COMP 424 - Artificial Intelligence
Lecture 6: Searching under uncertainty

Instructor: Jackie CK Cheung (jcheung@cs.mcgill.ca)

Readings: R&N Ch 4.3, 4.4

Based on slides by Joelle Pineau

Quick recap

- Uninformed and informed search.
- Search for optimization problems.
- Search for constraint satisfaction problems.
  - Discrete (vs continuous) state space.
  - Deterministic (vs stochastic) environment.
  - Observable (vs unobservable) environment.
  - Static (vs changing) environment.
- Search for many real-world domains:
  - Non-deterministic actions.
  - Partial observations.

Today

Sources of uncertainty

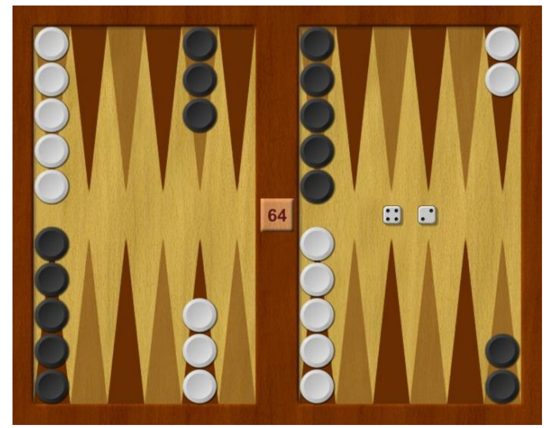
Two major issues:

- Agent may not know what the outcomes of its actions are!
  - In this case, the actions are **non-deterministic**.
- Agent may not be able to observe which state it is in!
  - We can classify problems as **observable**, **partially observable**, or **non-observable**.

Searching with non-deterministic actions

- Consider the case where the next state is not fully determined by the current state and action.

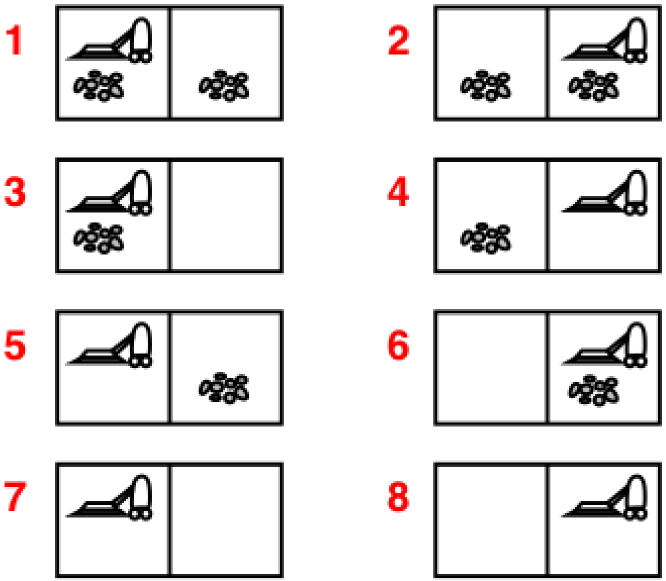
  e.g. Play backgammon



- Finite set of states and actions.
- Each **action** can have **many outcomes** depending on outcome of dice roll.
- Future states cannot be determined in advance.
- Solution to this problem is not a path, but a **contingency plan** (or **strategy**).

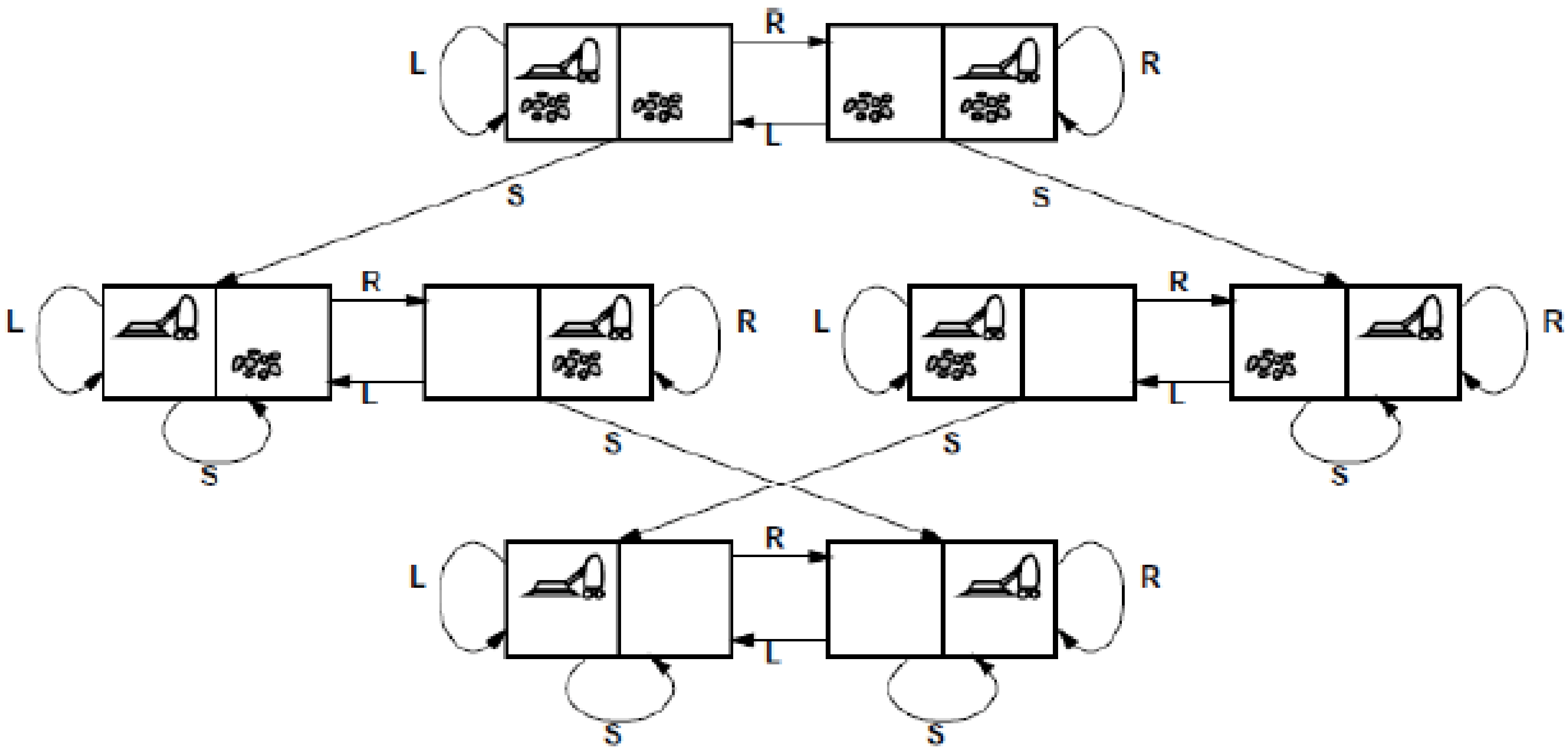
The vacuum world

1. **Observable case:**
   - Start in a given state: {5}
   - Plan: {Right; Sweep}



Possible actions:
{Left, Right, Sweep}
Goal states: {7, 8}

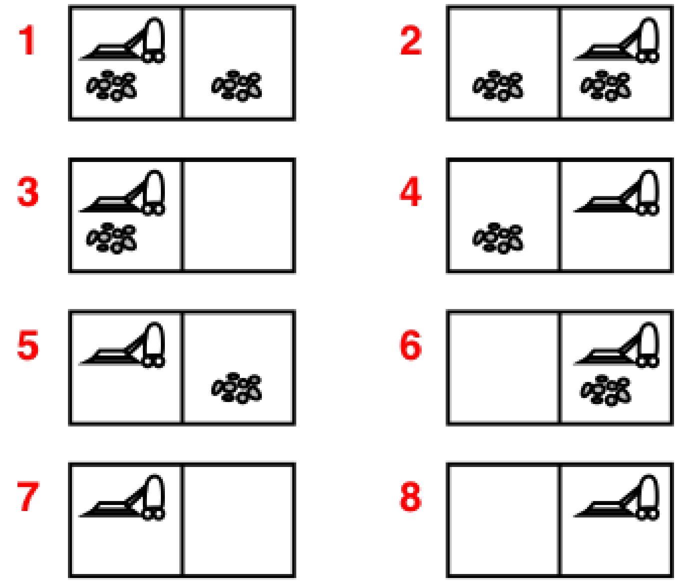
Observable vacuum: state space



The vacuum world

1. **Observable case:**
   - Start in a given state: {5}
   - Plan: {Right; Sweep}

2. **Non-observable case**:
   - Start in any of: {1,2,3,4,5,6,7,8}
   - Plan?

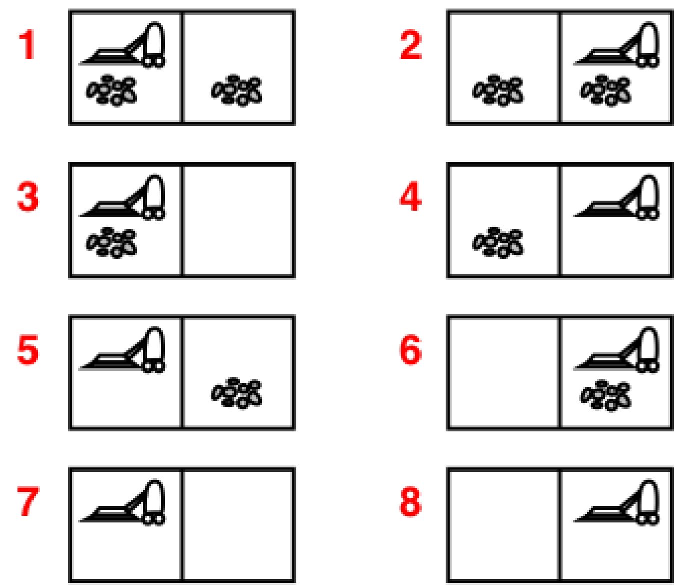


Possible actions:
{Left, Right, Sweep}
Goal states: {7, 8}

The vacuum world

1. **Observable case:**
    - Start in a given state: {5}
    - Plan: {Right; Sweep}

2. **Non-observable case**:
    - Start in any of: {1,2,3,4,5,6,7,8}
    - Plan: {Right; ?? }



What states are possible after Right action?

Need to reason over **sets of states** → **Beliefs**

Possible actions:

{Left, Right, Sweep}

Goal states: {7, 8}

Searching with unobservable states

- Consider case where agent is in one of several possible states, but cannot observe which.

- Represent uncertainty by a **belief state**
  - i.e., the agent's current belief about the possible states, given the sequence of actions and observations up to that point

- Total number of possible beliefs?

  $2^8 = 256$ (Technically 2^8-1, since the belief has to include at least 1 state.)

Searching with unobservable states

- Consider case where agent is in one of several possible states, but cannot observe which.

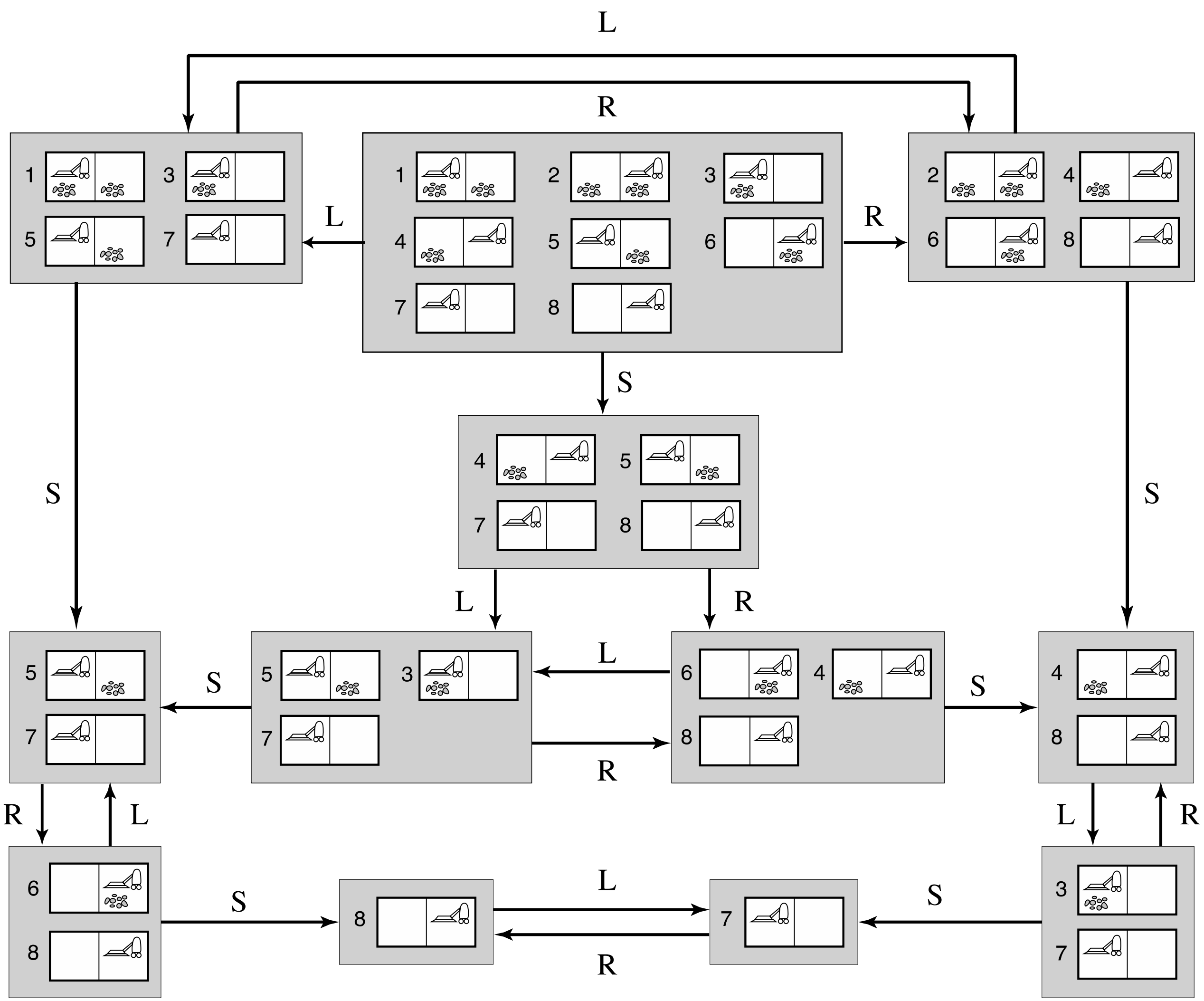
- Represent uncertainty by a **belief state**
  - i.e., the agent's current belief about the possible states, given the sequence of actions and observations up to that point

- Total number of possible beliefs?

  $2^8 = 256$ (Technically 2^8-1, since the belief has to include at least 1 state.)

- Number of **reachable** beliefs?

Non-observable vacuum: belief space



Searching with unobservable states

- Consider case where agent is in one of several possible states, but cannot observe which.

- Represent uncertainty by a **belief state**
  - i.e., the agent's current belief about the possible states, given the sequence of actions and observations up to that point

- Total number of possible beliefs?

  2^8 = 256 (Technically 2^8-1, since the belief has to include at least 1 state.)

- Number of **reachable** beliefs? Only 12 reachable beliefs.

Searching with unobservable states

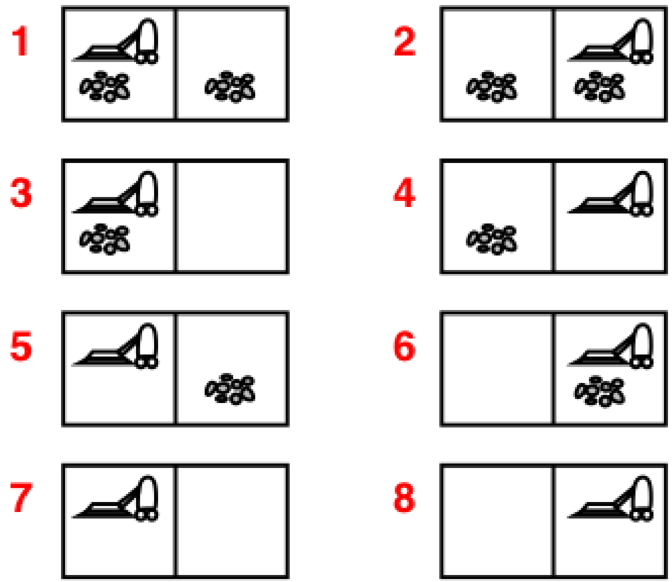
- Assume (for now) that there are no observations and actions are deterministic.
- Start in any state:

  belief = {1, 2, 3, 4, 5, 6, 7, 8}

- How to use actions?

  {1, 2, 3, 4, 5, 6, 7, 8} $\rightarrow_{\text{Right}}$ {2, 4, 6, 8}

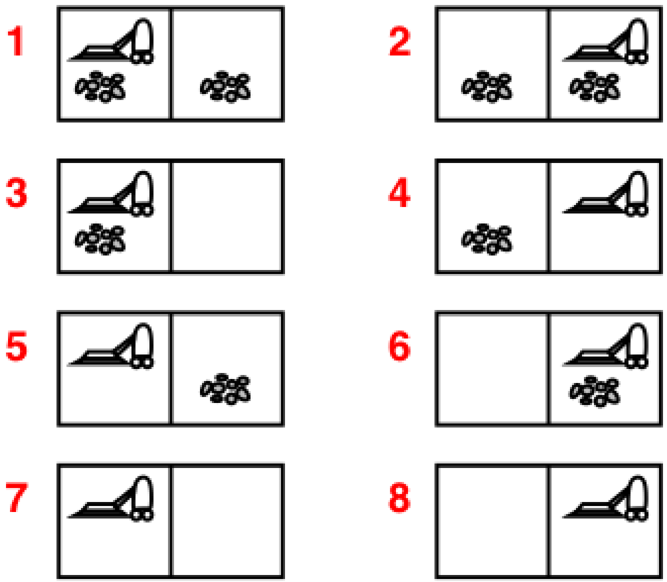
- Plan?



Searching with unobservable states

- Assume (for now) that there are no observations and actions are deterministic.
- Start in any state:

  belief = {1, 2, 3, 4, 5, 6, 7, 8}



- How to use actions?

  {1, 2, 3, 4, 5, 6, 7, 8} $\rightarrow_{\text{Right}}$ {2, 4, 6, 8}

- Plan? Use a **conformant plan**

  {1, 2, 3, 4, 5, 6, 7, 8} $\rightarrow_{\text{Right}}$ {2, 4, 6, 8} $\rightarrow_{\text{Sweep}}$ {4, 8} $\rightarrow_{\text{Left}}$ {3,7} $\rightarrow_{\text{Sweep}}$ {7}

- Other possible plans?

Conformant planning

- Goal is to find a plan that leads to the goal despite state uncertainty.

- Good heuristic is to use actions that reduce uncertainty → reduce belief to a single state.

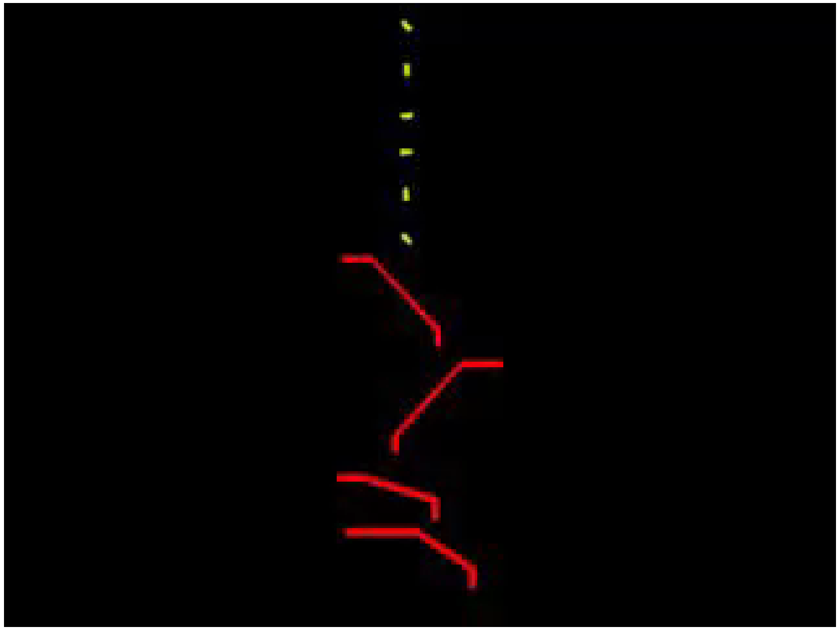
- Once you know what state you are in, apply standard search to reach the goal.

- Use conformant planning when there are NO observations.

Visual analogy

- Feeding parts for manufacturing:

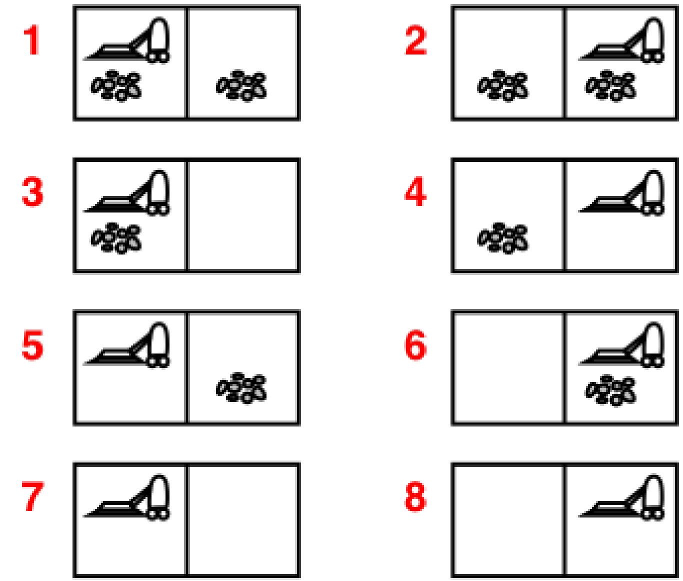
  http://goldberg.berkeley.edu/fences.mpg



The vacuum world

1. **Observable case:**
   - Start in a given state: {5}
   - Plan: {Right; Sweep}

2. **Non-observable case**:
   - Start in any of: {1,2,3,4,5,6,7,8}
   - Plan: {Right; Sweep; Left; Sweep}

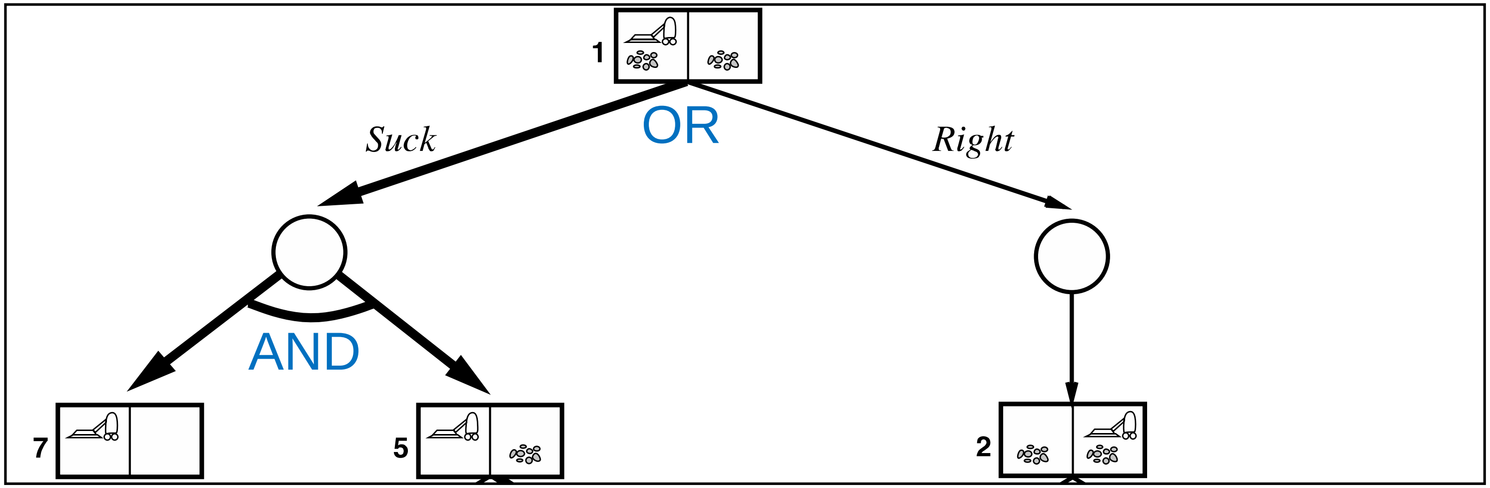


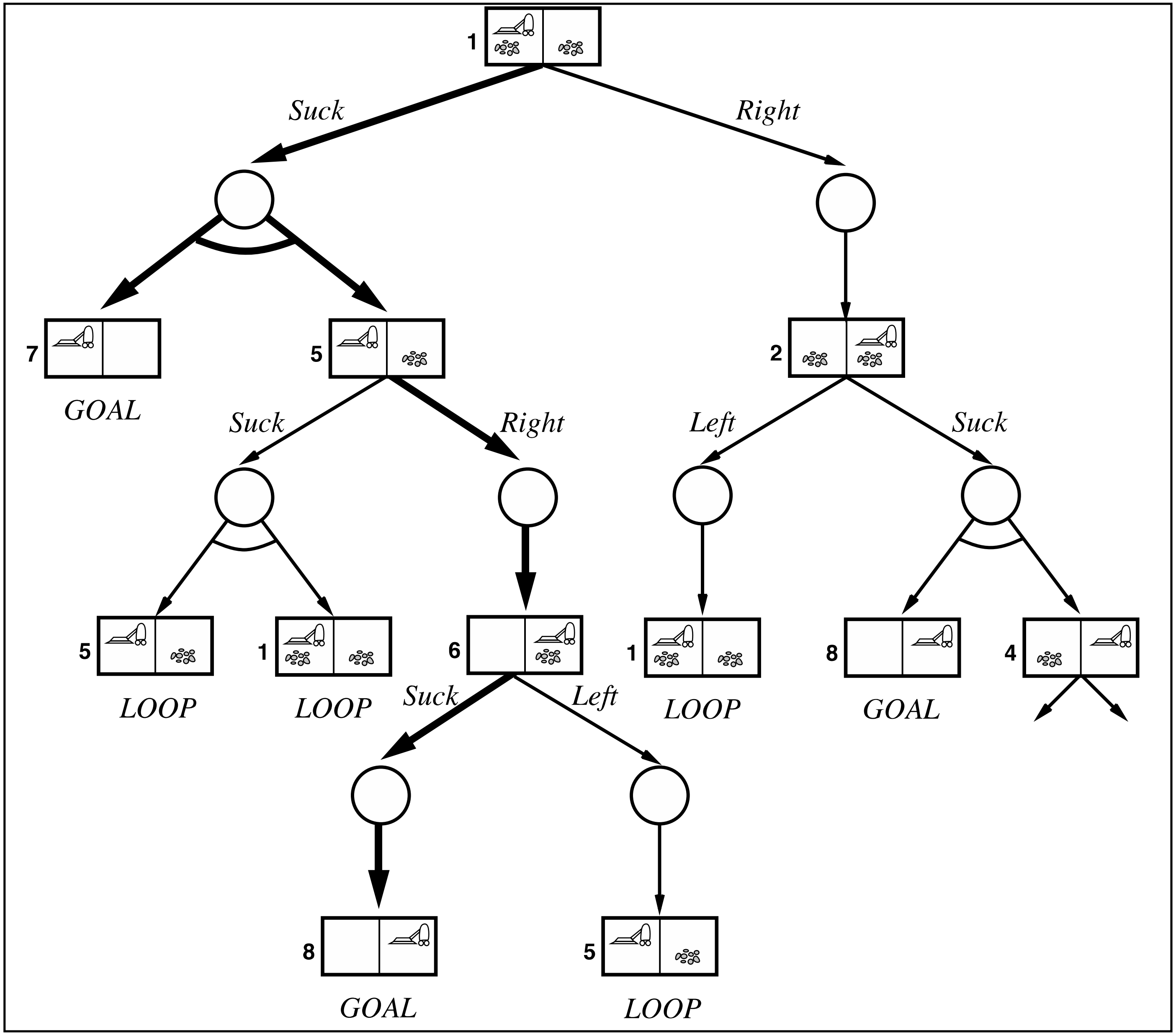
3. **Non-deterministic case**:
   - Start in a given state: {1}
   - Sweep clean square sometimes deposits dirt; Sweep dirty square sometimes cleans adjacent square.
   - Plan?

Non-deterministic vacuum: AND-OR search

AND-OR search tree:

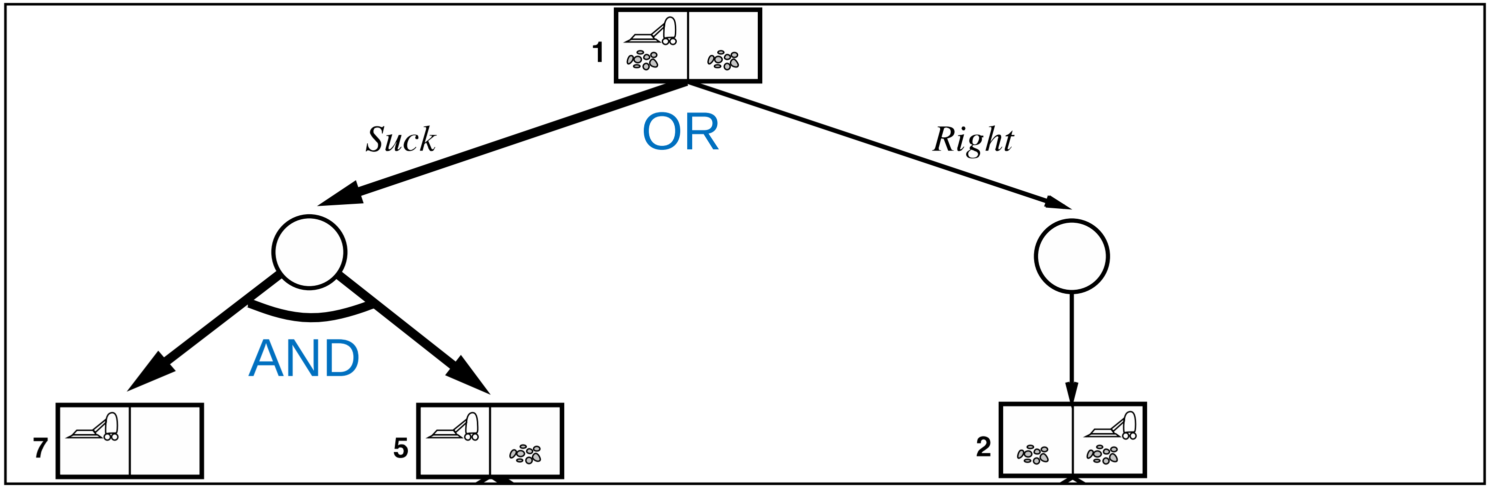
- OR nodes: the agent chooses between actions.
- AND nodes: choice is induced by the environment's choice of outcome.

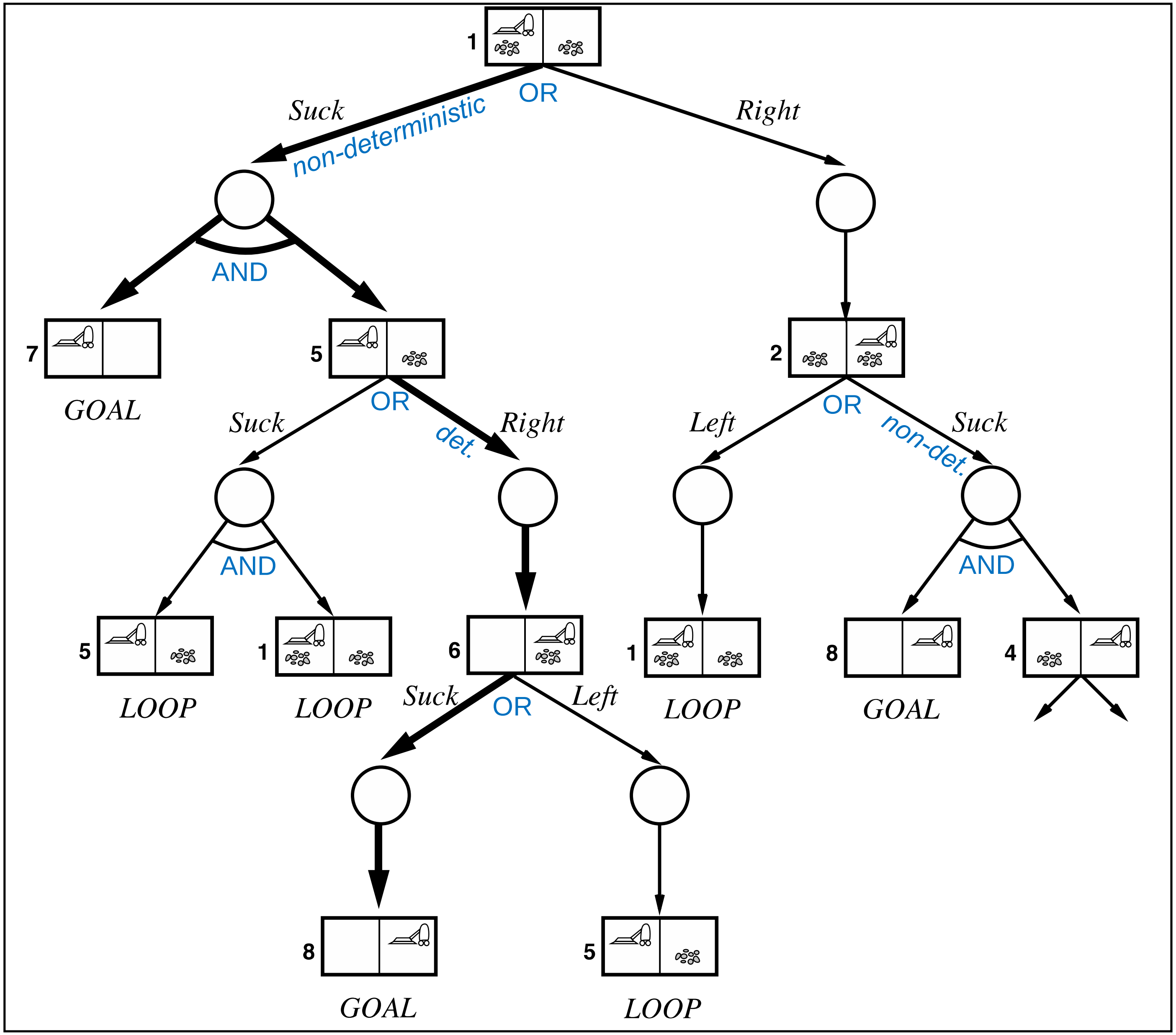


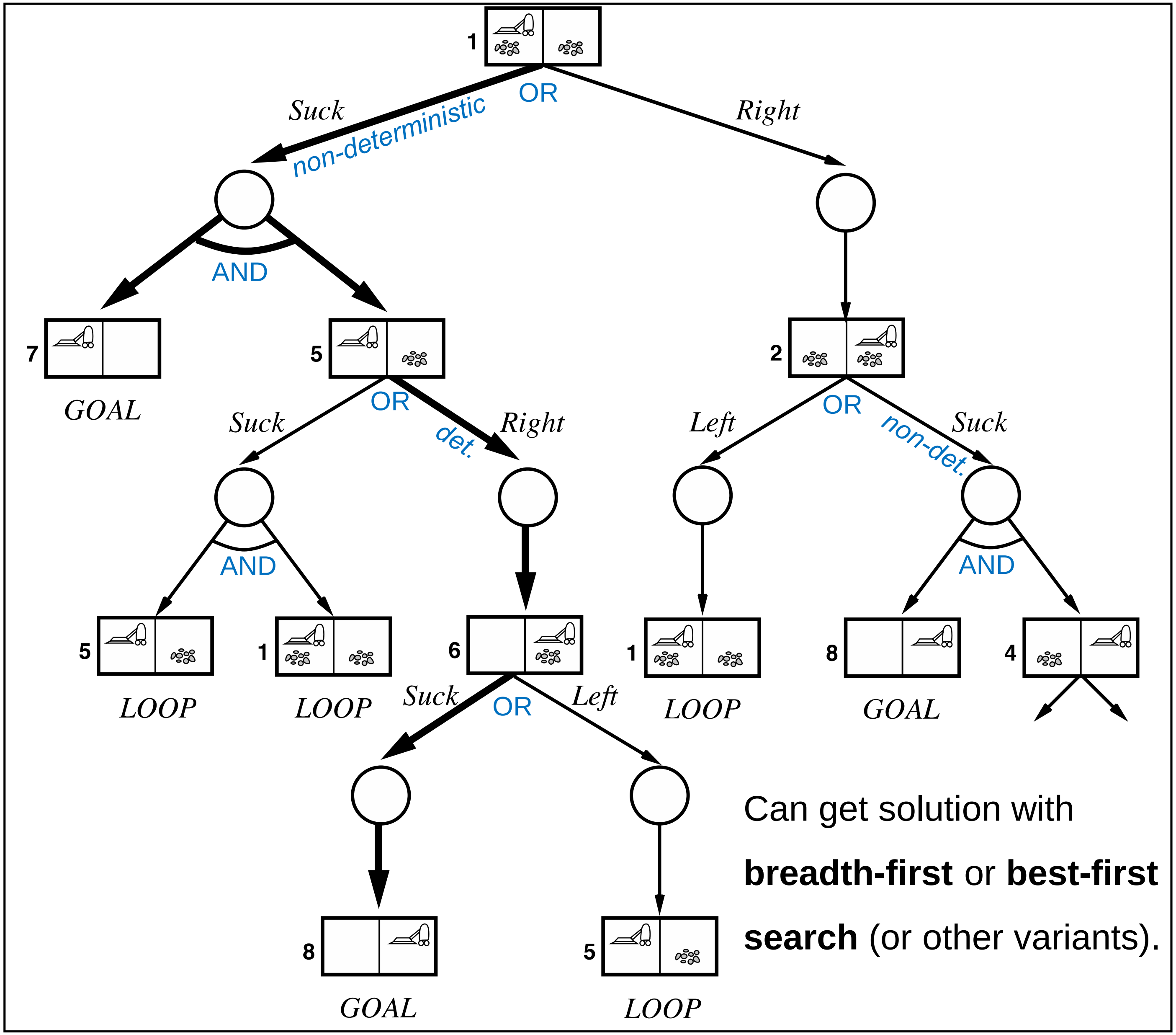

1
Suck
Right
7
GOAL
5
Suck
Right
2
Left
Suck
5
LOOP
1
LOOP
6
Suck
Left
1
LOOP
8
GOAL
4
8
GOAL
5
LOOP

Non-deterministic vacuum: AND-OR search

- Solution for AND-OR search problem is a **subtree** that:
  - specifies **one action** at each **OR** node;
  - includes **every outcome** at each **AND** node;
  - **has a goal node at every leaf**.



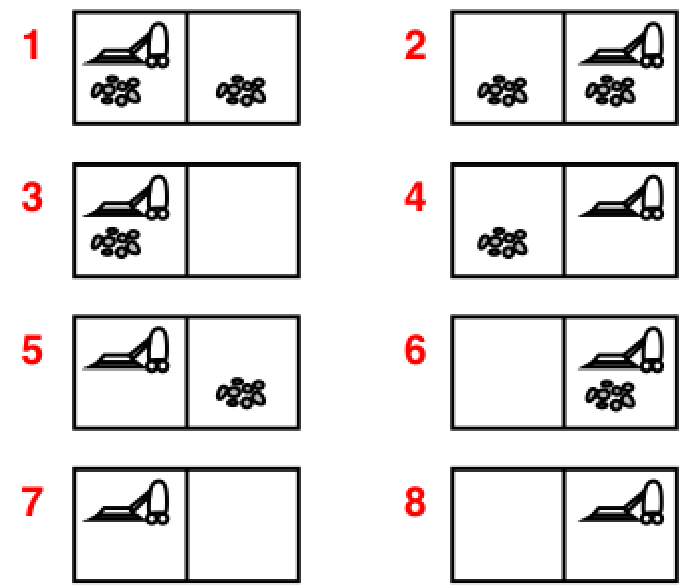

1
Suck
non-deterministic
OR
Right
AND
7
GOAL
5
Suck
OR
det.
Right
2
Left
OR
non-det.
Suck
AND
AND
5
LOOP
1
LOOP
6
Suck
OR
Left
1
LOOP
8
GOAL
4
8
GOAL
5
LOOP



Can get solution with **breadth-first** or **best-first search** (or other variants).

The vacuum world

1. **Observable case:**
   - Start in a given state: {5}
   - Plan: {Right; Sweep}



2. **Non-observable case**:
   - Start in any of: {1,2,3,4,5,6,7,8}
   - Plan: {Right; Sweep; Left; Sweep}

3. **Non-deterministic case**:
   - Start in a given state: {1}
   - Sweep clean square sometimes deposits dirt; Sweep dirty square sometimes cleans adjacent square.
   - Plan: {Sweep; Right; Sweep}

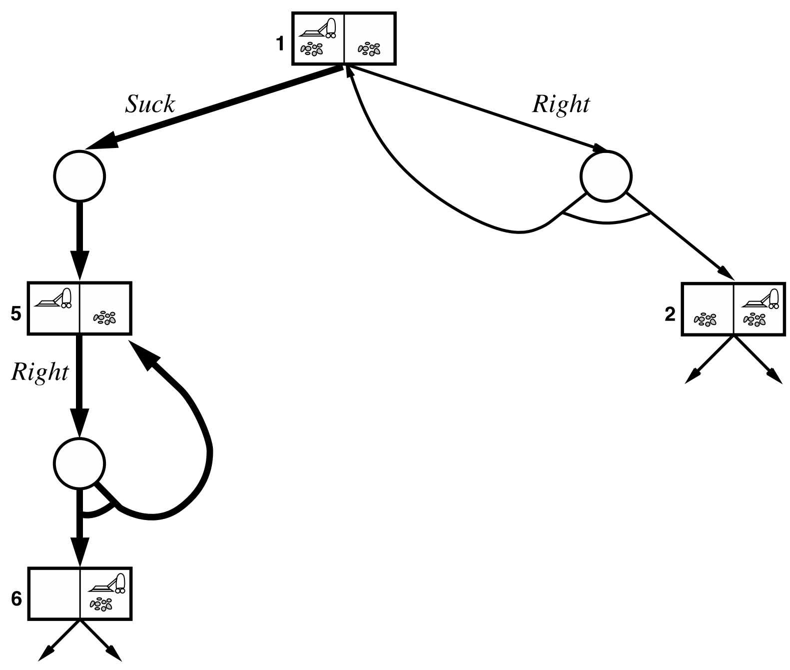
Another case: The slippery vacuum

- Movement actions, {Left, Right}, sometimes fail, leaving the agent in the same location. E.g. $\{1\} \rightarrow_{Right} \{1, 2\}$

Another case: The slippery vacuum

Movement actions, {Left, Right}, sometimes fail, leaving the agent in the same location.   E.g. {1} $\rightarrow_{\text{Right}}$ {1, 2}

Apply a **cyclic** solution: keep on trying an action until it works.



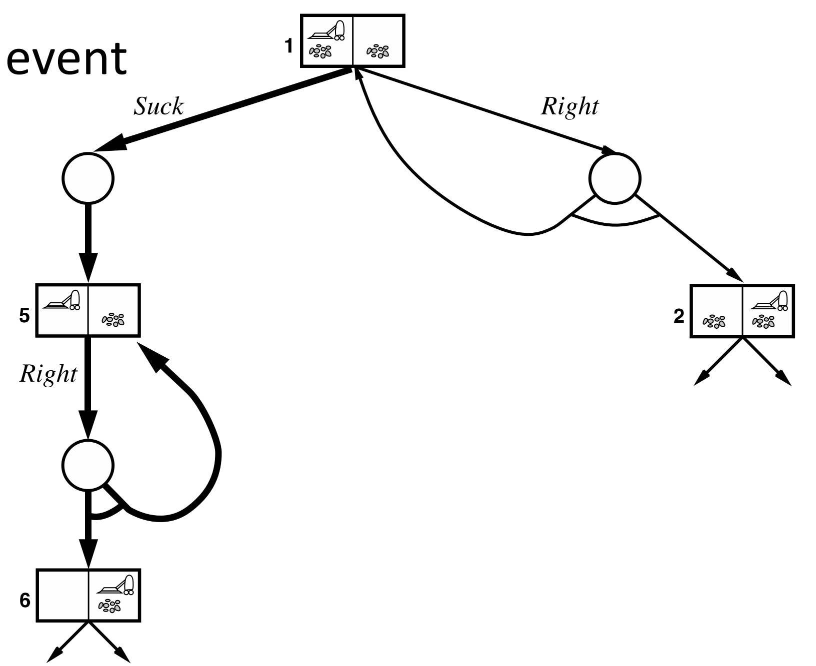
Keep trying until it works

Cyclic solution: keep on trying an action until it works.

Is this a good solution?

Depends on cause of the non-determinism:

- Ok if caused by random event
  - E.g. Die roll
- Not if caused by unobserved event
  - E.g. Broken vacuum

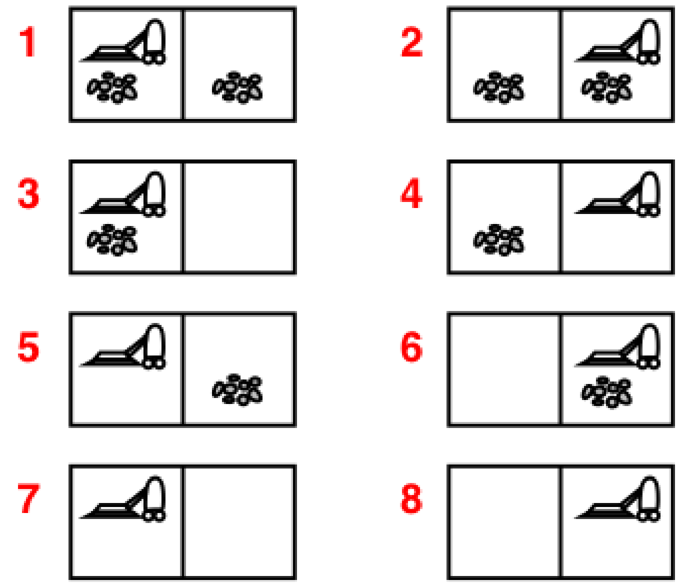


From unobservable to partially observable

- Need to add a **set of observations** and a **percept function**:
  - Defines how the environment generates percepts for the agent, as a function of the physical state
- Different classes of observability:
  1. Fully observable problems: *Percept(s) = s*
     - Actions can be deterministic or non-deterministic
  2. Non-observable problems: *Percept(s) = null*
  3. Partially observable problems: $Percept(s) = \{o_i\}$
     - o_i is a symbol providing (noisy) evidence about s
     - Recall the vacuum world with local sensing
  4. Part. Obs. with nondeterministic sensing: $Percept(s) = \{o_1, \ldots, o_k\}$
     - Can observe any o_i from a set of possible symbols $\{o_1, \ldots, o_k\}$

Local sensing in vacuum world

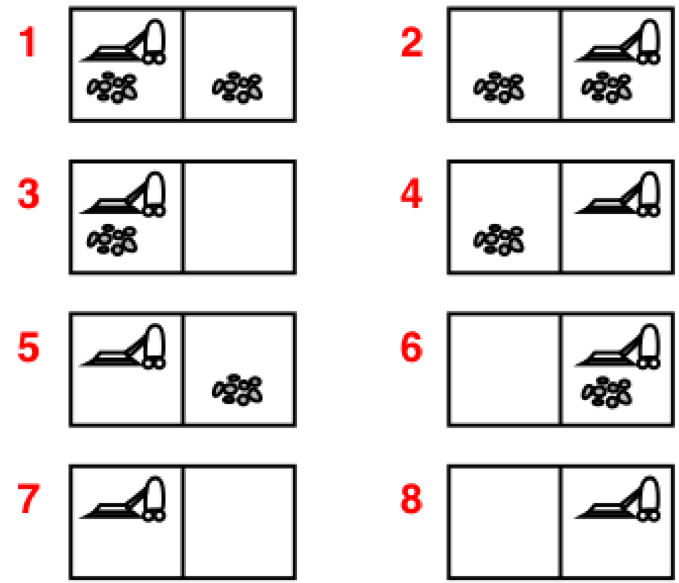
1. **Observable case:**
2. **Non-observable case**:
3. **Non-deterministic case**:



4. **Non-deterministic, partially observable case:**
   - Start in a given state: {1} <= Belief state
   - Sweep sometimes cleans adjacent square.
   - Local sensing is available: {dirty, clean} <= Sensing function
   - Plan?

Local sensing in vacuum world

1. **Observable case:**
2. **Non-observable case**:
3. **Non-deterministic case**:



4. **Non-deterministic, partially observable case:**
   - Start in a given state: {1} <= Belief state
   - Sweep sometimes cleans adjacent square.
   - Local sensing is available: {dirty, clean} <= Sensing function
   - Plan? $\{1\} \rightarrow_{\text{Sweep}} \{5,7\} \rightarrow_{\text{Right}} \{6,8\} \rightarrow_{\text{If dirty, then Sweep}} \{8\}$

Search in belief space

- Assume the original (fully observable problem) is defined by:

  $P = \{ States_P, InitialState_P, Actions_P, Predict_P, GoalTest_P, StepCost_P \}$

- Now define a new search problem:

  $B = \{ States_B, InitialState_B, Actions_B, Predict_B, GoalTest_B, StepCost_B \}$

- What do all of these represent?

Search in belief space

$B = \{ States_B, InitialState_B, Actions_B, Predict_B, GoalTest_B, StepCost_B \}$

$States_B$: every possible *set* of physical states, 2^N in total (N = size of $States_P$)

$InitialState_B$: usually every state in $States_P$, sometimes task-specific.

$Actions_B$: $Actions_B(b) = U_{s \text{ in } b}\ Actions_P(s)$

(This can be a problem if some actions are illegal in some states.)

$Predict_B$: b -> b'

Deterministic actions: b' = { s': s'=$Predict_P$(s,a) and s in b }

Nondeterministic actions: b' = { $U_{s \text{ in } b}$ $Predict_P$(s,a) }

$GoalTest_B$: Goal is satisfied when *all* the physical states in b satisfy the goal.

$StepCost_B$: $StepCost_P$ (If the cost of an action is the same in all states.)

Searching with observations

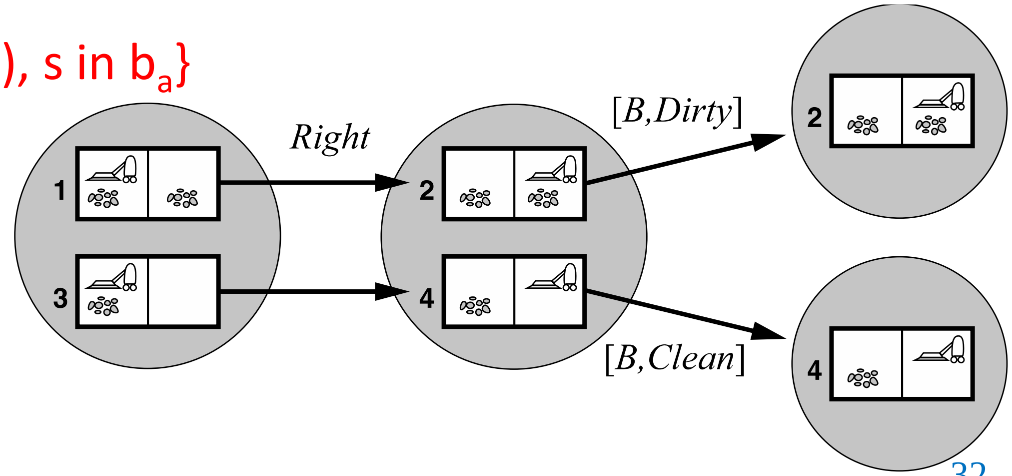
- Need to account for possible observations, and what observations tell us about the next state!
- State prediction stage:

  b_a = Predict(b, a)
- Observation prediction stage:

  Possible-Percept(b_a) = {o : o=Percept(s), s in b_a}
- Belief update stage:

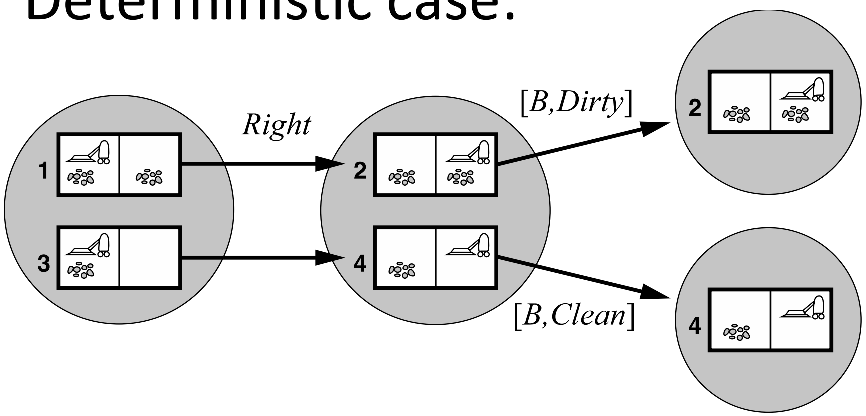
  b_{ao} = Update(b_a, o)

  = {s : o = Percept(s), s in b_a}

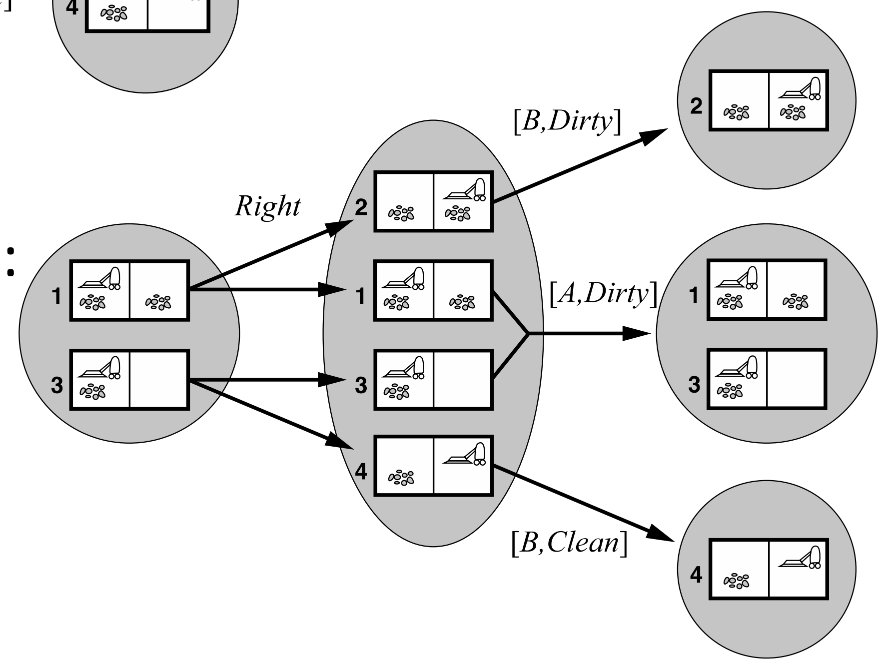


Searching with observations

- Deterministic case:



- Non-deterministic case:



Searching with observations

- State prediction stage:

  b_a = Predict(b, a)

- Observation prediction stage:

  Possible-Percept(b_a) = {o : o=Percept(s), s in b_a}

- Belief update stage:

  b_{ao} = Update(b_a, o)

  = {s : o = Percept(s), s in b_a}

- Putting it all together:

  b' = {b_{ao} : b_{ao} = Update(Predict(b,a),o), o in Possible-Percept(Predict(b,a))}

Use this procedure to build the nodes in the search tree.

Search properties

- Optimality / Completeness:
  - Depends on the search algorithm used.
- Time / Space complexity?
  - Let *S = #states, A=#actions, O=#observations, T=plan length*

Search properties

- Optimality / Completeness:
  - Depends on the search algorithm used.

- Time / Space complexity?
  - Let *S = #states, A=#actions, O=#observations, T=plan length*
  - Worst-case, search tree has $O((AO)^T)$ *nodes*
  - Each search tree has at most 2^S *nodes*
  - Each node has at most S states in its belief

BUT: The number of <u>reachable</u> nodes often depends on the number of states, and the number of possible observations in each state.

(Recall, for our vacuum world: 2^8 possible beliefs, only 12 are reachable.)

Exercise: Mastermind

http://www.web-games-online.com/mastermind/index.php

- Belief space?
- Initial belief?
- Action space?
- Deterministic / non-deterministic actions?
- Possible percepts?
- Perception function?
- Goal test?
- Step cost?

Searching the belief space

- In general, can apply any standard search algorithm.
- Scalability is an issue!
- **Problem #1**: Number of reachable beliefs can be very large.
  - Use sampling or pruning to reduce this.
- **Problem #2**: Number of states in each belief can be very large.
  - Use a compact state representation.
  - Plan for each state separately (not as a search over belief space) to quickly find out if a state in the belief (and therefore the belief state itself) is unsolvable.

Searching the belief space

- In general, can apply any standard search algorithm.
- Scalability is an issue!
- **Problem #1**: Number of reachable beliefs can be very large.
  - Use sampling or pruning to reduce this.
- **Problem #2**: Number of states in each belief can be very large.
  - Use a compact state representation.
  - Plan for each state separately (not as a search over belief space) to quickly find out if a state in the belief (and therefore the belief state itself) is unsolvable.

To be solved in future classes…

Summary

- Actions can be {**deterministic**, **nondeterministic**}
- Observations {**fully observable**, **unobservable**, **partially observable (deterministic)**, **partially observable (non-deterministic)**}
- **Belief states** represent the set of possible states an agent is in.
- AND-OR trees can be used to represent and search through a belief state space to produce a plan even when there is uncertainty.