Multiagent Reinforcement Learning: Rollout and Policy Iteration (Bertsekas, 2020)

Mikalai Korbit

IMT School for Advanced Studies Lucca

January 28, 2021

Outline

Motivation

Multi-agent systems are ubiquitous

Eg. fleet of drones, factory robots, self-driving cars.

Recent advances in RL applications

Eg. AlphaGo/AlphaZero, playing Starcraft, robotic control.

Utilize modern computer architecture and software frameworks

Eg. cloud computing, stacks of graphics cards, TPUs; PyTorch, OpenAI gyms.

Benefits of modeling a problem as MARL

Scalability, robustness, faster learning through experience sharing, parallel computation.

Multi-Agent Reinforcement Learning Problem

Inherits Reinforcement Learning characteristics:

- Learning how to map situations into actions
- Trial-and-error search
- Delayed feedback
- Trade-off between exploration and exploitation
- Sequential decision making
- Agent's actions affect the subsequent data it receives

Adds multi-agent features:

- Actions of one agent influence other agents' rewards
- Communication problem
- Curse of dimensionality (more severe than in RL)

"Bertsekas Dictionary"

Aligns optimal control definitions with the RL-world:

- Maximize value $\rightarrow$ minimize cost
- Agent $\rightarrow$ Decision maker or controller
- Action $\rightarrow$ Decision or control
- Environment $\rightarrow$ Dynamic system
- Learning $\rightarrow$ Solving a DP-related problem using simulation
- Self-learning (self-play) $\rightarrow$ Solving a DP problem using simulation-based policy iteration
- Planning vs Learning distinction $\rightarrow$ Solving a DP problem with model-based vs model-free simulation

Multi-Agent MDP

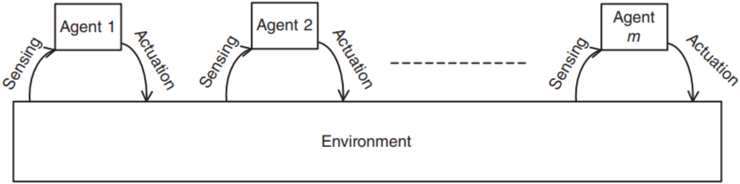

Figure: MARL Problem. Source: Sadhu, Konar (2020)

- All agents see the global state s
- Individual actions: $u^a \in U$
- State transitions: $P(s' \mid s, \mathbf{u}) : S \times \mathbf{U} \times S \to [0,1]$
- Shared team reward: $S \times \mathbf{U} \to \mathbb{R}$

Taxonomy

Cooperative

- The goal of cooperative agents is to achieve a common objective
- Coordination problem

Competitive

- Zero-sum games (eg. chess, tic-tac-toe)
- Minimax equilibria

Mixed

- General-sum games (win-win, lose-lose scenarios; eg. pollution model, “what movie to watch?”)
- Nash equilibria

Algorithms for Cooperative MARL

Static games

- JAL
- FMQ

Dynamic games

- Team-Q
- Distributed-Q
- OAL
- SCQL
- SQL
- FMRQ

TODO

- Cooperative $P1_F$

Reinforcement Learning Problem

- Learning how to map situations to actions
- Trial-and-error search
- Delayed feedback
- Trade-off between exploration and exploitation
- Sequential decision making
- Agent's actions affect the subsequent data it receives

Conclusion

- RL methods can be applicable to a wide variety of problems
- Out-of-the-box models work but require fine-tuning and take longer to converge
- Simple methods like state discretization are worth exploring when training speed and solution complexity are of the essence

References

Dimitri Bertsekas – Multiagent Reinforcement Learning: Rollout and Policy Iteration (2020). Web: https://web.mit.edu/dimitrib/www/Multiagent_Sinica_2020.pdf

Shimon Whiteson – Factored Value Functions for Cooperative Multi-Agent Reinforcement Learning (2020) [Seminar].

Arup Kumar Sadhu, Amit Konar – Multi-Agent Coordination, A Reinforcement Learning Approach (2020).

Thanks for your attention!