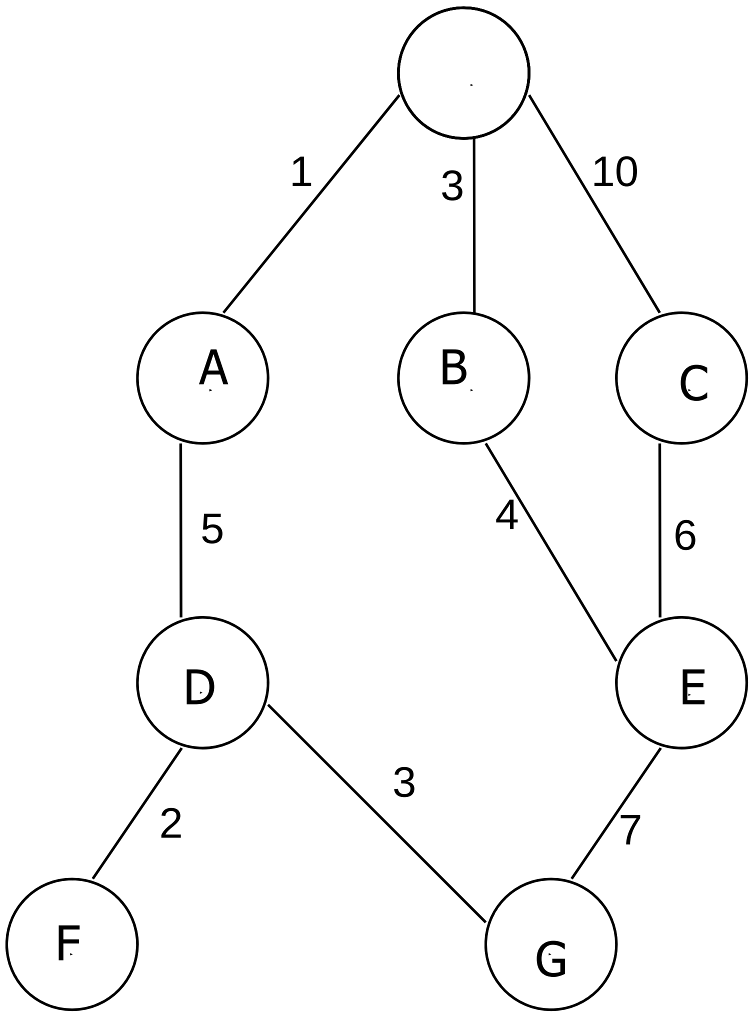
CS344: Introduction to Artificial Intelligence

Pushpak Bhattacharyya
CSE Dept.,
IIT Bombay

Lecture 17– Theorems in A* (admissibility, Better performance of more informed heuristic, Effect of Monotone Restriction or Triangular Inequality)

[Main Ref: Principle of AI by N.J. Nilsson]

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.*

Algorithm A

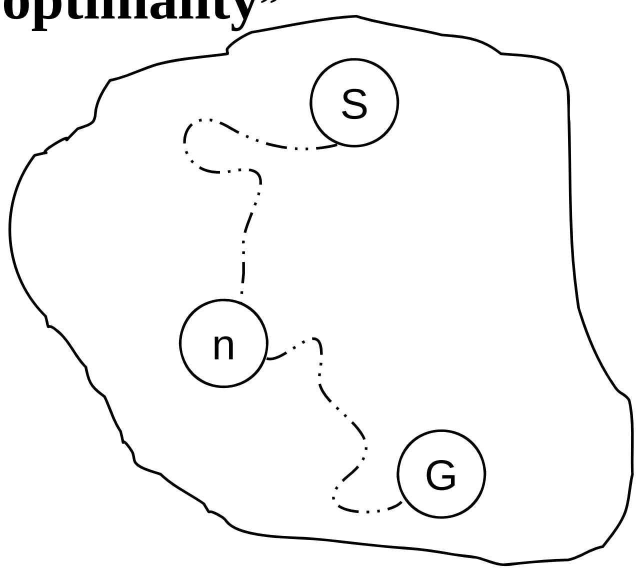
- A function f is maintained with each node

  $f(n) = g(n) + h(n)$, n is the node in the open list
- Node chosen for expansion is the one with least f value

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* 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.)*

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

Following set of equations show the above equality:

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

$$f^*(n_{i+1}) = g^*(n_{i+1}) + h^*(n_{i+1})$$

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

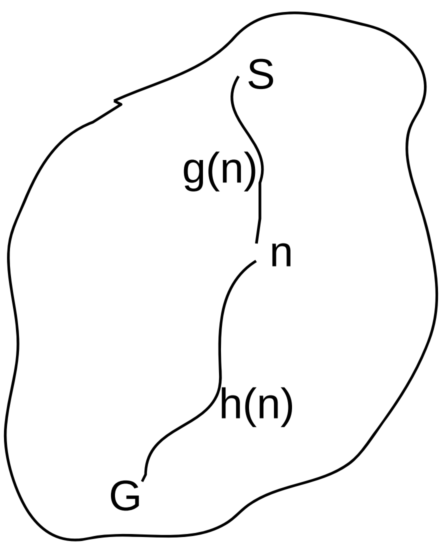
$$h^*(n_{i+1}) = h^*(n_i) - c(n_i, n_{i+1})$$

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 $f(n) <= f^*(S)$.

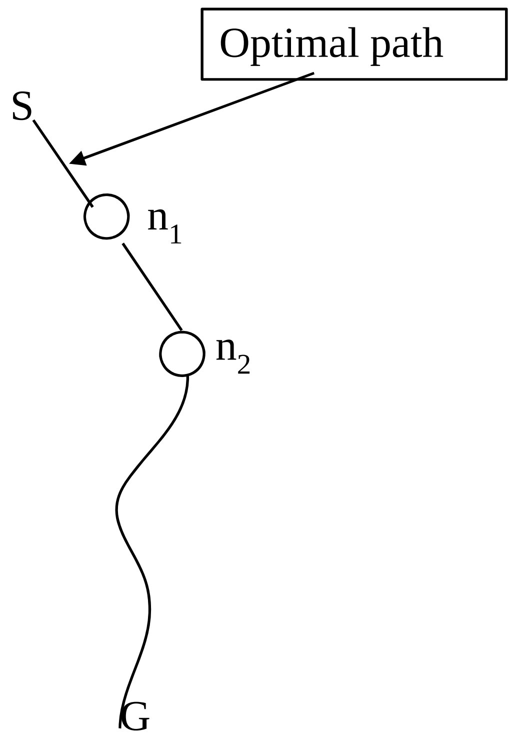
(2) If A* does not terminate, the *f* value of the nodes expanded become unbounded.

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 <u>all</u> 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.

2^{nd} 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

Better Heuristic Performs Better

Theorem

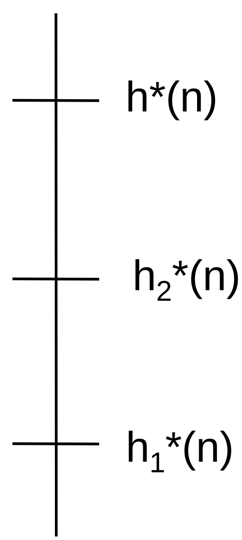
A version A_2^* of A* that has a "better" heuristic than another version A_1^* of A* performs at least "as well as" A_1^*

Meaning of "better"

$h_2(n) > h_1(n)$ for all n

Meaning of "as well as"

A_1^* expands at least all the nodes of A_2^*



For all nodes n, except the goal node

Proof by induction on the search tree of A_2*.

A* on termination carves out a tree out of *G*

Induction

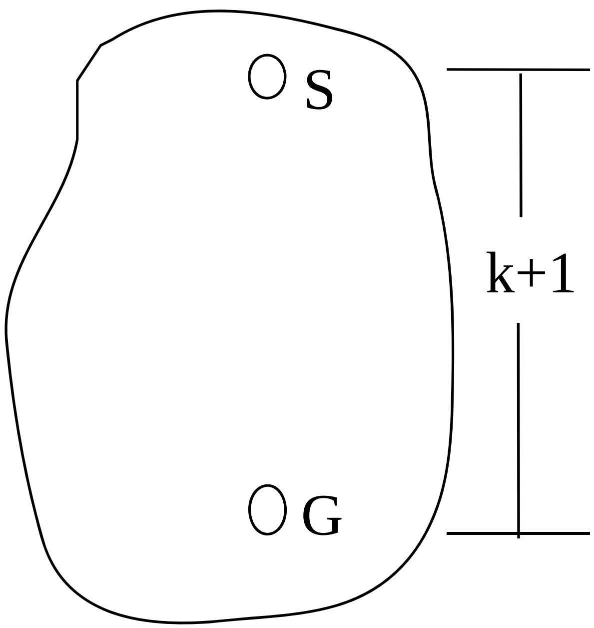
on the depth *k* of the search tree of A_2*. A_1* before termination expands all the nodes of depth *k* in the search tree of A_2*.

k=0. True since start node *S* is expanded by both

Suppose A_1* terminates without expanding a node *n* at depth *(k+1)* of A_2* search tree.

Since A_1* has seen all the parents of *n* seen by A_2*

$g_1(n) <= g_2(n) \qquad (1)$



Since A_1^* has terminated without expanding *n*,
$f_1(n) >= f^*(S)$ (2)

Any node whose *f* value is strictly less than $f^*(S)$ has to be expanded.
Since A_2^* has expanded *n*
$f_2(n) <= f^*(S)$ (3)

From (1), (2), and (3)
$h_1(n) >= h_2(n)$ which is a contradiction. Therefore, A_1^* has to expand all nodes that A_2^* has expanded.

<u>Exercise</u>

If better means $h_2(n) > h_1(n)$ for some *n* and $h_2(n) = h_1(n)$ for others, then Can you prove the result ?

Monotone Restriction or Triangular Inequality of the Heuristic Function

Statement:

if monotone restriction (also called triangular inequality) is satisfied, then for nodes in the closed list, redirection of parent pointer is not necessary.

In other words, if any node '*n*' is chosen for expansion from the open list, then $g(n)=g(n^*)$, where $g(n)$ is the cost of the path from the start node '*s*' to '*n*' at that point of the search when '*n*' is chosen, and $g(n^*)$ is the cost of the optimal path from '*s*' to '*n*'.

A heuristic $h(p)$ is said to satisfy the monotone restriction, if for all '*p*', $h(p) <= h(p_c) + cost(p, p_c)$, where '$p_c$' is the child of '*p*'.

Proof

- Let S-N_1- N_2- N_3- N_4... N_m ...N_k be an optimal path from S to N_k (all of which might or might not have been explored).
- Let N_m be the **last** node on this path which is on the open list, i.e., *all* the ancestors from S up to N_{m-1} are in the closed list.

Proof *(contd.)*

- For every node N_p on the optimal path,
  - $g*(N_p)+h(N_p)<= g*(N_p)+C(N_p,N_{p+1})+h(N_{p+1})$, by monotone restriction
  - $g*(N_p)+h(N_p)<= g*(N_{p+1})+h(N_{p+1})$ on the optimal path
  - $g*(N_m)+ h(N_m)<= g*(N_k)+ h(N_k)$ by transitivity
- Since all ancestors of N_m in the optimal path are in the closed list,
  - $g (N_m)= g*(N_m)$
  - $=> f(N_m)= g(N_m)+ h(N_m)= g*(N_m)+ h(N_m)<= g*(N_k)+ h(N_k)$

Proof *(contd.)*

- For every node N_p on the optimal path,
  - $g^*(N_p)+h(N_p)<= g^*(N_p)+C(N_p,N_{p+1})+h(N_{p+1})$, by monotone restriction
  - $g^*(N_p)+h(N_p)<= g^*(N_{p+1})+h(N_{p+1})$ on the optimal path
  - $g^*(N_m)+ h(N_m)<= g^*(N_k)+ h(N_k)$ by transitivity
- Since all ancestors of N_m in the optimal path are in the closed list,
  - $g\ (N_m)= g^*(N_m)$
  - $=> f(N_m)= g(N_m)+ h(N_m)= g^*(N_m)+ h(N_m)<= g^*(N_k)+ h(N_k)$

Proof *(contd.)*

- Now if N_k is chosen in preference to N_m,
  - $f(N_k) <= f(N_m)$
  - $g(N_k)+ h(N_k) <= g(N_m)+ h(N_m)$
  - $= g^*(N_m)+ h(N_m)$
  - $<= g^*((N_k)+ h(N_k)$
  - Hence, $g(N_k)<=g^*(N_k)$
- But $g(N_k)>=g^*(N_k)$, by definition
- Hence $\boldsymbol{g(N_k)=g^*(N_k)}$ *--proved*

Relationship between Monotone Restriction and Admissibility

- **MR=>Admissibility, but not *vice versa***
  - *i.e.*, if a heuristic *h(p)* satisfies the monotone restriction, for all '*p*', $h(p) \le h(p_c) + cost(p, p_c)$, where '$p_c$' is the child of '*p*', *then*
  - $h^*(p) \le h^*(p)$, for all *p*

Forward proof

- Let $p \rightarrow n_1 \rightarrow n_2 \rightarrow n_3 \rightarrow \ldots n_{k-1} \rightarrow G = n_k$, be the *optimal* from p to G
- By definition, $h(G)=0$
- Since $p \rightarrow n_1 \rightarrow n_2 \rightarrow n_3 \rightarrow \ldots n_{k-1} \rightarrow G = n_k$ is the optimal path from p to G,
- $C(n_1,n_2)+c(n_2,n_3)+\ldots+c(n_{k-1},n_k)=h^*(p)$

Forward proof *(contd.)*

Now by M.R.

$h(p) <= h(n_1) + c(p, n_1)$

$h(n_1) <= h(n_2) + c(n_1, n_2)$

$h(n_2) <= h(n_3) + c(n_2, n_3)$

$h(n_3) <= h(n_4) + c(n_3, n_4)$

...

$h(n_{k-1}) <= h(G) + c(n_{k-1}, G)$

$h(G)=0;$ summing the inequalities,

$h(p) <= C(n_1, n_2) + c(n_2, n_3) + \ldots + c(n_{k-1}, n_k) = h^*(p);$ *proved*

Backward proof, by producing a counter example.

Lab assignment

- Implement A* algorithm for the following problems:
  - 8 puzzle
  - Missionaries and Cannibals
  - Robotic Blocks world
- Specifications:
  - Try different heuristics and compare with baseline case, *i.e.*, the breadth first search.
  - Violate the condition $h \leq h^*$. See if the optimal path is still found. Observe the speedup.