COMP210: Artificial Intelligence

Lecture 4. Search problems

Trevor Bench-Capon
Room 215, Ashton Building

http://www.csc.liv.ac.uk/~tbc/COMP210

Overview

- Last Time
  – Introduction to Prolog
- Today:
  – introduce *problem solving;*
  – introduce *goal formulation;*
  – show how problems can be stated as *state space*
  – *search;*
  – show the importance and role of *abstraction;*
  – define main performance measures for search

Problem Solving

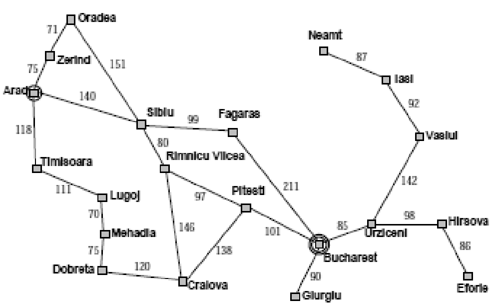
- What is a problem?
- A *goal* and a *means for achieving the goal.*
- The goal specifies the state of affairs we want to bring about.
- The means specifies the operations we can perform in an attempt to bring about the goal.
- The difficulty is deciding what *order* to carry out the operations.
- Solution will be a sequence of operations leading from initial state to goal state (plan)

Problem Formulation

- Once the goal is determined, formulate the problem to be solved.
- First determine set of possible states S of the problem.
- Then the problem has a:
  – set of states S;
  – initial state — the starting point, s0 — a member of S;
  – set of operations — the actions that can be performed, {o1, . . . , on}.
  – goal — what you are aiming at — subset of S.
- Operations cause changes in state. Performing an operation in a given state reaches some specified next state. Operations are functions s1-> s2
- A solution is a sequence of actions such that when applied to initial state s0, we reach the goal state.

Example: Romania

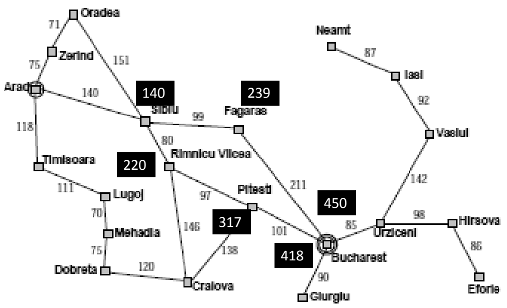
**On holiday in Romania; currently in Arad.
Flight leaves tomorrow from Bucharest**



Romania: Problem Formulation

- goal:
  – be in Bucharest
- states:
  – various cities
- operations:
  – drive between cities
    - In Arad: drive to Sidhu, drive to Zerind, drive to Timisoara, etc
- Solution:
  – A path: sequence of cities, e.g., Arad, Sibiu, Fagaras, Bucharest
    - A shortest path
    - A fastest path

Example: Romania

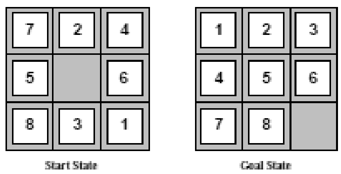


More Examples of Real World Problems

- game playing;
- route finding - routing in computer networks, rail travel, air travel;
- touring and travelling - find a route between Aberdeen and Glasgow; travelling salesperson problem
- assembly sequencing;
- VLSI layout
- robot navigation;
- . . . .

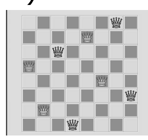
Toy Problems: The 8 Puzzle

States: 3 × 3 grid filled with numbers 1–8 and a blank.
Initial state: as shown on the left. Goal state: as shown on the right.
Operations:
o1: move any tile to left of empty square to right;
. . .



Toy Problems: The n Queens Problem

- This is a problem from chess.
- Place 8 queens on chess board so that no queen can be taken by another.
- (A queen attacks any piece in the same row, column or diagonal.)

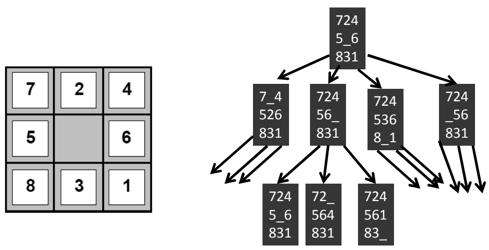


n ***Queens as a Search Problem***

- Initial state: empty chess board.
- Goal state: n queens on chess board so that none can take any other.
- Operations: place queen in empty square.
  - Place queens anywhere
  - For the 8-queens problem 64^8 possibilities.
- Place queens only where they are not attacked already.
  - Around 2,000 possible sequences to check

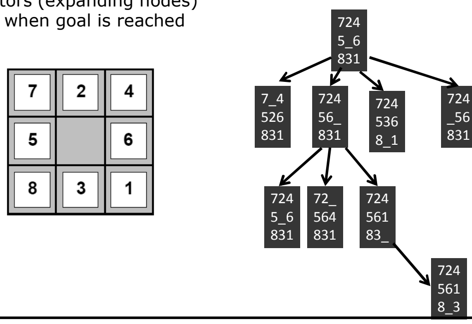
Search space

An imaginary tree showing
All possible states reachable from the initial state



Search tree

A tree showing nodes explored by a search algorithm
Root is the initial state: successor nodes found by applying operators (expanding nodes)
Stops when goal is reached



Search Strategies

- Completeness—does it always find a solution if one exists?
- Time complexity—number of nodes generated/expanded
- Space complexity—maximum number of nodes in memory
- Optimality—does it always find a least-cost solution?
- Time and space complexity are measured in terms of
  - b—maximum branching factor of the search tree
  - d—depth of the least-cost solution
  - m—maximum depth of the state space (may be infinite)

Abstraction

- Real world is absurdly complex
  - therefore state space must be *abstracted* for problem solving
- (Abstract) state = set of real states
- (Abstract) action = complex combination of real actions
  - "Arad to Zerind" represents a complex set of possible routes, detours, rest stops, etc.
- For guaranteed realisability, any real state "in Arad" must get to some real state "in Zerind"
- (Abstract) solution
  - a set of real paths that are solutions in the real world
- Abstraction should be "easier" than the original problem

Right Level of Abstraction

- Example: driving from city A to city B.
  - Some possible actions. . .
    - depress clutch;
    - turn steering wheel right 10 degrees;
  - inappropriate level of abstraction. Too much irrelevant detail.
- Better level of abstraction:
  - follow A143 to Colchester for 4 miles;
  - turn right onto M12;
  - . . . and so on.
- Getting abstraction level right lets you focus on the specifics of the problem and combats the combinatorial explosion

Solution Cost

- For most problems, some solutions are better than others:
  - 8 puzzle: number of moves to get to solution;
  - chess: number of moves to checkmate;
  - route planning: length of distance (or time) to travel.
- Mechanism for determining cost of solution is the path cost function.
- This is the length (cost) of the path through the state-space from the initial state to the goal state.

- Today
  - Some search problems
  - Representing a Search problem
  - Search Trees
- Next Time
  - Search strategies
  - Blind search (no additional information)
    - Breadth first, depth first
  - Heuristic Search (evaluates nodes)
    - Greedy search, A*