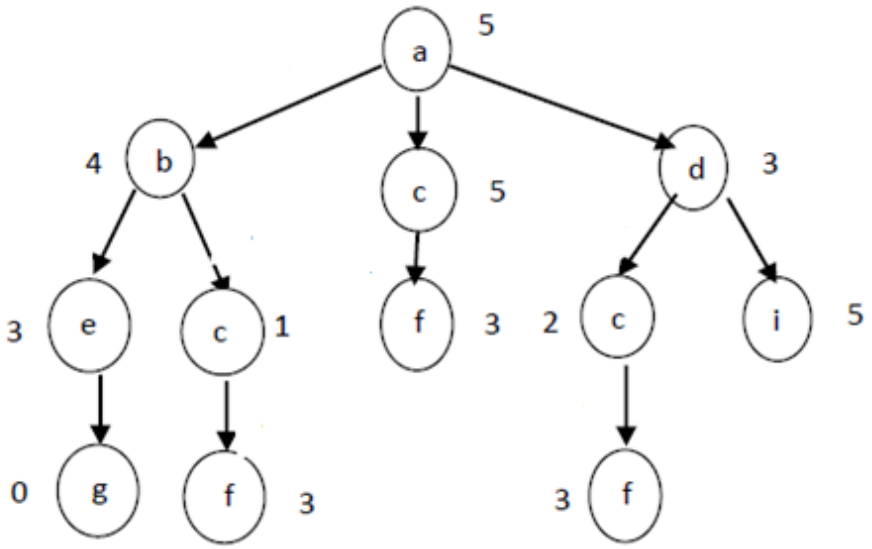
Lecture four

Topics that must be covered in this lecture:

- Another example of best first algorithm.
- Methods of Heuristic search (A- algorithm).
- Complex Search Space and problem solving Approach (in 8-puzzle problem).

Another example of best first algorithm:

Consider the directed graph shown below where the numbers at the states are heuristic estimates (h(n)). Note that the arcs are directed. Use a **best first algorithm** to find the goal where: A is the start state and G is the goal state.



Solution:

open	closed
[A5]	[]
[D3,B4,C5]	[A5]
[C2,B4,I5]	[D3,A5]
[F3,B4,I5]	[C2,D3,A5]
[B4,I5]	[F3,C2, D3,A5]
[C1,E3,I5]	[B4, F3,C2, D3,A5]
[E3,I5]	[C1, B4, F3, D3,A5]
[G0,I5]	[E3, C1, B4, F3, D3,A5]
	[G0, E3, C1, B4, F3, D3,A5]

3- A - Search algorithm

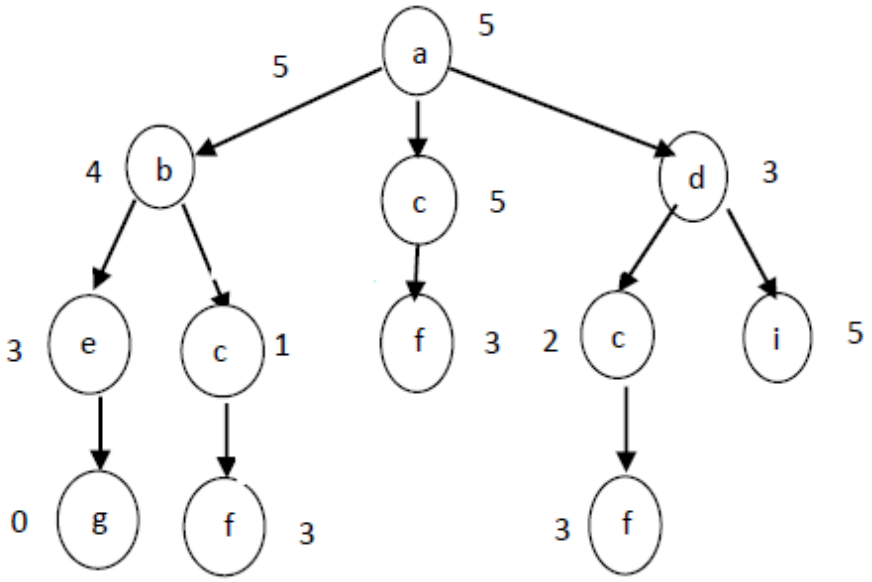
A algorithm is simply define as a best-first search plus a specific function. This specific function represents the actual distance (levels) between the initial state and the current state and is denoted by g(n). A notice will be mentioned here that the same steps that are used in the best first search are used in an A algorithm but in addition to the g (n) as follows;

$$F(n) = h(n) + g(n) \text{ , where:}$$

h(n):- is a heuristic estimate of the distance from state n to the goal.
g(n):- Measures the actual length of the path from any state (n) to the start state.

example about A-algorithm:

Consider the directed graph shown below where the numbers on the links are link costs and the numbers at the states are heuristic estimates (h(n)). Note that the arcs are directed. Use **A-algorithm** to find goal where: A be the start state and G be the goal state.

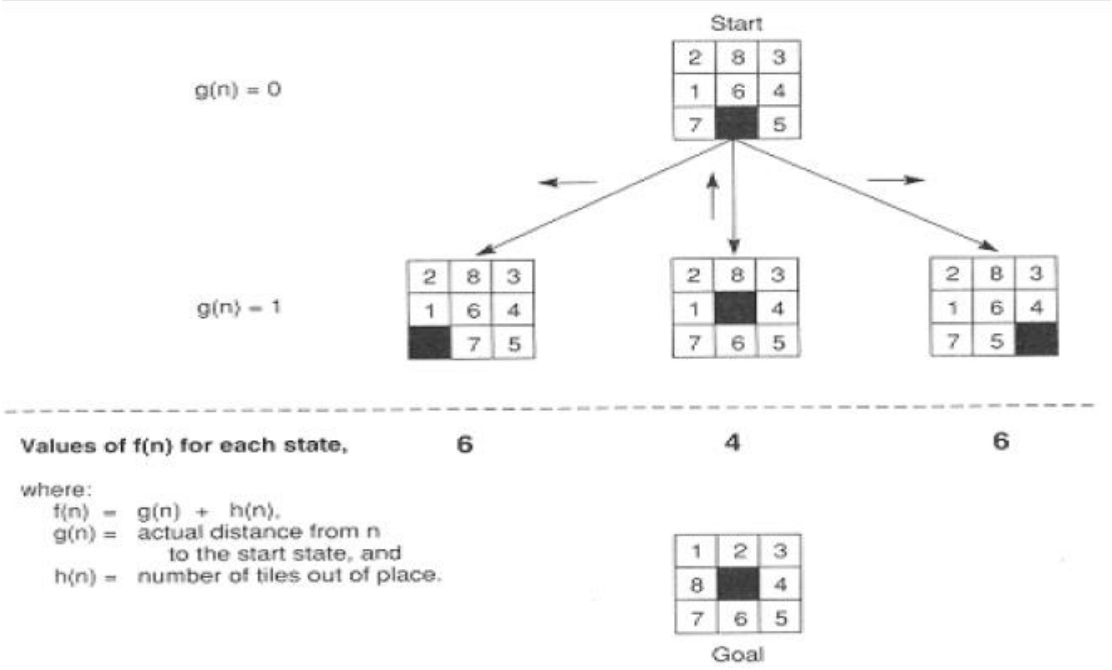


Solution:

open	closed
[A5]	[]
[D4,B5,C6]	[A5]
[C4,B5,I7]	[D4,A5]
[B5,F6,I7]	[C4,D4,A5]
[C3,E5,F6,I7]	[B5,D4,A5]
[E5,F6,I7]	[C3,B5,D4,A5]
[G3,F6,I7]	[E5,C3,B5,D4,A5]
	[G3,E5,C3,B5,D4,A5]

Complex Search Space and problem solving Approach (in 8-puzzle problem):

Example 1:



To summarize:

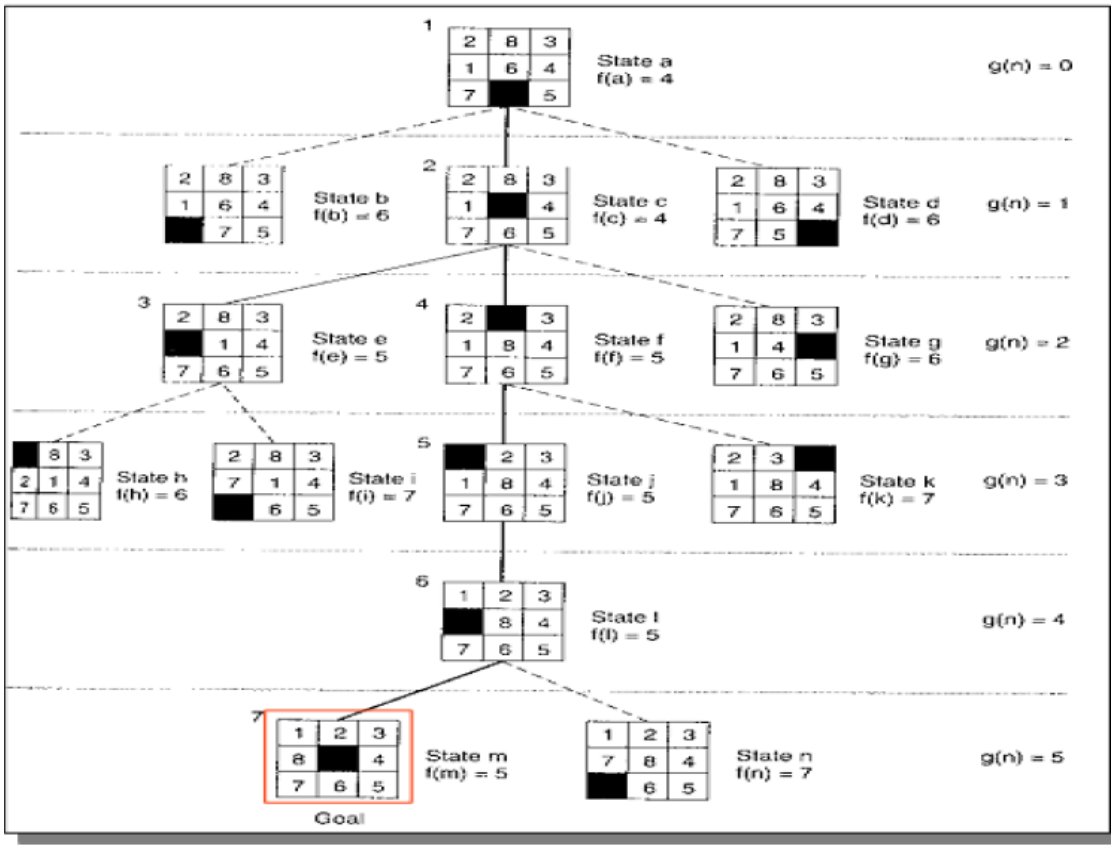
1. Operations on states generate children of the state currently under examination.

2. Each new state is checked to see whether it has occurred before (is on either open or closed), thereby preventing loops.

3. Each state n is given an I value equal to the sum of its depth in the search space g(n) and a heuristic estimate of its distance to a goal h(n). The h value guides search toward heuristically promising states while the **g** value prevents search from persisting indefinitely on a fruitless path.

4. States on open are sorted by their f values. By keeping all states on open until they are examined or a goal is found, the algorithm recovers from dead ends.

5. As an implementation point, the algorithm's efficiency can be improved through maintenance of the open and closed lists, perhaps as heaps or leftist trees.



After implementation of A algorithm, the Open and Closed is shown as follows:

1. Open=[a4], Closed=[]
2. Open=[c4,b6,d6],Closed=[a4]
3. Open=[e5,f5,b6,d6,g6],Closed=[c4,a4]
4. Open=[f5,b6,d6,g6,h6,i7],Closed=[e5, c4,a4]

5. Open=[j5,b6,d6,g6,h6,j7,k7], Closed=[,f5, e5,c4,a4]

6. Open=[l5, b6,d6,g6,h6,j7,k7],Closed=[j5,f5,e5,c4,a4]

7. Open=[m5, b6,d6,g6,h6,j7,k7,n7],Closed=[l5, j5,f5,e5, c4,a4]

8. Success, m=goal

Example2: Consider 8-puzzle problem with start state is shown as follows:

2	8	3
1	6	4
7		5

And the goal state is:

1	2	3
8		4
7	6	5

Assume the heuristic is calculated as following:

h (n) = -(number of tiles out of place)

Draw the path to get the goal using **Hill Climbing** search algorithm?

Answer: