Week 14: Multi-Agent decision making

ReadingList

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/3cbd32d7-e32b-43b5-a82b-5e06f645b7ec/OliehoekAmato16book.pdf

1. Different Dimensions in Multi-agent problems

1.1. Subjective or objective perspective

Subjective perspective:

focus on protagonist agent

objective perspective

focus on robot teams, that means **agent can take care of some aspects themselves**

1.2. On-line learning or off-line learning

online learning:

learning when executing

off-line learning:

learning without executing

1.3. what information is observed by each agent

2. Introduction

2.1. Uncertainty in Complex Systems

1. outcome uncertainty
2. partial observability
3. **uncertainty about others**

3. Fully Observable Conditions (Dec-MDPs)

3.1. Model:

need to consider **joint actions:** {(N,N,N), (N,N,W),...,(E,E,E)}
then the transition and rewards model should depend on joint actions

3.2. Subjective ways:

- Treat agents like 1 "puppeteer agent", that means a single team
- every agent executes its part
- members in the team share the same reward

Drawbacks:
number of joint actions is exponential

4. Partially Observable Conditions (Dec-POMDPs)

4.1. Model

- need to consider joint actions : transitions and rewards depend on joint actions
- and joint observations

4.2. Subjective ways

Treat agents like 1 "puppeteer agent", that means **a single team**, that partly means, there is a single agent or cloud brain (or all agent do the same things):

- receives joint observations
- takes joint actions and send it to different agent
- there is only a single reward function which gives the same reward for all agents.

Drawbacks

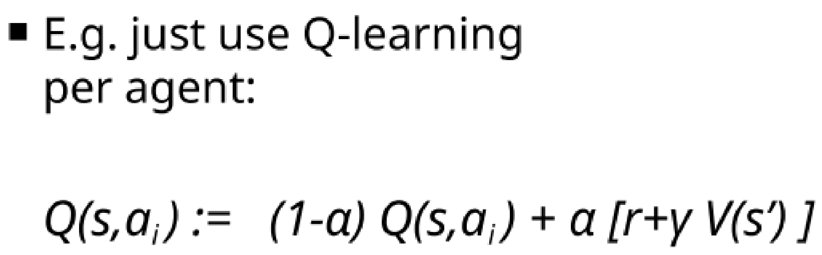
- requirest **optimal communicationa**: nstantaneous, cost-free, noise-free communication
- **number exponential**

5. Multiagent RL: objective way

5.1. Two Original Ideas

5.1.1. Individual Learning

Agents learning from the environment **individually**



- E.g. just use Q-learning per agent:

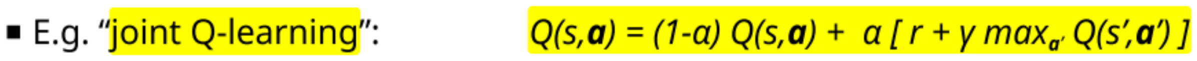
$$Q(s,a_i) := \ (1-\alpha)\ Q(s,a_i) + \alpha\ [r+\gamma\ V(s')\ ]$$

Drawbacks:

- **the environment is changing** (non-stationary learning problem): another agent action affect the whole environment
- no guarantees for convergence
- and it sometimes converge to a local optimum: agents do not know others and nobody try to help others
- did not take other's observations into consideration

5.1.2. Coupled Learning

learning based on joint actions (with individual state)



- E.g. "joint Q-learning": $Q(s,\mathbf{a}) = (1-\alpha)\ Q(s,\mathbf{a}) + \ \alpha\ [\ r + \gamma\ \max_{\mathbf{a}'} Q(s',\mathbf{a}')\ ]$

Drawbacks:

- **exponential** in number of agent
- **need to observe full state**, an agent cannot alwasy make decision based on its own observation

5.1.3. What we need

capability to reason over other agent's observations

5.2. Formal Frameworks

5.2.1. Formal Model

In a Dec-POMDP： $< S, A, P_T, O, P_o, R, h >$

$$\begin{aligned} n:&\quad \text{agent}\\ S:&\quad \text{set of states}\\ A:&\quad \text{set of joint actions}\\ P_T:&\quad \text{transition function}\\ O:&\quad \text{set of joint observations}\\ P_O:&\quad \text{observation function}\\ R:&\quad \text{reward function}\\ h:&\quad \text{horizon(finite)}\end{aligned}$$

Goal:

find an optimal joint policy

value function:

$$V(\pi) = \boldsymbol{E}\left[\sum_{t=0}^{h-1} R(s,a) \mid \pi, b^0\right]$$

5.2.2. Policy Domain

Comparing to the Non-Dec Situation:

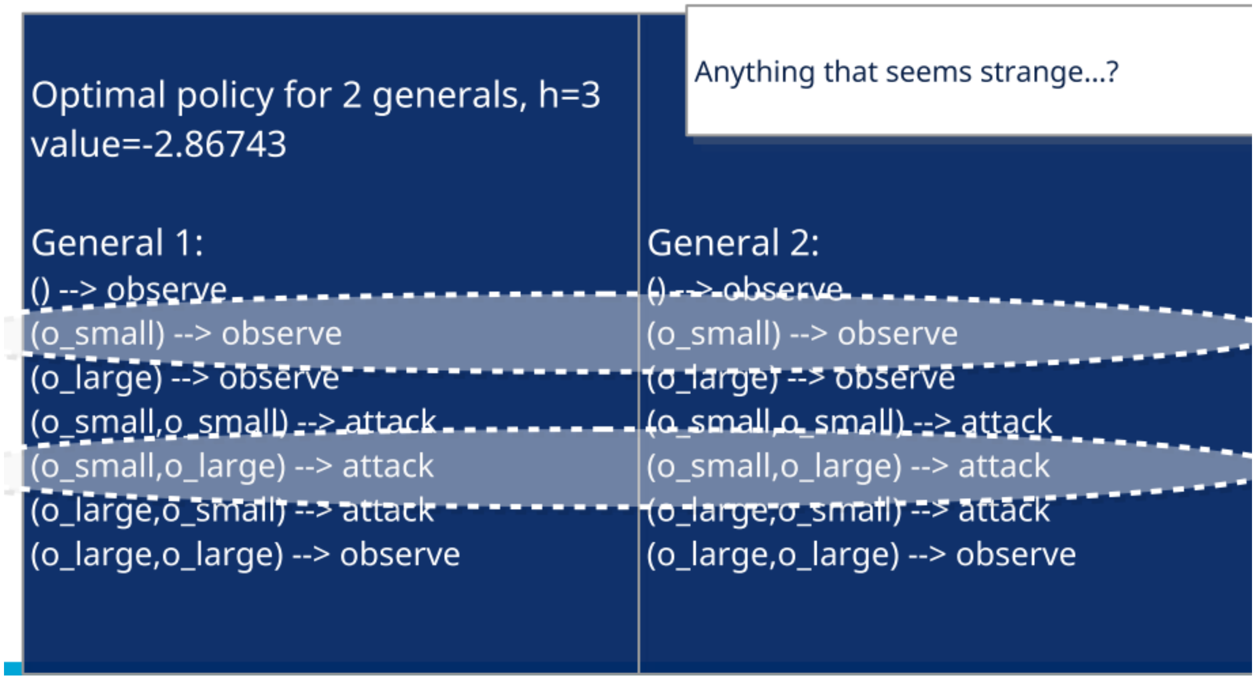
1. **MDPs:** an agent can ignore the history (of states) because of the Markov property
2. **POMDPs: use belief to compass the history** without sacrificing optimality
3. **But, in Dec-POMDPS:** we can only accecss to its individual actions and observations, so the agents do not have access to a Markovian signal during execution. **That means policy involves searching the space joint Dec-POMDP policies that map full-length individual histories to actions**
4. Someone has raised **Joint Belief b(s)**
   - compute b(s) using joint actions and observations
   - But, agents may not know those during execution

PRELIMINARIES:

1. Action-observation history:
   $\bar{\theta} i, t = (ai, 0, o_{i,1}, \ldots, a_{i,t-1}, o_{i,t})$
2. Observation history
   $\bar{o} i, t = (oi, 1, \ldots, o_{i,t})$
3. Action history
   $\bar{a} i, t = (ai, 0, a_{i,1}, \ldots, a_{i,t-1})$
4. degerministic policy:
   A deterministic policy πi for agent i is a mapping from observation histories to actions, $\pi_i : O_i \rightarrow A_i$

5.3. Optimal Policy

Optimal Policy should make tradeoff: **coordination vs. exploitation of local information**



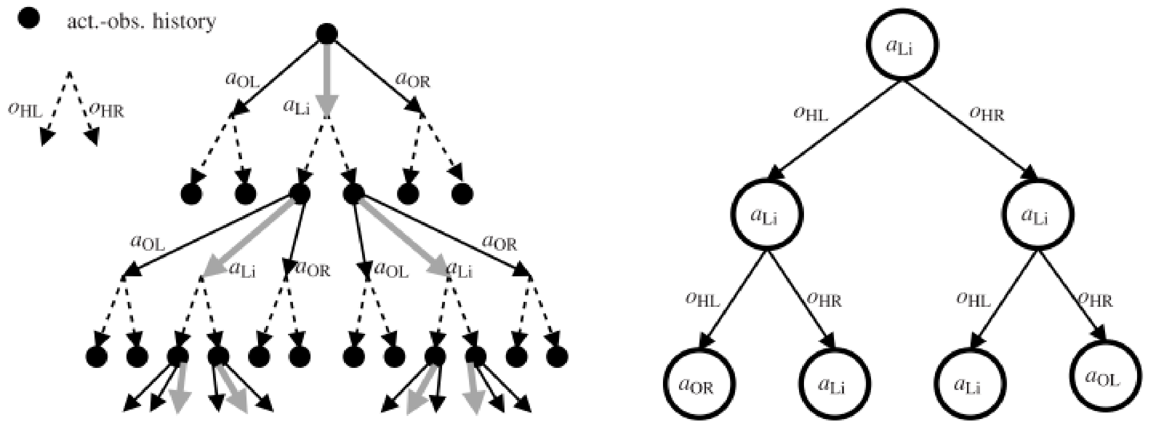
Anything that seems strange...?

Optimal policy for 2 generals, h=3
value=-2.86743

General 1:
() --> observe
(o_small) --> observe
(o_large) --> observe
(o_small,o_small) --> attack
(o_small,o_large) --> attack
(o_large,o_small) --> attack
(o_large,o_large) --> observe

General 2:
() --> observe
(o_small) --> observe
(o_large) --> observe
(o_small,o_small) --> attack
(o_small,o_large) --> attack
(o_large,o_small) --> attack
(o_large,o_large) --> observe

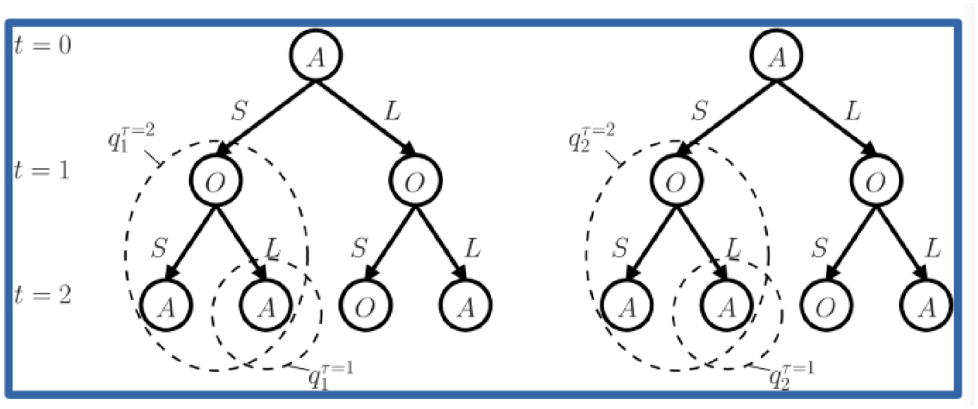
Under a deterministic policy, only a subset of possible action-observation histories can be reached



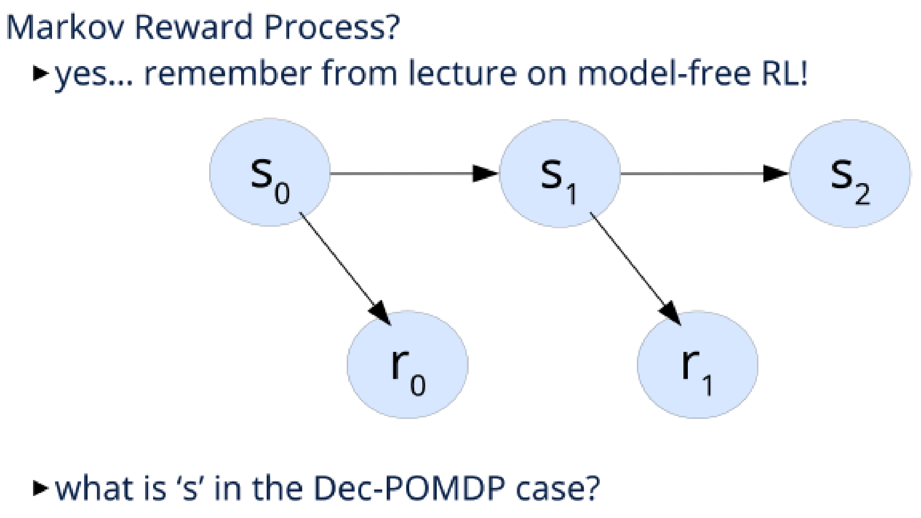
5.3.1. Value of Joint Policy

Representing Policy

Sub-tree policies:



Property


Markov Reward Process?

- yes... remember from lecture on model-free RL!

- what is 's' in the Dec-POMDP case?

s in Dec-POMDP is the history, in other words is the tree that has executed

Value

$$V\left(\vec{h}, q^{\tau=k}\right) = R(\vec{h}, a) + \sum_{o} P(o \mid \vec{h}, a) V\left(\vec{h}', q^{\tau=k-1}\right)$$

where

$\vec{h}$ is the joint action-observation history $\vec{h} = (\vec{h}_1, \vec{h}_2)$

And we can also conclude it as a

5.3.2. Ways to find optimal policy

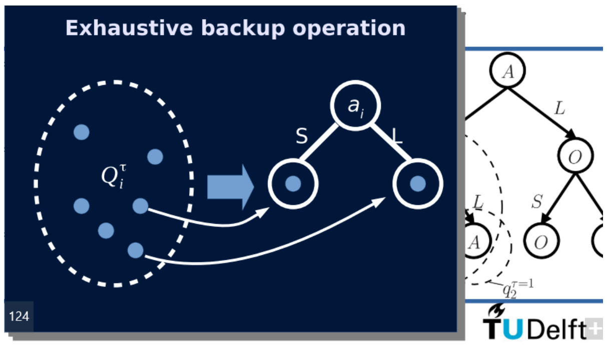
5.3.3. Brute Force Seach

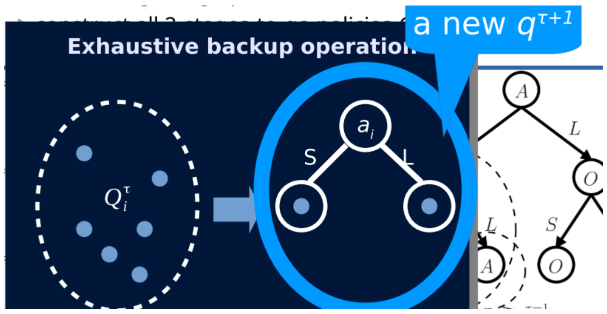
- **compute all value** of joint policy
- But Number of joint policies is **huge**

5.3.4. Dynamic Programming

Generate all policies in a special way:

- from 1 stage-to-go policies $Q^{\tau=1}$
- construct all 2-stages-to-go policies $Q^{\tau=2}$, etc.

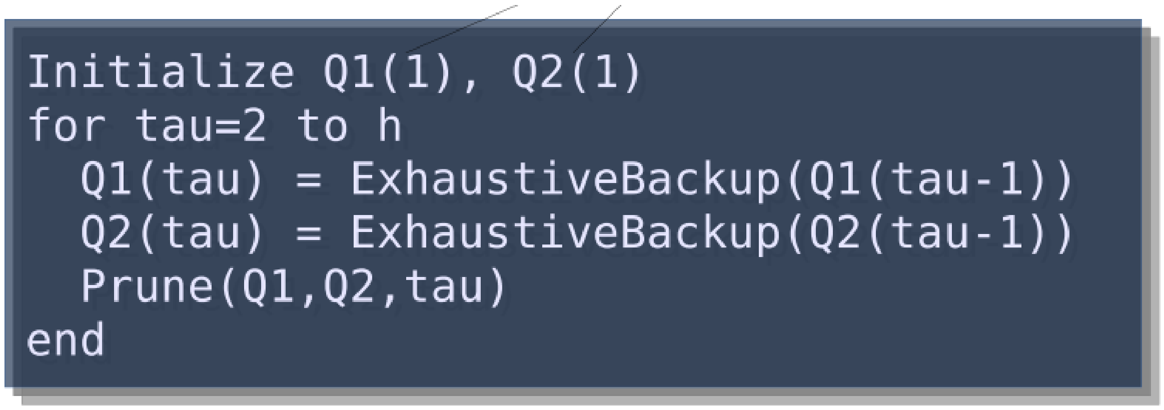




- Although better thant Brute Force Search, the scale are still very large

5.3.5. Dynamic Programming with Pruning

Perhaps not all those Q^τ is useful, so we can use **pruning** to prune **dominated policies**



```
Initialize Q1(1), Q2(1)
for tau=2 to h
  Q1(tau) = ExhaustiveBackup(Q1(tau-1))
  Q2(tau) = ExhaustiveBackup(Q2(tau-1))
  Prune(Q1,Q2,tau)
end
```

At the end of iteratio:

- evaluate all the ramaining combinations of policies
- select the best one

6. MARL: a subjective way

You cannot really compute the belief over states since the agent does not have a model of the world as it is in a partially observable environment

he number of joint actions grows exponentially in the number of agents(cross product of action set of each agent)

When you do Q learning, you are doing updates that are affected by the actions that are taken by the other agent. The agents are still learning over time and their strategy could be evolving over time and not static. This results in a non-stationary learning problem where there are no guarantees of convergence