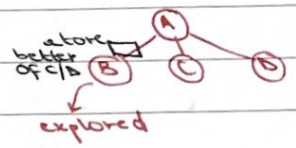
Till 27/9/2022

BFS
DFS
UCS
IDS
DLS
greedy | BFS | A* | IDA* | SMA* | RBFS

26/9/2022.

RBFS Algorithm (Recursive BFS) (Tree version)

↳ Tries to optimize memory.



* each time node gen., next best sibling stored

Root → gen. all successors.

next expansion ⇒ min. child cost.

can show / discard parent gen. from 1st successor expansion.

value stored only if stored value < parent's stored value of

else if stored value of next best sibling > parent stored value

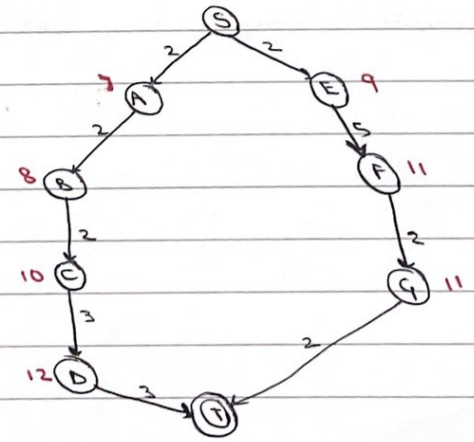
⇒ carry forward parent stored val.

If now on gen new level of successors, (no scope of proceeding)

no val < parent, go to stored parent val.

↳ update f-cost of parent to min. of successors.

Q)



SMA* → bound: on capacity of memory.

(simplified memory A*)

Forget node (w highest f cost)

remember @ parent level

update if cost less.

RULES

⇒ remove highest f-cost leaf & remember @ parent

⇒ If all child nodes generated, adjust f-value of parent to min.

⇒ give up too long paths by setting to '∞' not reach goal if beyond.

Now, if another node is to be forgotten w cost > (cost remembered from 1st forgotten)

directly discard.

If goal found w cost < parent cost,

regenerate; backtrack.

NOTE ⇒ Forgotten nodes also gen.

∴ when 2 succ. & 1 is forgotten & other gen,

adjust f cost of parent ✓

If after a particular path, you have NOTHING TO FORGET,

set to ∞. eg: after (E) only (9) left, set (E) to ∞