Cooperation through communication in decentralized Markov games

Raghav Aras, Alain Dutech, François Charpillet

Cooperation through communication in decentralized Markov games

Raghav Aras, Alain Dutech, François Charpillet

LORIA / INRIA-Lorraine, Campus Scientific, B.P. 239,
Cedex 54506 Vandœuvre-lès-Nancy, France.
emails: {aras@loria.fr, dutech@loria.fr, charpillet@loria.fr}

***Abstract*— In this paper, we present a comunication-integrated reinforcement-learning algorithm for a general-sum Markov game or MG played by independent, cooperative agents. The algorithm assumes that agents can communicate but do not know the purpose (the semantic) of doing so. We model agents that have different tasks, some of which may be commonly beneficial. The objective of the agents is to determine which are the commonly beneficial tasks, and learn a sequence of actions that achieves the common tasks. In other words, the agents play a multi-stage *coordination* game, of which they know niether the stage-wise payoff matrix nor the stage transition matrix. Our principal interest is in imposing realistic conditions of learning on the agents. Towards this end, we assume that they operate in a *strictly imperfect monitoring setting* wherein they do not observe one another's actions or rewards. A learning algorithm for a Markov game under this stricter condition of learning has not been proposed yet to our knowledge. We describe this Markov game with individual reward functions as a new formalism, *decentralized Markov game* or Dec-MG, a formalism borrowed from Dec-MDP (Markov decison process). For the communicatory aspect of the learning conditions, we propose a series of communication frameworks graduated in terms of facilitation of information exchange amongst the agents. We present results of testing our algorithm in a toy problem MG called a *total* guessing game.**

I. Introduction

A. Reinforcement learning

In reinforcement learning [1], an agent tries to learn the optimal way of doing a task. In effect, the agent tries to do this through maximizing a "reward" function, based on the reward signal that emanates from the environment. The model of the environment of the task is unknown to the agent. The model of the task is a Markov decision process (MDP) [2]. The agent collects experiences for the form $< state, action, reward >$, and progressively finds the optimal action for each state. A particular action executed in a state may be reinforced or not by the reward that the environment gives the agent for doing so. Thus the agent doesn't learn the model of the system, but learns *policies* of optimal behavior directly through experience. Q-learning [3] is a particularly interesting reinforcement learning algorithm because it allows the agent to build up the optimal policy while following a exploration-inclined sub-optimal policy (it is thus an "off-policy" algorithm). The Q-learning algorithm has been shown to converge to the optimal policy after the agent has collected an adequate number of experiences. For any given MDP, atleast one optimal reactive policy of the form $state \rightarrow action$, always exists [2]. It is termed reactive since the optimal action is dependent solely on current state, and not on history of past states and actions. Other notable reinforcement learning algorithms include the "on-policy" SARSA algorithm [1] and the gradient descent algorithm VAPS [4].

Our interest is in extending reinforcement learning for tasks controlled by more than one agent. Each agent acts independently of others, but for optimal control, it must coordinate its actions with those of other agents. The simplest extension of the MDP control problem to multiple agents is the Multiple MDP or MMDP framework [5]. However, the more general framework for multiagent learning which captures the notion on unequal rewards is the Markov game framework [6]. Making use of the Q-learning algorithm for parallel multi-agent learning has now a considerable body of work behind it ([6], [7], [8], [9], [10] to cite some examples). Since in these works, Q-learning is directly transplanted to the multi-agent setting, the conditions of learning are somewhat idealistic in that each agent is able to observe which actions other agents take and what rewards they receive. This assumption is understandable since the focus of learning in these works is *what to learn* and not really *how to learn* given the game-theoretic notion of "rationality" assumed of each agent.

B. Multi-agent learning conditions

Assuming omniscient conditions of learning makes the number of Q-values to learn exponential in the number of agents. An experience for each agent becomes of the form $< state, \overrightarrow{A}, \overrightarrow{R} >$, where the vectors are joint-actions and joint-rewards respectively. From the perspective of a multi-agent system it is more interesting to limit agents' knowledge about the workings of other agents. When agents don't observe, at any time t, the actions taken and the rewards received by other agents, such a condition constitutes a *strictly imperfect monitoring setting* [11]. However, limiting each agent's view of other agents' actions and rewards engenders the co-assumption that each agent has only "partial" view of the state. In other words, instead of knowing complete state information, the agent receives a state-dependent observation, from which it is required to deduce the actual state. The agent is no longer operating a MDP but rather a *partially observable* MDP or POMDP [12]. Under such a restriction, an agent cannot

Agent Actions	$Opera$	$Baseball$
$Opera$	**3, 6**	0, 0
$Baseball$	0, 0	**6, 3**

TABLE I

REWARD MATRIX FOR THE BATTLE OF THE SEXES GAME

Agent Actions	$Opera$	$Cinema$	$Baseball$
$Opera$	**3, 6**	0, 0	0, 0
$Cinema$	0, 0	**6, 6**	0, 0
$Baseball$	0, 0	0, 0	**6, 3**

TABLE II

REWARD MATRIX FOR A MODIFIED BATTLE OF THE SEXES GAME

form optimal reactive policies because of the state-observation aliasing; it must instead learn history-based policies [13]. Even when the model of the system is known, finding multi-agent optimal policies for POMDPs is an NEXP-complete problem [14]. A slightly complexity alleviating condition of multi-agent learning is partial observability with joint full-observability, wherein the pooled observations of all agents do give the actual state of the system.

Thus to summarize, we are interested in independents agents learning reactive polices for cooperation on a task under the two conditions,

- They operate in a strictly imperfect monitoring setting
- In each state, they have partial observability individually and full observability jointly

We propose a new formalism for describing multi-agent tasks that we target, that differs slightly from previous formulations. We call this new framework a **decentralized Markov game** or Dec-MG. It is similar to the Dec-MDP framework of [14] differing in the reward function; in Dec-MDP each agent receives the same reward, while in a Dec-MG this is not necessarily so.

C. Cooperation amongst agents

Our focus in this paper is on the modality of learning. We wish to approach multi-agent learning by imposing the above-stated learning conditions on a given class of Markov game problems. By doing so, we are specifying the general objective of the agents and proposing a learning mechanism which would teach the agents the right policies. So, our focus is on *how to learn*, rather than *what to learn*. In particular, we are interested in *coordination* Markov games. These have been studied in a perfect monitoring, fully-observable setting in [15]. Such games typically contain multiple Nash equilibria, and the problem is of aligning agents to choose one of them. A dilemma is created because not all agents might profit equally in all the equilibria (Table I). However, if the game provides a *Pareto-optimal* Nash equilibrium, satisfiability of all agents can be targeted. Attaining such an equilibrium is termed "optimal coordination" (Table II).

Hence the learning problem is two-fold, *(a)* detection of this equilibrium and *(b)* finding agent policies to achieve it. In a Markov game (or multi-stage game), played by n agents and with a state set S, the coordination condition is facilitated as follows. Each agent i has a set of desirable states called a goal set, $\Psi_i \subset S$. Attaining any state in the goal set (a goal state) gives the agent its maximum reward. For an agent, its goal states are identical in that they give the agent the same maximum reward. However, different agents have different goal sets. All goal states i.e., $(\Psi^{abs} = \Psi_1 \bigcup \Psi_2 \bigcup \ldots \bigcup \Psi_n)$ terminate the game. The condition that provides a Pareto-optimal Nash equilibrium is that $\Psi = (\Psi_1 \bigcap \Psi_2 \bigcap \ldots \bigcap \Psi_n)$ is non-empty. This set is the common goal set, and its elements, common goals. All goals that are not common are private goals. Agent i's private goals are $\Psi_i - \Psi$. Thus, the objective of the agents is to learn policies that attain common goals while avoiding private goals. Thus we add another learning condition,

- Atleast one Pareto-optimal Nash equilibrium exists for (each stage of) the Markov game.

D. Inter-agent communication

With the strict conditions of learning as stated above, our interest is in studying how agents can learn to cooperate if communication amongst them were possible. In other words, how can agents learn to use a communication faculty, if it were made available, in order to learn coordination of actions for the "real" task (as opposed to the incidental task of learning to communicate). When information-exchange is possible amongst agents, several issues arise and can be addressed; we point to the following:

- Do agents know the purpose of communication?
- Do the agents use the same language (or protocol) to communicate?
- What do the agents communicate?

In this paper, we respond to these issues as follows: we suppose that agents do not know the purpose of communication *a priori* (see [16] for a counter-example), agents use the same language to communicate ([17] assumes agents use different languages) and finally, we contend that agents transmit "null" impulses rather than content-based messages as in [18] or [19]. We thus propose three communication frameworks based on these features where each framework is more sophisticated than the previous one. We integrate these frameworks with the Q-learning algorithm and propose a new action-value update rule to take into account communication received and sent. This algorithm is meant for multi-agent learning of coordination for a given Markov game under the said learning conditions.

The rest of the paper is organized as follows. In section II, we describe the Dec-Markov game (Dec-MG) framework alongwith related frameworks as well as the basic Q-learning algorithm. In section III, we state our communication frameworks, and in section IV, we present their integration with Dec-MG. Section V contains our communication-based multi-agent algorithm, followed by results of experiments and testing in section VI. We close with a discussion on this work in section VII.

II. TASK FRAMEWORK

A. MDP and Q-learning

A Markov decision process is a four-tuple $< S, A, T, R >$.

- S, is a set of states.
- A, is a set of actions.
- $T(s, a, s')$, is the state transition function, which gives the probability of moving to state s' from state s on taking action a, $s, s' \in S$, $a \in A$.
- $R(s, a)$, is the reward function, which gives the reward (a real number) of taking action a in state s, $s \in S$, $a \in A$.

In reinforcement learning, the agent doesn't know T and R. When using Q-learning [3], the agent learns Q-values denoted as Q(s, a), which represent the long-term value of taking action a in state s. The agent gathers experiences of the form $< s, a, r, s' >$, and updates the concerned Q-value as follows:

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha(r + \gamma max_{a'} Q(s',a')) \quad (1)$$

where α is the learning rate and γ, the discount-factor. When Q-learning is used for multi-agent learning, the equation above changes to (for each agent i),

$$Q_i(s, \overrightarrow{A}) \leftarrow (1-\alpha)Q_i(s, \overrightarrow{A}) + \alpha(r + \gamma operator_{\overrightarrow{A'}} Q_i(s', \overrightarrow{A'})) \quad (2)$$

$\overrightarrow{A}$ is the joint-action. Since the updates are done independently by each agent i, the max operator cannot be employed. Some other, appropriate operator must be identified and used. For instance, in [7], the NashQ operator based on Nash equilibrium, is used, while in [9], an operator based on correlated equilibrium is used. In our approach we assume that the joint-action is not observable, and hence we are interested in the first form of Q-learning.

Algorithm 1 Single agent Q-learning

$\forall s \in S$, $\forall a \in A$, $Q(s, a) \leftarrow 0$.
Initialize system state to s.
(for each episode)
repeat
  Choose action $a \in A$ using a policy derived from Q ($\epsilon - greedy$, for example).
  Execute chosen action a.
  Observe reward r and next state s'.
  $Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha(r + \gamma max_{a'} Q(s',a'))$
  $s \leftarrow s'$
until s is terminal
Output: The agent constructs a reactive policy such that $\forall s, \pi(s) \leftarrow max_a Q(s,a)$

B. Decentralized Markov game (Dec-MG) for multi-agent control

In this section, we describe our framework for modeling multi-agent control of a task. We adapt the decentralized Markov decision process (Dec-MDP) framework described in [14] toward this end. We make one notable change to it: **agents have individual reward functions rather than one common reward function**. We call this different framework as a decentralized Markov game or Dec-MG.

- n is the number of agents involved.
- S is the set of system states.
- $A = A_1 \times A_2 \times \ldots \times A_n$ is the set of joint-actions. A_i is the set of agent i's actions.
- $R_i(s, a)$ is a rational representing the reward agent i obtains if agents take joint-action a in state s. $a \in A$, $s \in S$. There are thus n reward functions.
- $T(s, a_1, \ldots, a_n, s')$, the state transition function gives the probability of transitioning to state s' from state s if joint action $< a_1, \ldots, a_n >$ is taken. $s, s' \in S$. We are considering a deterministic system, hence T's values are from {0, 1}.
- Ω is the set of observations.
- $O(a_1, \ldots, a_n, o_1, \ldots, o_n, s)$, the observation function gives the probability of agent i receiving observation $o_i \in \Omega$, if the agents take joint action $< a_1, \ldots, a_n >$ and the system moves to state $s \in S$ as a result.
- There exists a mapping $J(o_1, \ldots, o_n)$, for each tuple of n observations, one observation per agent, to a unique state $s \in S$. Joint full-observability implies that at any time, if agents pool their observations, they can obtain the system state.

Note: If the observation received in each state is unique, then the Dec-MG reduces to a Markov game or MG [6] and it is defined as above.

In reinforcement learning, as stated before, the agent gathers experiences of a given form during learning. Usually, the experience gathering is divided into *episodes*. An episode ends when the agent enters a goal state (also called an absorbing state). The Q-value updates made during an episode are used in the next episode. The agents' goals are implicit through their reward functions. Just to be precise, we make the following designations:

- $\Psi_i (\subset S)$ is agent i's goal set.
- $\Psi = (\Psi_1 \bigcap \Psi_2 \bigcap \ldots \bigcap \Psi_n)$ is the common goal set.
- $\Psi^{Abs} = (\Psi_1 \bigcup \Psi_2 \bigcup \ldots \bigcup \Psi_n)$ is the absorbing state set.
- $\Psi, \Psi^{Abs} \neq \emptyset$ (this expresses the condition for permitting cooperation)

III. THE COMMUNICATION FRAMEWORKS

In this section, we describe a graduated series of communication frameworks. The problem with incorporating an express communication framework in a Markov game is that such an addition can be objected to as being purely prescriptive in nature. In other words, with the defined communication facility at their disposal, the agents would play a transformed, potentially simpler Markov game that might not address all the issues raised by the original game. It is thus important to remain limited in defining the communication framework. The frameworks that we propose are ordered by their increasing informativeness. Each framework proposed is seen as logical addendum to its predecessor. At the zeroth level is the Dec-MG itself as described in the previous section.

A. Nomenclature and Description of Gradation

Below, we describe three concepts on which we base our frameworks, namely, null message, sender and directedness:

- Null message. The simplest idea of communication is that a message $< sender, message - content >$ is received by an agent. However, even simpler than this form would be content-less messages. This is form we consider. We term messages of this sort as null messages. Null messages allow us to avoid any arbitrary normativeness of message-content and hence of the communication framework. Null messages can be represented just by $< sender >$. For the rest of the paper, we refer to null messages as simply messages.
- Sender. This represents the identity of the message-sending agent. Agents are uniquely numbered. If there are more than one agents that send a null message (to some agent) then 'sender' is a tuple containing each sending agent's identity (for e.g., $< 4, 1, 3, \ldots >$). Sender also serves another purpose. Suppose our agents are nameless (i.e., without identity numbers) and we are interested in indicating that a certain agent receives a message. Then sender could take just binary values, 0 indicating no message received and 1 indicating otherwise.
- Directedness. This is the degree of informativeness the communication framework supports. We consider three values of directedness, 1, 2 and 3. These categories of directedness, in fact, define the three communication frameworks that we study in this paper. They will be called frameworks 1, 2 and 3 accordingly, or F1, F2 and F3 in short. The Dec-MG without any communication framework is F0.
  - F1 *Broadcast, anonymous, idempotent messages*. All agents receive a sent message (broadcast). Agents cannot identify themselves while sending messages (anonymous), and two or more messages sent at the same time to some agent, appear to be just one message (idempotent). Sender takes values from $\{0,1\}$.
  - F2 *Broadcast, identifiable messages*. All agents receive a sent message (broadcast). Messages received contain the id (or ids) of sending agents (identifiable). Once messages can be identified, the property of idem potency disappears. Sender's value can be a combination of any size from 0 to n - 1, of the agents ids.
  - F3 *Non-broadcast, identifiable messages*. Agents can choose the recipient agents of their messages (non-broadcast), and messages received contain the ids of sending agents (identifiable). Sender's values are same as in F2.

IV. Communication-extended Dec-MG framework

We now describe what additions to the Dec-MG framework the three communication frameworks F1, F2 and F3 entail. We note that the functions T, J and $R_i s$ do not change in any of the additions given below. We shall see that F1 and F2 have the same action set, while F2 and F3 have the same observation set. F3 has the most *personalized* observation function.

A. Dec-MG with F1

F1 messages are broadcast, anonymous and idempotent.

- $A_i^F = A_i \times \{0,1\}$ is the agent i's augmented action set. It is the cross-product of the action set with a binary set, indicating that each action in A_i can now be exercised either with (1) or without (0) the sending of a message. Actions are thus tuples of the form $< a, x >$, x being 0 or 1.
- $A^F = A_1^F \times A_2^F \times \ldots \times A_n^F$ is the joint-action set.
- $\Omega_F = \Omega \times \{0,1\}$ is the enlarged observation set from which all agents receive observations. Under F1, any observation in $o \in \Omega$ can (1) or cannot (0) be accompanied by a message. Observations are thus tuples of the form $< o, y >$, y being 0 or 1.
- $O^F(< a_1, x_1 >, \ldots, < a_n, x_n >, < o_1, y >, \ldots, < o_n, y >, s)$ is the observation function that gives observations $o_1, \ldots, o_n$ according to O when agents take actions $a_1, \ldots, a_n$, and makes $y = 1$ iff $\exists x_i = 1$, else y equals 0. This indicates that if any agent sends a message with an action a (by setting x to 1), all agents receive the message (y equals 1) along with the original observation as given by O. The system moves to state $s \in S$ according to O as before.

B. Dec-MG with F2

F2 messages are broadcast and identifiable.

- Agents are numbered from 1 to n.
- $A_i^F s, A^F$ are the same as in F1.
- $\Omega^F = \Omega \times (\bigcup_{i=1}^{n} C_i^n + \{0\})$. Since messages are identifiable, each observation can be accompanied a tuple of up to size n, identifying the senders of the messages. These tuples form a union of set of combinations, from sizes 0 to n, of agent ids. $\{0\}$ indicates that no message was received. Observations are thus tuples of the form $< o, \overrightarrow{y} >$, $\overrightarrow{y}$ being a tuple of any size from 1 to n.
- $O^F(< a_1, x_1 >, \ldots, < a_n, x_n >, < o_1, \overrightarrow{y} >, \ldots, < o_n, \overrightarrow{y} >, s)$ is the observation function that gives observations $o_1, \ldots, o_n$ according to O when agents take actions $a_1, \ldots, a_n$, and $\overrightarrow{y}$ is the tuple $\bigcup_{i=1}^{n} i$ iff $x_i = 1$. All agents receive a tuple of senders' identities (the tuple $\overrightarrow{y}$) along with the original observation as given by O. The system moves to state $s \in S$ according to O as before.

C. Dec-MG with F3

F3 messages are non-broadcast and identifiable.

- Agents are numbered from 1 to n.
- $A_i^F = A_i \times (\bigcup_{i=1}^{n} C_i^n + \{0\})$ is the agent i's augmented action set. It is the cross-product of the action set with a union of sets that contain combinations of different sizes, of agents ids. Actions are thus tuples of the form $< a, \overrightarrow{x} >$, $\overrightarrow{x}$ being a tuple of any size from 1 to n,

containing the ids of agents to which the message is addressed. $\{0\}$ indicates that no message is being sent.
- $A^F = A_1^F \times A_2^F \times \ldots \times A_n^F$ is the joint-action set.
- Ω^F is as in F2.
- O^F is as in F2, except that the value of tuple $\overrightarrow{y}$ received with each observation may be different for different agents. Each agent's $\overrightarrow{y}$ will contain ids of only those agents who have sent it a message. Thus messages are not broadcast.

V. The learning algorithm

In this section, we describe a Q-learning based algorithm for the Dec-MG under a communication framework. It is appellated *Reinforcement by Messages*. It is applicable to all the discussed communication frameworks. As we have stated earlier, we are interested in finding reactive policies of the form $\pi : \Omega \rightarrow A$ (i.e., the optimal action based only on current observation). With our communication framework in place, we wish to generate optimal policies of the form $\pi_i : (\Omega^F) \rightarrow (A^F)$. Such a policy tells agent i which action to take, whom to send a message to on receiving an observation or an observation with a message. Thus agents learn Q-values of the form $Q_i(< o, y >, < a, x >)$, where y, x could be single values or tuples depending on the framework. Agents are independent learners, hence each agents uses and updates its own Q-values. In the algorithm,

- λ_i denotes an observation-message tuple $< o, y >$.
- μ_i denotes an action-message tuple $< a, x >$.
- X_i denotes the number of agents to which messages were sent; in the case of F1 and F2, this is n. For F3, this is the length of the tuple $\overrightarrow{x}$.
- Y_i denotes the number of agents from which messages were received; in the case of F1, this is n. For F2 and F3 this is the length of the tuple $\overrightarrow{y}$.
- α is the learning rate and γ is the discount factor.

A. Central principle of the algorithm

The algorithm's main difference with Q-learning is in the way it utilizes the obtained rewards. It works as follows: Each agent puts the reward received from the environment in two components. The first component is the *negative* of the environment reward. Reinforcing actions by negating environment rewards, just in itself, would result in an agent learning to avoid all goals. However, each agent doesn't wish to avoid all goals, merely those that are not common. To seek out the common goals, the agents calculate a second, compensatory component. This is an *amplification* of the reward received from the environment by a factor of the number of agents contacted (either through sending or receiving messages). One way of interpreting such a form of compensation is that, agents are seeking to confirm from one another, the worth of any individual action. One might say that the purpose of communication is implicit in the reward function. Thus the compensation for the negative reward is sought in the communication framework. The idea is the amplification will completely *offset* the earlier negativeness. In this way, agents calculate a virtual reward as a function of the actual reward received (This is similar to the principle of reward interpretation [20], wherein virtual rewards are supplied, in addition to actual ones, to speed up learning).

Algorithm 2 Reinforcement by Messages

for $i = 1$ to n **do**
  $\forall \lambda \in \Omega^F$, $\forall \mu \in A^F$, $Q_i(\lambda, \mu) \leftarrow 0$.
end for
(for each episode)
System state is initialized to s
$\forall i$, agent i observation is initialized to λ_i according to J and s.
repeat
  $\forall i$, Agent i chooses action $\mu_i \in A_i^F$ using a policy derived from Q_i ($\epsilon - greedy$, for example).
  $\forall i$, Agent i executes chosen action μ_i.
  System state s is updated to $s^{'}$ according to $T(s, \mu_1(a), \ldots, \mu_n(a), s^{'})$
  Agent observations are given according to $O^F(\mu_1(a), \ldots, \mu_n(a), \lambda_1^{'}, \ldots, \lambda_n^{'}, s^{'})$.
  for $i = 1$ to n **do**
    Agent i observes reward r_i and gets observation $\lambda_i^{'}$.
    $r_i^{priv} \leftarrow -r_i$.
    $r_i^{mess} \leftarrow r_i(X_i + Y_i)$.
    $Q_i(\lambda_i, \mu_i) \leftarrow (1-\alpha)Q_i(\lambda_i, \mu_i) + \alpha[r_i^{priv} + r_i^{mess} + \gamma max_{\mu_i^{'}} Q_i(\lambda_i^{'}, \mu_i^{'})]$.
  end for
until $s \in \Psi^{Abs}$
Output: Each agent i constructs a reactive policy such that $\forall \lambda, \pi_i(\lambda) \leftarrow max_\mu Q_i(\lambda, \mu)$

Following this logic, when all agents are in any common goal state, each agent will prefer to send a message to all other agents to extract maximum compensation. If such is the case, each agent will also receive messages from all agents, further adding to its compensation. This is the idea behind reinforcement by messages. In a private-goal state, an agent may not, in general, receive messages, and thus will obtain less reward. We contend that the algorithm is capable of making agents prefer common goals over private goals, but it cannot cope with the related objective i.e., making agents prefer non goals over private goals. The second objective would permit the game to continue, and increase the chance of agents finding a common goal.

VI. Experiments

A. Testbed problem

For testing our approach, we have formulated a simple *total* guessing game that simulates a Dec-MG in a strictly imperfect monitoring setting. Each agent contributes a number to a total. Some totals give high reward, while others give low reward. Some of the high rewarding totals are common to all agents. However, agents don't know the reward function, and hence they are unaware *a priori* of the good totals (thus in effect,

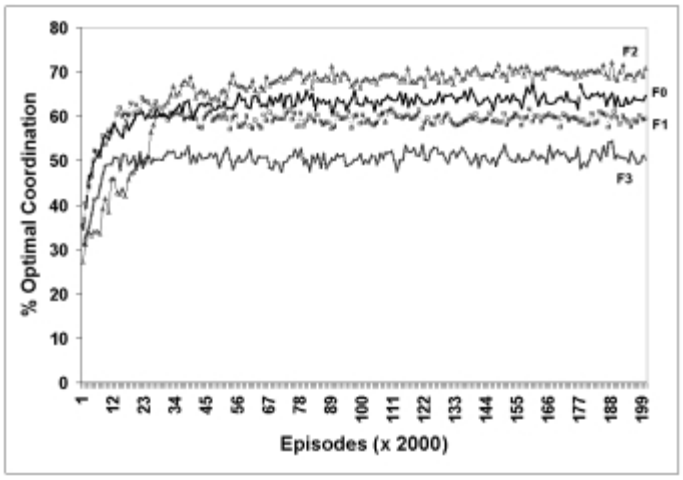

Fig. 1. Percent optimal coordination (four agents)

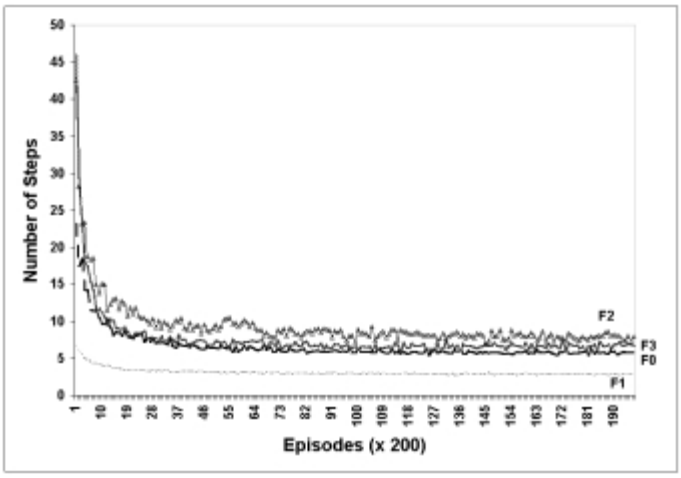

Fig. 2. Average episode length (four agents)

the reward function is defined over observations, rather than states). Moreover, agents cannot see each other's contributions, only the total. An agent can reduce or increase its contribution (within a range). Agents thus keep changing their contributions until the total satisfies at least one agent, at which point the game terminates. The objective of the agents is to learn to attain a total that satisfies all the agents. In this game,

- State of the system is the configuration of contributions of all the agents.
- Totals correspond to observations.
- A, an integer, is the maximum possible increase or decrease in an agent's contributions, and hence an agent's actions are integers from the interval $[-A, A]$.

We conducted simulations of this game under four conditions:

- F0 with full observability of state. It implies in fact a MG. We still assumed that agents do not observe each others actions or rewards. The algorithm we use for this condition is single agent Q-learning.
- Dec-MG with F1, F2 and F3 in an imperfectly monitoring setting using the Reinforcement by Messages algorithm.

Each game involved 4 agents and 400000 episodes. Goal totals gave a reward of 10 while other totals reward -2.

B. Results

Fig. 1 shows the results for a game where there was 1 common goal, and 1 private goal per agent. Using F1, agents attain optimal coordination comparable to F0. F2 outperforms all. F2 gives better results that F1 since the update rule uses the exact number of senders rather than an arbitrary number. F2 gives better results that F3 (in which the update rule has more information) because in F3, an agent can claim more reward by sending a message than in F2 or F1. This implies that an agent may still find private goals attractive (by sending "useless" messages) and pursue them resulting in lesser optimal coordination.

If we make the the reward function more precise over the states, i.e., the high reward is given not just for a particular total, but its constituent contributions too, then under all the four conditions, optimal (100 percent) coordination is reached after just a small number of episodes. In Fig 2., the number of steps taken per episode to achieve optimal coordination when rewards are state-based, is shown. As can be remarked, coordination is achieved much quicker since the rewards given are more consistent, and agents can update their actions with more surety. We notice also that F2 and F3 the most number of steps to achieve optimal coordination. This is understandable given that agents have a much higher number of actions (basic actions + communication actions) over which to coordinate. Agents could be said to be waiting for the right message from the right agent. The fact that F1 gives the fastest convergence indicates that our reward interpretation rule is valid and effective. F1 has double the number of actions than F0.

VII. Conclusion and Future Work

In this paper, we have presented a new approach to integrating communication into Markov games. As compared to previous endeavours of such nature, our approach is less ad-hoc in that niether do we attach any *semantic* to the message (such as exchange of state or observation information, or proposed action information) nor do we propose explicit rules indicating *why* the agent must communicate. Agents learn the virtue of communicating (as indeed of remaining silent) based on the basic reward that is inbuilt into the task they are supposed to do together. We have introduced a new reward interpretation rule that allows agents to test out all combinations of actions and communication acts, and choose the optimal, even when they do not observe the actions of other agents. We have proposed three rudimentary communication frameworks, each one of them a logical addendum to its predecessor, and proposed a new learning algorithm based on Q-learning, that teaches agents when to communicate, with whom and with what basic action. We have shown that for coordination Markov games, our algorithm performs better than independent learner Q-learning with full observability in terms of percentage optimal

coordination and episode length. We haven't included results from joint-action Q-learning here, but we note that joint-action learners perform only marginally better than independent-learners. See [15], for results on single-stage games.

In the future, we intend to continue this line of thought in studying related phenomena such as coalition formations, framework preference (with regards to attached costs), competitive games (where agents are non-cooperative), and more sophisticated reward interpretation rules.

REFERENCES

[1] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

[2] M. Puterman, *Markov Decision Processes*. New York: John Wiley and Sons, 1994.

[3] C. Watkins and P. Dayan, "Q-learning," *Proceedings of the Eighth International Conference on Machine Learning (ICML)*, vol. 8(3), pp. 279–292, 1992.

[4] L. Baird and A. Moore, "Reinforcement learning through gradient descent," *Advances in Neural Information Processing Systems (NIPS)*, 1999.

[5] C. Boutilier, "Sequential optimality and coordination in multiagent systems," *Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*, p. 478485, 1999.

[6] M. Littman, "Markov games as a framework for multi-agent reinforcement learning," *Proceedings of the Eleventh International Conference on Machine Learning (ICML)*, pp. 157–163, 1994.

[7] J. Hu and M. Wellman, "Multiagent reinforcement learning: Theoretical framework and an algorithm," *Machine Learning: Proceedings of the Fifteenth International Conference on Machine Learning (ICML)*, pp. 242–250, 1998.

[8] N. Suematsu and A. Hayashi, "A multi-agent reinforcement learning algorithm using extended optimal response," *Proceedings of the First International Joint Conference on Automated Agents and Multi-Agent Systems (AAMAS)*, pp. 370 – 377, 2002.

[9] A. Greenwald and K. Hall, "Correlated-q learning," *In Proceedings of the Twentieth International Conference on Machine Learning (ICML)*, p. 242249, 2003.

[10] M. Weinberg and J. S. Rosenschein, "Best response multiagent learning in non-stationary environments," *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2004.

[11] R. Brafman and M. Tennenholtz, "Efficient learning equilibrium," *Technical Report 02-06, Dept. of Computer Science, Ben-Gurion University*, 2002.

[12] L. Kaelbling, M. Littman, and A. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. Vol. 101, 1998.

[13] S. Singh, T. Jaakkola, and M. Jordan, "Learning without state-estimation in partially observable markovian decision processes," *Proceedings of the Eleventh International Conference on Machine Learning (ICML)*, pp. 284–292, 1994.

[14] D. Bernstein, R. Givan, N. Immerman, and S. Zilberstein, "The complexity of decentralized control of markov decision processes," *Mathematics of Operations Research*, vol. 27(4), pp. 819–840, November 2002.

[15] C. Claus and C. Boutilier, "The dynamics of reinforcement learning in cooperative multiagent systems," *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI)*, pp. 746–752, 1998.

[16] D. Szer and F. Charpillet, "Coordination through mutual notification in cooperative multiagent reinforcement learning," *Proceedings of the Third International Joint Conference on Automated Agents and Multi-Agent Systems (AAMAS)*, 2004.

[17] C. Goldman, M. Allen, and S. Zilberstein, "Decentralized language learning through acting," *Proceedings of the Third International Joint Conference on Automated Agents and Multi-Agent Systems (AAMAS)*, 2004.

[18] C. Goldman and S. Zilberstein, "Optimizing information exchange in cooperative multi-agent systems," *Proceedings of the Second International Joint Conference on Automated Agents and Multi-Agent Systems (AAMAS)*, 2003.

[19] S. Coradeschi, L. Karlsson, P. Stone, G. Kraetzschmar, T. Balch, and M. Asada, "Overview of robocup-99," *AI Magazine*, vol. 21, 2000.

[20] A. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations," *Proceedings of the Sixteenth International Conference on Machine Learning (ICML)*, 1999.