CPSC 481
Artificial Intelligence

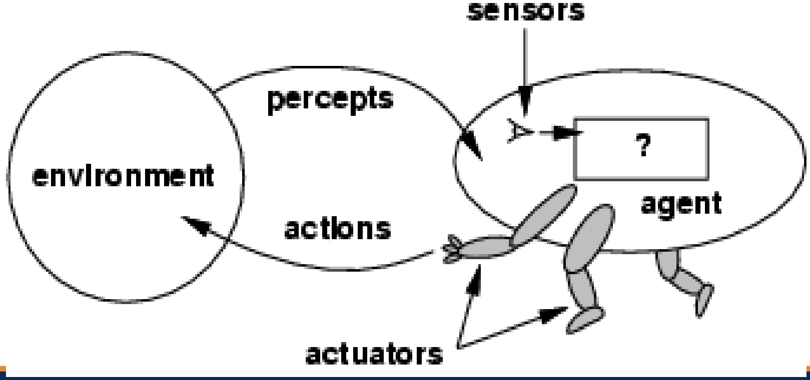
Mr. Ayush Bhardwaj
abhardwaj@fullerton.edu

What we will cover today

- Types of environments
- Search: a general purpose problem solving strategy
- State spaces
- Representing state spaces

Intelligent Agents

- An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators
- Human agent has eyes, ears, and other organs for sensors; hands, legs, mouth, and other body parts for actuators.
  - Robotic agent has cameras and infrared range finders for sensors; various motors for actuators.
- The agent function maps from percept histories to actions: [*f*: P* ☾ A]
- The agent program runs on the physical architecture to produce
  - agent = architecture + program
- Rational agents use
  - performance measures
  - environment
  - actuators
  - sensors



Properties of Task Environments

- **Fully observable vs. partially observable**: – Fully observable if sensors detect all aspects of environment relevant to choice of action
  - Partially observable due to noisy, inaccurate or missing sensors
  - E.g., in card games, not possible to see what cards others are holding
- **Single agent vs. multiple agents: in single agent environment, no other “thinking” entity**
  - Multiple agents: other entities whose goals depend on your own
  - Cooperative: improving other’s situation, helps your own
  - Competitive: improving other’s situation, worsens your own
- **Deterministic vs. stochastic**: Deterministic if the next state of the environment is completely determined by the current state and the action executed by the agent
  - Stochastic if what happens is affected by randomness
  - E.g., roll of dice in a game

Properties of Task Environments-cont.

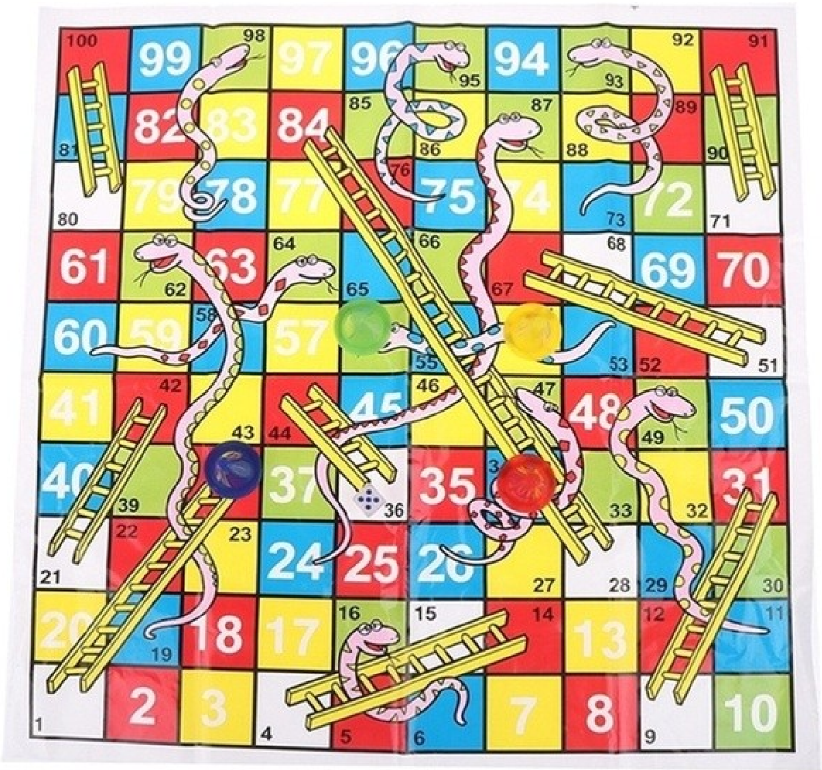
- **Episodic vs. sequential:** episodic if a current action will not affect future actions - the agent's experience is divided into independent episodes. E.g., mail sorting system
  - Sequential if current decisions affect future decisions.
  - E.g., navigating a car is sequential, most games
- **Static vs. dynamic**: Static if the environment does not change as the agent is deciding on an action. E.g., chess
  - dynamic if the environment may change while the agent is thinking
- **Discrete vs. continuous**: in discrete environments, actions/percepts can take only specific values (e.g., chess); in continuous environments, there can be infinite such values.
  - Signals constantly coming into sensors, actions continually changing. Car driving is continuous.
- **Known vs. unknown**: In a known environment, the outcomes for all actions are given, i.e., the rules of the game are known
  - If unknown, the agent will have to learn how it works.

Example: playing soccer

- Partially observable – Player cannot detect all the things on soccer field that can affect its action, for e.g. it cannot determine what other players are thinking.
- Multi-agent – There are many players in a soccer game.
- Stochastic – Given the current state and action executed by agent, the outcome cannot be exactly determined. E.g., if player kicks the ball, then the ball may or may not be stopped by other players, or the soccer field can change in many different ways depending on how players move.
- Sequential – The past history of actions in the game can affect the next action in the game.
- Dynamic – The environment can change while the agent is making decision, for e.g., position of other players changes when a player moves.
- Continuous – Positions of the ball and players are continuous. The speed or the direction (angle) at which the player kicks the ball is continuous.

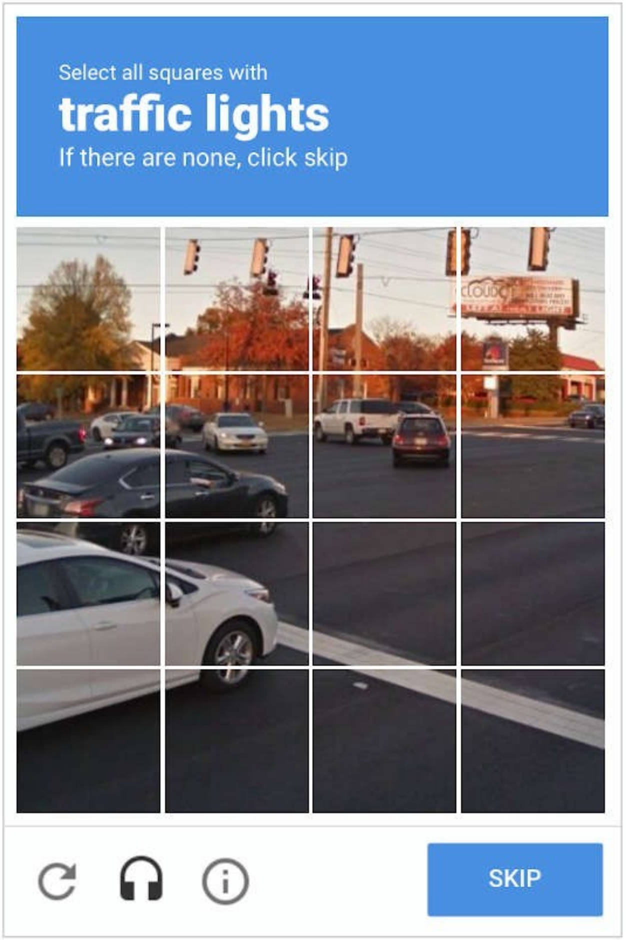
Chutes and ladders

- Fully observable vs. partially observable?
- Deterministic vs. stochastic?
- Episodic vs. sequential?
- Static vs. dynamic?
- Discrete vs. continuous?
- Known vs. unknown?
- Single agent vs. multiple agents?



Passing a CAPTCHA

- Fully observable vs. partially observable?
- Deterministic vs. stochastic?
- Episodic vs. sequential?
- Static vs. dynamic?
- Discrete vs. continuous?
- Known vs. unknown?
- Single agent vs. multiple agents?



Chess

- Fully observable vs. partially observable?
- Deterministic vs. stochastic?
- Episodic vs. sequential?
- Static vs. dynamic?
- Discrete vs. continuous?
- Known vs. unknown?
- Single agent vs. multiple agents?

Driving a car

- Fully observable vs. partially observable?
- Deterministic vs. stochastic?
- Episodic vs. sequential?
- Static vs. dynamic?
- Discrete vs. continuous?
- Known vs. unknown?
- Single agent vs. multiple agents?

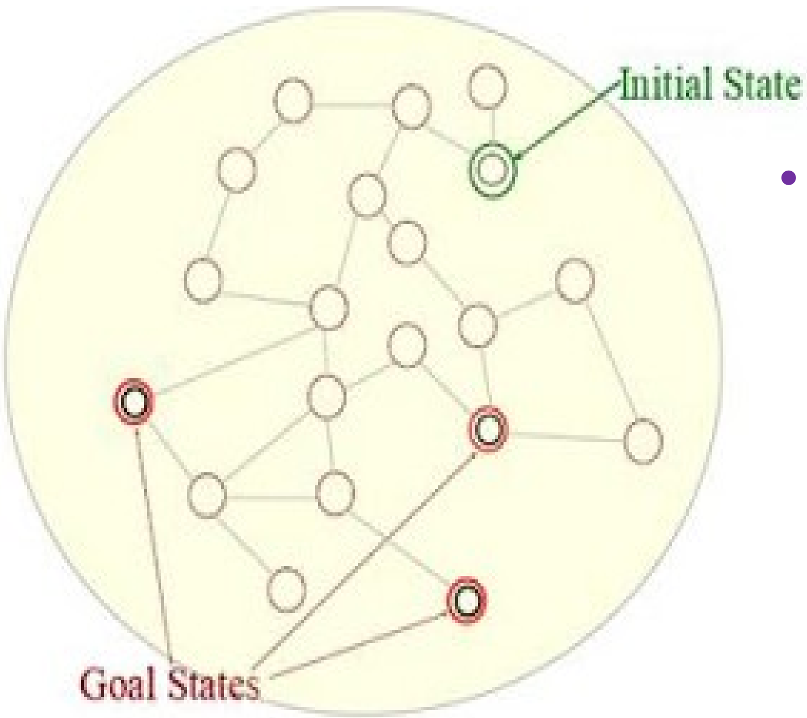
Classwork

- On Google Drive
  - **Link on Canvas**
- Go to today's date and section
  - Editable and viewable by all
- Create a new Google Document and name it with your groupmates' names
  - E.g.: Garcia-Joshi-Nguyen
  - Write the names and role of group members
  - Roles: manager, recorder+presenter, timekeeper
  - Add your answers
- Will not be graded but used for class participation points
  - Can easily share with rest of class outside the breakout room

General Problem Solving Strategy

- How does a human solve a problem in general?
  - Do we use thousands of algorithms to solve different problems, or
  - use only a few general methods to solve all types of problems?
- Is there a general purpose approach to solve all types of problems?
  - Driving a car, Playing chess, Finding the cheapest car, Buying a ticket, …
- State space search as a general problem solving strategy

Human Problem Solving Process



- **Think about these problems:**
  - Playing chess (Tic-tac-toe or 8 puzzle) game,
  - Navigate a maze
  - Driving a car
- What do we do to solve a problem?
  - **Understand** the problem
    - solution/goal, constraints, states
  - **Define** a **state** for each step and find a **sequence of states (or steps)**.
    - A state can be a problem solving step or status (information and available methods), e.g., a state of object in Object-Oriented programming.
  - Use available **information** and **methods** to **move from one state to next state**.

State Space Search

- **State**, **State Space**, and **Search**:
  - A state is a representation for a problem solving step that involves **available information** and **methods**.
    - A **state** captures only the features of a problem **essential** to solve it
  - The state space of a problem: set of all possible states.
  - A search is an algorithm for exploring the state space.
- State space search as a **general problem solving strategy** is based on a strategy used by humans to solve difficult problems or almost all problems if resources and time are unlimited!
  - AI was considered as a problem of state representation and search in early AI research.

State Representation

- *Expressiveness* and *efficiency* are the key factors.
  - Need to optimize the trade-off between **expressiveness** and **efficiency**
  - Ultimately we need a *powerful representation scheme* to solve AI problems.

- Different levels of state representation:
  - **Conceptual** (or mental) representation,
    - State
  - ***Symbolic*** *representation*,
    - *Graph*
  - **Computer** representation (data structure)
    - Variable, array, record, object, table, list, tree, queue, …

Definition of a graph

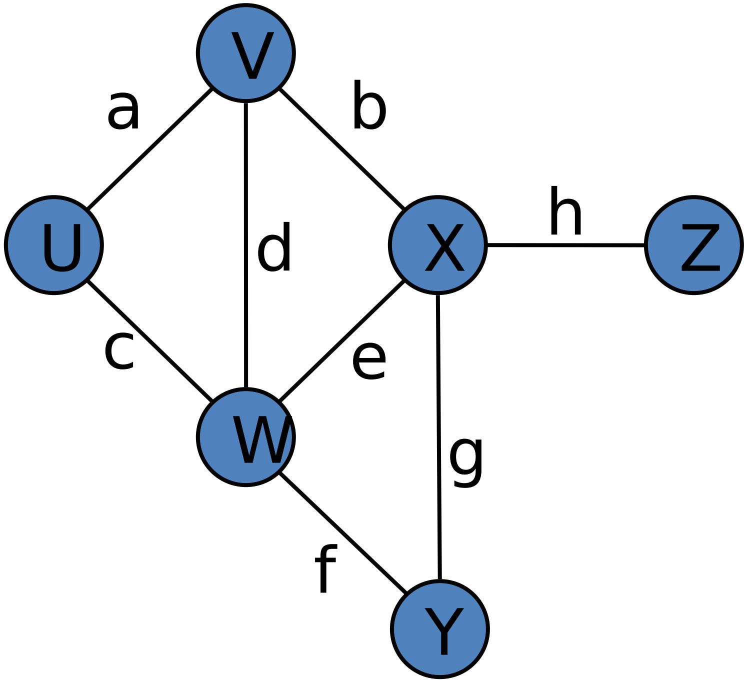
- A graph consists of
- A set of *nodes/vertexes/vertices*
    - can be infinite
- A set of *arcs/edges* that connect pairs of nodes
- An *arc/edge* is an ordered pair, e.g.,

  (i1, rb1)

  (rb1, i1)

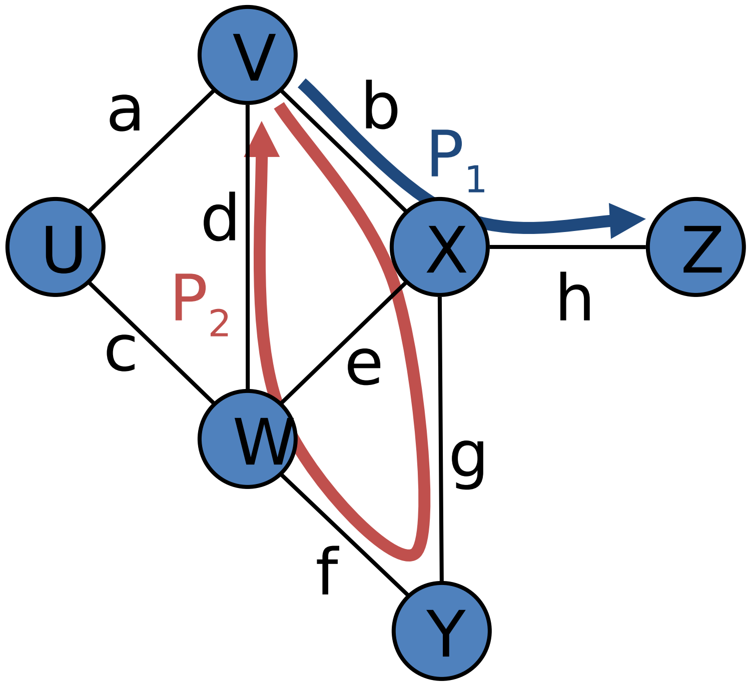
Terminology

- End vertices (or endpoints) of an edge
  - Endpoints of a?
  - U and V
- Edges incident on a vertex
  - Incident on V?
  - a, d, and b
- Adjacent vertices
  - U and V?
  - U and V are adjacent
  - U and X?
- Degree of a vertex
  - Degree of X?
  - X has degree 4

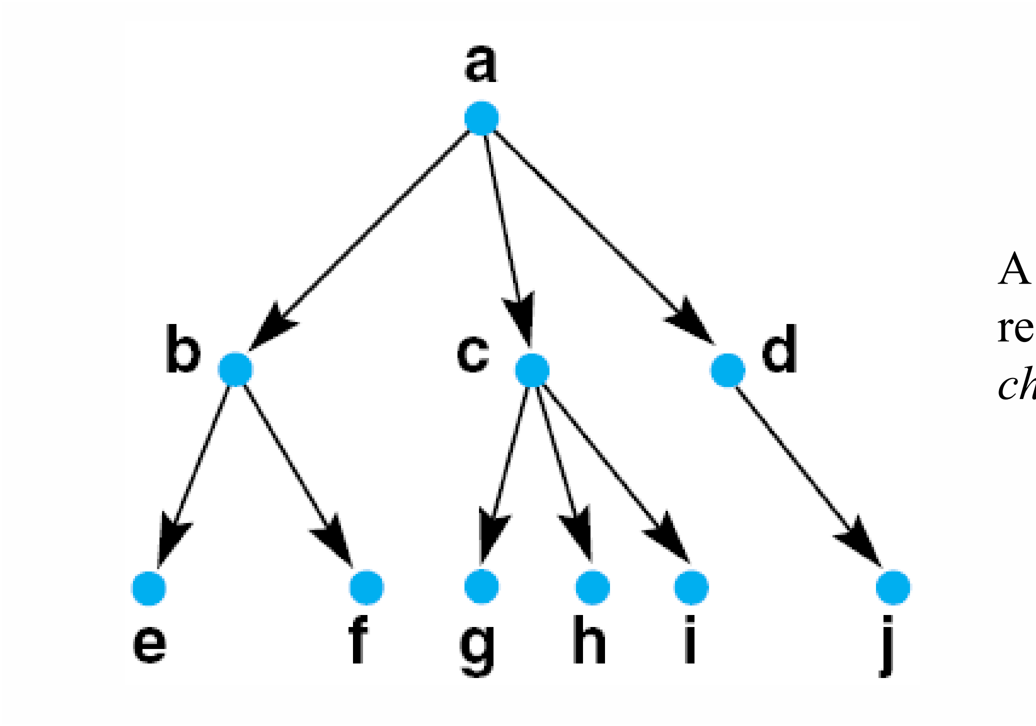


Terminology (cont.)

- Path
  - sequence of edges
  - begins with a vertex
  - ends with a vertex
- A *path of length n* has n edges
  - [U V X] is a path of length ?
- Cycle
  - path that begins and ends with the same vertex
- Examples
  - P_1=(V,b,h,Z) is a path
  - P_2=(V,b,g,f, d,V) is a cycle



A Tree is a Rooted Graph



A **tree** showing a family relationships, *parent* and *children*

****Tree**: has a root that has path from the root to all nodes and every path is unique without cycle.

Problem space

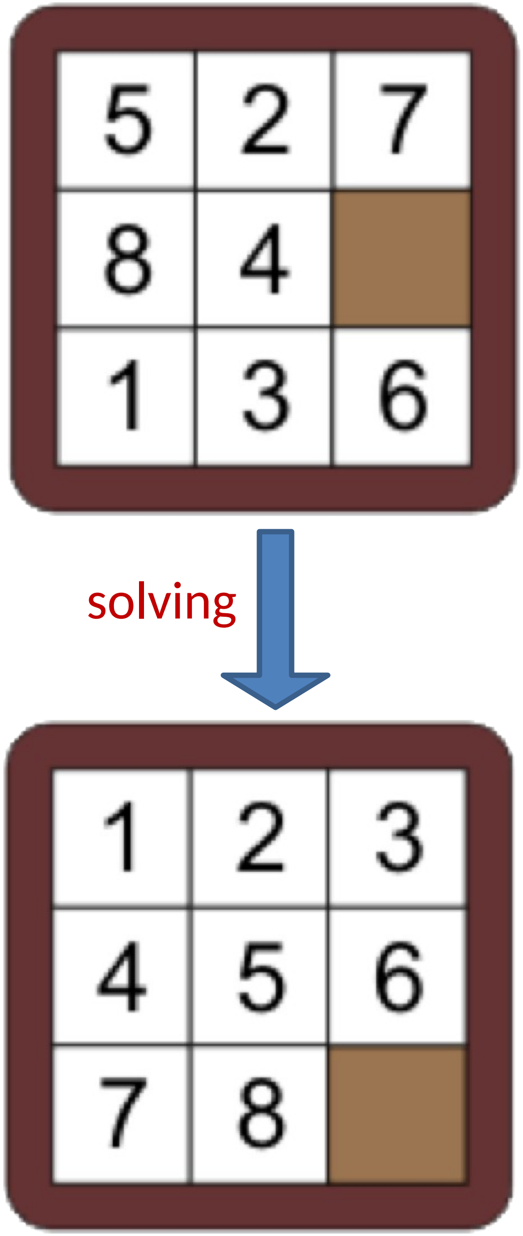
- a *state space:* a set of states representing the possible configurations of the world
- a set of *operators/actions:* change one state into another
- The problem space is a graph where the states are the nodes and the edges represent the operators.

State space search

- Represented by a four-tuple [*N*,*A*,*S*,*GD*], where:
- *N* is the problem space, the set of nodes/states
- *A* is the set of edges between nodes. These correspond to the actions/operators.
- *S* is a nonempty subset of *N*. It represents the start state(s) of the problem.
- *GD* is a nonempty subset of *N*. It represents the goal state(s) of the problem. The states in GD are described using either:
  - a measurable property of the states
  - a property of the path in the search (a *solution path* is a path from a node in *S* to a node in *GD*)

Sliding tile puzzle

https://www.helpfulgames.com/subjects/brain-training/sliding-puzzle.html



State space of the sliding tile puzzle

- State:
- operators:
- initial state:
- goal state:

The 8-puzzle problem as state space search

- states: possible board positions
- operators: one for sliding each square in each of four directions
  - Better, one for moving the blank square in each of four directions
- initial state: some given board position
- goal state: some given board position
- Note: the “solution” is not interesting here, we need the path.

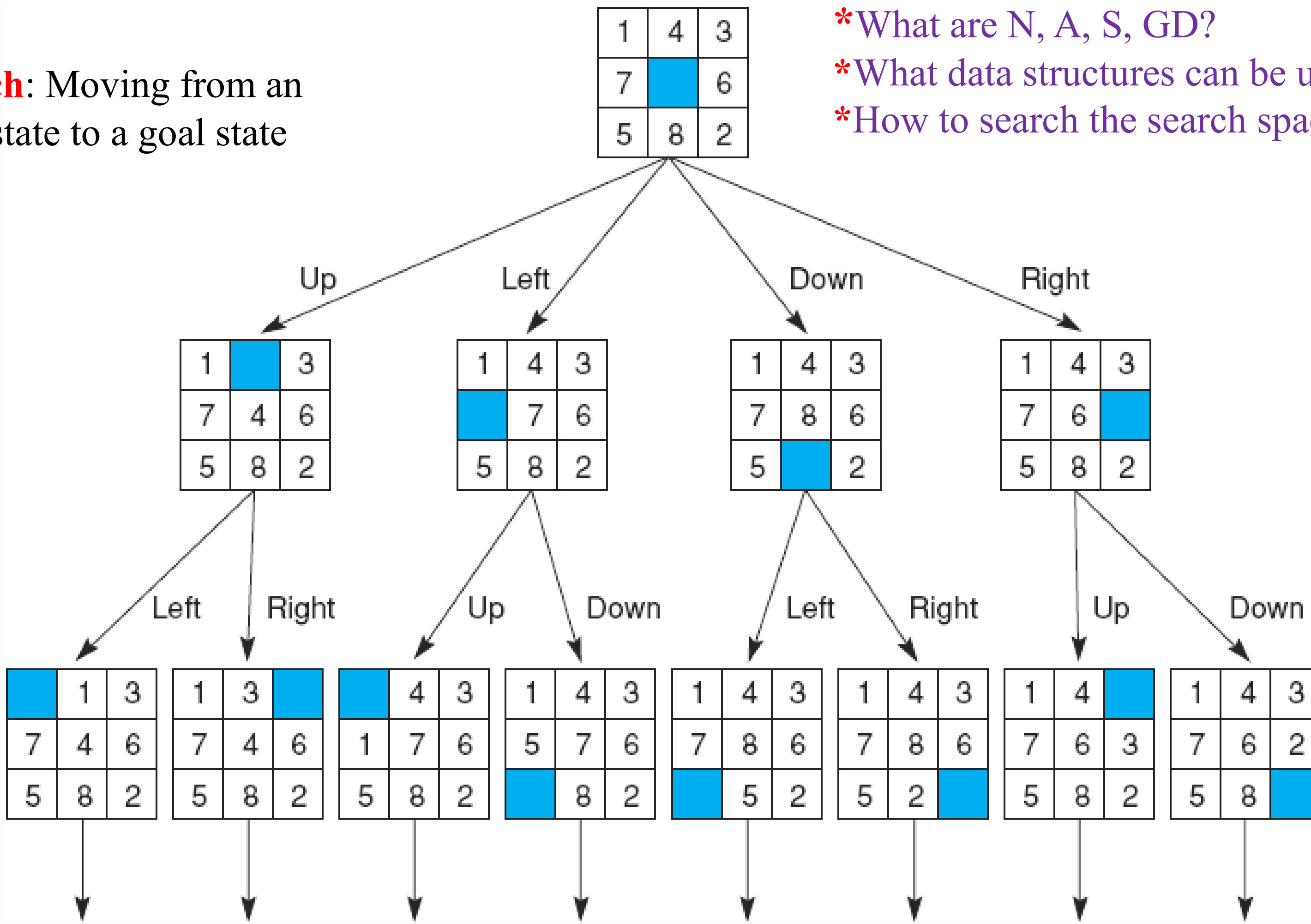
A State Space <u>Graph</u> for the 8-puzzle Generated by “move blank” Operations

*__Search__: Moving from an initial state to a goal state

*What are N, A, S, GD?

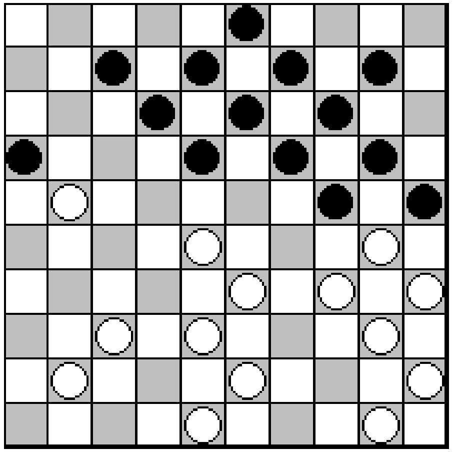
*What data structures can be used?

*How to search the search space?



A graph or a tree?

Classwork



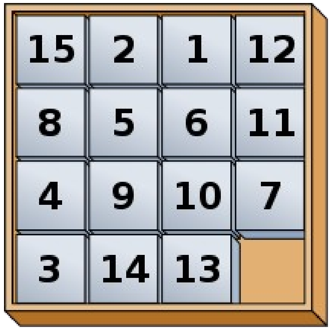
- Consider these two games:
  - Checkers/draughts
    - https://en.wikipedia.org/wiki/Draughts
  - Tic-tac-toe
- What is the state?
  - Think about how you would represent it in a computer program
- What is the state space?
- What are the operators?
- initial state?
- goal state?

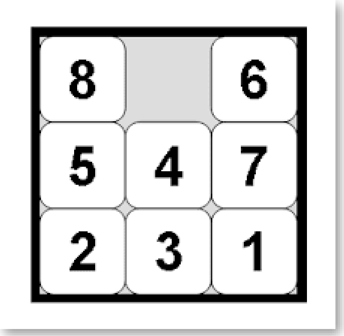
Classwork

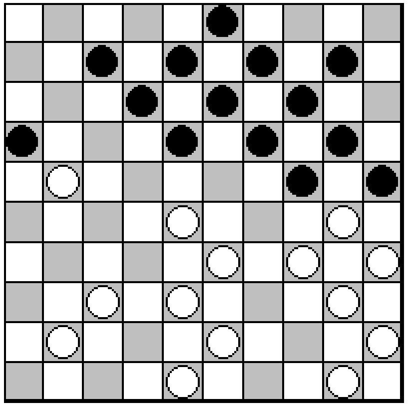
- On Google Drive
  - **Link on Canvas**
- Go to today's date and section
  - Editable and viewable by all
- Create a new Google Document and name it with your groupmates' names
  - E.g.: Garcia-Joshi-Nguyen
  - Write the names and role of group members
  - Roles: manager, recorder+presenter, timekeeper
  - Add your answers
- Will not be graded but used for class participation points
  - Can easily share with rest of class outside the breakout room

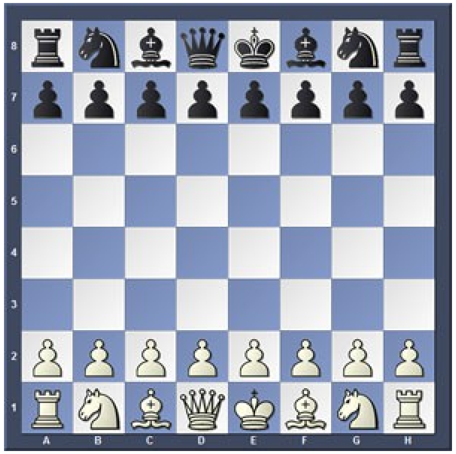
Branching factor

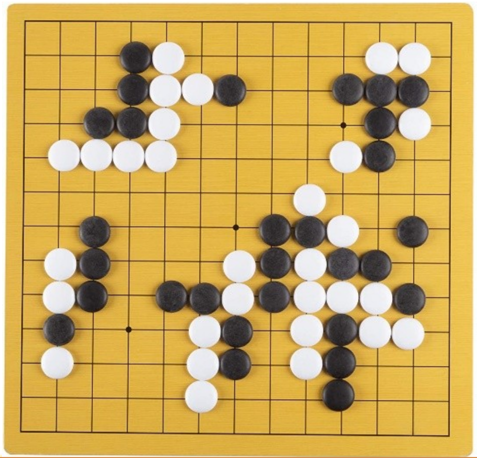
- Number of next states from a given state
- Number of children of the node
- Branching factor can vary at different nodes
  - Average branching factor
- Examples:
  - 8-sliding tile puzzle?
  - 15-sliding tile puzzle?
  - Checkers?
  - Chess?
  - Go?





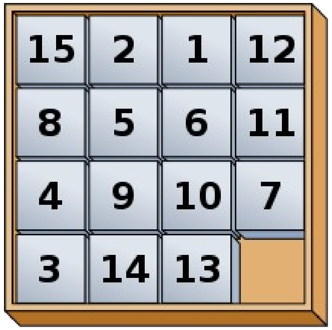


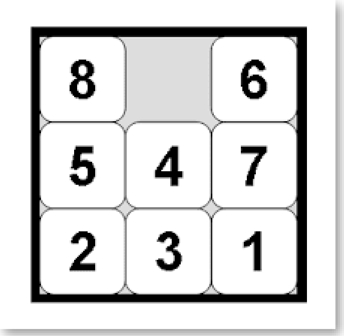


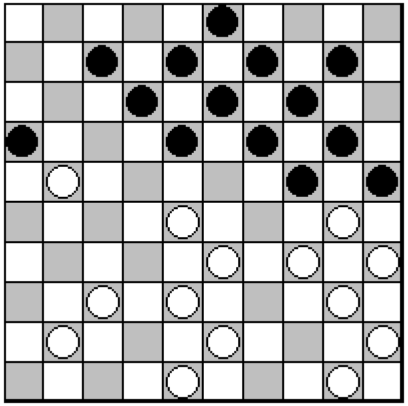


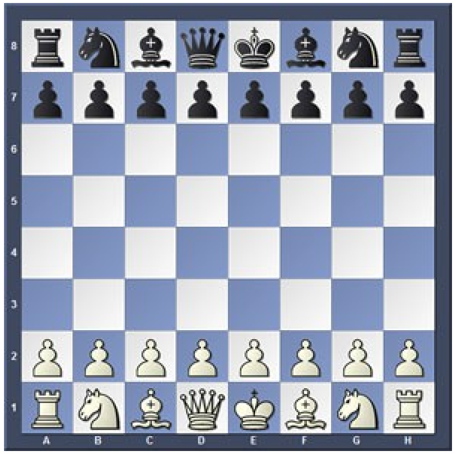
Branching factor

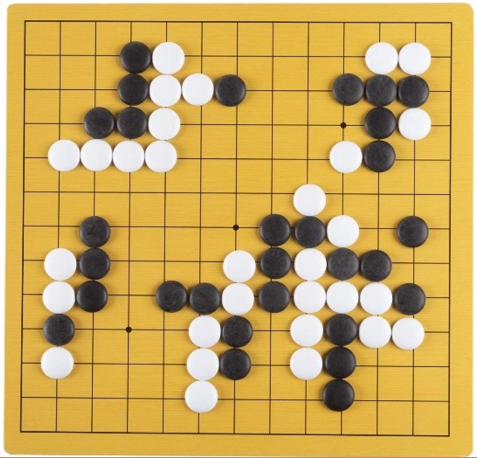
- Number of next states from a given state
- Number of children of the node
- Branching factor can vary at different nodes
  - Average branching factor
- Examples:
  - 8-sliding tile puzzle?
    - 2-4, average=2.13
  - 15-sliding tile puzzle?
    - 2-4
  - Checkers?
    - About 7
  - Chess?
    - About 35, average=31
  - Go?
    - 250



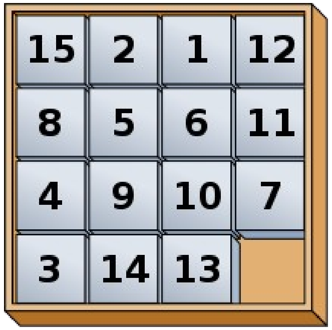




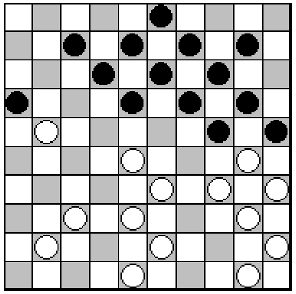


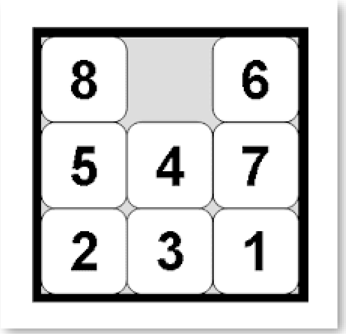


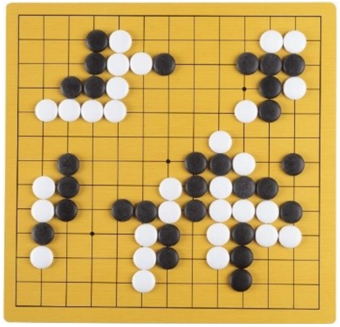
State space complexity



- State space: set of all possible states of a problem
- State space complexity: number of states in the state space
- Examples:
  - 8-sliding tile puzzle?
  - 15-sliding tile puzzle?
  - Checkers?
  - Chess?
  - Go?



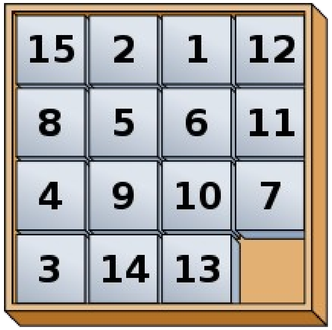


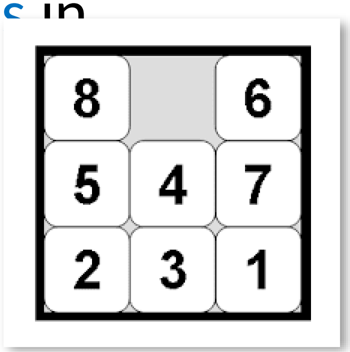


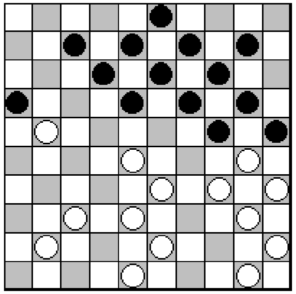
https://en.wikipedia.org/wiki/Game_complexity

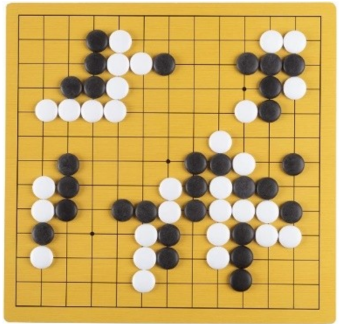
State space complexity

- State space complexity: number of states in the state space
- Examples:
  - 8-sliding tile puzzle?
  - 15-sliding tile puzzle?
  - Checkers?
  - Chess?
  - Go?
- if branching factor is 10, then
  - 10 nodes one level down from the current position
  - 10^2 (or 100) nodes two levels down
  - 10^3 (or 1,000) nodes three levels down, …
  - State space explosion









Searching a graph: the challenge

- Number of nodes is practically infinite
  - Cannot pre-compute all nodes and store it in a computer/data structure
  - “Search”, not a “traversal”
- Nodes can reappear in the search
  - Possible to get into loops

Searching a graph (simplified)

- Start with the initial state
- Loop until goal found
- find the nodes accessible from the root

References

- George Fluger, Artificial Intelligence: Structures and Strategies for Complex Problem Solving, 6th edition, Addison Wesley, 2009. **Chapter 3**.
- Russel and Norvig, Artificial Intelligence: A Modern Approach, 3rd edition, Prentice Hall, 2010. **Chapter 3.3**