Applying Search Techniques to Difficult Puzzles

AI2 Project set by Mike Uschold

Summary of Project: The basic idea is to choose a difficult puzzle (either from those listed below, or one of your favorites) and write (or modify) a computer program to solve it. The point of the project is to explore representation and search issues, and how they interrelate. You will experiment with different representations and search strategies and compare their performance.

The representation difficulties vary depending on which puzzle you choose. If there are few difficulties, then you are expected to do a more thorough analysis of the search techniques. Puzzles to choose from:

- Instant Insanity
- Vowels puzzle (see Mike Uschold)
- Peg jumping puzzle (Charniak and McDermott p297-300)
- Rubick's Cube (*for certified workaholics only*)
- Your own favorite (must be ok'd by project supervisor)

Project Supervisor: Mike Uschold

Program:

aiva:/user3/peter/prolog/teach/search.pl (*code*)
aiva:/user3/peter/prolog/teach/search.doc (*documentation*)

Program Description: This is a general search program which can be made to do breadth first, depth first, and best first search by suitable tailoring of the basic predicates.

References:

- Uschold, Planning and Search lecture notes
- Nilsson, Principles of AI, chapters 1-2
- Rich, Artificial Intelligence, chapters 2-3
- Charniak and McDermott, Intro to AI, chapter 5

Specific questions to be addressed:

1. Formulate the problem as state space search in at least two different ways. One will be the final result of careful thought and experimentation. The other may be a naive first crack, or perhaps another viable alternative which was experimented with along the way. You are to compare and contrast the two approaches.
2. Write your own program or modify the existing program to solve the problem. You are to experiment with the following search strategies:
   - Depth first search
   - Breadth first search
   - Best first search

   Note that the uninformed strategies will probably not work (*i.e.* it may take many hours or days to find the solution). It is best to start with a simpler version of the puzzle. For the full blown puzzle, it is important to think about how to reduce search. This may be done both by clever representations, and by use of good heuristics. See the discussion in the lecture notes on the 8-Queens problem. It describes the sort of process that you are expected to go through.
3. Do an analysis of the performance of the program for different search strategies and heuristics. Keep track of such things as total number of nodes expanded, length of solution path, total computation time, etc. Note tradeoffs with expensive heuristic evaluation functions which reduce the number of nodes expanded, but may take considerably longer to expand each node. For the exhaustive strategies, if the program does not finish in reasonable time, note how far it got in the time you let it run. Do some rough calculations of the total number of nodes in the search space, and try to estimate how long it would take if the computer was let run indefinitely. (The answer need not be less than 10000 years!). See end of chapter 2, Nilsson for a good discussion of these issues.

Proposals for extending the project:

- Experiment with one or more of the alternate search strategies listed below. For each discuss how it may be implemented for this problem. If it is fairly straightforward, then implement it. If it requires a major overhaul of the program, then provide a detailed discussion of what changes would be necessary, and illustrate the approach by hand using examples from the chosen puzzle. In either case, compare with the strategies above. Say whether the new approach is (or is likely to be) better than the others.
  - Hill Climbing
  - A
  - Backward search
  - Bi-directional search
  - Means-Ends Analysis
- Discuss how the puzzle you chose might be solved using problem reduction. Say whether problem reduction is a good approach or not and explain in as much detail as possible why. If possible implement the puzzle, or a simpler version of it.
- Do the same exercise on another puzzle. Analyse and compare whether different search strategies were more or less appropriate for the different problems.
- Follow up in more detail any ideas you may have had along the way that you find interesting.