CS344: Introduction to Artificial Intelligence
(associated lab: CS386)

Pushpak Bhattacharyya
CSE Dept.,
IIT Bombay

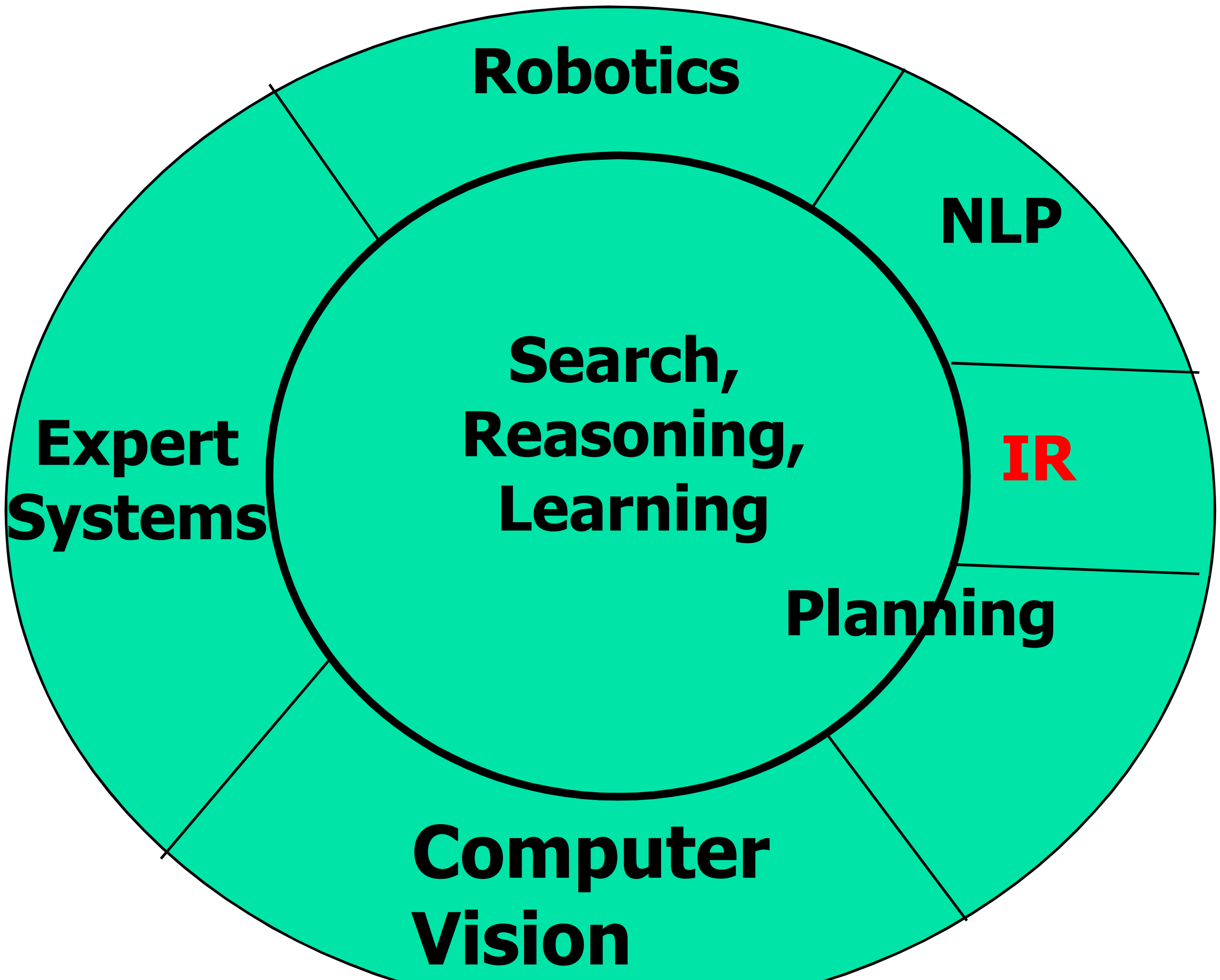
Lecture–2: Introduction (salient points repeat) + A*

4th Jan, 2011

Essential Facts

- Faculty instructor: Dr. Pushpak Bhattacharyya (www.cse.iitb.ac.in/~pb)
- TAs: *Ganesh, Kushal, Janardhan and Srijith "ganesh bhosale" <ganesh.bhosale.comp@gmail.com>, "Kushal Ladha" <kush@cse.iitb.ac.in>, <janardhan@cse.iitb.ac.in>, "Srijit Dutt" <srijitdutt@cse.iitb.ac.in>,*
- Course home page
  - www.cse.iitb.ac.in/~cs344-2011
- Venue: SIC 301, KR bldg
- 1 hour lectures 3 times a week: Mon-9.30, Tue-10.30, Thu-11.30 (slot 2)

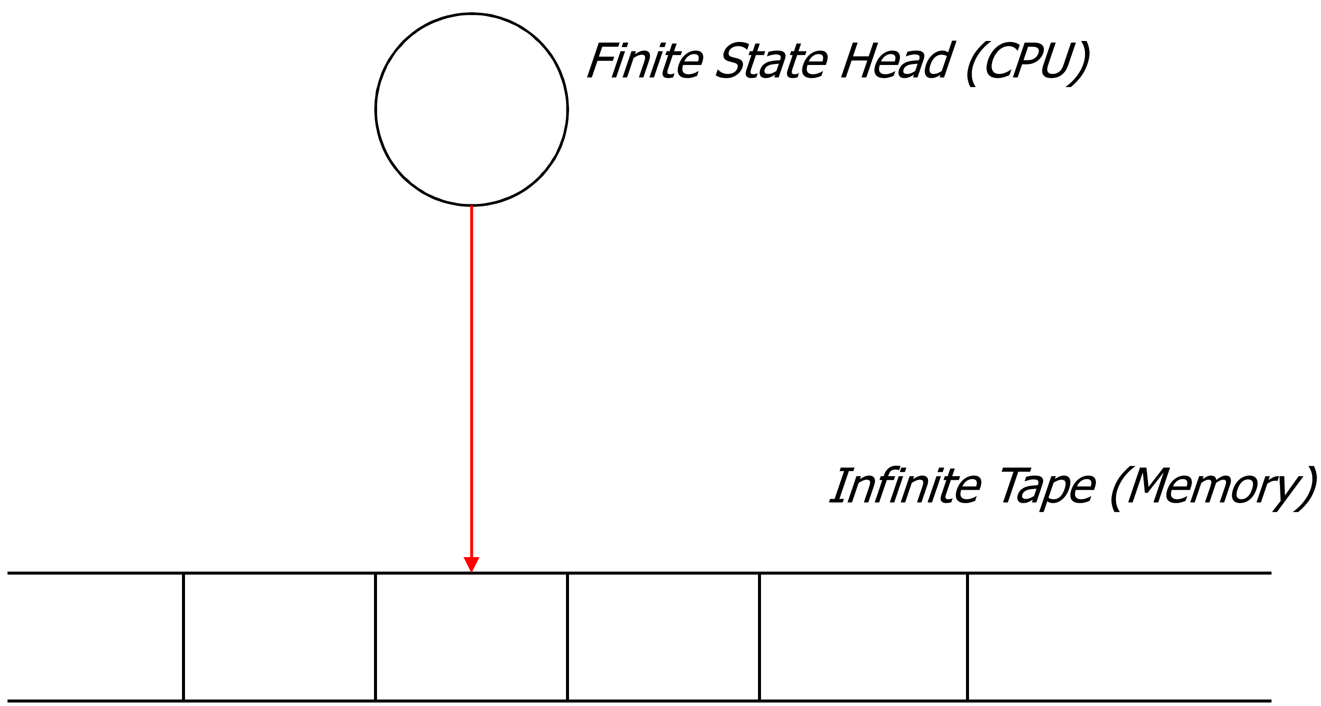
AI Perspective (post-web)



Foundational Points

- Church Turing Hypothesis
  - Anything that is computable is computable by a Turing Machine
  - Conversely, the set of functions computed by a Turing Machine is the set of ALL and ONLY computable functions

Turing Machine



Foundational Points *(contd)*

- Physical Symbol System Hypothesis (Newel and Simon)
  - *For Intelligence to emerge it is enough to manipulate symbols*

Foundational Points *(contd)*

- Society of Mind (Marvin Minsky)
  - *Intelligence emerges from the interaction of very simple information processing units*
  - *Whole is larger than the sum of parts!*

Foundational Points *(contd)*

- Limits to computability
  - *Halting problem: It is impossible to construct a Universal Turing Machine that given any given pair <M, I> of Turing Machine M and input I, will decide if M halts on I*
  - What this has to do with intelligent computation? *Think!*

Foundational Points *(contd)*

- Limits to Automation
  - *Godel Theorem: A "sufficiently powerful" formal system cannot be BOTH complete and consistent*
  - "Sufficiently powerful": at least as powerful as to be able to capture Peano's Arithmetic
  - Sets limits to automation of reasoning

Foundational Points *(contd)*

- Limits in terms of time and Space
  - *NP-complete and NP-hard problems: Time for computation becomes extremely large as the length of input increases*
  - *PSPACE complete*: Space requirement becomes extremely large
  - Sets limits in terms of resources

Two broad divisions of Theoretical CS

- Theory A
  - Algorithms and Complexity
- Theory B
  - Formal Systems and Logic

AI as the forcing function

- Time sharing system in OS
  - Machine giving the illusion of attending simultaneously with several people
- Compilers
  - Raising the level of the machine for better man machine interface
  - Arose from Natural Language Processing (NLP)
    - NLP in turn called the forcing function for AI

Goal of Teaching the course

- Concept building: firm grip on foundations, clear ideas
- Coverage: grasp of good amount of material, advances
- Inspiration: get the spirit of AI, motivation to take up further work

Resources

- Main Text:
  - Artificial Intelligence: A Modern Approach by Russell & Norvik, Pearson, 2003.
- Other Main References:
  - Principles of AI - Nilsson
  - AI - Rich & Knight
  - Knowledge Based Systems – Mark Stefik
- Journals
  - AI, AI Magazine, IEEE Expert,
  - Area Specific Journals e.g, Computational Linguistics
- Conferences
  - IJCAI, AAAI

Positively attend lectures!

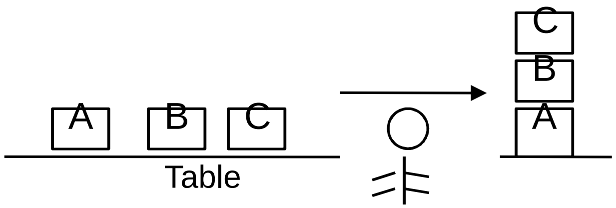
Grading

- Midsem
- Endsem
- Group wise assignments (closely follows lectures)
- Paper reading (possibly seminar)
- Quizzes

Search: Everywhere

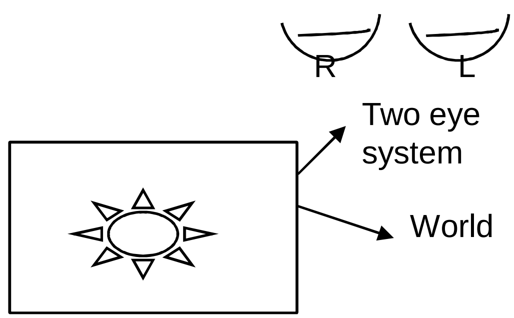
Planning

- (a) which block to *pick*, (b) which to *stack*, (c) which to *unstack*, (d) whether to *stack* a block or (e) whether to *unstack* an already stacked block. These options have to be searched in order to arrive at the right sequence of actions.



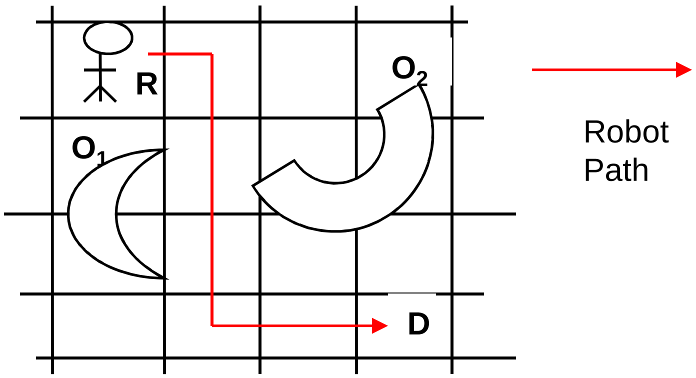
Vision

- A search needs to be carried out to find which point in the image of *L* corresponds to which point in *R*. Naively carried out, this can become an *O(n2)* process where *n* is the number of points in the retinal images.



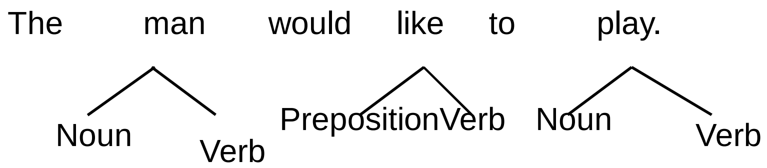
Robot Path Planning

- searching amongst the options of moving ***L***eft, ***R***ight, ***U***p or ***D***own. Additionally, each movement has an associated cost representing the relative difficulty of each movement. The search then will have to find the *optimal*, *i.e.*, the *least cost* path.



Natural Language Processing

- search among many combinations of parts of speech on the way to deciphering the meaning. This applies to every level of processing- *syntax, semantics, pragmatics* and *discourse*.



Expert Systems

Search among rules, many of which can apply to a situation:

If-conditions

the infection is primary-bacteremia
AND the site of the culture is one of the sterile sites
AND the suspected portal of entry is the gastrointestinal tract

THEN

there is suggestive evidence (0.7) that infection is bacteroid

(from MYCIN)

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

1	2	3
4	5	6
7	8	

G

Tile movement represented as the movement of the blank space.

Operators:

L : Blank moves left

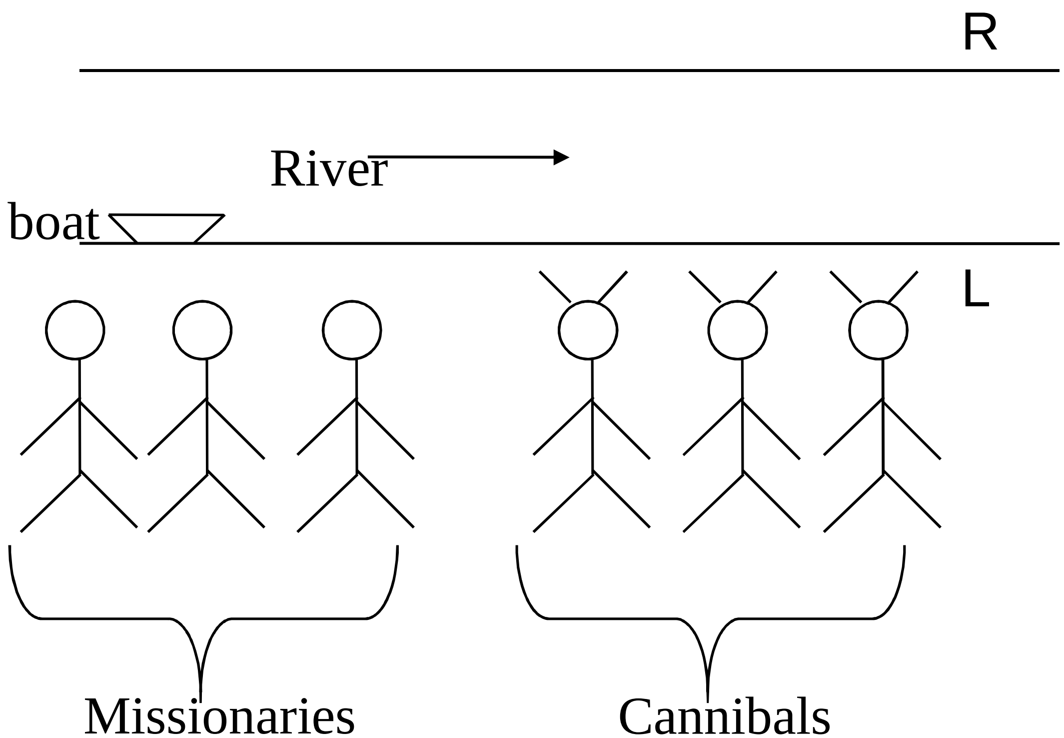
R : Blank moves right

U : Blank moves up

D : Blank moves down

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

Problem 2: Missionaries and Cannibals



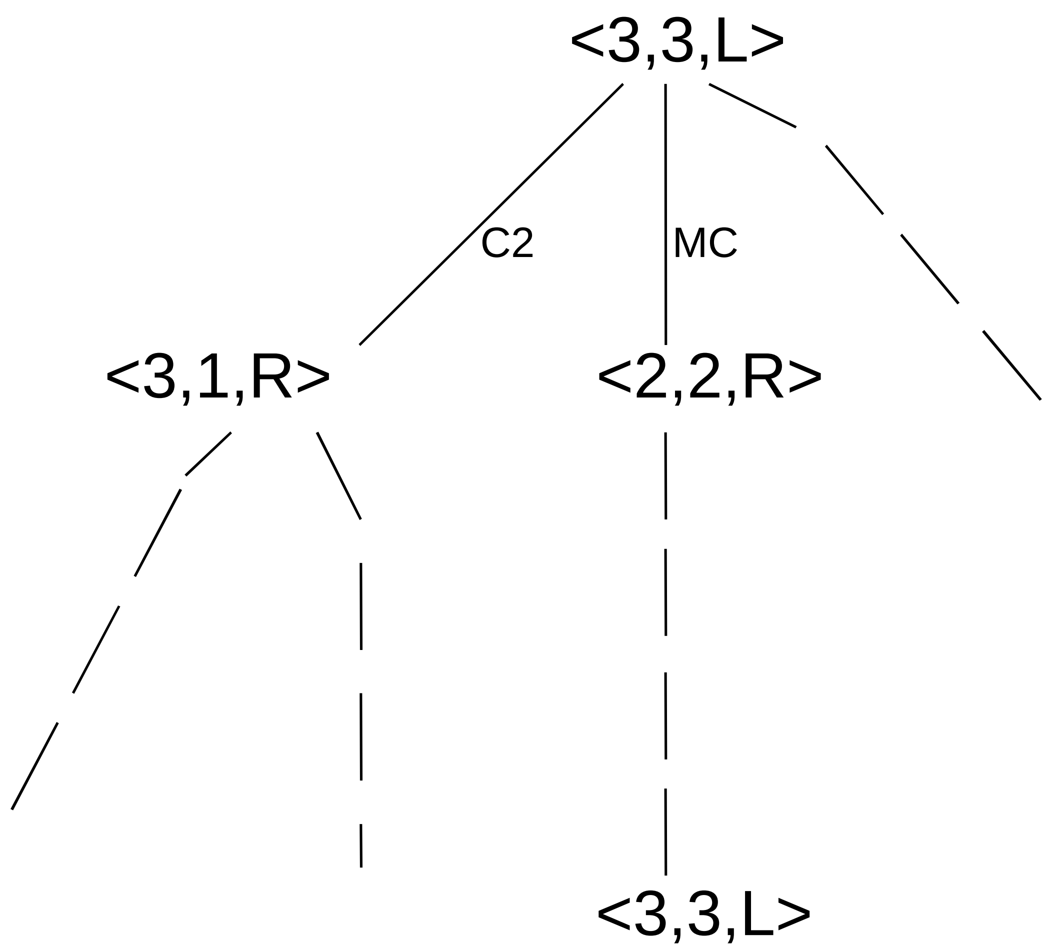
<u>Constraints</u>

- The boat can carry at most 2 people
- On no bank should the cannibals outnumber the missionaries

State : *<#M, #C, P>*
#M = Number of missionaries on bank *L*
#C = Number of cannibals on bank *L*
P = Position of the boat

S0 = <3, 3, L>
G = < 0, 0, R >

<u>Operations</u>
M2 = Two missionaries take boat
M1 = One missionary takes boat
C2 = Two cannibals take boat
C1 = One cannibal takes boat
MC = One missionary and one cannibal takes boat



Partial search tree

Problem 3

B	B	B	W	W	W	

G: States where no **B** is to the left of any **W**

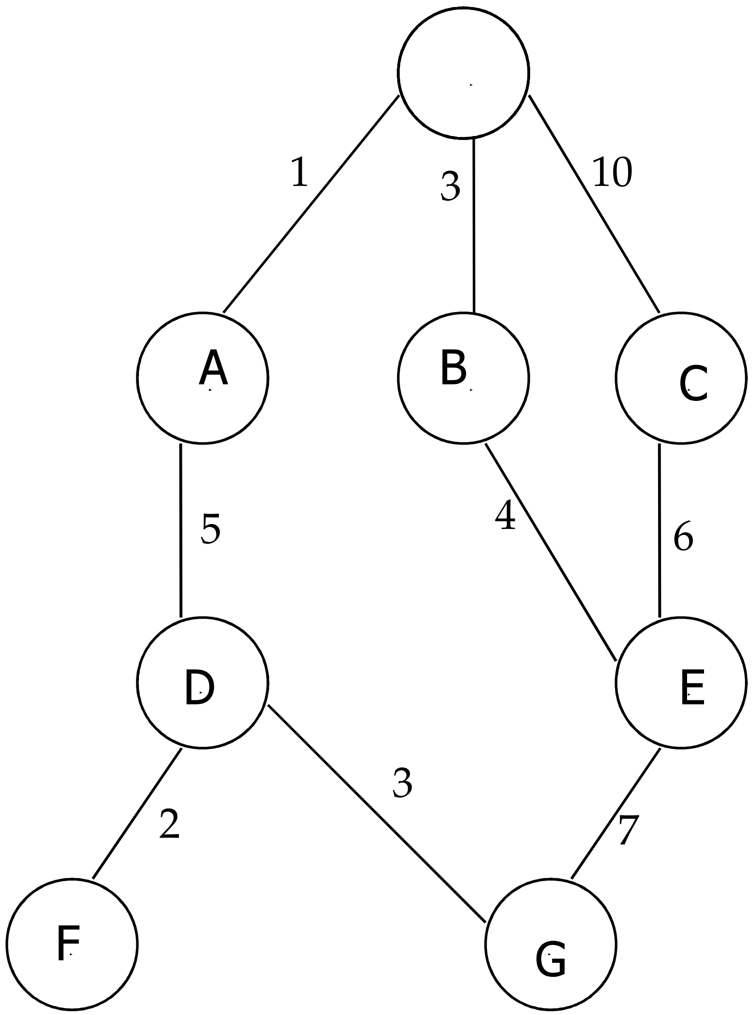
Operators:

1) A tile jumps over another tile into a blank tile with cost 2

2) A tile translates into a blank space with cost 1

Algorithmics of Search

General Graph search Algorithm



Graph G = (V,E)

1) Open List : S $^{(Ø, 0)}$
Closed list : Ø

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 : Ø
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.)

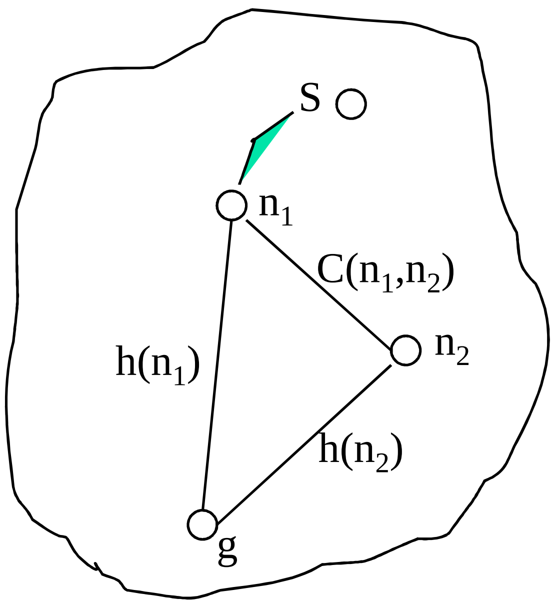
- 7. Establish a pointer to *n* from those members of *M* that were not already in *G* (*i.e.*, not already on either *OPEN* or *CLOSED*). Add these members of *M* to *OPEN*. For each member of *M* that was already on *OPEN* or *CLOSED*, decide whether or not to redirect its pointer to *n*. For each 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

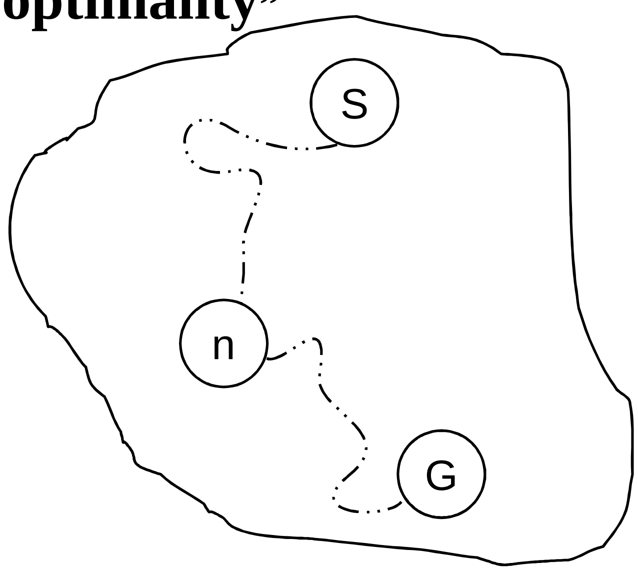
- A function f is maintained with each node

  $f(n) = g(n) + h(n)$, 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"



A*: Definitions and Properties

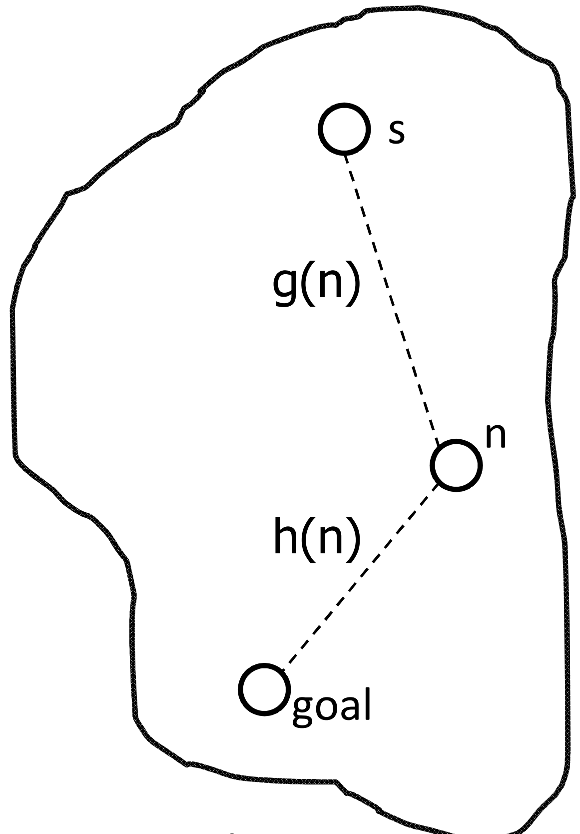
A* Algorithm – Definition and Properties

- $f(n) = g(n) + h(n)$
- 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

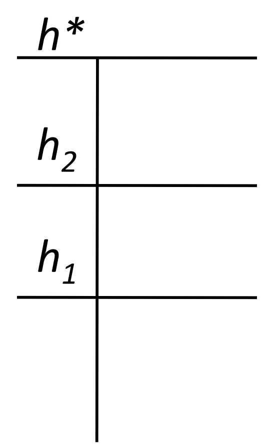
1	2	3
4	5	6
7	8	

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.

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



Comparison

A* Algorithm- Properties

- **Admissibility**: An algorithm is called admissible if it always terminates and terminates in optimal path
- **Theorem**: A* is admissible.
- **Lemma**: Any time before A* terminates there exists on *OL* a node *n* such that $f(n) <= f^*(s)$
- **Observation**: For optimal path $s \rightarrow n_1 \rightarrow n_2 \rightarrow \ldots \rightarrow g$,
  1. $h^*(g) = 0$, $g^*(s)=0$ and
  2. $f^*(s) = f^*(n_1) = f^*(n_2) = f^*(n_3) \ldots = f^*(g)$

A* Properties *(contd.)*

$f^*(n_i) = f^*(s)$, $n_i \neq s$ and $n_i \neq g$

Following set of equations show the above equality:

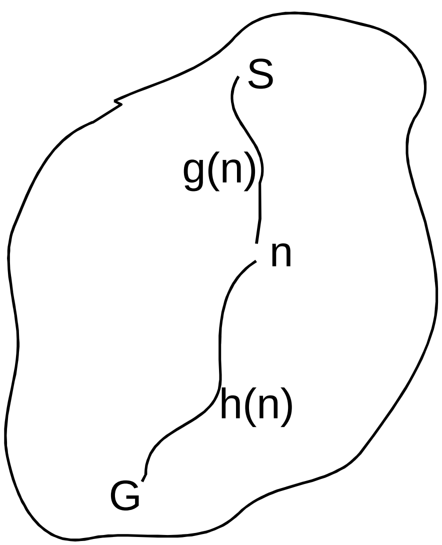
$$f^*(n_i) = g^*(n_i) + h^*(n_i)$$
$$f^*(n_{i+1}) = g^*(n_{i+1}) + h^*(n_{i+1})$$
$$g^*(n_{i+1}) = g^*(n_i) + c(n_i, n_{i+1})$$
$$h^*(n_{i+1}) = h^*(n_i) - c(n_i, n_{i+1})$$

Above equations hold since the path is optimal.

Admissibility of A*

A* always terminates finding an optimal path to the goal if such a path exists.

Intuition



(1) In the open list there always exists a node *n* such that $f(n) <= f^*(S)$.

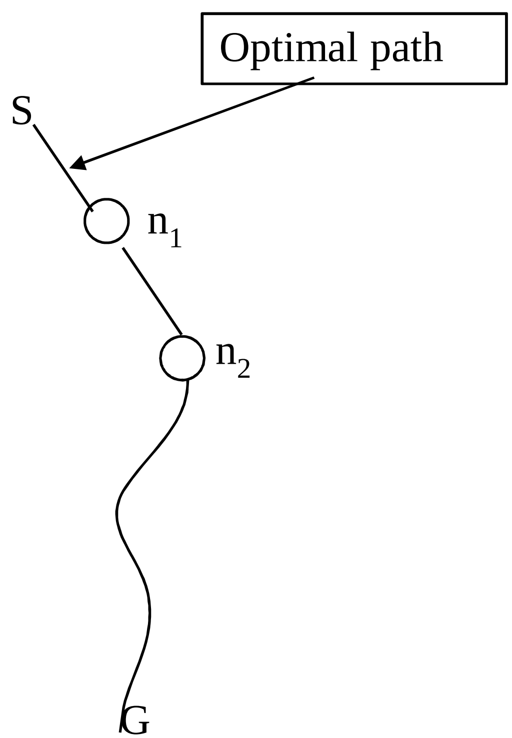
(2) If A* does not terminate, the *f* value of the nodes expanded become unbounded.

1) and 2) are together inconsistent

Hence A* must terminate

Lemma

Any time before A* terminates there exists in the open list a node n' such that $f(n') <= f^*(S)$



For any node n_i on optimal path,

$f(n_i) = g(n_i) + h(n_i)$

$<= g^*(n_i) + h^*(n_i)$

Also $f^*(n_i) = f^*(S)$

Let n' be the first node in the optimal path that is in OL. Since all parents of n' have gone to CL,

$g(n') = g^*(n')$ *and* $h(n') <= h^*(n')$

$=> f(n') <= f^*(S)$

If A* does not terminate

Let *e* be the least cost of all arcs in the search graph.

Then $g(n) >= e.l(n)$ where $l(n)$ = # of arcs in the path from *S* to *n* found so far. If A* does not terminate, $g(n)$ and hence $f(n) = g(n) + h(n)$ $[h(n) >= 0]$ will become unbounded.

This is not consistent with the lemma. So A* has to terminate.

2nd part of admissibility of A*

The path formed by A* is optimal when it has terminated

Proof

Suppose the path formed is not optimal

Let *G* be expanded in a non-optimal path.

At the point of expansion of *G*,

$f(G) = g(G) + h(G)$

$= g(G) + 0$

$> g^*(G) = g^*(S) + h^*(S)$

$= f^*(S)$ [$f^*(S)$ = cost of optimal path]

This is a contradiction

So path should be optimal