A Cross-Entropy Approach to Solving Dec-POMDPs

Frans A. Oliehoek[1], Julian F.P. Kooij[1], and Nikos Vlassis[2]

[1] Intelligent Systems Lab, University of Amsterdam, Kruislaan 403, 1098 SJ Amsterdam, The Netherlands ({faolieho,jkooij}@science.uva.nl)
[2] Dept. of Production Engineering and Management, Technical University of Crete, Kounoupidiana 73100 Chania, Greece (vlassis@dpem.tuc.gr)

1 Introduction

In this paper we focus on distributed multiagent planning under uncertainty. For single-agent planning under uncertainty, the partially observable Markov decision process (POMDP) is the dominant model (see [Spaan and Vlassis, 2005] and references therein). Recently, several generalizations of the POMDP to multiagent settings have been proposed. Here we focus on the *decentralized POMDP (Dec-POMDP)* model for multiagent planning under uncertainty [Bernstein et al., 2002, Goldman and Zilberstein, 2004]. Solving a Dec-POMDP amounts to finding a set of optimal policies for the agents that maximize the expected shared reward. However, solving a Dec-POMDP has proven to be hard (NEXP-complete): The number of possible deterministic policies for a single agent grows doubly exponentially with the planning horizon, and exponentially with the number of actions and observations available. As a result, the focus has shifted to approximate solution techniques [Nair et al., 2003, Emery-Montemerlo et al., 2005, Oliehoek and Vlassis, 2007].

In this paper we propose a novel approach for approximately solving Dec-POMDPs. We apply the *Cross-Entropy (CE) method* [de Boer et al., 2005], a sampling-based method for solving combinatorial problems, to policy search in Dec-POMDPs. [Mannor, Rubinstein, and Gat, 2003] applied CE for policy search in Markov Decision Processes (MDPs). This work proposes solutions for the problems encountered when going to a multiagent setting with state uncertainty, thereby extending work by [Mannor et al., 2003]. We show experimental results from a toy problem and a standard benchmark, from which encouraging conclusions can be drawn.

2 The DEC-POMDP Model

The *decentralized partially observable Markov decision process (Dec-POMDP)* describes a stochastic, partially observable environment for a set of cooperating agents. A Dec-POMDP for m agents can formally be defined as a tuple $\langle \mathcal{S}, \mathcal{A}, T, R, \mathcal{O}, O \rangle$ where:

- $\mathcal{S}$ is a finite set of states.
- The set $\mathcal{A} = \times_i \mathcal{A}_i$ is the set of *joint actions*, where $\mathcal{A}_i$ is the set of actions available to agent i. Every time step one joint action $\mathbf{a} = \langle a_1, ..., a_m \rangle$ is taken.
- T is the transition function, a mapping from states and joint actions to probability distributions over next states: $T : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$.
- R is the reward function, a mapping from states and joint actions to real numbers: $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$.
- $\mathcal{O} = \times_i \mathcal{O}_i$ is the set of *joint observations*, with $\mathcal{O}_i$ the set of observations available to agent i. Every time step one joint observation $\mathbf{o} = \langle o_1, ..., o_m \rangle$ is received. We will denote $\mathbf{o}^t$ the joint observation at time t.
- O is the observation function, a mapping from joint actions and successor states to probability distributions over joint observations: $O : \mathcal{A} \times \mathcal{S} \to \mathcal{P}(\mathcal{O})$.

In this paper we assume a finite *planning* horizon of h time steps, and an initial 'belief' $b^0 \in \mathcal{P}(\mathcal{S})$; this is the initial state distribution at $t = 0$. In a Dec-POMDP, an agent only knows its own actions a_i and observations o_i. The *action-observation history for agent i*, $\vec{\theta}_i^t = \big(a_i^0, o_i^1, a_i^1, ..., a_i^{t-1}, o_i^t\big)$, is the sequence of actions taken and observations received by agent i until time step t. The *joint action-observation history* is a tuple with the action-observation history for all agents $\vec{\boldsymbol{\theta}}^t = \langle \vec{\theta}_1^t, ..., \vec{\theta}_m^t \rangle$. The set of all action-observation histories for agent i at time t is denoted $\vec{\Theta}_i$. The *observation history for agent i*, $\vec{o}_i^t = \big(o_i^1, ..., o_i^t\big)$, is the sequence of observations an agent has received. $\vec{\mathbf{o}}^t$ denotes a joint observation history and $\vec{\mathcal{O}}_i$ denotes the set of all observation histories for agent i.

A *pure* or *deterministic policy* π_i for agent i in a Dec-POMDP is a mapping from observation histories to actions, $\pi_i : \vec{\mathcal{O}}_i \to \mathcal{A}_i$. A pure joint policy π is a tuple containing a pure policy for each agent. Solving a Dec-POMDP amounts to finding π^*, the joint policy that yields the highest expected cumulative reward: $\pi^* = \arg\max_\pi \; E_\pi\big(\sum_{t=0}^{h-1} R(t)\big)$. Bernstein et al. [2002] have shown that optimally solving a Dec-POMDP is NEXP-complete, implying that any optimal algorithm will most likely be doubly exponential in the horizon.

The naive way of going about is to enumerate all joint policies and evaluate their expected cumulative reward, or *value*. The value of a specific (state, joint observation history) pair under a joint policy π is given by:

$$V_\pi(s^t, \vec{\mathbf{o}}^t) = R(s^t, \pi(\vec{\mathbf{o}}^t)) + \sum_{s^{t+1}} P(s^{t+1}|s^t, \pi(\vec{\mathbf{o}}^t)) \sum_{\mathbf{o}^{t+1}} P(\mathbf{o}^{(t+1)}|s^{t+1}, \pi(\vec{\mathbf{o}}^t)) V_\pi(s^{t+1}, \vec{\mathbf{o}}^{t+1}). \quad (1)$$

The total expected reward $V(\pi)$, with respect to the initial state distribution b^0 is then given by $V(\pi) = \sum_s V_\pi(s, \vec{\mathbf{o}}^0) b^0(s)$, where $\vec{\mathbf{o}}^0$ is the initial (empty) joint observation history. For one joint policy this calculation requires evaluation of (1) for each of the $\sum_{t=0}^{h-1} |\mathcal{O}|^t = \frac{|\mathcal{O}|^h - 1}{|\mathcal{O}| - 1}$ observation histories and $|\mathcal{S}|$ states, leading to a total cost of $O(|S| \cdot \frac{|\mathcal{O}|^h - 1}{|\mathcal{O}| - 1})$.

3 Cross-Entropy Optimization

De Boer, Kroese, Mannor, and Rubinstein [2005] described the *Cross-Entropy (CE)* method as a general framework to both rare event estimation and combinatorial optimization. We will focus here only on the application to optimization. The cross entropy method can be used for optimization in cases where we want to find a (typically large) vector x from a hypothesis space $\mathcal{X}$, that maximizes some *performance function* $V : \mathcal{X} \to \mathbb{R}$. That is, we are looking for $x^* = \arg\max_{x \in \mathcal{X}} V(x)$. The CE method maintains a probability density function f_ξ over the hypothesis space, parametrized by a vector ξ. The core of the CE method for optimization is an iterative two-phase process:

1. Generate a set of samples $\mathbf{X}$ according to f_ξ.
2. For some $0 \leq \rho \leq 1$, select the best ρ-fraction of samples $\mathbf{X}_\rho$, and use those to update the parameter vector ξ.

The first step is rather trivial. The second step, however, deserves some more explanation. Let $\gamma^{(j)}$ be defined as the minimum performance within the best ρ-fraction of samples of the j-th iteration. I.e., $\gamma^{(j)} \equiv \min_{\mathbf{x} \in \mathbf{X}_\rho} V(\mathbf{x})$. The CE method requires that this lower bound performance is not allowed to decrease over time: $\gamma^{(j+1)} \geq \gamma^{(j)}$. This implies that $\mathbf{X}_\rho$ can contain less than a ρ-fraction of all samples $\mathbf{X}$. The set $\mathbf{X}_\rho$ is then used to create $\xi^{(j+1)}$, a maximum-likelihood estimate of the parameters. These new parameters can be smoothed using a parameter $0 \leq \alpha \leq 1$ by interpolating with $\xi^{(j)}$ the parameter vector of the previous iteration: $\xi^{(j+1)} = \alpha \xi^{(j+1)} + (1 - \alpha)\xi^{(j)}$. This reduces the probability that some components of the parameter vector will be 0 or 1 early in the CE process, which could cause the method to get stuck in local optima. Usually, the iterative process is stopped when $\gamma^{(j)}$ has not improved over some predefined number of steps. But other conditions could be used such as a time limit or a fixed number of iterations. When the stop condition is finally met, the best sample $\mathbf{x}$ found in the entire process is returned as an approximation of x^*.

Mannor et al. [2003] showed how the CE method can be applied to MDPs for which the optimal value function is stationary, that is, the expected value

of taking a particular action in a particular state does not depend on the time step. The optimal policy for such a MDP is a mapping from states to actions $\pi_{\text{MDP}} : \mathcal{S} \to \mathcal{A}$, which can be represented as an $|\mathcal{S}|$-vector. As above, we want to find the vector that maximizes a performance function, in this case the expected total reward. So we are looking for $\pi^*_{\text{MDP}} = \arg\max_{\pi_{\text{MDP}}} V(\pi_{\text{MDP}})$, where the performance function now is the value of the MDP-policy π_{MDP}. This problem is tackled by maintaining a parameter vector $\xi = \langle \xi_{s_0}, ..., \xi_{s_{|\mathcal{S}|}} \rangle$, where each ξ_s is a probability distribution over actions. Using these probabilities it is possible to sample N trajectories: starting from some state actions are randomly selected according to the probabilities as described by ξ until the goal state is reached. Now using the ρ-fraction of best (highest total reward) trajectories $\mathbf{X}_\rho$, the parameter vector can be updated as follows:

$$P(a|s) = \frac{\sum_{\mathbf{x} \in \mathbf{X}_\rho} I(\mathbf{x}, s, a)}{\sum_{\mathbf{x} \in \mathbf{X}_\rho} I(\mathbf{x}, s)}, \tag{2}$$

where $I(\mathbf{x}, s, a)$ is an indicator function that indicates that action a was performed at state s in trajectory $\mathbf{x}$, and $I(\mathbf{x}, s)$ indicates whether s was visited in trajectory $\mathbf{x}$. After updating the parameter vector ξ, a new set $\mathbf{X}$ of trajectories can be sampled, etc. Empirical evaluation shows that this process converges to near-optimal policies in only a few iterations.

4 CE for DEC-POMDPs

In this section we propose an adaptation of the CE method for Dec-POMDP policy optimization. Overall, the approach we describe here follows the algorithm described in the previous section. Unfortunately we cannot apply the above approach directly to Dec-POMDPs. The reason is that, since we consider finite-horizon Dec-POMDPs, there is no stationary value function. Moreover, the policies of the agents are not defined over states but over their *individual* observation histories $\vec{o}_i^t$. In the Dec-POMDP case, the hypothesis space is the space of joint policies; we need to define a parametrized distribution over this space and an evaluation function for sampled policies. Also we need to show how the parameters can be updated.

4.1 POLICY DISTRIBUTIONS

In the case of Dec-POMDPs, f_ξ denotes a probability distribution over pure joint policies. We will represent this probability as the product of probability distributions over individual pure joint policies: $f_\xi(\pi) = \prod_{i=1}^m f_{\xi_i}(\pi_i)$. Here ξ_i is the vector of parameters for agent i, i.e., $\xi = \langle \xi_1, ..., \xi_m \rangle$. The question is how to represent the probability distributions over individual pure policies. One option is to use a mixed policy [Osborne and Rubinstein, 1994] representation: a distribution over all agent i's pure policies. However, this approach suffers

from two drawbacks: the number of pure individual policies π_i might be huge and this representation is hard to parametrize in a meaningful way using some vector ξ_i. That is, it gives no access to the internals of the policies: parameters would specify probabilities for entire pure policies, rather than specifying behavior for particular observation histories. Rather then using a mixed policy representation, we will use a behavioral- [Osborne and Rubinstein, 1994] or stochastic policy [Koller and Pfeffer, 1997] description: a mapping from decision points to probability distributions over actions. We consider two such representations: observation- and action-observation history based.

Observation history based: The decision points in an MDP are the states. In a Dec-POMDP the decision points for an agent are its observation histories. Therefore, the simplest way to represent a policy distribution is as a probability distribution over actions for each observation history. In particular, for each $\vec{o}_i^t$ we maintain a $\xi_{\vec{o}_i^t}$, that specifies the distribution $\forall_{a_i} \quad \xi_{\vec{o}_i^t}(a_i) \equiv P(a_i|\vec{o}_i^t)$. Consequently, ξ_i is defined as $\xi_i \equiv \langle \xi_{\vec{o}_i^t} \rangle_{\vec{o}_i^t \in \vec{\mathcal{O}}_i}$, and the probability of a policy π_i for agent i as $f_{\xi_i}(\pi_i) = \prod_{\vec{o}_i^t \in \vec{\mathcal{O}}_i} \xi_{\vec{o}_i^t}(\pi_i(\vec{o}_i^t))$. We refer to this policy distribution representation as the OH-based representation.

Action-observation history based: Defining the parameters as above might be the most straightforward approach, because it is very closely related to the approach for MDPs. Nevertheless, this representation fails to take into account the action history: The choice for action $\pi_i(\vec{o}_i^t)$ has no influence on the choice for the action at the next time step $\pi_i(\vec{o}_i^{t+1})$. To overcome this problem, we propose to make the probability of actions conditional on the entire action-observation history $\vec{\theta}_i^t$. So now the parameter vector for agent i is defined as $\xi_i \equiv \langle \xi_{\vec{\theta}_i^t} \rangle_{\vec{\theta}_i^t \in \vec{\Theta}_i}$ and $\xi_{\vec{\theta}_i^t}$ is a probability distribution over actions: $\forall_{a_i} \quad \xi_{\vec{\theta}_i^t}(a_i) \equiv P(a_i|\vec{\theta}_i^t)$. An action-observation history consists of an action- and an observation history $\vec{\theta}_i^t = (\vec{a}_i^t, \vec{o}_i^t)$. Therefore, in this new representation, which we refer to as AOH-based, the probability of agent i's pure policy π_i becomes $f_{\xi_i}(\pi_i) = \prod_{\vec{o}_i^t \in \vec{\mathcal{O}}_i} \xi_{\vec{\theta}_i^{t=(\vec{o}_i^t, \vec{a}_{\pi_i}^t)}}(\pi_i(\vec{o}_i^t))$, where $\vec{a}_{\pi_i}^t$ is the action history as specified by π_i for observation history $\vec{o}_i^t$, i.e., $\vec{a}_{\pi_i}^t = \langle \pi_i(\vec{o}_i^0), \pi_i(\vec{o}_i^1), \cdots, \pi_i(\vec{o}_i^{t-1}) \rangle$. Put differently, for each $\vec{o}_i^t$, the parameter $\xi_{\vec{\theta}_i^t}$ used is that of the action-observation history $\vec{\theta}_i^t$ that is *consistent* with $\vec{o}_i^t$ and π_i: it specifies the observations from $\vec{o}_i^t$ and the actions that π_i specifies for those observations. Drawing a sample from this distribution is performed in a root-to-leaf fashion: first an action a_i^0 is sampled for the empty action-observation history $\vec{\theta}_i^0$ according to $\xi_{\vec{\theta}_i^0}$, then, for all possible action-observation histories $\vec{\theta}_i^1 = (\vec{a}_i^1, \vec{o}_i^1)$ at time $t = 1$ *that are consistent with the actions sampled so far*, new actions are sampled according to $\xi_{\vec{\theta}_i^1}$, etc.

4.2 PARAMETER ESTIMATION

The previous section described two ways to represent the probability distribution over policies. This section describes how the set of best policies $\mathbf{X}_\rho$ sampled from the previous distribution $f_{\xi^{(j)}}$, can be used to find new parameters $\xi^{(j+1)}$.

OH-BASED DISTRIBUTION: Let $I(\pi_i, \vec{o}_i^t, a)$ be an indicator function that indicates whether $\pi_i(\vec{o}_i^t) = a$. In the OH-based distribution the probability of agent i taking action $a^t \in \mathcal{A}_i$ after having observed $\vec{o}_i^t$ can be updated using: $\xi_{\vec{o}_i^t}^{(j+1)}(a^t) = \frac{1}{|\mathbf{X}_\rho|}\sum_{\pi\in\mathbf{X}_\rho} I(\pi_i, \vec{o}_i^t, a^t)$, where $|\mathbf{X}_\rho|$ normalizes the distribution.

AOH-BASED DISTRIBUTION: Estimating the parameters for the AOH-based distribution is more involved. The indicator function $I(\pi_i, \vec{\theta}_i^t, a_i^t)$ indicates now whether (i) $\pi_i(\vec{o}_i^t) = a_i^t$, and (ii) the action-observation history $\vec{\theta}_i^t$ is consistent with the policy, that is, whether $\vec{\theta}_i^t = (\vec{o}_i^t, \vec{a}_{\pi_i}^t)$ and $\vec{a}_{\pi_i}^t$ is the sequence of actions specified by π_i up to time step t for observation history $\vec{o}_i^t$. The new distribution parameters can be estimated by

$$\xi_{\vec{\theta}_i^t}^{(j+1)}(a^t) = \frac{\sum_{\pi\in\mathbf{X}_\rho} I(\pi_i, \vec{\theta}_i^t, a^t)}{\sum_{a\in\mathcal{A}_i}\sum_{\pi\in\mathbf{X}_\rho} I(\pi_i, \vec{\theta}_i^t, a)}. \tag{3}$$

However, it may happen that certain action-observation histories $\vec{\theta}_i^t$ are not consistent with any policy in $\mathbf{X}_\rho$, in which case both the nominator and denominator in (3) are 0. In such a case, one can simply keep the previous parameters, thus take $\xi_{\vec{\theta}_i^t}^{(j+1)} = \xi_{\vec{\theta}_i^t}^{(j)}$. We used another approach that defines the next distribution over actions as uniform, indicating that we are indifferent to which action to take since we have no reason to prefer one action above another based on $\mathbf{X}_\rho$.

4.3 APPROXIMATE EVALUATION

Section 2 explained how the value of a pure joint policy can be calculated. Unfortunately, the complexity of this calculation for a single pure joint policy scales exponentially with the planning horizon. Therefore we examine approximate evaluation by simulating a number of episodes, or *traces,* and using the average of outcomes as an estimate for the actual value $V(\pi)$. Although this approximation might introduce errors, notice that the CE method does not discriminate between policies within the set $\mathbf{X}_\rho$ of best samples. Therefore, as long as the relative ordering is preserved, the same policies are used to update the policy distribution, yielding the same results. In fact, only when the ranking of policies is disturbed near the cut-off threshold ρ, will approximate evaluation influence the distribution updating process. There is a second potential source of error, though. When the fraction of best samples $\mathbf{X}_\rho$ is used to update γ, the new γ might in fact be an over-estimation. This could make it very difficult to sample new instances with a higher (approximate) value.

5 Experiments

In this section we present some experimental results. First we describe a single-agent experiment performed in order to determine the influence of using different representations for policy distributions. Next, we show results on the multiagent Dec-Tiger problem, introduced by Nair et al. [2003]. We compare against dynamic programming JESP, a method introduced in the same paper, that works by alternatingly optimizing the policy of a single agent.

To determine the influence of the two policy representations, we devised the test problem illustrated in figure 1. In an effort to isolate the action history's influence as much as possible, the problem includes only one agent. Starting from s_0, the world changes (with equal probability) to either s_1 or s_2 if the agent takes action a_1, and to s_3 or s_4 if it takes action $\bar{a}_1$. When arriving in states s_1 and s_3, the agent (deterministically) receives observation o_1, in s_2 and s_4 this is $\bar{o}_1$. The difficulty in this problem is that the agent has to learn that performing action $\bar{a}_1$ after observation $\bar{o}_1$ is the best thing to do, but only after performing the optimal action a_1 at time step $t = 0$. This is hard because, as long as the distribution over the initial action $\xi_{()}$ has not converged, performing action $\bar{a}_1$ after $\bar{o}_1$ can result in a penalty of -1000. We tested the performance of the CE method for Dec-POMDPs on this problem, sampling four joint policies in each iteration, with $\rho = 0.5$. We used $\alpha = 0.2$, higher values resulted in faster learning, but also in more local optima. When using the CE method for policy search, the goal is to find (i.e., sample) good deterministic policies fast. Instead, here we are interested in the convergence behavior. A good indication of this convergence behavior is given by $V(f_\xi)$, the value of the induced stochastic policy throughout the CE process.

Figure 2(left) shows the mean and standard deviation of 100 runs of the CE method on the toy problem, for both OH- and AOH policy distributions (and each of those runs consists of 60 iterations). Upon convergence of f_ξ[3], the mean $V(f_\xi)$ using OH policy distributions is slightly lower (97.2 vs 103.5), and the standard deviation is greater (21.1 vs 9.12), both of which indicate that it gets stuck in sub-optimal local maxima more often. Also, the learning curve of the AOH based policy distributions grows faster. Even though the AOH representation seems to perform slightly better, the OH representation is still very useful; the latter requires less memory to store and has fewer parameters to update in each iteration, which means that it is faster. Experiments on enlarged versions of this problem showed similar results.

We also tested CE using exact and approximate evaluation on the 2-agent Dec-Tiger problem [Nair et al., 2003], including a variant that involves approximate evaluation without γ. This variant does not require that $\gamma^{(j+1)} \geq \gamma^{(j)}$, but always uses the best ρ-fraction to update the parameters, even if some of the policies have an approximate value worse than γ. Because we found little

[3] Note that the optimal policy always was discovered much earlier than that the distribution f_ξ converged.

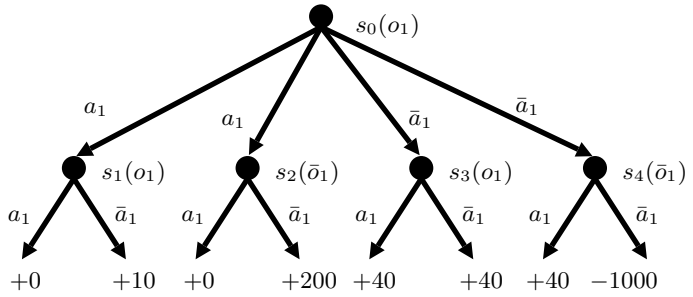
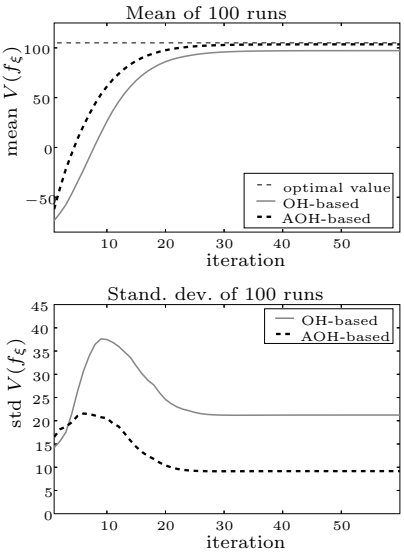


Fig. 1. The 1-agent test problem used to establish the influence of maintaining the action history.



h		CE	CE-A	CE-AW	JESP
2	max	−4.0	−4.0	−4.0	−4.0
	mean	−4.0	−4.0	−4.0	−18.30
	std	0.0	0.0	0.0	7.18
3	max	5.19	5.19	5.19	5.19
	mean	4.48	5.19	5.19	−19.35
	std	3.17	0.0	0.0	13.78
4	max	4.80	4.80	4.80	3.99
	mean	4.33	2.84	4.16	−20.49
	std	0.84	4.05	0.67	13.92
5	max	7.03	6.68	7.03	2.93
	mean	4.50	3.36	3.55	−26.73
	std	1.78	2.13	3.68	14.72
6	max	5.08	5.98	8.32	2.39
	mean	2.45	−2.63	1.66	−24.06
	std	1.28	4.51	3.11	11.86
7	max	-	−5.55	0.81	-
	mean	-	−22.28	−4.19	-
	std	-	14.00	3.12	-
8	max	-	−32.89	−14.22	-
	mean	-	−47.72	−20.79	-
	std	-	12.00	3.66	-

Fig. 2. Results for the toy problem of figure 1 (left), and results on the Dec-Tiger problem (right).

difference between the AOH- and OH-based representation, and the OH-based version is faster, we only used the latter representation in the experiments described here. All the CE variants run for a fixed number of 300 iterations and sampled 50 joint policies in each iteration of which they used a fraction of $\rho = 0.2$ to update the parameters. As before we set $\alpha = 0.2$. The approximate evaluation CE variants used 250 traces to estimate the value of the sampled policies (which was found to be roughly as good as using 1000 traces for $h = 4$).

Figure 2(right) shows the results of CE on the Dec-Tiger problem, using exact evaluation (CE), approximate evaluation (CE-A), approximate evalu-

ation without γ (CE-AW) and dynamic programming JESP. Shown are the maximum, mean and standard deviation of the value of the best pure policy sampled (out of all 300 iterations) over 20 runs. In case of approximate evaluation, the figure reports statistics over the exact values for the best ranked policies (found by an exact evaluation at the end). Empty entries indicate that the method was not able to complete all 20 runs within reasonable time. This immediately illustrates one of the advantages of the approximate evaluation CE methods: they were able to yield results up to horizon 8. For the horizons it did complete, exact evaluation CE performed very well. For horizon 2–5, it achieved the highest maximum, and for $h = 2, 4$–6, it has the highest mean and lowest variance, which indicates it reaches good solutions most frequently. JESP runs into problems for horizons greater than 6, especially memory requirements seem to be the bottleneck here. It is more striking, however, that its mean performance is significantly lower, which implies that it often gets stuck in worse local optima than the CE methods. Finally, the results show that CE-AW outperforms CE-A, implying that overestimating the lower bound γ indeed does negatively influence the search process. Closer analysis of the results confirms this: after an initial period of learning, CE-A has trouble sampling any policies with an (approximate) value higher than γ. This causes the set $\mathbf{X}_\rho$ to be empty, which in turn means there is no updating of the distribution, thus stalling the process.

6 Conclusions

In this paper we have shown how Dec-POMDPs can be approximately solved using an extension of the CE method. We discussed two different representations of probability distribution over joint policies, one based on the full action-observation history and one based only on the observation history, and how their parameters can be updated. Also we introduced an approximate, rather than exact evaluation of the policies sampled in each iteration.

An empirical evaluation showed that the CE approach to solving Dec-POMDPs is competitive with JESP, one of the state-of-the-art methods for approximately solving Dec-POMDPs. Moreover, this evaluation shows that the CE approach using the OH-based policy distributions performed nearly as good as with the AOH-based representation, and that good results can be obtained using approximate evaluation with relatively few traces. A last conclusion that can be drawn is that this approximate evaluation CE performs better when not enforcing a lower bound γ for the set $\mathbf{X}_\rho$ used to update the parameters.

There are a couple of directions for future research. More investigation is required regarding the influence of the action history. There might be problems where the effect of maintaining OH- rather than AOH-based policy distributions are more dramatic. Also the difference between OH- and AOH-based representations can be generalized by making the action distribution condi-

tional on the last k actions taken. Exact evaluation can most likely be accelerated by caching (intermediate) evaluation results of (parts of) joint policies. Finally, and somewhat related, the success of approximate evaluation raises the question whether it is necessary to sample complete joint policies if they are only partially inspected during approximate evaluation. The CE approach could greatly benefit from a construction that samples parts of (joint) policies.

Acknowledgments The research reported here is part of the Interactive Collaborative Information Systems (ICIS) project, supported by the Dutch Ministry of Economic Affairs, grant nr: BSIK03024.

References

D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein. The complexity of decentralized control of Markov decision processes. *Math. Oper. Res.*, 27 (4):819–840, 2002.

P.-T. de Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein. A tutorial on the cross-entropy method. *Annals of Operations Research*, 134(1):19–67, 2005.

R. Emery-Montemerlo, G. Gordon, J. Schneider, and S. Thrun. Game theoretic control for robot teams. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1175–1181, 2005.

C. V. Goldman and S. Zilberstein. Decentralized control of cooperative systems: Categorization and complexity analysis. *Journal of Artificial Intelligence Research (JAIR)*, 22:143–174, 2004.

D. Koller and A. Pfeffer. Representations and solutions for game-theoretic problems. *Artificial Intelligence*, 94(1-2):167–215, 1997.

S. Mannor, R. Rubinstein, and Y. Gat. The cross entropy method for fast policy search. In *International Conference on Machine Learning*, pages 512–519, 2003.

R. Nair, M. Tambe, M. Yokoo, D. V. Pynadath, and S. Marsella. Taming decentralized POMDPs: Towards efficient policy computation for multiagent settings. In *Proc. Int. Joint Conf. on Artificial Intelligence*, pages 705–711, 2003.

F. A. Oliehoek and N. Vlassis. Q-value functions for decentralized POMDPs. In *Proc. of Int. Joint Conf. on Autonomous Agents and Multi Agent Systems*, Honolulu, Hawai'i, 2007.

M. J. Osborne and A. Rubinstein. *A Course in Game Theory*. The MIT Press, July 1994.

M. T. J. Spaan and N. Vlassis. Perseus: Randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research*, 24:195–220, 2005.