COSC1125/1127 Artificial Intelligence

School of Science (Computer Science)
RMIT University
Semester 1, 2018
A/Prof. Sebastian Sardina

Tutorial Sheet 1
Search

- Open Discussion: *What is Artificial Intelligence?*
- Watch the videos:
  - Holy Grail of AI: https://www.youtube.com/watch?v=tlS5Y2vm02c
  - Humans Need Not Apply: https://www.youtube.com/watch?v=7Pq-S557XQU
  - Artificial Intelligence: https://www.youtube.com/watch?v=oYqXQw2CryI
  - The long-term future of AI: https://www.youtube.com/watch?v=CK5w3wh4G-M

Exercises

1. (RN) Define in your own words the following terms: state, state space, search tree, search node, goal, action, successor function, and branching factor.

Answer

A **state** is a situation that an agent can find itself in. We distinguish two types of states: world states (the actual concrete situations in the real world) and representational states (the abstract descriptions of the real world that are used by the agent in deliberating about what to do).
A **state space** is a graph whose nodes are the set of all states, and whose links are actions that transform one state into another.
A **search tree** is a tree (a graph with no undirected loops) in which the root node is the start state and the set of children for each node consists of the states reachable by taking any action.
A **search node** is a node in the search tree.
A **goal** is a state that the agent is trying to reach.
An **action** is something that the agent can choose to do.
A **successor function** described the agents options: given a state, it returns a set of (action, state) pairs, where each state is the state reachable by taking the action.
The **branching factor** in a search tree is the number of actions available to the agent.

2. (RN) Whats the difference between a world state, a state description, and a search node? Why is this distinction useful?

Answer

A world state is how reality is or could be. In one world state were in Arad, in another were in Bucharest. The world state also includes which street were on, whats currently on the radio, and the price of tea in China. A state description is an agents internal description of a world state. Examples are In(Arad) and In(Bucharest). These descriptions are necessarily approximate, recording only some aspect of the state.
We need to distinguish between world states and state descriptions because state description are lossy abstractions of the world state, because the agent could be mistaken about21 how the world is, because the agent might want to imagine things that arent true but it could make true, and because the agent cares about the world not its internal representation of it. Search nodes are generated during search, representing a state the search process knows how to reach. They contain additional information aside from the state description, such as the sequence of actions used to reach this state. This distinction is useful because we may generate different search nodes which have the same state, and because search nodes contain more information than a state representation.

3. Consider this problem: We have one 3 litre jug, one 5 litre jug and an unlimited supply of water. The goal is to be able to drink exactly one litre. Either jug can be emptied or filled, or poured into the other.

   For this problem give:

   (a) An appropriate data structure for representing a state.

   Answer

   The representation of the state can be simply: (nL, nR), where nL is the number of litres in the 3 litre jug, and nR is number of litres in the 5 litre jug.

   (b) The initial state.

   Answer

   The initial state is $(0, 0)$.

   (c) All the final goal state(s).

   Answer

   The goal states are $(1, n)$ or $(n, 1)$, for any number n, as we just need to get some jug with exactly one liter.

   (d) A specification of the operators (or actions) which includes the preconditions that must be satisfied before the operator can be used and the new state generated.

Answer

Operators could be (using FOL sentences):

$\underline{Fill(X)}$
Action: fill jug X
Precondition: jug X is not full
State generated: jug X is full - $(3, nR)$ or $(nL, 5)$

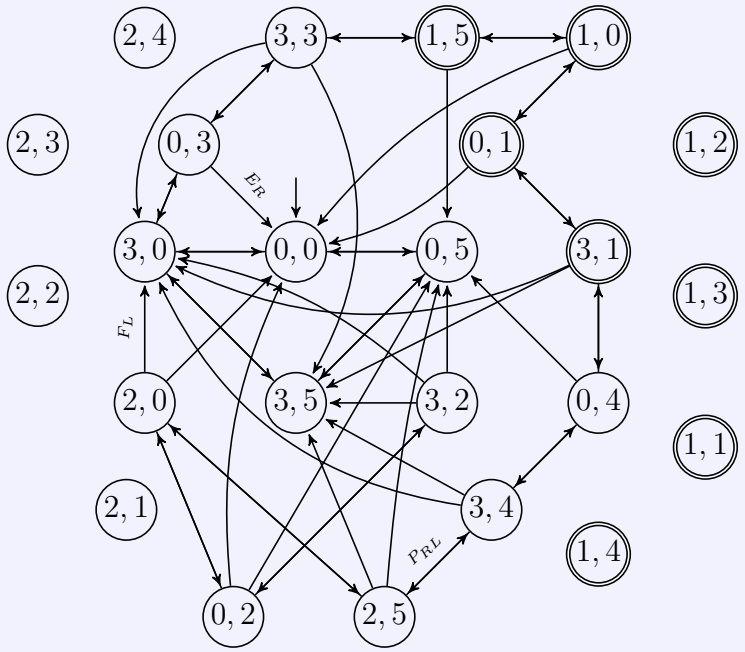
$\underline{Pour(X, Y)}$
Action: pour jug X into jug Y until either X is empty or Y is full
Precondtion: jug X is not empty and jug Y is not full
State Generated:

- jug X is empty and jug Y is partially full - $(0, nR)$ or $(nL, 0)$, or
- jug X is partially full and jug Y is full - $(3, nR)$ or $(nL, 5)$

$\underline{Empty(X)}$
Action: empty jug X
Precondition: jug X is not empty
State generated: jug X is empty - $(0, nR)$ or $(nL, 0)$

(e) Draw the full state space.

Answer



State transition labels:
F_X fill jug X
E_X empty jug Y
P_{XY} pour jug X into jug Y

Note: For clarity, not all state transition labels are not shown on this diagram, usually you should show all state transition labels above all arcs. Observe also that states that cannot be reached from the initial state, such as (1,4), are still part of the full state space. If you are not going to show inaccessible states, you should explicitly state so.

(f) What is the solution to the problem.

Answer

$(0,0) \rightarrow (3,0) \rightarrow (0,3) \rightarrow (3,3) \rightarrow (1,5) \rightarrow (1,0)$

(g) How did you find that solution? Would a computer be able to find it the same way? Explain either way. If not, how would you make an algorithm to find a solution given the search space?

Answer

If you just found it by "matching" a path connecting $(0, 0)$ to any state $(1, n)$ or $(n, 1)$, then it will probably be difficult to do with an algorithm, as it is not systematic approach. An algorithm would have to "visit" nodes in a systematic, regular, complete, and hopefully non-redundant manner.

Check Jonathon's nice Python solution `jug_problem.py` via a simple random search. What would you change there to make it non-random and systematic? How would you change it so as not to need a depth bound?

4. Does a finite state space always lead to a finite search tree? How about a finite state space that is a tree?

Answer

Finite state can yield an infinite state if there are loops in the state space; if the state space is a tree then there are no loops and the search tree will be finite

5. Consider the problem of getting from Arad to Bucharest in Romania. For this problem give:

   - Search state descriptions.

     Answer

     A possible search state description could be $(\langle city \rangle, g(n))$, where $g(n)$ is the total path cost to that node. (Note that the real search state in a search algorithm will also contain the link to the parent of the state node, but we omit it here.)

   - Initial State.

     Answer

     $(Arad, 0)$

   - Final goal search states.

     Answer

     $(Bucharest, g(n))$, where $g(n)$ is total path cost from Arad to Bucharest.

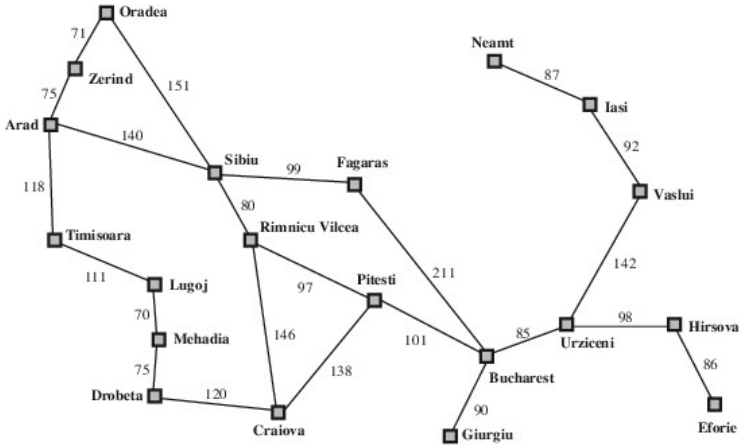
   - Operators (or actions).

Answer

Only one operator exists (using FOL sentences):

$\underline{Go(X, Y)}$
Action: travel from city X to city Y
Precondition: currently in city X and city X has an edge connecting it to city Y
State generated: in city Y - $(Y, g(n))$, where $g(n))$ is total path cost to city Y.

The reason $Go(X)$ is not used, is that it does not take into account the fact that we must travel between connected cities.



Using your representation, how would you build a "search tree" starting from state "Arad"?

6. Why is search an important technique in AI and CS? When would you use search and when you wouldn't? Give concrete examples and reasons.

Answer

It is important because it is an extremely general technique that can solve any (doable) problem, provided it is modelled as a search task.
One would use it when there is no direct technique to solve a problem (e.g., finding the route in a map or playing chess), but one would not use it if an effective algorithm exist for the problme (e.g., sorting a list or computing the factorial of a number), because, while general, is computationally demanding.

7. *You say more?* Lots of cool exercise in RN book, chapter 3....