COMP3002 Artificial Intelligence
Semester 1, 2003

Assignment 1. Uninformed and Informed Search Strategies

Due in week 6 (at the beginning of your tutorial class); can be done in pairs.

The 3 digit-puzzle problem

We are given two 3 digits numbers called S and G, and also a set of numbers called forbidden. To solve the puzzle, we want to get from S to G in the fewest possible moves. A move consists of transforming a number into another by adding or subtracting 1 to one of its digits; for instance, a move can take you from 123 to 124, or 953 to 853. Moves are subjects to the following constraints:

- You cannot add to the digit 9 or subtract from the digit 0;
- You cannot make a move that transforms the current number into one of the forbidden numbers;
- You cannot change the same digit twice in two successive moves.

Since the numbers have 3 digits, there are at most 6 possible moves from S at the start. Then, the branching factor is at most 4 due to the constraints above.

Questions

1. Find a simple admissible heuristic for this problem (one that does not involve extensive mathematical calculation, but that is still powerful). Justify why your heuristic is admissible.

2. Write a program (in a language of your choice) performing BFS, DFS, IDS, Greedy and A* to find the solution to the puzzle.
   - Use the input file called puzzle.txt, provided with this assignment. The first line specifies the start number, the second – the goal number, the third - the forbidden numbers separated by commas.
   - Perform the search according to the search strategy defined by the user (B for BFS, D for DFS, I for IDS, G for Greedy and A for A*).
     - Assume that the order of children for a node is: 1) 1 is subtracted to the first digit, 2) 1 is added to the first digit, 3) 1 is subtracted to the second digit, 4) 1 is added to the second digit, 5) 1 is subtracted to the third digit, 6) 1 is added to the third digit. Example: the order of children for node 678 coming from parent 668 is (578,778,677,679).
     - For the heuristic search strategies assume also that amongst the nodes with the same h-value, the last added is used first.
     - Do not let your program run indefinitely if it cannot find a solution. Set a limit of 1000 nodes maximum. When it cannot find a solution, your program should state it explicitly.
   - Return the *solution* found, the *memory costs* in number of nodes (maximum number of nodes stored at any given time), the total *number of nodes expanded* and the *time* to perform the search.
   - If the user requests so, your program should also trace *all the nodes expanded* with their respective f, g, and h values.

3. Present the results for each search strategy in a table and discuss them (what was found and if it is consistent with the theory; if there are any discrepancies, how they can be explained.)

Strategy	Complete	Optimal	Memory cost (in number of nodes)	Number of expanded odes	Search time
BFS					
DFS					
IDS					
Greedy					
A*					

Submission

1) hard copy (report+code) in a folder to your tutor at the beginning of your tutorial class and 2) electronically (only the code) by e-mail or netfile to your tutor.

Marking scheme (guidelines)

1. 1 mark
   - 0.5 mark for giving a suitable heuristic
   - 0.5 mark for justifying why it is admissible
2. 8 marks
   - 2 marks for the code: well-written, well-commented
   - 5 marks for the correct implementation of the 5 search strategies
   - 1 mark - marker's discretion.

   If the code does not compile or does not run you - 0 marks for this part.
3. 3 marks
   - 1 mark for a table showing reasonable results
   - 2 mark for the discussion