322
12/01

SEARCH

→ First part of course = Deterministic, Focus on Search.
→ Remember that seides are reported at the end of day.

• HIGHLIGHTS FROM STORY:

* Chess: search explores space of possible moves and consequence
  - 1st topic of research: Search (IBM)
  - Famous story: Gary Kasparov × Deep Blue
    ↳ world champion ↳ program
    → Deep Blue won 3, lost 2, tied 1
    → Search up to 126.000.000 nodes/sec
    + resources.

. Still used as base for current research.

• SEARCH

given specification of what a solution is
↓
search for a solution

* enumerate a set of potential partial solutions
* check to see if are solutions or could lead to one.
  ↳ GOAL

• SIMPLE SEARCH AGENT:

* Deterministic + goal-driven agent:
  ↳ start state
  ↳ given Goal → simplification, not true in real life
  ↳ (only) agent action → environment changes
  ↳ agent knows { action (given any state) = effects
  next state
  ↳ take agent from $\underbrace{\text{Initial} \rightarrow \text{Goal}}_{=\text{solution}}$.

322
12/01

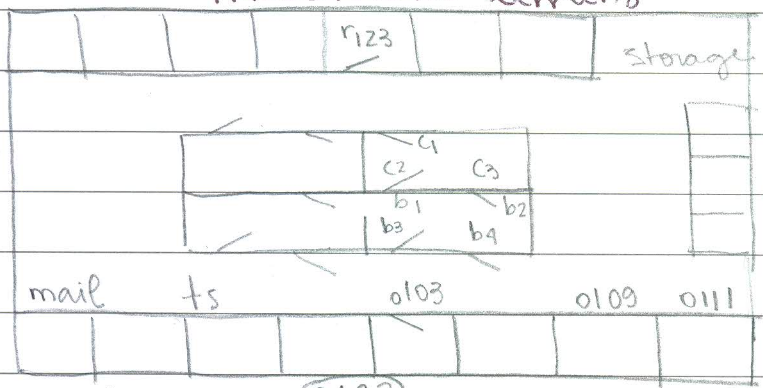
DEFINITIONS – SEARCH PROBLEM

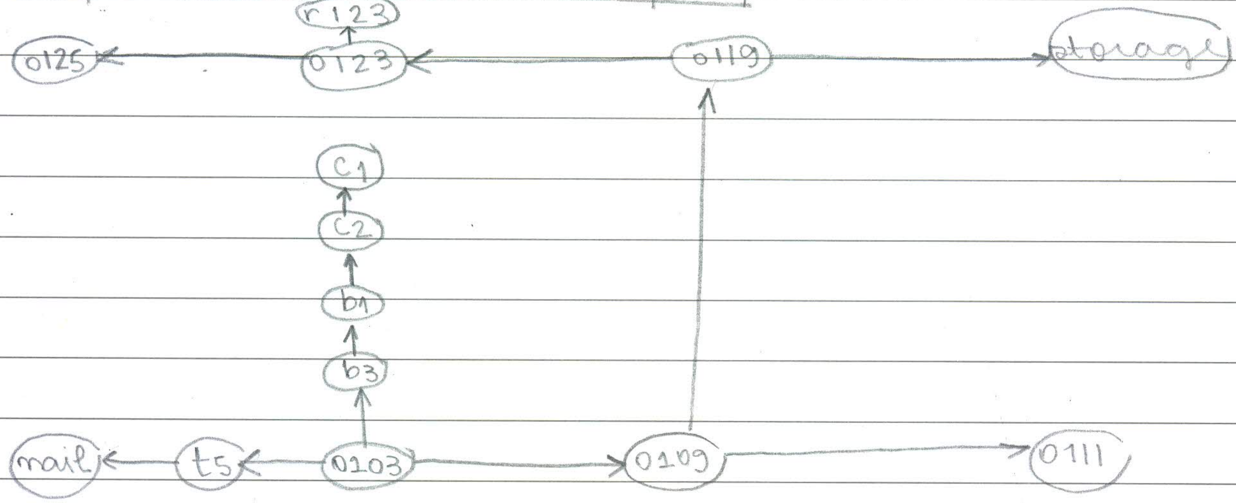
- Initial State(s)
- agent design operators (set of actions)
- Action function: state, action → new state
- Goal state(s)
- Search space: set of states that will be searched for a path from initial state to goal, given available actions.
  - → states = nodes
  - → actions = links
- Path Cost (later).

EXAMPLES:

① Delivery Robot – route in building } on book

- movement limits





322
12/01

② Vacuum cleaner world

States { 2 rooms: r_1, r_2
Each can be dirty/not
vacuum can be at r_1 or r_2

Features: entities { vacuum (robot) - properties { r_1 / r_2
rooms - properties { dirty / not

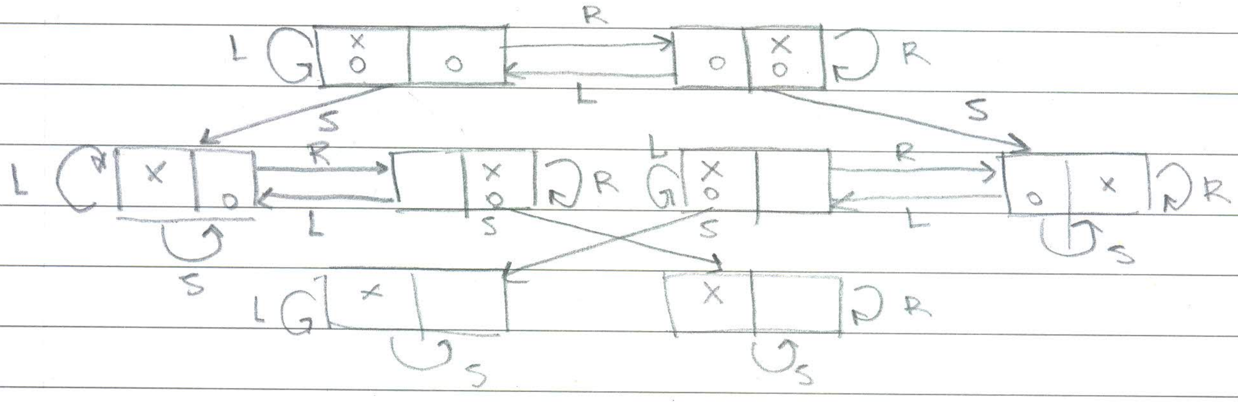
$dr_1(r) = T/F$
$dr_2(r) = T/F$
$loc = r_1$ or r_2
} all binary features

number of states = 8 (2^3)

Trickier: K rooms → $K * 2^K$

* States: x = vacuum, o = dirty

* Actions: left, right, suck
  - Successor states in graph show effect of each actions applied to a given state
* Goal: no dirt



③ EIGHT PUZZLE

* STATES: each state specifies which number/blank occupies each of the 9 tiles
  → 9! states.
* ACTIONS: Blank_UP, Blank_down, Blank_left, blank_right
* GOAL: Sequence, etc. Blank on middle.
  → Search Space:

HOW CAN WE FIND A SOLUTION:

- find sequence of actions and appropriate order to lead to goal
  ↳ agent to solve: given space graph, need smart ways to search into space and get into the goal.

SEARCH:

① Start at start state
② Evaluate the effect of taking different actions starting from states that have been encountered in the search so far.
③ Stop when goal state is encountered

GRAPHS SPECIFICATION FOR DELIVERY ROBOT (Example 1):

N = {mail, ts, o103, b3, o109, ...}
A = {<ts, mail>, <o103, ts>, <o103, b3>, <o103, o109>, ...}
One of several solution paths: <o103, o109, o119, o123, r123>

C

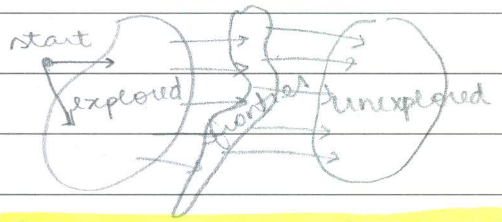
322
12/01

GRAPH SEARCHING

→ Generic Search Algorithms:

- given graph, start nodes and goal nodes
  - ↳ incrementally explore paths from start nodes
- maintain a frontier of paths
  (have been explored)
- Expand frontier as search continues
  until goal node is encountered.

* the way in which the frontier is expanded defines
the search strategy.



GENERIC SEARCH ALGORITHM:

Input: { graph
set of start nodes
boolean procedure goal(n) if n is a goal

frontier := [<s> : s is a start node];
↳ path that only contains start node

while frontier is not empty: { through graph
    select and remove $\boxed{\text{path}}$ $\langle n_0, \dots, n_k \rangle$ from frontier
    if goal (n_k)
        return $\langle n_0, \dots, n_k \rangle$ } solution
    for every neighbor n of n_k
        add $\langle n_0, \dots, n_k, n \rangle$ to frontier } new path for each of neighbors' to frontier
end

Clicker: b^n nodes n steps away, factor b, graph = tree
at every level, every node has b more.

NEXT: 3.5 textbook, AISpace Practice Ex 3.A, definitions