

## Measuring Problem Solving Performance

completeness  
optimality

metric to measure AI performance  
(most of the cases (গুরুতরু) practically used parameter)

মানে problem soln এর ক্ষেত্রে একটা algorithm use করলে, যামানের algo  
→ solvable

মনের সাথে বিনা → completeness

যামানের algo দিয়ে best soln পাই বিনা → optimality

## Uninformed Search Strategies:

search strategy

- uninformed / blind
- goal related (মানে রবতা ক্ষেত্র)
- info use করিব
- informed

## Breadth-First Search:

একটা node কে visit করার পর দেখছি এই node এর children কোনো  
expanded, unexpanded

FIFO → fringe Data structure use

প্রথম root node দখিঃ এটি goal state না হলে এটা কে expand করিঃ  
এই child কুলকে fringe দেখাই,  
shallowest node expand first

$b \rightarrow$  branching factor

time complexity  $O(b^{d+1})$

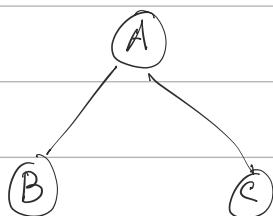
space complexity  $O(b^{d+1})$

1. completeness এর aspect যেখে  $\rightarrow$  যোগো node  $\Rightarrow$  untouched (কারবে না)  
unexpanded/  
unvisited

১০ যদি সোন আকে BFS অন্যান্য যদি সুস্থিত পার।  
 $\hookrightarrow$  complete.

2. optimality: সোন যেটি পাই যেটি best son র বাছাবাছি কিনা

fringe যেহেতু FIFO অন্ত এই node আগে এখনো যেটি আগে expanded হবে  
আমরা left child রে আগে রাখ।



B এর cost 5

C এর cost 3

এখানে visit রে উচিত ছিল,

১০ BFS guided না।

But uniform cost search  $\rightarrow$  exactly BFS but সবগুলো vertex  
কি-visit করার cost same

uniform cost search ৰে cost এৰ basis ৰে sort কৰে minimum cost এৰটা expand হয় so optimal হবে।

## Depth First Search

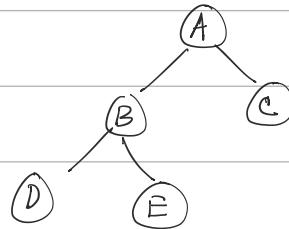
→ LIFO

completeness ৰে কাৰণা আছে।

ex: নিৰ্দিষ্ট computational budget আছে। এৰ মাধ্যমে কোন চাই, যিৰি tree টো  
depth প্রেমি হয়, তখন দেখা যাবে computational budget কেৰে হয়ে যাবে  
বিন্দু কোন চাচ্ছা পাইনা।

(যদ্যন যোৰ limited and soন পৰিৱে  
দিবে আছে তখন এই case) search কৰিবাব।

Completeness → No



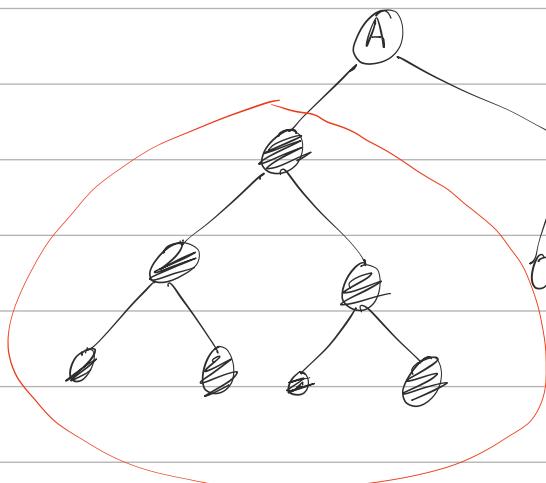
$A \rightarrow B \rightarrow D \rightarrow E \dots$

let C হলো কোন বিন্দু

পৰা পাওয়াৰ আগৈ

budget কেৰে কোন পাবাব।

So complete no.



দ্বাৰা নীচে আৰ বিছুনী। so ওই

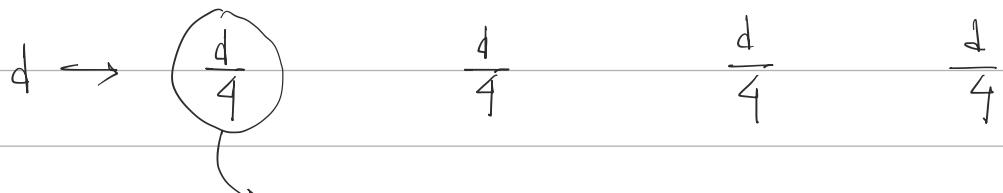
part কো memoryৰ বাইবে নিতে পাৰি

ধৰণঘৰ্য্যা  $L(n)$  depth. Because deepest node পুলোৱে মাত্ৰা search বাবে  
↪ depth limited search  $\rightarrow$  dfs-এর infinite loop এৰা problem solve কৰিব।

ex: depth  $d$

original dfs এ  $d$  পৰ্য্যন্ত মাধ্যা (ধৰণীয় branch এ),  
then পৰে ঠৰিব।

depth limited search এ  $d$  বৰ্তমানে for example: ১ অংশ কৰিব।



এখন এই কোনো  $d/4$  বৰ্তমানে।

এতো সোনা পাইন নেক্ষে অন্যোন্যো।

১ অংশ মে কৰিব এই ১ বিধিবে select কৰিবো?

↪ সম্ভৱত problem limit কোৱে সেই কৰা,

এতো user বৰ্তমানে choose কৰিব দেখো performance differ কৰিব।

অটো limit কো আটো set কৰিব কৰিব।

↪ iterative deepening search (IDS)

Iterative deepening search ( $L=1$ )

↪ ১ limit এ কো কো কো কো এক এক কৰে

depth limited search এর limit set কৰি problem fix.

এটা limit বাইরে DFS এর দিকে যাবে। কিন্তু এটা BFS এর মতো।

numerical comparison : memory requirement (এখন এটা এটা node generate করে)

$$b = 10, \quad d = 5$$

N<sub>DLS</sub> 1 1 1 1 1

N<sub>IDS</sub> 1 2 3 4 5 0

N<sub>BFS</sub> 1 1 1 1 1 0 0

DFS এর জন্ম থার্ভিং particular branch visit  
শেষে memory র বাইরে রাখতে পারি।

preferable  $\rightarrow$  IDS

IDS  $\rightarrow$  complete

$\hookrightarrow$  depth limit করে দেয়া

DFS  $\rightarrow$  complete না,

space complexity linear  $O(bd)$

BFS  $\hookrightarrow O(b^d) \rightarrow$  exponential

BFS, DFS  $\rightarrow$  search এবং জায়গা থেকে সুরক্ষিত। এটি একটি move করা  
যাবে, unidirectional search.

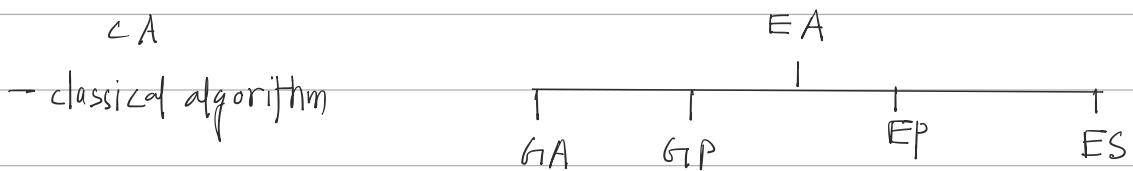
## Bi-directional Search

ex: tree এর root থেকে BFS

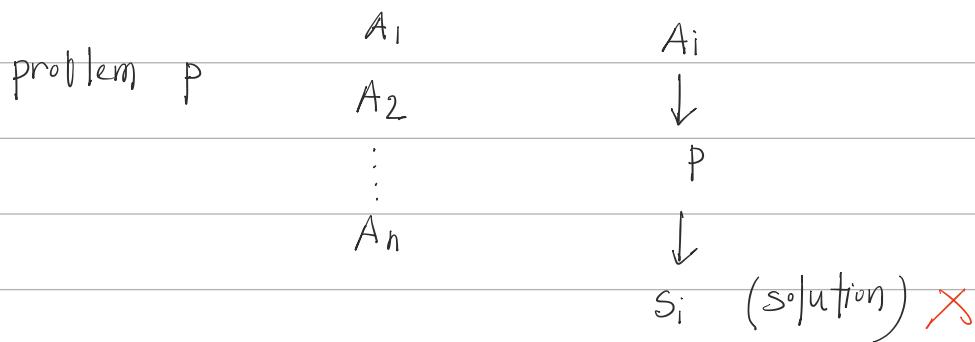
leaf থেকে DFS

এবং point to meet করবে।

Local search  $\rightarrow$  genetic / evolutionary (multiple direction search).



problem p ടോടി ഫോർ സോൾവ എൻവ ഫോർ പ്രിട്ടി, preference ഫൂല്ഫൂലി എൻട് ആം റാറി (Ai)



যদি requirement ঘূর্যোগী so  
produce হয় নাই, তার ক্ষেত্রে একটি  
selection ক্ষিতি হয় নাই।

मिस्टर चॉज़ ने चुना चुना



চূঢ়ান্ত behaviour always expected এবং ধৈর্য lower হয়

EA → family of algorithm

Genetic algo

" programming

Evolutionary "

" static

GA

GP

EP

ES

multiple direction দ্বারা  
search করে, timing  
(বেশি সময়ে) solution  
quality will be good.