Approximate Solutions for Factored Dec-POMDPs with Many Agents

Frans A. Oliehoek
Maastricht University
Maastricht, The Netherlands
frans.oliehoek@maastrichtuniversity.nl

Shimon Whiteson
University of Amsterdam
Science Park 904
Amsterdam, The Netherlands
s.a.whiteson@uva.nl

Matthijs T.J. Spaan
Delft University of Technology
Mekelweg 4
Delft, The Netherlands
m.t.j.spaan@tudelft.nl

ABSTRACT

Dec-POMDPs are a powerful framework for planning in multiagent systems, but are provably intractable to solve. Despite recent work on scaling to more agents by exploiting weak couplings in factored models, scalability for unrestricted subclasses remains limited. This paper proposes a factored forward-sweep policy computation method that tackles the stages of the problem one by one, exploiting weakly coupled structure at each of these stages. To enable the method to scale to many agents, we propose a set of approximations: approximation of stages using a sparse interaction structure, bootstrapping off smaller tasks to compute heuristic payoff functions, and employing approximate inference to estimate required probabilities at each stage and to compute the best decision rules. An empirical evaluation shows that the loss in solution quality due to these approximations is small and that the proposed method achieves unprecedented scalability, solving Dec-POMDPs with hundreds of agents.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent Systems*

General Terms

Algorithms, Performance

Keywords

Multi-agent planning, factored decentralized partially observable Markov decision processes.

1. INTRODUCTION

A key challenge of collaborative multiagent decision making is the presence of *imperfect information* [2]. Even in single-agent settings, incomplete knowledge of the environment's state (e.g., due to noisy sensors) complicates decision making. However, multiagent settings often exacerbate this problem, as agents have access only to their own sensors.

This paper focuses on the finite-horizon *decentralized partially observable Markov decision process* (Dec-POMDP) [2, 18], a model that can represent such problems under uncertainty. We consider factored Dec-POMDPs [21], which—by explicitly representing the structure present in the problem—open the door to methods that exploit this structure and thus scale to many more agents. While there have been positive results for specific subclasses that restrict the type of structure [1, 12, 33, 32], so far only moderate scalability has been achieved for general Dec-POMDPs. Since even finding an ϵ-approximate solution is NEXP-complete [24], any method that is guaranteed to be computationally efficient cannot guarantee an absolute bound on the error. Therefore, in this paper, we abandon optimality guarantees and aim for scalability in the number of agents.

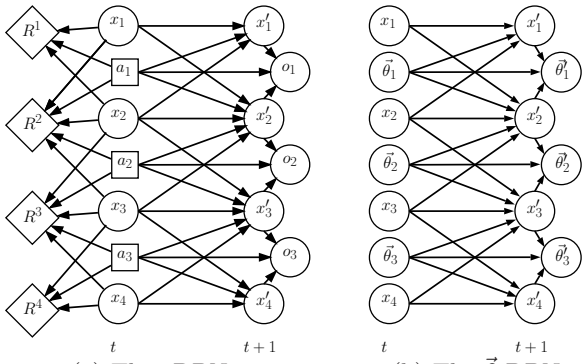
In particular, we propose an approximate approach based on *forward-sweep policy computation* (FSPC), a technique that approximates the Dec-POMDP by a series of one-shot *Bayesian games* (BGs), one for each stage [5, 20]. Our method, called *factored FSPC* (FFSPC), exploits weakly coupled structure at each stage by replacing the BGs with *collaborative graphical BGs* (CGBGs). The main algorithmic contribution of the paper is a set of approximations necessary to make FFSPC feasible for problems with many agents. First, we approximate the interaction structure between agents by constructing CGBGs with a predefined factorization. Second, instead of following the common practice of solving the underlying (PO)MDP, we employ a new class of heuristics based on *transfer planning* that directly approximate the factored Dec-POMDP value function. Third, we use approximate inference to efficiently construct the CGBGs. Finally, we approximately solve the CGBGs by applying *Max-Sum* to their *agent and type independence factor graphs*.

We present an extensive empirical evaluation that shows that FFSPC is highly scalable with respect to the number of agents, while attaining (near-) optimal values. A detailed analysis of our approximation indicates no significant decrease in value due to sparse factorization and approximate CGBG construction, and that the transfer planning heuristics significantly outperform two baselines. Finally, we test the limits of scalability of FFSPC, showing that it is able to scale to 1000 agents, in contrast to previous approaches for general factored Dec-POMDPs, which have not been demonstrated beyond 20 agents.

2. FACTORED DEC-POMDPS

In a Dec-POMDP, multiple agents must collaborate to maximize the sum of the common rewards they receive over multiple timesteps. Their actions affect not only their immediate rewards but also the state to which they transition. While the current state is not known to the agents, at each timestep each agent receives a private observation correlated

Appears in: *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2013), Ito, Jonker, Gini, and Shehory (eds.), May, 6–10, 2013, Saint Paul, Minnesota, USA.*



(a) The 2DBN. (b) The $\vec{\boldsymbol{\theta}}$-DBN.

Figure 1: The 2DBN and $\vec{\boldsymbol{\theta}}$-DBN (explained in Sec. 4.3) for FFG.

with that state. In a factored Dec-POMDP [21], the state consists of a vector of state variables and the reward function is the sum of a set of local reward functions.

A factored Dec-POMDP is a tuple $\langle \mathcal{D},\mathcal{S},\mathcal{A},T,\mathcal{R},\mathcal{O},O,\boldsymbol{b}^0,h \rangle$, where

- $\mathcal{D} = \{1,\ldots,n\}$ is the set of agents.
- $\mathcal{S} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_{|\mathcal{X}|}$ is the factored state space; $\mathcal{S}$ is spanned by a set $\mathcal{X} = \{\mathcal{X}_1,\ldots,\mathcal{X}_{|\mathcal{X}|}\}$, of state variables or *factors* and a state is an assignment of all factors $s = \langle x_1,\ldots,x_{|\mathcal{X}|} \rangle$.
- $\mathcal{A} = \times_i \mathcal{A}_i$ is the set of *joint actions*, where $\mathcal{A}_i$ is the set of actions available to agent i.
- T is a transition function specifying the state transition probabilities $\Pr(s'|s,\mathbf{a})$.
- $\mathcal{R} = \{R^1,\ldots,R^\rho\}$ is the set of ρ local reward functions. These correspond to an interaction graph [16] with (hyper-) edges $\mathcal{E}$ such that the total immediate reward $R(s,\mathbf{a}) = \sum_{e\in\mathcal{E}} R^e(\mathbf{x}_e,\mathbf{a}_e)$.
- $\mathcal{O} = \times_i \mathcal{O}_i$ is the set of joint observations $\mathbf{o} = \langle o_1,\ldots,o_n \rangle$.
- O is the observation function, which specifies observation probabilities $\Pr(\mathbf{o}|\mathbf{a},s')$.
- $\boldsymbol{b}^0$ is the initial state distribution at time $t = 0$.
- h is the horizon, i.e., the number of stages. We consider the case where h is finite.

The goal of planning in a Dec-POMDP is to find a joint policy $\boldsymbol{\pi} = \langle \pi_1,\ldots,\pi_n \rangle$ that maximizes the expected cumulative reward. A policy π_i specifies an action for each action-observation history (AOH) $\vec{\theta}_i^t = (a_i^0,o_i^1,\ldots,a_i^{t-1},o_i^t)$, (so $\pi_i(\vec{\theta}_i^t) = a_i^t$), but without loss of optimality we can restrict our search to deterministic policies that map each observation history $(o_i^1,\ldots,o_i^t) = \vec{o}_i^t \in \vec{\mathcal{O}}_i^t$, for each stage t, to an action: $\pi_i(\vec{o}_i^t) = a_i^t$. A decision rule δ_i^t of agent i is the part of its policy that specifies actions just for stage t. That is, δ_i^t maps observation histories $\vec{o}_i^t$ to actions a_i.

The transition and observation model in a factored Dec-POMDP can be compactly represented in a *two-stage dynamic Bayesian network (2DBN)* [3]. Fig. 1 shows the 2DBN for 'fire-fighting graph' (FFG), a benchmark problem where a team of agents must put out as many fires as possible but each agent can fight fire only at its two nearest houses [17].

Assuming a Dec-POMDP is factored is not restrictive since every Dec-POMDP can be converted to a factored one with just one factor. However, the formalism is most useful when the problem is *weakly coupled*, i.e., when the 2DBN contains sufficient conditional independence. This paper focuses on approximate methods that effectively exploit such independence to solve Dec-POMDPs with large numbers of agents. Because we do not make any restrictive assumptions about the factored Dec-POMDP, our approach is completely general, though the amount of expected speedup depends on how much structure is present.

3. FACTORED FSPC

Forward-sweep policy computation (FSPC) is an approximate solution method for (non-factored) Dec-POMDPs that works by solving a series of collaborative Bayesian games (CBGs), one for each stage of the Dec-POMDP [5, 20]. In this section, we extend FSPC to the factored setting to create FFSPC, which replaces the CBGs from FSPC with *collaborative graphical Bayesian games (CGBGs)* [21, 22]. Doing so opens the door to exploiting (conditional) independence between agents and thereby potentially much better scalability in the number of agents. The possibility of FFSPC was already mentioned in [21], but this paper develops and empirical evaluates this idea. Moreover, FFSPC depends on the availability of a factored heuristic, which has not been proposed, and in its naive form is still limited to small numbers of agents. In this paper we propose a number of approximation techniques that enable a novel, scalable, version of FFSPC. A more detailed description of some of these proposed ideas can be found in [17].

At a high level, FFSPC works by repeatedly constructing and solving CGBGs for consecutive stages. It starts by constructing a CGBG for $t = 0$, the solution to which is a joint decision rule $\boldsymbol{\delta}^0$. This $\boldsymbol{\delta}^0$ induces a partial joint policy $\boldsymbol{\varphi}^1 = (\boldsymbol{\delta}^0)$, which in turn induces a CGBG for the next stage $t = 1$. Solving this CGBG leads to $\boldsymbol{\delta}^1$, and hence $\boldsymbol{\varphi}^2 = (\boldsymbol{\delta}^0,\boldsymbol{\delta}^1)$, etc. The algorithm ends when the CGBG for the last stage $t = h-1$ is solved, at which point a full joint policy $\boldsymbol{\pi} = (\boldsymbol{\delta}^0,\ldots,\boldsymbol{\delta}^{h-1})$ has been constructed.

The CGBG for stage t, given $\boldsymbol{\varphi}^t$, consists of the following components: 1) the set of agents $\mathcal{D}$ and their joint actions $\mathcal{A}$ are the same as in the Dec-POMDP, 2) given $\boldsymbol{\varphi}^t = \langle \varphi_1^t,\ldots,\varphi_n^t \rangle$, each agent i has a set of AOHs $\vec{\theta}_i^t$ consistent with its past policy φ_i^t (called *types* in Bayesian game terminology), 3) a probability over joint AOHs $\Pr(\vec{\boldsymbol{\theta}}^t|\boldsymbol{b}^0,\boldsymbol{\varphi}^t)$, and 4) a set of local heuristic payoff functions $\{Q^1_{\boldsymbol{\varphi}^t},\ldots,Q^\rho_{\boldsymbol{\varphi}^t}\}$.

As in regular FSPC, $\Pr(\vec{\boldsymbol{\theta}}^t|\boldsymbol{b}^0,\boldsymbol{\varphi}^t) = \sum_{s^t} \Pr(s^t,\vec{\boldsymbol{\theta}}^t|\boldsymbol{b}^0,\boldsymbol{\varphi}^t)$. However, discussed in Sec. 4, we also use local joint AOH probabilities $\Pr(\vec{\boldsymbol{\theta}}_e^t|\boldsymbol{b}^0,\boldsymbol{\varphi}^t)$. Regular FSPC relies on *heuristic Q-value functions*, such as $\mathrm{Q}_{\mathrm{MDP}}$, $\mathrm{Q}_{\mathrm{POMDP}}$ and Q_{BG}, to specify the heuristic payoff function $Q(\vec{\boldsymbol{\theta}}^t,\mathbf{a})$ [20]. In FFSPC, however, the *local* payoff functions depend on the actions and AOHs of a subset of agents and their sum is the expected cumulative reward for the remaining stages: $Q(\vec{\boldsymbol{\theta}}^t,\mathbf{a}) = \sum_{e\in\mathcal{E}} Q^e_{\boldsymbol{\varphi}^t}(\vec{\boldsymbol{\theta}}_e^t,\mathbf{a}_e)$. This means that FFSPC requires a *factored* Q-value function to use as its heuristic, but so far no practical candidates have been proposed. Still, we know that such a decomposition is possible since the Q-function of every joint policy is factored [21]. However, in general, each local component Q^e of such a Q-function depends on a subset of state factors (the *state factor scope* $\mathbb{X}(e)$) and the AOHs and actions of a subset of agents (the *agent scope* $\mathbb{A}(e)$): $Q^e(\mathbf{x}^t_{\mathbb{X}(e)},\vec{\boldsymbol{\theta}}^t_{\mathbb{A}(e)},\mathbf{a}_{\mathbb{A}(e)})$ (abbreviated $Q^e(\mathbf{x}_e^t,\vec{\boldsymbol{\theta}}_e^t,\mathbf{a}_e)$). Fortunately, given a past joint policy $\boldsymbol{\varphi}^t$, we

can construct the local payoff functions for the CGBG from these local Q-functions:

$$Q^e_{\boldsymbol{\varphi}^t}(\vec{\boldsymbol{\theta}}^t_e,\mathbf{a}_e) = \sum_{\mathbf{x}^t_e} \Pr(\mathbf{x}^t_e|\vec{\boldsymbol{\theta}}^t_e,\boldsymbol{b}^0,\boldsymbol{\varphi}^t) Q^e(\mathbf{x}^t_e,\vec{\boldsymbol{\theta}}^t_e,\mathbf{a}_e). \quad (3.1)$$

The solution of a CGBG is thus given by:

$$\boldsymbol{\delta}^{t*} = \arg\max_{\boldsymbol{\delta}^t} \sum_{e\in\mathcal{E}} \sum_{\vec{\boldsymbol{\theta}}^t_e} \Pr(\vec{\boldsymbol{\theta}}^t_e|\boldsymbol{b}^0,\boldsymbol{\varphi}^t) Q^e_{\boldsymbol{\varphi}^t}(\vec{\boldsymbol{\theta}}^t_e,\boldsymbol{\delta}^t_e(\vec{\boldsymbol{\theta}}^t_e)), \quad (3.2)$$

where $\boldsymbol{\delta}^t_e(\vec{\boldsymbol{\theta}}^t_e)$ is the local joint action under policy $\boldsymbol{\delta}^t$ given local joint type $\vec{\boldsymbol{\theta}}^t_e$. The CGBG can be solved optimally using *non-serial dynamic programming* (NDP) by exploiting independence in the CGBG but this is efficient only when the corresponding factor graph has low *induced width* [21].

Although there are no guarantees when applying FFSPC with an approximate Q-value function as the heuristic, it is in principle an exact method: it yields the optimal solution when using an optimal Q-value function [21]. Also, (factored) FSPC is closely related to variants of multiagent A* [28, 20]; these methods compensate for lack of an optimal heuristic by using an admissible heuristic in combination with backtracking. However, multiagent A* cannot scale one horizon beyond simple brute-force search [26] because it relies on full expansion of all non-leaf nodes to allow the backtracking.[1] FSPC does not backtrack, and hence does not suffer from this problem. Moreover, despite the lack of backtracking, it still affords good solutions in many cases [20], making it a good starting point for an efficient approximate method.

4. FACTORED FSPC FOR MANY AGENTS

The main goal of this paper is to develop an approximate method for factored Dec-POMDPs that can scale to many agents. Because it exploits structure between agents, FFSPC has the potential to scale in this fashion. However, there are a number of barriers that prevent a direct application of FFSPC to Dec-POMDPs with many agents from being feasible. The main contribution of this paper is a set of approximations to overcome these barriers.

In particular, this section describes four barriers and proposes an approximation to address each one. As a result of these heuristic approximations, no theoretical guarantees about FFSPC's solution quality can be made. However, due to the immense difficulty of the Dec-POMDP setting and the fact that computing even bounded approximations is NEXP-complete [24], such heuristics are necessary to construct an efficient and scalable method. In addition, we demonstrate in Section 5 that, in practice, this approach enables the (near-)optimal solution of Dec-POMDPs with many more agents than would otherwise be possible.

4.1 Predetermined Scope Structure

While FFSPC is exact when using an optimal factored Q-value function Q^*, computing such a Q^* is intractable, just as in the non-factored setting [20]. However, even if we could compute Q^*, the scopes of its components Q^e would contain many (or all) agents and state factors for earlier stages, because these scopes grow when performing value backups from the last stage towards the earlier stages [21]. Consequently, the first barrier to making FFSPC scalable is that, even if we could compute Q^*, the resulting CGBGs would not be efficiently solvable. Therefore, we must approximate not only the values of the Q-function, but *also its factorization*.

To this end, we propose to use factored Q-value functions with predetermined *scope structure*, i.e., for each stage we specify the scope for each component of the Q-function. For instance, such a scope structure could simply use the immediate reward scopes at each stage. Alternatively, it could use the optimal scopes for the last two stages (i.e., immediate reward scopes for $t = h-1$ and the back-projection of the immediate reward scopes for $t = h-2$) and then continue to use the scopes specified for $h-2$ for the remaining stages.

To understand the motivation for this approximation, consider an extreme example of weakly coupled structure: transition and observation independent (TOI) Dec-POMDPs. In such problems, agents have their own local states and cannot influence the transitions or observation of one another. As a result, in the TOI case (e.g., in ND-POMDPs), optimal scopes equal those of the factored immediate reward function [16, 21]. However, it is likely that in many cases where agents are weakly coupled (but not completely TOI), the influence of agent i's actions on a particular state factor decreases with the number of links by which a_i is separated from it in the 2DBN. If so, there are many cases for which a suitable scope structure affords a good approximation. In this sense, our approach is similar to that of work on factored (PO)MDPs [11, 8], which assumes that the value function of a factored MDP is 'close to factored'. Following the literature on factored MDPs, we use manually specified scope structures (this is equivalent to specifying basis functions). Developing methods for finding such scope structures automatically is an important goal for future work.

4.2 Transfer Planning

A second barrier to scaling FFSPC to many agents is that the heuristic payoff functions such as $\mathrm{Q_{MDP}}$ and $\mathrm{Q_{POMDP}}$ that are typically used for small non-factored problems can no longer be computed, since solving the underlying MDP or POMDP is intractable. In fact, the problem is even more severe for sparsely connected factored Dec-POMDPs: heuristic payoff functions such as $\mathrm{Q_{MDP}}$ and $\mathrm{Q_{POMDP}}$ become fully coupled through just one backup (due to the maximization over joint actions that is conditioned on the state or belief), i.e., these value functions are 'less factored' than an optimal factored Q^* for the Dec-POMDP.

Therefore, since our aim is to approximate factored Dec-POMDPs with Q-functions of restricted scope, these heuristics might not be very useful even if we could compute them. Instead, we propose to compute heuristic payoff functions directly using a new approach, called *transfer planning (TP)*. TP computes heuristic values $Q^e_{\boldsymbol{\varphi}^t}(\vec{\boldsymbol{\theta}}^t_e,\mathbf{a}_e) \triangleq Q^\sigma(\vec{\boldsymbol{\theta}}^t,\mathbf{a})$ by solving similar tasks σ with fewer agents and using their value functions Q^σ.

We call this approach transfer planning because it is inspired by *transfer learning* [29]. In reinforcement learning, transfer learning methods often use value functions learned on a set of source tasks to construct an initial value function for a related target task, which is then refined by further learning. In contrast, we consider a planning task and use the value functions of smaller source tasks as heuristics for the larger target task.

For simplicity, in the following we assume some particular

[1] Incremental expansion [27] can mitigate this problem.

Table 1: Scopes of 5-agent FFG. Left: immediate reward scopes, right: reduced scopes of Q^*.

(a) Non-reduced R.

e	$\mathbb{A}(R^e)$	$\mathbb{X}(R^e)$
1	{1}	{1,2}
2	{1,2}	{1,2,3}
3	{2,3}	{2,3,4}
4	{3,4}	{3,4,5}
5	{4,5}	{4,5,6}
6	{5}	{5,6}

(b) Reduced Q^* scopes.

t	e	$\mathbb{A}(e,t)$	$\mathbb{X}(e,t)$
$h-3$	1	{1,2,3,4,5}	{1,2,3,4,5,6}
$h-2$	1	{1,2,3,4}	{1,2,3,4}
	2	{2,3,4,5}	{2,3,4,5,6}
$h-1$	1	{1,2}	{1,2,3}
	2	{2,3}	{2,3,4}
	3	{3,4}	{3,4,5}
	4	{4,5}	{4,5,6}

stage t. We formalize TP using the following components:

- Σ, the set of source problems $\sigma \in \Sigma$;
- $\mathcal{D}^\sigma = \{1^\sigma, \ldots, n^\sigma\}$, the agent set for source problem σ;
- E maps each Q-value component e to a source problem $E(e) = \sigma \in \Sigma$;
- $A^e : \mathbb{A}(e) \to \mathcal{D}^{E(e)}$ maps agent indices $i \in \mathbb{A}(e)$ in the target task to indices $A^e(i) = j^\sigma \in \mathcal{D}^\sigma$ in the source task σ for that particular component $\sigma = E(e)$.

With some abuse of notation, we overload A^e to also work on profiles of agents, actions and histories, e.g., we write $\langle \vec{\theta}^t_{A^e(i)} \rangle_{i \in e} = \vec{\boldsymbol{\theta}}^t_{A^e(e)}$. Given a set of source problems and some (heuristic) Q-value functions for them, the transfer planning Q-value function Q_{TP} is defined as

$$Q^e_{\mathrm{TP}}(\vec{\boldsymbol{\theta}}^t_e, \mathbf{a}_e) \triangleq Q^\sigma(\vec{\boldsymbol{\theta}}^t_{A^e(e)}, \mathbf{a}_{A^e(e)}), \qquad \sigma = E(e). \tag{4.1}$$

Thus, we only have to define the source problems, define the corresponding mapping A^e for each component e, and find a heuristic Q-value function $Q^\sigma(\vec{\boldsymbol{\theta}}^t, \mathbf{a})$ for each source problem. Since the source problems are typically chosen to be small, we can treat them as non-factored and use the heuristic $\mathrm{Q}_{\mathrm{MDP}}$, $\mathrm{Q}_{\mathrm{POMDP}}$ and Q_{BG} value functions [20], or even the true Dec-POMDP value functions.

Since no formal claims can be made about the resulting approximate Q-values, we cannot guarantee that they constitute an admissible heuristic. However, since we rely on FSPC, which does not backtrack, an admissible heuristic is not necessarily better. Performance depends on the accuracy, not the admissibility of the heuristic [20]. The experiments we present in Section 5 demonstrate that these approximate Q-values are accurate enough to enable high quality solutions.

Example 1. To illustrate the application of Q_{TP}, we consider the 5-agent FFG problem, which has 6 houses. However, the immediate reward scopes of the first and last house are sub-scopes of other scopes as illustrated in Table 1(a) and as such we can reduce them such that the desired Q-value function will be factored as shown in Table 1(b) (at $h-1$). We propose to use 2-agent FFG as the source task for each of the four components. The agent mapping is defined such that the agent with the lower index is mapped to 1^σ and the agent with the higher index to 2^σ. E.g., for $e = 3$, $A^e(3) = 1^\sigma$,$A^e(4) = 2^\sigma$. Finally, computation of, e.g., $\mathrm{Q}_{\mathrm{MDP}}$ for 2-agent FFG provides the values to define Q^e_{TP} using (4.1).

4.3 Approximate Inference

The third barrier to scaling FFSPC lies in the marginalization required to compute the probabilities needed to construct the CGBG for stage t. In particular, constructing each CGBG requires generating each component e separately. However, as (3.1) shows, in general this requires the probabilities $\Pr(\mathbf{x}^t_e | \vec{\boldsymbol{\theta}}^t_e, \boldsymbol{b}^0, \boldsymbol{\varphi}^t)$. Moreover, in any efficient solution algorithm for Dec-POMDPs, the probabilities $\Pr(\vec{\boldsymbol{\theta}}^t_e | \boldsymbol{b}^0, \boldsymbol{\varphi}^t)$ are necessary, as illustrated by (3.2). Since maintaining and marginalizing over $\Pr(s, \vec{\boldsymbol{\theta}}^t | \boldsymbol{b}^0, \boldsymbol{\varphi}^t)$ is intractable, we resort to approximate inference, as is standard practice when computing probabilities over states with many factors. Such methods perform well in many cases and their error can be theoretically bounded [4].

In particular, we use the factored frontier algorithm [15] to perform inference on a DBN, which we call $\vec{\boldsymbol{\theta}}$-DBN, constructed for the past joint policy $\boldsymbol{\varphi}^t$ under concern. This $\vec{\boldsymbol{\theta}}$-DBN, illustrated in Fig. 1(b), models stages $0, \ldots, t$ and has both state factors and action-observation histories as its nodes. The dynamics of the $\vec{\boldsymbol{\theta}}$-DBN follow from those of the original DBN given $\boldsymbol{\varphi}^t$. This formulation can be further generalized to allow influence of other agents' actions and observations on o_i^{t+1} and thus $\vec{\theta}_i^{t+1}$. For simplicity, we consider only the above setting here.

Starting at $t = 0$, factored frontier represents the distribution in a fully factored form, i.e., as the product of marginal probabilities $p(x_i^t)$ and $p(\vec{\theta}_i^t)$. Given such a distribution for a stage t, the next-stage distribution can be approximately computed by directly computing the new marginals on each node X_i^{t+1} (either x_i^{t+1} or $\vec{\theta}_i^{t+1}$): the node's CPT is multiplied by the marginals of its parents $X^t_{P_1} \ldots X^t_{P_l}$ and the parents are marginalized out directly. Given the completely factored distribution as computed by factored frontier, we can estimate local probabilities using

$$\forall_e \qquad \Pr(\mathbf{x}^t_e, \vec{\boldsymbol{\theta}}^t_e | \boldsymbol{b}^0, \boldsymbol{\varphi}^t) \approx \prod_{i \in \mathbb{X}(Q^e)} p(x_i) \prod_{i \in \mathbb{A}(Q^e)} p(\vec{\theta}_i^t). \tag{4.2}$$

However, the special structure of the $\vec{\boldsymbol{\theta}}$-DBN enables a slightly more accurate representation that makes use of the intermediate results of the factored frontier algorithm. This is particularly important when using heuristic Q-functions of the form $Q^e(\mathbf{x}^t_e, \mathbf{a}^t_e)$ (as do some baselines in our experimental evaluation): using (4.2) would then result in a heuristic that specifies the same payoff for all $\vec{\boldsymbol{\theta}}^t_e$ (cf. (3.1)).

To overcome this problem, we propose to perform the computation of the new marginals in a fixed order: first the marginals of all state factors x_j, then the marginals of all $\vec{\theta}_i$. These latter $\vec{\theta}_i$ marginals are computed in two phases. The first phase marginalizes only over the parents in the *previous* time slice. This leads to a *nearly* completely factored distribution p:

$$p(s, \vec{\boldsymbol{\theta}}) = \prod_{j=1}^{|\mathcal{X}|} p(x_j) \cdot \prod_{i=1}^{n} p(\vec{\theta}_i | \mathbf{x}_{\mathcal{I}}), \tag{4.3}$$

where $\mathbf{x}_{\mathcal{I}}$ denotes the set of state factors *from the same time slice* that influence the probability of $\vec{\theta}_i$. This distribution can now be used to compute $\Pr(\mathbf{x}^t_e | \vec{\boldsymbol{\theta}}^t_e, \boldsymbol{b}^0, \boldsymbol{\varphi}^t)$ in (3.1). In the second phase, the $\mathbf{x}_{\mathcal{I}}$ are also marginalized out.

4.4 Approximate Solution of CGBGs

The fourth barrier to scaling FSPC is the need to efficiently solve the large CGBGs constructed as described above. While optimal solutions can be computed via nonserial dynamic programming (NDP) [21], this approach will

not scale well, as it constructs a factor graph where the variables are entire decision rules, the number of which grows doubly exponentially with the horizon. Moreover, NDP scales exponentially in the induced width of the graph, which can be large if there are many agents.

However, in recent work we showed that CGBGs contain additional structure called *type independence* [17, 22]: for a particular agent, the action choice for one AOH is conditionally independent of the action choice for another AOH. While this structure cannot be exploited effectively by NDP because it induces a high width, the popular Max-Sum algorithm [10, 7, 9, 25] can effectively exploit the structure via an *agent and type independence (ATI)* factor graph [22]. Here, each variable is the action taken for a particular type (i.e., AOH) of a particular agent, and each factor is the contribution (to the expected value) of a local joint type. The resulting solution method is exponential only in a parameter k, the maximum number of agents participating in a payoff function (but not in the number of histories), and was empirically shown to lead to (near-)optimal solutions at low computational costs on a range of CGBGs [22]. We employ this combination of Max-Sum applied to the ATI factor graph to solve the CGBGs in FFSPC.

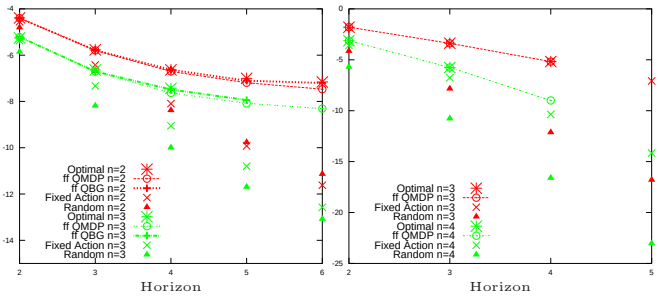
5. EXPERIMENTS

We evaluate FFSPC on two problem domains: FFG and ALOHA [17]. The latter consists of a number of islands that each need to regularly transmit radio messages to its local population. However, transmissions of neighboring island will lead to collisions. We consider a number of island configurations where the islands form a line and one with four islands in a square configuration. ALOHA is considerably more complex than FFG for several reasons. First, it has 3 observations per agent (transmission, no transmission, collision), which means that the number of observation histories grows much faster. Also, the transition model of ALOHA is more densely connected than FFG: the reward component for each island is affected by the island itself and by all its neighbors. As a result, in all the ALOHA problems we consider, there is at least one immediate reward function whose scope contains 3 agents, i.e., $k = 3$. For a more detailed description of ALOHA see [17].

In all cases, we use immediate reward scopes that have been reduced (i.e., scopes that form a proper sub-scope of another scope are removed) before computing the factored Q-value functions. This means that for all stages, the factored Q-value function has the same factorization.

To compute $\mathrm{Q_{TP}}$, the transfer-planning heuristic, for the FFG problem, we use 2-agent FFG as the source problem for all the components and map the lower agent index in a scope to agent 1 and the higher index to agent 2. For ALOHA, we use the 3-island in-line variant as the source problem and perform a similar mapping (e.g., the lowest agent index in a scope is mapped to agent 1, the middle to agent 2 and the highest to agent 3).

To test the efficacy of transfer planning, we compare to two baselines. The first, *naive regression* (NR), computes the non-factored $\mathrm{Q_{MDP}}(s,\mathbf{a})$ and then applies linear regression to find the least-squares factored approximation of the form $Q = \sum_e Q^e(\mathbf{x}_e,\mathbf{a}_e)$. While this yields a factored $\mathrm{Q_{MDP}}$, it scales poorly: the numbers of joint actions and states in the underlying MDP grow exponentially with the numbers of agents and state factors, respectively. Hence, both the solution of the underlying MDP and the regression problem become intractable. The second approach avoids exact solution of the underlying MDP by bootstrapping in an *approximate dynamic programming* (ADP) algorithm. We extended techniques from [11] to enable efficient regression.



(a) FFG. (b) ALOHA.

Figure 2: FFSPC (ff) solution quality compared to optimal and the baselines.

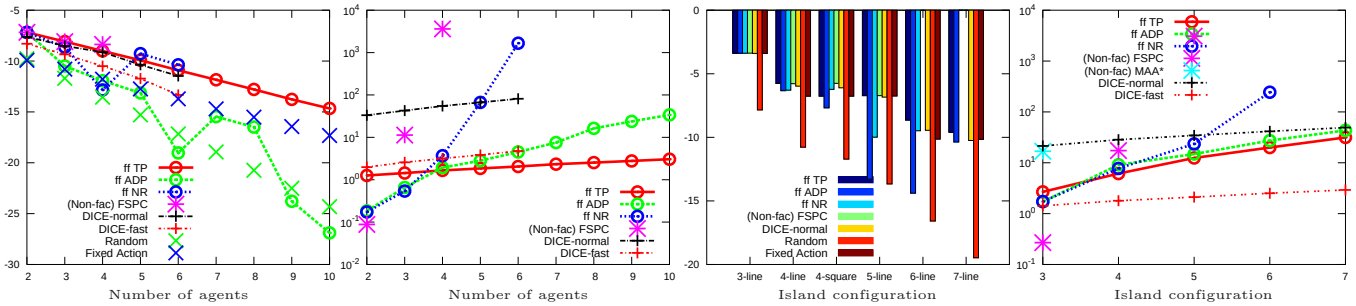
Reported timing results are CPU times with a resolution of 0.01s. Each method receives 1 hour of wall-clock time in which to compute solutions for all considered horizons (typically $h = 2,\ldots,5$ or 6). The reported statistics are means over 10 restarts of each method. Once joint policies have been computed, we perform 10,000 simulation runs to estimate their true values.

5.1 Comparison to Optimal Methods

For problems small enough to solve optimally, we compare the solution quality of FFSPC to that of GMAA*-ICE, a state-of-the-art method for optimally solving DecPOMDPs [27]. We use TP with the $\mathrm{Q_{MDP}}$ and $\mathrm{Q_{BG}}$ heuristic for the source problems. As baselines, we include a joint policy that selects actions uniformly at random, and the best joint policy in which each agent selects the same fixed action for all possible histories (though the agents can select different actions from each other).

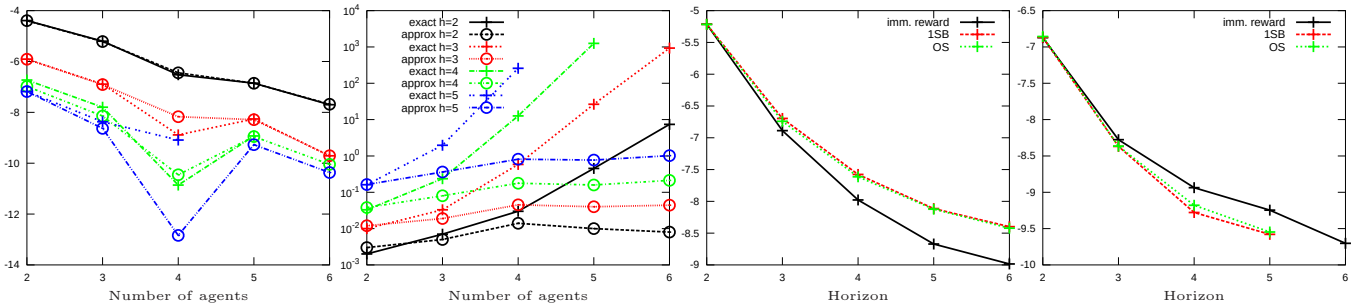
Fig. 2 compares FFSPC's solutions to optimal solutions on both problems. Fig. 2(a) show the results for FFG with two (red) and three agents (green). Optimal solutions were computed up to $h = 6$ in the former and $h = 4$ in the latter. FFSPC with $\mathrm{Q_{TP}}$ heuristic and using $\mathrm{Q_{BG}}$ for the source problem ('ff QBG') achieves the optimal value for all these instances. When using the $\mathrm{Q_{MDP}}$ TP heuristic ('ff QMDP'), results are near optimal. For three agents, the optimal value is available only up to $h = 4$. Nonetheless, the curve of FFSPC's values has the same shape of as that of the optimal values for two agents, which suggests these points are near optimal as well. While the fixed action baseline performs relatively well for shorter horizons, it is worse than random for longer horizons because there always is a chance that the unselected house will keep burning forever.

Fig. 2(b) shows results for ALOHA. The $\mathrm{Q_{BG}}$ TP heuristic is omitted since it performed the same as $\mathrm{Q_{MDP}}$. For all settings at which we could compute the optimal value, FFSPC matches this value. Since the ALOHA problem is more complex, FFSPC, in its current form, runs out of memory for higher horizons, because the number of local joint types for each component grows quickly in the ALOHA problem. For instance, since the immediate reward scopes



(a) FFG, Value, $h = 5$ (b) FFG, Runtime, $h = 5$ (c) ALOHA, Value, $h = 3$ (d) ALOHA, Runtime, $h = 3$

Figure 3: A comparison of FFSPC (ff) with different heuristics and other methods on FFG and Aloha.



(a) Value. (b) Runtime. (c) 3 agents. (d) 5 agents.

Figure 4: FFG problem. (a), (b) Influence of approximate inference. (c), (d) Comparison of different scopes.

contain 3 agents, the 3-agent problem is fully coupled. Furthermore, the agents have 3 observations, yielding $3^3 = 27$ observation histories (and thus types) per agent, for stage $t = 3$. This in turn yields $27^3 = 19{,}683$ local joint types (and thus factors) associated with each component of a CGBG for $t = 3$. In addition, the fixed action baseline performs surprisingly well, performing optimally for 3 islands and near optimally for 4 islands. As with FFG, we expect that it would perform worse for longer horizons: if one agent sends messages for several steps in a row, its neighbor is more likely to have messages backed up in its queue.

5.2 Comparison to Approximate Methods

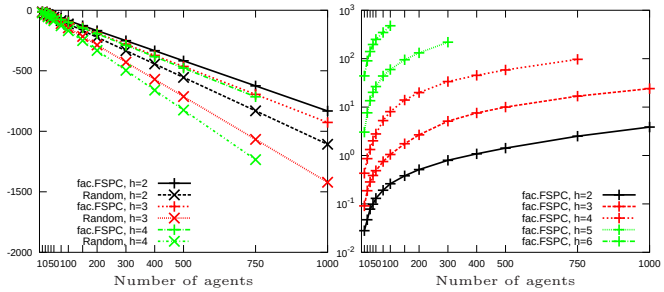
Our goal in proposing FFSPC is to create an algorithm that is scalable in terms of the number of agents. However, when increasing the number of agents, optimal methods break down. Therefore, we perform a comparison to other approximate methods, including non-factored FSPC and direct cross-entropy policy search (DICE) [19], one of the few methods demonstrated to work on Dec-POMDPs with more than three agents that are not transition and observation independent. For non-factored FSPC, we use alternating maximization with 10 restarts to solve the CBGs. For DICE we use two parameter settings known as DICE-normal and DICE-fast. In addition, in these experiments we compare the TP heuristic (using Q_{MDP}) with ADP and NR.

Fig. 3(a) and (b) show the results for FFG with $h = 5$. For all numbers of agents except 4, FFSPC finds solutions as good as or better than those of non-factored FSPC, DICE-normal, DICE-fast, and the fixed-action and random baselines. In addition, its running time scales much better than that of non-factored FSPC and even the fixed-action baseline. Hence, this result highlights the complexity of the problem, as even a simple baseline scales poorly. FFSPC also runs substantially more quickly than DICE-normal and slightly more quickly than DICE-fast, both of which run out of memory when there are more than five agents. The figures show that TP is faster than ADP and provides much better solutions. For 5 and 6 agents, NR finds slightly better solutions than TP, but it performs very badly for 4 agents (investigation indicated that NR needs to solve an ill-conditioned systems of equations for 4 agents). Also, NR scales poorly.

Fig. 3(c) and (d) present a similar comparison for ALOHA with $h = 3$. DICE-fast yielded poor results and is omitted from these plots. Fig. 3(c) shows that the value achieved by FFSPC matches or nearly matches that of all the other methods on all island configurations. TP, especially, performs consistently very well, while ADP and NR have instances with poor value. Fig. 3(d) shows the runtime results for the inline configurations. While the runtime of FFSPC is consistently better than that of DICE-normal, non-factored FSPC and the fixed-action baseline are faster for fewer agents. Non-factored FSPC is faster for 3 agents because the problem is fully coupled: there are 3 local payoff functions, 2 with 2 agents and one with 3 agents (so $k = 3$). Thus FFSPC incurs the overhead of dealing with multiple factors and constructing the FG but gets no speedup in return. However, the runtime of FFSPC scales better in the number of agents.

5.3 Analysis of Factored FSPC

The experiments described above evaluate FFSPC against a large number of baselines and compare the TP heuristic against the ADP and NR alternatives, on the FFG problem. Here we report experiments that analyze the other proposed approximations.



(a) Value. $h = 2 \ldots 4$. (b) Runtime. $h = 2 \ldots 6$.

Figure 5: FFSPC results on FFG with many agents.

To test the influence of approximate inference, we compare against a variant of FACTORED FSPC that uses exact inference. For this comparison, the NR heuristic is used because it gives fairly good results while depending on inference of *both* the states factors and histories to compute the resulting heuristic, as explained in Sec. 4.3. Fig. 4(a) and (b) show the results. In Fig. 4(b), exact inference clearly suffers from exponential scaling with respect to the runtime while approximate inference behaves much better. Fig. 4(a) shows that this increase in runtime efficiency comes at little cost: typically there is little or no loss in value, except for 4 agents. As mentioned, for 4 agents the resulting linear systems become ill conditioned, leading to very large Q-values, and thus a change in inference can have a large impact on the solution. However, these figures also show that this change can go in either direction: for $h = 5$ exact inference is much better, but approximate inference performs better for $h = 3, 4$.

We also performed experiments to establish the influence of using different scopes, shown in Fig. 4(c) and (d). In particular, we compared the use of immediate reward scopes to two alternatives: optimal scopes (OS) and 1-step back projected scopes (1SB). For 3 agents, shown in Fig. 4(c), OS and 1SB are the same, since 1 back-projection fully couples the problem. For 5 agents, and for $h \geq 3$, OS and 1SB actually specify different scope structures (as illustrated by Table 1(b)), which allows for different values. Still, Fig. 4(d) shows that the performance is often the same for these settings too. Using OS or 1SB can result in slightly higher payoffs than immediate reward scopes, as in Fig. 4(c). However, the reverse also occurs, as shown in Fig. 4(d).

The approximations introduced by approximate solutions of the CGBGs are hard to quantify, since the CGBGs of the sizes we consider cannot be tackled by any other methods than Max-Sum. However, for small CGBGs the method was shown to perform near-optimally [22].

5.4 Scaling Up

To determine the limits of the scalability of FFSPC in the number of agents, we conducted additional experiments applying FFSPC with the Q_{MDP} TP heuristic to FFG with many more agents. The results, shown in Fig. 5, do not include a fixed action baseline because it does not scale to the considered problem sizes (it requires performing simulations for all considered fixed action joint policies which becomes expensive for many agents and the number of such joint policies grows exponentially with the number of agents).

As shown in Fig. 5(a), FFSPC successfully computed solutions for up to 1000 agents for $h = 2, 3$ and 750 agents for $h = 4$. For $h = 5$, it computed solutions for up to 300 agents; even for $h = 6$ it computed solutions for 100 agents, as shown in Fig. 5(b). Due to lack of space, we omit the values for $h = 5, 6$, but we found that that, for the computed entries for $h = 6$, the expected value is roughly equal to $h = 5$. This implies that the probability of any fire remaining at stage $t = 5$ is close to zero, a pattern we also observed for the optimal solution in Fig. 2. As such, we expect that the found solutions for these settings with many agents are in fact close to optimal. The runtime results, shown in Fig. 5(b), increase linearly with respect to the number of agents. While the runtime increases with the number of agents, the bottleneck in our experiments preventing even further scalability was insufficient memory, not computation time.

6. RELATED WORK

The work most closely related to ours is that on other approaches for finite-horizon (factored) Dec-POMDPs. However, many these approaches, such as MBDP (see, e.g., [26]) are geared towards scaling with respect to the horizon, not the number of agents. MBDP methods typically use an MDP heuristic for belief sampling, and the proposed TP heuristic could be used instead for problems with many agents. An optimal approach for factored Dec-POMDPs was shown to scale to three agents [21]. This method is similar to FFSPC in that it is also based on CGBGs, but, in order to guarantee optimality, it performs backtracking, only exploits structure in the last stage, and does not apply any of the efficiency-increasing approximations we propose. An approximate approach based on MBDP yielded results for up to 20 agents [34]. For the (substantially different) infinite-horizon case there have been results scaling to 10 agents [23].

Many research efforts have focused on special cases that are more tractable. In particular, assumptions of *transition and observation independence (TOI)* [1] have been investigated to exploit independence between agents, e.g., as in ND-POMDPs [16]. For this latter class, there have also been a number of improvements in scalability with respect to the number of agents (up 15 agents) [31, 14, 12, 13]. Recently, [33] proposed *transition-decoupled* Dec-POMDPs, which loosen the severe restrictions of TOI. Still, many interesting tasks, such as two robots carrying a chair, cannot be modeled. Another framework that allows for certain transition dependencies is the DPCL [30], which depends on the specification of coordination locales to express in which states the agents can influence each other. An approximate solution method was shown to scale to 100 agents [32]. Except for the work on TD-POMDPs [33], these approaches, like our own, do not offer quality guarantees. However, in contrast to these approaches, FFSPC does not place any restrictions on the class of problems that can be represented.

7. CONCLUSIONS

This paper proposed FFSPC, which approximately solves factored Dec-POMDPs by representing them as a series of CGBGs. To estimate the payoff functions of these CGBGs, we computed approximate factored value functions given predetermined scope structures. We do so via *transfer planning*, which uses value functions for smaller source problems as components of the factored Q-value function for the original target problem. An empirical evaluation showed that FFSPC significantly outperforms state-of-the-art methods

for solving Dec-POMDPs with more than two agents and scales well with respect to the number of agents. In particular, FFSPC found optimal or near-optimal solutions on problems for which the optimum can be computed. For larger problems, it found solutions as good as or better than comparison Dec-POMDP methods in almost all cases and outperformed the baselines in all cases. Most strikingly, FFSPC computed solutions for problems that cannot be tackled by any other methods at all, not even the baselines. In particular, it found near-optimal solutions and scaled up to 1000 agents, where previously only problems with small to moderate numbers of agents (up to 20) had been tackled.

While these results demonstrate that FFSPC substantially advances the state of the art in scaling approximate Dec-POMDP methods with respect to the number of agents, its ability to scale with respect to the horizon remains limited, since the number of types in the CGBGs still grows exponentially with the horizon. In future work we hope to address this problem by clustering types [6]. In particular, by clustering the individual types of an agent that induce similar payoff profiles, we hope to scale to much longer horizons. When aggressively clustering to a constant number of types, runtime could even be made linear in the horizon.

Acknowledgements

We would like to thank Nikos Vlassis for extensive discussions on the topic, and David Silver and Leslie Kaelbling for their valuable input. Research supported in part by NWO CATCH project #640.005.003, and by AFOSR MURI project #FA9550-09-1-0538. M.S. is funded by the FP7 Marie Curie Actions Individual Fellowship #275217 (FP7-PEOPLE-2010-IEF).

8. REFERENCES

[1] R. Becker, S. Zilberstein, V. Lesser, and C. V. Goldman. Transition-independent decentralized Markov decision processes. In *AAMAS*, 2003.

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

[3] C. Boutilier, T. Dean, and S. Hanks. Decision-theoretic planning: Structural assumptions and computational leverage. *JAIR*, 11:1–94, 1999.

[4] X. Boyen and D. Koller. Tractable inference for complex stochastic processes. In *UAI*, 1998.

[5] R. Emery-Montemerlo, G. Gordon, J. Schneider, and S. Thrun. Approximate solutions for partially observable stochastic games with common payoffs. In *AAMAS*, 2004.

[6] R. Emery-Montemerlo, G. Gordon, J. Schneider, and S. Thrun. Game theoretic control for robot teams. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2005.

[7] A. Farinelli, A. Rogers, A. Petcu, and N. R. Jennings. Decentralised coordination of low-power embedded devices using the max-sum algorithm. In *AAMAS*, 2008.

[8] C. Guestrin, D. Koller, R. Parr, and S. Venkataraman. Efficient solution algorithms for factored MDPs. *JAIR*, 19:399–468, 2003.

[9] Y. Kim, M. Krainin, and V. Lesser. Application of max-sum algorithm to radar coordination and scheduling. In *Workshop on Distributed Constraint Reasoning*, 2010.

[10] J. R. Kok and N. Vlassis. Collaborative multiagent reinforcement learning by payoff propagation. *JMLR*, 7:1789–1828, 2006.

[11] D. Koller and R. Parr. Computing factored value functions for policies in structured MDPs. In *IJCAI*, 1999.

[12] A. Kumar and S. Zilberstein. Constraint-based dynamic programming for decentralized POMDPs with structured interactions. In *AAMAS*, 2009.

[13] A. Kumar, S. Zilberstein, and M. Toussaint. Scalable multiagent planning using probabilistic inference. In *IJCAI*, 2011.

[14] J. Marecki, T. Gupta, P. Varakantham, M. Tambe, and M. Yokoo. Not all agents are equal: scaling up distributed POMDPs for agent networks. In *AAMAS*, 2008.

[15] K. P. Murphy and Y. Weiss. The factored frontier algorithm for approximate inference in DBNs. In *UAI*, 2001.

[16] R. Nair, P. Varakantham, M. Tambe, and M. Yokoo. Networked distributed POMDPs: A synthesis of distributed constraint optimization and POMDPs. In *AAAI*, 2005.

[17] F. A. Oliehoek. *Value-Based Planning for Teams of Agents in Stochastic Partially Observable Environments*. PhD thesis, Informatics Institute, Univ. of Amsterdam, 2010.

[18] F. A. Oliehoek. Decentralized POMDPs. In M. Wiering and M. van Otterlo, editors, *Reinforcement Learning: State of the Art*. Springer Berlin Heidelberg, 2012.

[19] F. A. Oliehoek, J. F. Kooi, and N. Vlassis. The cross-entropy method for policy search in decentralized POMDPs. *Informatica*, 32:341–357, 2008.

[20] F. A. Oliehoek, M. T. J. Spaan, and N. Vlassis. Optimal and approximate Q-value functions for decentralized POMDPs. *JAIR*, 32:289–353, 2008.

[21] F. A. Oliehoek, M. T. J. Spaan, S. Whiteson, and N. Vlassis. Exploiting locality of interaction in factored Dec-POMDPs. In *AAMAS*, 2008.

[22] F. A. Oliehoek, S. Whiteson, and M. T. J. Spaan. Exploiting structure in cooperative Bayesian games. In *UAI*, 2012.

[23] J. Pajarinen and J. Peltonen. Efficient planning for factored infinite-horizon DEC-POMDPs. In *IJCAI*, 2011.

[24] Z. Rabinovich, C. V. Goldman, and J. S. Rosenschein. The complexity of multiagent systems: the price of silence. In *AAMAS*, 2003.

[25] A. Rogers, A. Farinelli, R. Stranders, and N. Jennings. Bounded approximate decentralised coordination via the max-sum algorithm. *Artif. Intel.*, 175(2):730–759, 2011.

[26] S. Seuken and S. Zilberstein. Formal models and algorithms for decentralized decision making under uncertainty. *Autonomous Agents and Multi-Agent Systems*, 17(2):190–250, 2008.

[27] M. T. J. Spaan, F. A. Oliehoek, and C. Amato. Scaling up optimal heuristic search in Dec-POMDPs via incremental expansion. In *IJCAI*, 2011.

[28] D. Szer, F. Charpillet, and S. Zilberstein. MAA*: A heuristic search algorithm for solving decentralized POMDPs. In *UAI*, 2005.

[29] M. E. Taylor and P. Stone. Transfer learning for reinforcement learning domains: A survey. *JMLR*, 10:1633–1685, 2009.

[30] P. Varakantham, J. Kwak, M. E. Taylor, J. Marecki, P. Scerri, and M. Tambe. Exploiting coordination locales in distributed POMDPs via social model shaping. In *ICAPS*, 2009.

[31] P. Varakantham, J. Marecki, Y. Yabu, M. Tambe, and M. Yokoo. Letting loose a SPIDER on a network of POMDPs: Generating quality guaranteed policies. In *AAMAS*, 2007.

[32] P. Velagapudi, P. Varakantham, P. Scerri, and K. Sycara. Distributed model shaping for scaling to decentralized POMDPs with hundreds of agents. In *AAMAS*, 2011.

[33] S. J. Witwicki and E. H. Durfee. Influence-based policy abstraction for weakly-coupled Dec-POMDPs. In *ICAPS*, 2010.

[34] F. Wu, S. Zilberstein, and X. Chen. Rollout sampling policy iteration for decentralized POMDPs. In *UAI*, 2010.