# Lab # 07 - Informed Search

#### **Best First Searches**

Best first search uses the concept of a **priority queue** and **heuristic search**. It is a search algorithm that works on a specific rule. The aim is to reach the **goal** from the **initial** state via the shortest path. If we consider searching as a form of **traversal in a graph**, an *uninformed search* algorithm would blindly traverse to the next node in a given manner without considering the cost associated with that step.

An **informed search**, like **Best first search**, on the other hand would use **an evaluation function** to decide which among the various available nodes is the most promising (or 'BEST') before traversing to that node.

To search the graph space, the BFS method uses two lists for tracking the traversal. An 'Open' list which keeps track of the current 'immediate' nodes available for traversal and 'CLOSED' list that keeps track of the nodes already traversed. In the python implementation OPEN list is alternatively named as QUEUE and CLOSED list as VISITED.

**Greedy** Best First Search and A\* are the two most popular variants of Best First Searches.

There are various flavors of Best First Search algorithm with different heuristic evaluation functions f(n). The only difference between **Greedy** BFS and  $A^*$  BFS is in the evaluation function. Essentially, since  $A^*$  is more optimal of the two approaches as it also takes into consideration the total distance travelled so far i.e. g(n).

f(n) = Evaluation Function, g(n) = Total cost from Initial to current state h(n) = An estimate value from current state to the goal state

g(n)	Path Distance
h(n)	Estimate to Goal
f(n)	Combined Hueristics i.e. g(n) + h(n)

For Greedy BFS the evaluation function is: f(n) = h(n)For A\* the evaluation function is: f(n) = g(n) + h(n).

Let's have a look at the graph below and try to implement both Greedy Best-First-Search and A\* algorithms step by step using the two list, OPEN and CLOSED.

			S with e	evaluati = h(n)	on		1	A* BFS	with	eval	uatio g			tion f(n)	= h(n) +
Ste	ep 1 - St	art by	adding t	he start r	nod	е	Γ								
(S)	to the o	pen li	st with th	ne path			S	tep 1 - 9	Start b	y add	ling th	ie	start r	node (S) f	to the
dis	tance a	s 0					O	pen list	with t	ne pat	th dist	а	nce as	0	
	OPE	N	CLC	SED					OPE	N			CL	OSED	
	Node	h(n)	Node	Parent Node				Node	g(n)	h(n)	f(n)		Node	Parent Node	
	S	10						S	0	10	10				
							Γ					Г			
Re	peat the	next	steps un	til the OF	PEN		Γ								
Lis	t is emp	ty or t	he Goal	node is r	nov	ed	R	epeat th	ne nex	t step	s unt	il	the OF	PEN List i	s empty
to	the CLO	SEDI	ist				01	the Go	al no	de is r	nove	d	to the	CLOSED	list

OPEN	l list to nediat	the s	ve the firs e CLOSE uccessor et	D list an	d e	expand		Step 2 (a list to the immedia the OPE	CLOS te suc	SED I	st and	d	expand	l its	ΞN
OPI	EN		CLO						OPE	N				SED	
Node	h(n)		Node	Parent Node				Node	g(n)	h(n)	f(n)		Node	Parent Node	
Α	9		S		Т		Γ	Α	7	9	16	Г	S		П
В	7				Т		Γ	В	2	7	9	Γ			
С	8				T			С	3	8	11				П
					Ι										

			he list in a			)							scending	
order o	f the c	combined	hueristic v	alue	<del>)</del>			order o	of the o	combi	ned <u>h</u>	ueristic v	alue	
OPE	N	CL	DSED						OPE	N		CL	OSED	
Node	h(n)	Node	Parent Node					Node	g(n)	h(n)	f(n)	Node	Parent Node	
В	7	S						В	2	7	9	S		
С	8							С	3	8	11			
Α	9							Α	7	9	16			

OPEN	list to nediat	the CLOS e success	irst node in SED list and ors by addi	d ex	(pai		to the	CLO	SED	list and	j	expand i	n the OPEN list ts immediate e OPEN list	st
OPE	ΞN	CLC	DSED					OF	PEN			С	LOSED	
Node	h(n)	Node	Parent Node				Node	g(n)	h(n)	f(n)		Node	Parent Node	
С	8	S		П		T	С	3	8	11		S		Γ
Α	9	В	S	П		$\top$	Α	7	9	16		В	S	Γ
D	8			П	$\Box$	T	D	6	8	14				Г
Н	6					I	Н	3	6	9				
				Ш										

				the list in asce d <u>hueristic</u> valu		ding		Step 3 the co						scending order o
OPE	ΞN		C	LOSED		П		OP	EN			(	CLOSED	
Node	h(n)	П	Node	Parent Node	Г		П	Node	g(n)	h(n)	f(n)	Г	Node	Parent Node
Н	6		S		Г		П	Н	3	6	9		S	
С	8		В	S			П	С	3	8	11		В	S
Þ⊡	8				ļ .			D	6	8	14	ļ		
Α	9						П	Α	7	9	16			
							П							

Step 4 (a) - Move the first node in the OPEN list to the CLOSED list and expand its immediate successors by adding them to the OPEN list

Step 4 (a) - Move the first node in the OPEN list to the CLOSED list and expand its immediate successors by adding them to the OPEN list

OPE	N	CL	OSED					OPE	N		CLC	OSED	
Node	h(n)	Node	Parent Node			Node g(n) h(n) f(n) C 3 8 11				f(n)	Node	Parent Node	
С	8	S					С	3	8	11	S		П
D	8	В	S				D	6	8	14	В	S	П
Α	9	Н	В				Α	7	9	16	Н	В	П
F	6						F	6	6	12			П
G	3						G	5	3	8			П

			e list in ascendi ueristic value	ng		Step 4 (				n ascendi	ng order of
OPE	N	CI	LOSED				OPI	ΞN		CLO	SED
Node	h(n)	Node	Parent Node			Node	g(n)	h(n)	f(n)	Node	Parent Node
G	3	S				G	5	3	8	S	
F	6	В	S			С	3	8	11	В	S
С	8	Н	В			F	6	6	12	Н	В
D	8				П	D	6	8	14		
Α	9					Α	7	9	16		

Step 5 (a) - Move the first node in the OPEN list to the CLOSED list and expand its immediate successors by adding them to the OPEN list

Step 5 (a) - Move the first node in the OPEN list to the CLOSED list and expand its immediate successors by adding them to the OPEN list

OPE	N		C	LOSED					OPE	ΞN		(	CLOSED	
Node	h(n)	n(n) Node Parent No			П			Node	g(n)	h(n)	f(n)	Node	Parent Node	
F	6		S		П			O	3	8	11	S		
С	8		В	S	П			F	6	6	12	В	S	
D	8		Н	В	П			D	6	8	14	Н	В	
Α	9	Г	G	Н	Т			Α	7	9	16	G	Н	
E	0							Ε	7	0	7			

Step 5 (	b) - R	e-ord	ler the li	st in ascen	din	g or	der		Step 5	(b) - F	Re-orde	er the li	st in asc	ending order o	f
of the co	ombin	ed <u>h</u> u	ueristic v	/alue					the con	nbine	d hueri:	stic val	ue		
OPE	N		CLO	DSED						OP	EN		C	LOSED	
Node	h(n)		Node	Parent Node					Node	g(n)	h(n)	f(n)	Node	Parent Node	
E	0		S					Ε	7	0	7	S			
F	6		В	S					С	3	8	11	В	S	
С	8		Н	В				Г	F	6	6	12	Н	В	
D	8		G	Н				Г	D	6	8	14	G	Н	
Α	9							Γ	Α	7	9	16			
								Γ							

list to t	he CL iate su	OSED lis	first node in the t and expand <u>it's</u> by adding then	è				the CL	OSED	list an	nd exp	and <u>it's</u> in	the OPEN list to nmediate OPEN list	
OPE	EN	(	CLOSED						OPE	ΞN			CLOSED	
Node	h(n)	Node	Parent Node	П		П	Т	Node	g(n)	h(n)	f(n)	Node	Parent Node	П
F	6	S				П	Т	С	3	8	11	S		П
С	8	В	S	П		П	Т	F	6	6	12	В	S	П
D	8	Н	В	П		П	Т	D	6	8	14.5	Н	В	П
Α	9	G	Н	П		П	Τ	Α	7	9	16	G	Н	П
		E	G	П		П	T					E	G	П
				П		П	T							П
moved	to the	CLOSEI get the o	as the Goal node D list. Backtrack ptimal path (E	th	e			to the	CLOSE	ED list	Back		ode (E) is moved closed list to get B> S)	

Even though you would find that both Greedy BFS and A\* algorithms find the path equally efficiently, number of steps, you may notice that the A\* algorithm is able to come up with is a more optimal path than Greedy BFS. So in summary, both Greedy BFS and A\* are Best first searches but Greedy BFS is neither complete, nor optimal whereas A\* is both complete and optimal. However, A\* uses more memory than Greedy BFS, but it guarantees that the path found is optimal.

### **Advantages and Disadvantages of Best First Search**

## Advantages:

- 1. Can switch between BFS and DFS, thus gaining the advantages of both.
- 2. More efficient when compared to DFS.

# Disadvantages:

1. Chances of getting stuck in a loop are higher.

Try changing the graph and see how the algorithms perform on them.