

Gemini - learning cooperative behaviors without communicating

Gemini

Masayuki Ohta

MO: Tokyo Institute of Technology

Abstract. *This paper describes the design of Gemini a client program for SoccerServer. The goal of Gemini is learning cooperative behaviors without direct communication in multi-agent environment. With recent implementation, it can select the best strategy against opponent, statistically.*

1 Introduction

To make agents behave cooperatively in multi-agent environment, it is important to have a consensus of the state. In case of RoboCup problem, it is easy to make agreements, because they can obtain the information from the global viewpoint, if every agent “say” the information it have. So that our interest is to make agents cooperate efficiently in the following occasion, Gemini dare to prohibit using “say” command.

- No information from global viewpoint
- No consensus for world model between agents
- No protocol is decided before

We use reinforcement learning to approach this environment. When we apply reinforcement learning to the multi-agent environment, each agent which has their own goal try a lot of strategy to find the best one. In such case, the “reward collision problem” will occur. We solve this problem with “excessively self-conscious” agent technique.

In this paper, we show the architecture fundamentals of Gemini: “action flow”, “strategy hierarchy” and “role”, then introduce the learning method we use.

2 Architecture

2.1 Action flow

Using timer interruption, Gemini execute the following actions in every 100m second.

- **estimate the object's position and velocity**
Calculate all object's position in 100m second.
- **receive perceptual information**
If perceptual information was received, convert it to the internal representation.
- **change strategy**
If the strategy now selected is completed or some unexpected accident occur, select a new strategy, and memorize(for learning).
- **execute a basic action**
In the action phase, the next command written by if-then rules will be decided along with the strategy, reactively.
- **learning**
Changes parameters of the strategies. The strategies include “positioning”, “when to dribble or pass”, “how offensive does it act”, etc. This change of parameters affect the selection of strategies.

This loop will be repeated until the end of the game.

2.2 Strategy Hierarchy

In the RoboCup environment, an agent can send only one primitive command to SoccerServer at a time. But, there are some cases that only a sequence of action can achieve the goal. Then, the agent should not decide one action for one perceptual information reactively, but the action based on the longer term strategy.

We define the strategies hierarchically for Gemini. It have four layers (from the top to the bottom): Long Term Strategy, Short Term Strategy, Advanced Skill, Basic Action. Upper layer strategy will be a constraint, and lead a sequence of action in the lower layer.

- **Long Term Strategy**
This layer has the strategy for the prospect of the game, such as “steal the ball”, “assist a teammate”, etc. This layer outputs the short term strategy to achieve the long term strategy which is now selected.
- **Short Term Strategy**
This layer is responsible for more concrete strategy than upper layer. It generates the best way to realize the short term strategy, such as

“side change”, “centering”, etc. The output of this layer is one of the advanced skill defined in the next layer.

- **Advanced Skill**

Advanced skills such as “pass”, “shoot”, etc. are defined in this layer. The strategy in this layer means the parameters to calculate the attribute for the basic actions. The output of this layer is a sequence of the basic action.

- **Basic Action**

Basic action such as “kick”, “dash”, etc. are described in this layer. Each action corresponds to the command from the client to the SoccerServer. This layer checks the parameter of those command, and send a message to the SoccerServer.

2.3 Role

Role is not defined beforehand in Gemini. All players (except for goal keeper) behave just the same at the beginning. Start with different position, each player learn their best behavior respectively. For example, players at the forward position may learn to play offensively, because getting a goal is so profitable that forward players ignore some risk of losing the ball, and players in front of our goal may learn to play defensively, because being scored is so harmful to defenders. The result of this learning is depend on the position, and we call it our role.

This role is effectual only with the teammates who learn together. The agent pass the ball and the agent receive the ball should have consensus of the strategy.

3 Learning

Gemini select the most effective strategy against the opponent. This is implemented by learning the variables (like “positioning”, “when to dribble or pass”, “how offensive does it act”, etc.) to select the strategy. Each agent do the reinforcement learning[3] during the match, start with the value adjusted by human.

In the “change strategy” phase, Gemini memorize the strategy now selected, and in the “learning” phase, it reinforce the stored strategy using stochastic gradient ascent[4] a kind of profit sharing[2], when their team get the score, or they kick the ball. Furthermore, if the ball moved nearer to the opponent’s goal than 5 seconds ago, reinforce the stored strategies, otherwise, forget the stored strategies. Each reward is adjusted by human to the value expedite learning.

Because each agent learns individually, all agents evolve into raising their own reward. This possibly cause collision of reward, and reduce whole reward of the team. This problem can be solved when the agents negotiate each other, but if we think of such environment that describe above section, an agent have to avoid this problem itself.

We solve this problem with “excessively self-conscious” agent technique[1]. “excessively self-conscious” agent have following rewarding feature: if a teammate contribute to achieve the goal, the agent reinforce the strategy at that time. With this reinforcement method, we can improve the reward of whole team, and prevent from losing individuality.

Now we are adding some learning items, but there will be another problem that increasing the number of learning item disturb the advance of learning. This is one of our future job.

4 Conclusion

In this paper, we describe the outline of Gemini: a client program for SoccerServer. Using reinforcement learning, it can change their behavior against opponent, statistically. This learning also generate the role of each agent as there position.

Because we are interested in the environment that all agent are prohibited communicating directly, Gemini never use the “say” command. In such circumstances “reward collision problem” will occur. We solved it using “excessively self-conscious” agent.

References

- [1] M.Ohta and T.Ando “Cooperative Reward in Reinforcement Learning” Proc. of 3rd JSAI RoboMech Symposia pages 7-11 April 1998.
- [2] Grefenstette, John.J. “Credit Assignment in Rule Discovery Systems Based on Genetic Algorithms” Machine Learning, Vol.3 pages 225-245 1988.
- [3] Kaelbling L. P., Littman M. L. and Moore A. W. “Reinforcement Learning: A Survey” Journal of Artificial Intelligence Research 4, pages 237-285 1996.
- [4] Kimura H., Yamamura M. and Kobayashi S. “Reinforcement Learning in Partially Observable Markov Decision Processes: A Stochastic Gradient Method” Journal of Japanese Society for Artificial Intelligence, Vol.11, No.5 pages 761-768 1996