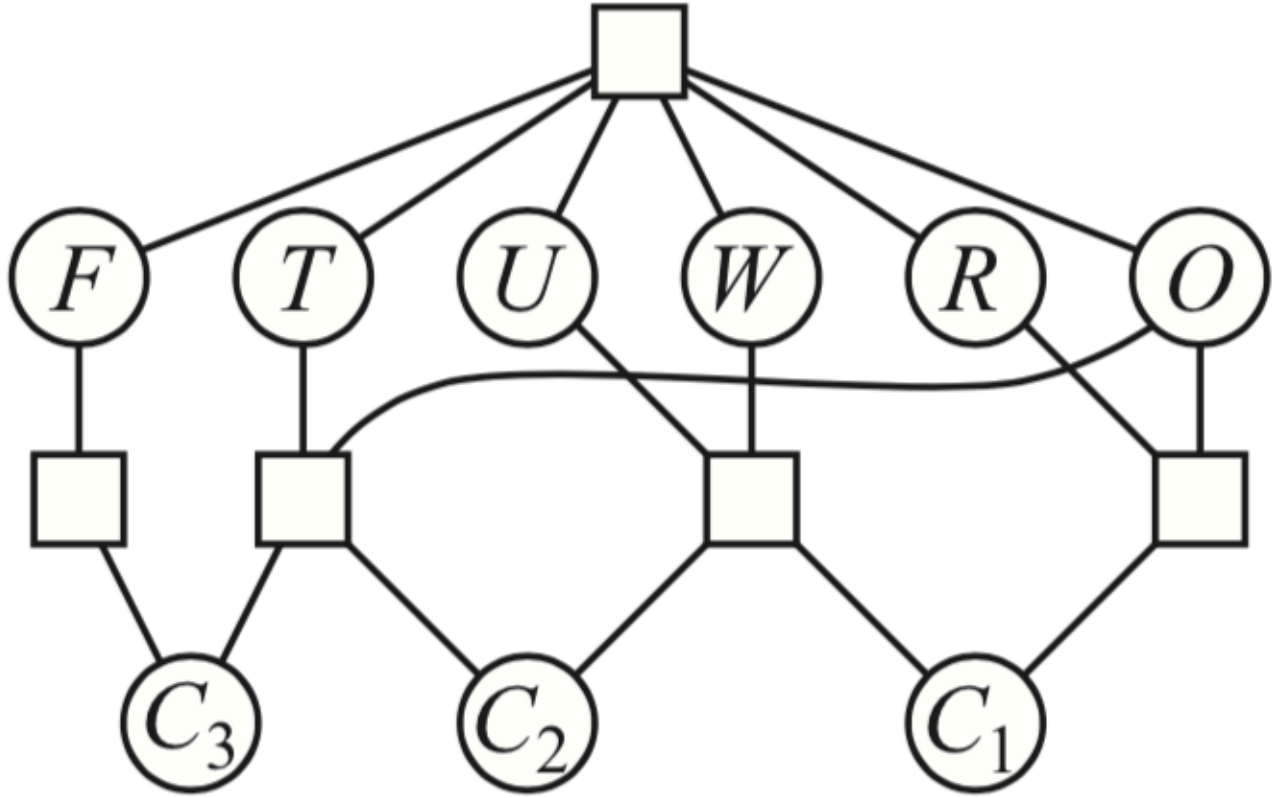


$$\{ WA = red, NT = green, Q = red, NSW = green, V = red, SA = blue, T = red \}$$

Example: cryptarithmetic

$$\begin{array}{r} T\ W\ O \\ +\ T\ W\ O \\ \hline F\ O\ U\ R \end{array}$$

$$\begin{aligned} O + O &= R + 10 \cdot C_{10} \\ C_{10} + W + W &= U + 10 \cdot C_{100} \\ C_{100} + T + T &= O + 10 \cdot C_{1000} \\ C_{1000} &= F\ , \end{aligned}$$



Example: sudoku

	1	2	3	4	5	6	7	8	9
A			3		2		6		
B	9			3		5			1
C			1	8		6	4		
D			8	1		2	9		
E	7								8
F			6	7		8	2		
G			2	6		9	5		
H	8			2		3			9
I			5		1		3		

	1	2	3	4	5	6	7	8	9
A	4	8	3	9	2	1	6	5	7
B	9	6	7	3	4	5	8	2	1
C	2	5	1	8	7	6	4	9	3
D	5	4	8	1	3	2	9	7	6
E	7	2	9	5	6	4	1	3	8
F	1	3	6	7	9	8	2	4	5
G	3	7	2	6	8	9	5	1	4
H	8	1	4	2	5	3	7	6	9
I	6	9	5	4	1	7	3	8	2

Alldiff constraints: one for each row, column, and box

$$Alldiff(A1, A2, A3, A4, A5, A6, A7, A8, A9)$$
$$Alldiff(B1, B2, B3, B4, B5, B6, B7, B8, B9)$$
$$\ldots$$
$$Alldiff(A1, B1, C1, D1, E1, F1, G1, H1, I1)$$
$$Alldiff(A2, B2, C2, D2, E2, F2, G2, H2, I2)$$
$$\ldots$$
$$Alldiff(A1, A2, A3, B1, B2, B3, C1, C2, C3)$$
$$Alldiff(A4, A5, A6, B4, B5, B6, C4, C5, C6)$$
$$\ldots$$

Logical Agents

In which we design agents that can form representations of a complex world, use a process of inference to derive new representations about the world, and use these new representations to deduce what to do.

Logical agents

- Propositional logic
  - A BNF (Backus–Naur Form) grammar of sentences in propositional logic, along with operator precedences, from highest to lowest.

$$
\begin{aligned}
\mathit{Sentence} &\rightarrow \mathit{AtomicSentence} \mid \mathit{ComplexSentence} \\
\mathit{AtomicSentence} &\rightarrow \mathit{True} \mid \mathit{False} \mid P \mid Q \mid R \mid \ldots \\
\mathit{ComplexSentence} &\rightarrow (\ \mathit{Sentence}\ ) \mid [\ \mathit{Sentence}\ ] \\
&\mid \neg\ \mathit{Sentence} \\
&\mid \mathit{Sentence} \wedge \mathit{Sentence} \\
&\mid \mathit{Sentence} \vee \mathit{Sentence} \\
&\mid \mathit{Sentence} \Rightarrow \mathit{Sentence} \\
&\mid \mathit{Sentence} \Leftrightarrow \mathit{Sentence} \\
\text{OPERATOR PRECEDENCE} &: \neg, \wedge, \vee, \Rightarrow, \Leftrightarrow
\end{aligned}
$$

Logical agents

- Propositional logic
  - Truth tables for five logical connectives

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
false	*false*	*true*	*false*	*false*	*true*	*true*
false	*true*	*true*	*false*	*true*	*true*	*false*
true	*false*	*false*	*false*	*true*	*false*	*false*
true	*true*	*false*	*true*	*true*	*true*	*true*

Logical agents

- Propositional logic
  - Syntax
  - Semantics
  - Knowledge Base (KB)
  - Inference procedure

First-order logic

- Motivation
  - Propositional logic lacks the expressive power to concisely describe an environment with many objects.
- Idea:
  - Combine the best of formal and natural languages

- The language of first-order logic is built around objects and relations.
- It has been so important to mathematics, philosophy, and artificial intelligence precisely because those fields—and indeed, much of everyday human existence—can be usefully thought of as **dealing with objects and the relations among them**.
- First-order logic can also express facts about some or all of the objects in the universe. This enables one to represent **general laws or rules** …

Ontological and epistemological commitments

Language	Ontological Commitment (What exists in the world)	Epistemological Commitment (What an agent believes about facts)
Propositional logic	facts	true/false/unknown
First-order logic	facts, objects, relations	true/false/unknown
Temporal logic	facts, objects, relations, times	true/false/unknown
Probability theory	facts	degree of belief $\in [0, 1]$
Fuzzy logic	facts with degree of truth $\in [0, 1]$	known interval value

First-order logic: example of a model

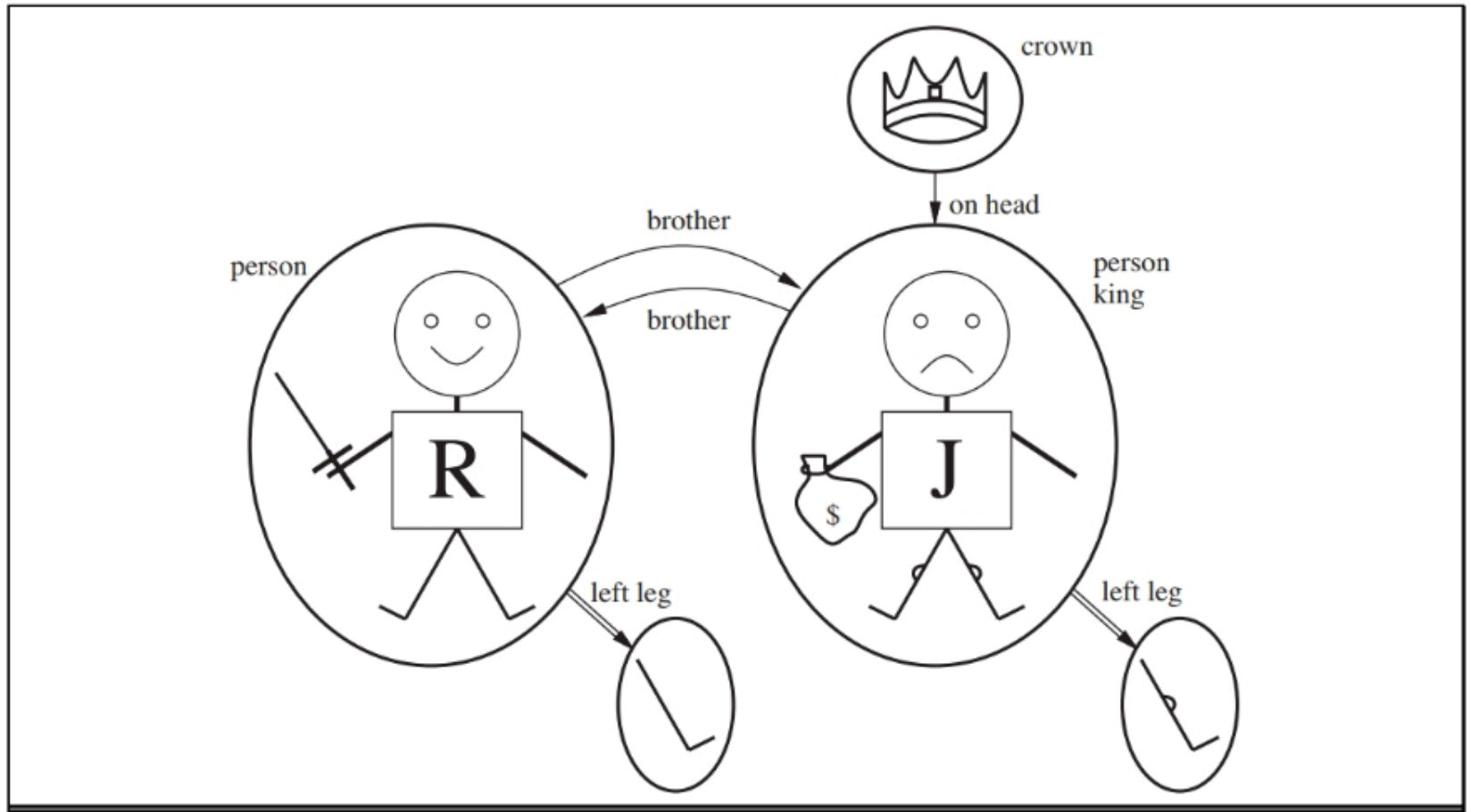


Figure 8.2 A model containing five objects, two binary relations, three unary relations (indicated by labels on the objects), and one unary function, left-leg.

First-order logic: syntax

$$
\begin{aligned}
\textit{Sentence} &\rightarrow \textit{AtomicSentence} \mid \textit{ComplexSentence} \\
\textit{AtomicSentence} &\rightarrow \textit{Predicate} \mid \textit{Predicate}(\textit{Term}, \ldots) \mid \textit{Term} = \textit{Term} \\
\textit{ComplexSentence} &\rightarrow (\ \textit{Sentence}\ ) \mid [\ \textit{Sentence}\ ] \\
&\mid \neg\ \textit{Sentence} \\
&\mid \textit{Sentence} \wedge \textit{Sentence} \\
&\mid \textit{Sentence} \vee \textit{Sentence} \\
&\mid \textit{Sentence} \Rightarrow \textit{Sentence} \\
&\mid \textit{Sentence} \Leftrightarrow \textit{Sentence} \\
&\mid \textit{Quantifier}\ \textit{Variable}, \ldots\ \textit{Sentence} \\
\textit{Term} &\rightarrow \textit{Function}(\textit{Term}, \ldots) \\
&\mid \textit{Constant} \\
&\mid \textit{Variable} \\
\textit{Quantifier} &\rightarrow \forall \mid \exists \\
\textit{Constant} &\rightarrow A \mid X_1 \mid \textit{John} \mid \cdots \\
\textit{Variable} &\rightarrow a \mid x \mid s \mid \cdots \\
\textit{Predicate} &\rightarrow \textit{True} \mid \textit{False} \mid \textit{After} \mid \textit{Loves} \mid \textit{Raining} \mid \cdots \\
\textit{Function} &\rightarrow \textit{Mother} \mid \textit{LeftLeg} \mid \cdots \\
\text{OPERATOR PRECEDENCE} &: \neg, =, \wedge, \vee, \Rightarrow, \Leftrightarrow
\end{aligned}
$$

Inference in first-order logic

- Suppose our knowledge base contains the standard folkloric axiom stating that all greedy kings are evil:

$$\forall x \;\; King(x) \wedge Greedy(x) \;\Rightarrow\; Evil(x)$$

- Then it seems quite permissible to infer any of the following sentences:

$$King(John) \wedge Greedy(John) \;\Rightarrow\; Evil(John)$$
$$King(Richard) \wedge Greedy(Richard) \;\Rightarrow\; Evil(Richard)$$
$$King(Father(John)) \wedge Greedy(Father(John)) \;\Rightarrow\; Evil(Father(John))$$
$$\vdots$$