

## Solution for CS 540, HW 2, Spring 2008

### Problem 1

#### a. Breadth First

Open List	Closed List	Popped State	Children	New Children
{S}	{}	S	{A,B}	{A,B}
{A,B}	{S}	A	{B,C,D}	{C,D}
{B,C,D}	{S,A}	B	{D,E,G3}	{E,G3}
{C,D,E,G3}	{S,A,B}	C	{S}	{}
{D,E,G3}	{S,A,B,C}	D	{G1,G2}	{G1,G2}
{E,G3,G1,G2}	{S,A,B,C,D}	E	{D}	{}
{G3,G1,G2}	{S,A,B,C,D,E}	G3	{}	{}

Popped List: S, A, B, C, D, E, G3

Goal reached: G3

#### b. Depth First (using a closed list)

Open List	Closed List	Popped State	Children	New Children
{S}	{}	S	{A,B}	{A,B}
{A,B}	{S}	A	{B,C,D}	{C,D}
{C,D,B}	{S,A}	C	{S}	{}
{D,B}	{S,A,C}	D	{G1,G2}	{G1,G2}
{G1,G2,B}	{S,A,C,D}	G1	{}	{}

Popped List: S, A, C, D, G1

Goal reached: G1

#### c. Iterative Deepening

Open List	Closed List	Popped State	Children	New Children
{S}	{}	S	{}	{}
{}	{S}	{}	{}	{}
{S}	{}	S	{A,B}	{A,B}
{A,B}	{S}	A	{}	{}
{B}	{S,A}	B	{}	{}
{}	{S,A,B}	{}	{}	{}
{S}	{S,A,B}	S	{A,B}	{A,B}
{A,B}	{S}	A	{B,C,D}	{C,D}
{C,D,B}	{S,A}	C	{}	{}
{D,B}	{S,A,C}	D	{}	{}
{B}	{S,A,C,D}	B	{D,E,G3}	{E,G3}
{E,G3}	{S,A,C,D,B}	E	{}	{}
{G3}	{S,A,C,D,B,E}	G3	{}	{}

Popped List: S, S, A, B, S, A, C, D, B, E, G3

Goal reached: G3

**d. Uniform Cost** (will get optimal – ie, shortest path – solution)

Open List	Closed List	Popped State	Children	New Children
{S <sub>0</sub> }	{}	S	{A,B}	{A,B}
{A <sub>1</sub> <sup>S</sup> , B <sub>7</sub> <sup>S</sup> }	{S <sub>0</sub> }	A	{B,C,D}	{C,D}
{C <sub>2</sub> <sup>A</sup> , B <sub>3</sub> <sup>A</sup> , D <sub>16</sub> <sup>A</sup> }	{S <sub>0</sub> , A <sub>1</sub> <sup>S</sup> }	C	{S}	{}
{B <sub>3</sub> <sup>A</sup> , D <sub>16</sub> <sup>A</sup> }	{S <sub>0</sub> , A <sub>1</sub> <sup>S</sup> , C <sub>2</sub> <sup>A</sup> }	B	{D,E,G3}	{E,G3}
{E <sub>7</sub> <sup>B</sup> , D <sub>13</sub> <sup>B</sup> , G <sub>3</sub> <sub>14</sub> <sup>B</sup> }	{S <sub>0</sub> , A <sub>1</sub> <sup>S</sup> , C <sub>2</sub> <sup>A</sup> , B <sub>3</sub> <sup>A</sup> }	E	{D}	{}
{D <sub>10</sub> <sup>E</sup> , G <sub>3</sub> <sub>14</sub> <sup>B</sup> }	{S <sub>0</sub> , A <sub>1</sub> <sup>S</sup> , C <sub>2</sub> <sup>A</sup> , B <sub>3</sub> <sup>A</sup> , E <sub>7</sub> <sup>B</sup> }	D	{G1,G2}	{G1,G2}
{G <sub>2</sub> <sub>13</sub> <sup>D</sup> , G <sub>3</sub> <sub>14</sub> <sup>B</sup> , G <sub>1</sub> <sub>15</sub> <sup>D</sup> }	{S <sub>0</sub> , A <sub>1</sub> <sup>S</sup> , C <sub>2</sub> <sup>A</sup> , B <sub>3</sub> <sup>A</sup> , E <sub>7</sub> <sup>B</sup> , D <sub>10</sub> <sup>E</sup> }	G2	{}	{}

Popped List: S, A, C, B, E, D, G2

Goal reached: G2 (via path S-A-B-E-D-G2)

**e. Best-first (using f = h)**

Open List	Closed List	Popped State	Children	New Children
{S <sub>100</sub> }	{}	S	{A,B}	{A,B}
{A <sub>10</sub> , B <sub>25</sub> }	{S}	A	{B,C,D}	{C,D}
{C <sub>1</sub> , D <sub>3</sub> , B <sub>25</sub> }	{S,A}	C	{S}	{}
{D <sub>3</sub> , B <sub>25</sub> }	{S,A,C}	D	{G1,G2}	{G1,G2}
{G <sub>1</sub> <sub>0</sub> , G <sub>2</sub> <sub>0</sub> , B <sub>25</sub> }	{S,A,C,D}	G1	{}	{}

Popped List: S, A, C, D, G1

Goal: G1

**f. Best-first (using f = g + h)** Note: h is not admissible, so we might not get the optimal soln.

Open_List	Closed_List	Popped State	Children	New Children
{S <sub>0+100</sub> }	{}	S	{A,B}	{A,B}
{A <sub>1+10</sub> <sup>S</sup> , B <sub>7+25</sub> <sup>S</sup> }	{S <sub>0+100</sub> }	A	{B,C,D}	{C,D}
{C <sub>2+1</sub> <sup>A</sup> , D <sub>16+3</sub> <sup>A</sup> , B <sub>3+25</sub> <sup>A</sup> }	{S <sub>0+100</sub> , A <sub>1+10</sub> <sup>S</sup> }	C	{S}	{}
{D <sub>16+3</sub> <sup>A</sup> , B <sub>3+25</sub> <sup>A</sup> }	{S <sub>0+100</sub> , A <sub>1+10</sub> <sup>S</sup> , C <sub>2+1</sub> <sup>A</sup> }	D	{G1,G2}	{G1,G2}
{G <sub>2</sub> <sub>19+0</sub> <sup>D</sup> , G <sub>1</sub> <sub>21+0</sub> <sup>D</sup> , B <sub>3+25</sub> <sup>A</sup> }	{S <sub>0+100</sub> , A <sub>1+10</sub> <sup>S</sup> , C <sub>2+1</sub> <sup>A</sup> , D <sub>16+3</sub> <sup>A</sup> }	G2	{}	{}

Popped List: S, A, C, D, G2

Goal reached: G2 (closest goal, but incorrect path, S-A-D-G2, because B's h value is too high)

**g. Beam Search (with beam width = 2 and f = h)**

Open List	Closed List	Popped State	Children	New Children
{S <sub>100</sub> }	{}	S	{A,B}	{A,B}
{A <sub>10</sub> , B <sub>25</sub> }	{S}	A	{B,C,D}	{B,C,D}
{C <sub>1</sub> , D <sub>3</sub> }	{S,A}	C	{S}	{}
{D <sub>3</sub> }	{S,A,C}	D	{G1,G2}	{G1,G2}
{G <sub>1</sub> <sub>0</sub> , G <sub>2</sub> <sub>0</sub> }	{S,A,C,D}	G1		

Popped List: S, A, C, D, G1

Goal reached: G1

**h. Hill Climbing (using the  $h$  function only)**

Open List	Closed List	Popped State	Children	New Children
{ $S_{100}$ }	{}	S	{A,B}	{A,B}
{ $A_{10}$ }	{S}	A	{B,C,D}	{B,C,D}
{ $C_1$ }	{S,A}	C	{S}	{ $S_{100}$ }

Popped List: S, A, C

Goal reached: None!

**Problem 2**

*a. Name three (3) searches that may never find a solution, even when one exists. Briefly explain.*

**Depth-first Search** When, say, the left sub tree is of unbounded depth, but contains no solutions, depth-first search will never terminate.

**Hill Climbing** It could get stuck at local maxima.

**Beam Search** Could have dropped the path to the only solution off the end of the beam.

*b. When is breadth-first search an admissible search strategy? Briefly explain.*

When all step costs are equal, breadth-first search will be an admissible search strategy, which will find out the optimal solution, if there is one. Because it always expands the shallowest unexpanded nodes and the shallowest goal node will be the optimal solution, since it costs the least.

*c. Assume  $h_1$ ,  $h_2$ , and  $h_3$  are all admissible heuristic functions.*

*i. Is  $h_4 = h_1 + h_2 + h_3$  admissible? Explain.*

No. Although  $h_1$ ,  $h_2$  and  $h_3$  are all no bigger than the cost to the goal, their sum  $h_4$  may be bigger than the cost. E.g. true cost = 10,  $h_1 = 5$ ,  $h_2 = 6$ ,  $h_3 = 4$ , we get  $h_4 = 15 >$  true cost.

*ii. Is  $h_5 = (h_1 + h_2 + h_3) / 3$  admissible? Explain.*

Yes. Since  $h_1$ ,  $h_2$ , and  $h_3$  are all less than the true cost (and non-negative), their average must be as well since the average cannot exceed the largest of the  $h_1$ ,  $h_2$ ,  $h_3$ .

*iii. Is  $h_6 = \max(h_1, h_2, h_3)$  admissible? Explain.*

Yes. Because none of  $h_1$ ,  $h_2$ ,  $h_3$  is bigger than the true cost (ie, max is only as big as the highest of  $h_1$ ,  $h_2$ , and  $h_3$ ).

*iv. Would you prefer  $h_7 = \min(h_1, h_2, h_3)$  or  $h_6$ ? Why?*

$h_6$  is preferred. Using  $h_6$ , fewer (or the same number of) states will need to be considered, since  $h_6$  is closer to the true cost.

**Problem 3 (35 points)****a. How might you represent states for this problem?**

Suppose initially everyone is on the left side of the river, the goal is to get everyone on the right side of the river. We represent each state by three elements IJX,

I is the number of missionaries on the left side,  $0 \leq I \leq 3$

J is the number of cannibals on the left side,  $0 \leq J \leq 3$

(we don't need to represent the numbers on the right side, since they can be directly computed from the numbers on the left – note we also assume that the boat instantaneously crosses the river, and passengers embark and disembark simultaneously)

X represents where the boat is. X is either the left side (L) or the right side (R).

The constraint requires  $I=0$ ,  $I=3$ , or  $I \geq J$  and  $(3-I) \geq (3-J)$  for all legal states.

**b. What would be the initial state in your representation?**

The Initial state is 33L.

**c. What would be the goal test for this task?**

The goal state is 00R

**d. Draw the state space explored by breadth-first search. Clearly label the arcs between nodes.**

In a separate file that is linked to the class home page.

**e. Answer Part c of Exercise 3.9 in the text**

Although the state space is simple, too many transitions between states and the constraints of the problem make it hard for people to solve. Besides, there exist infinite loops, which don't contain a solution. Executing a systematic search makes the problem easily solvable.