

OUTLINE Class #19

Homework: Read Chapters 9 & 10 in Nilsson

Agents that Plan & Search {see Class 17-18 notes}

Uninformed Search (Chapter 8)

- Formulating the State Space
- Components of Implicit State Graphs
- Breadth-First Search
- Backtracking & Depth-First Search

PROCEDURE GRAPH-SEARCH

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 it from *OPEN*, and put it on *CLOSED*. Call this node n .
5. If n is a goal node, exit successfully with the solution obtained by tracing a path along the pointers from n to s in G . (see step 7.)
6. Expand node n , generating the set, M , of its successors and install them as successors of n in G .
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 descendants in G whether or not to redirect its pointer.
8. Reorder the list *OPEN*, either according to some arbitrary scheme or according to heuristic merit.
9. GOLOOP

Recursive Procedure BACKTRACK1(DATALIST)

1. $DATA \leftarrow first(DATALIST)$
2. *if member(DATA, rest(DATALIST)) return fail*
3. *if term(DATA) return nil*
4. *if deadend(DATA) return fail*
5. *if length(DATALIST) > BOUND return fail*
6. $RULES \leftarrow Apprules(DATA)$
7. LOOP: *if null(RULES) return fail*
8. $R \leftarrow first(RULES)$
9. $RULES \leftarrow rest(RULES)$
10. $RDATA \leftarrow R(DATA)$
11. $RDATALIST \leftarrow cons(RDATA,DATALIST)$
12. $PATH \leftarrow BACKTRACK1(RDATALIST)$
13. *if PATH=fail go LOOP*
14. *return cons(R,PATH)*