Beyond Classical Search:

- Searching with non-deterministic actions
- Searching with partial observations
- Online search & unknown environments

Chapter 4, Sections 4.3-4.5

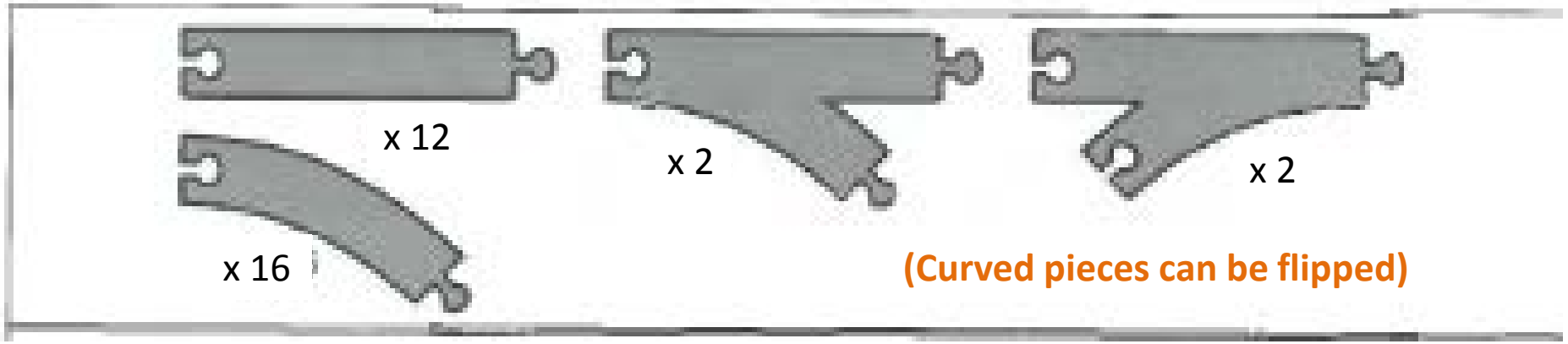
"Machines will be capable, within twenty years, of doing any work that a man can do." *Herbert Simon, 1965.*

Remember Reading Assignment!

- Next week: Chapter 5 (Adversarial Search)

Class Exercise: Wooden Railway Set

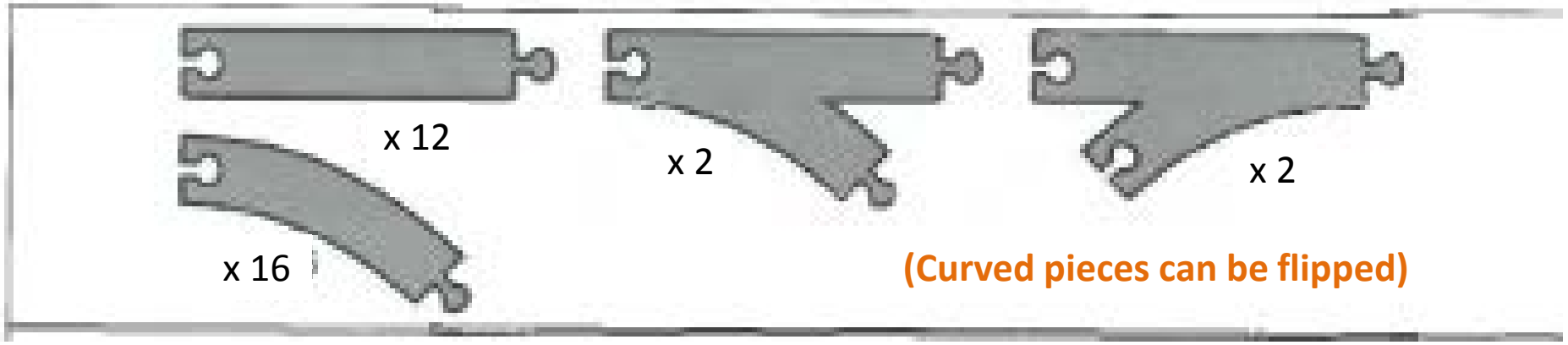
Track pieces from wooden railway set:



Q1: Suppose the pieces fit together exactly. Give formulation of the task as a search problem

Class Exercise: Wooden Railway Set (con't.)

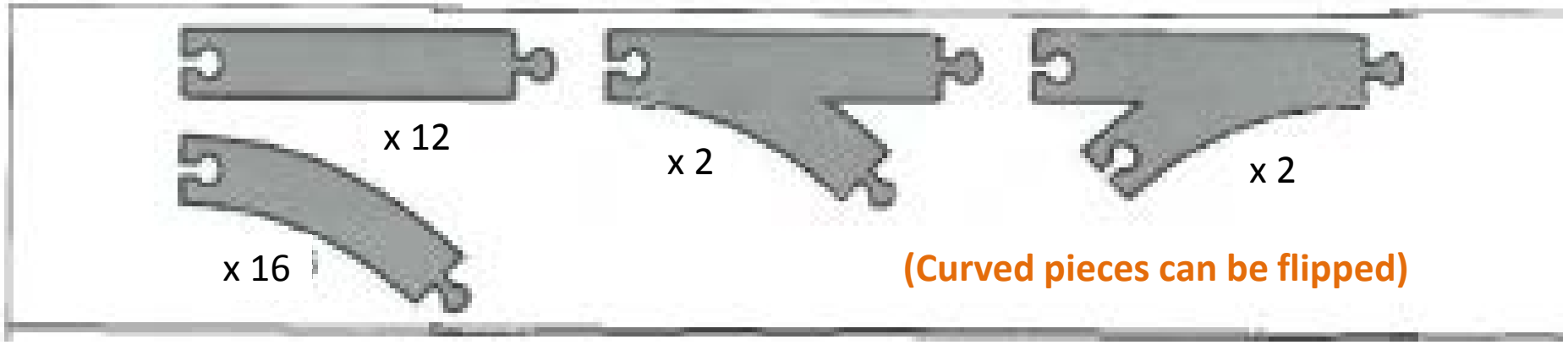
Track pieces from wooden railway set:



Q2: Identify a suitable uninformed search algorithm for this task, and explain why it is suitable.

Class Exercise: Wooden Railway Set (con't.)

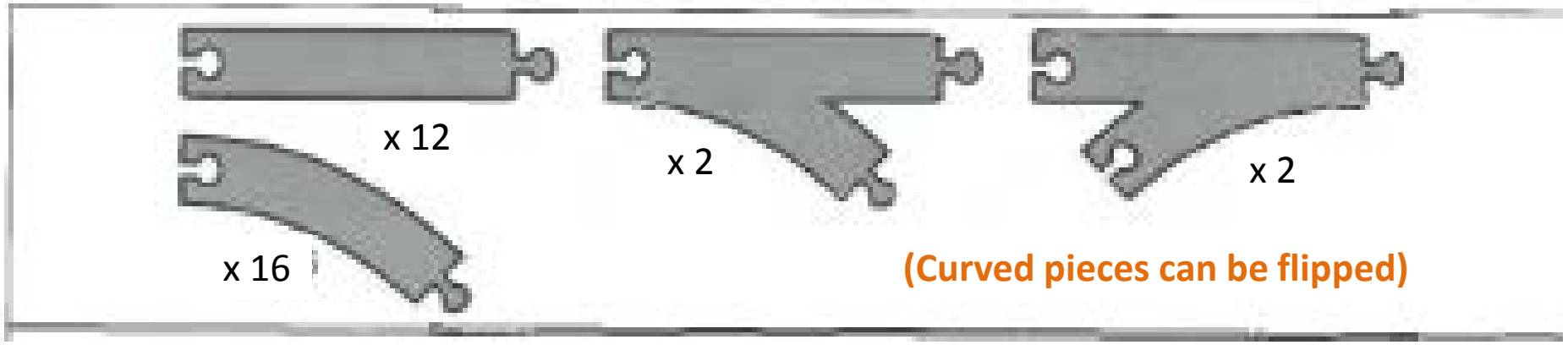
Track pieces from wooden railway set:



Q3: Why does removing any one of the "fork" pieces make the problem unsolvable?

Class Exercise: Wooden Railway Set (con't.)

Track pieces from wooden railway set:



Q4: Give an upper bound on the total size of the state space defined for this formulation. (Ignore problem of overlapping pieces and loose ends. Reason primarily about max branching factor and max depth. Pretend unique pieces.)

Searching with Nondeterministic Actions

- Can't be sure of the outcome of actions
  - Percepts help narrow down possible resultant states
- Need a contingency plan (or strategy) that specifies what to do depending on what percepts are received

Erratic Vacuum World

- Outcome of *Suck* action:
  - When applied to dirty square the action cleans the square and sometimes cleans up dirt in an adjacent square, too
  - When applied to a clean square the action sometimes deposits dirt on the carpet

- To handle, need transition model that returns a set of possible outcome states

- And, need to generalize solution to a contingency plan
  - E.g., [*Suck*, **if** *State = 5* **then** [*Right, Suck*] **else** []]

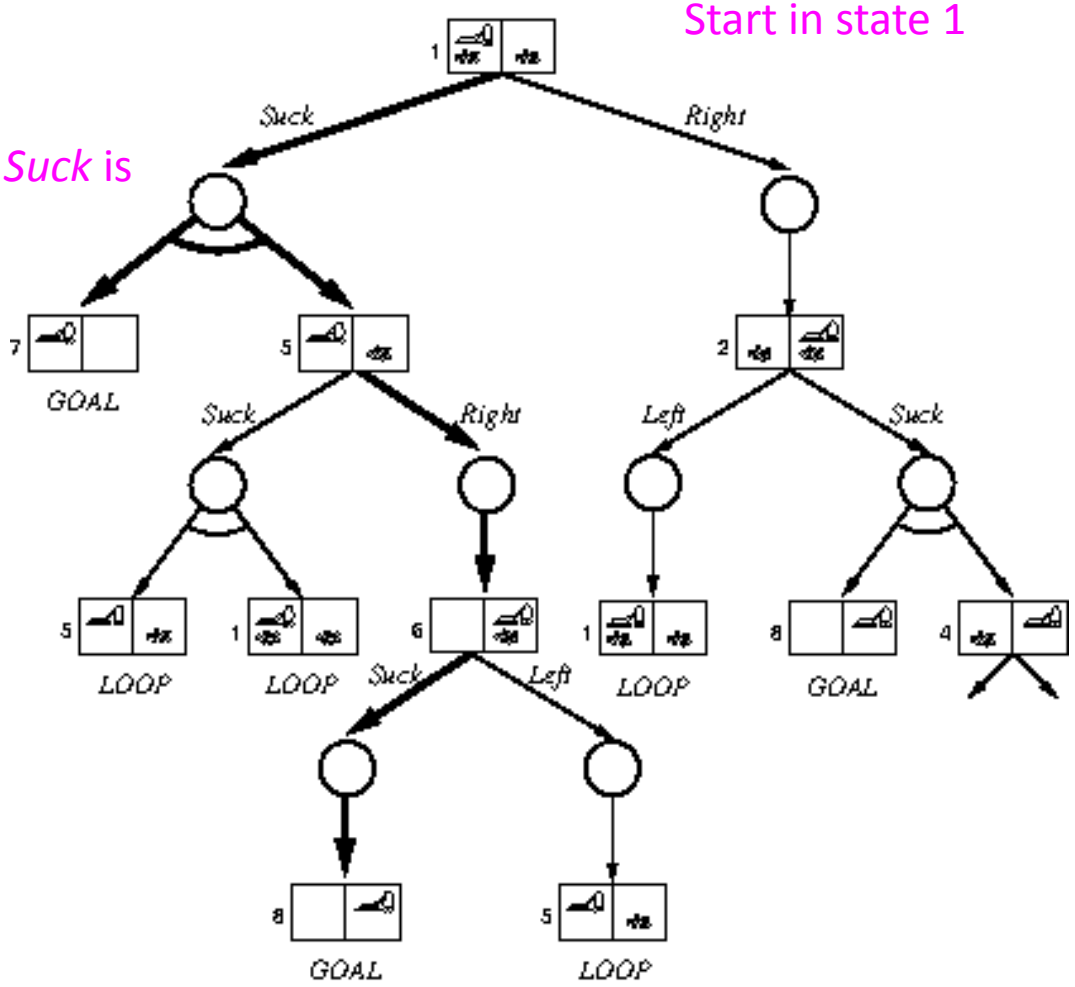
New type of Search Tree: AND-OR Search Trees

- OR nodes: Represent agent's own choices
- AND nodes: Environment's choice of outcome for each action

Start in state 1

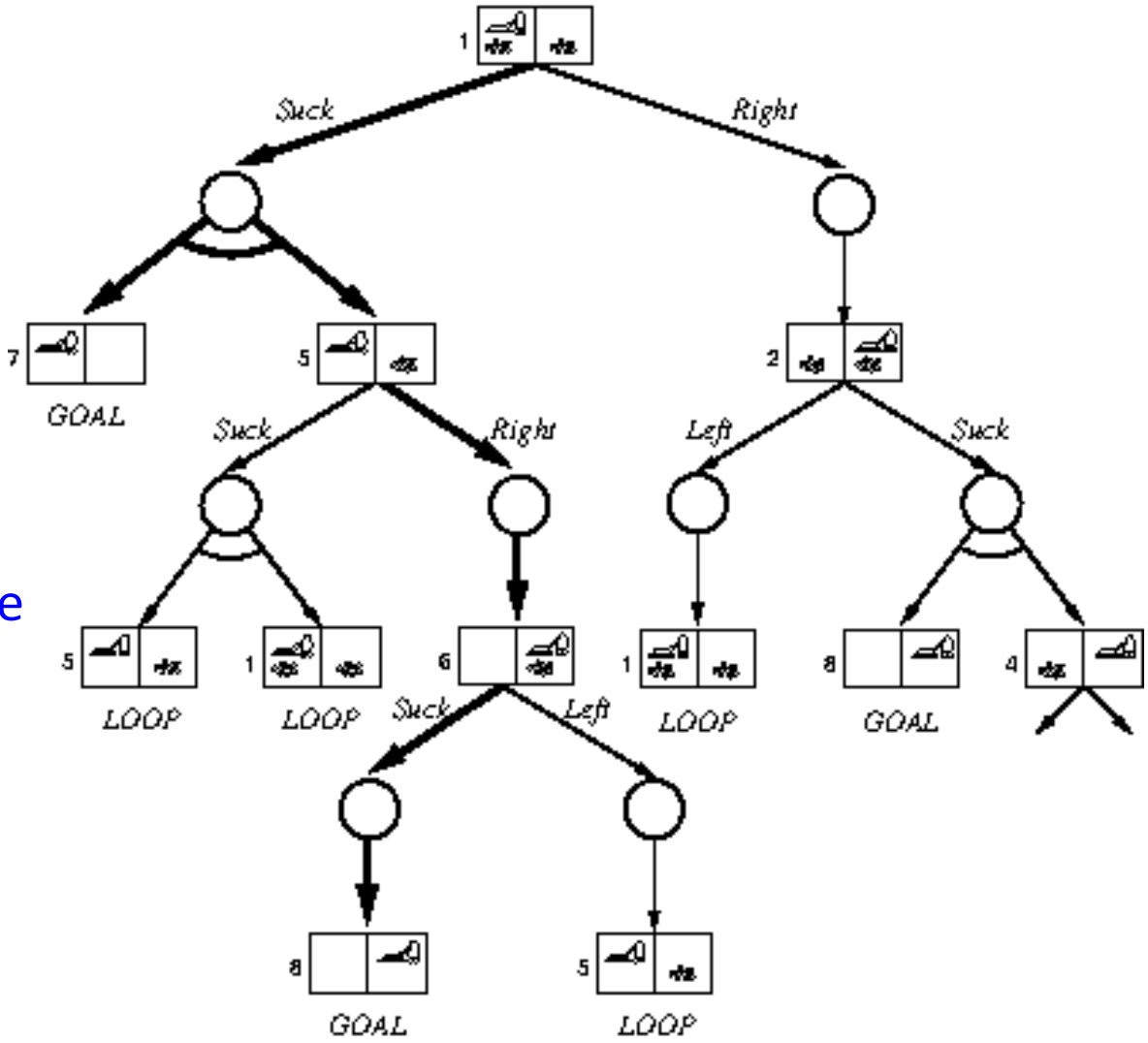
Uncertain result of *Suck* is {State 5, State 7}

Need a plan for all possible outcome states



New type of Search Tree: AND-OR Search Trees

- Solution is a subtree that:
  - Has goal node at every leaf
  - Specifies one action at each OR node
  - Includes every outcome branch at each AND node

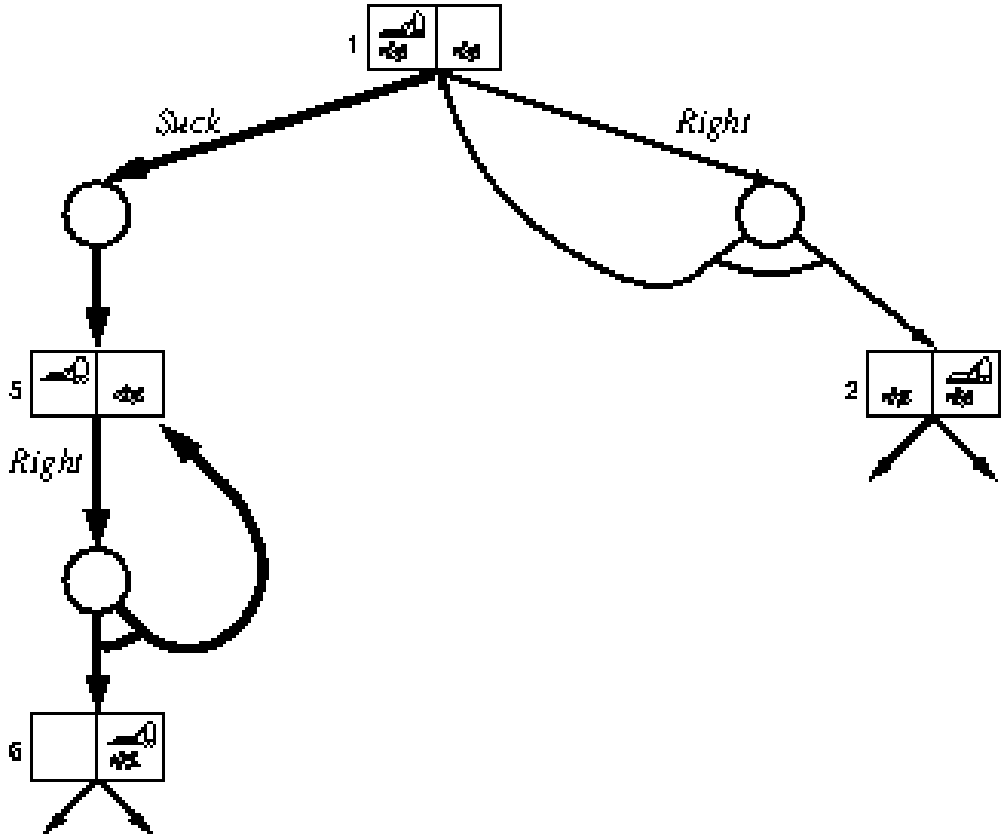


Searching AND-OR Graphs

- Can be done with a variety of search techniques:
  - DFS
  - BFS
  - Best-first
  - A*
  - Etc.

Slippery (Non-Erratic) Vacuum World

- Outcome of *Left* or *Right* actions:
  - Agent might not move, even if given *Left* or *Right* motion commands
  - Keep trying an action until it works
  - We assume that each possible outcome of a nondeterministic action eventually occurs



- Need to define cyclic solutions
  - Keep trying an action until it works
  - E.g., [*Suck*, L_1: Right, **if** *State = 5* **then** L_1 *else Suck*]
  - Equivalent to: "**while** *State* = 5 **do** *Right*"

Searching with Partial Observations

- Introduce *belief state:* represents agent's current belief about possible physical states it could be in, given sequence of actions and percepts up to that point
- 3 scenarios:
  - Searching with no observation
  - Searching with observations
  - Solving partially observable problems

Searching with no observations: Sensorless

- Sensorless = conformant

- Example: Vacuum world with no sensors:
  - Agent knows geography of environment
  - Agent doesn't know its location or distribution of dirt
  - Initial state: {1, 2, 3, 4, 5, 6, 7, 8}
  - Action outcomes:
    - [*Right*]: possible successor states: {2, 4, 6, 8}
    - [*Right, Suck*]: {4, 8}
    - [*Right, Suck, Left Suck*]: {7}

Sensorless problems: Search in Space of Belief States

- Beliefs are fully observable

- Belief states: every possible set of physical states; N physical states ➔ 2^N belief states
- Initial state: Typically the set of all physical states
- Actions: Either the union or intersection of the legal actions for the current belief states
- Transition model: set of all possible states that could result from taking any of the actions in any of the belief states
- Goal test: all states in current belief set are goal states
- Path cost: (it depends. application-specific)

- Use any search technique we've discussed

Challenge: size of belief state

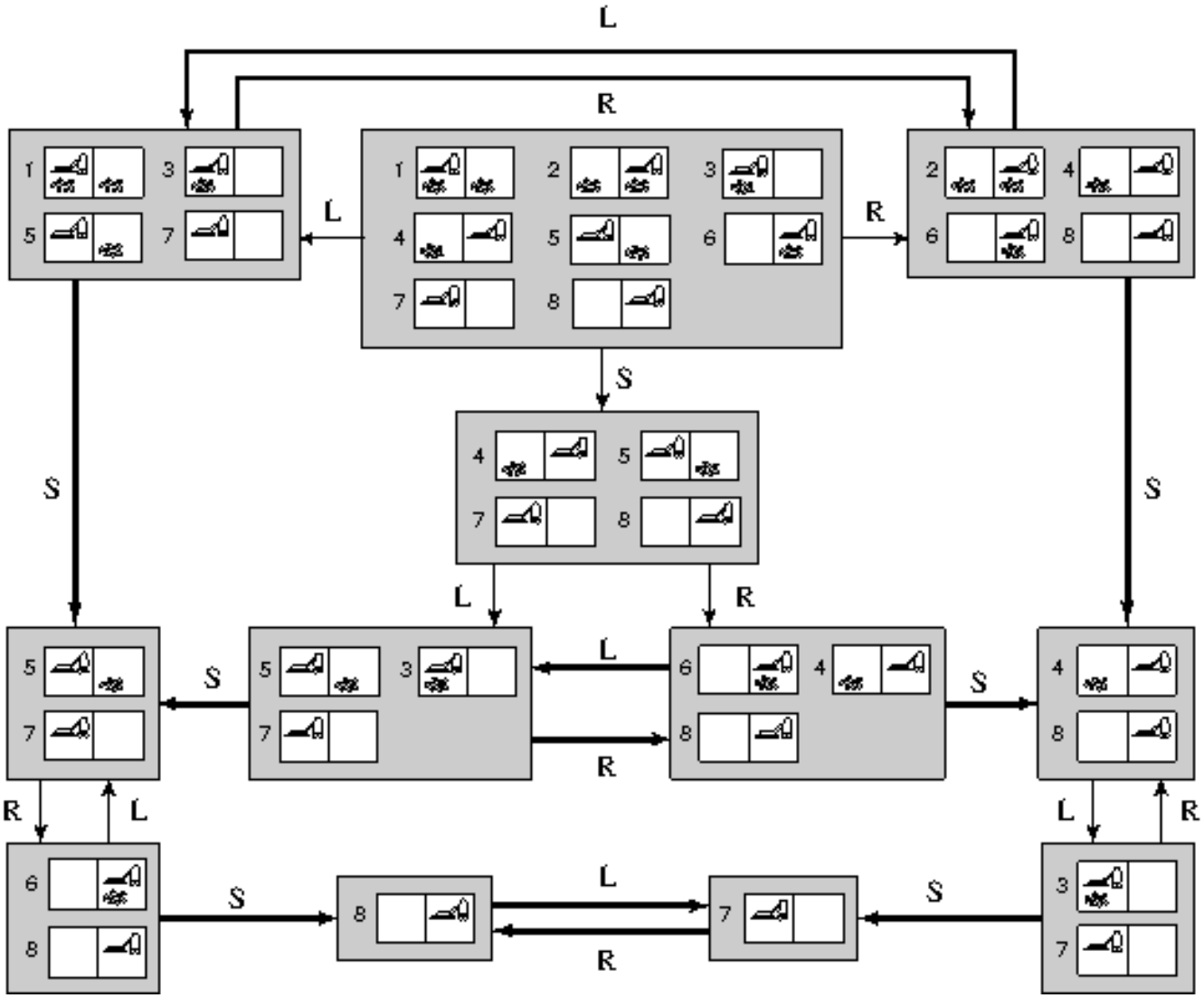
- Example: belief state for 10 x 10 vacuum world has 100 x 2^{100} = 1032 physical states!

- Alternatives:
  - Better (more compact) representation
  - Solving problem incrementally (incremental belief-state search)
    - E.g.,
      - solve for first state,
      - see if it works for other states;
      - if not, find another solution for first state,
      - and iterate

- But, generally tough to solve w/o sensors!

Belief-state space for deterministic, sensorless vacuum world

At any point, agent knows which belief state it is in, but not which physical state it is in



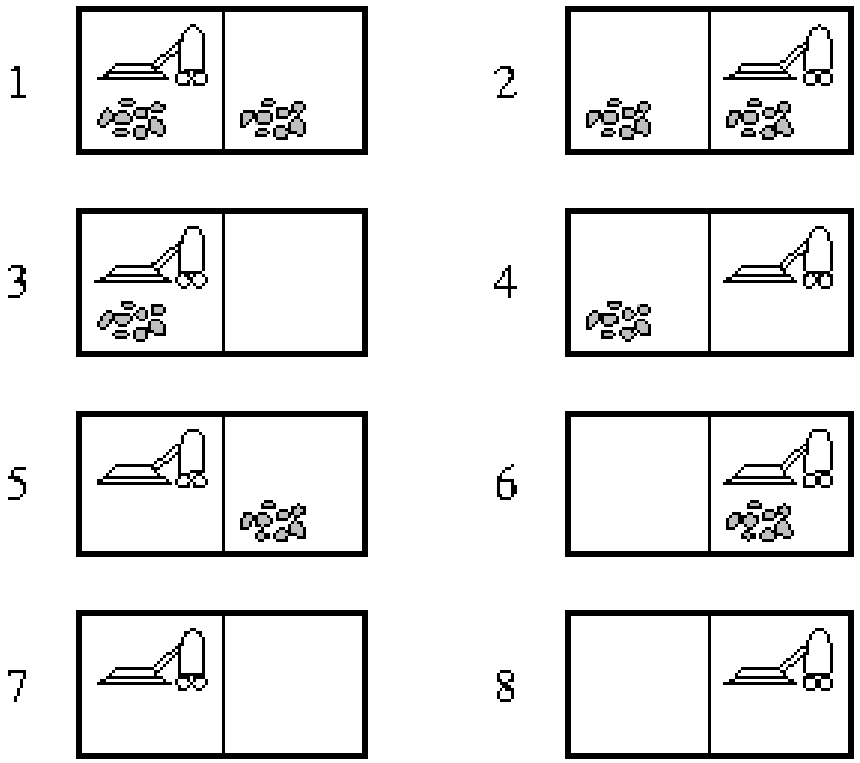
Note: there are $2^8 = 256$ possible belief states, but only 12 reachable belief states

Searching with observations

- Define PERCEPT(*s*) that returns the agent's percept, given the state *s*
- *E.g., In Vacuum world,* PERCEPT(state$_1$)=[*A,Dirty*]

- Special cases:
  - Fully observable problems: PERCEPT(*s*) = *s*
  - Sensorless problems: PERCEPT(*s*) = *Null*

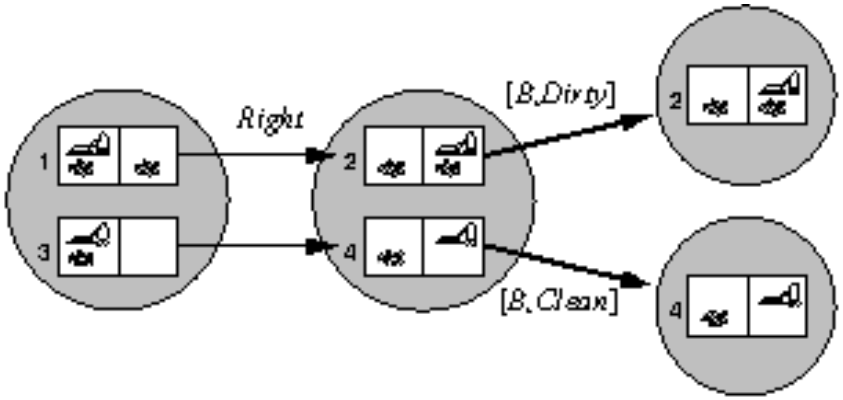
Vacuum World Examples



- PERCEPT = [*A,Dirty*] yields belief state {1, 3}
- PERCEPT = [*B,Clean*] yields belief state {4, 8}

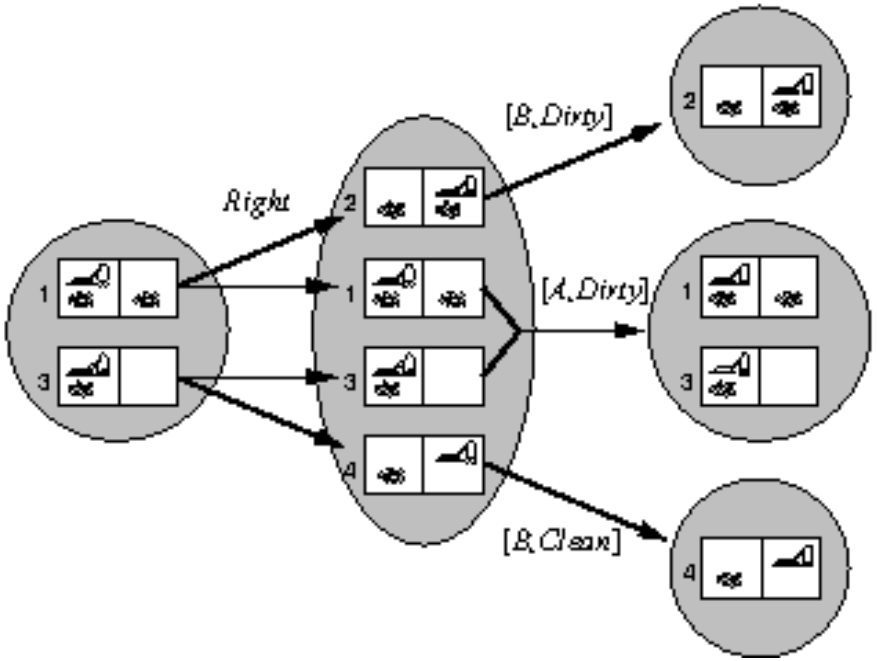
Example Transitions

Deterministic world:



Grey circles represent belief states

Slippery world:



Searching with observations

- Prediction: given action *a* in belief state *b*, predict the resulting belief state: $\hat{b}$ = Predict(b, a)
- Observation prediction: determine set of percepts *o* that could be observed in the predicted belief state:

  $\text{POSSIBLE\_PERCEPTS}(\hat{b}) = \{o: o = \text{PERCEPT}(s) \text{ and } s \in \hat{b}\}$
- Update: determine belief state that results from each possible percept (i.e., which set of states in $\hat{b}$ could have produced the percept)

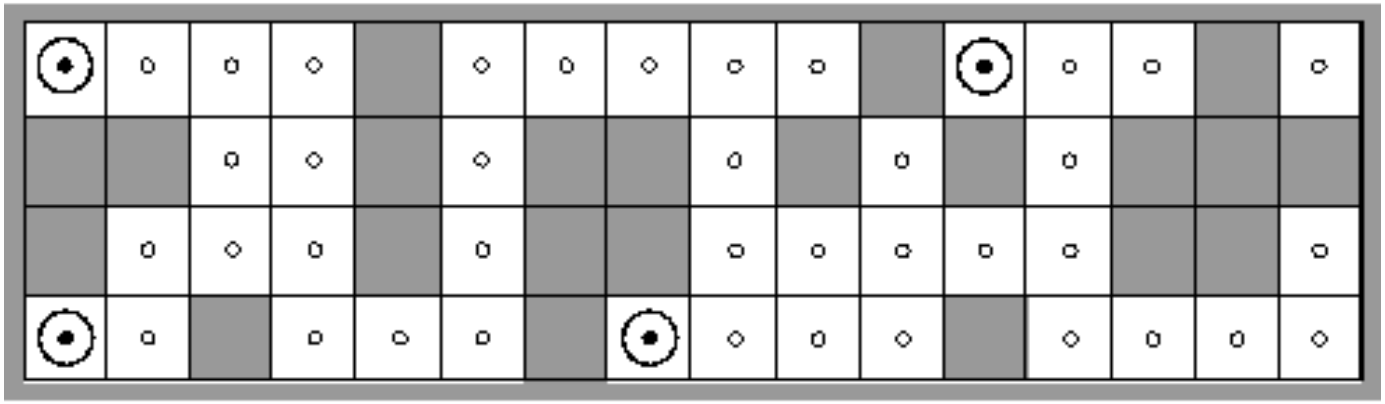
  $b_0 = \text{UPDATE}(\hat{b}, o) = \{s: o = \text{PERCEPT}(s) \text{ and } s \in \hat{b}\}$
- Then, obtain possible belief states resulting from action and subsequent possible percepts:

  $\text{RESULTS}(b, a)$
  $= \{b_0: b_0 = \text{UPDATE}(\text{PREDICT}(b, a), o) \text{ and } o \in \text{POSSIBLE-PERCEPTS}(\text{PREDICT}(b, a))\}$

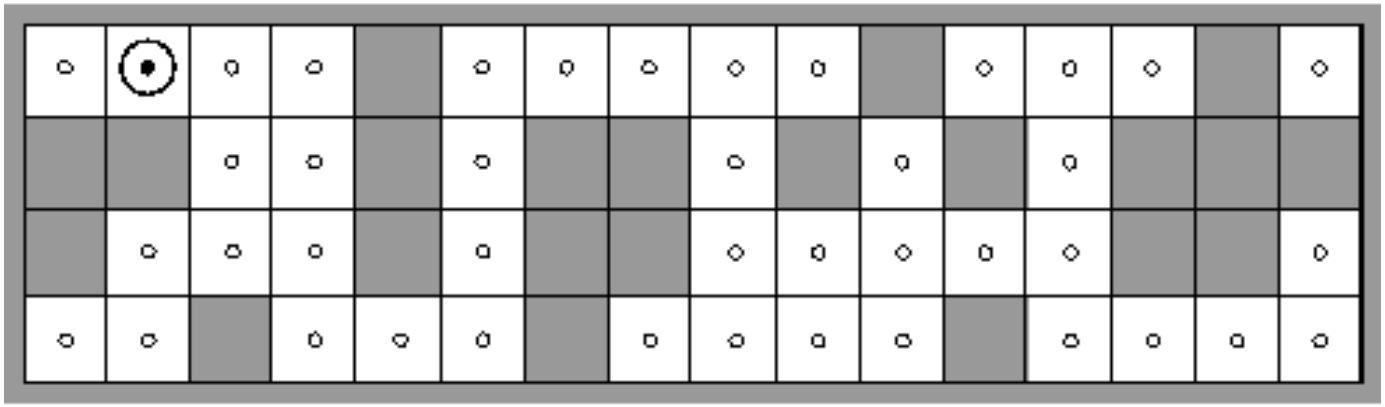
Can use AND-OR search algorithm to solve

More interesting application: Robot localization

- Initially, robot has no idea of where it is, but it knows geography of environment;
- Robot has perfect sensors to detect obstacles in each compass direction (NSEW)
- Robot has slippery motion – so it lands in any adjacent square after *Move*



(a) Possible locations of robot after E_1 = NSW



(b) Possible locations of robot After E_1 = NSW, E_2 = NS

Online search problems

- Until now: offline search, where complete solution is generated before anything changes in physical world

- Online search:
  - Interleaves computation and action
    - "Solved" by an agent executing actions
  - Useful for dynamic environments
  - Useful for nondeterministic environments, since it allows agent to focus on contingencies that actually arise – not just those that *might* arise
  - Necessary for unknown environments

Online search problems (con't.)

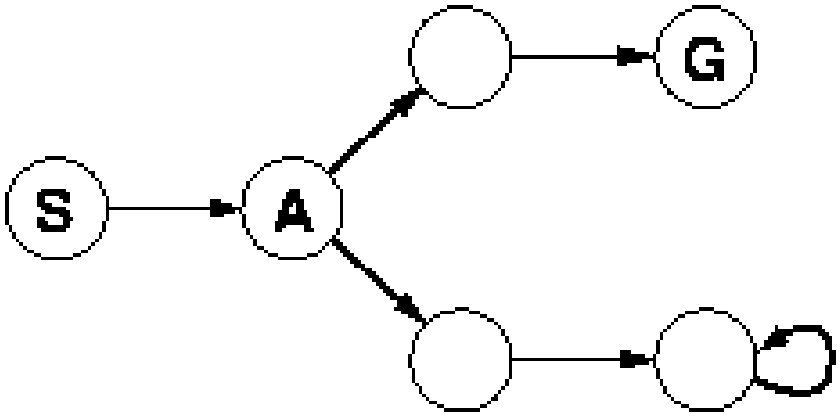
- Agent only knows:
  - Actions(s) – list of actions allowed in state *s*
  - Step-cost function c(s, a, s') – can only be determined after s' discovered
  - Goal-Test(s)

- Agent might know: admissible heuristic to determine distance to goal state

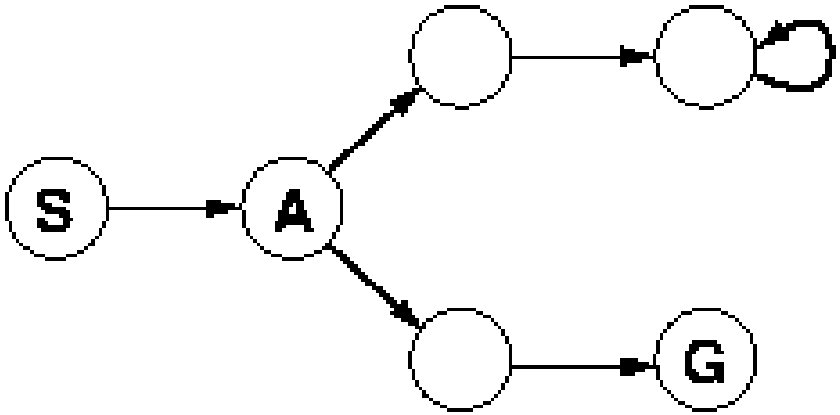
- Cost: total path cost of the path the agent actually travels
- Competitive ratio: ratio of actual cost to optimal cost (i.e., best case if the agent knew the search space in advance)
  - "1" is optimal actual cost

Online search problems (con't.)

- Irreversible actions: fall off cliff!
- Dead-end state: locked in a freezer!

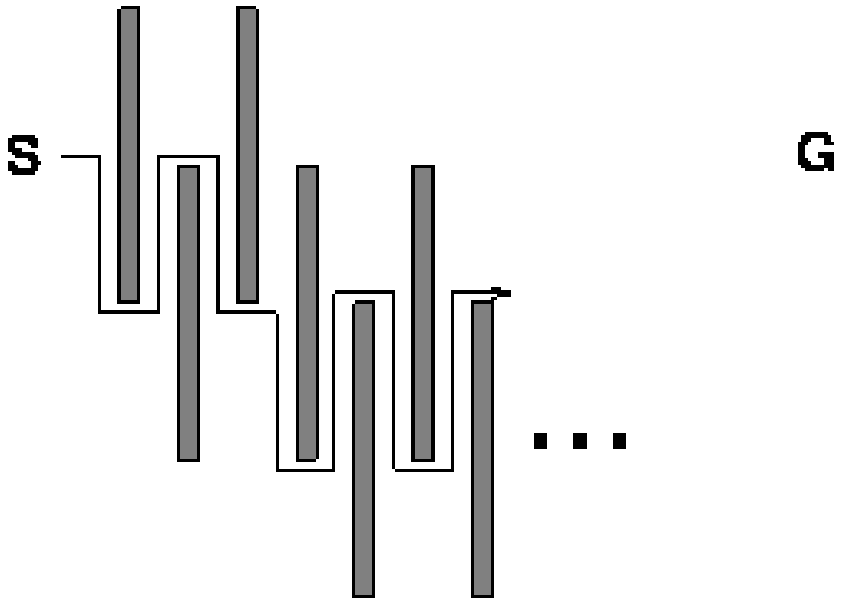


- No algorithm can avoid dead ends in all state spaces
- Easier: assume state space is safely explorable, where some goal is reachable from every state



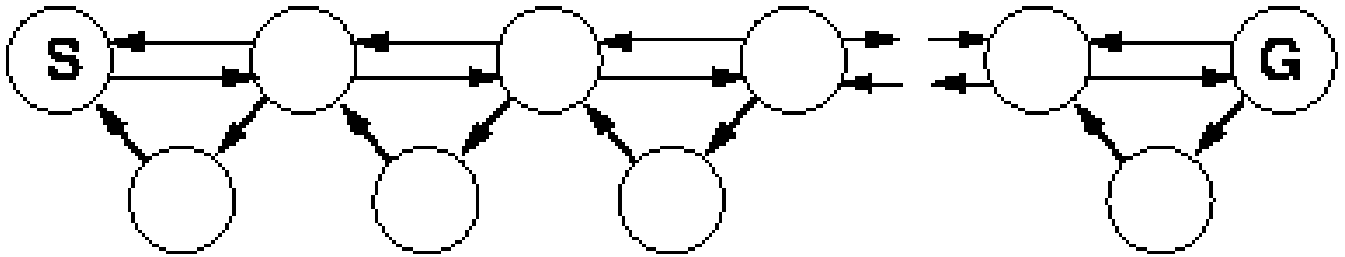
Online search problems (con't.)

- No bounded competitive ratio can be guaranteed, even in safely explorable environments



Online search problems (con't.)

- Agent can only explore from current state
  - Can't bounce around to other search paths, like offline A*
  - DFS can be used, as long as actions are reversible
  - Also online versions of iterative deepening will work
  - Random walk is possible
    - Select at random an available action from current state
    - Preference to previously unexplored actions
    - Will eventually find goal, if space is finite
    - But slow – exponential search in worst case:



Exercise: Sensorless heuristics

- A sequence of actions solves a sensorless problem if it maps every physical state in initial belief state *b* to a goal state.
- Suppose agent knows $h^*(s)$, the true optimal cost of solving the fully observable version of the problem, for every state *s* in *b*.
- What would be an admissible heuristic $h(b)$ for the sensorless problem, in terms of $h^*(s)$?

Remember Reading Assignment!

- Next week: Chapter 5 (Adversarial Search)