香港浸會大學
HONG KONG BAPTIST UNIVERSITY

DEPARTMENT OF
COMPUTER SCIENCE
計算機科學系

COMP7015 Artificial Intelligence

Welcome to COMP7015!

Instructor: Dr. Kejing Yin

September 8, 2022

Instructor and TAs

Dr. Kejing Yin

Instructor

cskjyin@comp.hkbu.edu.hk

AI in healthcare,

Machine Learning & Data Mining

Mr. Zhenheng Tang

Teaching Assistant

zhtang@comp.hkbu.edu.hk

Federated Learning,

Distributed Deep Learning

Mr. Ruiqi Li

Teaching Assistant

csrqli@comp.hkbu.edu.hk

Computer Vision,

Neural Rendering

COMP7015 Logistics

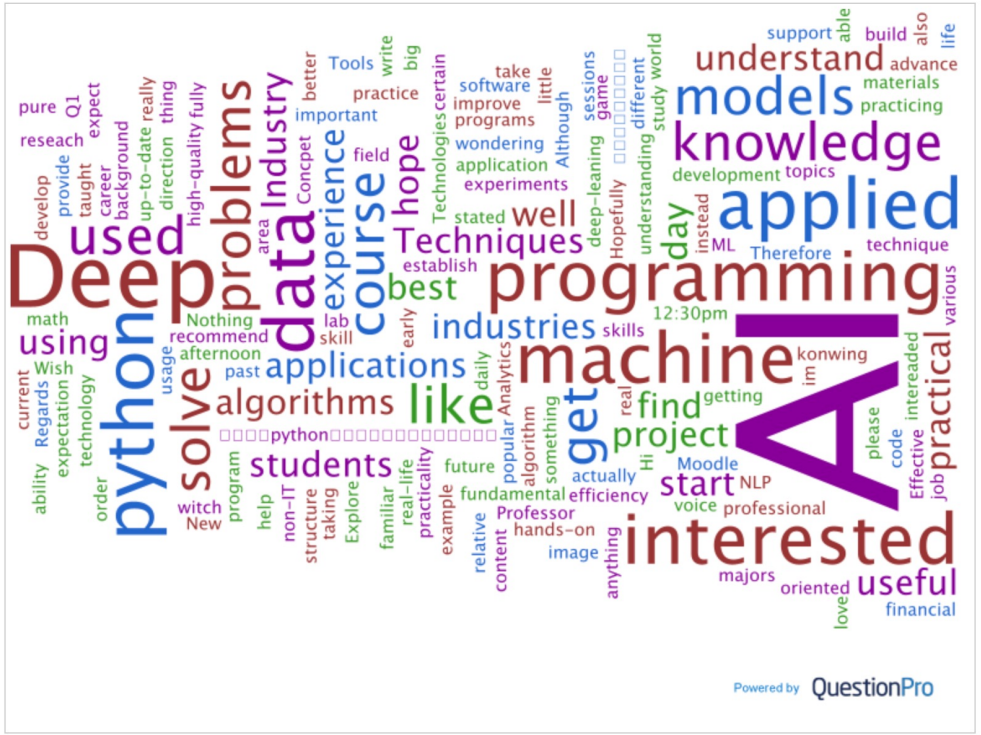
- Course Info Sheet: available in Moodle.
- Moodle for learning materials and public communications. https://buelearning.hkbu.edu.hk/course/view.php?id=98572

- Office Hours (Q&A for lectures):
  - Every Saturday 2 – 4 pm via zoom (link in info sheet)
  - For lab days: postponed to the following Monday.
  - Other time: Email for appointment

COMP7015 Logistics

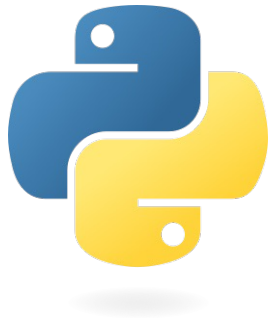
- Labs:
  - Help you gain hands-on experiences
  - Prepare you for the course project
  - Five sessions on Saturday pm:
    - Sep. 17 (Python tutorial; optional)
    - Sep. 24, Oct. 22, Oct. 29, and Nov. 5
  - Two sections (80 PCs each section):
    - Section registration will open later

6. [Q6] What is your expectation for t...



COMP7015 Logistics

- Python Programming
  - Mainstream programming language for AI
  - We provide a tutorial (Lab 0)
  - Other resources:
    - https://www.w3schools.com/PYTHON/
    - https://www.youtube.com/watch?v=rfscVS0vtbw
  - Practice makes perfect.



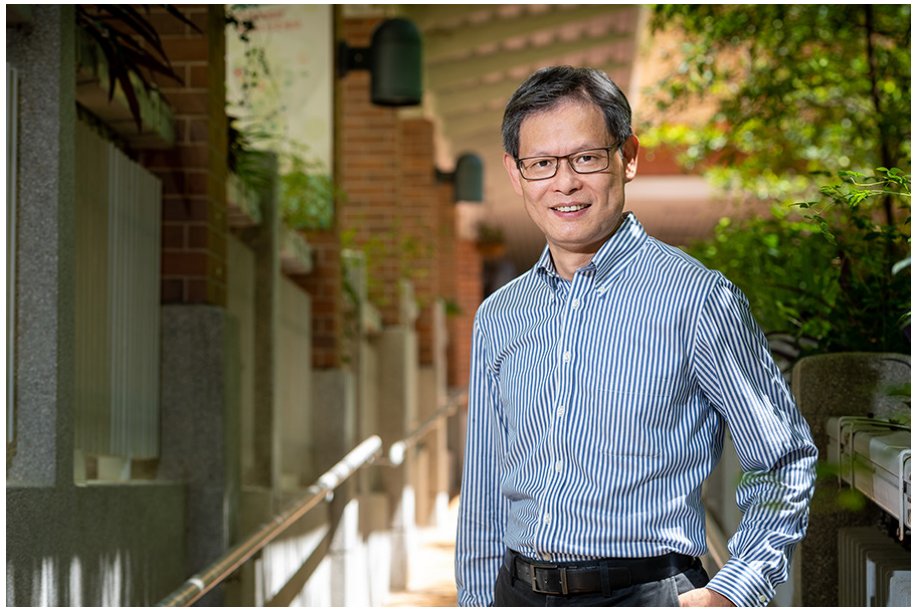
COMP7015 Logistics

- Assessment
  - Four assignments — 20% (*two written & two programming*)
  - In-class Quiz — 5% (*Oct. 6*)
  - Mini-project — 25%
  - Examination — 50% (*score at least 30 to pass the course*)
    (*this is not an easy course, need to work hard*)

- Late policy
  - 10% penalty for each 12 hours late.
  - Late submissions exceeding two days will receive zero scores.
  - No late submission for course project.

COMP7015 Logistics

- Invited Talk by Prof. Jiming Liu (6:30pm, Sep. 15)



Prof. Jiming Liu (*Fellow, IEEE*)

Chair Professor in Computer Science

Dean of Faculty of Science

香港浸會大學
HONG KONG BAPTIST UNIVERSITY

DEPARTMENT OF
COMPUTER SCIENCE
計算機科學系

COMP7015 Artificial Intelligence

Lecture 1: Introduction to AI & Search I

Instructor: Dr. Kejing Yin

September 8, 2022

outline

- What is artificial intelligence (AI)?
- What are we going to learn in this course?
- History, current status and future of AI
- Searching Algorithm Part I

What is artificial intelligence (AI)?

What comes to your mind when you hear "AI"?

What is your definition of AI?
What are the applications of AI?
You can submit as many times as you like

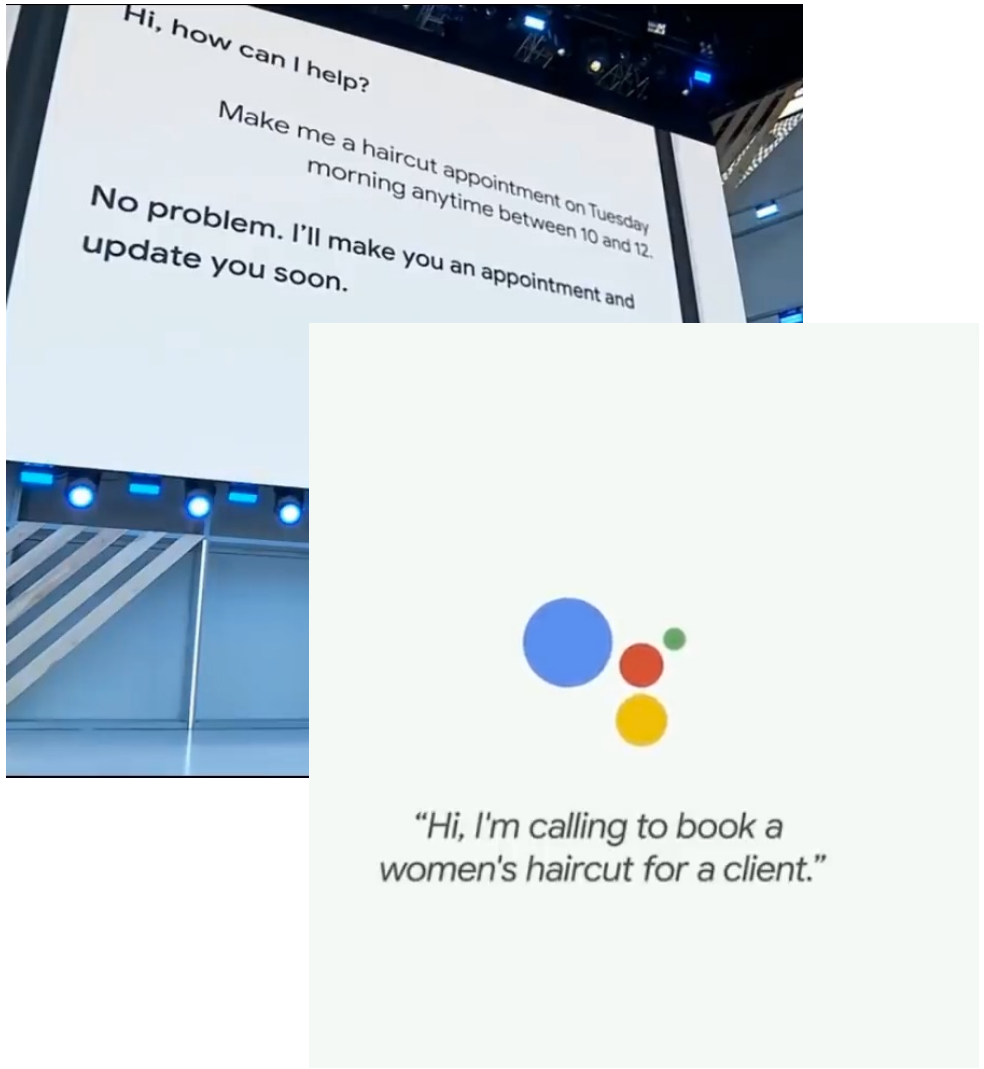
What is artificial intelligence (AI)?

Let's look at some real examples of AI

Digital Assistants

Google CEO Sundar demonstrating Google Duplex

Digital Assistants: Behind the Scenes



Voice to text

Natural language understanding

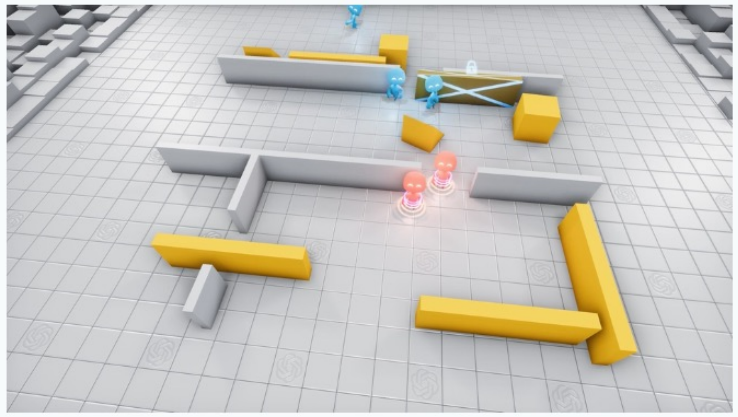
Natural language generation

Text to speech (*sounding natural*)

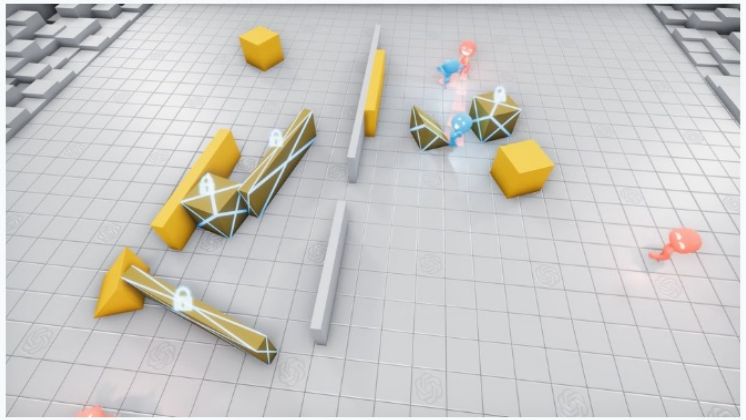
Hide and Seek Games

https://openai.com/blog/emergent-tool-use/

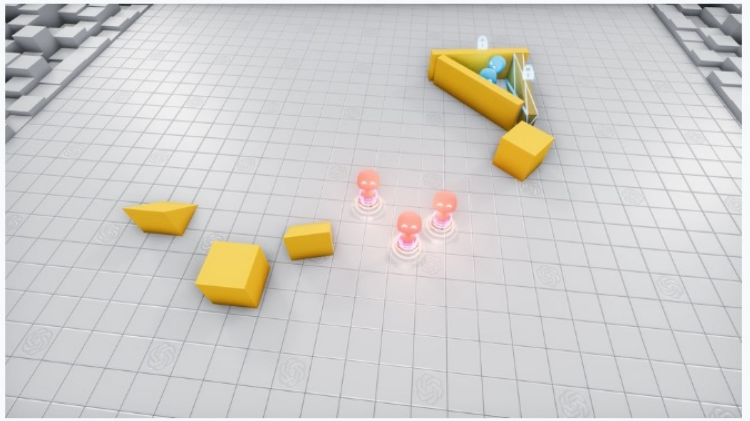
Hide and Seek Games: Behind the Scenes



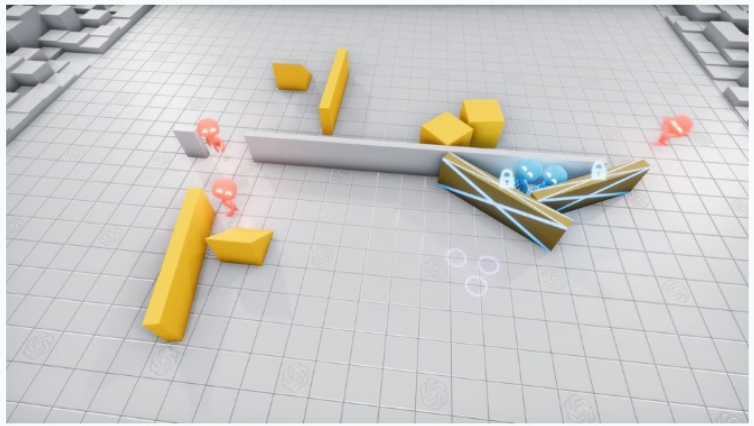
Episode 0
Random The agents move randomly.



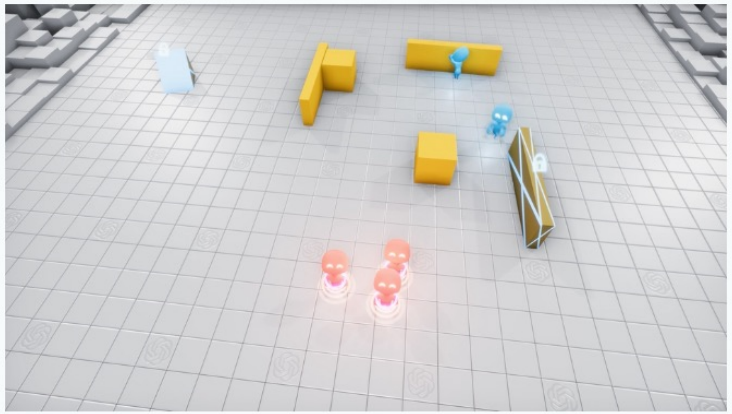
Episodes 0–22 million
Chasing Seekers learn to chase hiders.



Episodes 22–88 million
Shelter Construction Hiders learn to construct a shelter to hide in.



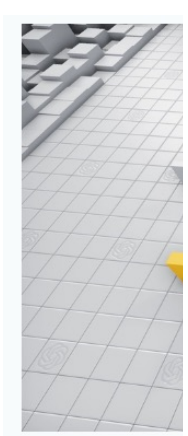
Episodes 88–115 million
Ramp Use Seekers learn to use ramps to jump over obstacles.



Episodes 115–388 million
Ramp Defense Hiders learn to lock the ramps to prevent seekers from using them.

Self-play

Hide and Seek Games: Behind the Scenes

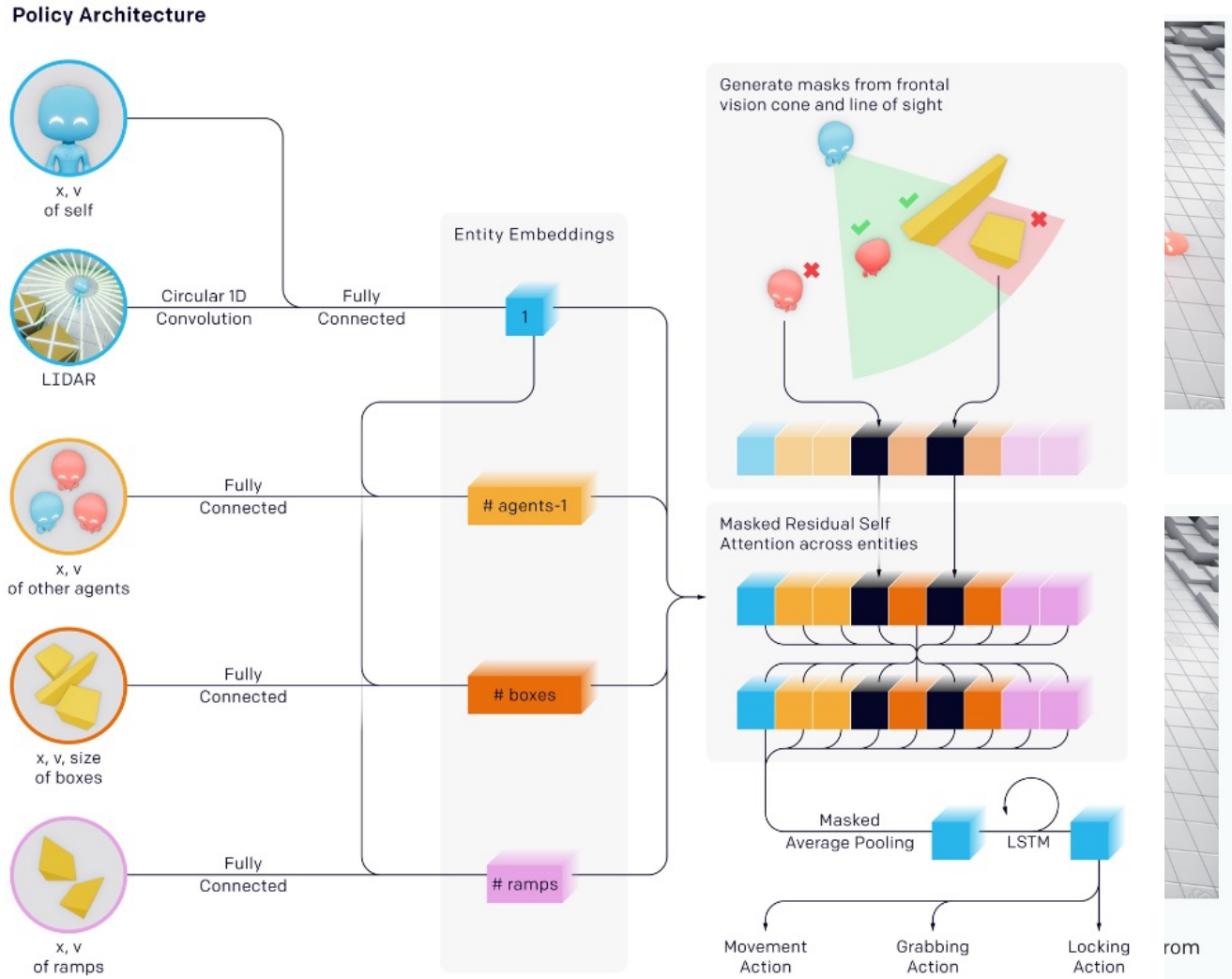


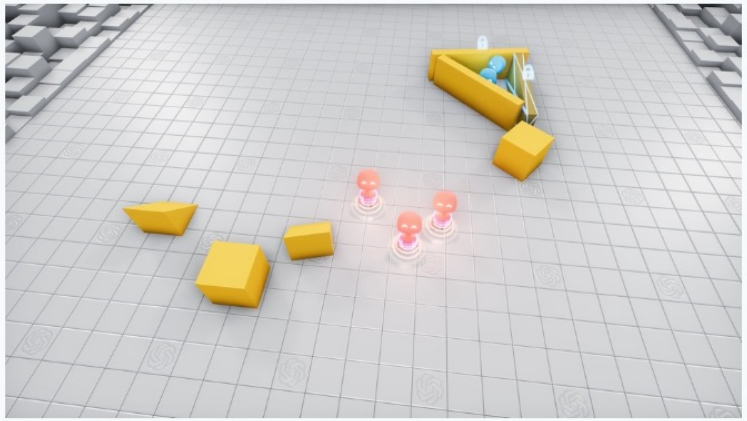
Episode 0

Random The ag

Episodes 88–115 millio

Ramp Use Seek





Episodes 22–88 million

Shelter Construction Hiders learn to construct a shelter to hide in.

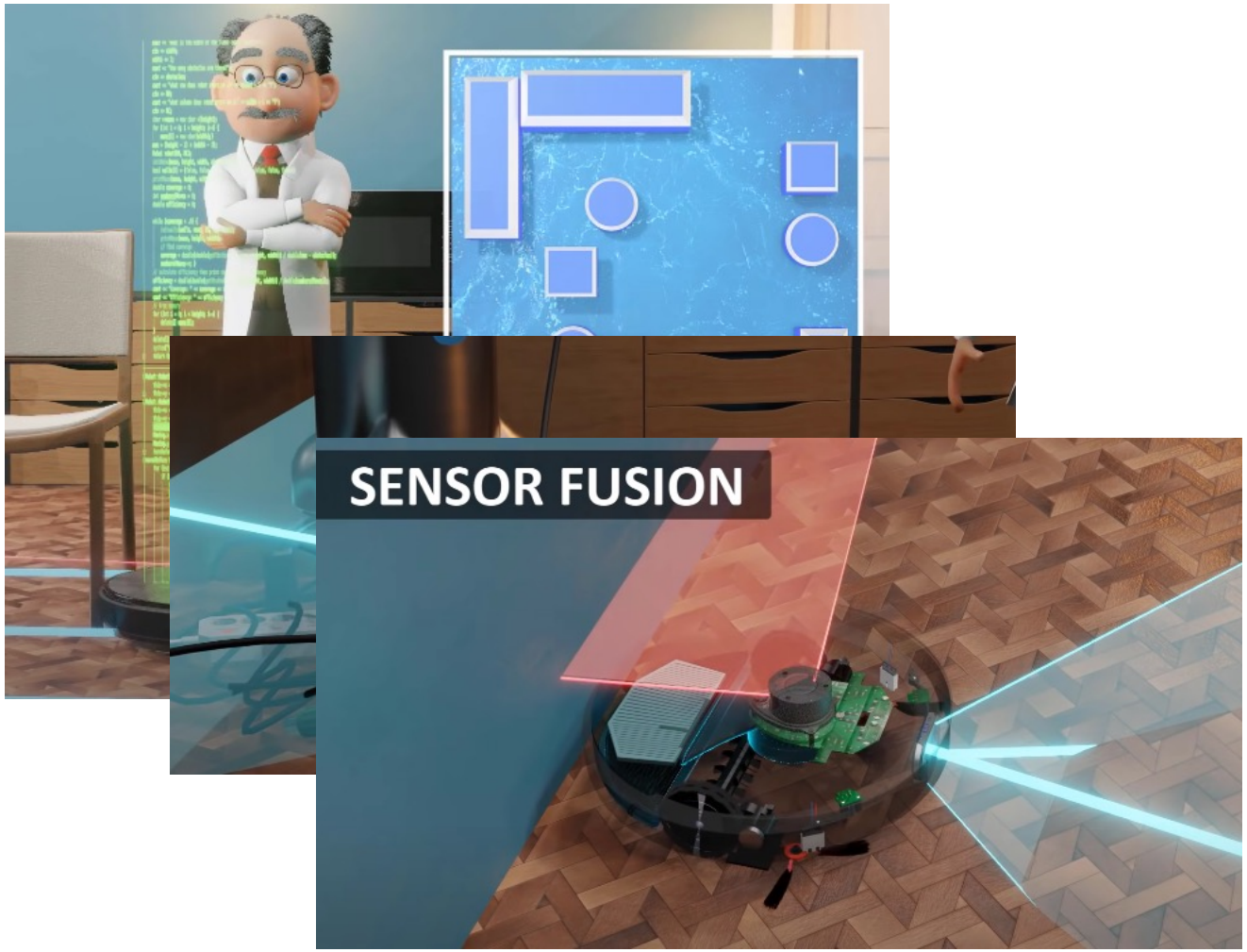
Self-play

Reinforcement learning

Cleaner Robots

Source: https://www.youtube.com/watch?v=hoY2YxLGV98

Cleaner Robots: Behind the Scenes



Signal processing

2D map construction

Path planning

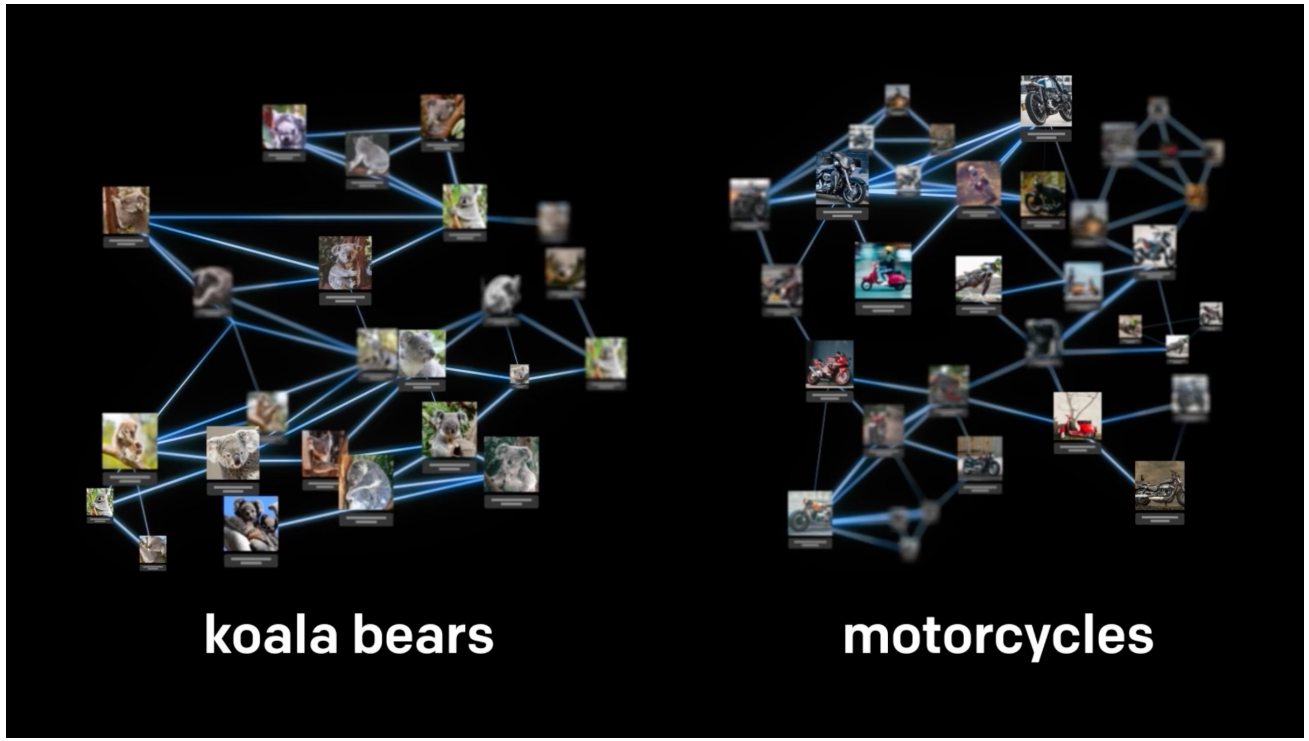
Computer vision

Sensor fusion

Generating Images from Text

Demo of DALL·E 2 System: https://openai.com/dall-e-2/#demos

Generating Images from Text: Behind the Scenes

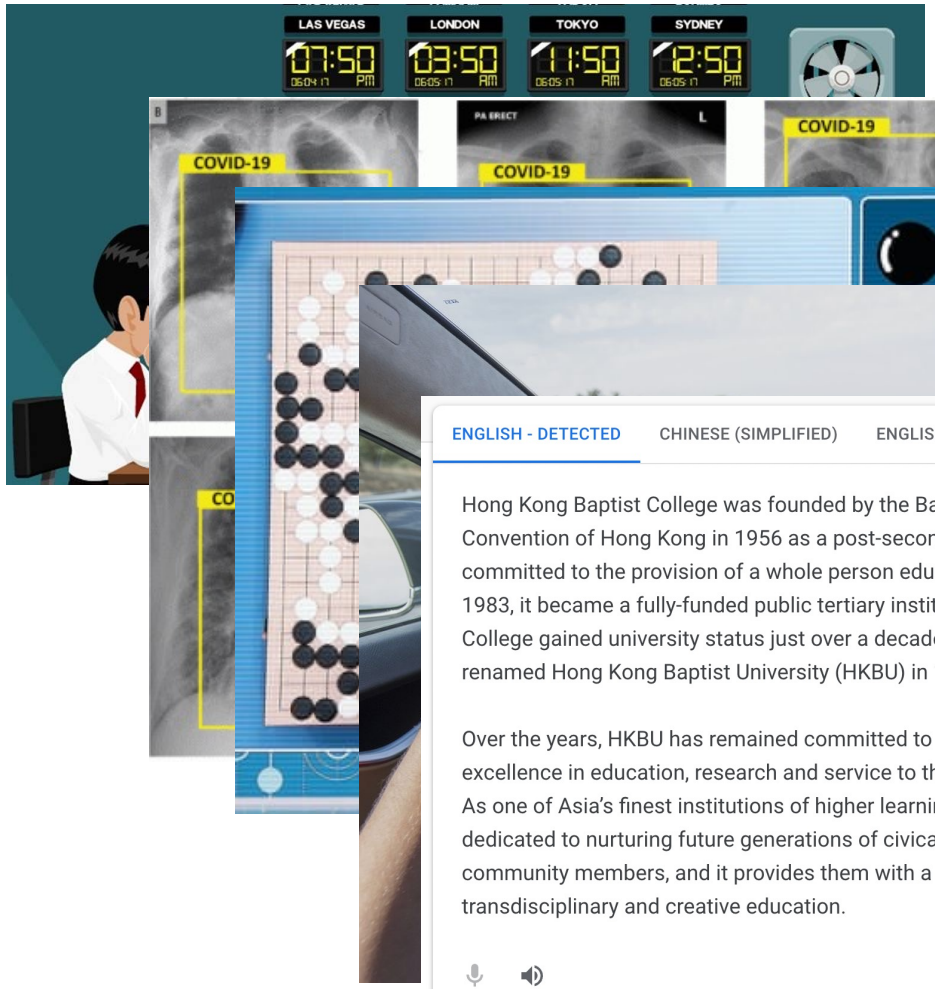


Millions of labeled images

Supervised learning

Contrastive learning

Other Examples

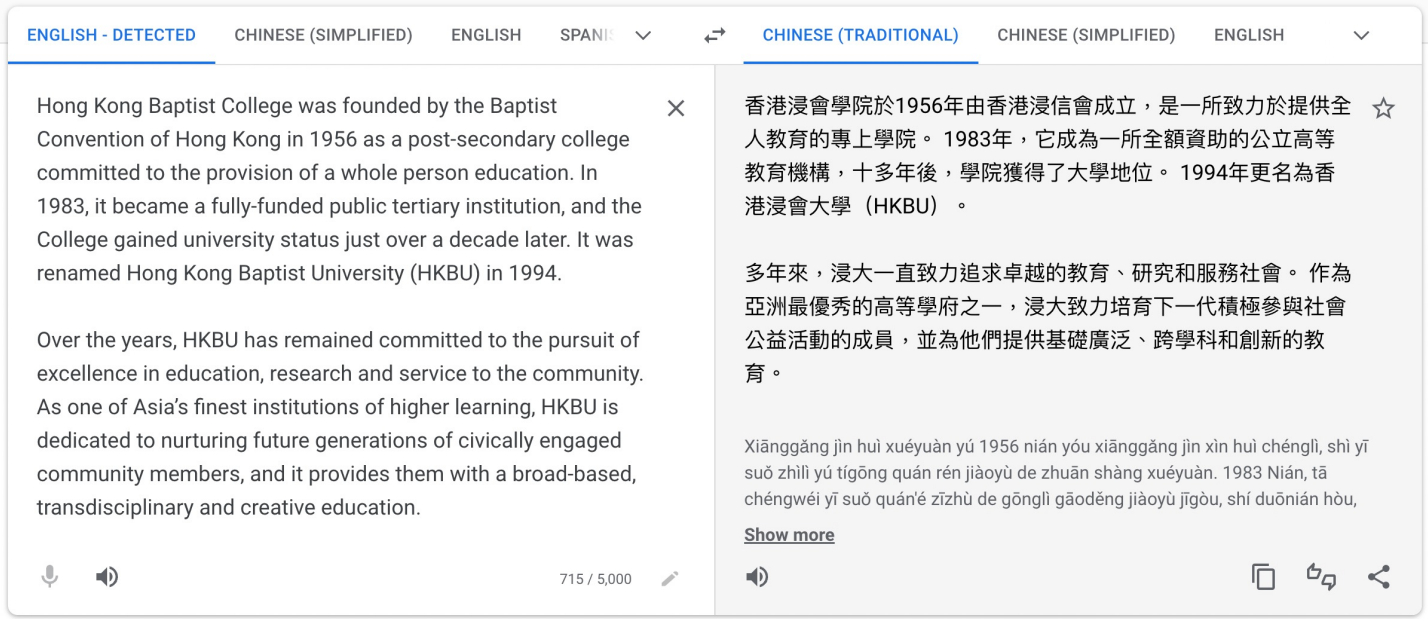


Algorithm trading

Medical imaging

AlphaGo

Self-driving vehicles



Machine translation

Images credits: https://blog.quantinsti.com/algorithmic-trading-finance-grads-fundamental-traders/, https://www.itnonline.com/content/ai-fight-against-covid-19-automatic-detection-chest-x-ray-images-possible-reports-Incheon, https://www.theguardian.com/technology/2017/may/23/alphago-google-ai-beats-ke-jie-china-go, https://www.bloomberg.com/news/articles/2022-08-21/musk-says-tesla-full-service-driving-price-will-rise-to-15-000

Come back to our question: What is artificial intelligence (AI)?

> Artificial intelligence leverages computers and machines to mimic the problem-solving and decision-making capabilities of the human mind.
>
> -- IBM's definition of AI

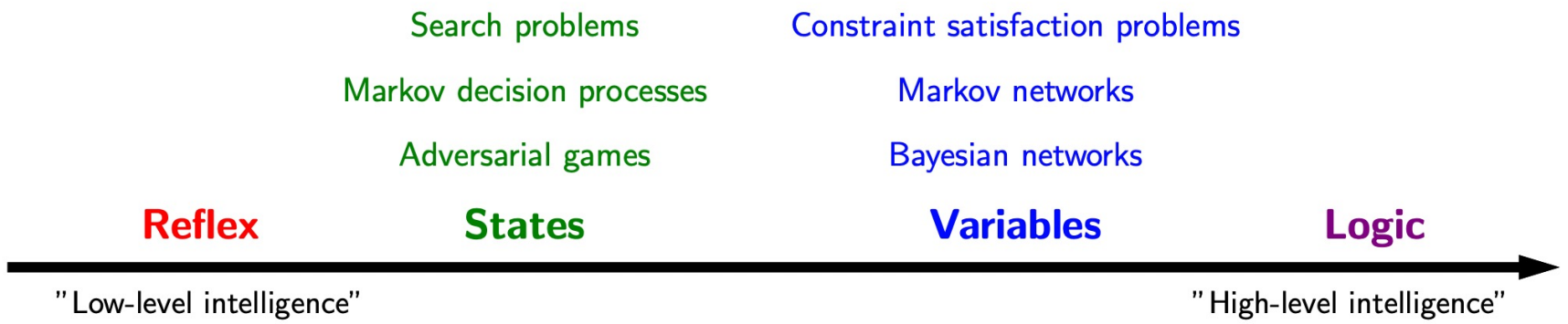
https://www.ibm.com/cloud/learn/what-is-artificial-intelligence

outline

- What is artificial intelligence (AI)?
- What are we going to learn in this course?
- History, current status and future of AI
- Searching Algorithm Part I

What are we going to learn in this course?

- AI covers a broad range of models and complementary approaches.

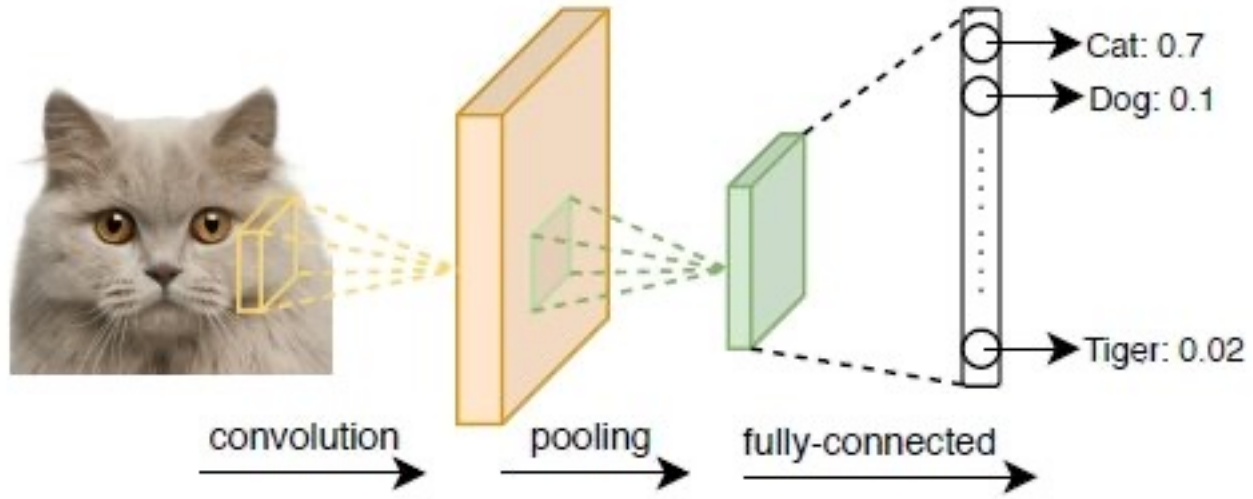


- Reflex-based: perform a fixed sequence of computations on an input, fully feed-forward, "mimic human reflex".
- *Examples: most models in machine learning.*

CS221 slides: https://stanford-cs221.github.io/spring2022-extra/modules/general/first-lecture.pdf

What are we going to learn in this course?

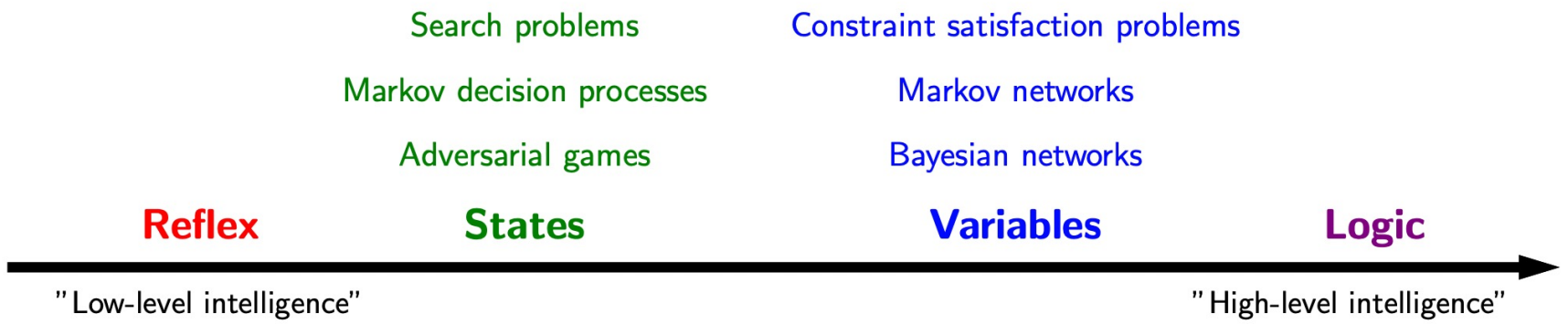
- *Example of Reflex-based models: Convolutional Neural Network*



You can recognize this is a cat in split second.

What are we going to learn in this course?

- AI covers a broad range of models and complementary approaches.



- State-based: state (e.g., current position) & action (e.g., where to go next)
- *Examples: motion planning in robots, Chess game, Go game, StarCraft, etc.*

CS221 slides: https://stanford-cs221.github.io/spring2022-extra/modules/general/first-lecture.pdf

What are we going to learn in this course?

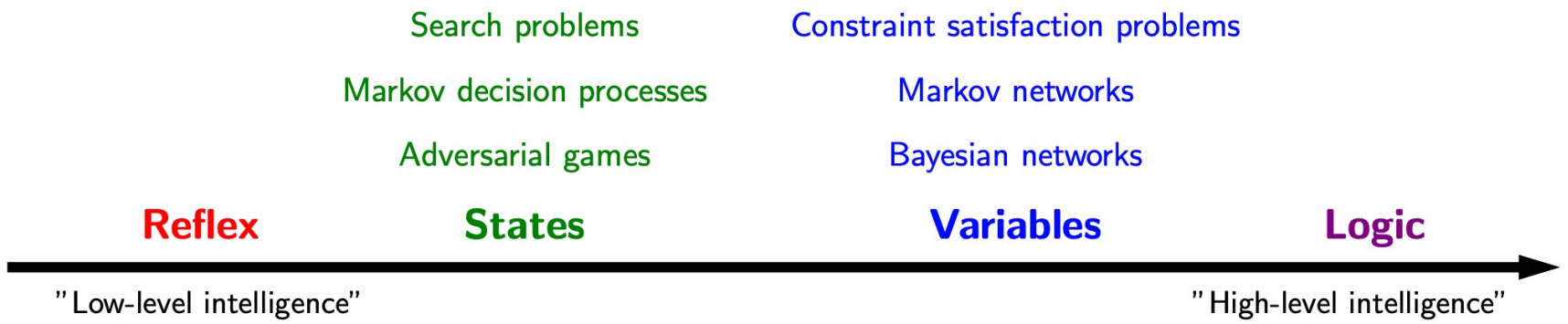
- *Example of state-based models: The chess game*

Harder than recognizing the cat.

Need to consider the current state and the step-by-step actions.

What are we going to learn in this course?

- AI covers a broad range of models and complementary approaches.

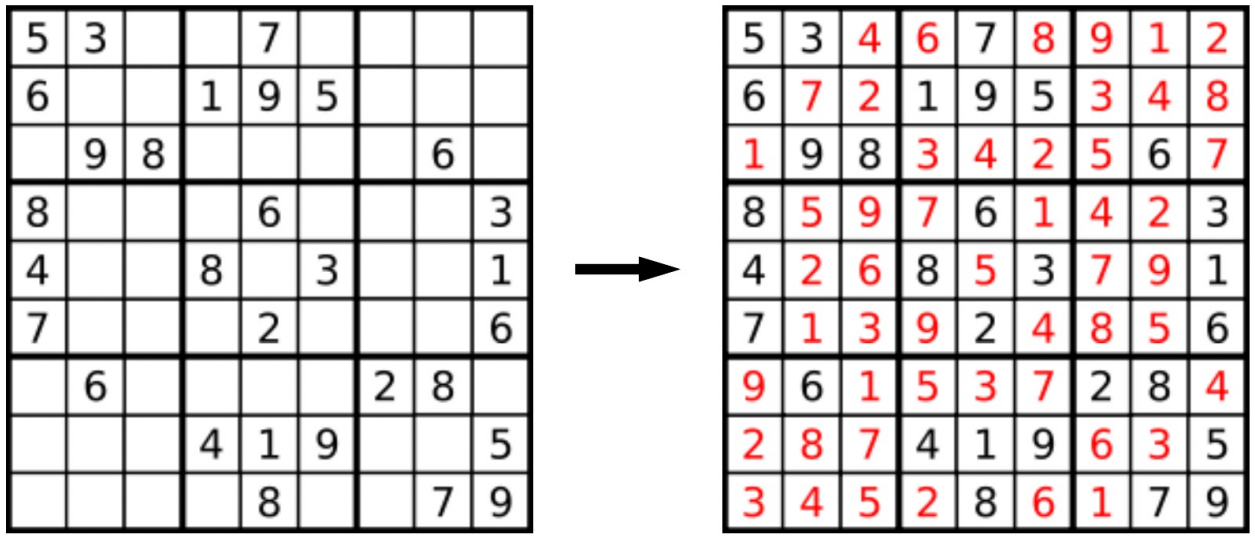


- Variable-based: the final goal state matters, but the action order does not.
- *Examples: Sudoku, class scheduling*

CS221 slides: https://stanford-cs221.github.io/spring2022-extra/modules/general/first-lecture.pdf

What are we going to learn in this course?

- *Example of variable-based models: Sudoku*

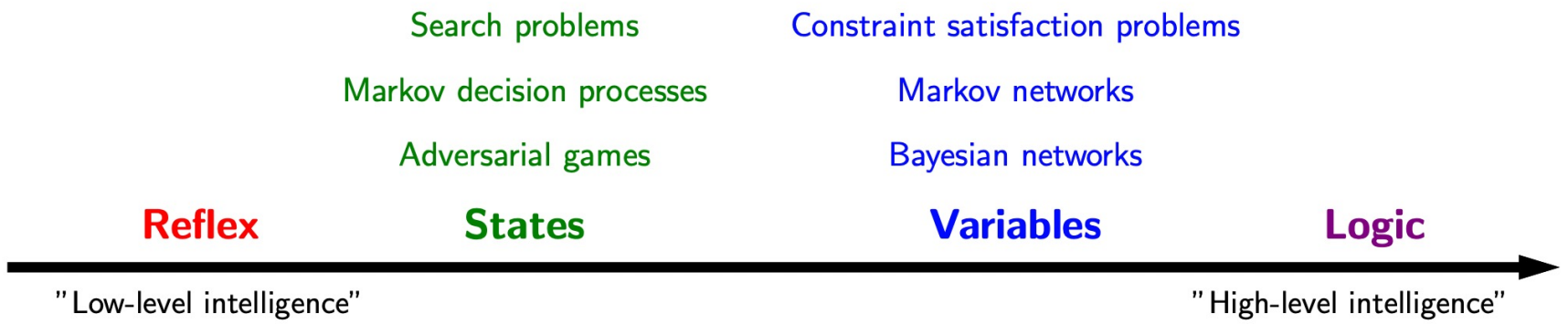


The order of filling in numbers does not matter, as long as the uniqueness constraints are satisfied.

State-based models (e.g., searching) is very inefficient.

What are we going to learn in this course?

- AI covers a broad range of models and complementary approaches.

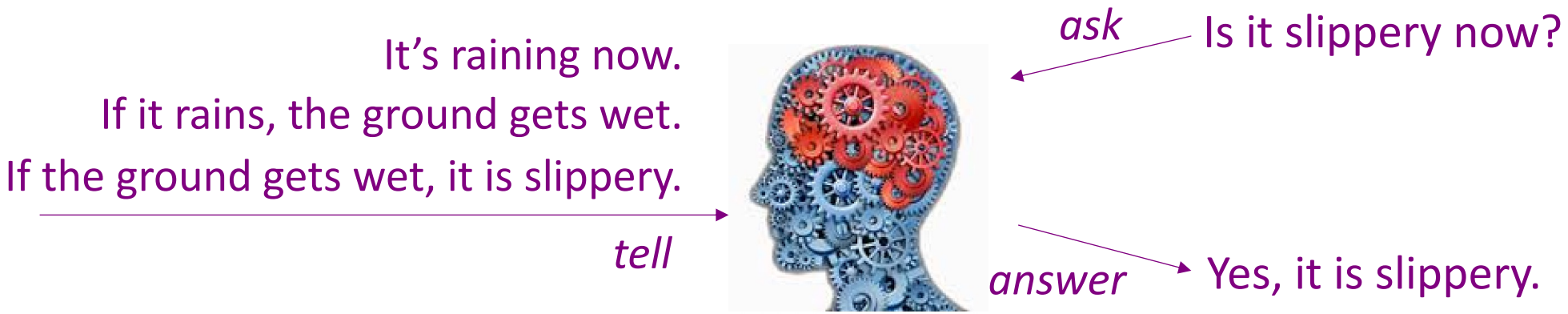


- Logic-based: representing and storing knowledge and reasoning with the knowledge.
- *Examples: virtual assistant*

CS221 slides: https://stanford-cs221.github.io/spring2022-extra/modules/general/first-lecture.pdf

What are we going to learn in this course?

- *Example of logic-based models: The virtual assistant*



Understand the information

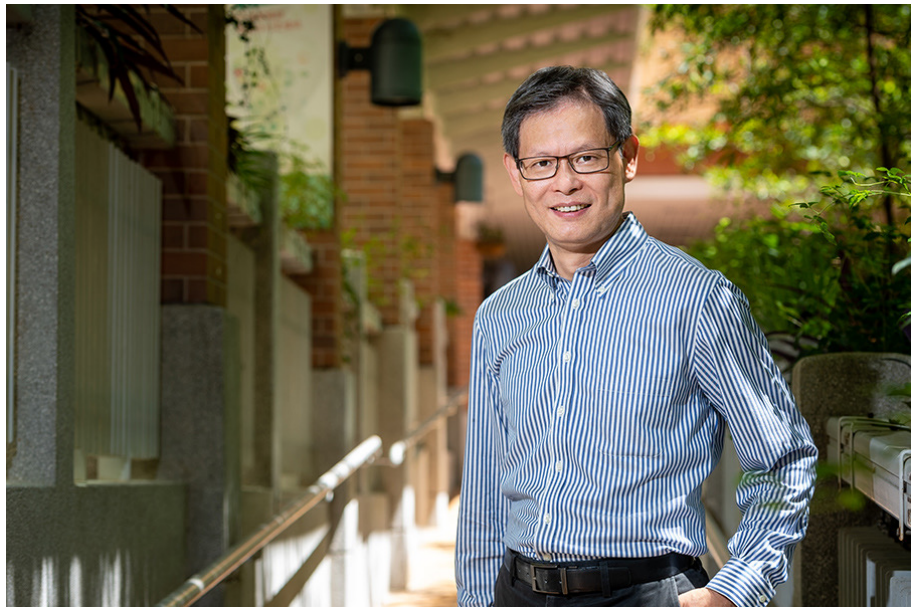
Reason using the information

outline

- What is artificial intelligence (AI)?
- What are we going to learn in this course?
- History, current status and the future of AI
- Searching Algorithm Part I

History, current status and the future of AI

- Invited Talk by Prof. Jiming Liu (6:30pm, Sep. 15)



Prof. Jiming Liu (*Fellow, IEEE*)

Chair Professor in Computer Science

Dean of Faculty of Science

outline

- What is artificial intelligence (AI)?
- What are we going to learn in this course?
- History, current status and future of AI
- Searching Algorithm Part I

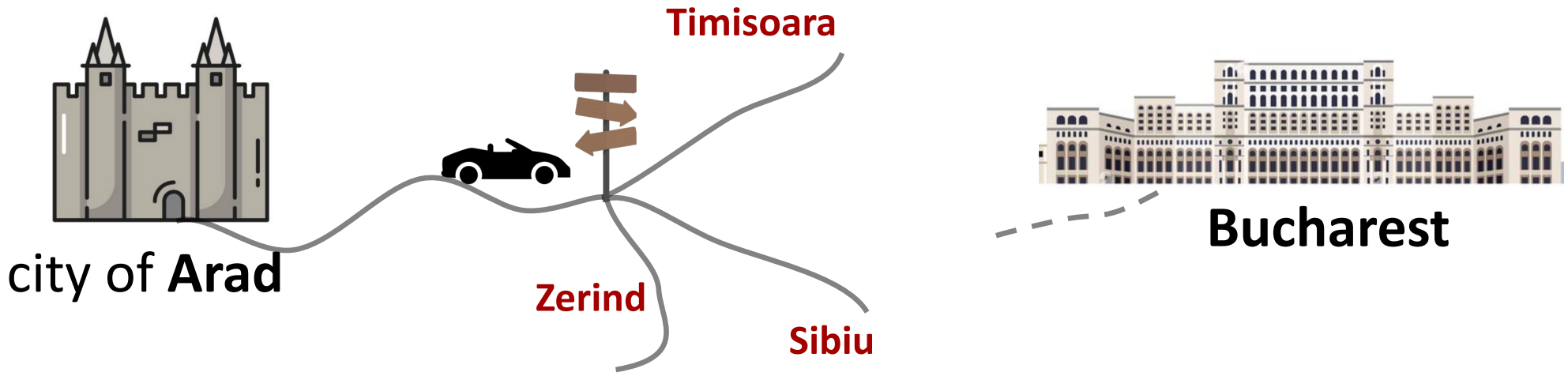
Let's imagine, you are taking a road trip in Romania …

You are in the city of **Arad** and need to arrive at **Bucharest** to meet your friends.

- Unluckily, your cell phone and GPS are not working.
- Even worse, you don't speak Romanian.

Let's imagine, you are taking a road trip in Romania …

You are in the city of **Arad** and need to arrive at **Bucharest** to meet your friends.

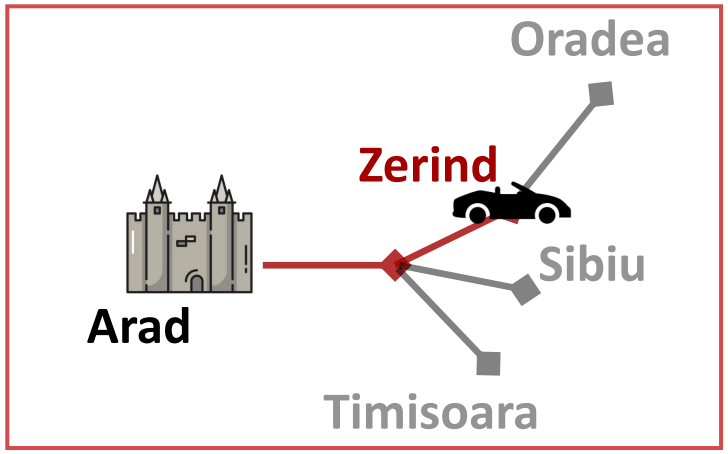


- There are three roads in front of you, toward **Sibiu**, **Timisoara**, and **Zerind**.

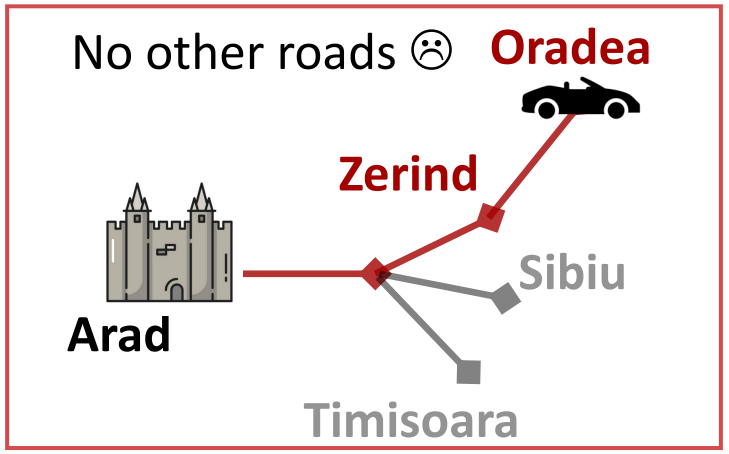
How would you find your way to Bucharest?

Randomly select one direction and explore?

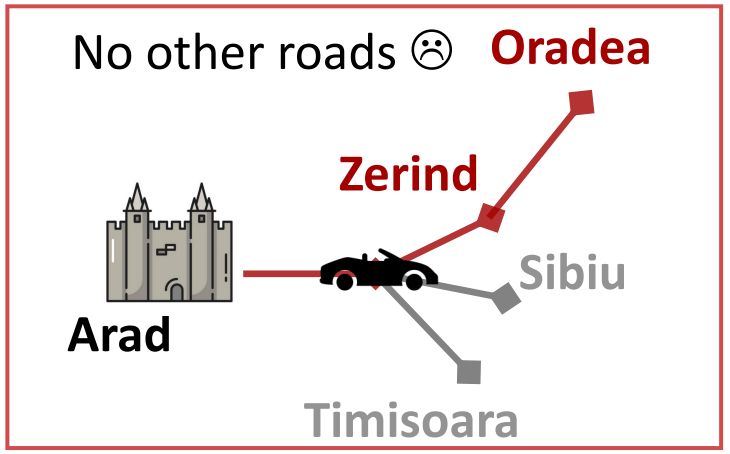
(1) Select Zerind:



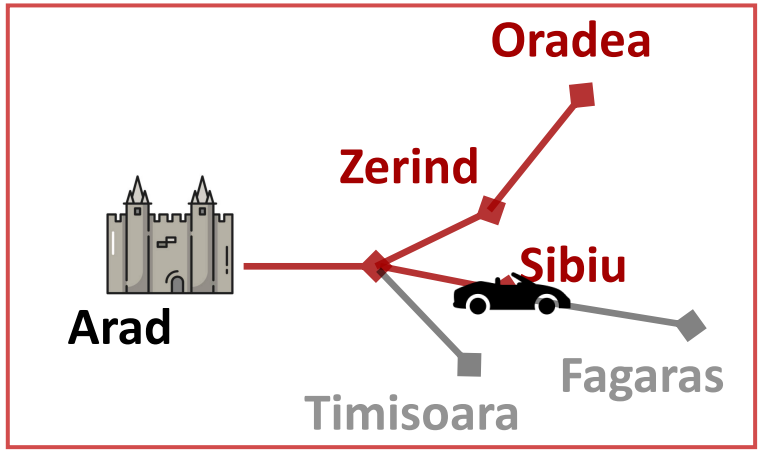
(2) Go to Oradea:



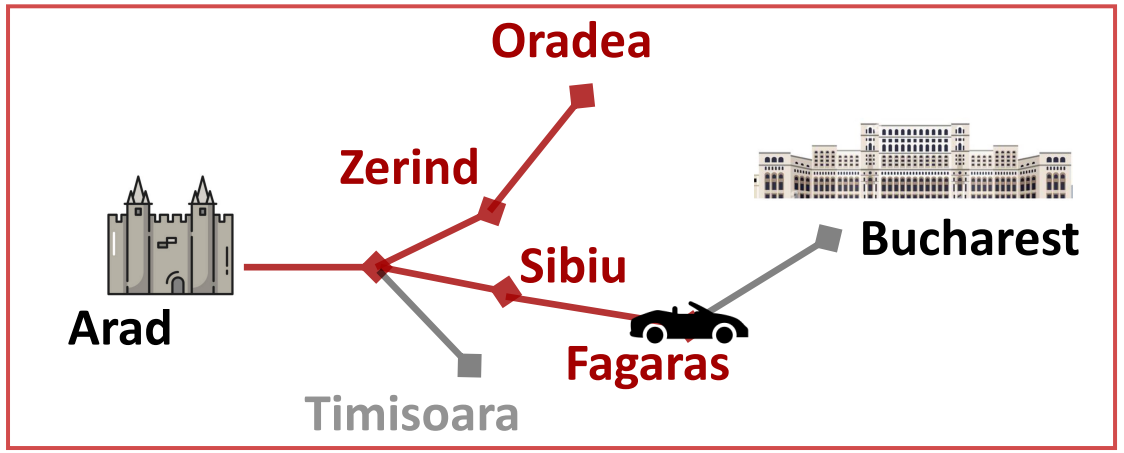
(3) Go back to Arad:



(4) Select Sibiu:



(5) Go to Fagaras:

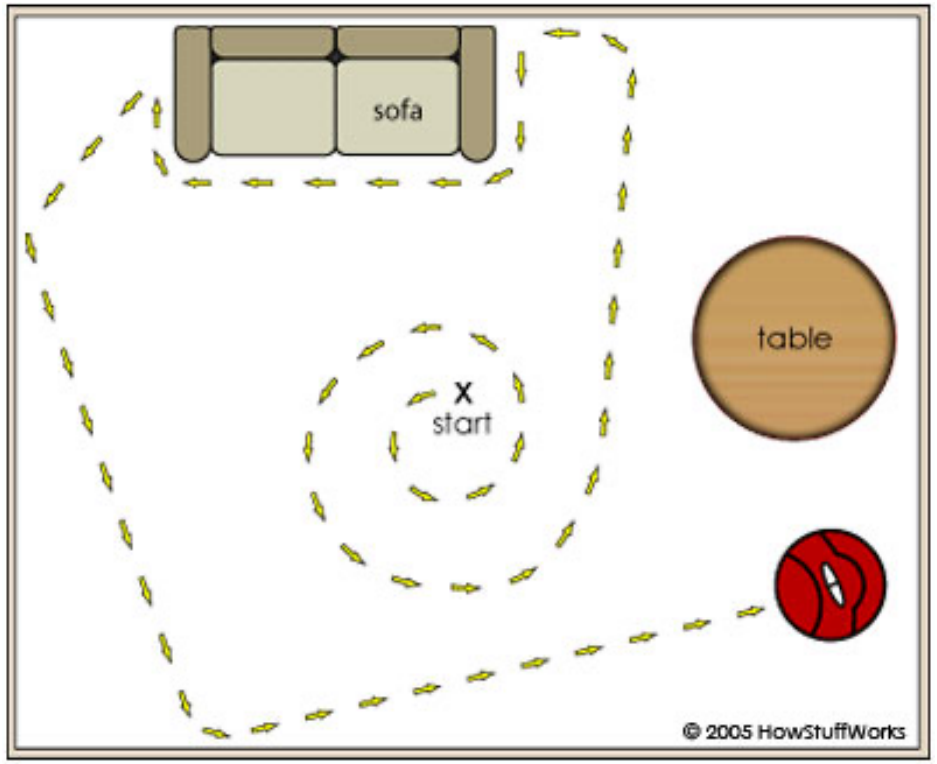


We made it!

Example: route finding

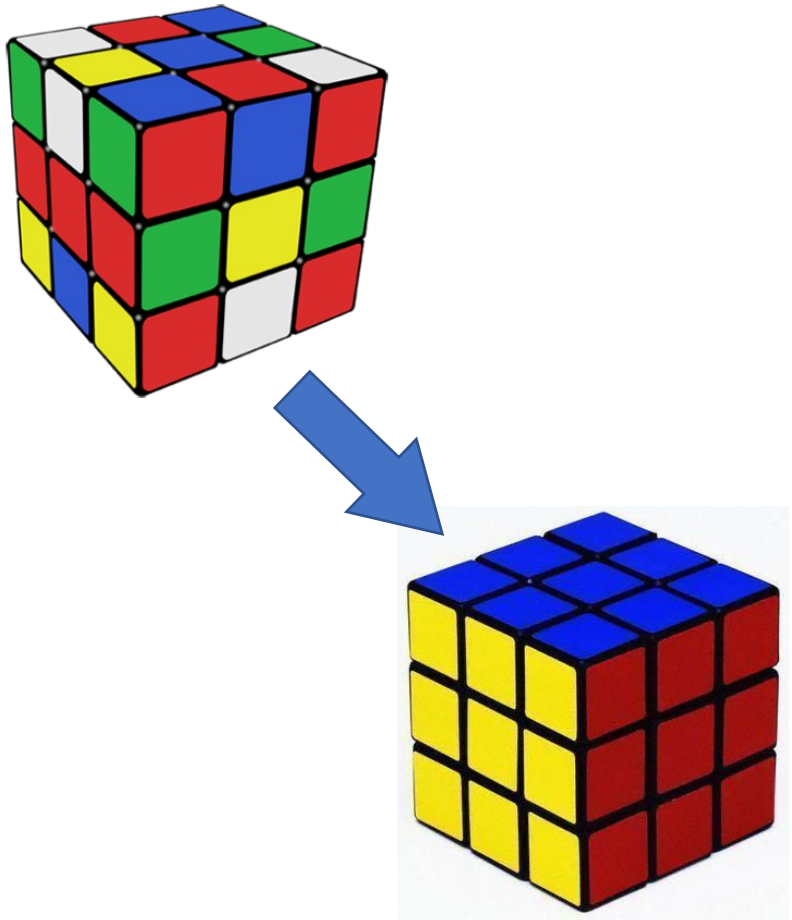
- Initial state: at Kowloon Tong Station
- Objective: shortest path to the classroom
- Actions: Go straight, turn left, turn right
- Given a map, an initial location, an objective. The goal is a sequence of actions leading to the destination.

Example: cleaner robot



- Initial state: at the charging dock
- Objective: most energy efficient? fastest?
- Actions: Move forward, rotate (degrees)
- The goal is to move to a specific position. Output a trajectory of a sequence of actions.

Example: Rubik's cube



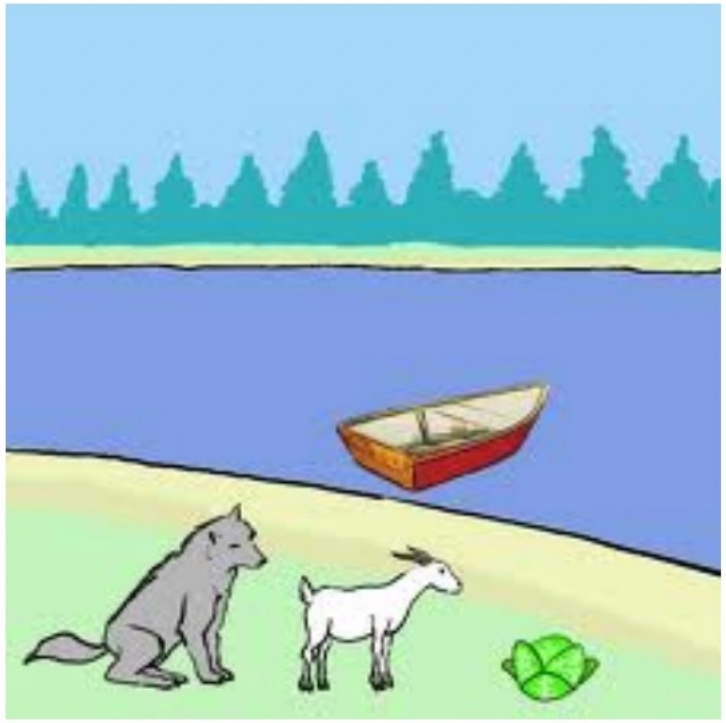
- Initial state: a particular configuration
- Objective: reach a certain configuration
- Actions: rotate front side clockwise, etc.
- The goal is to move to a specific position. Output a trajectory of a sequence of actions.

How to formulate such problems?

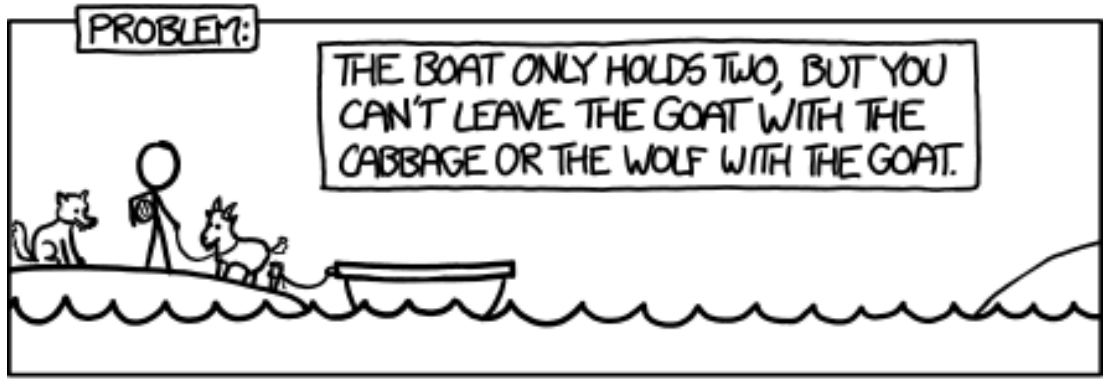
- possible states (*s*) that the environment can be in: *the state space.*
- An initial state and a set of goal states (or one goal state)

- The actions: ACTION(*s*) *a finite set of actions given a state s*
- A transition function: RESULT(s, *a*) *result of taking action a at state s*
- An action cost function:
  ACTION-COST(*s, a, s'*) *cost of applying action a at state s to reach state s'*

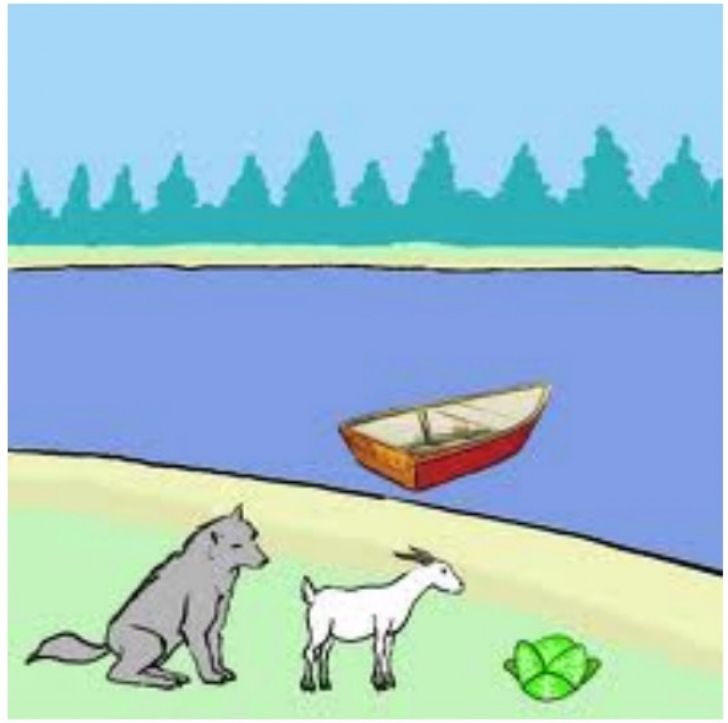
Example: River crossing puzzle



Farmer Cabbage Goat Wolf



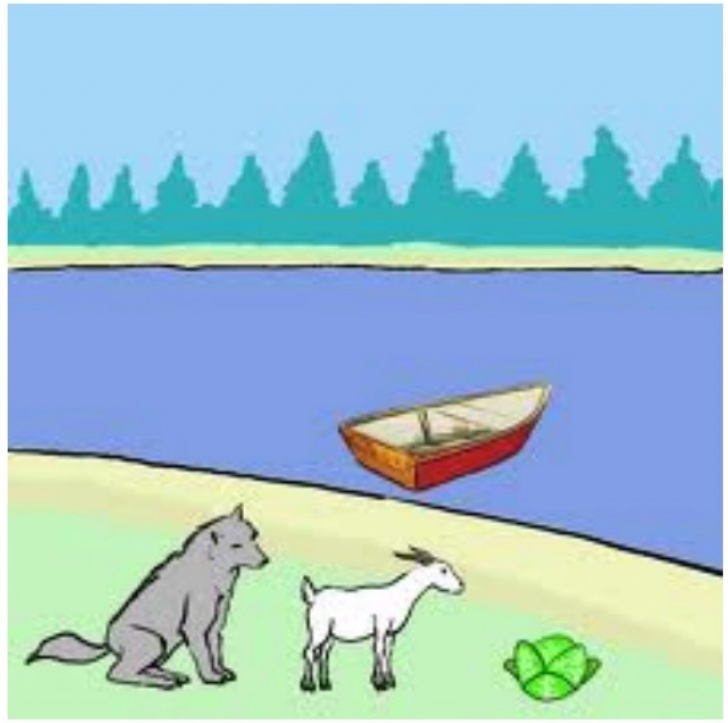
Example: River crossing puzzle



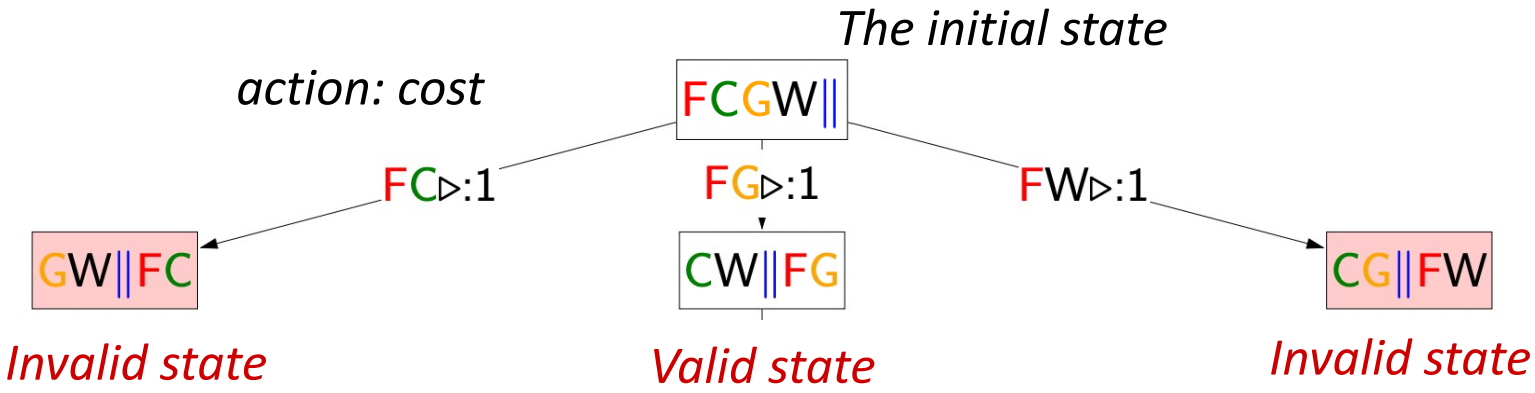
Farmer Cabbage Goat Wolf

- Actions:
  - F▷ Farmer cross the river alone
  - FC▷ Farmer cross the river with Cabbage
  - FG▷ Farmer cross the river with Goat
  - FW▷ Farmer cross the river with Wolf
- Approach:
  Building a **search tree**

Search tree



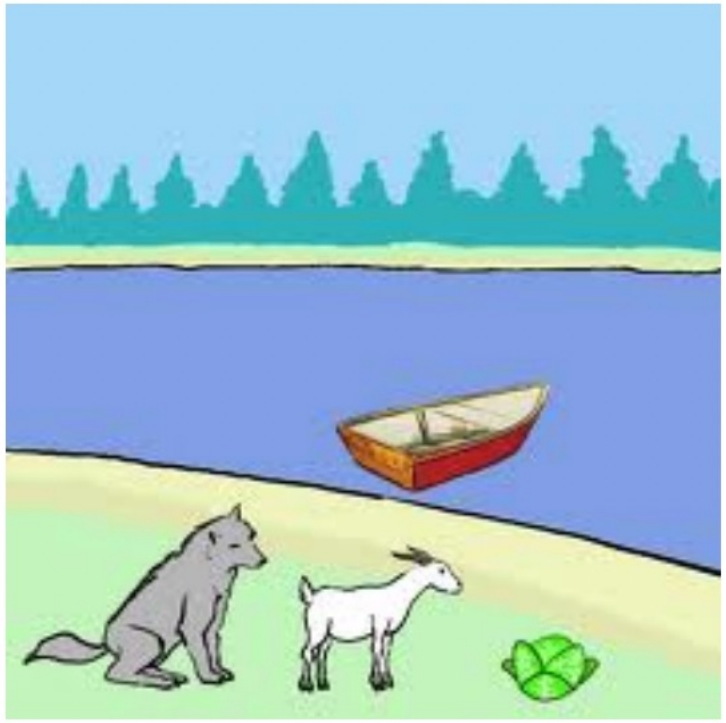
Farmer Cabbage Goat Wolf



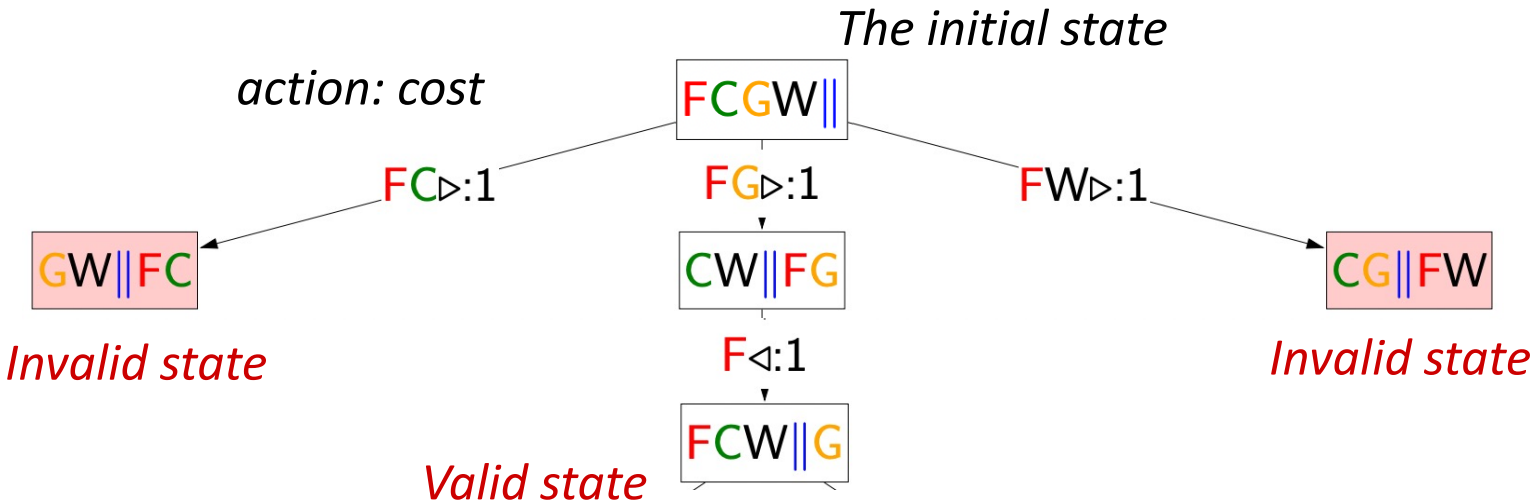
At each state *s* , get the return of ACTIONS(*s*) what are the possible actions at this state?

- F▷ Farmer cross the river alone
- FC▷ Farmer cross the river with Cabbage
- FG▷ Farmer cross the river with Goat
- FW▷ Farmer cross the river with Wolf

Search tree



Farmer Cabbage Goat Wolf

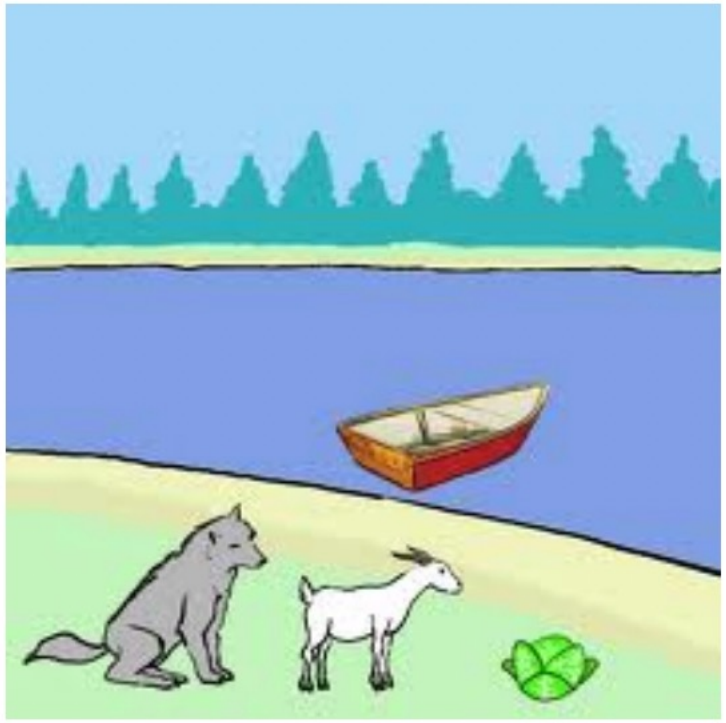


At each state *s* , get the return of ACTIONS(*s*)
what are the possible actions at this state?

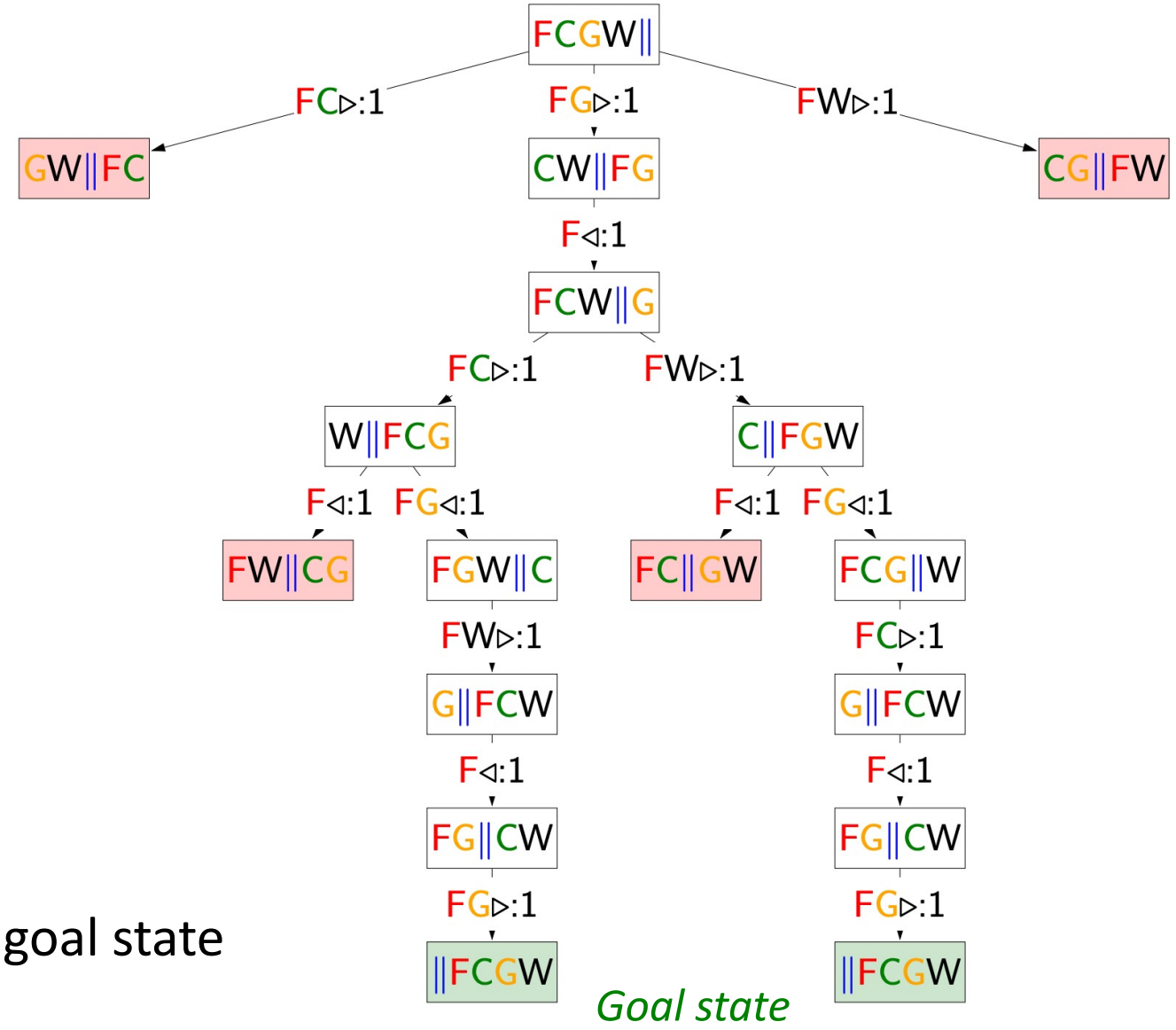
F◁ Farmer go back

~~FG◁ Farmer go back with Goat~~

Search tree



Farmer Cabbage Goat Wolf



Repeat this process, until we reach the goal state

How to formulate such problems?

- possible states (*s*) that the environment can be in: *the state space.*
- An initial state and a set of goal states (or one goal state)

- The actions: ACTION(*s*) *a finite set of actions given a state s*
- A transition function: RESULT(s, *a*) *result of taking action a at state s*
- An action cost function:
  ACTION-COST(*s, a, s'*) *cost of applying action a at state s to reach state s'*
- A solution: a path from initial to goal state. *It is optimal if it has minimal cost.*

Example: Two-cell Vacuum World

- **State**: agent loc . x dirt loc. = 2 x 2^2 = 8 reachable states
- **Initial state**: any state can be initial.
- **Actions**: {Left, Right, Suck}
- **Goal test**: both location A and B are clean
- **Action cost**: each step costs 1 (i.e., path cost = #actions in the path)
- **Transition function**

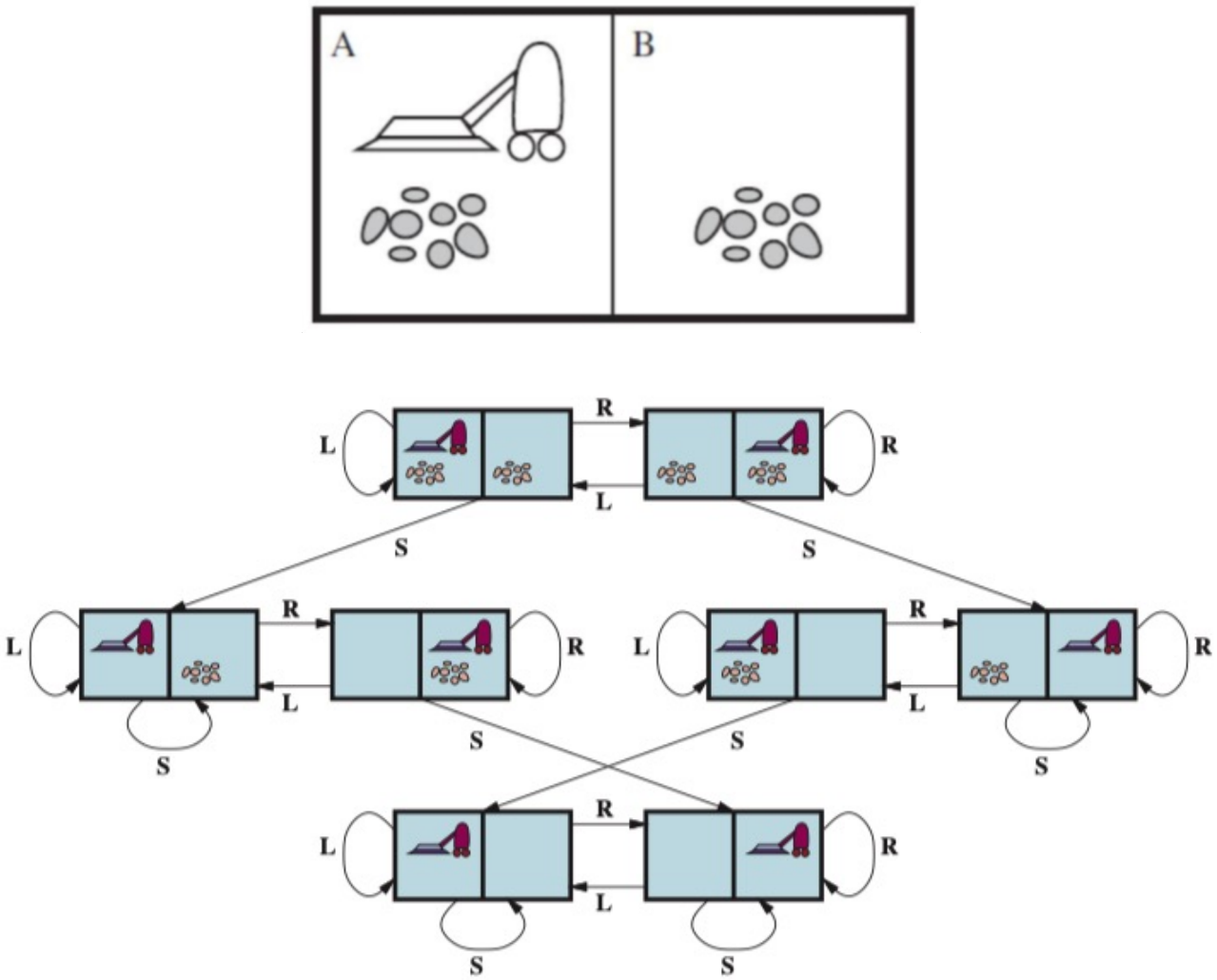
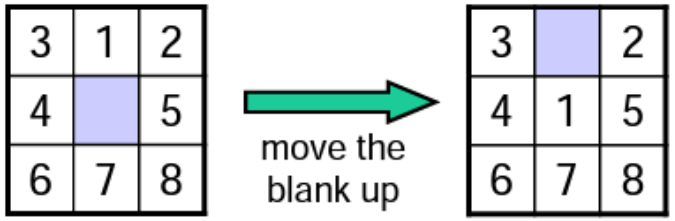
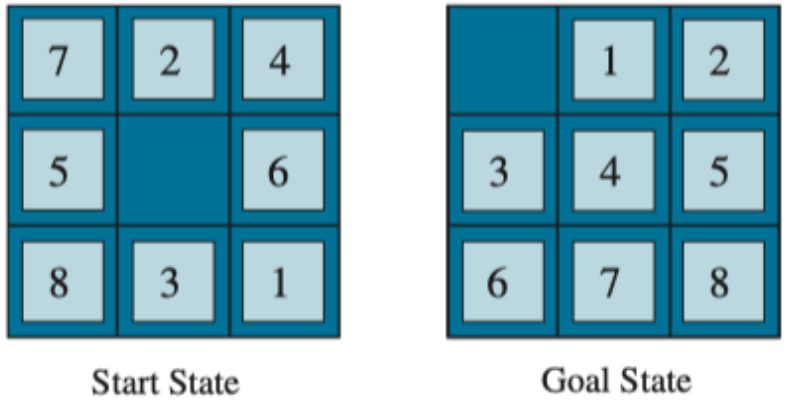


Figure 3.2 The state-space graph for the two-cell vacuum world. There are 8 states and three actions for each state: L = *Left*, R = *Right*, S = *Suck*.

Example: Slide 8-puzzle

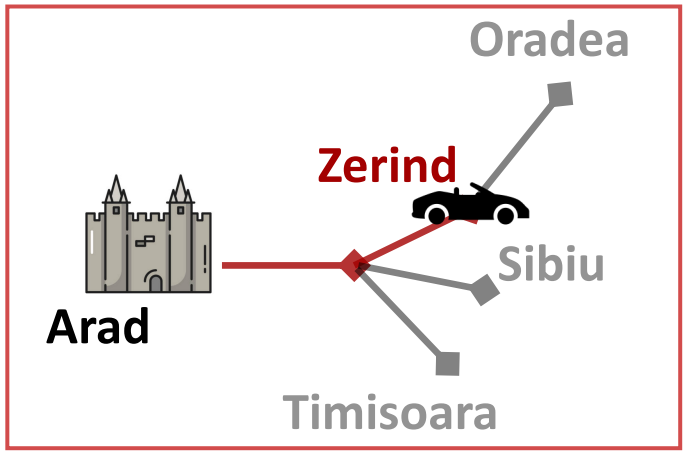
- **State**: all possible configurations of numbers 1-8 and blank cell.
- **Initial state:** any state can be initial.
- **Actions:** {Left, Right, Up, Down}
- **Goal test:** state matches goal configuration
- **Action cost:** each action costs 1
- **Transition function:** *e.g.,*





Two Types of search algorithms

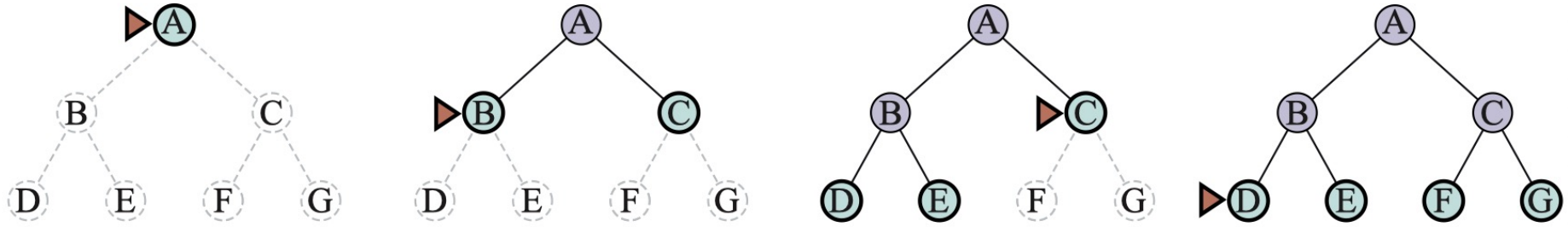
- Uniformed search (blind search)
  - No clue about how close a state is to the goal state.
  - It only distinguishes a goal state from a non-goal state.
  - Breadth-first Search (BFS), Uniform-cost Search, Depth-first Search (DFS), Depth-limited Search



- Informed search (heuristic search)
  - We have some hints about the location of goals.
  - Greedy Search, A* Search.

Breadth-first Search (BFS)

- Always expand the nodes at depth *d-1* before expanding nodes at depth *d*.
- Store the visited nodes (in purple) and frontiers (nodes to visit; in cyan).

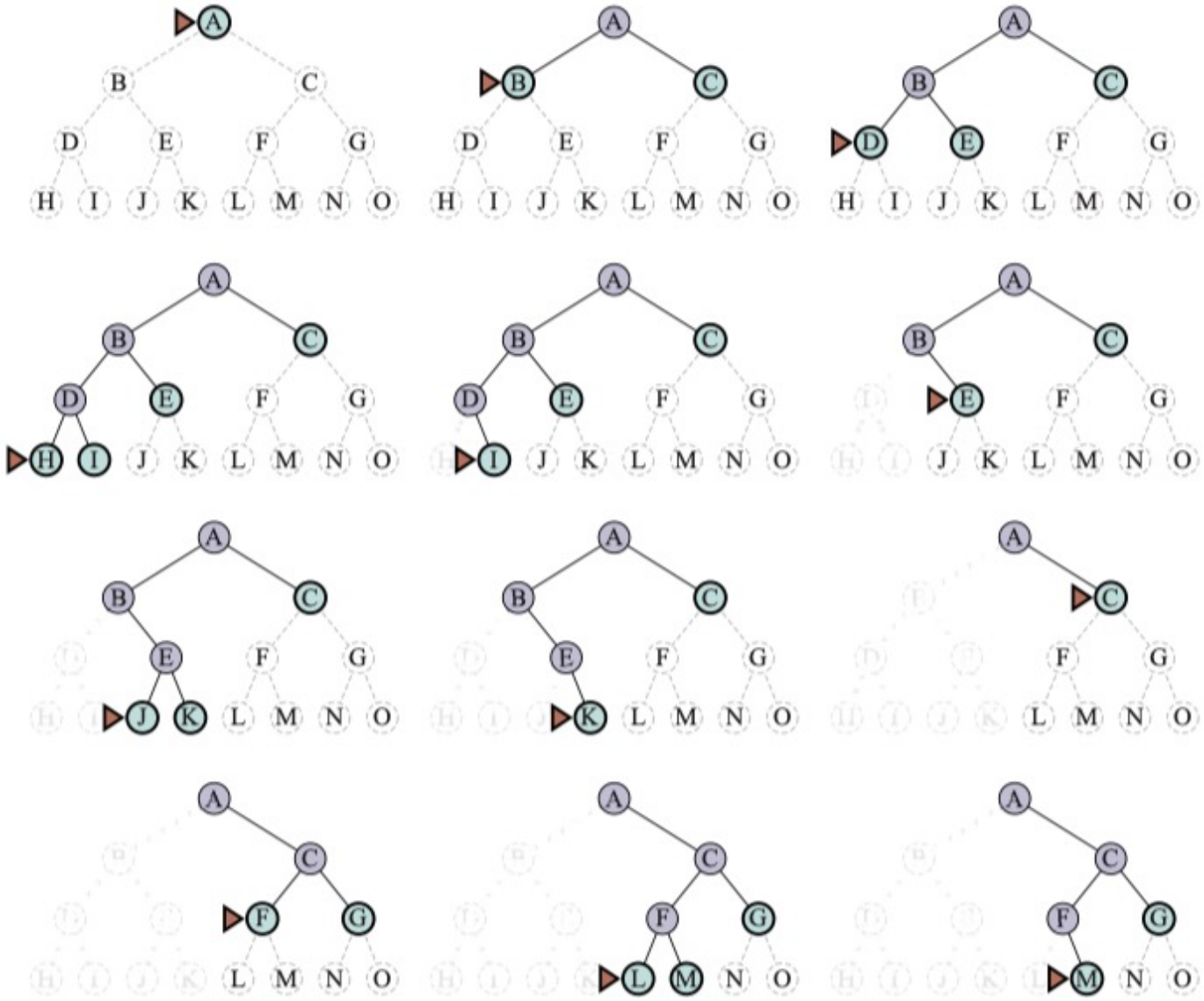


- Guaranteed to find the solution, if it exists.
- It always finds the shallowest goal state first (minimal number of actions)
  → *cost-optimal if all actions have the same cost.*

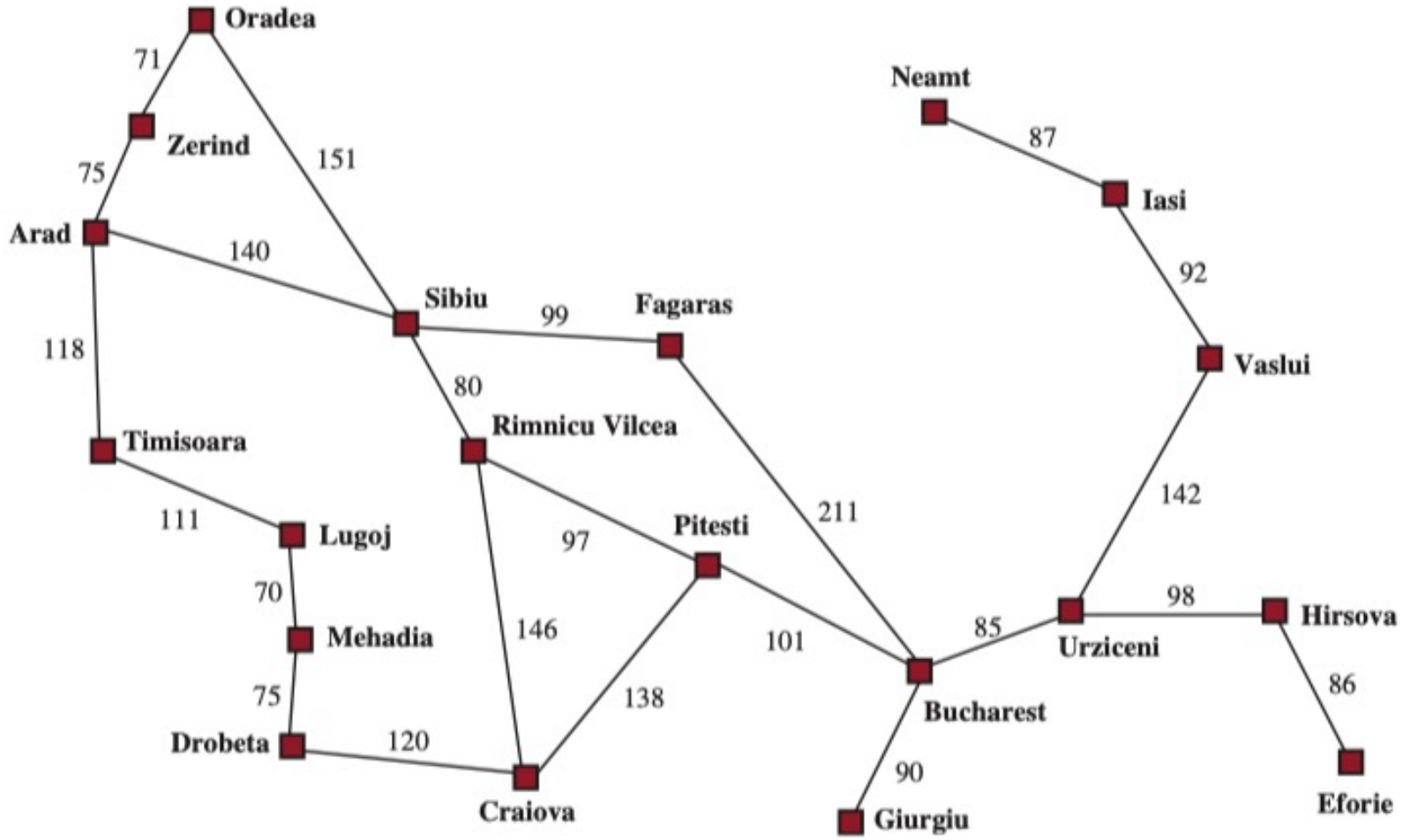
- $O(b^d)$ time and memory complexity.
  b: avg. number of successors of each node; *d*: depth of the goal state.

Depth-first Search (DFS)

- Always expand one of the nodes at the deepest level of the tree.
- Store the visited nodes (in purple) and frontiers (nodes to visit; in cyan).
- Modest memory requirements.
- Get stuck going down the wrong path.



Exercise: Perform BFS and DFS to reach Bucharest from Arad



Wish you a happy mid-autumn festival!