Solving Problems by Searching

Problem solving is a process of generating solutions from observed data.

- a problem is characterized by a set of goals,
- a set of objects, and
- a set of operations.

These could be ill-defined and may evolve during problem solving.

Searching Solutions:

To build a system to solve a problem:

1. Define the problem precisely
2. Analyze the problem
3. Isolate and represent the task knowledge that is necessary to solve the problem
4. Choose the best problem-solving techniques and apply it to the particular problem.

● State space

The state space representation forms the basis of most of the AI methods.

- Formulate a problem as a state space search by showing the legal problem states, the legal operators, and the initial and goal states.
- A state is defined by the specification of the values of all attributes of interest in the world.
- An operator changes one state into the other; it has a precondition which is the value of certain attributes prior to the application of the operator, and a set of effects, which are the attributes altered by the operator.
- The initial state is where you start.

- The goal state is the partial description of the solution.

Formal Description of the problem:

1. Define a state space that contains all the possible configurations of the relevant objects.

2. Specify one or more states within that space that describe possible situations from which the problem solving process may start (initial state)

3. Specify one or more states that would be acceptable as solutions to the problem. (goal states)

Specify a set of rules that describe the actions (operations) available

State-Space Problem Formulation:

Example: A problem is defined by four items:

1. initial state e.g., "at Arad"

2. actions or successor function : S(x) = set of action–state pairs

e.g., S(Arad) = {<Arad Zerind, Zerind>, … }

3. goal test (or set of goal states)

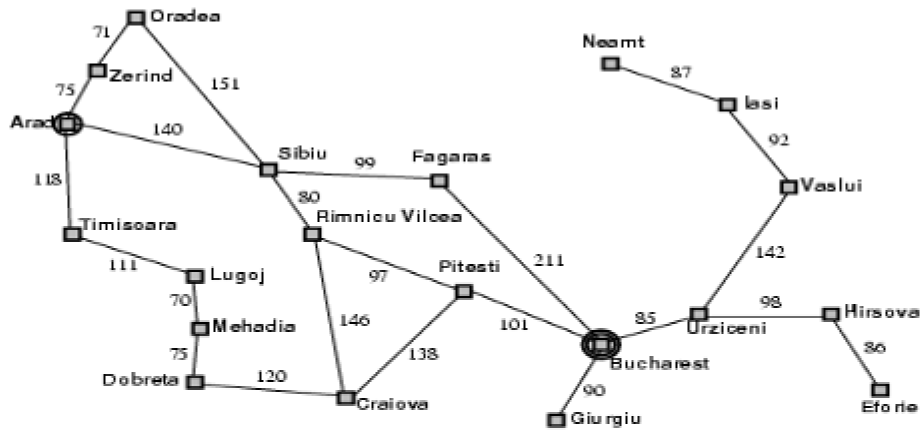
e.g., x = "at Bucharest‖, Checkmate(x)

4. path cost (additive)

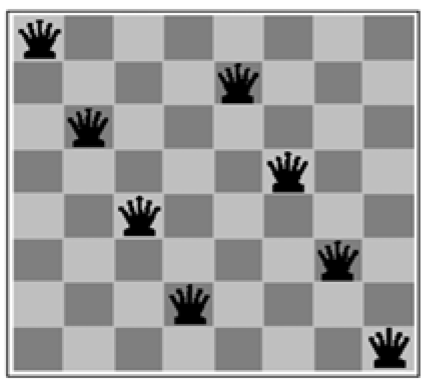
e.g., sum of distances, number of actions executed, etc.

c(x,a,y) is the step cost, assumed to be ≥ 0

A solution is a sequence of actions leading from the initial state to a goal state



Example: 8-queens problem



1. Initial State: Any arrangement of 0 to 8 queens on board.

2. Operators: add a queen to any square.

3. Goal Test: 8 queens on board, none attacked.

4. Path cost: not applicable or Zero (because only the final state counts, search cost might be of interest).

State Spaces versus Search Trees:

State Space

- Set of valid states for a problem
- Linked by operators
- e.g., 20 valid states (cities) in the Romanian travel problem
- Search Tree:
    - Root node = initial state
    - Child nodes = states that can be visited from parent
    - Note that the depth of the tree can be infinite

      E.g., via repeated states – Partial search tree
- Portion of tree that has been expanded so far
- Fringe : Leaves of partial search tree, candidates for expansion
- Search trees = data structure to search state-space

● Solving Problems by Searching

Following are the four essential properties of search algorithms to compare the efficiency of these algorithms:

1. Completeness: A search algorithm is said to be complete if it guarantees to return a solution if at least any solution exists for any random input.
2. Optimality: If a solution found for an algorithm is guaranteed to be the best solution (lowest path cost) among all other solutions, then such a solution is said to be an optimal solution.
3. Time Complexity: Time complexity is a measure of time for an algorithm to complete its task.
4. Space Complexity: It is the maximum storage space required at any point during the search, as the complexity of the problem.