

META-LEARNING PROCESSES IN MULTI-AGENT SYSTEMS

RON SUN

CECS, University of Missouri, Columbia, MO 65211, USA

E-mail: rsun@cecs.missouri.edu

Straightforward reinforcement learning for multi-agent co-learning settings often results in poor outcomes. Meta-learning processes beyond straightforward reinforcement learning may be necessary to achieve good (or optimal) outcomes. Algorithmic processes of meta-learning, or “manipulation”, will be described, which is a cognitively realistic and effective means for learning cooperation. We will discuss various “manipulation” routines that address the issue of improving multi-agent co-learning. We hope to develop better adaptive means of multi-agent cooperation, without requiring a priori knowledge, and advance multi-agent co-learning beyond existing theories and techniques.

1 Introduction

It is common that a group of agents deal with a situation jointly, with each agent having its own goal and performing a sequence of actions to maximize its own payoffs. However, difference sequences of actions (by different agents) interact in determining the final outcome for each agent involved. In such a situation, each agent has to learn to adapt to other agents that are present and “negotiate” an equilibrium state that is beneficial to itself (e.g. Kahan and Rapoport 1984). Our focus will be on nonverbal “communications” in which (sequences of) actions by agents may serve the purpose of communicating intentions and establishing cooperation, in an incremental and gradual way. The ultimate goal is to avoid mutually harmful outcomes and to distribute payoffs to individual agents in a rational way.

We need some framework that determines a proper strategy for each agent in order to deal with the presence of other agents. The framework should allow adaptive determination of strategies on the fly during interaction. The framework should allow learning from scratch without a priori domain knowledge.

Game theory (e.g., van Neumann and Morgenstern 1944) has been focusing on *static* equilibria of strategies in a variety of game settings (Osborne and Rubinstein 1994), which furthermore unrealistically assumes *unbounded* rationality on the part of agents (Simon 1957). The recent surge of study of game learning (e.g., Fudenberg and Levine 1998, Camerer and Ho 1999) brings adaptive processes of reaching equilibria into the focus. To study the dynamics of reaching equilibria, learning algorithms need to be developed. However,

learning algorithms studied in much existing work on game theoretic learning have been overly simple and, thus, to a large extent, unrealistic.

Beyond static equilibria and simple learning, complex *algorithmic* processes involved in learning by cognitive agents (Sun et al 2001) need to be studied. By complex *algorithmic* processes, I mean procedures that include detailed, varied, and subtle steps of manipulation of information, strategies, or equilibria (see Sun and Qi 2000, Sonsino 1997 for preliminary versions of such processes). I hope that, by incorporating such complex algorithmic processes, we can extend game theoretic studies to more realistic settings of multi-agent interaction.

What I emphasize in this work is not just end results (i.e., equilibria; Osborne and Rubinstein 1994), not just simple processes involving simple operators (Fudenberg and Levine 1998), but complex algorithmic operations and processes. This is because what an agent may do is not limited to completely rational choices as assumed by many game theorists, but also some apparently irrational behaviors which may nevertheless lead to desirable outcomes in the future. We are interested in learning and action selection that are more opponent-oriented and more determined on the fly, than many existing processes. This kind of algorithmic process helps to improve cooperation.

2 Background

2.1 Game Theory

Game theory studies decision making involving multiple agents (Osborne and Rubinstein 1994). A *strategic* game is the one in which all agents choose their actions simultaneously and once for all. In contrast, in *an extensive* game, agents perform actions sequentially. Formally, an extensive game is a 4-tuple: (N, H, P, U) , where N is a set of agents, H is a set of history (see Osborne and Rubinstein 1994 for further specifications), P is the player function such that $P(h)$ specifies the player after history $h \in H$, U is the payoff function that maps each terminal history to a real value.

For simplicity, in the following discussions, we assume the length of games is finite, that is, each game always terminates in a finite number of steps. Each agent has perfect information.

Given these assumptions, we will look into extending current game theory, incorporating more complex algorithmic processes that capture more realistic cognitive and social processes during game learning.

2.2 Reinforcement Learning

Reinforcement learning is a general learning framework suitable for learning extensive games. In a single-agent learning setting, there is a discrete-time system, the state transitions of which depend on actions performed by an agent. A Markovian process determines state transition after an action is performed. Costs (or rewards) can occur for certain states and/or actions. Normally the costs/rewards accumulate additively, with or without a discount factor $\gamma \in (0, 1]$.

One algorithm for learning optimal policies is the Q-learning algorithm of Watkins (1989):

$$Q(s_t, a_t) := (1 - \alpha)Q(s_t, a_t) + \alpha(r(s_{t+1}) + \gamma \max_{a_{t+1} \in A} (Q(s_{t+1}, a_{t+1})))$$

where α is the learning rate, which goes toward zero gradually. Action a_t is determined by an exploration action policy, e.g., using (1) alternating exploration (random actions) and exploitation (greedy actions) periods, (2) a small fraction of random actions (with probability ϵ , a random action is chosen; with probability $1 - \epsilon$, a greedy action is chosen), or (3) stochastic action selection with the Boltzmann distribution. Such an algorithm allows completely autonomous learning from scratch, without a priori domain knowledge.

Extending Q-learning to co-learning in extensive games, we may simply use the above single-agent Q-learning equation, or we may use multi-agent Q-learning equations (Littman 2001).

We assume that each state (used in Q-learning), or information set (as termed by game theorists), is comprised of all the actions up to the current point and, optionally, information about the initial situation at the time when the game begins. State transitions are deterministic.

We assume that there is sufficient exploration during reinforcement learning (which is a standard requirement for ensuring convergence of RL), so that each agent knows the payoff outcomes of all the paths on the game tree. But, eventually, each agent converges to a deterministic action policy, i.e., a pure strategy in game theoretic terms.

3 Types of Meta-Learning

In practice, performance of Q-learning in extensive games tends to be very poor (Shoham and Tennenholtz 1994, Sandholm and Crites 1995, Haynes and

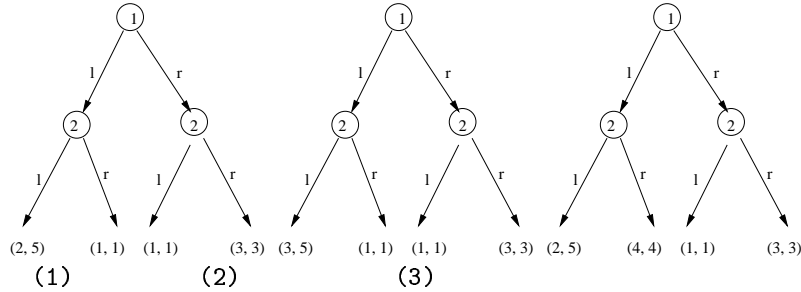


Figure 1. Three cases of the left/right game. The numbers in circles indicate agents. l and r are two possible actions. The pair of numbers in parentheses indicate payoffs (where the first number is the payoff for agent 1 and the second for agent 2).

Sen 1996, Sun and Qi 2000), despite the fact that Q-learning is cognitively justifiable (Sun et al 2001). The problem may lie in the fact that other cognitive mechanisms may also be needed, on top of such trial-and-error learning, in order to achieve good performance (Sun et al 2001). In this paper, I shall explore additional adaptive mechanisms (i.e., meta-learning routines), within a RL framework, to facilitate the attainment of optimal or near optimal results.

3.1 Manipulation by Preemptive Actions

An agent can manipulate other agents by adopting suboptimal actions (likely temporarily) in order to force or induce opponents to take actions that are more desirable to the agent but result in lower payoff (and are thus less desirable) to themselves. For example, in the game of Figure 1 (1), with reinforcement learning, agent 1 will end up rationally choosing action r , which will lead agent 2 to choose its best action r , and hence a payoff of (3,3) for them. However, agent 2 prefers the outcome of (2,5). Therefore, it may deliberately choose l after agent 1 chose r , leading to a payoff of (1,1), which forces agent 1 to change its action. With further reinforcement learning, agent 1 may adapt to this change and choose l instead (because this action can lead to the best possible outcome, if agent 2 rationally chooses l afterwards, given its manipulative action of l after agent 1's r). This change gives agent 2 its preferred outcome of (2,5).

We assume that there are only two agents in an extensive game, which take turn in acting. Assume that after sufficient learning using Q-learning, a *subgame perfect equilibrium* (Osborne and Rubinstein 1994) is reached. Sufficient exploration is done during reinforcement learning and, thus, each agent

have fairly accurate knowledge of the payoffs of different paths on the entire game tree.

After reinforcement learning settles into a particular payoff outcome, assume there is an alternative outcome (not necessary a Nash equilibrium) that is more desirable to an agent but less desirable to its opponent. Suppose this alternative outcome can be reached if the opponent takes a different action at a certain point (but follows the current optimal policy as determined by the reached subgame perfect equilibrium thereafter). This point (the targeted switch point) can be determined by a search of the game tree:

Algorithm 1.

1. Search from the root of the tree along the current (equilibrium) path using depth-first search. At each point of action by the opponent, do the following:
 - 1.1. Adopt an alternative action.
 - 1.2. Follow the current, optimal policy (the subgame perfect equilibrium strategy) of each agent thereafter.
 - 1.3. If one of these alternative actions leads to a more desirable outcome for the agent, add the whole path to the candidate path set.
2. Choose from the candidate path set the most desirable path. Start the manipulation process at the point of the alternative action by the opponent in the chosen path.

Here is what the agent can do to change the action by the opponent at the point (the manipulation):

Algorithm 2.

- Search the subtree that starts at the action (by the opponent) that the agent aims to change (using depth-first search):
1. If there is an alternative action by the agent at any point along the current path in the subtree, that creates a path (1) that leads to a payoff for the opponent that is lower than the payoff of the most desired path, and (2) on which all other actions (by either agents) conform to the optimal policies (determined by the equilibrium), then, commit to that action (i.e., perform that action whenever that point of the game tree is reached).
 2. If there are multiple such actions by the agent, choose the one highest in the tree (that is, the closest to the current action by the opponent to be changed).

The algorithm is based on the following result:

Theorem 1 *For the subgame described by the part of the game tree below the point of the committed (manipulating) action (of the manipulating agent),*

the original subgame perfect equilibrium strategies for both agents remain the subgame perfect equilibrium strategies.

Thus, the acquired policies below the changed action need not be changed and re-learned. Similarly,

Theorem 2 *For the subgame described by the part of the game tree below the point of an alternative action by the opponent, the original subgame perfect equilibrium strategies for both agents remain the subgame perfect equilibrium strategies.*

An obvious shortcoming of Algorithm 1 is the cost of the exhaustive search used. An alternative is to search and find only one desirable path for the agent, with a straightforward modification of Algorithm 1, and then to force the opponent to go down that path using Algorithm 2. We may similarly eliminate exhaustive search in Algorithm 2.

In either case, the hope is that the opponent will opt for an alternative action at the targeted switch point that leads to a better outcome for the agent (as a result of further trials and further learning by the opponent during those trials). However, there may be multiple action choices for the opponent at this point or another (before the committed action of the agent). The opponent may opt for an action that is not the desired action. To force the opponent to take the desired action, the agent needs to close off all loopholes (all “distractor” paths). That is, the above algorithm can be repeatedly applied, if the desired outcome is not reached due to the opponent taking an unintended action at a point above the committed action by the agent. This process can continue until all the other alternatives are “eliminated” except the desired path (or, when an outcome that is equivalent to, or better than, the desired outcome is reached).^a

As a result of further trials during which further reinforcement learning occurs, the opponent may adapt to the manipulation and take the target action intended for it by the agent. Thus the game settles into a new state that is a subgame perfect equilibrium state *given* the manipulation (i.e., with the original action by the agent at the point of manipulation being “prohibited” or removed).

However, the opponent may counter-react to the manipulation. First of all, counter-reaction may take the form of *obstinacy*: The opponent can refuse to change any action despite the worsened outcome as a result of the manipulation and despite the existence of alternative actions that can lead to better outcomes (although they may not be as good as the original outcome). Sec-

^aAlternatively, we may at once lower the payoffs of all the alternative actions for the opponent, if they are higher than that of the desired outcome for the opponent (see Sun et al 2001).

ond, counter-reaction may also take the form of *counter-manipulation* using the same algorithm described above. The opponent can, e.g., eliminate the outcome that is the most desirable for the original agent (and thus was the goal of the original manipulation). These issues are dealt with elsewhere and not repeated here due to the page limit.

3.2 Manipulation by Nudging Actions

This is the case of an agent adopting some suboptimal actions in order to direct its opponent to take actions that are equally, or more, desirable to each agent involved. As a result of the manipulation, everyone receives a payoff that is equal to, or higher than, the payoff each would receive otherwise (without the manipulation). This type of manipulation is obviously easier to accomplish, and does not call for counter-reaction from opponents.

If there is a point (the targeted switch point) along the equilibrium path in the game tree (as determined by the reached subgame perfect equilibrium) where an alternative action by the opponent may lead to better payoffs to the agent and no worse payoffs to its opponent, then the agent can take a non-optimal action at a point below the afore-identified targeted switch point to force a worse payoff on the opponent if it follows the old path. The algorithms for reaching the desired outcome, including selecting the switch point and forcing the switch, have been discussed earlier and remain the same.

For example, in the game of Figure 1 (2), with reinforcement learning, agent 1 may choose action r , which leads agent 2 to choose action r , and hence a payoff of $(3, 3)$ for them. However, agent 2 prefers the outcome of $(3, 5)$. Therefore, it decides to take l after agent 1 took r , leading to a payoff of $(1, 1)$ for them, in order to nudge agent 1 to change its action as well. With further reinforcement learning, agent 1 adapts to this change and chooses l instead, which leads to the outcome of $(3, 5)$ — a better outcome for agent 2. This is a special case of the previously discussed scenario where there is no need for the opponent to consider counter-reaction.

3.3 Manipulation by (Mutual) Compromise

An agent can adopt some suboptimal actions, in order to induce its opponent to take actions that are suboptimal too, which together, however, can lead to outcomes more desirable to both agents involved. This case can be viewed as reaching a mutually beneficial compromise in order to maximize the payoffs of all those involved.

For example, in the game of Figure 1 (3), with reinforcement learning, agent 1 will end up choosing action r , which leads agent 2 to choose action r ,

and hence a payoff of (3, 3) for them. However, agent 2 prefers the outcomes of (2, 5) or (4, 4). It cannot easily induce agent 1 to an outcome of (2, 5), because it gives agent 1 a worse payoff. But it can induce agent 2 to an outcome of (4, 4). Therefore, it consistently takes r if agent 1 takes l , which gives agent 1 incentives to take l instead of r (because it leads to a better payoff for agent 1). With further reinforcement learning, agent 1 settles on action l , which leads to the outcome of (4, 4) — a compromise between the two agents.^b

As a result of the manipulation, everyone receives a payoff that is higher than the payoff each would receive otherwise (without the manipulation). However, as with the previous cases of manipulations, the resulting outcome is not a Nash equilibrium, and it is stable only under the reached compromise (i.e., *given* the committed action choice).

Algorithm 3.

1. Search from the root of the tree along the current (the subgame perfect equilibrium) path. At each point of action by the opponent, and at each point of action by the agent itself following that, try a pair of alternative actions. That is, repeat the following (using depth-first search):
 - 1.1. Adopt an alternative action at a point of action by the opponent.
 - 1.2. Follow thereafter the current policy (the subgame perfect equilibrium strategy) of each agent, except the following change.
 - 1.3. At a point of action by the agent itself, try an alternative action.
 - 1.4. If this pair of alternative actions leads to more desirable outcomes for both agents, store the pair as a candidate pair.

Now, the agent commits to his part of this compromise (a chosen pair of alternative actions):

Algorithm 4.

- If there is at least one candidate pair (that is, if at least one of these pairs of alternative actions led to more desirable outcomes for both agents), start the manipulation process:
1. Find the best such pair (based on a criterion such as the maximum increase of payoffs for the manipulating agent, or the highest total increase of payoffs for both agents).
 2. Commit to the action (of the agent) from the chosen pair of actions.

Without explicit communication, the agent has to wait for the opponent to discover this commitment through exploration during further reinforcement learning. Most likely, the opponent will discover the advantage of taking the

^bNote that, in this game, it is also possible for agent 2 to take preemptive actions to force an outcome of (2,5), as in section 3.1.

corresponding action determined by this compromise, and thereafter both agents will be able to reap the benefit.

We show below that the desired action change of the opponent as determined by the compromise will lead to the highest possible payoffs for both agents, given the manipulation, and thus there is sufficient incentive for the opponent to take that action determined by the compromise:

Theorem 3 (1) *The targeted action change of the opponent will lead to the highest possible payoffs for both agents, given the manipulation.* (2) *The optimal policies of both agents, either below the agent's manipulating action, above the targeted alternative action by the opponent, or in between the two points, will not be changed due to the manipulation.*

Note that, compared with earlier types of manipulations, here the agent chooses to *induce* rather than to *force* its opponent to take the action it wants it to take. This mutual compromise process can be extended to more than two steps.

4 Concluding Remarks

This paper considers algorithmic meta-learning processes of joint sequential decision making that are both cognitively realistic and practically effective. In essence, we incorporate more cognitively realistic learning processes by combining simple decision making studied in game theory with complex algorithmic processes (Sun et al 2001). Armed with extended senses of rationality, we are aiming at an algorithmic account of multi-agent learning of cooperation (that starts from scratch without a priori domain knowledge). Of course, it is important that we extend our basic assumptions to deal with more general cases, which are being worked on right now.

References

1. C. Camerer and T. Ho, (1999). Experience-weighted attraction learning in normal-form games. *Econometrica*, 67, 827-874.
2. C. Claus and C. Boutilier, (1998). The dynamics of reinforcement learning in cooperative multiagent systems. *Proceedings of AAAI'98*. AAAI Press, San Mateo, CA.
3. D. Fudenberg and D. Levine, (1998). *The Theory of Learning in Games*. MIT Press, Cambridge, MA.

4. T. Haynes and S. Sen, (1996). Co-adaptation in a team. *International Journal of Computational Intelligence and Organizations*.
5. J. Kahan and A. Rapoport, (1984). *Theories of Coalition Formation*. Lawrence Erlbaum Associates, London.
6. M. Littman, (2001). Value-function reinforcement learning in Markov games. special issue on multi-agent learning (edited by R. Sun), *Cognitive Systems Research*, Vol.2, No.1, 2001.
7. J. Nash, (1950). Equilibrium points in N-person games. *Proceedings of National Academy of Science*, vol.36, 48-49.
8. M. Osborne and A. Rubinstein, (1994). *A Course on Game Theory*. MIT Press, Cambridge, MA.
9. T. Sandholm and R. Crites, (1995). Multiagent reinforcement learning in the iterated prisoner's dilemma. *Biosystems*, 37, 147-166.
10. S. Sen and M. Sekaran, (1998). Individual learning of coordination knowledge. *Journal of Experimental and Theoretical Artificial Intelligence*, 10, 333-356.
11. Y. Shoham and M. Tennenholtz, (1994). Co-learning and the evolution of social activity. Technical Report STAN-CS-TR-94-1511, Stanford University.
12. H. Simon, (1957). *Administrative Behavior* (2nd ed.). New York: Macmillan.
13. D. Sonsino, (1997). Learning to learn, pattern recognition, and Nash equilibrium. *Games and Economic Behavior*, 18, 2, 286-331.
14. V. Soo, (2000). Agent negotiation under uncertainty and risk. *Design and Applications of Intelligent Agents*. pp.31-45. Springer-Verlag, Heidelberg, Germany.
15. R. Sun, E. Merrill, and T. Peterson, (2001). From implicit skills to explicit knowledge: a bottom-up model of skill learning. *Cognitive Science*.
16. R. Sun and D. Qi, (2000). Rationality assumptions and optimality of co-learning. *Design and Applications of Intelligent Agents*. Lecture Notes in Artificial Intelligence, Volume 1881. pp.61-75. Springer-Verlag, Heidelberg, Germany.
17. J. von Neumann and O. Morgenstern, (1944). *Theory of Games and Economic Behavior*. John Wiley and Sons, New York.
18. C. Watkins, (1989). *Learning with Delayed Rewards*. Ph.D Thesis, Cambridge University, Cambridge, UK.