

Artificial Intelligence

What is it?

- Machines mimicking humans by displaying *intelligent* behavior, learning, and adaptation.
- What is intelligence?
 - being autonomous, reactive
 - being proactive, social
 - ability to solve problems
 - *abstract* from details and *generalize* solutions
 - knowing all facts, rules, detailed information
 - ability to look beyond simple facts, *derive* and *infer* connections and dependencies



Understanding Intelligence

- How is knowledge encoded and acquired?
- How are abstract ideas (mood, feelings, intuitions) represented?
- How is experience gained?
- What are the motivations for acting intelligently?
- Is intelligent action always rational?
- Are all humans intelligent?

Rationality

- Rational action: doing the “right thing”
- The “right thing”: reach the goal in an optimal way, given available information
- Irrational \neq insane, irrationality is making a sub-optimal action
- Rational \neq successful
- In the real world, usually lots of uncertainty and complexity in “available information”



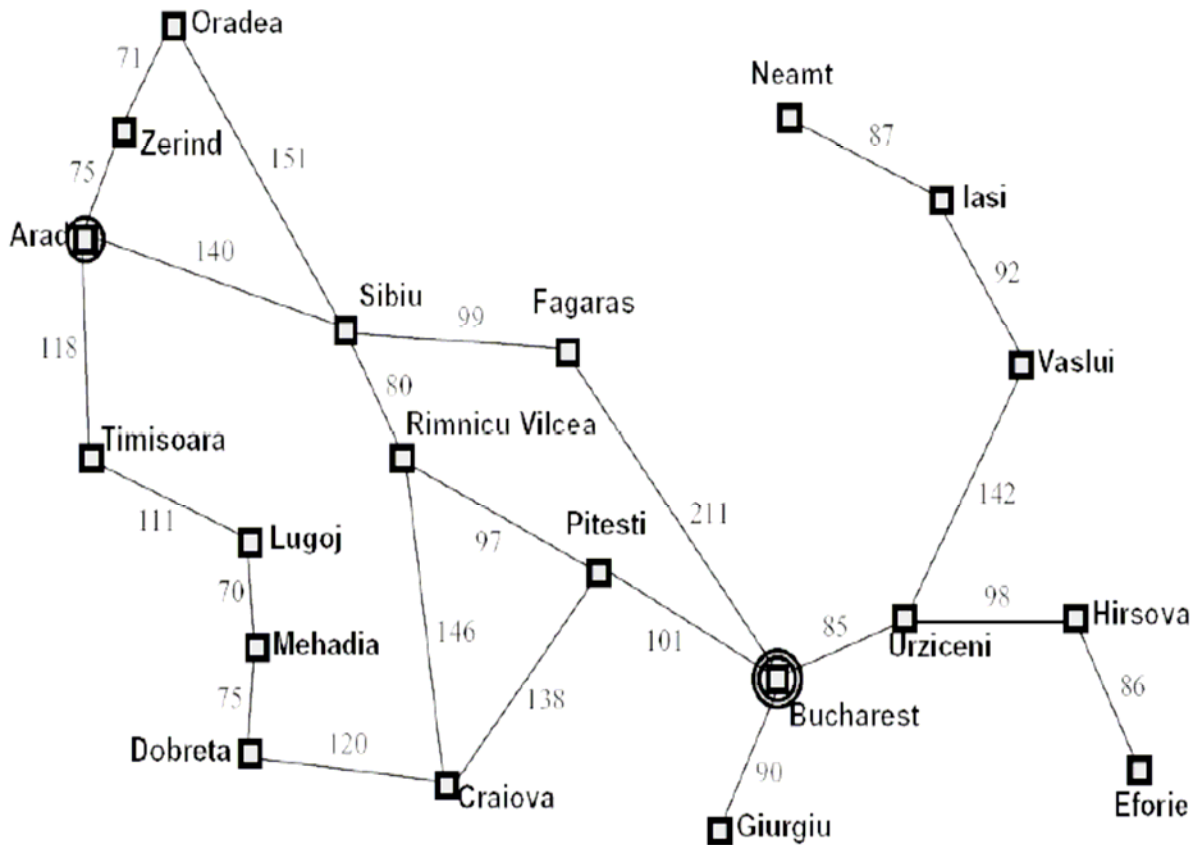
Designing a Rational Agent

- A *rational agent* is an entity that gathers **knowledge** about its **environment** and takes rational **actions** with the purposes of achieving its **goals**
- Given its **goals** and prior **knowledge** about its **environment**, a rational agent should:
 1. Use the information available in new observations of its **environment** to update its **knowledge**, and
 2. Use its **knowledge** to **act** in a way that is expected to achieve its **goals** in the world

Search Problems – Planning

- Planning problem:
 - Searching for sequences of actions that enable reaching the *goal* state from the *start* state
- One of the first successes of early AI research
- Need to specify:
 - States (start state, intermediate states, goal states)
 - Actions (moves from a state to an adjacent state)
 - Method for choosing action at each state

Planning a trip



Problem: Get from Arad to Bucharest as quickly as possible

States: cities

Actions: travel along roads between adjacent cities

Here the search space is known in advance.⁷

The 8 puzzle

7	2	4
5		6
8	3	1

Start State

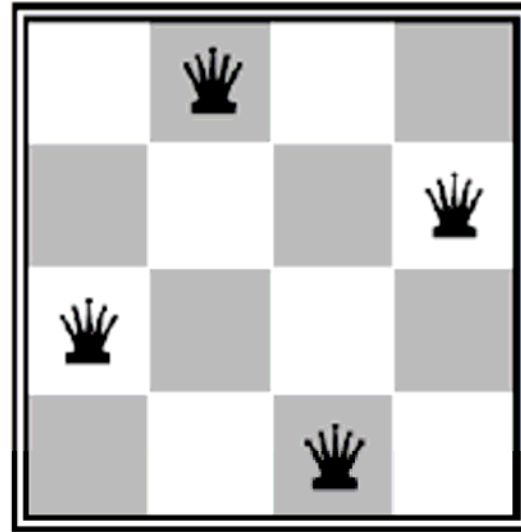
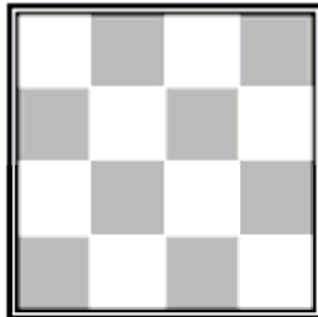
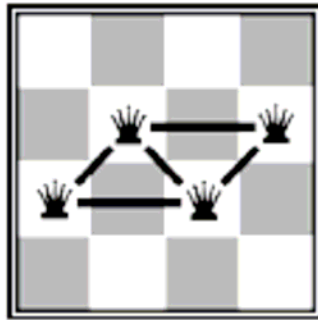
	1	2
3	4	5
6	7	8

Goal State

- **Problem:** reach goal state from start state using as few moves as possible
- **States:** location of the tiles
- **Actions:** move blank left, right, top, down, if possible

The search space could be figured out on the go

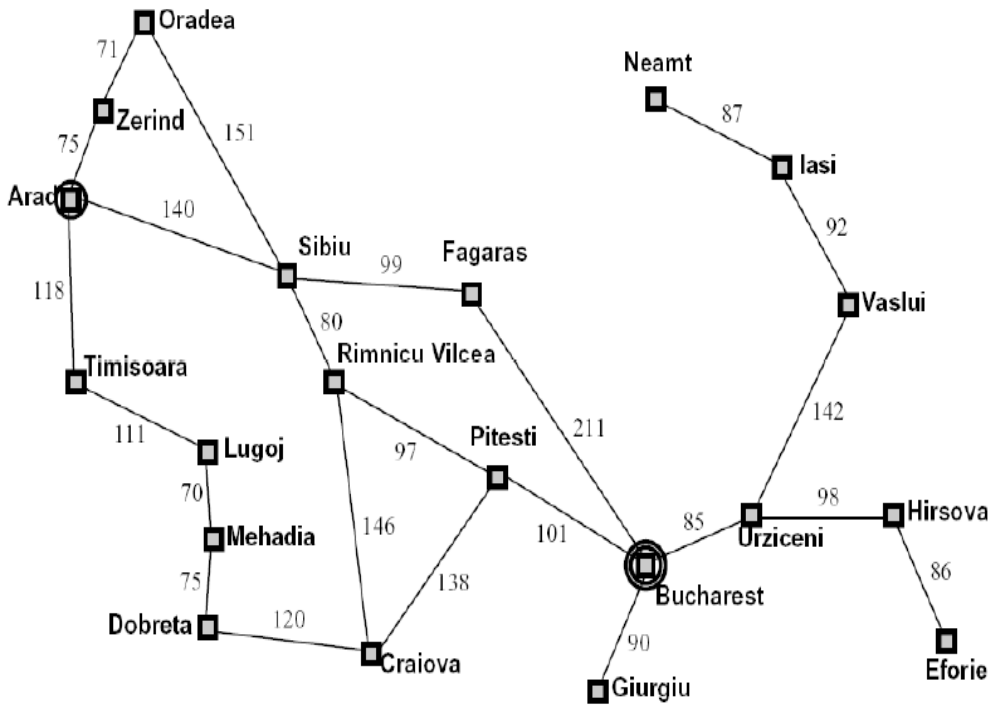
4 queens



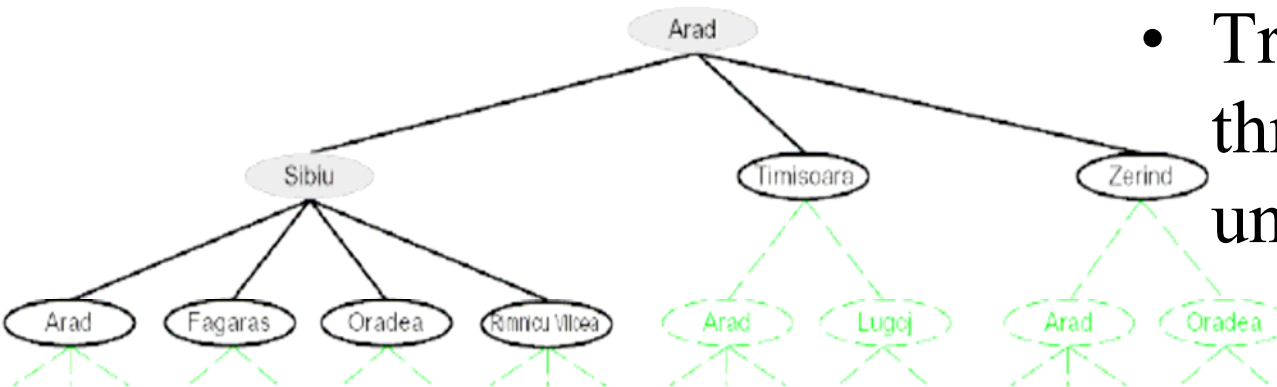
- **Problem:** place 4 queens on a 4x4 chess board so that they don't kill each other
- **States:** valid configurations of queens on a board (including empty board)
- **Actions:** add more queens in a valid location

The search space could be figured out on the go

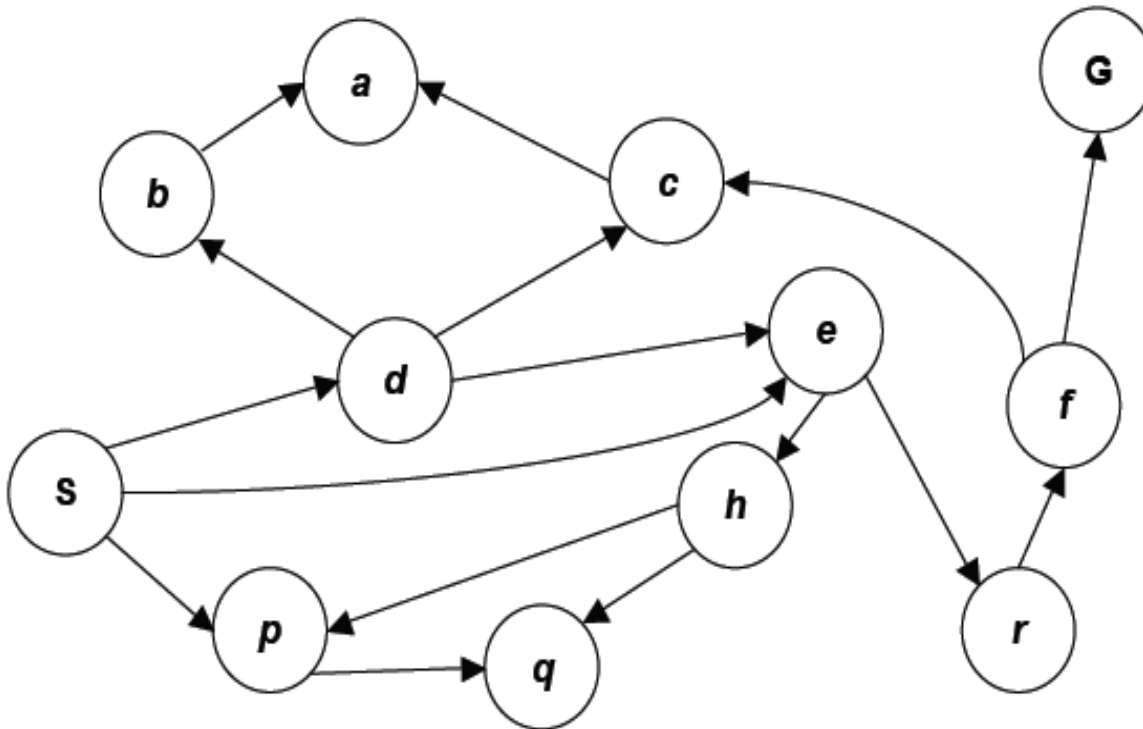
Tree Search



- Represent states (cities) as **vertices**
- The action of traveling from one city to an adjacent city is represented by **edges**
- Travel from Arad through the graph until reach Bucharest

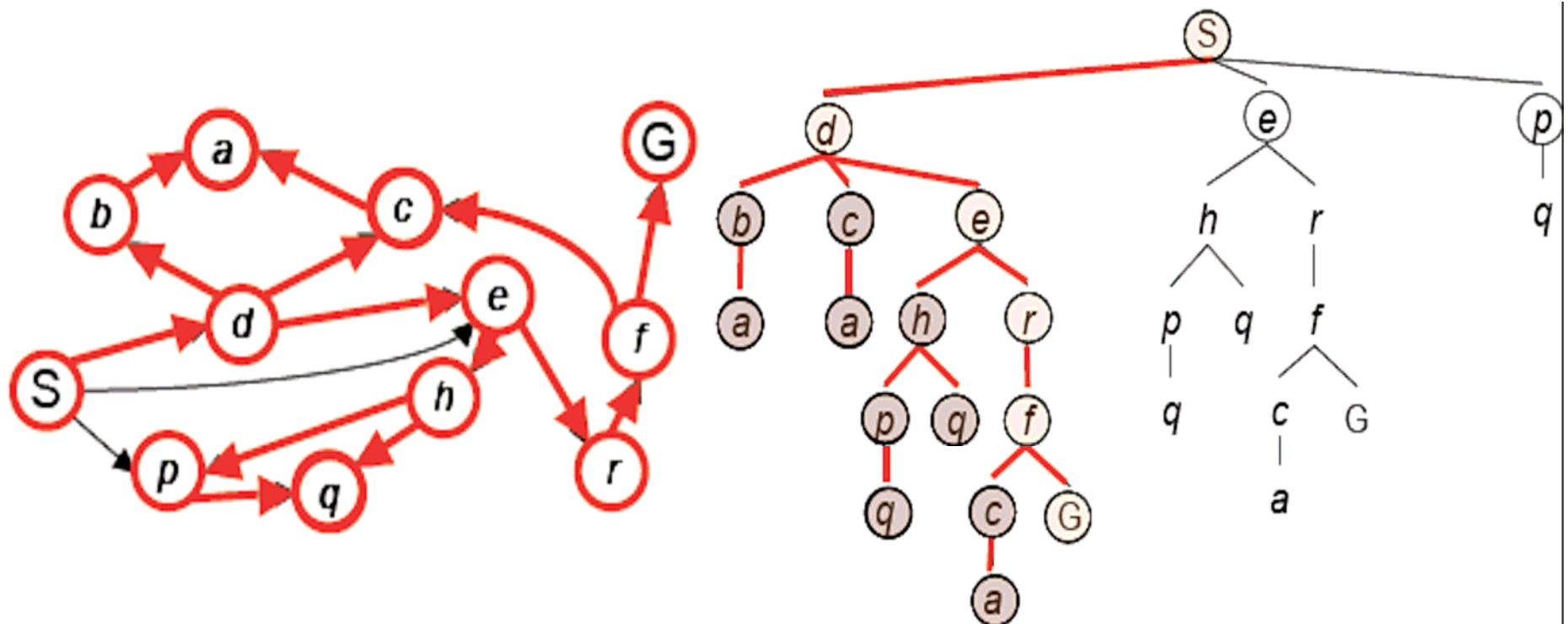


Graph Search



- Given an arbitrary graph, where arrows between vertices represent that a vertex is reachable from another vertex, **get from the start vertex to the end vertex**
- **The graph may or may not be known in advance**

Tree Search – Depth First

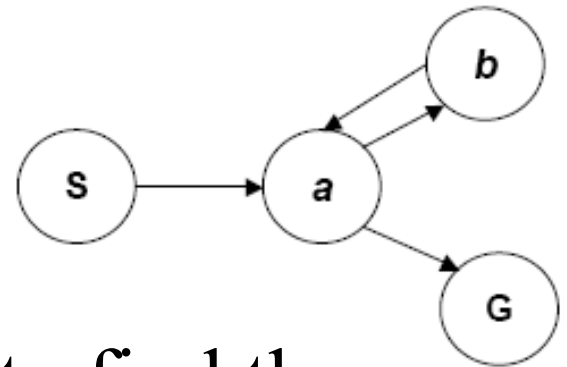


- Starting from the start vertex, do the “leftmost” action, until reach the goal
- If no more “leftmost” actions available, go back up one level in the tree and try the action to the right of the action previously taken. When all these are exhausted, go back up one more level in the tree.

Tree Search – Depth First

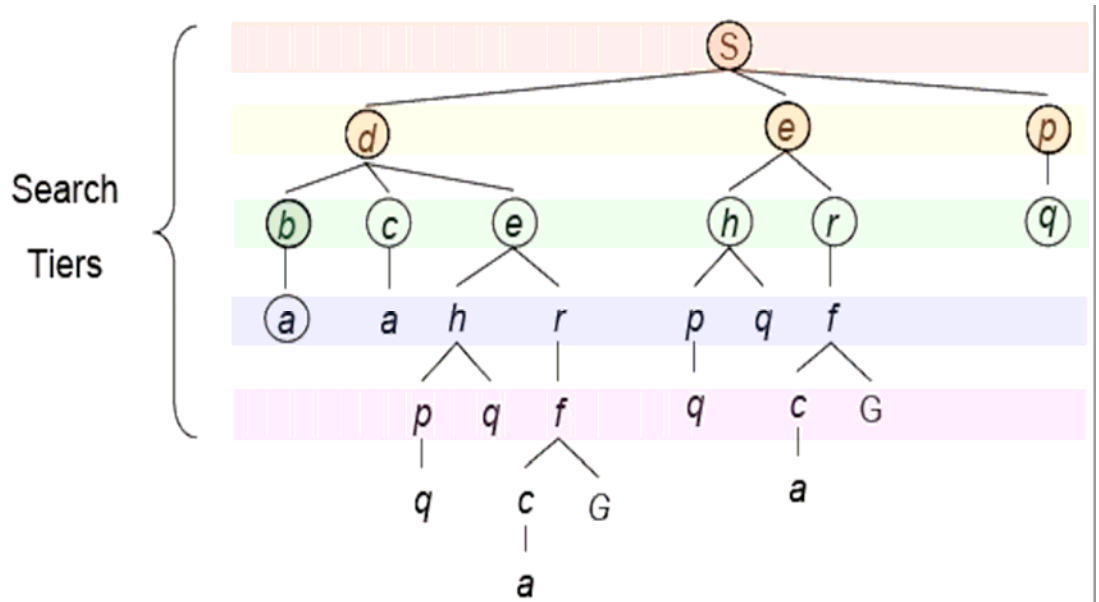
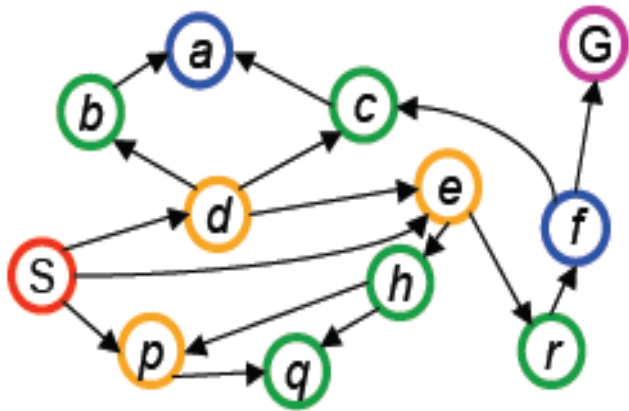
- Why “depth” first?
- If each action has cost ‘1’, is this method guaranteed to find the **cheapest** path to the goal?
- Could it keep going **forever**?

Check for cycles (not returning to previously visited vertex)!



- What method can you suggest to find the cheapest path to the goal?

Tree Search – Breadth First



- Explore the vertices 1 level away from the start position
- Keep exploring all vertices at levels 2, 3, ... until find the goal vertex

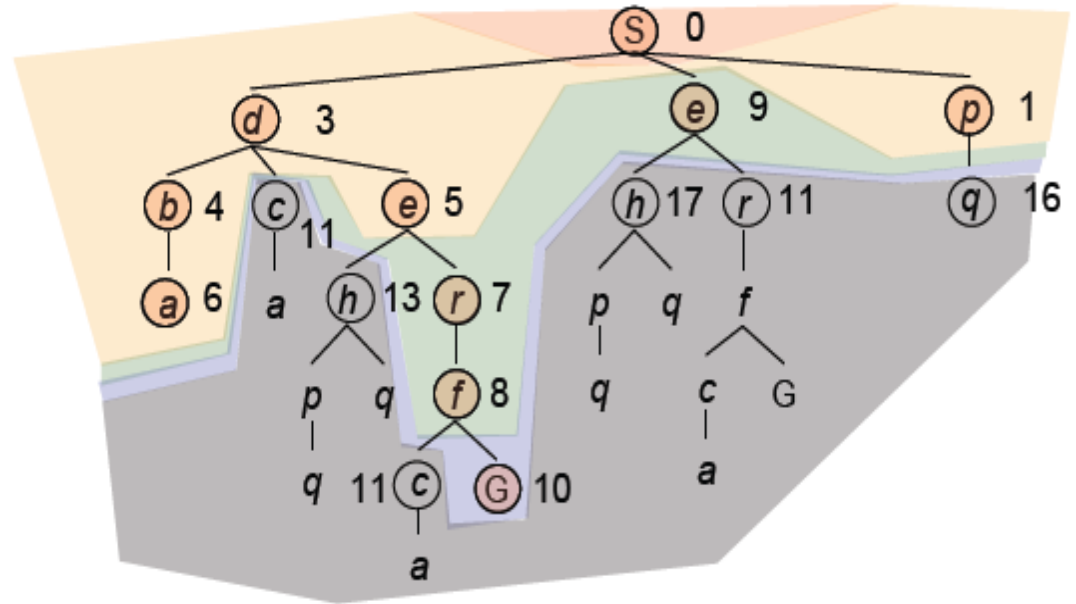
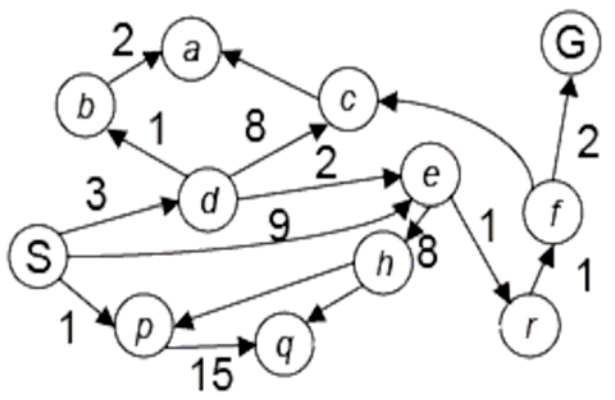
Tree Search – Breadth First

- Why “breadth” first?
- If each action has cost ‘1’, is this method guaranteed to find the **cheapest** path to the goal?
- Could it keep going **forever**?

Comparison of the 2 search algorithms

- When will Breadth First Search find the goal state faster than Depth First Search?
- When will Depth First Search find the goal state faster than Breadth First Search?
- Recall that when considering optimality, assumed that the cost to transition from one vertex to another was 1!

Uniform Cost Search

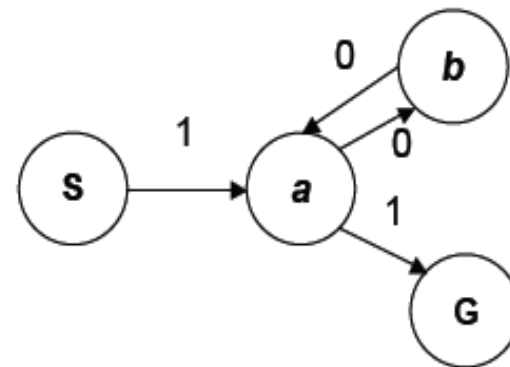


- What about going from Arad to Bucharest?
Here the edges have weights!
- Version of Breadth First Search:
Keep track of cumulative path cost;
consider vertices reachable giving shortest path length first

Uniform Cost Search

- How does this algorithm compare to Breadth First Search when just looking for any path to the goal?
- Could this algorithm keep going forever?

Weights must be positive!



Examples of Intelligent Systems:

- Machine translation from English to Japanese
 - Playing chess against the computer
 - Expert systems for medical diagnosis
 - Predict financial market fluctuations
 - Detect spam
 - Understanding visual information
 - Understanding speech
 - Having a virtual conversation
 - Writing poetry
 - Searching the web
 - Self-navigating robots
- Learning patterns in data and classifying new data
 - Airline scheduling and routing
 - Mapquest

Credits

- Bowen Hui, “Artificial Intelligence, What’s ‘Artificial’ and ‘Intelligent’ about it?”
<http://www.cs.toronto.edu/~bowen/104/intro.pdf>
- Trond Grenager, “Intro to AI, Intro to Search”, CS 121, Stanford University