CMPT 310 (2001-1) Assignment 1 Solution

1. [20%] We want to reformulate a given state-space search problem into an equivalent problem so that a state in the new formulation corresponds to a path in the original problem:

   - *State Representation (8%):* A state is a sequence $\langle s_0 s_1 s_2 \ldots s_n \rangle$, where each s_i is a state in the original problem.
   - *Initial State (3%):* The sequence $\langle s_0 \rangle$, where s_0 is the initial state of the original problem.
   - *Operators (3%):* For each operator o in the original problem, formulate an operator o' for the new problem, so that $s_n \stackrel{o}{\longrightarrow} s_{n+1}$ implies $\langle s_0 s_1 s_2 \ldots s_n \rangle \stackrel{o'}{\longrightarrow} \langle s_0 s_1 s_2 \ldots s_n s_{n+1} \rangle$.
   - *Goal Test (3%):* To test if a sequence $\langle s_0 s_1 s_2 \ldots s_n \rangle$ is a goal state, apply the goal test of the original problem to s_n.
   - *Path Cost Function (3%):* A path in the new problem formulation is a sequence of state sequences, e.g. "$\langle s_0 \rangle \cdot \langle s_0 s_1 \rangle \cdot \langle s_0 s_1 s_2 \rangle \cdots \langle s_0 s_1 s_2 \ldots s_n \rangle$". To evaluate the cost of a path like the above, apply the path cost function of the original problem to the last sequence.

   When the pathless search procedure returns the state $\langle s_0 s_1 s_2 \ldots s_n \rangle$, we know the solution path for the original problem.

2. [20%] *State space construction (10%):* We define a class of state spaces, parameterized by the solution depth d. An instance of the class is a state space graph composing of two parts (see figure 1): (1) a complete binary tree of height $d-1$, with its root being the initial state and the children of a tree node being the successor states of the parent, and (2) a goal state, which is the successor of the leaves in the binary tree. For this search problem, the branching factor is 2 and the solution depth is d.

   Analysis of DFS (5%): DFS finds the goal state without backtracking, and thus expands only $\Theta(d)$ nodes.

   Analysis of IDS (5%): IDS finds the goal by applying $d+1$ depth-limited searches with limits 0, 1, ... , d. The first d depth-limited searches expand nodes strictly in the complete binary tree, and thus perform exponentially many node expansions. The last depth-limited search finds the goal with no backtracking, and thus expands only d nodes. The precise number of node expansions performed by each depth-limited search is summarized as follows:

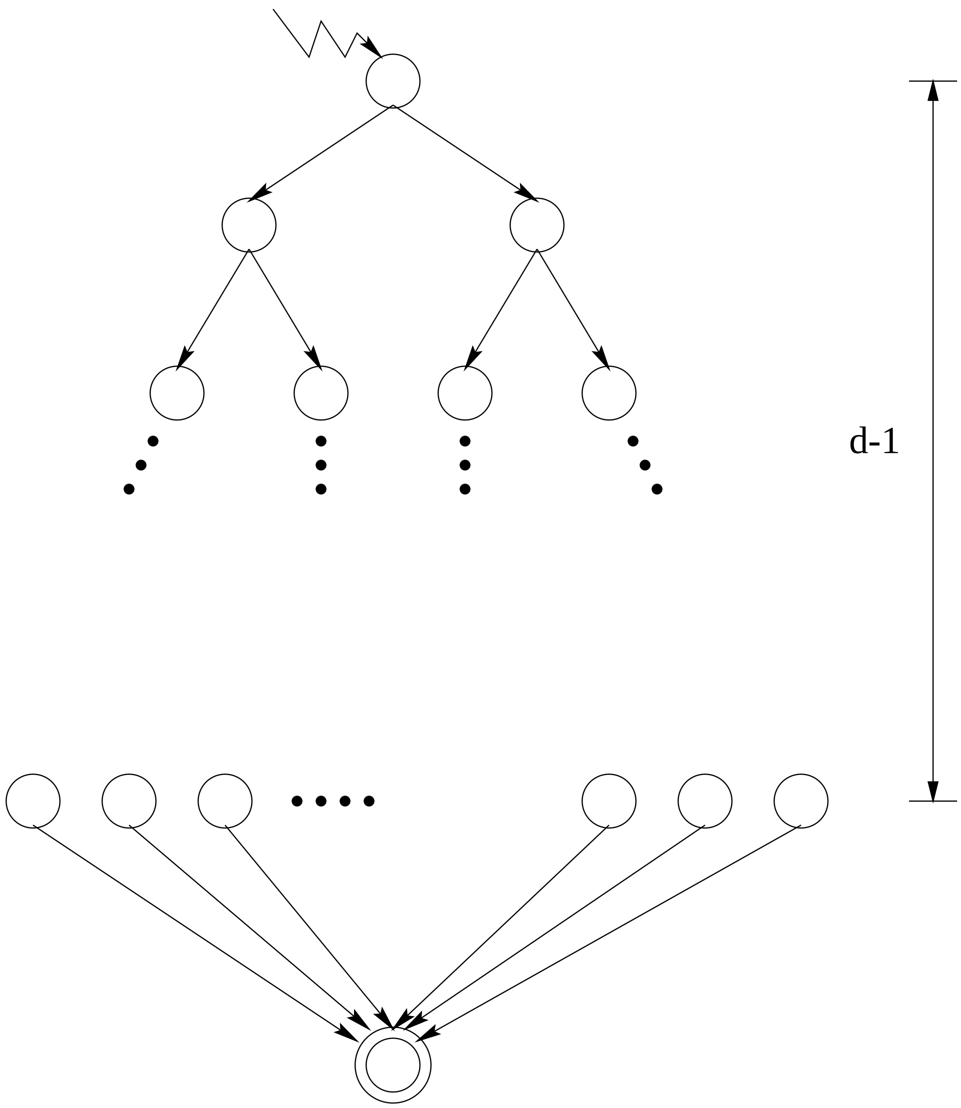

Figure 1: The state space graph for the contructed class of search problems.

Depth limit	# of nodes expanded	
0	1	$= 2^1 - 1$
1	$1 + 2$	$= 2^2 - 1$
2	$1 + 2 + 2^2$	$= 2^3 - 1$
3	$1 + 2 + 2^2 + 2^3$	$= 2^4 - 1$
$\vdots$	$\vdots$	$\vdots$
$d - 1$	$1 + 2 + 2^2 + 2^3 \ldots 2^{d-1}$	$= 2^d - 1$
d	d	$= d$
Total		$= 2^1 + 2^2 + 2^3 + 2^4 + \ldots + 2^d$ $= 2(2^d - 1) = \Theta(2^d)$

3. (a) i. [5%] h_3 *is admissible:* Let $h^*(s)$ be the actual cost of the cheapest path from state s to a goal state. Both $h_1(s)$ and $h_2(s)$ are admissible, so they never overestimate $h(s)$:

$$\begin{aligned} & h_1(s) \leq h^*(s) \\ & h_2(s) \leq h^*(s) \\ \Rightarrow\ & h_3(s) = \max\{h_1(s), h_2(s)\} \leq h^*(s) \end{aligned}$$

ii. [5%] h_3 *is at least as informed as* $h_1(s)$ *and* $h_2(s)$*:*

$$h_1(s), h_2(s) \leq \max\{h_1(s), h_2(s)\} = h_3(s)$$

(b) i. [5%] Let $h_F(s)$ and $h_M(s)$ be the heuristic functions developed by Fee and Moo respectively. We can combine the two heuristics by defining a new heuristic function $h(s) = \max\{h_F(s), h_M(s)\}$, which is an admissible heuristics at least as informed as h_F and h_M.

ii. [5%] Fee must have demonstrated that Moo's heuristics is no more informed than her own, that is,

$$h_M(s) \leq h_F(s) \quad \text{for all state } s$$

Therefore, we have

$$h_F(s) = \max\{h_F(s), h_M(s)\} = h(s)$$

Fee gains nothing by incorporating Moo's heuristics into her own.

4. [0%] Consult `fast-intersection.lisp`.

5. [40%] Consult `ADT.lisp`. *(10% for comments and style; 10% for test cases; 20% for code.)*