2.1 STOCHASTIC GAMES

A stochastic game is a collection of normal-form games that the agents play repeatedly .

The particular game played at any time depends probabilistically on,

- the previous game played
- the actions of the agents in that game

A stochastic (or Markov) game includes the following:

- a finite set Q of states (games),
- a set N = {1, …, n} of agents,
- For each agent i, a finite set Ai of possible actions
- A transition probability function P : Q × A1 ×· · ·× An × Q → [0, 1] P(q, a1, …, an , q'') = probability of transitioning to state q'' if the action profile (a1, …, an) is used in state q
- For each agent i, a real-valued payoff function ri : Q × A1 ×· · ·× An → ℜ

This definition makes the inessential but simplifying assumption that each agent‟s strategy space is the same in all games. So the games differ only in their payoff functions. Wewill only consider strategies called **Markov strategies**. A strategy is call a **Markov strategy** if the behaviour dictated is not time dependent.

2.2 SEARCHING WITH PARTIAL OBSERVATIONS

Above we (unrealistically) assumed that the environment is fully observable and deterministic. Moreover, we assumed that the agent knows what the effects of each action are. Therefore, the agent can calculate exactly which state results from any sequence of actions and always knows which state it is in. Its percepts provide no new information after each action. In a more realistic situation the agent‟s knowledge of states and actions is incomplete.

If the agent has no sensors at all, then as far as it knows it could be in one of several possible initial states, and each action might therefore lead to one of several possible successor states. An agent without sensors, thus, must reason about sets of states that it mightget to, rather than single states. At each instant the agent has a belief of in which state it might be.

If the environment is partially observable or if the effects of actions are uncertain, then each new action yields new information. Every possible contingency that might arise during execution need considering. The cause of uncertainty may be another agent, an adversary.

When the states of the environment and actions are uncertain, the agent has to explore its environment to gather information. In a partially observable world one cannot determine a fixed action sequence in advance, but needs to condition actions on future percepts. As the agent can gather new knowledge through its actions, it is often not useful to plan for each possible situation. Rather, it is better to interleave search and execution.

Now we come back to a world where the actions of the robot are deterministic again (no erratic behavior like before) but, the robot no longer has complete sense of its current state or its environment.

<u>Vacuum World with no observation</u>

In this world, the vacuum cleaner has no idea initially about its own location and the location of dirt in the world. Since the robot has no percept, it should be able to figure out a sequence of actions that will work despite its current state.

Given below are 8 random initial states. You can record a sequence of actions and see it in action just like before. Assume that illegal moves (like moving right in the right-most tile) have no effect on the world. Try to find a sequence of actions that will lead to a final state (Clean all the dirt), no matter what the initial state of the world. Different types of incompleteness lead to three distinct problem types:

Sensorless problems (conformant): If the agent has no sensors at all

Contingency problem: if the environment if partially observable or if action are uncertain (adversarial)

Exploration problems: When the states and actions of the environment are unknown.

- ✓ No sensor
- ✓ Initial State(1,2,3,4,5,6,7,8)
- ✓ After action [Right] the state (2,4,6,8)
- ✓ After action [Suck] the state (4, 8)
- ✓ After action [Left] the state (3,7)
- ✓ After action [Suck] the state (8)
- ✓ Answer : [Right, Suck, Left, Suck] coerce the world into state 7 without any sensor
- ✓ Belief State: Such state that agent belief to be there

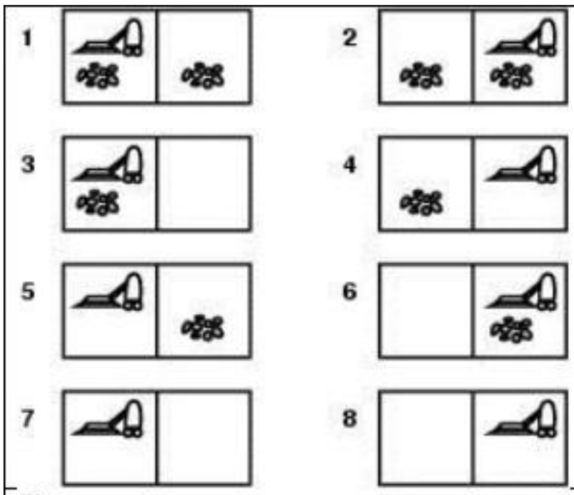
Partial knowledge of states and actions:

Sensorless or conformant problem

Agent may have no idea where it is; solution (if any) is a sequence.

Contingency problem **-** Percepts provide new information about current state; solution is a tree or policy; often interleave search and execution. If uncertainty is caused by actions of another agent: adversarial problem

Exploration problem- When states and actions of the environment are unknown.



Figure

Contingency, start in {1,3}.

By Murphy‟s law, Suck can dirty a clean carpet.

Local sensing: dirt, location only.

Percept = [L,Dirty] ={1,3}

[Suck] = {5,7} – [Right] ={6,8}

[Suck] in {6}={8} (Success) – BUT [Suck] in {8} = failure

Solution:

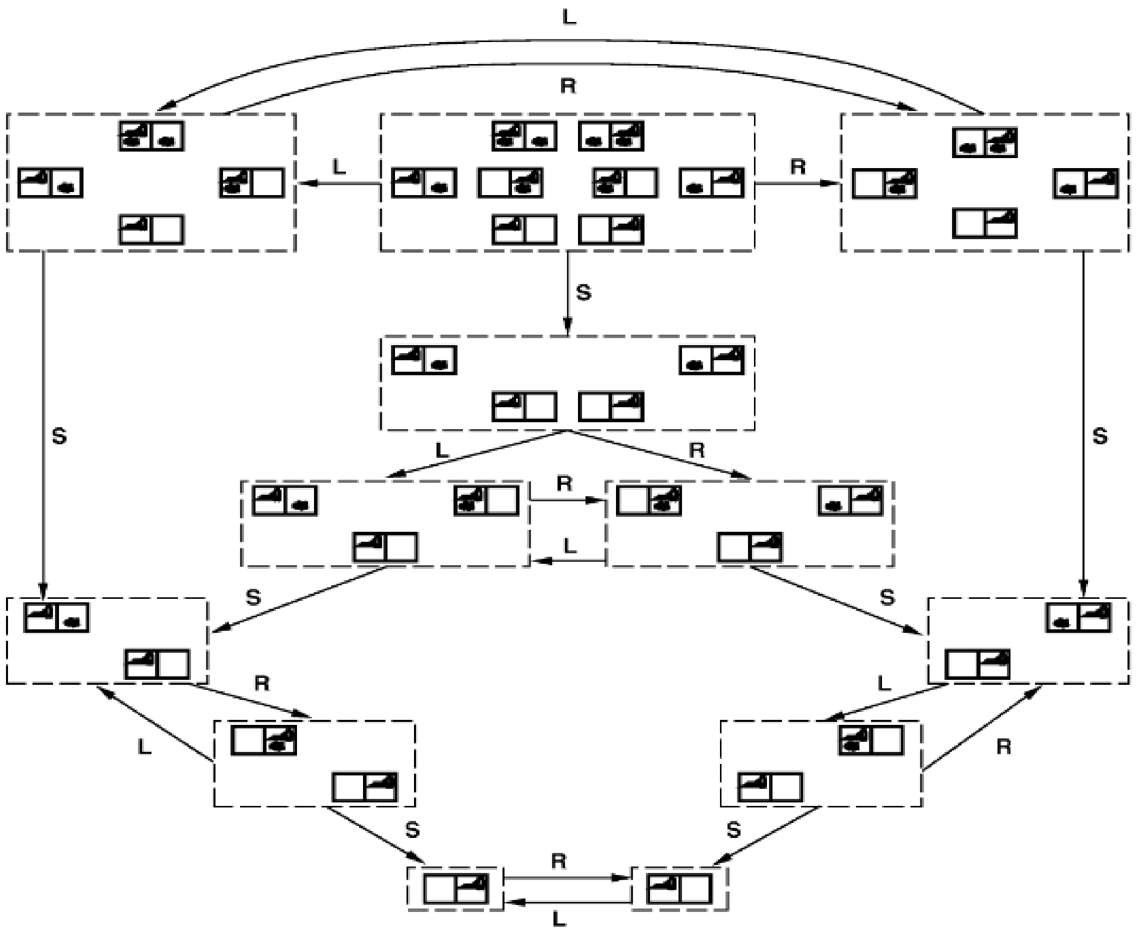
– Belief-state: no fixed action sequence guarantees solution

Relax requirement:

– [Suck, Right, if [R,dirty] then Suck]

– Select actions based on contingencies arising during execution.

Time and space complexity are always considered with respect to some measure of the problem difficulty. In theoretical computer science, the typical measure is the size of the state space.


L
R
L
R
S
S
S
L
R
R
L
S
S
R
L
L
R
S
S
R
L