Midterm 2 Solution

Question 1 (25 points). Knowledge base representation

1) Represent the domain as a DATALOG knowledge base

% Pat is a tiger
tiger(pat).
% Tigers are cat
cat(X) ← tiger(X).
% Tigers like to eat animals
like_to_eat(X,Y) ← tiger(X), animal(Y).
% Rat is a rabbit
rabbit(rat).
% Tigers, cats, and rabbits are all animals
animal(X) ← tiger(X).
animal(X) ← cat(X).
animal(X) ← rabbit(X).

2) How are the queries 'Does Pat like to eat Rat' and 'Does Pat like to eat Pat' expressed in your representation? What will be the answer to these queries?

- ?like_to_eat(pat,rat). The answer is yes.

  Since pat is a tiger, it likes to eat animals. In addition, rat is an animal because it is a rabbit. Therefore, pat likes to eat rat.
- ?like_to_eat(pat, pat). The answer is yes because pat is a tiger and thus, it is also an animal.

Question 2 (50 points). 8-puzzle game

a) The search graph is described as follows.

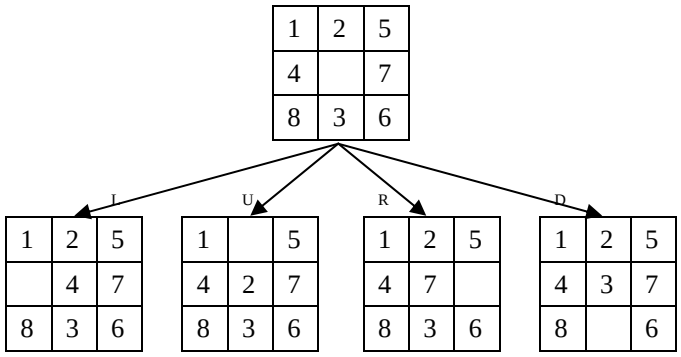
- Each *node* represents a state (or configuration) of the board, that is, the positions of numbered tiles and the position of the blank space on the board.
- A *neighbor* N of a node S is the node representing the configuration that can be obtained from S by sliding a numbered tile adjacent to the blank space into the blank space. Arcs are labeled with L, U, R or D, according to the direction of the blank space movement: Left, Up, Right, or Down.
- The start node is

1	2	5
4		7
8	3	6

and the goal node is

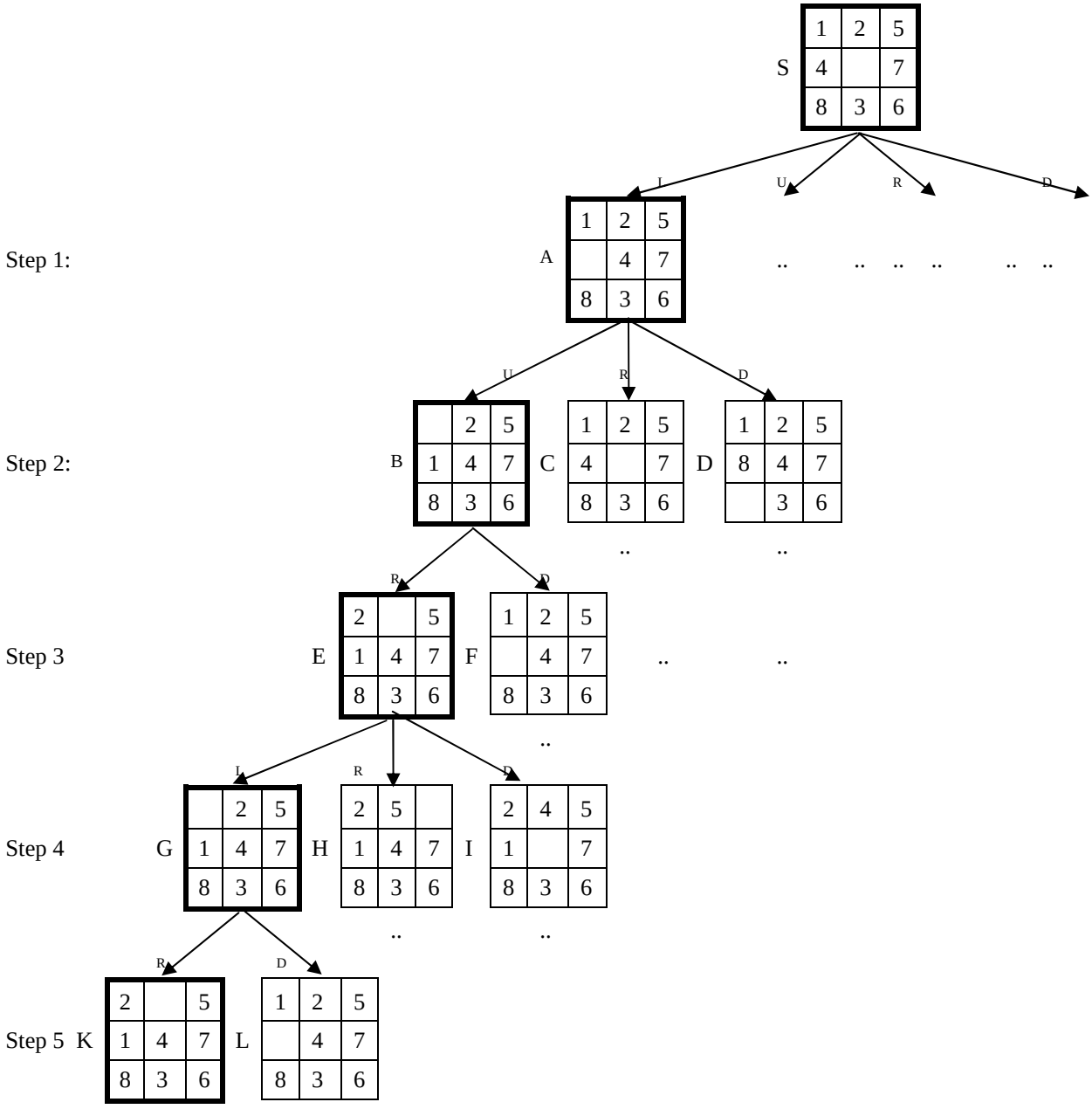
1	2	3
4	5	6
7	8	

b) The neighbors of the initial node are represented below.



c) I. Depth-first search

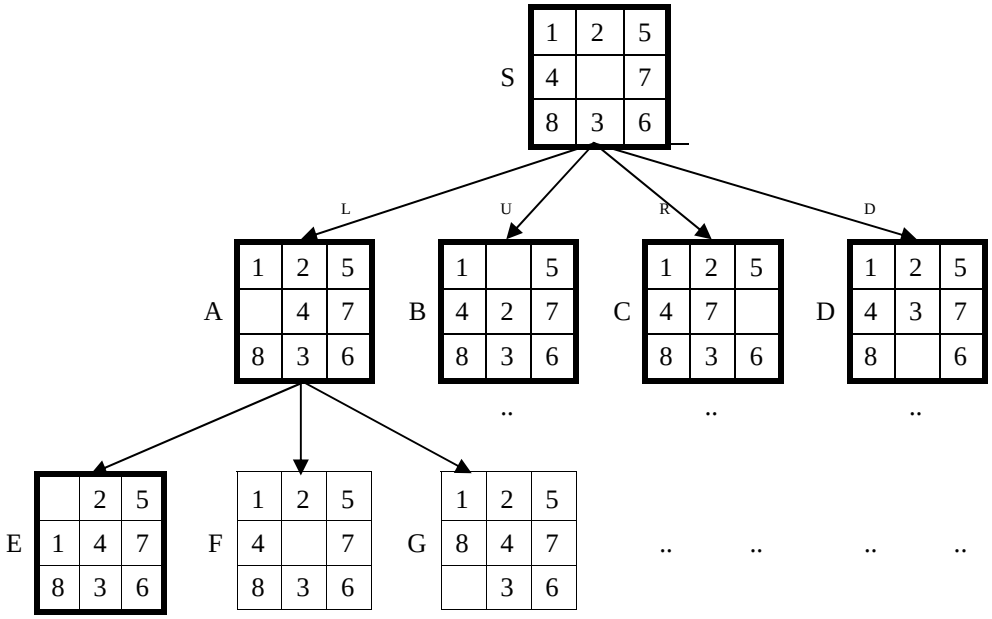
Note that the depth-first strategy selects the first node in the frontier at each choice point and adds that node's neighbors to the front of the frontier. The figure below demonstrates how the depth-first search works. Nodes selected in order are S, A, B, E, G, K.



The board after 5 moves is:

2		5
1	4	7
8	3	6

II. Breadth-first Search



The nodes selected in order after 5 moves are S, A, B, C, D and E. The board after 5 moves is:

	2	5
1	4	7
8	3	6

III. Best first search

The best-first search always chooses the node with the lowest value of the *h* function in the frontier.

1. First the start node S with the *h* value of 6 is added to the frontier: Frontier = {**S/6**}.

2. Select S and remove it from the frontier. As S is not a goal, its neighbors A, B, C and D are added to the frontier:

A=

1	2	5
	4	7
8	3	6

h = 7

B=

1		5
4	2	7
8	3	6

h = 7

C=

1	2	5
4	7	
8	3	6

h = 6

D=

1	2	5
4	3	7
8		6

h = 6

After sorted according to the *h* value, the frontier is {**C/6**, D/6, A/7, B/7}.

3. Since C has the lowest value of *h* function, select and remove it from the frontier. As it is not a goal, its neighbors E, F and G are added to the frontier:

E =

1	2	5
4		7
8	3	6

h = 6

F =

1	2	
4	7	5
8	3	6

h = 6

G =

1	2	5
4	7	6
8	3	

h = 4

Frontier = {**G/4**, D/6, E/6, F/6, A/7, B/7}

4. Select the node with the minimum *h* value G and add its neighbors to the frontier:

H =

1	2	5
4	7	6
8		3

h = 5

I =

1	2	5
4	7	
8	3	6

h = 6

Frontier = {**H/5**, D/6, E/6, F/6, I/6, A/7, B/7}

5. Select H and add its neighbors to the frontier:

J =

1	2	5
4	7	6
	8	3

h = 4

K =

1	2	5
4		6
8	7	3

h = 5

L =

1	2	5
4	7	6
8	3	

h = 4

Frontier = {**J/4**, L/4, K/5, D/6, E/6, F/6, I/6, A/7, B/7}

6. Select L and add its neighbors to the frontier:

M =

1	2	5
4	7	6
8		3

h = 5

N =

1	2	5
4	7	
8	3	6

h = 6

Frontier = {**L/4**, K/5, M/5, D/6, E/6, F/6, I/6, N/6, A/7, B/7}

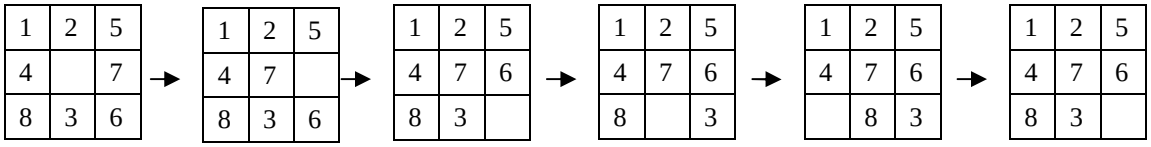
7. L is selected.

(Note that there may be a case in which one node is denoted by two different letters. As this has no effect on the final result, I used this for clarity).

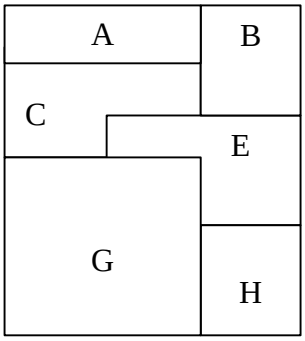
Therefore if we're using the best-first search strategy, the board after 5 moves will be:

1	2	5
4	7	6
8	3	

The nodes selected at choice points step by step after 5 moves are:



8. Question 3 (25 points). Map coloring problem



a) Let A1, A2, …, An denote *n* countries and let C1, C2, …, Cn be variables denoting the colors of these countries. The map-coloring problem can be represented in terms of a CSP problem as follows.

- Variables: C1, C2, …, Cn.
- Domains: D(Ci) = {1,2,…,k} for all i, $1 \le i \le n$.
- Constraints: $Ci \neq Cj$ for all $1 \le i \le n$, $1 \le j \le n$, $i \neq j$ such that adjacent(Ai,Aj) is true.

The given 6-country map coloring problem therefore can be represented as:

- Variables: A, B, C, E, G, H.
- Domains: D(A) = D(B) = D(C) = D(E) = D(G) = D(H) = {R(ed), G(reen), B(lue)}.
- Constraints:

(1) $A \neq B$	(3) $B \neq C$	(5) $C \neq E$	(7) $E \neq G$	(9) $G \neq H$
(2) $A \neq C$	(4) $B \neq E$	(6) $C \neq G$	(8) $E \neq H$	

b) *Solve the problem using the backtracking algorithm*

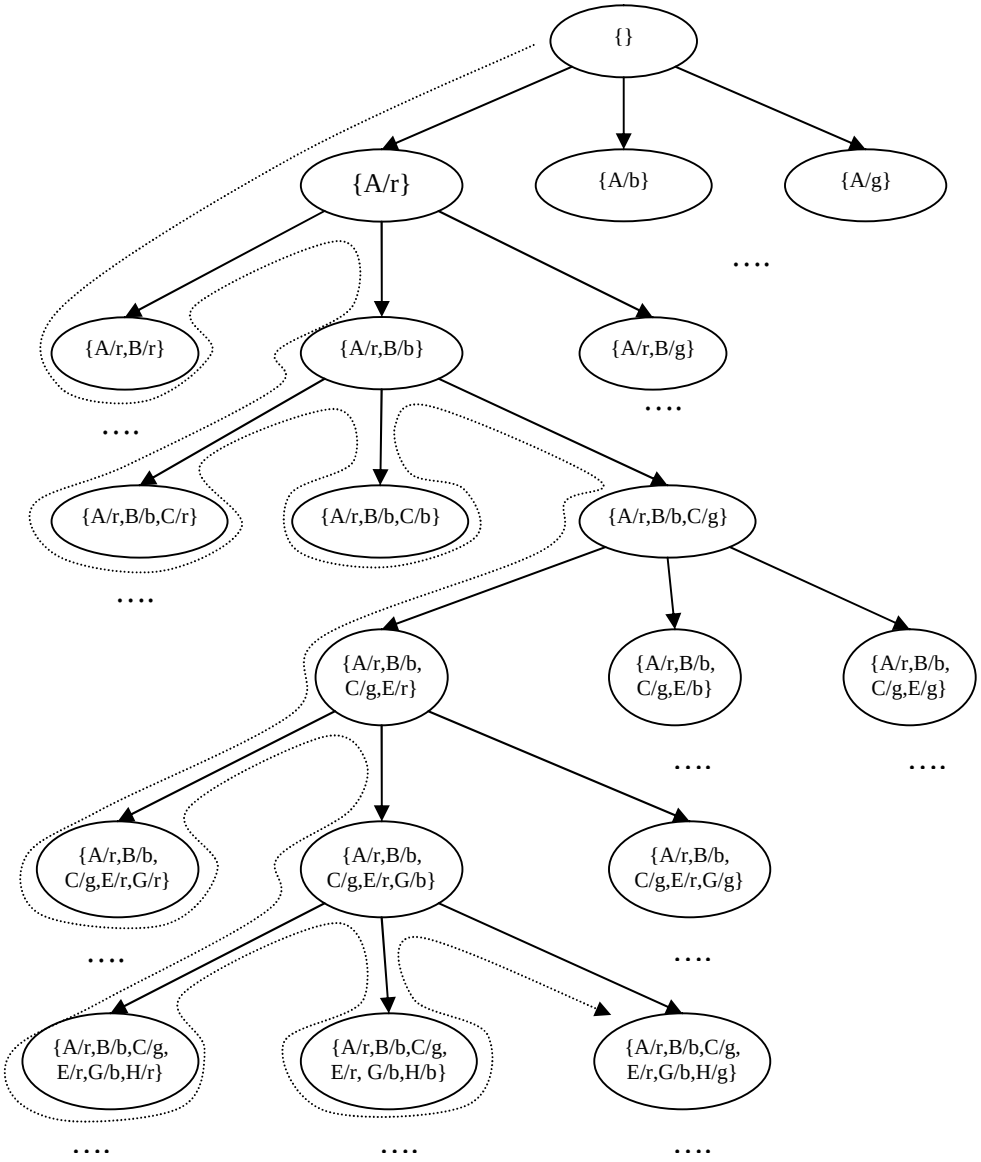
Recall that the backtracking algorithm can be viewed as a graph-searching algorithm in which each node is a substitution that assigns values to the some variables. The start node is the empty substitution and a goal node is a substitution that assigns values to all the variables and satisfies all the constraints. Let's describe how the backtracking algorithm works by looking at our 6-country map coloring problem.

First, the variables are sorted in lexical order: A, B, C, E, G, H.

1. Choose the start node, which is empty substitution {}.
2. There are 3 possibilities for coloring A, so the neighbors of the start node are {A/r}, {A/b}, and {A/g}. Select node {A/r}. No constraints are tested at this time point.
3. Select node {A/r,B/r}, which is a neighbor of {A/r}. Constraint (1) is tested and failed.
4. Select another neighbor of {A/r}, say {A/r,B/b}. Constraint (1) is satisfied.
5. Select node {A/r,B/b,C/r} from the neighbors of {A/r,B/b}. Constraint (2) is failed.
6. Select node {A/r,B/b,C/b}. Constraint (3) is failed.
7. Select node {A/r,B/b,C/g}. All constraints whose variables are grounded are satisfied and thus continue to examine the neighbors of this node.

This process continues until we reach the node where every variable is grounded and all the constraints are satisfied. The figure below illustrates how this process works step by step. It terminates when we reach the node {A/r,B/b,C/g, E/r,G/b,H/g}.

Note that only part of the whole graph is drawn in the figure.


{}
{A/r}
{A/b}
{A/g}
{A/r,B/r}
{A/r,B/b}
{A/r,B/g}
{A/r,B/b,C/r}
{A/r,B/b,C/b}
{A/r,B/b,C/g}
{A/r,B/b,
C/g,E/r}
{A/r,B/b,
C/g,E/b}
{A/r,B/b,
C/g,E/g}
{A/r,B/b,
C/g,E/r,G/r}
{A/r,B/b,
C/g,E/r,G/b}
{A/r,B/b,
C/g,E/r,G/g}
{A/r,B/b,C/g,
E/r,G/b,H/r}
{A/r,B/b,C/g,
E/r, G/b,H/b}
{A/r,B/b,C/g,
E/r,G/b,H/g}