On the Study of Cooperative Multi-Agent Policy Gradient

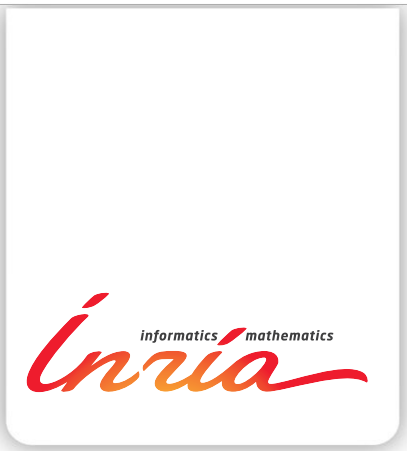
Guillaume Bono, Jilles Steeve Dibangoye, Laëtitia Matignon, Florian Pereyron, Olivier Simonin

On the Study of Cooperative Multi-Agent Policy Gradient

Guillaume Bono, Jilles Steeve Dibangoye, Laëtitia Matignon , Florian Pereyron, Olivier Simonin

RESEARCH REPORT
N° 9188
June 2018
Project-Team Chroma

ISSN 0249-6399 ISRN INRIA/RR--9188--FR+ENG



On the Study of Cooperative Multi-Agent Policy Gradient

Guillaume Bono*, Jilles Steeve Dibangoye*, Laëtitia Matignon†
, Florian Pereyron‡, Olivier Simonin

Project-Team Chroma

Research Report n° 9188 — June 2018 — 22 pages

Abstract: Reinforcement Learning (RL) for decentralized partially observable Markov decision processes (Dec-POMDPs) is lagging behind the spectacular breakthroughs of single-agent RL. That is because assumptions that hold in single-agent settings are often obsolete in decentralized multi-agent systems. To tackle this issue, we investigate the foundations of policy gradient methods within the centralized training for decentralized control (CTDC) paradigm. In this paradigm, learning can be accomplished in a centralized manner while execution can still be independent. Using this insight, we establish policy gradient theorem and compatible function approximations for decentralized multi-agent systems. Resulting actor-critic methods preserve the decentralized control at the execution phase, but can also estimate the policy gradient from collective experiences guided by a centralized critic at the training phase. Experiments demonstrate our policy gradient methods compare favorably against standard RL techniques in benchmarks from the literature.

Key-words: Decentralized Control, Partial Observable Markov Decision Processes, Multi-Agent Systems, Actor Critic.

* Univ Lyon, INSA Lyon, INRIA, CITI, F-69621 Villeurbanne, France
† Univ Lyon, Université Lyon 1, LIRIS, CNRS, UMR5205, Villeurbanne, F-69622, France
‡ Volvo Group, Advanced Technology and Research

RESEARCH CENTRE
GRENOBLE – RHÔNE-ALPES

Inovallée
655 avenue de l'Europe Montbonnot
38334 Saint Ismier Cedex

Sur l'étude du Gradient de la Politique pour les Systèmes Multi-Agents Collaboratifs

Résumé : L'apprentissage par renforcement (RL) pour les processus décisionnels de Markov partiellement observables décentralisés (Dec-POMDPs) accuse un certain retard par rapport aux progrès spectaculaires du RL mono-agent. Ceci s'explique en partie par un certain nombre d'hypothèses valables dans le cadre mono-agent, mais invalides dans les systèmes multi-agents. Pour combler ce retard, nous explorons les fondements mathématiques des méthodes par ascension du gradient de la politique dans le paradigme de l'entraînement centralisé pour un contrôle décentralisé (CTDC). Dans ce paradigme, l'apprentissage peut avoir lieu de façon centralisée tout en gardant la contrainte d'une exécution décentralisée. En partant de cette intuition, nous établissons dans ce document une extension multi-agents du théorème du gradient de la politique et du théorème de compatibilité des fonctions d'approximation de la valeur. Nous en tirons des méthodes " acteur critique " (AC) qui parviennent (i) à estimer le gradient de la politique à partir d'expériences collectives mais aussi (ii) à préserver le contrôle décentralisé du système à l'exécution. Nos expérimentations montrent que nos méthodes ne souffrent pas de la comparaison avec les techniques standard en RL sur un ensemble de bancs de test de la littérature.

Mots-clés : Contrôle décentralisé et stochastique, Processus Décisionnel de Markov Partiellement Observable, Systèmes Multi-Agents, Méthodes Acteur Critique

Contents

1 Introduction

The past years have seen significant breakthroughs in agents that can gain abilities through interactions with the environment [18, 19], thus promising spectacular advances in the society and the industry. These advances are partly due to single-agent (deep) RL algorithms. That is a learning scheme in which the agent describes its world as a Markov decision process (MDP), other agents being part of that world, and assumptions at both learning and execution phases being identical [25]. In this setting, policy gradient and (natural) actor-critic variants demonstrated impressive results with strong convergence guarantees [1, 6, 14, 26]. These methods directly search in the space of parameterized policies of interest, adjusting the parameters in the direction of the policy gradient. Unfortunately, extensions to cooperative multi-agent systems have restricted attention to either independent learners [22, 29] or multi-agent systems with common knowledge about the world [32], which are essentially single-agent systems.

In this paper, we instead consider cooperative multi-agent settings where we accomplished learning in a centralized manner, but execution must be independent. This paradigm allows us to break the independence assumption in decentralized multi-agent systems but only during the training phase, while still preserving the ability to meet it during the execution phase. In many real-world cooperative multi-agent systems, conditions at the training phase do not need to be as strict as those at the execution phase. During rehearsal, actors can read the script, take breaks, or receive feedback from the director, but none of these will be possible during the show [16]. To win matches, a soccer coach develops (before the game) tactics players will apply during the game. So, it is natural to wonder whether the policy gradient approach in such a paradigm could be as successful as for the single-agent learning paradigm.

The CTDC paradigm has been successfully applied in planning methods for Dec-POMDPs, *i.e.,* a framework of choice for sequential decision making by a team of cooperative agents [4, 7, 13, 21, 27]. In the literature of game theory, Dec-POMDPs are partially observable stochastic games with identical payoffs. They subsume many other collaborative multi-agent models, including multi-agent MDPs [5]; stochastic games with identical payoffs [24]; to cite a few. The critical assumption that makes Dec-POMDPs significantly different from MDPs holds only at the execution phase: agents can neither see the real state of the world nor explicitly communicate with one another their noisy observations. Nonetheless, agents can share their local information at the training phase, as long as they act at the execution phase based solely on their individual experience. Perhaps surprisingly, this insight has been neglected so far, explaining the formal treatment of CTDC received little attention from the RL community [16]. When this centralized training takes place in a simulator or a laboratory, one can exploit information that may not be available at the execution time, *e.g.,* hidden states, local information of the other agents, etc. Recent work in the (deep) multi-agent RL community builds upon this paradigm to design domain-specific methods [11, 12, 17], but the theoretical foundations of decentralized multi-agent RL are still in their infancy.

This paper investigates the theoretical foundations of policy gradient methods within the CTDC paradigm. In this paradigm, among policy gradient algorithms, actor-critic methods can train multiple independent actors (or policies) guided by a centralized critic (Q-value function) [11]. Methods of this family differ only through how they represent and maintain the centralized critic. The primary result of this article generalizes the policy gradient theorem and compatible function approximations from (PO)MDPs to Dec-POMDPs. In particular, these results show the compatible centralized critic is the sum of individual critics, each of which is linear in the "features" of its corresponding individual policy. Even more interestingly, we derive update rules adjusting individual critics in the direction of the gradient of the centralized critic. Experiments demonstrate our policy gradient methods compare favorably against techniques from standard

RL paradigms in benchmarks from the literature.

We organized the rest of this paper as follows. Section 2 gives formal definitions of POMDPs and Dec-POMDPs along with useful properties. In section 3, we review the policy gradient methods for POMDPs, then pursue the review for cooperative multi-agent settings in section 4. Section 5 develops the theoretical foundations of policy gradient methods for Dec-POMDPs and derives the algorithms. Finally, we present empirical results in section 6.

2 Backgrounds

2.1 Partially observable Markov decision processes

Consider a (centralized coordinator) agent facing the problem of influencing the behavior of a POMDP as it evolves through time. This setting often serves to formalize cooperative multi-agent systems, where all agents can explicitly and instantaneously communicate with one another their noisy observations.

Definition 1. *Let $M_1 \doteq (\mathcal{X}, \mathcal{U}, \mathcal{Z}, p, R, T, s_0, \gamma)$ be a POMDP, where X_t, U_t, Z_t and R_t are random variables taking there values in $\mathcal{X}$, $\mathcal{U}$, $\mathcal{Z}$ and $\mathbb{R}$, and representing states of the environment, controls the agent took, observations and reward signals it received at time step $t = 0, 1, \dots, T$, respectively. State transition and observation probabilities $p(x', z'|x, u) \doteq \mathbb{P}(X_{t+1} = x', Z_{t+1} = z'|X_t = x, U_t = u)$ characterize the world dynamics. $R(x, u) \doteq \mathbb{E}[R_{t+1}|X_t = x, U_t = u]$ is the expected immediate reward. Quantities s_0 and $\gamma \in [0, 1]$ define the initial state distribution and the discount factor.*

We call t^{th} history, $o_t \doteq (o_{t-1}, u_{t-1}, z_t)$ where $o_0 \doteq \emptyset$, a sequence of controls and observations the agent experienced up to time step $t = 0, 1, \dots, T$. We denote $\mathcal{O}_t$ the set of histories of the agent might experience up to time step t.

Definition 2. *The agent selects control u_t through time using a parametrized policy $\pi \doteq (a_0, a_1, \dots, a_T)$, where $a_t(u_t|o_t) \doteq \mathbb{P}_{\theta_t}(u_t|o_t)$ denotes the decision rule at time step $t = 0, 1, \dots, T$, with parameter vector $\theta_t \in \mathbb{R}^{\ell_t}$ where $\ell_t \ll |\mathcal{O}_t|$.*

In practice, we represent policies using a deep neural network; a finite-state controller; or a linear approximation architecture, *e.g.,* Gibbs. Such policy representations rely on different (possibly lossy) descriptions of histories, called internal states. It is worth noticing that when available, one can use p to calculate a unique form of internal-states, called *beliefs*, which are sufficient statistics of histories [2]. If we let $b^o \doteq \mathbb{P}(X_t|O_t = o)$ be the current belief induced by history o, with initial belief $b^\emptyset \doteq s_0$; then, the next belief after taking control $u \in \mathcal{U}$ and receiving observation $z' \in \mathcal{Z}$ is: $\forall x' \in \mathcal{X}$,

$$b^{o,u,z'}(x') \doteq \mathbb{P}\big(X_{t+1} = x'|O_{t+1} = (o, u, z')\big) \propto \sum_{x \in \mathcal{X}} p(x', z'|x, u) b^o(x)$$

Hence, using beliefs instead of histories in the description of policies preserves the ability to act optimally, while significantly reducing the memory requirement. Doing so makes it possible to restrict attention to stationary policies, which are particularly useful for infinite-horizon settings, *i.e.,* $T = \infty$. Policy π is said to be stationary if $a_0 = a_1 = \ldots = a$ and $\theta_0 = \theta_1 = \ldots = \theta$; otherwise, it is non-stationary.

Through interactions with the environment under policy π, the agent generates a trajectory of rewards, observations, controls and states $\omega_{t:T} \doteq (r_{t:T}, x_{t:T}, z_{t:T}, u_{t:T})$. Each trajectory produces

return $\mathfrak{R}(\omega_{t:T}) \doteq \gamma^0 r_t + \cdots + \gamma^{T-t} r_T$. Policies of interest are those that achieve the highest expected return starting at s_0

$$J(s_0; \theta_{0:T}) \doteq \mathbb{E}_{\pi, M_1}[\mathfrak{R}(\Omega_{0:T})] = \int \mathbb{P}_{\pi, M_1}(\omega_{0:T}) \mathfrak{R}(\omega_{0:T}) \mathrm{d}\omega_{0:T} \tag{1}$$

where $\mathbb{P}_{\pi, M_1}(\omega_{0:T})$ denotes the probability of generating trajectory $\omega_{0:T}$ under π. Finding the best way for the agent to influence M_1 consists in finding parameter vector $\theta^*_{0:T}$ that satisfies: $\theta^*_{0:T} \in \arg\max_{\theta_{0:T}} J(s_0; \theta_{0:T})$.

It will prove useful to break the performance under policy π into pieces to exploit the underlying structure —*i.e.,* the performance of π from time step t onward depend on earlier controls only through the current states and histories. To this end, the following defines value, Q-value and advantage functions under π. The Q-value functions under π is given by:

$$Q^\pi_t : (x, o, u) \mapsto \mathbb{E}_{\pi, M_1}[\mathfrak{R}(\Omega_{t:T}) | X_t = x, O_t = o, U_t = u], \forall t \tag{2}$$

where $Q^\pi_t(x, o, u)$ denotes the expected return of executing u starting in x and o at time step t and then following policy π from time step $t+1$ onward. The value functions under π is given by:

$$V^\pi_t : (x, o) \mapsto \mathbb{E}_{a_t}[Q^\pi_t(x, o, U_t)], \forall t \tag{3}$$

where $V^\pi_t(x, o)$ denotes the expected return of following policy π from time step t onward, starting in x and o. Finally, the advantage functions under π is given by:

$$A^\pi_t : (x, o, u) \mapsto Q^\pi_t(x, o, u) - V^\pi_t(x, o), \forall t \tag{4}$$

where $A^\pi_t(x, o, u)$ denotes the relative advantage of executing u starting in x and o at time step t and then the following policy π from time step $t+1$ onward. The nice property of these functions is that they satisfy certain recursions.

Lemma 1 (Bellman equations [3])**.** *Q-value functions under π satisfy the following recursion: $\forall t = 0, 1, \ldots, T$, $\forall x \in \mathcal{X}, o \in \mathcal{O}_t, u \in \mathcal{U}$,*

$$Q^\pi_t(x, o, u) = R(x, u) + \gamma \mathbb{E}_{a_{t+1}, p}[Q^\pi_{t+1}(X_{t+1}, O_{t+1}, U_{t+1}) | X_t = x, O_t = o, U_t = u]$$

Lemma 1 binds altogether $V^\pi_{0:T}$, $Q^\pi_{0:T}$ and $A^\pi_{0:T}$ along time, including overall performance $J(s_0; \theta_{0:T}) = \mathbb{E}_{s_0}[V^\pi_0(X_0, \emptyset)]$.

So far we restricted our attention to systems under the control of a single agent. Next, we shall generalize to settings where multiple agents cooperate to control the same system in a decentralized manner.

2.2 Decentralized partially observable Markov decision processes

Consider a slightly different framework in which n agents cooperate when facing the problem of influencing the behavior of a POMDP, but can neither see the state of the world and nor communicate with one another their noisy observations.

Definition 3. *A Dec-POMDP $M_n \doteq (\mathcal{I}_n, \mathcal{X}, \mathcal{U}, \mathcal{Z}, p, R, T, \gamma, s_0)$ is such that $i \in \mathcal{I}_n$ indexes the i^{th} agent involved in the process; $\mathcal{X}, \mathcal{U}, \mathcal{Z}, p, R, T, \gamma$ and s_0 are as in M_1; $\mathcal{U}^i$ is an individual control set of agent i, such that $\mathcal{U} = \mathcal{U}^1 \times \cdots \times \mathcal{U}^n$ specifies the set of controls $u = (u^1, \ldots, u^n)$; $\mathcal{Z}^i$ is an individual observation set of agent i, where $\mathcal{Z} = \mathcal{Z}^1 \times \cdots \times \mathcal{Z}^n$ defines the set of observations $z = (z^1, \ldots, z^n)$.*

We call the individual history of agent $i \in \mathcal{I}_n$, $o_t^i = (o_{t-1}^i, u_{t-1}^i, z_t^i)$ where $o_0^i = \emptyset$, the sequence of controls and observations up to time step $t = 0, 1, \ldots, T$. We denote $\mathcal{O}_t^i$, the set of individual histories of agent i at time step t.

Definition 4. *Agent $i \in \mathcal{I}_n$ selects control u_t^i at the t^{th} time step using a parametrized policy $\pi^i \doteq (a_0^i, a_1^i, \ldots, a_T^i)$, where $a_t^i(u_t^i|o_t^i) \doteq \mathbb{P}_{\theta_t^i}(u_t^i|o_t^i)$ is a parametrized decision rule, with parameter vector $\theta_t^i \in \mathbb{R}^{\ell_t^i}$, assuming $\ell_t^i \ll |\mathcal{O}_t^i|$.*

Similarly to M_1, individual histories grow every time step, which quickly becomes unmanageable. The only sufficient statistic for individual histories known so far [7, 10] relies on the *occupancy state* given by: $s_t(x, o) \doteq \mathbb{P}_{\theta_{0:T}^{1:n}, M_n}(x, o)$, for all $x \in \mathcal{X}$ and $o \in \mathcal{O}_t$. The individual occupancy state induced by individual history $o^i \in \mathcal{O}_t^i$ is a conditional distribution probability: $s_t^i(x, o^{-i}) \doteq \mathbb{P}(x, o^{-i}|o^i, s_t)$, where o^{-i} is the history of all agents except i. Learning to map individual histories to internal states close to individual occupancy states is hard, which limits the ability to find optimal policies in M_n. One can instead restrict attention to stationary individual policies, by mapping the history space into a finite set of internal states $\varsigma \doteq (\varsigma^1, \ldots, \varsigma^n)$ (possibly lossy representations of individual occupancy states), *e.g.,* nodes in finite-state controllers or hidden state of a Recurrent Neural Network (RNN). We define transition rules prescribing the next internal state given the current internal state, control and next observation as follows: $\psi\colon (\varsigma, u, z') \mapsto (\psi^1(\varsigma^1, u^1, z'^1), \ldots, \psi^n(\varsigma^n, u^n, z'^n))$ where $\psi^i\colon (\varsigma^i, u^i, z'^i) \mapsto \varsigma'^i$ is an individual transition rule. In general, ψ and $\psi^{1:n}$ are stochastic transition rules. In the following, we will consider these rules fixed a-priori.

The goal of solving M_n is to find a joint policy $\pi \doteq (\pi^1, \ldots, \pi^n)$, *i.e.,* a tuple of individual policies, one for each agent —that achieves the highest expected return, $\theta_{0:T}^{*,1:n} \in \arg\max_{\theta_{0:T}^{1:n}} J(s_0; \theta_{0:T}^{1:n})$, starting at initial belief s_0:

$$J(s_0; \theta_{0:T}^{1:n}) \doteq \mathbb{E}_{\pi, M_n}[\mathfrak{R}(\Omega_{0:T})] = \int \mathbb{P}_{\pi, M_n}(\omega_{0:T})\mathfrak{R}(\omega_{0:T})\mathrm{d}\omega_{0:T} \tag{5}$$

where $\mathbb{P}_{\pi, M_n}(\omega_{0:T})$ denotes the probability of generating joint trajectory $\omega_{0:T}$ under π.

M_n inherits all definitions introduced for M_1, including functions $V_{0:T}^\pi$, $Q_{0:T}^\pi$ and $A_{0:T}^\pi$ for a given joint policy π.

3 Policy Gradient For POMDPs

In this section, we will review the literature of policy gradient methods for centralized single-agent systems. In this setting, the policy gradient approach consists of a centralized algorithm which searches the best $\theta_{0:T}$ in the parameter space. Though, we restrict attention to non-stationary policies, methods discussed here easily extend to stationary policies when $a_t = a$, i.e. $\theta_t = \theta$, for all $t = 0, 1, \ldots, T$. Assuming π is differentiable w.r.t. its parameter vector, $\theta_{0:T}$, the centralized algorithm updates $\theta_{0:T}$ in the direction of the gradient:

$$\Delta\theta_{0:T} = \alpha \frac{\partial J(s_0; \theta_{0:T})}{\partial \theta_{0:T}}, \tag{6}$$

where α is the step-size. Applying iteratively such a centralized update rule, assuming a correct estimation of the gradient, $\theta_{0:T}$ can usually converge towards a local optimum. Unfortunately, correct estimation of the gradient may not be possible. To overcome this limitation, one can rely on an unbiased estimation of the gradient, actually restricting (6) to stochastic gradient: $\Delta\theta_{0:T} =$

$\alpha\mathfrak{R}(\omega_{0:T})\frac{\partial}{\partial\theta_{0:T}}\log\mathbb{P}_{\pi,M_n}(\omega_{0:T})$. We compute $\frac{\partial}{\partial\theta_{0:T}}\log\mathbb{P}_{\pi,M_n}(\omega_{0:T})$ with no knowledge of the trajectory distribution $\mathbb{P}_{\pi,M_n}(\omega_{0:T})$. Indeed $\mathbb{P}_{\pi,M_n}(\omega_{0:T}) \doteq s_0(x_0)\prod_{t=0}^{T} p(x_{t+1}, z_{t+1}|x_t, u_t)a_t(u_t|o_t)$ implies:

$$\frac{\partial\log\mathbb{P}_{\pi,M_n}(\omega_{0:T})}{\partial\theta_{0:T}} = \frac{\partial\log a_0(u_0|o_0)}{\partial\theta_0} + \ldots + \frac{\partial\log a_T(u_T|o_T)}{\partial\theta_T}.$$

3.1 Likelihood Ratio Methods

Likelihood ratio methods, *e.g.,* `Reinforce` [30], exploit the separability of parameter vectors $\theta_{0:T}$, which leads to the following update rule:

$$\Delta\theta_t = \alpha\mathbb{E}_{\mathcal{D}}\left[\mathfrak{R}(\omega_{0:T})\frac{\partial\log a_t(u_t|o_t)}{\partial\theta_t}\right], \qquad \forall t = 0, 1, \ldots, T \tag{7}$$

where $\mathbb{E}_{\mathcal{D}}[\cdot]$ is the average over trajectory samples $\mathcal{D}$ generated under policy π. The primary issue with this centralized update-rule is the high-variance of $\mathfrak{R}(\Omega_{0:T})$, which can significantly slow down the convergence. To somewhat mitigate this high-variance, one can exploit two observations. First, it is easy to see that future actions do not depend on past rewards, *i.e.,* $\mathbb{E}_{\mathcal{D}}[\mathfrak{R}(\omega_{0:t-1})\frac{\partial}{\partial\theta_t}\log a_t(u_t|o_t)] = 0$. This insight allows us to use $\mathfrak{R}(\omega_{t:T})$ instead of $\mathfrak{R}(\omega_{0:T})$ in (7), thereby resulting in a significant reduction in the variance of the policy gradient estimate. Second, it turns out that the absolute value of $\mathfrak{R}(\omega_{t:T})$ is not necessary to obtain an unbiased policy gradient estimate. Instead, we only need a relative value $\mathfrak{R}(\omega_{t:T}) - \beta_t(x_t, o_t)$, where $\beta_{0:T}$ can be any arbitrary value function, often referred to as baseline.

3.2 Actor-Critic Methods

To moderate even more the variance for the gradient estimate in (7), the policy gradient theorem [26] suggests replacing $\mathfrak{R}(\omega_{t:T})$ by $Q_t^{\mathrm{w}}(x_t, o_t, u_t)$, *i.e.,* an approximate value of taking control u_t starting in state x_t and history o_t and then following policy π from time step $t+1$ onward: $Q_t^{\mathrm{w}}(x_t, o_t, u_t) \approx Q_t^{\pi}(x_t, o_t, u_t)$, where $\mathrm{w}_t \in \mathbb{R}^{l_t}$ is a parameter vector with $l_t \ll |\mathcal{X}||\mathcal{O}_t||\mathcal{U}|$. Doing so leads us to the actor-critic algorithmic scheme, in which a centralized algorithm maintains both parameter vectors $\theta_{0:T}$ and parameter vectors $\mathrm{w}_{0:T}$: $\forall t = 0, 1, \ldots, T$,

$$\Delta w_t = \alpha\mathbb{E}_{\mathcal{D}}\left[\delta_t\frac{\partial\log a_t(u_t|o_t)}{\partial\theta_t}\right] \tag{8a}$$

$$\Delta\theta_t = \alpha\mathbb{E}_{\mathcal{D}}\left[Q_t^{\mathrm{w}}(x_t, o_t, u_t)\frac{\partial\log a_t(u_t|o_t)}{\partial\theta_t}\right] \tag{8b}$$

where $\delta_t \doteq \widehat{Q}_t^{\pi}(x_t, o_t, u_t) - Q_t^{\mathrm{w}}(x_t, o_t, u_t; w_t)$ and $\widehat{Q}_t^{\pi}(x_t, o_t, u_t)$ is an unbiased estimate of true Q-value $Q_t^{\pi}(x_t, o_t, u_t)$, *e.g.,* Monte-Carlo or temporal-difference learning methods.

The choice of parameter vector $\mathrm{w}_{0:T}$ is critical to ensure the gradient estimation remains unbiased [26]. There is no bias whenever Q-value functions $Q_{0:T}^{\mathrm{w}}$ are *compatible* with parametrized policy π. Informally, a compatible function approximation $Q_{0:T}^{\mathrm{w}}$ of $Q_{0:T}^{\pi}$ should be linear in "features" of policy π, and its parameters $\mathrm{w}_{0:T}$ are the solution of a linear regression problem that estimates $Q_{0:T}^{\pi}$ from these features. In practice, we often relax the second condition and update parameter vector $\mathrm{w}_{0:T}$ using Monte-Carlo or temporal-difference learning methods.

3.3 Natural Actor-Critic Methods

Following the direction of the gradient might not always be the right option to take. In contrast, the natural gradient suggests updating the parameter vector $\theta_{0:T}$ in the steepest ascent direction

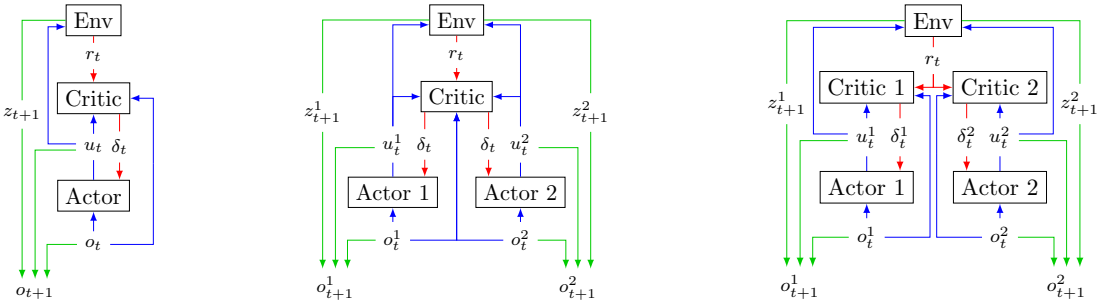


Figure 1: Best viewed in color. Actor-critic algorithmic schemes in a two-agent setting for paradigms: (*left*) CTCC; (*center*) CTDC; and (*right*) DTDC. In all figures, blue arrows represent forward control flow, green arrows show the aggregation of information for the next time step, and red arrows are the feedback signals back-propagated to update all parameters.

w.r.t. the Fisher information metric

$$\mathbf{\Phi}(\theta_t) \doteq \mathbb{E}_{\mathcal{D}}\left[\frac{\partial \log a_t(u_t|o_t)}{\partial \theta_t}\left(\frac{\partial \log a_t(u_t|o_t)}{\partial \theta_t}\right)^{\top}\right]. \tag{9}$$

This metric is invariant to re-parameterizations of the policy. Combining the policy gradient theorem with the compatible function approximations and then taking the steepest ascent direction, $\mathbb{E}_{\mathcal{D}}[\mathbf{\Phi}(\theta_t)^{-1}\mathbf{\Phi}(\theta_t)w_t]$, results in natural actor-critic algorithmic scheme, which substitutes Eq.(8b) in Eq.(8) by: $\Delta\theta_t = \alpha\mathbb{E}_{\mathcal{D}}[w_t]$.

4 Policy Gradient For Multi-Agent Systems

In this section, we review extensions of single-agent policy gradient methods to cooperative multi-agent settings. We shall distinguish between three paradigms: centralized training for centralized control (CTCC) *vs* distributed training for decentralized control (DTDC) *vs* centralized training for decentralized control (CTDC), illustrated in Figure 1.

4.1 Centralized training for centralized control (CTCC)

In certain cooperative multi-agent applications, agents have cost-free instantaneous communications. Such applications can be modeled as POMDPs, making it possible to use single-agent policy gradient methods (Section 3). In such a CTCC paradigm, *see* Figure 1 *(left)*, centralized single-agent policy gradient methods use a single critic and a single actor. The major limitation of this paradigm is also its strength: the requirement for instantaneous, free and noiseless communications among all agents till the end of the process both at the training and execution phases.

4.2 Distributed training for decentralized control (DTDC)

Perhaps surprisingly, the earliest multi-agent policy gradient method aims at learning in a distributed manner policies that are to be executed in a decentralized way, *e.g.,* distributed `Reinforce` [22]. In this DTDC paradigm, *see* Figure 1 *(right)*, agents simultaneously but independently learn via `Reinforce` their individual policies using multiple critics and multiple

actors. The independence of parameter vectors $\theta^1_{0:T}, \ldots, \theta^n_{0:T}$, leads us to the following distributed update-rule:

$$\Delta\theta^i_t = \alpha\mathbb{E}_{\mathcal{D}}\left[\mathfrak{R}(\omega_{0:T})\frac{\partial \log a^i_t(u^i_t|o^i_t)}{\partial\theta^i_t}\right], \qquad \forall t = 0, 1, \ldots, T, \forall i \in I_n \tag{10}$$

Interestingly, the sum of individual policy gradient estimates is an unbiased estimate of the joint policy gradient. However, how to exploit insights from actor-critic methods (Section 3) to combat high-variance in the joint policy gradient estimate remains an open question. Distributed `Reinforce` restricts to on-policy setting, off-policy methods instead can significantly improve the exploration, *i.e.,* learns target joint policy π while following and obtaining trajectories from behavioral joint policy $\bar{\pi}$ [6].

4.3 Centralized training for decentralized control (CTDC)

The CTDC paradigm has been successfully applied in planning for M_n [4, 7, 8, 10, 13, 16, 21, 28, 27]. In such a paradigm, a centralized coordinator agent learns on behalf of all agents at the training phase and then assigns policies to corresponding agents before the execution phase takes place. Actor-critic algorithms in this paradigm, *see* Figure 1 *(center)*, maintain a centralized critic but learn multiple actors, one for each agent. Recent work in the (deep) multi-agent RL builds upon this paradigm. [12] focuses on policies with shared parameters, and used locally shaped reward and pre-defined subtasks to speed up convergence. The CTDC paradigm was implemented in [17] through centralized critics, one per agent, having access to additional information on the whole system modeled as a Markov Game, where agents did not necessarily share a common reward. The authors focused on task with continuous actions by extending the Deep Deterministic Policy Gradient method to a multi-agent setting. The counterfactual multi-agent policy gradient (COMA) algorithm [11] focuses on a centralized critic which estimate a particular form of advantage for individual actions that can address the multi-agent credit assignment problem. Unfortunately, the method is only applied to the challenging Starcraft unit micromanagement task, which has specific state and reward structures more favorable to credit assignment. Indeed, in (weakly-)separable M_n, *see* [9], one can maintain the relative contribution of each agent and hence address the multi-agent credit assignment problem. In general M_n however, dynamics and rewards of agents can be strongly correlated, making it hard to assign credit to agents. [20] presents theoretical results very similar to ours, but in the specific *collective* multi-agent planning domain, modelled as a $\mathbb{C}$Dec-POMDP, which relies on a count-based approach where numerous agents are involved (≈ 8000), undistinguishable from one another, which enables them to share a homogeneous policy. For general Dec-POMDPs however, and in contrast to planning, the theoretical foundations that prescribe the design of the centralized critic and define the update-rules used during the training phase to preserve certain guarantees are missing.

5 Policy Gradient For Dec-POMDPs

In this section, we address the limitation of both CTCC and DTDC paradigms and extend both 'vanilla' and natural actor-critic algorithmic schemes from M_1 to M_n.

5.1 The Policy Gradient Theorem

Our primary result is an extension of the policy gradient theorem [26] from M_1 to M_n. Before proceeding any further, we start with simple yet necessary results to establish the main results of this section.

Lemma 2. *For any separable function $f\colon (x^1,\dots,x^n)\mapsto f^1(x^1)\cdots f^n(x^n)$, its partial derivative w.r.t. any x^j,for any $x=(x^1,\dots,x^n)$ where f is differentiable and $f(x)>0$, can be written as:$\frac{\partial}{\partial x^j}f(x)=f(x)\frac{\partial}{\partial x^j}\log f^j(x^j)$.*

Proof.

$$\begin{aligned}
\frac{\partial f(x)}{\partial x^j} &= f(x)\frac{\partial \log f(x)}{\partial x^j} \text{ (derivate of composition } \log\circ f)\\
&= f(x)\frac{\partial}{\partial x^j}\left(\log\prod_{i=1}^{n} f^i(x^i)\right) \text{ (separability)}\\
&= f(x)\frac{\partial}{\partial x^j}\sum_{i=1}^{n}\log f^i(x^i) \text{ (log properties)}\\
&= f(x)\sum_{i=1}^{n}\frac{\partial \log f^i(x^i)}{\partial x^j}\\
&= f(x)\frac{\partial \log f^j(x^j)}{\partial x^j} \text{ (null } \forall i\neq j)
\end{aligned}$$

□

Lemma 3. *For any distribution p and q of the same random variable X, assuming the support of q include that of p:$\mathbb{E}_p[f(X)]=\mathbb{E}_q[\frac{p(X)}{q(X)}f(X)]$.*

Proof. Let $\mathcal{X}$ be the domain of variable X.

$$\begin{aligned}
\mathbb{E}_p\left[f(X)\right] &= \int_{x\in\mathcal{X}} p(x)f(x)dx\\
&= \int_{x\in\mathcal{X}} \frac{q(x)}{q(x)}p(x)f(x)dx\\
&= \int_{x\in\mathcal{X}} q(x)\frac{p(x)}{q(x)}f(x)dx\\
&= \mathbb{E}_q\left[\frac{p(X)}{q(X)}f(X)\right]
\end{aligned}$$

□

Next, we state the partial derivatives of value functions $V^{\pi}_{0:T}$ w.r.t. the parameter vectors $\theta^{1:n}_{0:T}$ for finite-horizon settings.

Lemma 4. *For any arbitrary M_n, target joint policy $\pi\doteq(a_0,\dots,a_T)$ and behavior joint policy $\bar{\pi}\doteq(\bar{a}_0,\dots,\bar{a}_T)$, the following holds, for any arbitrary $t=0,1,\dots,T$, and agent $i\in\mathcal{I}_n$, hidden state $x_t\in\mathcal{X}$, and joint history $o_t\in\mathcal{O}_t$:*

$$\frac{\partial V^{\pi}_t(x_t,o_t)}{\partial\theta^i_t}=\mathbb{E}_{\bar{a}_t}\left[\frac{a_t(U_t|o_t)}{\bar{a}_t(U_t|o_t)}Q^{\pi}_t(x_t,o_t,U_t)\frac{\partial\log a^i_t(U^i_t|o^i_t)}{\partial\theta^i_t}\right]. \tag{11}$$

Proof. Let's compute partial derivative of $V_t^\pi(x_t, o_t)$ w.r.t. θ_t^i:

$$\frac{\partial V_t^\pi(x_t, o_t)}{\partial \theta_t^i} \doteq \frac{\partial}{\partial \theta_t^i} \sum_{u_t \in U} a_t(u_t|o_t) Q_t^\pi(x_t, o_t, u_t) \tag{12}$$

$$= \sum_{u_t \in U} \Big[\frac{\partial a_t(u_t|o_t)}{\partial \theta_t^i} Q_t^\pi(x_t, o_t, u_t) + a_t(u_t|o_t) \frac{\partial Q_t^\pi(x_t, o_t, u_t)}{\partial \theta_t^i} \Big] \tag{13}$$

$$= \sum_{u_t \in U} \Big[\frac{\partial a_t(u_t|o_t)}{\partial \theta_t^i} Q_t^\pi(x_t, o_t, u_t) + 0 \Big] \tag{14}$$

It is easy to see in Eq. (14) that $\frac{\partial}{\partial \theta_t^i} Q_t^\pi(x_t, o_t, u_t) = 0$.To better understand this, one can refer to Lemma 1, in which non of the components involved in the definition of $Q_t^\pi(x_t, o_t, u_t)$ depend on θ_t^i.The proof directly follows by rearranging terms on the top of (14) using Lemma 3 and Lemma 4. □

We are now ready to state the main result of this section.

Theorem 1. *For any arbitrary M_n, target joint policy $\pi \doteq (a_0, \dots, a_T)$ and behavior joint policy $\bar{\pi} \doteq (\bar{a}_0, \dots, \bar{a}_T)$, the following holds:*

1. *for finite-horizon settings $T < \infty$, any arbitrary $t = 0, 1, \dots, T$ and $i \in \mathcal{I}_n$,*

$$\frac{\partial J(s_0; \theta_{0:T}^{1:n})}{\partial \theta_t^i} = \gamma^t \mathbb{E}_{\bar{a}_t, M_n} \left[\frac{a_t(U_t|O_t)}{\bar{a}_t(U_t|O_t)} Q_t^\pi(X_t, O_t, U_t) \frac{\partial \log a_t^i(U_t^i|O_t^i)}{\partial \theta_t^i} \right].$$

2. *for finite-horizon settings $T = \infty$, and any arbitrary agent $i \in \mathcal{I}_n$,*

$$\frac{\partial J(s_0; \theta^{1:n})}{\partial \theta^i} = \mathbb{E}_{\bar{s}, \bar{a}} \left[\frac{a(U|\Sigma)}{\bar{a}(U|\Sigma)} Q^\pi(X, \Sigma, U) \frac{\partial \log a^i(U^i|\Sigma^i)}{\partial \theta^i} \right],$$

where $\bar{s}(x, \varsigma) \doteq \sum_{t=0}^\infty \gamma^t \mathbb{P}_{\bar{a}, \psi, M_n}(X_t = x, \Sigma_t = \varsigma)$.

Proof: Finite case. We prove the theorem for the expected discounted reward criterion.Let split the performance measure in two part, the first from time step 0 to $t-1$ and the second from time step t onward: for any arbitrary $t = 1, 2, \dots, T$,

$$\begin{aligned} J(s_0; \theta_{0:T}^{1:n}) &\doteq \mathbb{E}_{s_0}[V_0^\pi(X_0, \emptyset)] \\ &= \mathbb{E}_{s_0}[\mathbb{E}_{a_0}[Q_0^\pi(X_0, \emptyset, U_0)]] \\ &= \mathbb{E}_{a_0, s_0}[Q_0^\pi(X_0, \emptyset, U_0)] \\ &= \mathbb{E}_{a_{0:1}, M_n}[R(X_0, U_0) + \gamma Q_1^\pi(X_1, O_1, U_1)] \\ &= \mathbb{E}_{a_{0:t}, M_n}[\mathfrak{R}(\Omega_{0:t-1}) + \gamma^t Q_t^\pi(X_t, O_t, U_t)] \\ &= J(s_0; \theta_{0:t-1}^{1:n}) + \gamma^t J(s_t; \theta_{t:T}^{1:n}) \end{aligned}$$

where $J(s_0; \theta_{0:t-1}^{1:n})$ denotes the performance measure under partial joint policy $a_{0:t-1}$ starting at initial occupancy state (resp. belief) s_0:$J(s_0; \theta_{0:t-1}^{1:n}) \doteq \mathbb{E}_{a_{0:t-1}, M_n}[\mathfrak{R}(\Omega_{0:t-1})]$and $J(s_t; \theta_{t:T}^{1:n})$ denotes the performance measure under partial joint policy $a_{t:T}$ starting occupancy state s_t given by $s_t(x_t, o_t) \doteq \mathbb{P}_{a_{0:t-1}, M_n}(x_t, o_t)$:

$$\begin{aligned} J(s_t; \theta_{t:T}^{1:n}) &\doteq \mathbb{E}_{s_t, M_n}[V_t^\pi(X_t, O_t)] \\ &= \mathbb{E}_{s_t, M_n}[\mathbb{E}_{a_t}[Q_t^\pi(X_t, O_t, U_t)]] \\ &= \mathbb{E}_{s_t, a_t, M_n}[Q_t^\pi(X_t, O_t, U_t)] \end{aligned}$$

Taking the partial derivatives of $J(s_0;\theta_{0:t-1}^{1:n})$ and $\gamma^t J(s_t;\theta_{t:T}^{1:n})$ over θ_t^i for any arbitrary $t = 1,2,\ldots,T$ and agent $i \in I_n$, leads to:

$$\frac{\partial J(s_1;\theta_{0:T}^{1:n})}{\partial \theta_t^i} = \gamma^t \mathbb{E}_{s_t,M_n}\left[\frac{\partial V_t^\pi(X_t,O_t)}{\partial \theta_t^i}\right] \tag{15}$$

where $\frac{\partial}{\partial \theta_t^i} J(s_0;\theta_{0:t-1}^{1:n}) = 0$.Combining Eq. (15) along with Lemma 4 ends the proof. □

Proof: Infinite case. First let's compute the partial derivative of the value function V^π w.r.t. θ^i: $\forall x \in \mathcal{X}, \forall \varsigma$,

$$\begin{aligned}
\frac{\partial V^\pi(x,\varsigma)}{\partial \theta^i} &= \frac{\partial}{\partial \theta^i}\left(\mathbb{E}_a\left[Q^\pi(x,\varsigma,U)|\varsigma\right]\right) \\
&= \frac{\partial}{\partial \theta^i}\left(\sum_{u\in\mathcal{U}} a(u|\varsigma) Q^\pi(x,\varsigma,u)\right) \\
&= \sum_{u\in\mathcal{U}}\Big(\frac{\partial a(u|\varsigma)}{\partial \theta^i} Q^\pi(x,\varsigma,u) + a(u|\varsigma)\frac{\partial Q^\pi(x,\varsigma,u)}{\partial \theta^i}\Big) \\
&= \sum_{u\in\mathcal{U}} a(u|\varsigma)\left[\frac{\partial \log a^i(u^i|\varsigma^i)}{\partial \theta^i} Q^\pi(x,\varsigma,u) + \frac{\partial Q^\pi(x,\varsigma,u)}{\partial \theta^i}\right] \\
&= \mathbb{E}_a\left[\frac{\partial \log a^i(U^i|\varsigma^i)}{\partial \theta^i} Q^\pi(x,\varsigma,U) + \frac{\partial Q^\pi(x,\varsigma,U)}{\partial \theta^i}\Big|\varsigma\right]
\end{aligned} \tag{16}$$

Next let's develop partial derivative of $Q^\pi(x,\varsigma,u)$:

$$\begin{aligned}
\frac{\partial Q^\pi(x,\varsigma,u)}{\partial \theta^i} &= \frac{\partial}{\partial \theta^i}\left(R(x,u) + \gamma \mathbb{E}_{p,\psi}\left[V^\pi(X',\Sigma')|x,u,\varsigma\right]\right) \\
&= \gamma \mathbb{E}_{p,\psi}\left[\frac{\partial V^\pi(X',\Sigma')}{\partial \theta^i}\Big|x,u,\varsigma\right]
\end{aligned} \tag{17}$$

Injecting (17) into (16):

$$\frac{\partial V^\pi(x,\varsigma)}{\partial \theta^i} = \mathbb{E}_a\left[\frac{\partial \log a^i(U^i|\varsigma^i)}{\partial \theta^i} Q^\pi(x,\varsigma,U)\Big|\varsigma\right] + \gamma \mathbb{E}_{a,p,\psi}\left[\frac{\partial V^\pi(X',\Sigma')}{\partial \theta^i}\Big|\varsigma,x\right] \tag{18}$$

Iterating on this process from $\frac{\partial}{\partial \theta^i} V^\pi(X',\Sigma')$ in (18):

$$\begin{aligned}
\frac{\partial V^\pi(x,\varsigma)}{\partial \theta^i} = \mathbb{E}_a &\left[\frac{\partial \log a^i(U^i|\varsigma^i)}{\partial \theta^i} Q^\pi(x,\varsigma,U) \;\Big|\; \varsigma\right] \\
&+\gamma \mathbb{E}_{a,p,\psi}\left[\frac{\partial \log a^i(U'^i|\Sigma'^i)}{\partial \theta^i} Q^\pi(X',\Sigma',U') \;\Big|\; \varsigma,x\right] \\
&+\gamma^2 \mathbb{E}_{a,p,\psi}\left[\mathbb{E}_{a,p,\psi}\left[\frac{\partial V^\pi(X'',\Sigma'')}{\partial \theta^i} \;\Big|\; \Sigma',X',U',Z\right] \;\Big|\; \varsigma,x\right]
\end{aligned}$$

We keep going until:

$$\frac{\partial V^\pi(x,\varsigma)}{\partial \theta^i} = \lim_{t\to\infty} \sum_{k=0}^{t} \gamma^k \mathbb{E}_{a,M_n,\psi}^k\left[\frac{\partial \log a^i(U^i|\Sigma'^i)}{\partial \theta^i} Q^\pi(X',\Sigma',U) \;\Big|\; \varsigma,x\right] \tag{19}$$

where expected value is taken over distribution of state X' and internal representation Σ' when following policy π for k steps starting in state x and internal representation ς.Combining (19) with initial state distribution s_0 yields the final expression and ends the proof. □

While the policy gradient theorem for M_1 [26] assumes a single agent learning to act in a (PO)MDP, Theorem 1 applies to multiple agents learning to control a POMDP in a decentralized manner. Agents act independently, but their policy gradient estimates are guided by a centralized Q-value function $Q^{\pi}_{0:T}$. To use this property in practice, one needs to replace $Q^{\pi}_{0:T}$ with a function approximation of $Q^{\pi}_{0:T}$. To ensure this function approximation is compatible—*i.e.,* the corresponding gradient still points roughly in the direction of the real gradient, we carefully select its features. The following addresses this issue for M_n.

5.2 Compatible Function Approximations

The main result of this section characterizes compatible function approximations $V^{\sigma}_{0:T}$ and $A^{\nu}_{0:T}$ for both the value function $V^{\pi}_{0:T}$ and the advantage function $A^{\pi}_{0:T}$ of any arbitrary M_n, respectively. These functions together shall provide a function approximation for $Q^{\pi}_{0:T}$ assuming $Q^{\pi}_t(x_t, o_t, u_t) \doteq V^{\pi}_t(x_t, o_t) + A^{\pi}_t(x_t, o_t, u_t)$, for any time step $t = 0, 1, \ldots, T$, state x_t, joint history o_t and joint control u_t.

Theorem 2. *For any arbitrary M_n, function approximations $V^{\sigma}_{0:T}$ and $A^{\nu}_{0:T}$, with parameter vectors $\sigma^{1:n}_{0:T}$ and $\nu^{1:n}_{0:T}$ respectively, are compatible with parametric joint policy $\pi \doteq (a_0, \ldots, a_T)$, with parameter vector $\theta^{1:n}_{0:T}$, if one of the following holds: $\forall t = 0, 1, \ldots, T$*

1. *for any state $x_t \in \mathcal{X}$, joint history $o_t \in \mathcal{O}_t$, and agent $i \in \mathcal{I}_n$,*

$$\frac{\partial V^{\sigma}_t(x_t, o_t)}{\partial \sigma^i_t} = \mathbb{E}_{a^i_t}\left[\frac{\partial \log a^i_t(U^i_t|o^i_t)}{\partial \theta^i_t}\right]. \tag{20}$$

   and σ minimizes the MSE $\mathbb{E}_{\pi, M_n}[\epsilon_t(X_t, O_t, U_t)^2]$

2. *for any state $x_t \in \mathcal{X}$, joint history $o_t \in \mathcal{O}_t$, joint control $u_t \in \mathcal{U}$, and agent $i \in \mathcal{I}_n$,*

$$\frac{\partial A^{\nu}_t(x_t, o_t, u_t)}{\partial \nu^i_t} = \frac{\partial \log a^i_t(u^i_t|o^i_t)}{\partial \theta^i_t} \tag{21}$$

   and ν minimizes the MSE $\mathbb{E}_{\pi, M_n}[\epsilon_t(X_t, O_t, U_t)^2]$

where $\epsilon_t(x, o, u) \doteq Q^{\pi}_t(x, o, u) - V^{\sigma}_t(x, o) - A^{\nu}_t(x, o, u)$.

If one of the two conditions is verified, $\frac{\partial}{\partial \theta^i_t} V^{\pi}_t(x_t, o_t)$ follows

$$\mathbb{E}_{\bar{a}_t}\left[\frac{a_t(U_t|o_t)}{\bar{a}_t(U_t|o_t)}\left(V^{\sigma}_t(x_t, o_t) + A^{\nu}_t(x_t, o_t, U_t)\right)\frac{\partial \log a^i_t(U^i_t|o^i_t)}{\partial \theta^i_t}\right]. \tag{22}$$

for any behavior joint policy $\bar{\pi} \doteq (\bar{a}_0, \ldots, \bar{a}_T)$.

Proof. The proof shows that the gradient estimate based on approximation functions $V^{\sigma}_{0:T}$ et $A^{\nu}_{0:T}$such that $\sigma^{1:n}_{0:T}$ et $\nu^{1:n}_{0:T}$ satisfy one of (20) or (21) preserve the true direction of the policy gradient.We will only consider the case when (21) is verified, but a similar development holds for (20).In this case, $\nu^{1:n}_{0:T}$ minimises the mean square error (MSE) $\mathbb{E}[\epsilon_t(X_t, O_t, U_t)^2]$.So, for all time-step $t = 0, 1, \ldots, T$, and all agents $i \in \mathcal{I}_n$:

$$\frac{\partial \mathbb{E}[\epsilon^2_t(X_t, O_t, U_t)]}{\partial \nu^i_t} = 2\mathbb{E}\left[\epsilon_t(X_t, O_t, U_t)\frac{\partial A^{\nu}_t(X_t, O_t, U_t)}{\partial \nu^i_t}\right] = 0. \tag{23}$$

because distribution of X_t, O_t, U_t does not depend on ν.Injecting compatibility condition (21) into (23), we get:

$$\mathbb{E}\left[\epsilon_t(X_t, O_t, U_t)\frac{\partial \log a_t^i(U_t^i|O_t^i)}{\partial \theta_t^i}\right] = 0.$$

We can now develop $\epsilon_t(X_t, O_t, U_t) \doteq Q_t^\pi(x, o, u) - V_t^\sigma(x, o) - A_t^\nu(x, o, u)$, and re-arrange terms to obtain:

$$\mathbb{E}\left[Q_t^\pi(x_t, o_t, u_t)\frac{\partial \log a_t^i(u_t^i|o_t^i)}{\partial \theta_t^i}\right] = \mathbb{E}\left[\big(V_t^\sigma(x_t, o_t) + A_t^\nu(x_t, o_t, u_t)\big)\frac{\partial \log a_t^i(u_t^i|o_t^i)}{\partial \theta_t^i}\right]$$

Using Lemma 4 the expression becomes:

$$\frac{\partial V_t^\pi(x_t, o_t)}{\partial \theta_t^i} = \mathbb{E}\left[\big(V_t^\sigma(x_t, o_t) + A_t^\nu(x_t, o_t, U_t)\big)\frac{\partial \log a_t^i(U_t^i|o_t^i)}{\partial \theta_t^i}\right]$$

Finally we introduce behavioral policy $\bar{\pi}$ using Lemma 3 to deduce the final expression (22). □

We state Theorem 2 for non-stationary policies and $T < \infty$, but the result naturally extends to infinite-horizon and stationary policies, assuming $a_t = a$ and $\theta_t = \theta$ for all $t = 0, 1, \ldots, \infty$ and $i \in \mathcal{I}_n$. The theorem essentially demonstrates how compatibility conditions generalize from M_1 to M_n. Notable properties of the compatible centralized approximations, the single critic, include their *separability* w.r.t. individual approximations:

$$(20) \Rightarrow V_t^\sigma : (x_t, o_t) \mapsto \sum_{i \in I_n} \mathbb{E}_{a_t^i}\left[\frac{\partial \log a_t^i(U_t^i|o_t^i)}{\partial \theta_t^i}\right]^\top \sigma_t^i + \beta_t(x_t, o_t), \tag{24}$$

$$(21) \Rightarrow A_t^\nu : (x_t, o_t, u_t) \mapsto \sum_{i \in I_n}\left(\frac{\partial \log a_t^i(u_t^i|o_t^i)}{\partial \theta_t^i}\right)^\top \nu_t^i + \tilde{\beta}_t(x_t, o_t, u_t), \tag{25}$$

where $\beta_{0:T}$ and $\tilde{\beta}_{0:T}$ are baselines independent of $\theta_{0:T}^{1:n}$, $\nu_{0:T}^{1:n}$ and $\sigma_{0:T}^{1:n}$. Only one of (20) or (21) needs to be verified to preserve the direction of the policy gradient. Similarly to the compatibility theorem for M_1, the freedom granted by the potentially unconstrained approximation and the baselines can be exploited to reduce the variance of the gradient estimation, but also take advantage of extra joint or hidden information unavailable to the agents at the execution phase. We can also benefit from the separability of both approximations at once to decrease the number of learned parameters and speed up the training phase for large-scale applications. Finally, the separability of the approximation functions does not allow us to independently maintain individual critics, the gradient estimation is still guided by a centralized critic.

5.3 Actor-Critic for Decentralized Control Algorithms

In this section, we derive actor-critic algorithms for M_n that exploit insights from Theorem 2, as illustrated in Algorithm 1, namely *Actor-Critic for Decentralized Control* (`ACDC`). This algorithm is model-free, centralized[1], off-policy and iterative. Each iteration consists of policy evaluation and policy improvement. The policy evaluation composes a mini-batch based on trajectories sampled from $\mathbb{P}_{\bar{\pi}, M_n}(\Omega_{0:T})$ and the corresponding temporal-difference errors, *see* lines (6–11). The policy improvement updates θ, ν, and σ by taking the average over mini-batch samples and exploiting compatible function approximations, *see* lines (12–16).

[1]One can easily extend this algorithm to allow agents to collaborate during the training phase by exchanging their local information, and hence makes it a distributed algorithm.

Algorithm 1: Actor-Critic for Decentralized Control (ACDC).

```
ACDC()
    Initialize θ_0, ν_0, σ_0 arbitrarily and h ← 0.
    while θ_h has not converged do
        evaluation() and improvement()
        h ← h + 1
evaluation()
    Initialize D^h_{0:T} ← ∅
    for j = 1...m and t = 0...T do
        Create trajectories (x_{t:t+1}, o_{t:t+1}, u_t) ~ ā_t, p
        Evaluate δ_t ← r_t + γV^σ_{t+1}(x_{t+1}, o_{t+1}) − V^σ_t(x_t, o_t)
        Compose batch D_{t,h} ← {(o_t, u_t, δ_t, a_t(u_t|o_t)/ā_t(u_t|o_t))} ∪ D_{t,h}
improvement()
    for i = 1...n do
        Baseline σ^i_{t,h+1} ← σ^i_{t,h} + α^σ_h E_{D_{t,h}}{δ_t a_t(u_t|o_t)/ā_t(u_t|o_t) φ^i_t(o^i_t)}
        Critic ν^i_{t,h+1} ← ν^i_{t,h} + α^ν_h E_{D_{t,h}}{δ_t a_t(u_t|o_t)/ā_t(u_t|o_t) φ^i_t(o^i_t, u^i_t)}
        Actor θ^i_{t,h+1} ← θ^i_{t,h} + α^θ_h E_{D_{t,h}}{a_t(u_t|o_t)/ā_t(u_t|o_t) φ^i_t(o^i_t, u^i_t) ν^i_t}
```

The step-sizes α_h^θ, α_h^ν and α_h^σ should satisfy the standard Robbins and Monro's conditions for stochastic approximation algorithms [23], *i.e.,* $\sum_{h=0}^{\infty} \alpha_h = \infty$, $\sum_{h=0}^{\infty} \alpha_h^2 < \infty$. Moreover, according to [15], they should be scheduled such that we update θ at a slower time-scale than ν and σ to ensure convergence. To ease the maximum improvement of a joint policy for a constant fixed change of its parameters, the method of choice is the natural policy gradient [1, 14]. The natural `ACDC` (`NACDC`) differs from `ACDC` only in the update of the actors: $\theta^i_{t,h+1} \leftarrow \theta^i_{t,h} + \alpha^\theta_h \mathbb{E}_{\mathcal{D}_{t,h}}[\frac{a_t(u_t|o_t)}{\bar{a}_t(u_t|o_t)} \nu^i_t]$.

We conclude this section with remarks on theoretical properties of `ACDC` algorithms. First, they are guaranteed to converge with probability one under mild conditions to local optima as they are true gradient descent algorithms [6]. The basic argument is that they minimize the mean square projected error by stochastic gradient descent, *see* [6] for further details. They further terminate with a local optimum that is also a Nash equilibrium, *i.e.,* the partial derivatives of the centralized critic w.r.t. any parameter is zero only at an equilibrium point.

6 Experiments

In this section, we empirically demonstrate and validate the advantage of CTDC over CTCC and DTDC paradigms. We show that `ACDC` methods compare favorably w.r.t. existing algorithms on many decentralized multi-agent domains from the literature. We also highlight limitations that preclude the current implementation of our methods to achieve better performances.

6.1 Experimental Setup

As discussed throughout the paper, there are many key components in actor-critic methods that can affect their performances. These key components include: training paradigms (CTCC *vs* DTDC *vs* CTDC); policy representations (stationary *vs* non-stationary policies); approximation architectures (linear approximations *vs* deep recurrent neural networks); history representations (truncated histories *vs* hidden states of deep neural networks). We implemented three variants of actor-critic methods that combine these components. Unless otherwise mentioned, we will refer

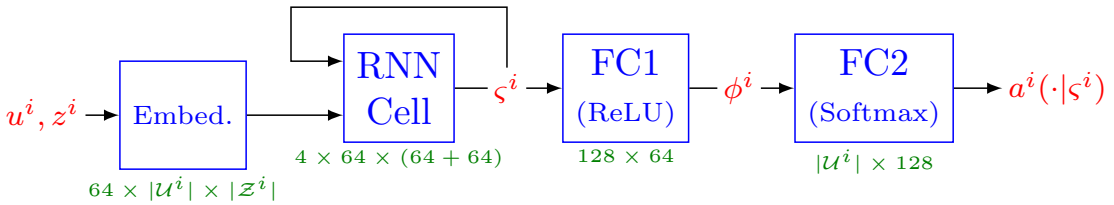


Figure 2: Best viewed in color. Recurrent neural network architecture used to represent actors of agent $i \in I_n$. The blue boxes are standard neural network layers, red text denotes intermediate tensors computed during forward pass, and green text indicates the number of parameters in each layer. An LSTM cell maintains an internal state updated using an embedding of the action-observation pair. A fully connected layer followed by an ReLU generates a feature vector ϕ^i, which are combined by a second FC layer then normalized by Softmax to get conditional decision rule $a^i(\cdot|\varsigma^i)$.

to actor-critic methods from: the acronym of the paradigm in which they have been implemented, *e.g.,* CTDC for ACDC; plus the key components, "*CTDC_TRUNC(K)*" for `ACDC` where we use K last observations instead of histories (non-stationary policy); or "*DTDC_RNN*" for distributed `Reinforce` where we use RNNs (stationary policy), see Figure 2.

We conducted experiments on a Dell Precision Tower 7910 equipped with a 16-core, 3GHz Intel Xeon CPU, 16GB of RAM and a 2GB nVIDIA Quadro K620 GPU. We run simulations on standard benchmarks from Dec-POMDP literature, including *Dec. Tiger*, *Broadcast Channel*, *Mars*, *Box Pushing*, *Meeting in a Grid*, and *Recycling Robots*, see `http://masplan.org`. Meta-parameters are reported in Table 1. Learning rate is updated as following: At iteration h,

$$\alpha_h = \alpha_{\min} + (\alpha_0 - \alpha_{\min}) \times \exp\left(\frac{h}{\lambda_\alpha} \log \kappa_\alpha\right)$$

Representation	RNN	trunc(K)
Horizon	10 / ∞[1]	10 / ∞[1]
Discount	1 / 0.9	1 / 0.9
Trials	3	3
Iterations	100000	100000
Mini-batch	256	32
Initial learning rate α_0	0.05	1.0
Minimal learning rate $\alpha_{\min}$	0	0
Learning rate decay λ_α	0.96	0.37
Learning rate step-size κ_α	40000	80000
Error type	Monte-Carlo	Time-Difference
Gradient clipping	20	Aucune
Sampling	On-policy	On-policy

Table 1: Meta-parameters for `ACDC` methods.- Note[1]: Algorithms sampled trajectories of size T such that $\frac{1}{(1-\gamma^T)} \max_{x,u} R(x,u) \ll \epsilon$

6.2 History Representation Matters

In this section, we conducted experiments with the goal of gaining insights on how the representation of histories affects the performance of ACDC methods. Figure 3 depicts the comparison of

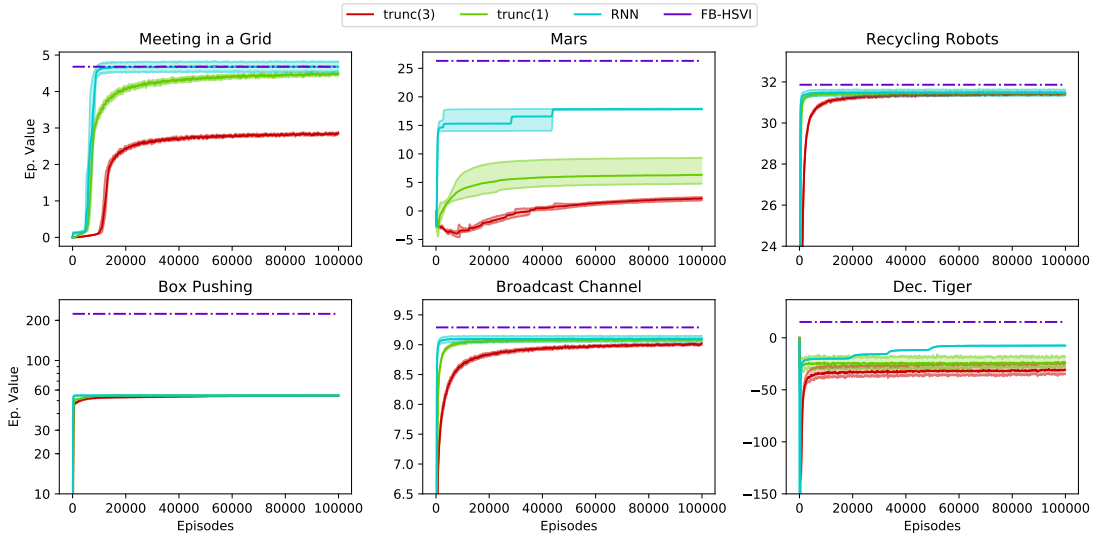


Figure 3: Comparison of different structures used to represent histories.

truncated histories *vs* hidden states of deep neural networks. Results obtained using an ϵ-optimal planning algorithm called `FB-HSVI` [7] are included as reference. For short planning horizons, *e.g.*, $T = 10$, *CTDC_RNN* quickly converges to good solutions in comparison to *CTDC_TRUNC(1)* and *CTDC_TRUNC(3)*. This suggests CTDC rnn learns more useful and concise representations of histories than the truncated representation. However, for some of the more complex tasks such as *Dec. Tiger*, *Box Pushing* or *Mars*, no internal representation was able to perform optimally.

Overall, our experiments on history representations show promising results for RNNs, which have the advantage over truncated histories to automatically learn equivalence classes and compact internal representations based on the gradient back-propagated from the reward signal. Care should be taken though, as some domain planning horizons and other specific properties might cause early convergence to poor local optima. We are not entirely sure which specific features of the problems deteriorate performances, and we leave for future works to explore better methods to train these architectures.

6.3 Comparing Paradigms Altogether

In this section, we compare paradigms, CTCC, DTDC, and CTDC. We complement our experiments with results from other Dec-POMDP algorithms: an ϵ-optimal planning algorithm called `FB-HSVI` [7]; and a sampling-based planning algorithm called Monte-Carlo Expectation-Maximization (`MCEM`) algorithm [31], which shares many similarities with actor-critic methods. It is worth noticing that we are not competing against FB-HSVI as it is model-based. As for MCEM, we reported performances[2] recorded in [31].

In almost all tested benchmarks, CTDC seems to take the better out of the two other paradigms, for either $T = 10$ (Fig. 4) or $T = \infty$ (Fig. 5). CTCC might suffer from the high dimensionality of the joint history space, and fail to explore it efficiently before the learning step-sizes become negligible, or we reached the predefined number of training episodes. Our on-policy sampling evaluation certainly amplified this effect. Having a much smaller history space

[2]Two results in MCEM [31] were above optimal values, so we reported optimal values instead.

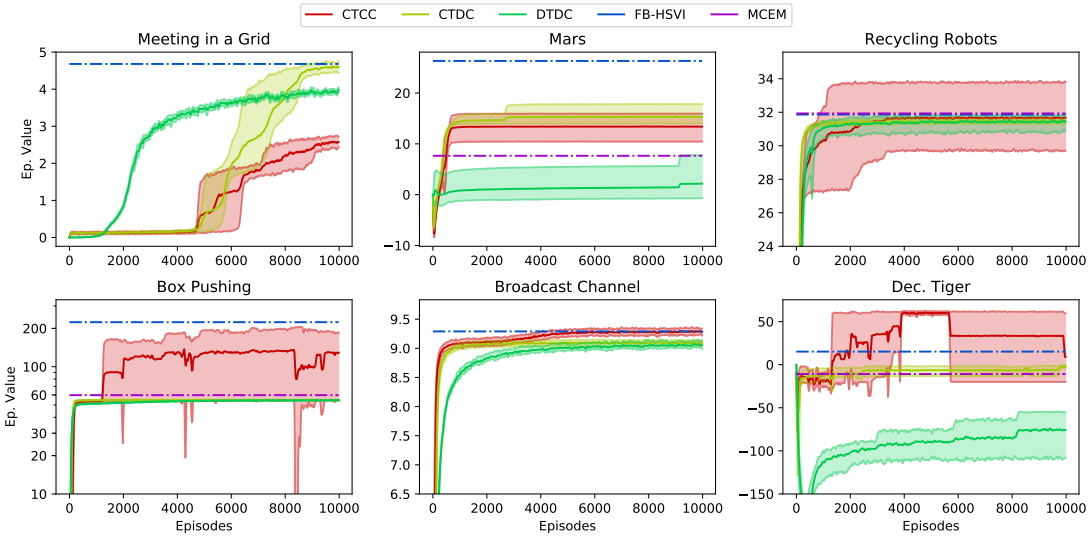


Figure 4: Comparison of the three paradigms for $T = 10$.

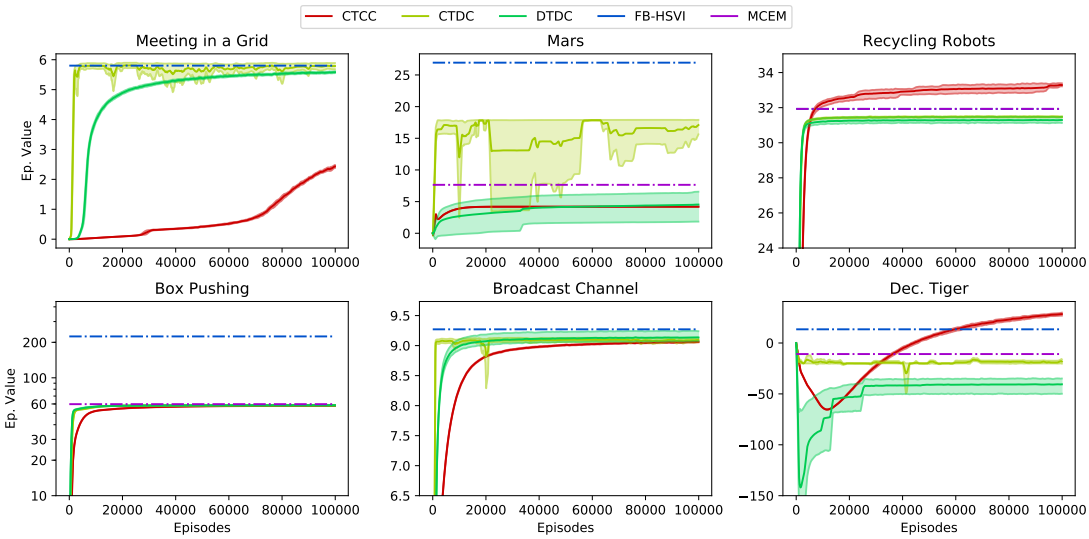


Figure 5: Comparison of the three paradigms for $T = \infty$.

to explore, CTDC outperforms CTCC in these experiments. Compared to DTDC which also explores smaller history space, there is a net gain to consider a compatible centralized critic in the CTDC paradigm, resulting in better performances. Even if CTDC achieves performances better or equal to the state of the art MCEM algorithm, there is still some margins of improvements to reach the global optima given by FB-HSVI in every benchmark. As previously mentioned, this is partly due to inefficient representations of histories.

7 Conclusion

This paper establishes the theoretical foundations of centralized actor-critic methods for Dec-POMDPs within the CTDC paradigm. In this paradigm, a centralized actor-critic algorithm learns independent policies, one for each agent, using a centralized critic. In particular, we show that the compatible centralized critic is the sum of individual critics, each of which is linear in the "features" of its corresponding individual policy. Experiments demonstrate our actor-critic methods, namely ACDC, compares favorably against methods from standard RL paradigms in benchmarks from the literature. Current implementations of ACDC reveal a challenging and open issue, namely the representation learning problem of individual histories, *e.g.,* learning to map individual histories to individual occupancy states. We plan to address this limitation in the future. Whenever the representation of individual histories is not an issue, ACDC can exploit the separability of the centralized critic to scale up the number of agents. We are currently investigating a large-scale decentralized multi-agent application, where we plan to exploit this scalability property.

References

[1] Amari, S.I.: Natural Gradient Works Efficiently in Learning. Neural Comput. **10**(2) (1998)

[2] Aström, K.J.: Optimal Control of Markov Decision Processes with Incomplete State Estimation. Journal of Mathematical Analysis and Applications **10** (1965)

[3] Bellman, R.E.: The Theory of Dynamic Programming. Bulletin of the American Mathematical Society **60**(6) (1954)

[4] Bernstein, D.S., Givan, R., Immerman, N., Zilberstein, S.: The Complexity of Decentralized Control of Markov Decision Processes. Mathematics of Operations Research **27**(4) (2002)

[5] Boutilier, C.: Planning, Learning and Coordination in Multiagent Decision Processes. In: Proc. of the Sixth Conf. on Theoretical Aspects of Rationality and Knowledge (1996)

[6] Degris, T., White, M., Sutton, R.S.: Linear off-policy actor-critic. In: Proc. of the 29th Int. Conf. on ML, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012 (2012)

[7] Dibangoye, J.S., Amato, C., Buffet, O., Charpillet, F.: Optimally Solving Dec-POMDPs as Continuous-State MDPs. Journal of AI Research **55** (2016)

[8] Dibangoye, J.S., Amato, C., Buffet, O., Charpillet, F.: Optimally Solving Dec-POMDPs As Continuous-state MDPs. In: Proc. of the Twenty-Fourth Int. Joint Conf. on AI (2013)

[9] Dibangoye, J.S., Amato, C., Buffet, O., Charpillet, F.: Exploiting Separability in Multi-Agent Planning with Continuous-State MDPs. In: Proc. of the Thirteenth Int. Conf. on Autonomous Agents and Multiagent Systems (2014)

[10] Dibangoye, J.S., Amato, C., Buffet, O., Charpillet, F.: Optimally solving Dec-POMDPs as Continuous-State MDPs: Theory and Algorithms. Research Report RR-8517 (2014)

[11] Foerster, J., Farquhar, G., Afouras, T., Nardelli, N., Whiteson, S.: Counterfactual multi-agent policy gradients (2018)

[12] Gupta, J.K., Egorov, M., Kochenderfer, M.: Cooperative multi-agent control using deep reinforcement learning. In: Autonomous Agents and Multiagent Systems (2017)

[13] Hansen, E.A., Bernstein, D.S., Zilberstein, S.: Dynamic Programming for Partially Observable Stochastic Games. In: Proc. of the Nineteenth National Conf. on AI (2004)

[14] Kakade, S.: A Natural Policy Gradient. In: Advances in Neural Information Processing Systems 14 (NIPS 2001) (2001)

[15] Konda, V.R., Tsitsiklis, J.N.: Actor-critic algorithms. In: Adv. in Neural Information Processing Systems 12 (2000)

[16] Kraemer, L., Banerjee, B.: Multi-agent reinforcement learning as a rehearsal for decentralized planning. Neurocomputing **190** (2016)

[17] Lowe, R., WU, Y., Tamar, A., Harb, J., Pieter Abbeel, O., Mordatch, I.: Multi-agent actor-critic for mixed cooperative-competitive environments. In: Adv. in Neural Information Processing Systems 30 (2017)

[18] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., Hassabis, D.: Human-level control through deep reinforcement learning. Nature **518**(7540) (feb 2015)

[19] Moravčík, M., Schmid, M., Burch, N., Lisý, V., Morrill, D., Bard, N., Davis, T., Waugh, K., Johanson, M., Bowling, M.: DeepStack: Expert-level artificial intelligence in heads-up no-limit poker. Science **356**(6337) (2017)

[20] Nguyen, D.T., Kumar, A., Lau, H.C.: Policy gradient with value function approximation for collective multiagent planning. In: Adv. in Neural Information Processing Systems 30 (2017)

[21] Oliehoek, F.A., Spaan, M.T.J., Amato, C., Whiteson, S.: Incremental Clustering and Expansion for Faster Optimal Planning in Dec-POMDPs. Journal of AI Research **46** (2013)

[22] Peshkin, L., Kim, K.E., Meuleau, N., Kaelbling, L.P.: Learning to Cooperate via Policy Search. In: Sixteenth Conf. on Uncertainty in Artificial Intelligence (UAI-2000) (2000)

[23] Robbins, H., Monro, S.: A stochastic approximation method. The annals of mathematical statistics **22**(3) (1951)

[24] Shoham, Y., Leyton-Brown, K.: Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations. New York, NY, USA (2008)

[25] Sutton, R.S., Barto, A.G.: Introduction to Reinforcement Learning. Cambridge, MA, USA, 2nd edn. (2016)

[26] Sutton, R.S., McAllester, D., Singh, S., Mansour, Y.: Policy Gradient Methods for Reinforcement Learning with Function Approximation. In: Proc. of the 12th Int. Conf. on Neural Information Processing Systems. Cambridge, MA, USA (1999)

[27] Szer, D., Charpillet, F.: An Optimal Best-First Search Algorithm for Solving Infinite Horizon DEC-POMDPs. In: Proc. of the Fifteenth European Conf. on ML (2005)

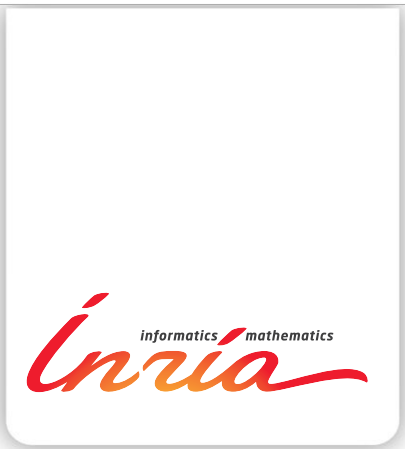
[28] Szer, D., Charpillet, F., Zilberstein, S.: MAA*: A Heuristic Search Algorithm for Solving Decentralized POMDPs. In: Proc. of the Twenty-First Conf. on Uncertainty in AI (2005)

[29] Tan, M.: Multi-agent Reinforcement Learning: Independent vs. Cooperative Agents. In: Readings in Agents. San Francisco, CA, USA (1998)

[30] Williams, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. ML **8**(3) (1992)

[31] Wu, F., Zilberstein, S., Jennings, N.R.: Monte-Carlo Expectation Maximization for Decentralized POMDPs. In: Proc. of the Twenty-Fourth Int. Joint Conf. on AI (2013)

[32] Zhang, X., Aberdeen, D., Vishwanathan, S.V.N.: Conditional Random Fields for Multi-agent Reinforcement Learning. Proc. of the 24th Int. Conf. on ML (2007)



RESEARCH CENTRE
GRENOBLE – RHÔNE-ALPES

Inovallée
655 avenue de l'Europe Montbonnot
38334 Saint Ismier Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399