EECS 3401 — AI and Logic Prog. — Lecture 10

Adapted from slides of Yves Lesperance

Vitaliy Batusov
vbatusov@cse.yorku.ca

York University

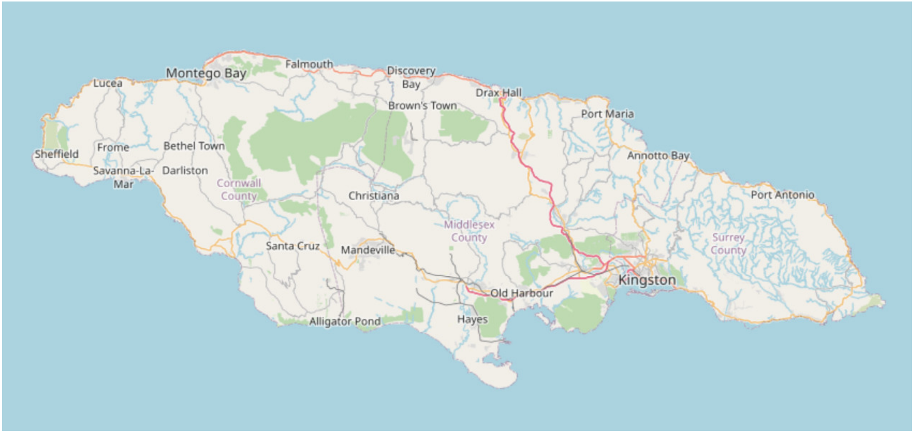
October 21, 2020

- Today: **Search**
- Required reading: Russell & Norvig Chapters 3.1–3.4

Why Search

- Successful
  - In game-playing applications
  - In many other practical AI applications
- Practical
  - Many problems don't have a simple algorithmic solution. Casting these problems as search problems is often the easiest way of solving them. Search can also be useful in approximation (e.g., local search in optimization problems)
  - Specialized algorithms often cannot be easily modified to take advantage of extra knowledge. Heuristics in search provide a natural way of utilizing extra knowledge.
- Some critical aspects of intelligent behaviour (e.g., planning) can be naturally cast as search

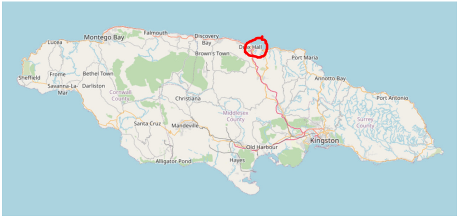
Example: a Holiday in Jamaica



Example: a Holiday in Jamaica

Things to consider:

- Prefer to avoid the hurricane season
- Rules of the road are different, large vehicles have the right of way
- Want to climb up to the top of Dunn's river falls

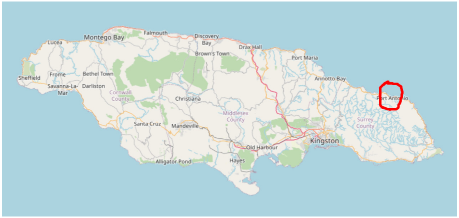


- But want to start your climb at 8:00AM, before the crowds arrive!

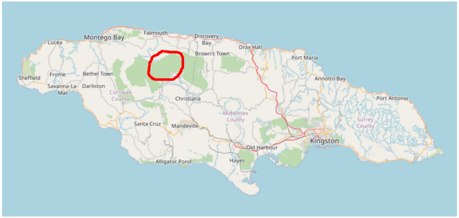
Example: a Holiday in Jamaica

Things to consider (continued):

- Want to swim in the Blue Lagoon



- Want to hike the Cockpit Country

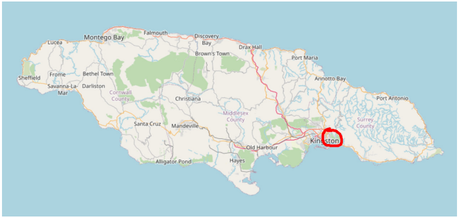


No roads, need local guide and supplies

Example: a Holiday in Jamaica

Things to consider (continued):

- Easier goal: climb to the top of Blue Mountain



  - Near Kingston
  - Organized hikes available
  - But need to arrive on the peak at dawn, before the fog sets in

How to Plan a Holiday

- Must take into account various preferences and constraints to develop a schedule
- An important technique in developing a schedule is “hypothetical” reasoning
    - *If I fly into Kingston and drive a car to Port Antonio, I’ll have to drive on the roads at night. How desirable is this?*
    - *If I’m in Port Antonio and leave at 6:30am, I can arrive at Dunns river falls by 8:00am.*
- To decide *“what do I do”*, must first take stock of *“what can I do”*, and pick the most desirable option

Hypothetical Reasoning

- Hypothetical reasoning involves asking: *what state will I be after the following sequence of events?*
- From this, we can reason about what sequence of events one should try to bring about to achieve a desirable state
- **Search** is a computational method for capturing a particular version of this kind of reasoning

Limitations of Search

- There are many difficult questions that are **not** resolved by search
- E.g., the whole question of how does an intelligent system formulate its problem as a search problem is not addressed by search
- Search only shows how to solve the problem *once we have it correctly formulated*

The Formalism

To formulate a problem as a search problem, we need the following components:

- Formulate a **state space** over which to search

  The state space necessarily involves **abstracting** the real problem
- Formulate **actions** that allow one to move between different states

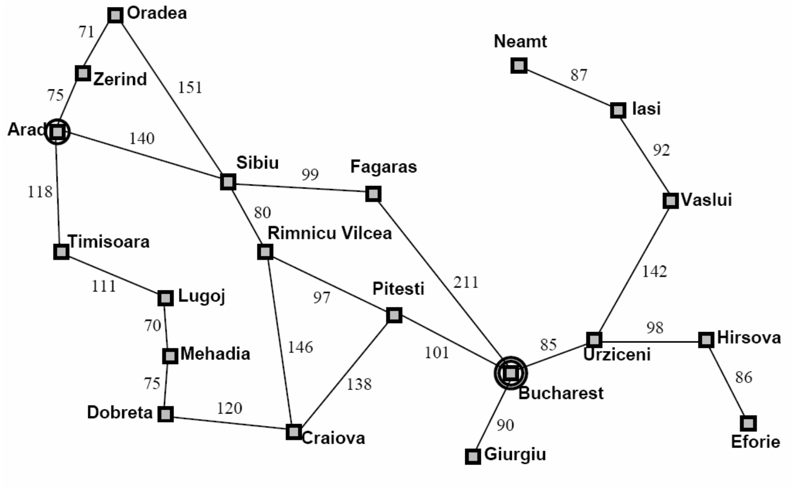
  The actions are abstractions of actions you could actually perform
- Identify the **initial state** that best represents your current state and the **desired condition** one wants to achieve
- Formulate various **heuristics** to help guide the search process

The Formalism

- Once the problem has been formulated as a state space search problem, various algorithms can be utilized to solve it
- A solution to the problem will be a sequence of actions/moves that can transform your current state into a state where your desired condition holds

Example 1: Travelling in Romania

Currently in Arad, need to get to Bucharest by tomorrow to catch a flight



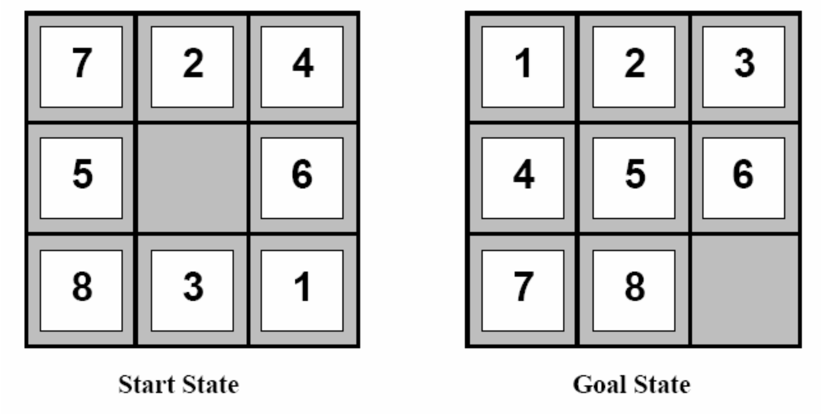
Example 1: Travelling in Romania

- State space!

States	The various cities you could be located in Note abstraction: we are ignoring the low-level details of driving, states where you are on the road between cities, etc.
Actions	Drive from one city to next
Initial State	In Arad
Goal	Be in a state where you are in Bucharest How many states satisfy this condition?

- Solution will be the route—the sequence of cities to travel through to get to Bucharest

Example 2: The 8-Puzzle



Can slide a tile into the blank spot. (Can also thing about this as moving the "blank spot" around.)

Example 2: The 8-Puzzle

- State space

States	Each unique configuration of the tiles on the board is a state. How many different states?
Actions	Moving the blank up, down, left, or right Can every action be performed in every state?
Initial State	As shown on previous slide
Goal	Be in a state where the tiles are all in the positions shown on the previous slide

- Solution will be a sequence of moves of the blank that transform the initial state to a goal state

Example 2: The 8-Puzzle

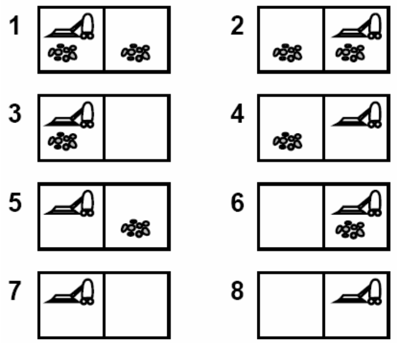
- Although there are $9! = 362880$ different configurations of the tiles, in fact, the state space is divided into two disjoint parts
- Only when the blank is in the middle are all four actions possible
- Our goal condition is satisfied by only a single state. However, we might as well get a goal condition like "Tile #8 is in the upper left-hand corner". (How many states fit that description?)

Example 3: Vacuum World

- In the previous two examples, a state in the search space corresponded to a unique state of the world (modulo the details we have abstracted away)
- However, states need not map directly to world configurations. Instead, a state could map to the agent's **mental** conception of how the world is configured: the agent's **knowledge** state

Example 3: Vacuum World

- Have a vacuum cleaner and two rooms
- Each room may or may not be dirty
- The vacuum cleaner can move left or right (the action has no effect if there's no room to the left/right)
- The vacuum cleaner can suck, with the effect of making the room clean (even if it was clean to begin with)

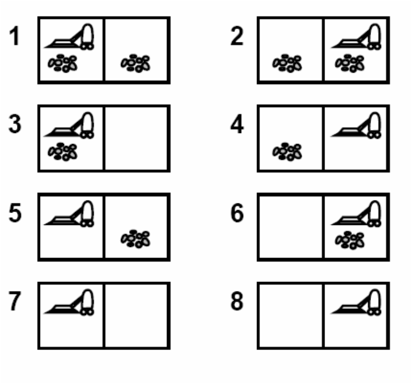


Physical states

Example 3: Vacuum World

Knowledge-level State Space

- The state space can consist of **sets** of states
- The agent knows that it is in one of these states, but doesn't know which

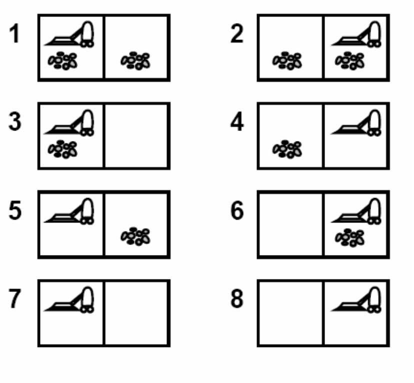


Goal: all rooms clean

Example 3: Vacuum World

Knowledge-level State Space

- Complete knowledge of the world: agent knows exactly which state it is in
- State space consists of singleton states
- Start in $\{5\}$: $\langle right, suck \rangle$

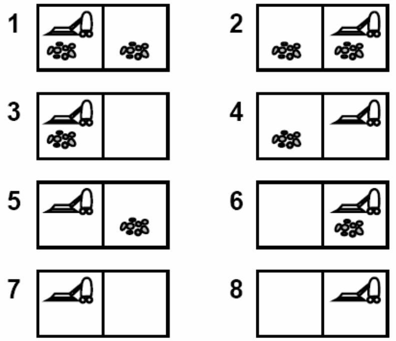


Goal: all rooms clean

Example 3: Vacuum World

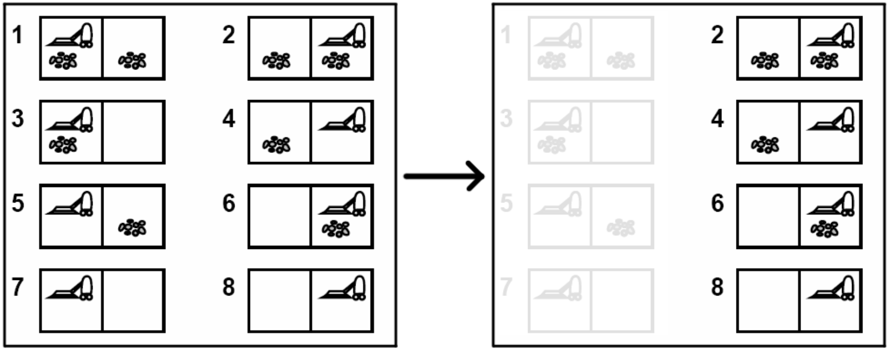
Knowledge-level State Space

- No knowledge of the world. States are sets of physical states
- Start in $\{1, 2, 3, 4, 5, 6, 7, 8\}$ (agent has no idea of where it is)
- Nevertheless, the actions $\langle \mathit{right}, \mathit{suck}, \mathit{left}, \mathit{suck} \rangle$ achieve the goal



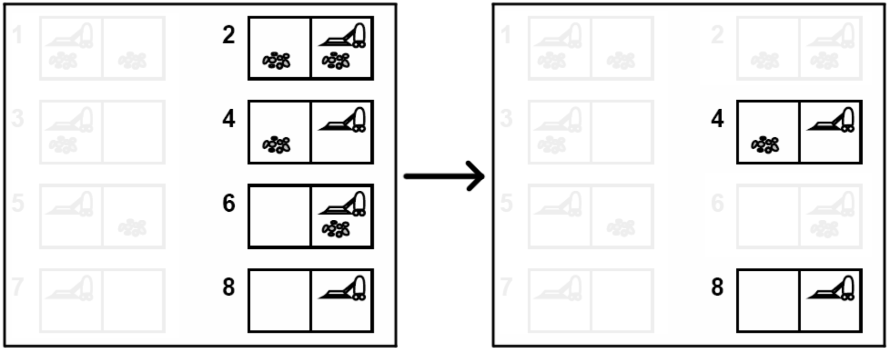
Goal: all rooms clean

Example 3: Vacuum World



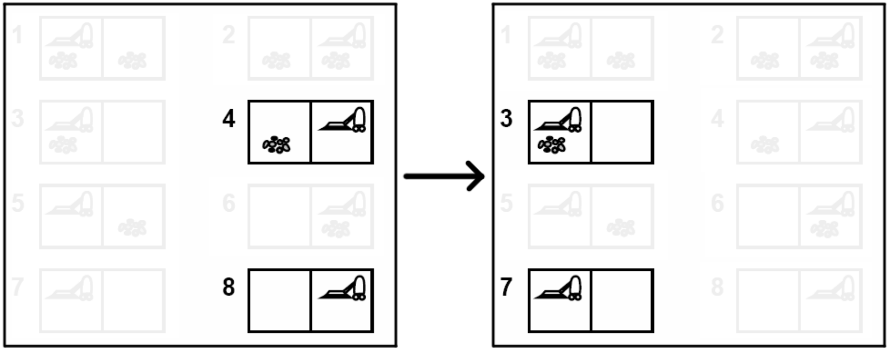
Action: **right**

Example 3: Vacuum World



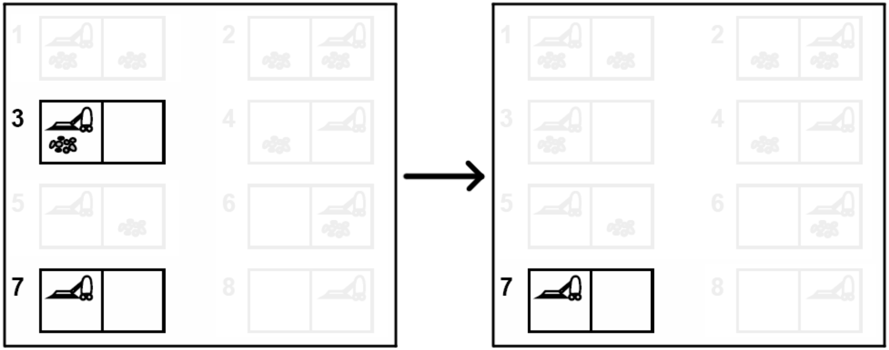
Action: **suck**

Example 3: Vacuum World



Action: **left**

Example 3: Vacuum World



Action: **suck**

More Complex Cases

- The agent might be able to perform some sensing actions. These actions change the agent's mental state, but not the world configuration
- With sensing, can search for a **contingent** solution: a solution that is contingent on the outcome of the sensing acitons
  $\langle \textit{right}, \text{if } \textit{dirt} \text{ then } \textit{suck} \rangle$
- Brings into play the issue of interleaving execution and search

More Complex Cases

- Instead of complete lack of knowledge, the agent might think that some states of the world are more **likely** than others
- This leads to probabilistic models of the search space and different algorithms for solving the problem
- Will see some techniques for reasoning and making decisions under uncertainty later in the course

Algorithms for Search

Inputs:

- an **initial state**—a specific world state or a set of world states representing the agent's knowledge, etc.
- a **successor function** $S(x) =$ a set of states that can be reached from state x via a single action
- a **goal test**—a function that can be applied to a state and returns *True* if the state satisfies the goal condition
- a **step cost function** $C(x, a, y)$ which determines the cost of moving from state x to state y using action a.
  $C(x, a, y) = \infty$ if a does not lead to y from x

Algorithms for Search

Output:

- a sequence of states leading from the initial state to a state satisfying the goal test
- The sequence might be
  - annotated by the name of the actions used;
  - optimal in cost (for some algorithms)

Algorithms for Search

Obtaining the action sequence:

- The set of successors of a state x might arise from different actions, e.g.,
  - $x \Rightarrow a \Rightarrow y$
  - $x \Rightarrow b \Rightarrow z$
- Successor function $S(x)$ yields a set of states that can be reached from x via any single action
- Rather than just return a set of states, we might want to annotate these states by the action used to obtain them:
  - $S(x) = \{\langle y, a\rangle, \langle z, b\rangle\}$
    y via action a, z via action b
  - $S(x) = \{\langle y, a\rangle, \langle y, b\rangle\}$
    y via action a, also y via action b

End of Lecture

- Next time: **Midterm!**
- After that: **Algorithms for Search**