Artificial Intelligence
Solving Problems By Searching

What have we learnt so far?

* Fundamental Concepts of AI
* Agents
  * How do they differ from each other?
* That it is essential to understand the environment
  * What types of environment did we discuss?
* That an agent is more or less defined by its performance measure
  * What kind of measures did we discuss?
* How can we use PEAS to define the agent program?

Overview for the next 3 Lectures

* Problems and Solutions
* Solving Problems by Searching
* Performance
* Uninformed Search Strategies
* Informed Search Strategies
* Heuristic Strategies
* Online Search Strategies

Overview for this lecture

* Environment represenation (continuation of previous lecture)
* Performance
* State space search
* Uninformed Search Strategies
  * Uniform Cost,
  * Dept-First
  * Depth-Limited
  * Iterative Deepening
* Next Week: Informed Search Strategies

Environment Representation

* First some terminology…
  * Important if you are to go on and study AI in more detail in the future.

* How do you represent the environment the agent inhabits?
  * Atomic Representation – each state is indivisible
  * Factored Representation – attributes and values for states
  * Structured Representation – things in the world are related

Atomic Representation

* Each state of the world is indivisible
  * E.g. finding route from one city to another. It may be enough to have each city as a state
* There is no internal structure (black box)
* Used in:
  * Searching
  * Game Playing (Chess)
  * Hidden Markov Models (Markov process with unobservable states)

Factored Representation

* Each ‘state’ has attributes and values
* When reaching a destination the agent not only knows what the current state is but also what implications restrictions have, such as:
  * Amount of Fuel
  * GPS coordinates etc.
* The same location can have a different state
  * If we take a different route, the values inside the state will be different in a given city
* Factored representation can deal with uncertainty

Structured Representation

* Variables cannot deal with complex implications
  * Things in the world are related to each other
* If we specify relationships between objects, the agent can have an even better understanding of the environment it is inhabiting
  * Agent will have an understanding of what effects different objects behaviour has on the environment.
* Used in:
  * Knowledge-Based Learning
  * Natural Language Understanding

Solving Problems by Searching

* **Problem-solving Agents** are Goal-based Agents!
* Use **atomic representation**
* Agents working on factored or structured representation are usually called '**Planning Agents**'.

Problem Solving Agents

Formulate → Search → Execute

Goal

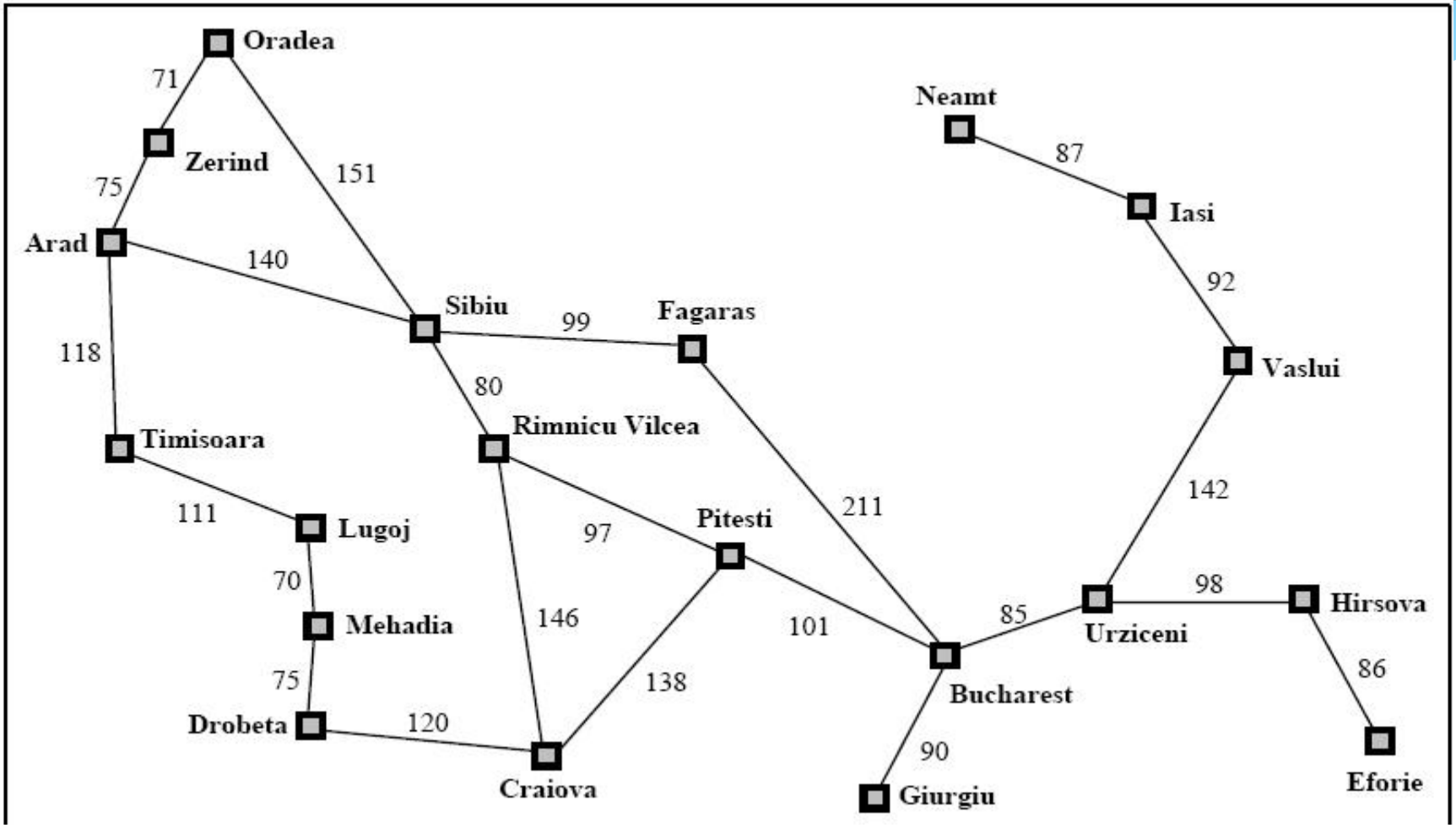
Problem

1. General Purpose Search
2. Uninformed Search
3. Informed Search

A problem is defined as:

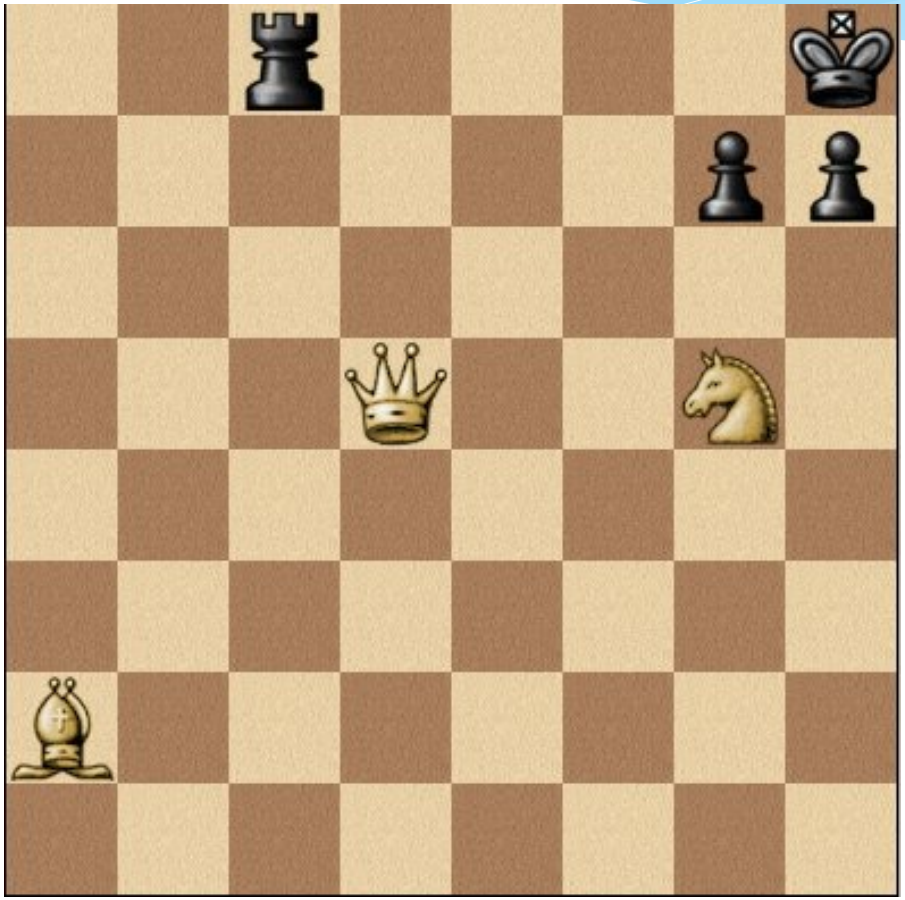
* The **initial state** the agent starts in.
* A description of the **possible actions** available
* A **transition model** that returns the state that results from doing action *a* in state *s*.
* Together the initial state, actions and transition model are defined as the **state space** of a problem. It forms a directed network or graph (see next slide)
  * nodes are states and the links between the nodes are actions.
  * A path in a state space are a sequence of states connected by a sequence of actions.
* The **goal test**, which determines if a given state is a goal.
* A **path costs function** that assigns numeric costs to each path.

Example of a state space



Source: AI: A modern approach p. 68

What about chess?



Mate in 2 moves for white (white to move)

What about…

* Towers of Hanoi

And how can we define a solution?

* A solution to a problem is an action sequence that leads from the initial state to a **goal** state.

* Solution quality is measured by the path cost function, and an optimal solution has the **lowest path cost** among all solutions.

* We need some type of process / algorithm to help us.

Formulating Problems

* This is a key concept of AI and it is what defines a good AI researcher.
* Abstraction is essential
  * The process of removing details from the representation of the problem without harming the problem.

Real-world Problems

* Travel Planning Web Page
* Routing packets through the Internet
* Travelling Salesman Problem (TSP)
  * Each city must be visited once. What's the shortest tour?
* VLSI (Very Large Scale Integration)
  * Positioning components on chip to minimise area required
* Robot Navigation

Travel Planning Web Page – 1/2

* **States:** Each state obviously includes a location (e.g. an airport) and the current time.
  * Furthermore, because the cost of an action (a flight segment) may depend on previous segments, their fare bases, and their status as domestic or international, the state must record extra information about these "historical" aspects.
* **Initial state:** This is specified by the user's query.
* **Actions:** Take any flight from the current location, in any seat class, leaving after the current time, leaving enough time for within-airport transfer if needed.

Travel Planning Web Page – 2/2

* **Transition model:** The state resulting from taking a flight will have the flight's destination as the current location and the flight's arrival time as the current time.

* **Goal test:** Are we at the final destination specified by the user?

* **Path cost:** This depends on monetary cost, waiting time, flight time, customs and immigration procedures, seat quality, time of day, type of airplane, frequent-flyer mileage awards, and so on.

Travel Planning

* Using, https://www.skyscanner.net evaluate the path cost from flying from Dublin to **Brazzaville (BZV), flying from Dublin on the 12th of October and travelling back on Friday the 28th of October, using time. Illustrate your solution by drawing a suitable state space.**

Other Real World Problems

* Travelling Salesperson Problem (TSP)
* Robot Navigation

Performance Measure

* A search algorithms performance can be assessed in four ways:
  * **Completeness:** Is the algorithm guaranteed to find a solution?
  * **Optimality:** Can the algorithm find the optimal solution?
  * **Time complexity:** How long does it take?
  * **Space complexity:** How much memory is needed for the search?

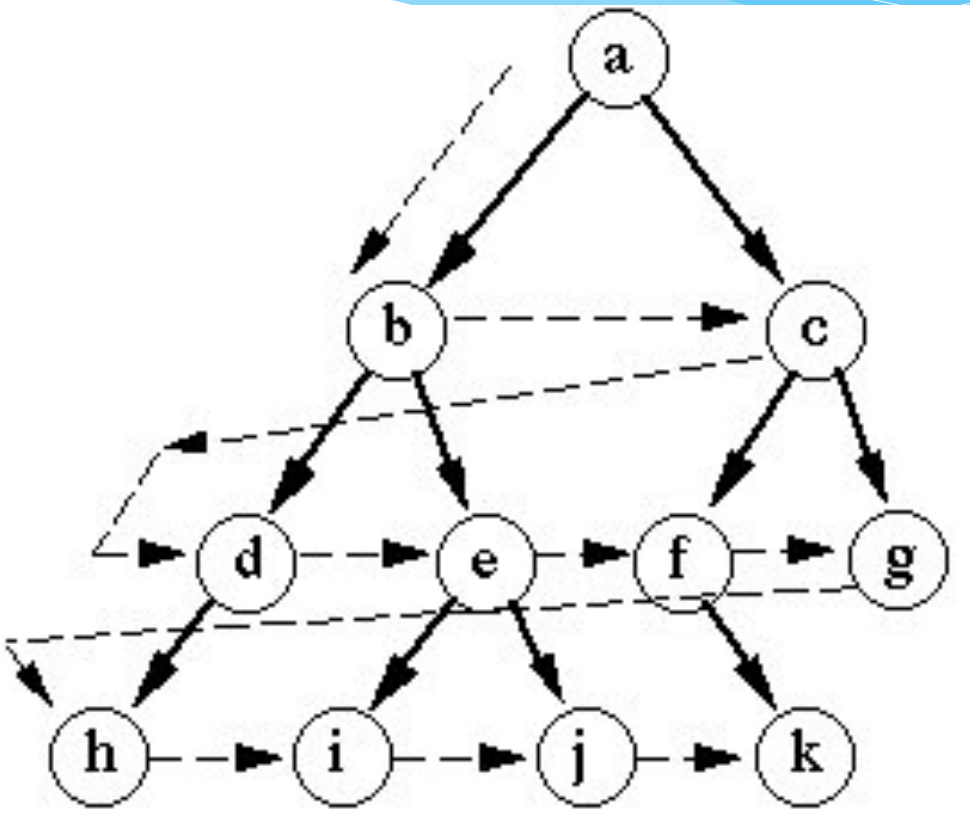
Uninformed Search Strategies

* Breadth First Search
* Uniform Cost Search
* Depth First Search
* Depth Limited Search
* Interactive deepening depth-first search

Breadth First Search

* **Breadth-first search** (**BFS**) is a graph search algorithm
  * begins at the root node and explores all the neighboring nodes.
  * Then for each of those nearest nodes, it explores their unexplored neighbor nodes, and so on, until it finds the goal.
* It is an exhaustive search algorithm.
* Disadvantage: exponential memory requirements

Breadth First Search

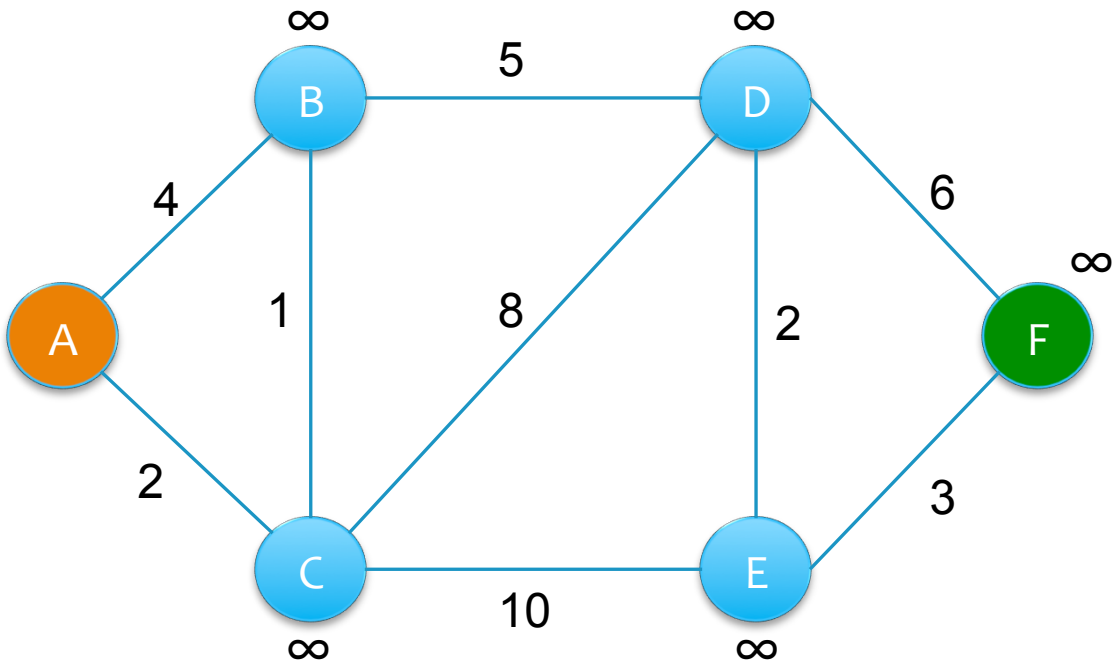


Breadth-first search

Uniform-cost search

* **Uniform-cost search** (**UCS**) is a tree search algorithm used for traversing or searching a weighted tree, tree structure, or graph. The search begins at the root node.
* The search continues by visiting the next node which has the **least total cost** from the root. Nodes are visited in this manner until a goal state is reached.
* The Dijkstra Algorithm is the most famous algorithm based on this approach

Uniform-cost search



Solve using Dijkstra

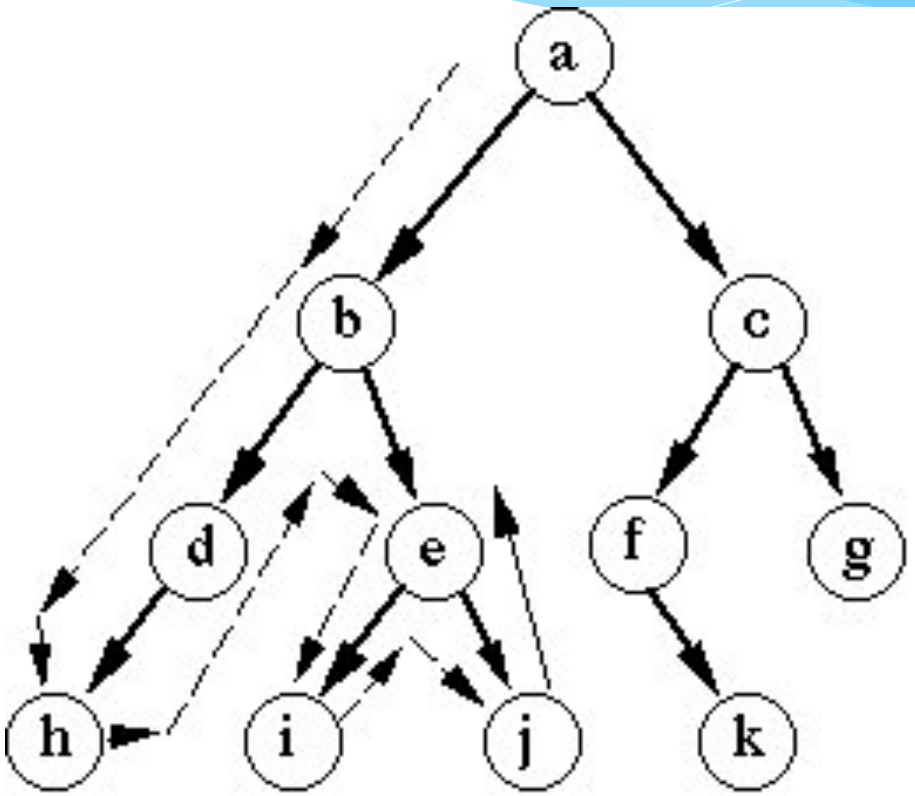
Uniform-cost search

* Solution to be inserted by **you** from class interaction

Depth-First Search

* **Depth-first search** (**DFS**) is an algorithm for traversing or searching a tree, tree structure, or graph. One starts at the root (selecting some node as the root in the graph case) and explores as far as possible along each branch before backtracking.

Depth-First Search



Depth-first search

Depth-limited Search

* **Depth-limited Search** is the same as Depth First Search with the difference of a limit.
* By imposing a limit in tree depth the unbounded tree problem is solved (a.k.a. infinite path problem).
  * i.e. depth could be a long way down or there could be an infinite loop
  * But what limit do you impose?

Iterative deepening depth-first search

* **Iterative deepening depth-first search** (IDDFS) is a strategy in which a depth-limited search is run repeatedly, increasing the depth limit with each iteration until it reaches *d*, the depth of the shallowest goal state.
* On each iteration, IDDFS visits the nodes in the search tree in the same order as depth-first search
* Wasteful?
  * Less memory needed than BFS, therefore can potentially be faster (no need to use slower memory)
* In general, iterative deepening is the preferred uninformed search method when the search space is large and the depth of the solution is not known.

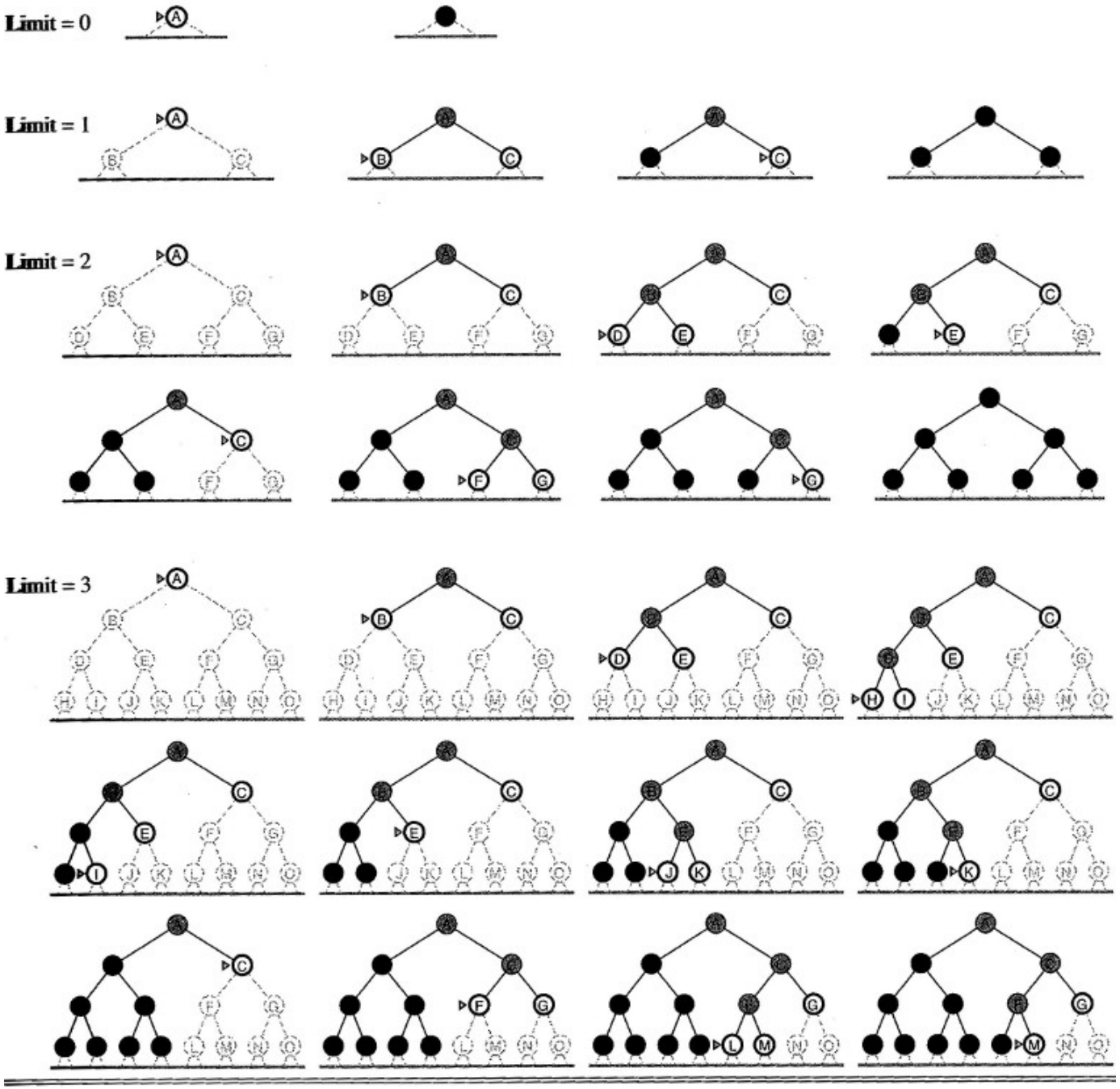
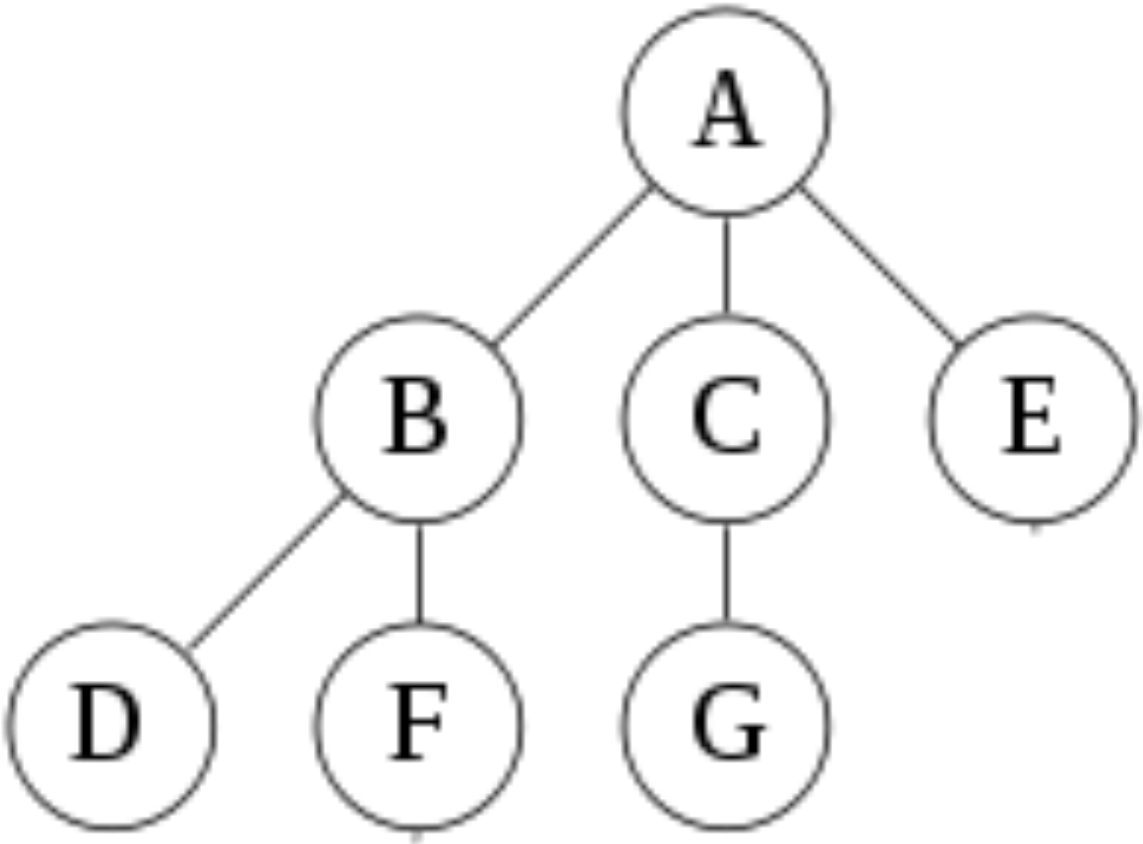


Figure 3.19 Four iterations of iterative deepening search on a binary tree.

Exercises

* Breadth-First Search and Depth-First Search



Exercises

* Answers:
  * BFS: A, B, C, E, D, F, G
  * DFS: A, B, D, F, C, G, E

Next week

* Informed (heuristically) search strategies