CS60045 Artificial Intelligence Autumn 2023

State Space Search

Terminology

State space: set of all possible states (may be infinite)

State: Each element of the state space

Start state

Goal state(s)

Actions applicable at a state

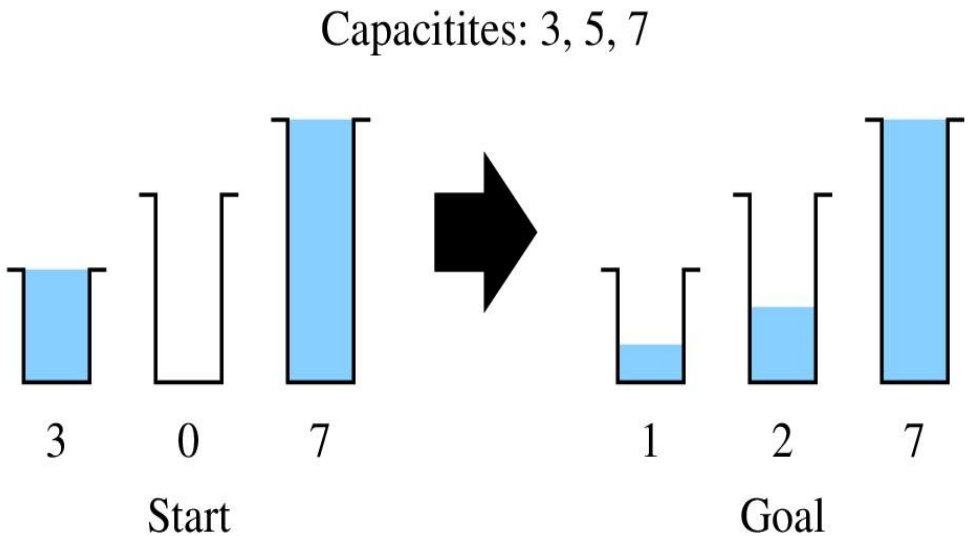
Results of performing the actions at a state (**transitions**)

Cost of each transition

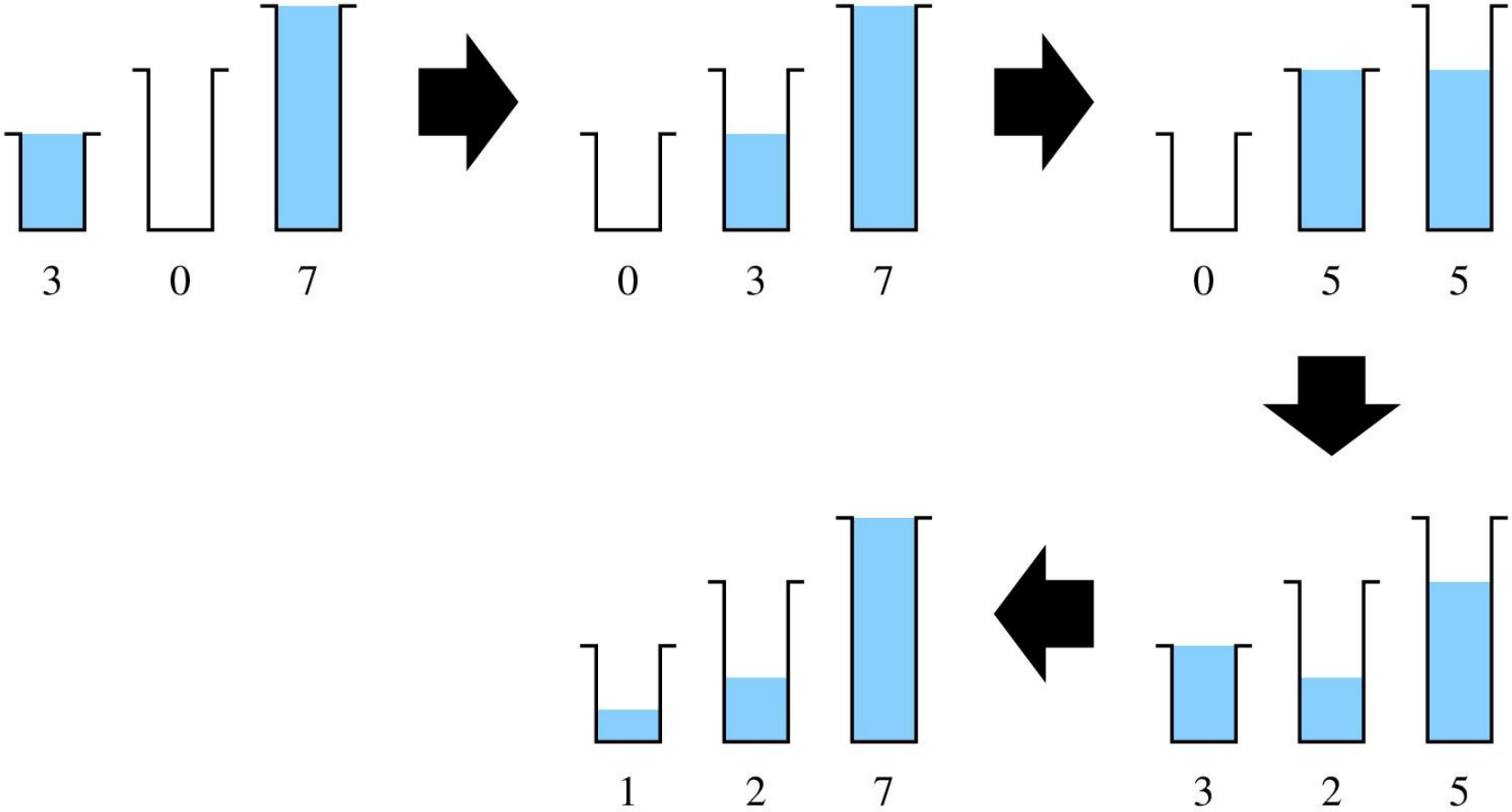
Search Example 1: The problem

- There are three water jugs of capacities *u*, *v*, *w*.
- The jugs have no markings.
- You can transfer water from one jug to another such that
  - Either the source jug becomes empty
  - Or the destination jug becomes full
- Initial contents of the jugs: *a*, *b*, *c*
- Desired contents of the jugs: *r*, *s*, *t*
- $a + b + c = r + s + t$
- *u*, *v*, *w*, *a*, *b*, *c*, *r*, *s*, *t* are integers
- Question: Is it achievable?
- If so, how?
- If not, why?

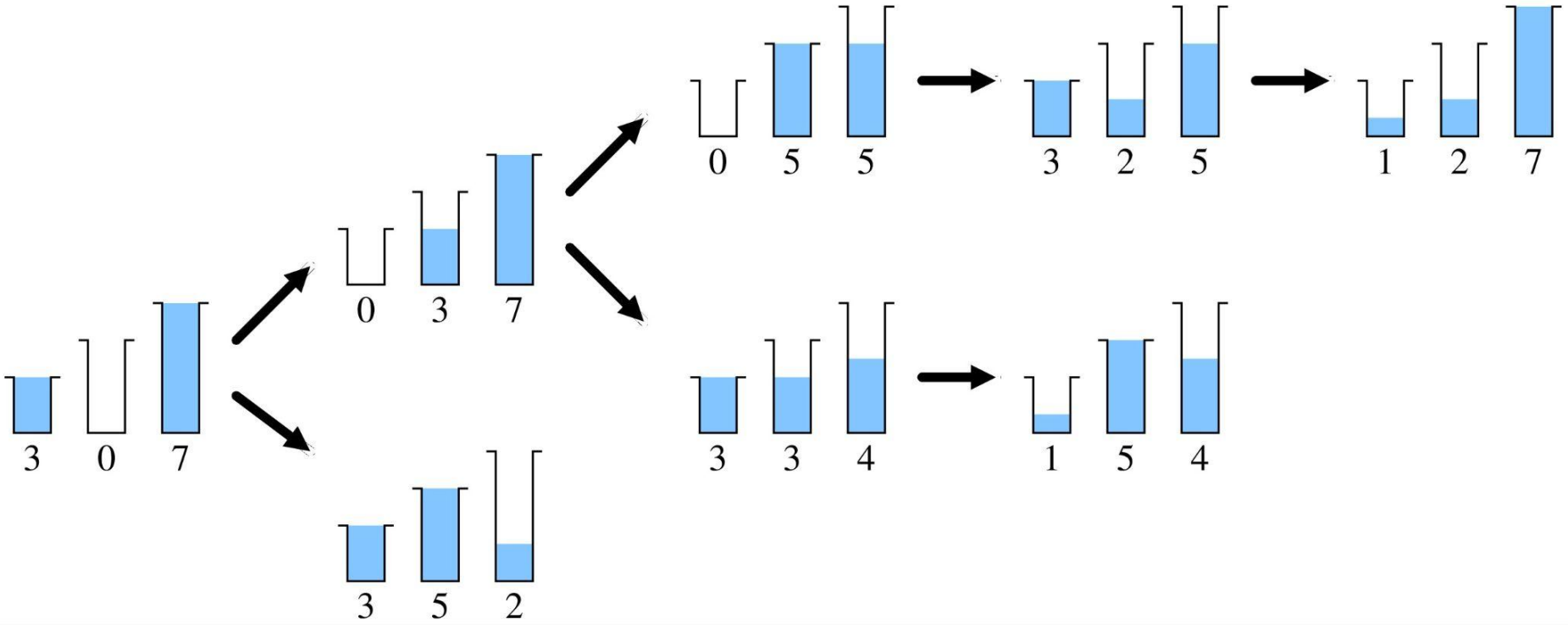
http://cse.iitkgp.ac.in/~abhij/course/lab/Algo1/Autumn16/
Programming Assignment 11



Search Example 1: The solution



Search Example 1: (2, 1, 7) cannot be achieved



Search Example 1: The search space is a graph

The nodes are (x, y, z) with $0 \leqslant x \leqslant 3$, $0 \leqslant y \leqslant 5$, $0 \leqslant z \leqslant 7$, and $x + y + z = 10$.

At least one jug must be full or at least one jug must be empty (except possibly at the beginning).

```
Node  0 : ( 0, 3, 7) ->  2  7 10
Node  1 : ( 0, 4, 6) ->  0  2  8 11
Node  2 : ( 0, 5, 5) ->  0  9 12
Node  3 : ( 1, 2, 7) ->  0  4  7  9
Node  4 : ( 1, 5, 4) ->  2  3 10 12
Node  5 : ( 2, 1, 7) ->  0  6  7  8
Node  6 : ( 2, 5, 3) ->  2  5 11 12
Node  7 : ( 3, 0, 7) ->  0 12
Node  8 : ( 3, 1, 6) ->  1  5  7 12
Node  9 : ( 3, 2, 5) ->  2  3  7 12
Node 10 : ( 3, 3, 4) ->  0  4  7 12
Node 11 : ( 3, 4, 3) ->  1  6  7 12
Node 12 : ( 3, 5, 2) ->  2  7
```

Graph statistics: 13 nodes, 46 edges

Search Algorithm:

Any graph traversal (DFS or BFS)

Notes

- The entire graph is not needed. For example, only 8 nodes are reachable from (3, 0, 7).
- The visited nodes should not be revisited.
- How do we keep track of the visited nodes?

Search Example 1: Where is intelligence here?

- Think of a bigger instance with six jugs of capacities 89, 97, 101, 103, 107, 109 and with amount of water (throughout) equal to 300.
- Assume that we can have any start state.
- There are 5,973,255,042 nodes in the graph. This number does not fit in a 32-bit int.
- Add two new operations
    - You can empty a jug in a sink.
    - You can fill a jug completely from a tap.
- These extra operations mean that the constraint on the total amount of water is no longer applicable.
- Now we can have a total at the end different from the total at the beginning.
- The number of nodes in the graph now becomes 1,111,523,212,800.
- It is impossible to store such a graph in the memory.

Search Example 2: The search space may be infinite

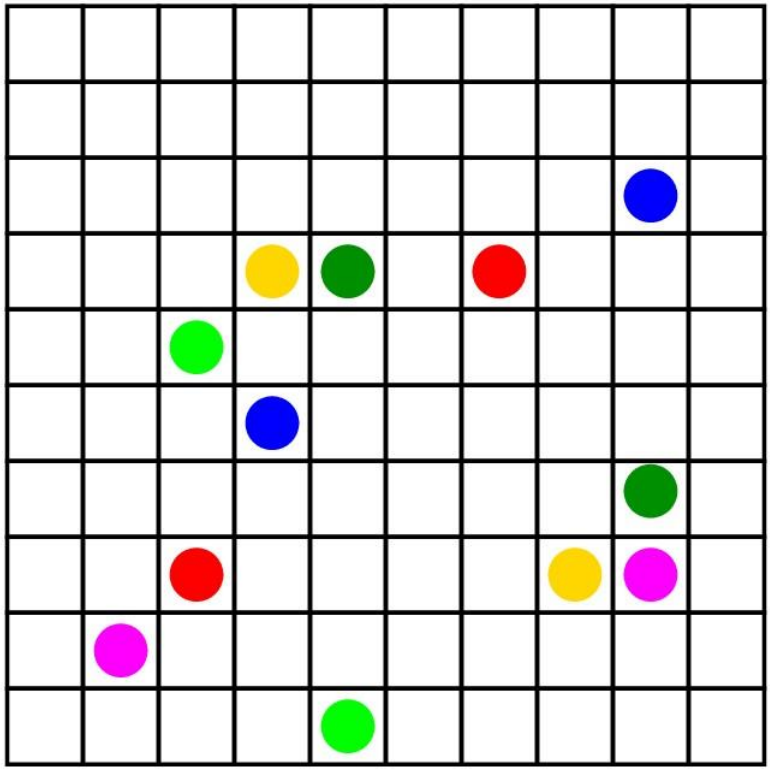
Problem: Start from a positive integer *a*. Generate a positive integer *b*.
Operations permitted: Factorial, square root, floor.
Example: Generate 5 from 4.

4
→ 4! = 24
→ 24! = 620448401733239439360000
→ √620448401733239439360000 = 787685471322.938…
→ √787685471322.938… = 887516.462…
→ √887516.462… = 942.080…
→ √942.080… = 30.693…
→ √30.693… = 5.540…
→ ⌊ 5.540… ⌋ = 5

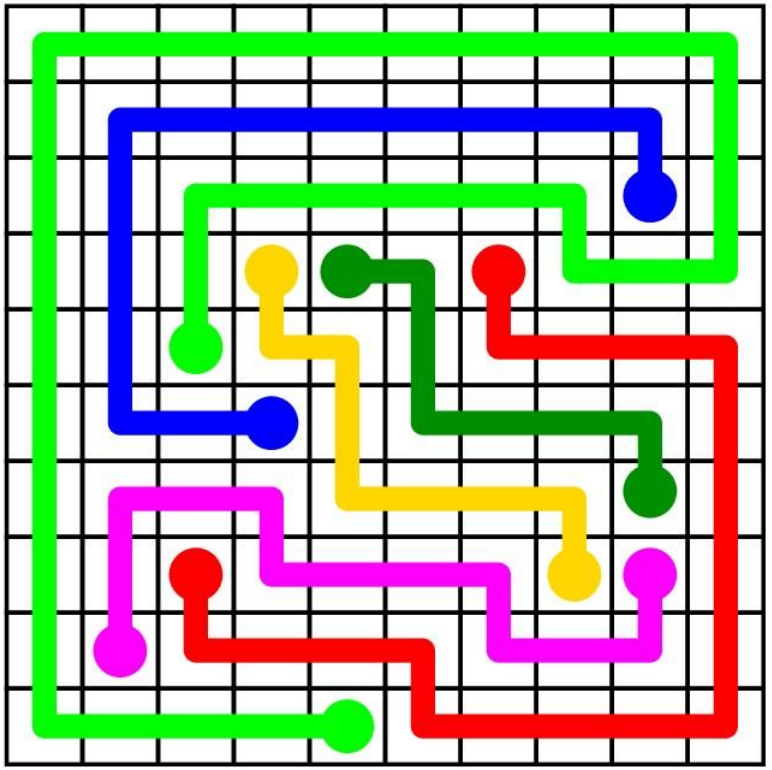
Exercise: Generate 6 from 5.

The search space is the set of all positive real numbers. This is an infinite set.

Search Example 3: Playing Games (Numberlink)



The puzzle



The solution

Search Example 3: Playing Games (Numberlink)

For a general $m \times n$ board, a state is of the form

$$\begin{matrix} c_{11} & c_{12} & c_{13} & \dots & c_{1n} \\ c_{21} & c_{22} & c_{23} & \dots & c_{2n} \\ \dots & & & & \\ c_{m1} & c_{m2} & c_{m3} & \dots & c_{mn} \end{matrix}$$

where each c_{ij} is one of the given c colors or empty. The size of the search space is $(c + 1)^{mn - 2c}$. In our example, this quantity is $7^{88} \approx 2.3368 \times 10^{74}$.

You may discard some invalid states (like those having a square of some color surrounded on all four sides by squares of different colors). There would still be an enormous number of possibilities.

Humans play Numberlink by extending paths of given colors, not after preparing the entire state space.

Search Example 3: Playing Games (Sudoku)

1					8			9
		2						8
	8		5	4	9			
	4		2			9		
3		9				2		1
		1			5		4	
			9	1	2		3	
7						1		
2			7					6

The puzzle

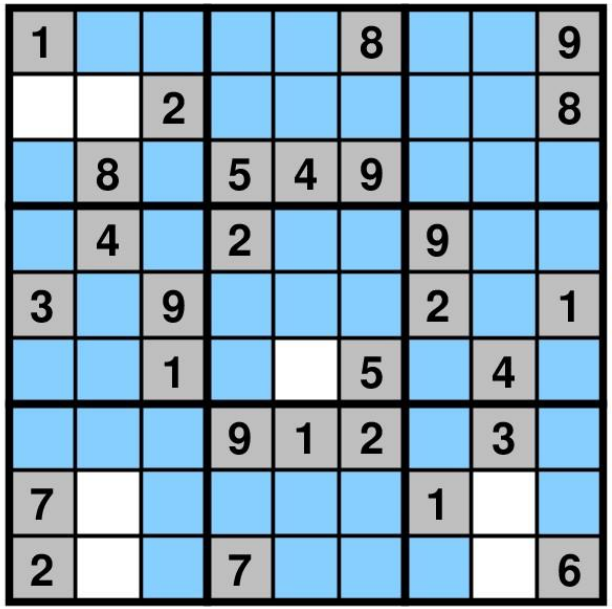
1	3	4	6	2	8	5	7	9
9	5	2	1	3	7	4	6	8
6	8	7	5	4	9	3	1	2
5	4	6	2	7	1	9	8	3
3	7	9	4	8	6	2	5	1
8	2	1	3	9	5	6	4	7
4	6	8	9	1	2	7	3	5
7	9	5	8	6	3	1	2	4
2	1	3	7	5	4	8	9	6

The solution

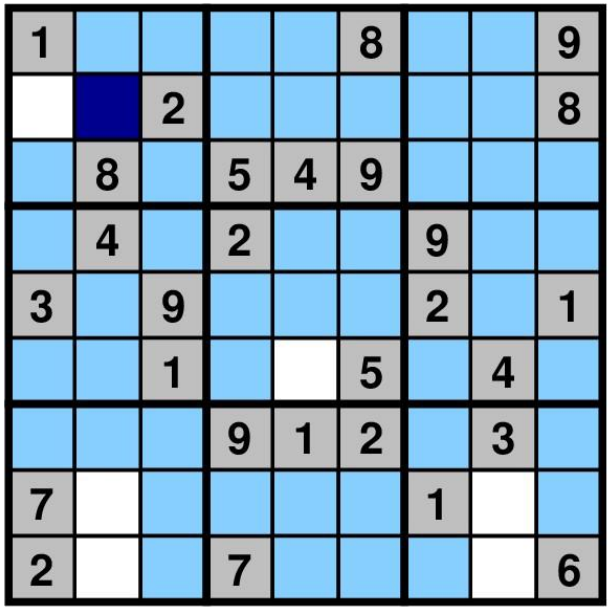
Search Example 3: Playing Games (Sudoku)

The search space consists of boards with additionally filled cells under the constraints imposed by the rules of Sudoku.

Constraints provided by the 9's on the board



Immediate constraints



Derived constraint

Search Example 4: Playing game with (human) opponents

Two-player games like Tic-Tac-Toe, Chess, GO, Poker, … pose additional challenges.

Search space consists of the intermediate configurations (like the positions of the pieces on the chessboard).

The moves of the opponent cannot be controlled by the AI program.

This means some (like half) of the moves are chosen by the opponent.

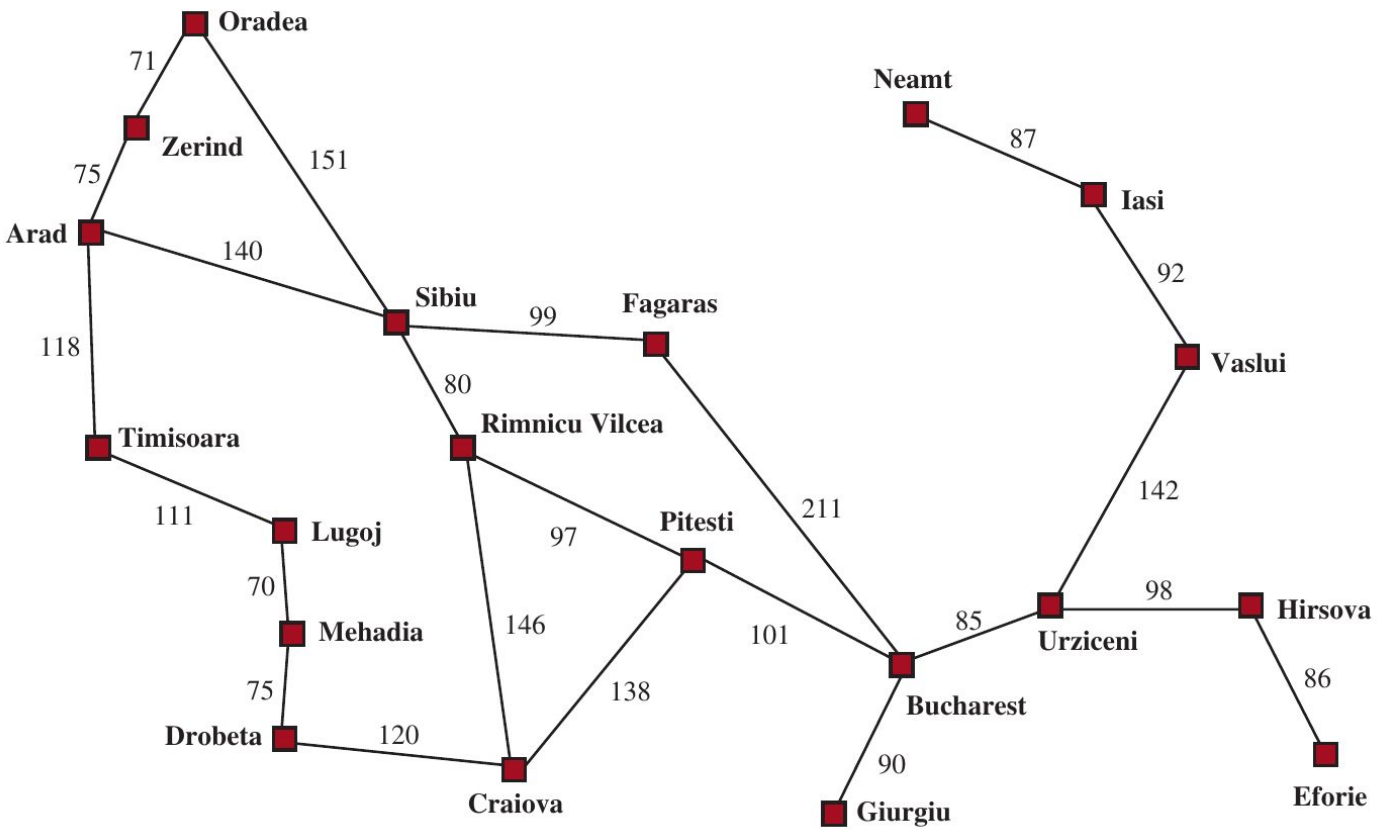
The moves of the opponent are targeted to defeat the AI program.

How can the computer play optimally?

Searches based on game trees help the AI program choose good moves.

Multiplayer games (like Ice Hockey and Robot Soccer) are more complex.

Search Example 5: Traveling in Romania [AIMA]



Task:

Travel from Arad to Bucharest

Objective:

Minimum distance covered

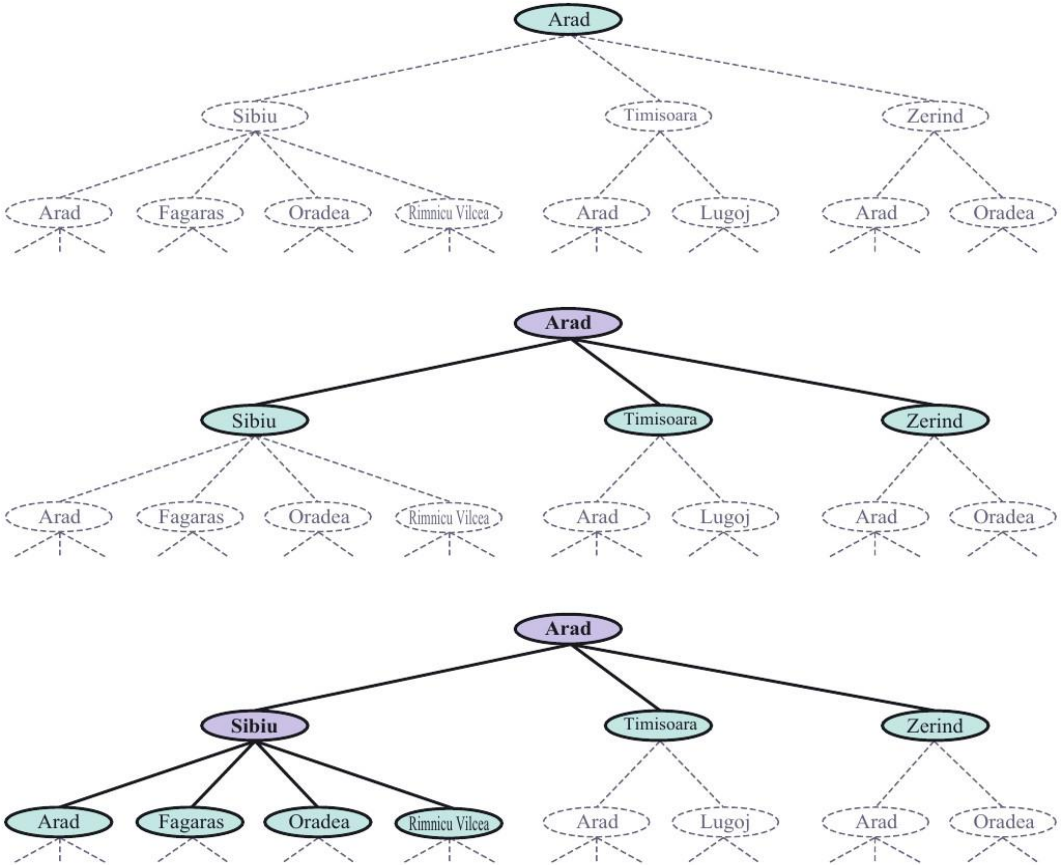
Search space:

Weighted graph with the cities as nodes and the distances as edge weights

Real-world search problems

- Route-finding problems (like the Romania example)
- Tour problems (like the Traveling Salesperson Problem)
- Circuit-layout problem (as used in VLSI design)
- Robot navigation (on complicated terrains, here, on mars, and so on)
- Automatic assembly sequencing
- …

Basic search mechanism: Expand and Explore



Source: AIMA