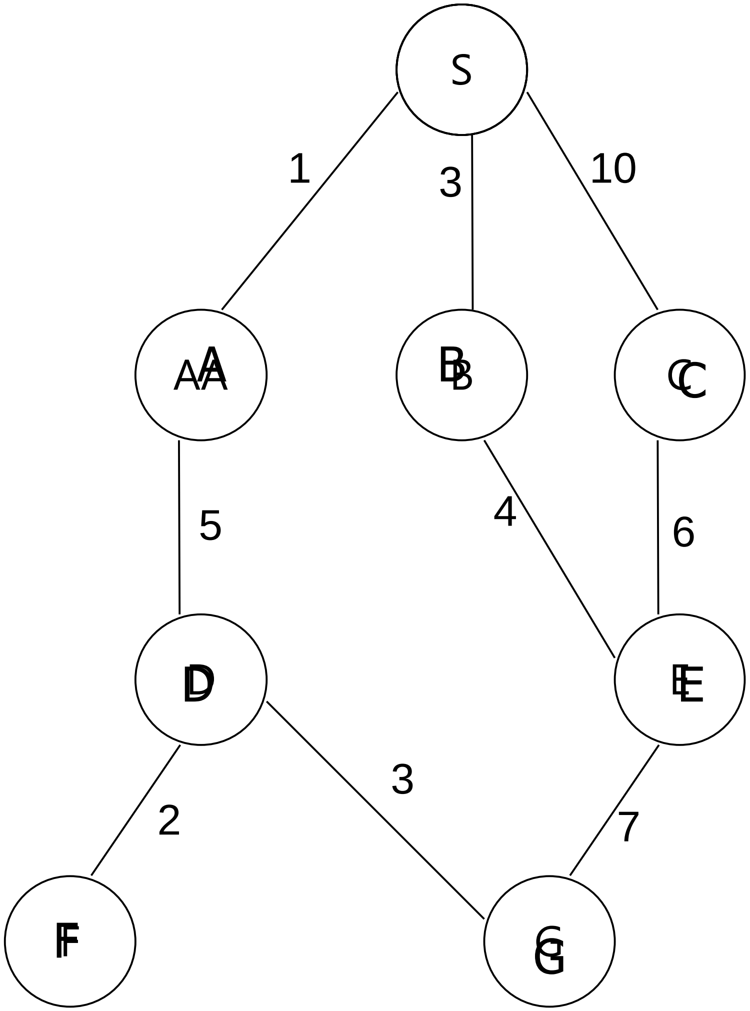
Artificial Intelligence

Algorithmics of Search

General Graph search Algorithm



Graph G = (V,E)

1) Open List : $S^{(\varnothing, 0)}$

Closed list : $\varnothing$

2) OL : $A^{(S,1)}$, $B^{(S,3)}$, $C^{(S,10)}$

CL : S

3) OL : $B^{(S,3)}$, $C^{(S,10)}$, $D^{(A,6)}$

CL : S, A

4) OL : $C^{(S,10)}$, $D^{(A,6)}$, $E^{(B,7)}$

CL: S, A, B

5) OL : $D^{(A,6)}$, $E^{(B,7)}$

CL : S, A, B , C

6) OL : $E^{(B,7)}$, $F^{(D,8)}$, $G^{(D, 9)}$

CL : S, A, B, C, D

7) OL : $F^{(D,8)}$, $G^{(D,9)}$

CL : S, A, B, C, D, E

8) OL : $G^{(D,9)}$

CL : S, A, B, C, D, E, F

9) OL : $\varnothing$

CL : S, A, B, C, D, E,
F, G

Steps of GGS
(*principles of AI, Nilsson,)*

- 1. Create a search graph *G*, consisting solely of the start node *S*; put *S* on a list called *OPEN.*
- *2.* Create a list called *CLOSED* that is initially empty.
- 3. Loop: if *OPEN* is empty, exit with failure.
- 4. Select the first node on *OPEN*, remove from *OPEN* and put on *CLOSED*, call this node *n*.
- 5. if *n* is the goal node, exit with the solution obtained by tracing a path along the pointers from *n* to *s* in *G*. (ointers are established in step 7).
- 6. Expand node *n*, generating the set *M* of its successors that are not ancestors of *n*. Install these memes of *M* as successors of *n* in *G*.

GGS steps (contd.)

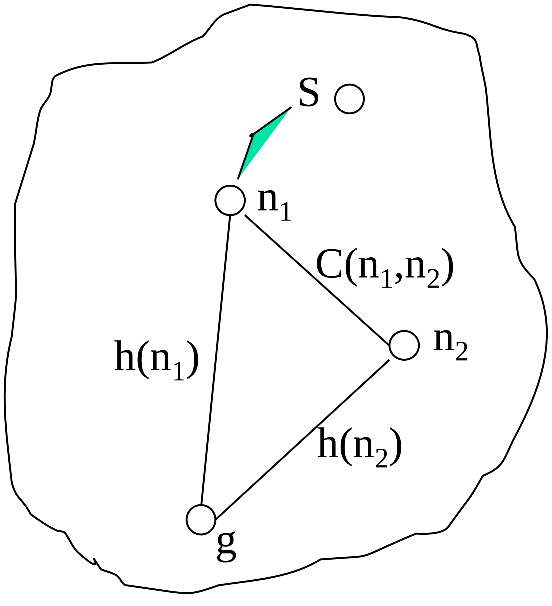
- 7. Establish a pointer to *n* from those members of *M* that were not already in *G* (*i.e.*, not already on either *OPEN* or *CLOSED*). Add these members of *M* to *OPEN*. For each member of *M* that was already on *OPEN* or *CLOSED*, decide whether or not to redirect its pointer to *n*. For each member of M already on *CLOSED*, decide for each of its descendents in *G* whether or not to redirect its pointer.
- 8. Reorder the list *OPEN* using some strategy.
- 9. Go *LOOP.*

GGS is a general umbrella

OL is a queue (BFS)

OL is stack (DFS)

OL is accessed by using a functions $f = g + h$ (Algorithm A)



$$h(n_1) \leq C(n_1, n_2) + h(n_2)$$

Algorithm A

- A function f is maintained with each node

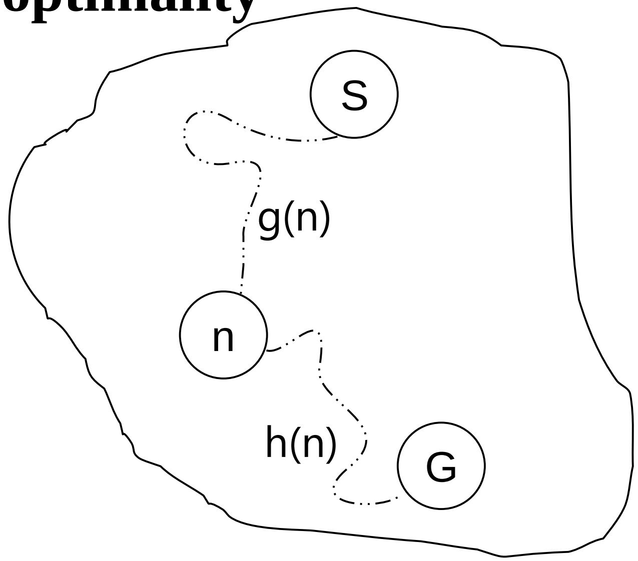
  $f(n) = g(n) + h(n)$, n is the node in the open list

- Node chosen for expansion is the one with least f value
- For BFS: $h = 0$, g = number of edges in the path to S
- For DFS: $h = 0$, $g = \frac{1}{\text{No of edges in the path to S}}$

Algorithm A*

- One of the most important advances in AI
- $g(n)$ = least cost path to n from S found so far
- $h(n) <= h^*(n)$ where $h^*(n)$ is the actual cost of optimal path to G(node to be found) from n

"Optimism leads to optimality"



Search building blocks

- State Space : Graph of states (Express constraints and parameters of the problem)
- Operators : Transformations applied to the states.
- Start state : S_0 (Search starts from here)
- Goal state : $\{G\}$ - Search terminates here.
- Cost : Effort involved in using an operator.
- Optimal path : Least cost path

Examples

Problem 1 : 8 – puzzle

4	3	6
2	1	8
7		5

S_0

1	2	3
4	5	6
7	8	

G

Tile movement represented as the movement of the blank space.

Operators:

L : Blank moves left

R : Blank moves right

U : Blank moves up

D : Blank moves down

$$C(L) = C(R) = C(U) = C(D) = 1$$

A*: Definitions and Properties

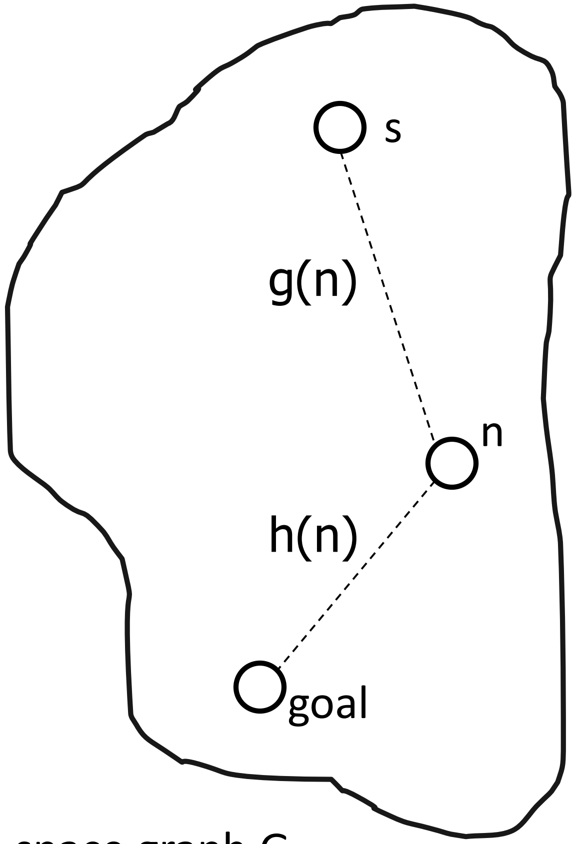
A* Algorithm – Definition and Properties

- *$f(n) = g(n) + h(n)$*
- The node with the least value of *f* is chosen from the *OL*.

- *$f^*(n) = g^*(n) + h^*(n)$*, where,
  $g^*(n)$ = actual cost of the optimal path *(s, n)*
  $h^*(n)$ = actual cost of optimal path *(n, g)*

- *$g(n) \geq g^*(n)$*

- By definition, *$h(n) \leq h^*(n)$*



State space graph G

8-puzzle: heuristics

Example: 8 puzzle

2	1	4
7	8	3
5	6	

s

1	6	7
4	3	2
5		8

n

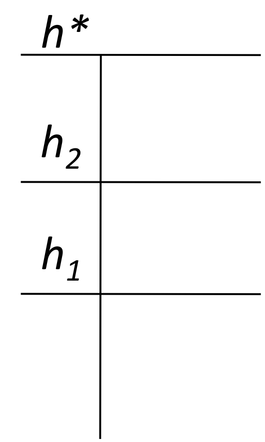
1	2	3
4	5	6
7	8	

g

$h^*(n)$ = actual no. of moves to transform n to g

1. $h_1(n)$ = no. of tiles displaced from their destined position.
2. $h_2(n)$ = sum of Manhattan distances of tiles from their destined position.

$$h_1(n) \leq h^*(n) \text{ and } h_2(n) \leq h^*(n)$$



Comparison

Classes of Search

Class	Name	Operation
Any Path Uninformed	Depth First Breadth First	Systematic exploration of the whole tree until a goal is found.
Any Path Informed	Best First	Uses Heuristic measure of Goodness of a state, eg. Estimated distance to goal.
Optimal Uninformed	Uniform Cost	Uses Path ‘length’ measure. Finds shortest path.
Optimal Informed	A*	Uses Path ‘length’ measure and Heuristic. Finds shortest path.