

Homework 1

Due: Feb 13, 2003

Notes on electronic submission: You may, if you wish, submit these assignments electronically (although you are not required to—paper submission is acceptable). For the written parts (Questions 1 and 2), please submit either PDF, PostScript, or HTML (with gif or jpeg images). I do not parse MS Word or other proprietary formats. For the code and output (Q 3–5), please hand in the complete code that you wrote. If you used any special packages (other than the standard class libraries, builtin LISP functions, etc.), please document what you used, what part they play in your code, and where they came from. *DO NOT* actually hand in said packages. Also, please do not hand in compiled code, executables, or compiled classes. For the output, you may either hand in plain text or some reasonably formatted version. Please group together all files into a directory named “yourLastName.yourFirstInitial.hw1”, tar and gzip the entire directory, and email it to me as a MIME attachment. E.g., if Frodo Baggins is submitting this assignment, he would put all of his documents in `baggins.f.hw1`.¹ Electronic submissions are still due at the beginning of class—I will check the timestamp on the delivery into my mailbox.

1. Ch. 4, Probs. 6, 7.
2. Prepsi Cola Inc. produces two different beverages: their flagship product, Prepsi Cola, and their high-caffeine swill for hacker types and extreme athlete wanna-be's, Mountain Drool. They have two production plants, one in Kentucky (where labor is cheap, but shipping to large markets is expensive) and one in Massachusetts (where labor is expensive but shipping is cheap). Let the cost of producing a unit of Prepsi Cola in Kentucky be $C_{p,k}$, the cost of producing Mountain Drool in Massachusetts be $C_{m,m}$, etc. Because of differential staffing, it takes different amounts of work to make the product at each plant, $w_{m,k}$ to make a unit of Mountain Drool in Kentucky and so on. Each factory has a maximum number of work units it can put in: m_k and m_m . To fill the insatiable maws of the public with delightful carbonated beverages, PrepsiCo must produce a total of g_p units of Prepsi Cola and g_m units of Mountain Drool. They would like to produce all the necessary product at the minimum total cost.

Formulate this problem as a linear programming instance. You do not have to *solve* the linear program (which would be difficult, as I haven't given you any actual numbers – PrepsiCo is pretty tightfisted with their trade secrets), but you do have to write the optimization criteria and boundary conditions. What is the dimension of the optimization (i.e., search) space for this problem? How many hyper-planes are expressed by the constraints?

3. Write a single function (or class, if you're using an OO language) that can implement DFS, BFS, best-first, and A* graph searches on arbitrary (discrete) graphs. You should provide mechanisms for customizing the next-state generation function, the goal-state recognizer, the heuristic function, the order of node expansion (i.e., the order in which children are inserted into the OPEN list), and a bound on the total number of nodes generated (i.e., a way to specify that the search should terminate and fail if no solution is found within x nodes). You should also provide a mechanism to report the path to the solution, the total length of the solution, the total number of nodes created during the search, and the maximum length of the OPEN queue at any point in the search.

It is easy to implement a generic function of this type in LISP/Scheme or Java or possibly C++. It's considerably harder in C. You may employ classes for “standard” data structures (i.e., linked lists, trees, queues, stacks, etc.), but you may *not* employ any code you can find that actually implements a

¹I'm sure that Frodo wished he had had it so easy. Please do not send me the One Ring of Sauron in my email.

search routine. If you have questions on whether some function or library is legal to use, please ask me.

Deliverables: You should turn in the entire code for your function/class/program. This may be a print-out or submitted electronically (see notes on electronic submission, above).

4. Use your program from problem 3 to solve the 8-puzzle between the following start and goal states, using these A* heuristic functions:

- Number of tiles out of place (heuristic A).
- Manhattan distance between the tiles and their destinations (heuristic B).

Deliverables: For each pair, you must turn in a printout (or file) that describes the results of running the search with each search mechanism (DFS, BFS, and best-first/A* with each of the above heuristics). I do *not* want a complete dump of the run. What I *do* want is (i): a list of the moves on the optimal path to the solution (e.g., (U, L, L, D, R, D, R, U)), (ii) the length of the optimal path, (iii) the total number of nodes examined, (iv) the largest number of nodes in the OPEN queue at any point, and (v) the number of times that the search removed something from CLOSED and added it to OPEN. You may set a termination threshold of 2^{19} nodes examined – if the search does not terminate in that time, you may report simply that the search failed and items (iv) and (v).

	START		GOAL
(a)	2	8	3
	1	6	4
	7		5
	1	2	3
	8		4
	7	6	5

	START		GOAL
(b)	2	1	6
	4		8
	7	5	3
	1	2	3
	8		4
	7	6	5

5. **The 8 Queens Problem** The problem is to place 8 queens on an otherwise empty chess board such that no queen is directly threatening any other queen. (Recall that a queen threatens every piece in the same row and column that she occupies as well as both diagonals that she occupies.)
- Devise and describe a state space representation for this problem and two search heuristics. Are your heuristics admissible?
 - Employ your two heuristics with the best-first/A* search program from question 3 to solve this problem. For this question, turn in a diagram of your solution (assuming you find one), as well as items (iii)-(iv) from question 4.