Exploiting Locality of Interaction in Factored Dec-POMDPs

Frans A. Oliehoek
Informatics Institute,
University of Amsterdam
Kruislaan 403, 1098 SJ
Amsterdam, The Netherlands
faolieho@science.uva.nl

Matthijs T.J. Spaan
Institute for Systems Robotics
Instituto Superior Técnico
Av. Rovisco Pais, 1, 1049-001
Lisbon, Portugal
mtjspaan@isr.ist.utl.pt

Shimon Whiteson
Informatics Institute,
University of Amsterdam
Kruislaan 403, 1098 SJ
Amsterdam, The Netherlands
whiteson@science.uva.nl

Nikos Vlassis
Department of Production
Engineering and Management
Technical University of Crete
Chania, Greece
vlassis@dpem.tuc.gr

ABSTRACT

Decentralized partially observable Markov decision processes (Dec-POMDPs) constitute an expressive framework for multiagent planning under uncertainty, but solving them is provably intractable. We demonstrate how their scalability can be improved by exploiting locality of interaction between agents in a factored representation. Factored Dec-POMDP representations have been proposed before, but only for Dec-POMDPs whose transition and observation models are fully independent. Such strong assumptions simplify the planning problem, but result in models with limited applicability. By contrast, we consider general factored Dec-POMDPs for which we analyze the model dependencies over space (locality of interaction) and time (horizon of the problem). We also present a formulation of decomposable value functions. Together, our results allow us to exploit the problem structure as well as heuristics in a single framework that is based on collaborative graphical Bayesian games (CGBGs). A preliminary experiment shows a speedup of two orders of magnitude.

Categories and Subject Descriptors

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

General Terms

Algorithms, Performance, Theory, Experimentation

Keywords

Planning under uncertainty, cooperative multiagent systems, decentralized POMDP.

1. INTRODUCTION

Planning under uncertainty is a central topic within artificial intelligence. Especially during the last two decades models such as Markov decision processes (MDPs), for single-agent planning in a stochastic environment, and partially observable MDPs (POMDPs), for single-agent planning in an environment that is both stochastic and partially observable, have become popular. The decentralized POMDP (Dec-POMDP) is a natural extension to multiple agents that each have their individual observations, but have to optimize the same reward function.

Unfortunately, optimally solving Dec-POMDPs is NEXP-complete [2], and the same holds for finding an ϵ-approximate solution [14]. As a result, research has focused on special cases to overcome this complexity barrier. For instance models with more assumptions on observability and/or communication have been considered [6, 9]. These approaches try to exploit independence between agents, but they either require the agents to observe the state of the system (as in a multiagent MDP), or to observe a subset of state variables (as in a factored Dec-MDP) and communicate at every stage. A recent approach [15] tries to minimize the amount of communication.

For the particular case of transition and observation independent Dec-POMDPs [1], exploitation of independence between agents has been suggested by Nair et al. [11] and extended by Varakantham et al. [19]. However, the assumption of transition and observation independence (TOI) severely limits the applicability of these models. In particular, TOI implies that the agents have disjoint sets of individual states and observations, and that one agent cannot influence the state transition nor the observation of another agent. In practice this means that many interesting tasks, such as two robots carrying a chair, cannot be modeled.

In this paper, we present a more general analysis of locality of interaction in factored Dec-POMDPs. In particular we show that the last stage in the process contains the highest degree of independence but, when moving back in time (towards the initial stage $t = 0$), the scope of dependence grows. Nevertheless, we show that most of the independence is located—and can be exploited—where it is most needed, since the overwhelming majority of the computational effort of solving a Dec-POMDP is spent in the last stage [18].

We use this result to extend our previous work, in which we solved Dec-POMDPs by representing them as a series of Bayesian games (BGs), one for each stage [13, 12]. In

Cite as: Exploiting Locality of Interaction in Factored Dec-POMDPs, Oliehoek, Spaan, Whiteson and Vlassis, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons (eds.), May, 12-16., 2008, Estoril, Portugal, pp. 517-524.

this setting, the increase in computational cost at the later stages results from the size of the BGs and grows doubly exponentially with t. Using locality of interaction, we introduce collaborative graphical BGs (CGBGs), which are less computationally expensive to solve in later stages where independence is high. We propose an extension of GMAA*, a generalized policy search algorithm, which uses CGBGs to exploit locality of interaction in the last stage of a Dec-POMDP. A preliminary experiment shows a speedup of two orders of magnitude.

2. BACKGROUND

Here we introduce the regular (non-factored) Dec-POMDP, then we describe the extensions for the factored version.

2.1 Dec-POMDPs

Formally, a Dec-POMDP with n agents is defined as a tuple $\langle \mathcal{A}g, \mathcal{S}, \mathcal{A}, T, R, \mathcal{O}, O, b^0, h \rangle$, where

- $\mathcal{A}g = \{1, \ldots, n\}$ is the set of agents.
- $\mathcal{S}$ is a finite set of states.
- $\mathcal{A} = \times_i \mathcal{A}_i$ is the set of *joint actions*, where $\mathcal{A}_i$ is the set of actions available to agent i. Every time step, the agents take one joint action $\mathbf{a} = \langle a_1, ..., a_n \rangle$, but agents do not observe each other's actions. We also write $\mathbf{a}_{\mathcal{I}}$ for the joint action of a subset of agents $\mathcal{I} \subseteq \mathcal{A}g$.
- The transition probabilities $P(s'|s, \mathbf{a})$ are specified by the transition function T.
- $\mathcal{O} = \times_i \mathcal{O}_i$ is the set of joint observations. Every stage one joint observation $\mathbf{o} = \langle o_1, ..., o_n \rangle$ is received. Each agent i only observes its own component o_i.
- O is the observation function, which specifies the probability of joint observations: $P(\mathbf{o}|\mathbf{a}, s')$.
- R is the immediate reward function, which maps states and joint actions to reals: $R(s, \mathbf{a})$.
- $b^0 \in \mathcal{P}(\mathcal{S})$ is the initial state distribution at time $t = 0$, where $\mathcal{P}(S)$ denotes the infinite set of probability distributions over the finite set S.
- h is the horizon of the problem.

At each stage $t = 0 \ldots h-1$ the agents take an action and receive an observation. Their goal is to maximize the expected *cumulative reward* or *return*. The planning task entails finding a *joint* policy $\pi = \langle \pi_1 \ldots \pi_n \rangle$, which specifies an individual policy π_i for each agent i. Such an individual policy in general specifies an individual action for each action-observation history $\vec{\theta}_i^t = (a_i^0, o_i^1, \ldots, a_i^{t-1}, o_i^t)$, e.g., $\pi_i(\vec{\theta}_i^t) = a_i^t$. The set of $\vec{\theta}_i^t$ is denoted $\vec{\Theta}_i^t$. However, when only allowing *deterministic* or *pure* policies, a π_i maps each observation history $(o_i^1 \ldots o_i^t) = \vec{o}_i^t \in \vec{\mathcal{O}}_i^t$ to an action, e.g. $\pi_i(\vec{o}_i^t) = a_i^t$. For a more detailed introduction to Dec-POMDPs see [12].

2.2 Factored Dec-POMDPs

A *factored* Dec-POMDP has a state space $\mathcal{S} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_{|\mathcal{X}|}$ that is spanned by $\mathcal{X} = \{\mathcal{X}_1, \ldots, \mathcal{X}_{|\mathcal{X}|}\}$ a set of state variables, or *factors*. A state corresponds to an assignment of values for all factors $s = \langle x_1, \ldots, x_{|\mathcal{X}|} \rangle$. In a factored Dec-POMDP, the transition and observation model can be compactly represented by exploiting conditional independence between variables. In particular, the transition and observation model can be represented by a dynamic Bayesian network (DBN) [3]. In such a DBN arrows represent causal influence, and each node with incoming edges has a conditional probability table (CPT) associated with it. Although the size of these CPTs is exponential in the number of parents, the parents are typically a subset of all state factors and actions, leading to exponentially smaller tables than a flat model.

Similarly, the reward function often can be compactly represented by exploiting *additive separability,* which means that it is possible to decompose the reward function into the sum of 'smaller' *local* reward functions: $R = R^1 + \cdots + R^\rho$. Here 'smaller' means that the functions are defined over a smaller number of state and action variables, i.e., the *scope* of these functions is smaller. We can represent the smaller functions in the DBN [3]. The reward nodes have conditional reward tables (CRTs) associated with them that represent the smaller reward functions. The local reward functions should not be confused with *individual* reward functions in a system with self-interested agents, such as partially observable stochastic games [7] and (graphical) BGs [16, 17]. In such models agents compete to maximize their individual reward functions, while we consider agents that collaborate to maximize the sum of local payoff functions.

Decision trees [3, 4] or algebraic decision diagrams [8] can be used to further reduce the size of representation (of both CPTs and CRTs) by exploiting *context specific independence.* E.g., the value of factor x_2 is of no influence when factor x_1 has a particular value. We will not consider these further enhancements in this paper.

2.3 Factored Firefighting problem

To illustrate the model we will introduce the factored firefighting (FFF) problem as a running example. It models a team of n firefighters that have to extinguish fires in a row of $n_H = n + 1$ houses. We will assume $n = 3$, $n_H = 4$ as illustrated in Figure 1 (Left). Each house H is characterized by a fire level fl_H, an integer parameter in $[0, n_f)$. A state in FFF is an assignment of fire levels $s = \langle fl_1, fl_2, fl_3, fl_4 \rangle$. At every time step, each agent i can choose to fight fires at house i or $i+1$. Figure 1 (Middle) shows the DBN for the problem. If a house H is burning ($fl_H > 0$) and no firefighting agent is present, its fire level will increase by one point with probability 0.8 if any of its neighboring houses are burning, and with probability 0.4 if none of its neighbors are on fire. A house that is not burning can only catch fire with probability 0.8 if one of its neighbors is on fire. When two agents are in the same house, they will extinguish any present fire completely, setting the house's fire level to 0. A single agent present at a house will lower the fire level by one point with probability 1 if no neighbors are burning, and with probability 0.6 otherwise. Each agent can only observe whether there are flames, $o_i = F$, or not, $o_i = N$, at its location. Flames are observed with probability 0.2 if $fl_H = 0$, with probability 0.5 if $fl_H = 1$, and with probability 0.8 otherwise. Initially, the fire level fl_H of each house is drawn from a uniform distribution. The agents receive a reward of $-fl_H$ for each house H. In particular, the rewards are specified by the fire levels at the next time step

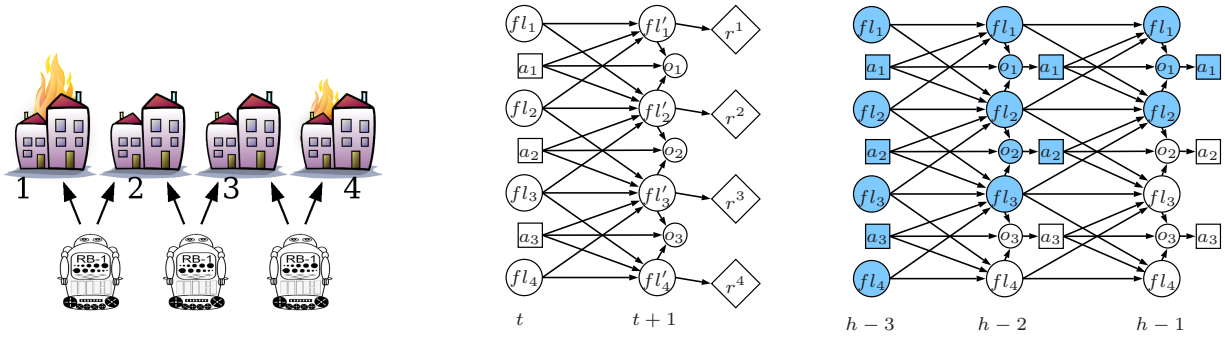

Figure 1: Left: An illustration of the FFF problem. Middle: A DBN modeling the transition, observation and reward function for the FFF problem. Right: The dynamics of the FFF problem over multiple stages. The scope of Q^1, illustrated by shading, increases when going back in time.

$\forall_{1 \le H \le 4}\, r^H(fl'_H) = -fl'_H$.

We can reduce these rewards to ones of the type $R^H(s, \mathbf{a})$ by considering the expectation over fl'_H. For instance, for house 1

$$R^1(fl_{\{1,2\}}, a_1) = \sum_{fl'_1} P(fl'_1 | fl_{\{1,2\}}, a_1) r^1(fl'_1),$$

where $fl_{\{1,2\}}$ denotes $\langle fl_1, fl_2 \rangle$. This formulation is possible because, as Figure 1 shows, fl_1, fl_2 and a_1 are the only variables that influence the probability of fl'_1. Similarly, the other local reward functions are given by $R^2(fl_{\{1,2,3\}}, \mathbf{a}_{\{1,2\}})$, $R^3(fl_{\{2,3,4\}}, \mathbf{a}_{\{2,3\}})$ and $R^4(fl_{\{3,4\}}, a_3)$.

3. FACTORED Q-VALUE FUNCTIONS

We propose to exploit independence between agents. In this section, we first analyze this independence by considering Q-value functions that express the expected return. In particular, rather than using a single Q-value function that is defined over joint actions of all agents, we show that the value can be decomposed into ρ local Q-value functions (corresponding to the ρ local reward functions) involving only a subset of agents (although this subset may grow with each stage further from the last stage $t = h - 1$). Such a decomposed Q-value function is called *factored*. In section 4 we will exploit such independence using a variation of BGs.

3.1 Scope and scope backup

As an example of a factored Q-value function, we consider the last stage of FFF. Clearly, Q^{h-1} is defined to be equal to the immediate reward function, and thus can be decomposed in 4 local Q-value functions $Q = Q^1 + \cdots + Q^4$. Where each Q^k is defined over the same subset of variables as R^k. This subset of variables is called the *scope* of Q^k. In order to exploit independence between agents, we discriminate between the variables that pertain to state factors and those that pertain to agents (i.e., actions and observations). Exploitation of independence between state factors will be mentioned, but is left for future work.

Definition 3.1 (Scope) Let $\mathcal{H} \subseteq \{1, \ldots, |\mathcal{X}|\}$ be a subset of state factor indices, and let $\mathcal{I} \subseteq \{1, \ldots, n\}$ be a subset of agent indices, and let us write $\mathcal{X}_\mathcal{H} = \times_{h \in \mathcal{H}} \mathcal{X}_h$, $\mathcal{A}_\mathcal{I} = \times_{i \in \mathcal{I}} \mathcal{A}_i$, $\vec{\mathbf{\Theta}}^t_\mathcal{I} = \times_{i \in \mathcal{I}} \vec{\Theta}^t_i$. Then a function $q : \mathcal{X}_\mathcal{H} \times \vec{\mathbf{\Theta}}^t_\mathcal{I} \times \mathcal{A}_\mathcal{I} \to \mathbb{R}$ has *state factor scope* $\mathbb{X}(q) = \mathcal{H}$ and *agent scope* $\mathbb{A}(q) = \mathcal{I}$.

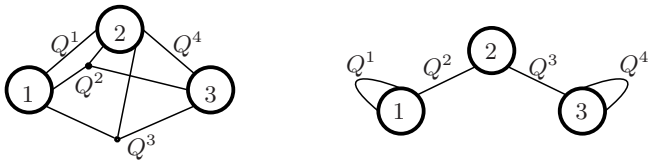

Figure 2: Interaction graphs for $t = h - 2$ (left) and $t = h - 1$ (right) of the FFF problem.

This definition has the most general form of value function we will use, involving state-factors, action-observation histories and actions. It also applies when not all of these elements are used. For example, the function $R^1(fl_{\{1,2\}}, a_1)$ has scope $\mathbb{X}(R^1) = \{1, 2\}$ and $\mathbb{A}(R^1) = \{1\}$.

Now we can formalize the e-th component of the Q-value function for joint policy π of the last stage $h - 1$.

$$Q^e_\pi(\mathbf{x}^{h-1}_{\mathbb{X}^R_e}, \vec{\boldsymbol{\theta}}^{h-1}_{\mathbb{A}^R_e}, \mathbf{a}^{h-1}_{\mathbb{A}^R_e}) \equiv R^e(\mathbf{x}^{h-1}_{\mathbb{X}^R_e}, \mathbf{a}^{h-1}_{\mathbb{A}^R_e}), e = 1 \ldots \rho, \quad (1)$$

where $\mathbb{X}^R_e = \mathbb{X}(R^e)$ and $\mathbb{A}^R_e = \mathbb{A}(R^e)$. E.g., in the FFF problem $Q^1_\pi(fl_{\{1,2\}}, \vec{\theta}^{h-1}_1, a_1) \equiv R^1(fl_{\{1,2\}}, a_1)$. The scope of Q^1_π is shown in the right diagram of Figure 1 at $h - 1$.

This Q-value function can be used to define an *interaction hyper-graph* $G = \langle \mathcal{A}g, \mathcal{E} \rangle$. In this graph, the nodes correspond with agents and the ρ hyper-edges $e \in \mathcal{E}$ (edges that can involve more than two nodes) correspond to the ρ local Q-value functions $Q^{e,h-1}$ for stage $h - 1$.[1] In particular, an edge e connects $\mathbb{A}(Q^{e,h-1})$, the agents involved in $Q^{e,h-1}$. For example, the interaction hyper-graph for the last two stages of the FFF problem is shown in Figure 2. It is the independence represented by sparsity of such interaction graphs that we will exploit in section 4 to reduce computation costs.

Given $Q^{e,h-1}$, we consider the Q-value function at $h - 2$, which will clearly depend on $P(\mathbf{x}^{h-1}_{\mathbb{X}^R_e}, \mathbf{o}^{h-1}_{\mathbb{A}^R_e} | \cdot)$ where $(\cdot)$ represents any factors of influence at stage $h - 2$. These factors of influence are defined by the scope backup.

Definition 3.2 (Scope backup) Given a set $\mathcal{K}$ of particular state factors and observations of particular agents $\mathcal{K} \subseteq \{\mathcal{X}'_1, \ldots, \mathcal{X}'_{|\mathcal{X}|}, \mathcal{O}_1, \ldots, \mathcal{O}_n\}$, the scope backup operator $\Gamma(\mathcal{K})$ returns the subset of variables of the previous stage $\{\mathcal{X}_1, \ldots, \mathcal{X}_{|\mathcal{X}|}, \mathcal{A}_1 \ldots, \mathcal{A}_n\}$ (i.e., from the left side of

[1] We omit the stage index to Q-functions if it is implicit in their arguments, as in (1).

the DBN) that are ancestors of variables in $\mathcal{K}$. We discriminate the state-factor component and the agent component of the scope backup:

$$\Gamma^{\mathbb{X}}(\mathcal{K}) = \{j \mid \exists_{Y \in \mathcal{K}} \text{ s.t. } \mathcal{X}_j = \text{ancestor}(Y)\},$$

$$\Gamma^{\mathbb{A}}(\mathcal{K}) = \{j \mid \exists_{Y \in \mathcal{K}} \text{ s.t. } \mathcal{A}_j = \text{ancestor}(Y)\}.$$

For example, in FFF the probability of $fl^{h-1}_{\{1,2\}}, o^{h-1}_1$ depends only on $fl^{h-2}_{\{1,2,3\}}$ and actions $\mathbf{a}^{h-2}_{\{1,2\}}$ as shown at $h-2$ in Figure 1 (right). When the policy is fixed, the probability of actions $\mathbf{a}^{h-2}_{\{1,2\}}$ is determined by $\mathbf{o}^{h-2}_{\{1,2\}}$. In turn, to determine $P(fl^{h-2}_{\{1,2,3\}}, \mathbf{o}^{h-2}_{\{1,2\}}|\cdot)$ we need to consider all variables of influence at stage $h-3$. As indicated in Figure 1, this set encompasses all state factors and agents.

3.2 Decomposition of Value Functions

In order to more formally define factored value functions, we need to specify the probability of states and joint action-observation histories. In order to do so, we will consider joint policies that are partially specified with respect to time. In particular, the past joint policy specifies a policy for each agent up to stage t, i.e., $\varphi^t = (\delta^0 \dots \delta^{t-1})$, where $\delta^{t'} = \langle \delta^{t'}_1, \dots, \delta^{t'}_n \rangle$ is a joint decision rule for stage t'. An individual decision rule for stage t' maps individual length-t' observation histories to actions $\delta^{t'}_i : \vec{\mathcal{O}}^{t'}_i \rightarrow \mathcal{A}_i$. Now we can recursively define the following joint distribution

$$P(s^t, \vec{\boldsymbol{\theta}}^t | b^0, \varphi^t) = P(\mathbf{o}^t | \mathbf{a}^{t-1}, s^t) \sum_{s^{t-1} \in \mathcal{S}} P(s^t | s^{t-1}, \mathbf{a}^{t-1}) P_{\varphi^t}(\mathbf{a}^{t-1} | \vec{\boldsymbol{\theta}}^{t-1}) P(s^{t-1}, \vec{\boldsymbol{\theta}}^{t-1} | b^0, \varphi^t). \quad (2)$$

When we are interested in a subset of state factors and agents we can marginalize as follows:[2]

$$P(\mathbf{x}^t_f, \vec{\boldsymbol{\theta}}^t_g | b^0, \varphi^t) = \sum_{\mathbf{x}^t_{\bar{f}}} \sum_{\vec{\boldsymbol{\theta}}^t_{\bar{g}}} P(\mathbf{x}^t_f, \mathbf{x}^t_{\bar{f}}, \vec{\boldsymbol{\theta}}^t_g, \vec{\boldsymbol{\theta}}^t_{\bar{g}} | b^0, \varphi^t) \quad (3)$$

We will write $V^t(\pi)$ for the expected return of π over stages $t \dots (h-1)$. The value function $V^{h-1}(\pi)$ for the last stage $t = h-1$ can now be decomposed as

$$V^{h-1}(\pi) = \sum_{e \in \mathcal{E}} V^{e,h-1}(\pi) = \sum_{e \in \mathcal{E}} \sum_{\mathbf{x}^{h-1}_{\mathbb{X}^R_e}} \sum_{\vec{\boldsymbol{\theta}}^{h-1}_{\mathbb{A}^R_e}} P(\mathbf{x}^{h-1}_{\mathbb{X}^R_e}, \vec{\boldsymbol{\theta}}^{h-1}_{\mathbb{A}^R_e} | b^0, \pi) Q^e_\pi(\mathbf{x}^{h-1}_{\mathbb{X}^R_e}, \vec{\boldsymbol{\theta}}^{h-1}_{\mathbb{A}^R_e}, \pi_{\mathbb{A}^R_e}(\vec{\boldsymbol{\theta}}^{h-1}_{\mathbb{A}^R_e})). \quad (4)$$

Such a decomposition is possible for every stage t.

Proposition 3.1 *Given a decomposed immediate reward function, $V^t(\pi)$ of a finite-horizon Dec-POMDP is decomposable for any t.*

Proof The proof is analogous to the one given in [11] and is omitted here. □

[2]When there is independence between the histories in the joint probability distribution this may be exploited. However, such independence quickly vanishes due to the influence of the transition and observation model. The result is a space of joint probability distributions that is exponential in the number of state factors, which typically renders the problem intractable, although approximations with bounded error are possible [5]. We focus on settings where exact evaluations of (2) are feasible.

Let us write $\mathbb{X}_e, \mathbb{A}_e$ for the state factor and agent scope of the Q-value function under concern. That is, $\mathbb{X}_e \equiv \mathbb{X}(Q^{e,t})$ and $\mathbb{A}_e \equiv \mathbb{A}(Q^{e,t})$. We use similar shorthand $\mathbb{X}'_e, \mathbb{A}'_e$ for the next-stage Q-values. Now, the value of a joint policy π for a particular stage is given by

$$V^t(\pi) = \sum_{e \in \mathcal{E}} \sum_{\mathbf{x}^t_{\mathbb{X}_e}} \sum_{\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e}} P(\mathbf{x}^t_{\mathbb{X}_e}, \vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e} | b^0, \pi) Q^e_\pi(\mathbf{x}^t_{\mathbb{X}_e}, \vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e}, \pi_{\mathbb{A}_e}(\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e})) = \sum_{e \in \mathcal{E}} V^{e,t}(\pi). \quad (5)$$

where, using shorthand notation $\Gamma^{\mathbb{X}} = \Gamma^{\mathbb{X}}(\mathbf{x}^{t+1}_{\mathbb{X}'_e} \cup \mathbf{o}^{t+1}_{\mathbb{A}'_e})$ and $\Gamma^{\mathbb{A}} = \Gamma^{\mathbb{A}}(\mathbf{x}^{t+1}_{\mathbb{X}_e} \cup \mathbf{o}^{t+1}_{\mathbb{A}_e})$,

$$Q^e_\pi(\mathbf{x}^t_{\mathbb{X}_e}, \vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e}, \mathbf{a}_{\mathbb{A}_e}) = R^e(\mathbf{x}^t_{\mathbb{X}^R_e}, \mathbf{a}_{\mathbb{A}^R_e}) + \sum_{\mathbf{x}^{t+1}_{\mathbb{X}'_e}} \sum_{\mathbf{o}^{t+1}_{\mathbb{A}'_e}} P(\mathbf{x}^{t+1}_{\mathbb{X}'_e}, \mathbf{o}^{t+1}_{\mathbb{A}'_e} | \mathbf{x}^t_{\Gamma^{\mathbb{X}}}, \mathbf{a}_{\Gamma^{\mathbb{A}}}) Q^e_\pi(\mathbf{x}^{t+1}_{\mathbb{X}'_e}, \vec{\boldsymbol{\theta}}^{t+1}_{\mathbb{A}'_e}, \pi_{\mathbb{A}'_e}(\vec{\boldsymbol{\theta}}^{t+1}_{\mathbb{A}'_e})) \quad (6)$$

Note that the scope of $Q^{e,t}_\pi$ is given by

$$\mathbb{X}_e \equiv \mathbb{X}^R_e \cup \Gamma^{\mathbb{X}}, \quad (7)$$

$$\mathbb{A}_e \equiv \mathbb{A}^R_e \cup \Gamma^{\mathbb{A}} \cup \mathbb{A}'_e. \quad (8)$$

It is necessary that $\mathbb{A}'_e \subseteq \mathbb{A}_e$, otherwise $\vec{\boldsymbol{\theta}}^{t+1}_{\mathbb{A}'_e}$ can be undefined. For instance, for stage $h-2$, the scope of the value function corresponding to reward function 1 is given by

$$\mathbb{X}(Q^{1,h-2}_\pi) = \mathbb{X}(R^1) \cup \Gamma^{\mathbb{X}}(\{fl_1, fl_2, o_1\}) = \{fl_1, fl_2\} \cup \{fl_1, fl_2, fl_3\} = \{fl_1, fl_2, fl_3\}$$

$$\mathbb{A}(Q^{1,h-2}_\pi) = \mathbb{A}(R^1) \cup \Gamma^{\mathbb{A}}(\{fl_1, fl_2, o_1\}) \cup \mathbb{A}(Q^{1,h-1}_\pi) = \{1\} \cup \{1,2\} \cup \{1\} = \{1,2\}$$

We can identify the scopes for the other 3 ($e = 2 \dots 4$) value functions $Q^{e,h-2}_\pi$ in a similar way. The agent scopes define a new interaction graph for stage $h-2$ of the FFF problem, as illustrated in Figure 2. The figure clearly shows that Q^2 and Q^3 both involve all agents, which means that at this stage there is no reduction possible. However, our representation will allow us to exploit independence for the last stage, which is the bottleneck of computation. Also, to avoid duplicate work for stage $h-2$ it is possible to merge the Q-functions by adding them.

3.3 Locality of Interaction

When assuming transition and observation independence, the probability in (6) reduces easily: each variable x^{t+1}_i (representing a local state for agent i) is only dependent on x^t_i (its value at the previous state), and each o_i is only dependent on x^{t+1}_i.[3] This means that the scope does not expand and, as a result, Nair et al. [11] have a stationary interaction graph. Let $\mathcal{N}_i$ denote the neighbors of agent i, including i itself, in this graph. They define the *local neighborhood utility* of an agent i as the expected return for all the edges that contain agent i:

$$V(\pi_{\mathcal{N}_i}) = \sum_s b^0(s) \sum_{e \text{ s.t. } i \in e} Q^{e,0}_\pi(\mathbf{x}^0_{\mathbb{X}_e}, \vec{\boldsymbol{\theta}}_\emptyset, \pi_{\mathbb{A}_e}(\vec{\boldsymbol{\theta}}_\emptyset)). \quad (9)$$

[3]In fact the probabilities can also depend on an 'external' uninfluenceable state, but this is of no consequence.

Consequently they show that when an agent $j \notin \mathcal{N}_i$ changes its policy, $V^t(\pi_{\mathcal{N}_i})$ is not affected, a property they refer to as *locality of interaction*.

In our more general case, such a notion of locality of interaction over full policies is not properly defined, simply because the interaction graph and hence agent i's neighborhood can be different at every stage. However, at a particular stage t, given a past joint policy φ^t that has been followed, a future policy ψ^t that will be followed and $\pi = (\varphi^t, \delta^t, \psi^t)$, we can define

$$V^t(\delta^t_{\mathcal{N}_i}) = \sum_{e \text{ s.t. } i \in \mathbb{A}(Q^{e,t})} \sum_{\mathbf{x}^t_{\mathbb{X}_e}} \sum_{\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e}} P(\mathbf{x}^t_{\mathbb{X}_e}, \vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e} | b^0, \varphi^t) Q^e_{\pi_{\mathbb{A}_e}}(\mathbf{x}^t_{\mathbb{X}_e}, \vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e}, \delta^t_{\mathbb{A}_e}(\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e})). \quad (10)$$

Consequently, locality of interaction holds within each stage. The definition of $\mathbb{A}_e$ (8) implies that the agent scope is non-decreasing when going back in time[4], which means that the scope of dependency is typically larger in earlier stages (as also described in [10]) and the interaction graphs for those earlier stages are denser, as shown in Figure 2.

However, even when there is no independence in transitions and observations (i.e., the scope includes all factors and agents with just one backup), the decomposed value function for the last stage still has limited scope, allowing for a significant speedup of the last stage, and thus for the entire computation.

4. FACTORED DEC-POMDPS VIA GRAPHICAL BAYESIAN GAMES

A non-factored Dec-POMDP can be represented by a series of Bayesian games, one for each time step [13]. Consequently, a policy can be found by solving the BGs for stage $0, 1, \ldots, h-1$ consecutively, a procedure we refer to as *forward-sweep policy computation*. However, the cost of solving these BGs grows doubly-exponentially with the horizon. The number of joint policies for a BG for the last stage, and thus the cost of optimally solving such a BG, is

$$O\Big(|\mathcal{A}_*|^{n(|\mathcal{O}_*|^{h-1})}\Big), \quad (11)$$

where $\mathcal{A}_*$ and $\mathcal{O}_*$ denote the largest individual action and observation sets. In this section we provide tools to exploit the independence between agents as represented by the interaction graphs, reducing the agent component n in (11). First, we show that a factored Dec-POMDP with additively separable rewards can be modeled using a series of collaborative graphical BGs. Then we show how such CGBGs can be solved efficiently.

4.1 Collaborative Graphical Bayesian Games

Here we define the collaborative graphical BG as a graphical BG [16, 17], in which the agents try to optimize the sum of local rewards, rather than an individual payoff function.

Definition 4.1 A *collaborative graphical BG (CGBG),* is a tuple $\langle \mathcal{A}g, \langle \mathcal{A}_1, \ldots, \mathcal{A}_n \rangle, \boldsymbol{\Theta}, P(\boldsymbol{\Theta}), \langle u_1, ..., u_\rho \rangle \rangle$, with:

[4]In contrast, the state scope is not necessarily non-decreasing. It is, however, when the probability of all factors x'_i is dependent on their x_i value in the previous stage, a property that often holds in the real world.

$Q^{e=1,t=1}(\vec{\theta}^1_1, a^1_1)$

$\vec{\theta}^1_1$		
(H_1, F)	H_1	-0.25
	H_2	-1.10
(H_1, N)	H_1	-0.14
	H_2	-0.79

$Q^{e=2,t=1}(\vec{\boldsymbol{\theta}}^1_{\{1,2\}}, \mathbf{a}^1_{\{1,2\}})$

$\vec{\theta}^1_1$		$\vec{\theta}^1_2$ (H_2, F)		(H_2, N)	
		H_2	H_3	H_2	H_3
(H_1, F)	H_1	-0.55	-1.60	-0.50	-1.50
	H_2	0	-0.55	0	-0.50
(H_1, N)	H_1	-0.16	-1.10	-0.14	-1.00
	H_2	0	-0.16	0	-0.14

$Q^{e=4,t=1}(\vec{\theta}^1_3, a^1_3)$

$\vec{\theta}^1_3$		
(H_4, F)	H_3	-1.50
	H_4	-0.51
(H_4, N)	H_3	-1.10
	H_4	-0.15

$Q^{e=3,t=1}(\vec{\boldsymbol{\theta}}^1_{\{2,3\}}, \mathbf{a}^1_{\{2,3\}})$

$\vec{\theta}^1_3$		$\vec{\theta}^1_2$ (H_2, F)		(H_2, N)	
		H_2	H_3	H_2	H_3
(H_4, F)	H_3	-1.10	0	-0.71	0
	H_4	-1.90	-1.10	-1.70	-0.71
(H_4, N)	H_3	-1.00	0	-0.58	0
	H_4	-1.90	-1.00	-1.60	-0.58

Figure 3: The CGBG for stage $t = 1$ of the horizon $h = 2$ FFF problem for past joint policy $\varphi^1 = \langle H_1, H_2, H_4 \rangle$. The shading indicates an arbitrary policy for agent 2.

- $\mathcal{A}g = \{1, \ldots, n\}$ is the set of agents.
- $\mathcal{A}_i$ is the set of actions of agent i. $\mathcal{A} = \times_{i=1}^n \mathcal{A}_i$ is the set of joint actions.
- $\boldsymbol{\Theta} = \times_i \Theta_i$ is the set of joint types over which a probability function $P(\boldsymbol{\Theta})$ is given (and is common knowledge).
- u_k are the local payoff functions.

A CGBG is collaborative, which means that all agents try to maximize the same payoff. It is also graphical, which means that the payoff can be decomposed into a sum of local payoff functions u_k, each of which can depend on only a subset of agents, just like in factored Dec-POMDPs. Formally, a local payoff function is specified as a mapping from the types and actions of a subset of agents to reals: $u_k : \boldsymbol{\theta}_{\mathcal{I}} \times \mathbf{a}_{\mathcal{I}} \to \mathbb{R}$, where $\mathcal{I} = \mathbb{A}(u_k)$ is the subset of agent indices that participate in payoff function k.

As for factored Dec-POMDPs, it is possible to construct a hyper-graph $G = \langle \mathcal{A}g, \mathcal{E} \rangle$ from the set of payoff functions such that the agent scope of each local payoff function corresponds to a hyper-edge $e \in \mathcal{E}$. We will write u_e as the local payoff function corresponding to edge e and $\mathbb{A}^u_e = \mathbb{A}(u_e)$.

Finally, the goal is to maximize the expected sum of rewards:

$$\beta^* = \arg\max_{\beta} \sum_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} P(\boldsymbol{\theta}) \sum_{e \in \mathcal{E}} u_e(\boldsymbol{\theta}_{\mathbb{A}^u_e}, \beta_{\mathbb{A}^u_e}(\boldsymbol{\theta}_{\mathbb{A}^u_e})), \quad (12)$$

where $\beta_{\mathbb{A}^u_e}(\boldsymbol{\theta}_{\mathbb{A}^u_e}) = \langle \beta_i(\theta_i) \rangle_{i \in \mathbb{A}^u_e}$ is the joint action of that subset resulting from application of the individual BG-policies β_i. It can be shown that β^* constitutes a Pareto-optimal Bayes-Nash equilibrium [12, 20].

4.2 A stage as a Collaborative Graphical BG

Here we explain how a CGBG can model a stage of a factored Dec-POMDP. For each stage t, we want to find a joint decision rule which is specified by $\delta^t \equiv \beta^{t,*}$ the solution of the CGBG for that stage. Also, the private information each agent i holds is its action-observation history $\vec{\theta}^t_i$, which therefore naturally corresponds to agent i's type: $\theta_i \equiv \vec{\theta}^t_i$. The past joint policy φ^t is available when planning, therefore

the probability of joint types is specified by $P(\vec{\boldsymbol{\theta}}^t|b^0,\varphi^t) = \sum_{s^t} P(s^t,\vec{\boldsymbol{\theta}}^t|b^0,\varphi^t)$ which is given by (2).

What is left to be determined is the set of payoff functions. For now, we will assume that there is a factored Q-value function $Q^t(\vec{\boldsymbol{\theta}}^t,\mathbf{a})$ for the Dec-POMDP that can be used as the payoff function. We will describe such factored Q-value functions in section 4.4.

In Figure 3, a CGBG for stage $t=1$ of the horizon $h=2$ FFF problem is shown. The past joint policy φ^1 is a joint decision rule for the first stage, which is equivalent to joint action $\langle H_1,H_2,H_4\rangle$.

4.3 Solving the CGBGs

Given a past joint policy φ^t, the solution of a CGBG is

$$\begin{aligned}\beta^{t,*} &= \arg\max_{\beta^t} \sum_{e\in\mathcal{E}} \sum_{\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e}} P(\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e}|\varphi^t,b^0) Q^{e,t}(\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e},\beta^t_{\mathbb{A}_e}(\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e})) \\ &= \arg\max_{\beta^t} \sum_{e\in\mathcal{E}} V^{e,t}_{\varphi^t}(\beta^t_{\mathbb{A}_e}) \qquad (13)\end{aligned}$$

where $\mathbb{A}_e = \mathbb{A}(Q^{e,t})$ and where the probability can be computed as a marginal of (2). In contrast to solving a regular BG, the maximization of (13) does not require enumeration of all joint BG policies. For example, in Figure 3, agent 1 and 3 can separately compute their best response to the shaded policy of agent 2.

More formally, we can optimally solve the CGBG using non-serial dynamic programming (NDP), also known as variable elimination (VE) [6, 20]. We will illustrate this technique for the CGBG of Figure 3. Let us write V^e for $V^{e,t=1}_{\varphi^1}$ with $\varphi^1 = \langle H_1,H_2,H_4\rangle$, the maximization we are then performing is

$$\max_{\beta} \left[V^1(\beta_1) + V^2(\beta_{\{1,2\}}) + V^3(\beta_{\{2,3\}}) + V^4(\beta_3)\right], \qquad (14)$$

where $\beta_{\{1,2\}} = \langle\beta_1,\beta_2\rangle$. Now, by isolating the functions in which agent 1 participates

$$\max_{\beta_{\{2,3\}}} \left[V^3(\beta_{\{2,3\}}) + V^4(\beta_3) + \max_{\beta_1}\Big[V^1(\beta_1) + V^2(\beta_{\{1,2\}})\Big]\right],$$

it is possible to split the problem in two smaller ones. NDP continues by isolating a next agent, etc. The time needed by NDP is exponential in the induced width of the interaction graph [6, 20]. Let us denote this width w, then we have that the complexity of solving the last stage CGBG is

$$O\Big(n\cdot|\mathcal{A}_*|^{w(|\mathcal{O}_*|^{h-1})}\Big), \qquad (15)$$

yielding an exponential speedup over (11) as long as $n \gg w$.

Conceptually, the $V^{e,t}_{\varphi^t}$ functions from (14) define a graphical normal-form game, which is then solved by NDP. This normal-form game does not need to be constructed explicitly, however. Alternatively, it is possible to reduce to a standard GBG by considering the *local neighborhood utility,* defined in a similar fashion as (9), as the individual payoff function for agent i. Therefore all methods that will find a Bayes-Nash equilibrium for GBGs, e.g. [16, 17], can be used to find a local optimum. A different option is to solve the CGBGs using max-plus [9, 20], which only is optimal for tree-shaped graphs, but in practice works well on graphs with cycles. Also, it may be possible to employ upper bounds over BG-policies that are partially specified with respect to individual BG-policies as in SPIDER [19].

4.4 Factored Q-value Functions

Here we will consider the payoff functions for the CGBG for each stage, which constitute factored Q-value function for the factored Dec-POMDP. In particular, we will consider a normative description of the optimal Q-value function, which demonstrates that such a series of CGBGs exactly models a Dec-POMDP.

In Section 3.2, we derived an expression for the expected value of a given policy. This description also holds for an optimal joint policy. Let us assume we are given an optimal joint policy π^*. Then substitution in (5) and (6) yields a normative description of the optimal value function. Here we transform the resulting Q-value function $Q^e_{\pi^*}(\mathbf{x}^t_{\mathbb{X}_e},\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e},\mathbf{a}_{\mathbb{A}_e})$ to the form $Q^e_{\pi^*}(\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e},\mathbf{a}_{\mathbb{A}_e})$.

It is possible to split $P(\mathbf{x}^t_{\mathbb{X}_e},\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e}|b^0,\pi)$ in a marginal and a conditional. Also $\pi^* = \left(\varphi^{t,*},\delta^{t,*},\psi^t\right)$ with $\varphi^{t,*},\psi^t$ the optimal past and future joint policy. This allows us to write:

$$V^t(\pi^*) = \sum_{e\in\mathcal{E}} \sum_{\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e}} P(\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e}|b^0,\varphi^{t,*}) Q^e_{\pi^*}(\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e},\delta^{t,*}_{\mathbb{A}_e}(\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e})) \qquad (16)$$

with

$$\begin{aligned}Q^e_{\pi^*}(\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e},\mathbf{a}_{\mathbb{A}_e}) &= \sum_{\mathbf{x}^t_{\mathbb{X}_e}} P(\mathbf{x}^t_{\mathbb{X}_e}|\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e},b^0,\pi^*) R^e(\mathbf{x}^t_{\mathbb{X}^R_e},\mathbf{a}_{\mathbb{A}^R_e}) + \\ &\sum_{\mathbf{o}^{t+1}_{\mathbb{A}'_e}} P(\mathbf{o}^{t+1}_{\mathbb{A}'_e}|\vec{\boldsymbol{\theta}}^t_{\mathbb{A}_e},b^0,\pi,\mathbf{a}_{\mathbb{A}_e}) Q^e_{\pi^*}(\vec{\boldsymbol{\theta}}^{t+1}_{\mathbb{A}'_e},\pi^{t+1}_{\mathbb{A}'_e}(\vec{\boldsymbol{\theta}}^{t+1}_{\mathbb{A}'_e})). \qquad (17)\end{aligned}$$

the Q-value function of optimal policy π^*.

Theorem 4.1 *When using* (17) *as the payoff functions for the series of CGBGs representing a Dec-POMDP, forward-sweep policy computation yields an optimal policy.*

Proof Following the reasoning from [12], we only have to show that, given that the optimal past policy $\varphi^{t,*}$ is followed, using (17) as the payoff function of a CGBG yields $\delta^{t,*}$. When using this factored Q-value function, the solution of that CGBG—given by (13)—maximizes the quantity from (16). Now, because the optimal policy per definition maximizes the expected value, $\delta^{t,*}$ per definition maximizes (16) and as a result we have that $\beta^{t,*}$ found by (13) is optimal. □

The implication of this theorem is that modeling a factored Dec-POMDP using CGBGs is exact. Unfortunately, as in the non-factored case, computation of the optimal Q-value function is impractical and seems to require the optimal policy.

5. HEURISTIC POLICY SEARCH

Because it is impractical to compute (16), it is not practical to apply forward-sweep policy computation to compute an optimal policy. However, for the last—and most computationally demanding—stage we in fact do have the factored optimal Q-value function available: it is given by the immediate reward function. We propose a policy search method based on Generalized MAA* (GMAA*) [12] that employs a factored Q-value function for, and thus exploits locality of interaction in, the last stage.

GMAA* consists of three procedures. The first procedure iterates over a pool of partial joint policies, pruning this

Algorithm 1 GMAA*

```
1: v* := −∞
2: P := {φ^0 = ()}
3: repeat
4:    φ^t := Select(P)
5:    Φ_Next := Next(φ^t)
6:    if Φ_Next contains full policies Π_Next ⊆ Φ_Next then
7:       π' := arg max_{π∈Π_Next} V(π)
8:       if V(π') > v* then
9:          v* := V(π')
10:         π* := π'
11:         P := {φ ∈ P | V̂(φ) > v*}{prune P}
12:      end if
13:      Φ_Next = Φ_Next \ Π_Next {remove full policies}
14:   end if
15:   P := (P \ φ^t) ∪ {φ ∈ Φ_Next | V̂(φ) > v*}
16: until P is empty
```

Algorithm 2 Next(φ^t)—ELSI

```
1: if t < h − 1 then
2:    {Perform regular MAA* for t = 0, ..., h − 2 }
3:    Φ^{t+1} = construct all policies φ^{t+1} as in [18, 13]
4:    return Φ^{t+1}
5: else
6:    {For t = h − 1 exploit independence }
7:    G = ConstructCGBG(φ^{h−1}, b^0)
8:    δ^{h−1} = NDPSolve(G)
9:    π = (φ^{h−1}, δ^{h−1})
10:   return π
11: end if
```

	V^*	T_{GMAA^*}	$T_{\text{GMAA}^*\text{-ELSI}}$
$h = 2$	−5.213685	≤ 0.01 s	0.03 s
$h = 3$	−6.654551	0.15 s	0.46 s
$h = 4$	−7.462685	4834.07 s	12.67 s

Table 1: Results for the FFF for horizon 2, 3, and 4. Column V^* indicates the value of an optimal policy, T_{GMAA^*} the computation time for the plain GMAA* algorithm, and $T_{\text{GMAA}^*\text{-ELSI}}$ the computation time for our proposed method.

pool whenever possible. This is the core of GMAA* and is fixed, while the other two procedures can be performed in many ways. The second procedure, `Select`, selects which policy to process next and thus determines the type of search (e.g., depth-first, breadth-first, A*-like). The third procedure, `Next`, constructs a new set of (partial) joint policies based on the selected policy.

The core procedure of GMAA* (see Algorithm 1) maintains a set of partial joint policies φ together with their heuristic values $\widehat{V}(\varphi)$. This 'policy pool' P is initialized with a completely unspecified joint policy $\varphi^0 = ()$ and the maximum lower bound (found so far) $\underline{v}^\star$ is set to $-\infty$. $\pi^\star$ denotes the best joint policy found so far. First, `Select` selects a partial joint policy φ from P. We assume that, as in MAA* [18], the partial policy with the highest heuristic value is selected. Next, the selected policy is processed by the policy search operator `Next`, which returns a set of (partial) joint policies $\Phi_{\texttt{Next}}$ and their heuristic values. When `Next` returns one or more full policies $\pi \in \Phi_{\texttt{Next}}$, the provided values $\widehat{V}(\pi) = V(\pi)$ are a lower bound for an optimal joint policy, which can be used to prune the search space. Any found partial joint policies $\varphi \in \Phi_{\texttt{Next}}$ with a heuristic value $\widehat{V}(\varphi) > \underline{v}^\star$ are added to P. Then process is repeated until the policy pool is empty.

The original MAA* can be seen as an instance of the generalized case with a `Next` operator that, given a partial joint policy $\varphi^t = (\delta^0, \ldots, \delta^{t-1})$, constructs and returns all possible $\varphi^{t+1} = (\delta^0, \ldots, \delta^{t-1}, \delta^t)$ that are consistent with φ^t along with their heuristic values. (This corresponds with constructing a regular BG for stage t and evaluating all BG-policies δ^t.) Because the MAA* `Next` operator returns all 'children' (φ^{t+1} consistent with φ^t), it is guaranteed to find an optimal joint policy, as long as the used heuristic is an upper bound to the optimal value function [18].

We propose GMAA*-ELSI, a method for exploiting the last-stage independence (ELSI), which is described by Algorithms 1 and 2 jointly. For stages $0, \ldots, h-2$ regular MAA* is used, while for the last stage $t = h-1$, a CGBG is constructed rather than a regular BG. This CGBG is then solved using NDP and the solution of this CGBG used to construct a full-length joint policy π.

6. EXPERIMENTS

In Table 1 we compare results for GMAA*-ELSI with regular GMAA* (that solves a regular BG even for the last stage) for FFF, introduced in Section 2.3. We used Q_{BG} [13] as the heuristic Q-value function for all time steps of GMAA*, and for time steps 0–$h-2$ of GMAA*-ELSI. As both methods are optimal, they compute identical optimal policies, whose values are shown in the first column. The remaining two columns show computation time of both methods. For the low horizons, we can see that GMAA* is faster, as the Bayesian games to be solved are small, and not worth the overhead caused by the construction and solution of CGBGs in GMAA*-ELSI's last time step. However, for horizon 4 we can see the dramatic speedup provided by exploiting the local interactions in this factored Dec-POMDP.

7. DISCUSSION AND FUTURE WORK

In this paper we presented a framework for exploiting locality of interaction in factored Dec-POMDPs. By formalizing the dependence propagation through time, and analyzing the dependencies per stage, it is possible to construct an interaction graph for each stage. The independence represented by this interaction graph can then be exploited by efficiently solving the corresponding collaborative graphical Bayesian game (CGBG). Also, we presented a formulation of factored Q-value functions that can be used as the payoff function for these CGBGs, and have shown how GMAA*, an existing heuristic search method, can be extended to exploit the independence in the last stage by using CGBGs.

The analysis of this paper strengthens our belief that general Dec-POMDPs should be tackled per stage, rather than trying to find individual full-length policies in an iterative procedure. It also shows that solution of CGBGs scale exponentially in the induced width of the interaction graph (w), but only polynomially in the number of agents (n). In accordance, a preliminary experimental evaluation shows that the resulting algorithm GMAA*-ELSI obtains a speedup of two orders of magnitude. To our knowledge, these are the first experimental results for general (factored) Dec-POMDPs with more than 2 agents.

There are two main directions for future research. The first is the identification and evaluation of approximate fac-

tored Q-value functions. Analogous to our previous work [13, 12], one idea is to first use dynamic programming to compute a Q-function for the underlying (factored) MDP or POMDP and then 'project' this (typically non-decomposable) function onto an interaction graph. Such projection operators can possibly be integrated in the dynamic programming backups [9]. Using such an approach, it will be possible to exploit independence at all stages, by computing CGBGs for stages $0 \dots h-1$.

The second main research direction is the unification of two previous heuristic search methods for Dec-POMDPs. On the one hand MAA* [18] searches over joint policies which are partially specified with respect to time (e.g., a policy only specified for the first stage). On the other hand SPIDER [19] searches over joint policies that are partially specified w.r.t. the individual agents (e.g., a joint policy only specifying an individual policy for agent 1). The algorithm proposed here takes a step towards this unification: intuitively GMAA* corresponds to the former, while solution of the CGBGs with NDP corresponds to the latter. Future algorithms may truly unify the two approaches.

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. This work was partially supported by Fundação para a Ciência e a Tecnologia (ISR/IST pluriannual funding) through the POS_Conhecimento Program that includes FEDER funds and through grant PTDC/EEA-ACR/73266/2006.

8. REFERENCES

[1] R. Becker, S. Zilberstein, V. Lesser, and C. V. Goldman. Solving transition independent decentralized Markov decision processes. *Journal of Artificial Intelligence Research (JAIR)*, 22:423–455, December 2004.

[2] 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.

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

[4] C. Boutilier, R. Dearden, and M. Goldszmidt. Stochastic dynamic programming with factored representations. *Artificial Intelligence*, 121(1-2):49–107, 2000.

[5] X. Boyen and D. Koller. Tractable inference for complex stochastic processes. In *Proc. of Uncertainty in Artificial Intelligence*, pages 33–42. San Francisco: Morgan Kaufmann, 1998.

[6] C. Guestrin, D. Koller, and R. Parr. Multiagent planning with factored MDPs. In *Advances in Neural Information Processing Systems 14*, pages 1523–1530, 2002.

[7] E. A. Hansen, D. S. Bernstein, and S. Zilberstein. Dynamic programming for partially observable stochastic games. In *AAAI '04: Proceedings of the Nineteenth National Conference on Artificial Intelligence*, pages 709–715, 2004.

[8] E. A. Hansen and Z. Feng. Dynamic programming for POMDPs using a factored state representation. In *Proceedings of the Fifth International Conference on Artificial Intelligence Planning Systems*, pages 130–139, Apr. 2000.

[9] J. R. Kok and N. Vlassis. Collaborative multiagent reinforcement learning by payoff propagation. *Journal of Machine Learning Research*, 7:1789–1828, 2006.

[10] D. Koller and R. Parr. Computing factored value functions for policies in structured MDPs. In *Proc. Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1332–1339, 1999.

[11] R. Nair, P. Varakantham, M. Tambe, and M. Yokoo. Networked distributed POMDPs: A synthesis of distributed constraint optimization and POMDPs. In *Proc. of the National Conference on Artificial Intelligence*, pages 133–139, 2005.

[12] F. A. Oliehoek, M. T. J. Spaan, and N. Vlassis. Optimal and approximate Q-value functions for decentralized POMDPs. *Journal of Artificial Intelligence Research*, 2008. (to appear).

[13] F. A. Oliehoek and N. Vlassis. Q-value functions for decentralized POMDPs. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, pages 833–840, May 2007.

[14] Z. Rabinovich, C. V. Goldman, and J. S. Rosenschein. The complexity of multiagent systems: the price of silence. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, pages 1102–1103, 2003.

[15] M. Roth, R. Simmons, and M. Veloso. Exploiting factored representations for decentralized execution in multi-agent teams. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, pages 467–463, May 2007.

[16] S. Singh, V. Soni, and M. Wellman. Computing approximate Bayes-Nash equilibria in tree-games of incomplete information. In *EC '04: Proceedings of the 5th ACM conference on Electronic commerce*, pages 81–90, 2004.

[17] V. Soni, S. Singh, and M. Wellman. Constraint satisfaction algorithms for graphical games. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, pages 423–430, 2007.

[18] D. Szer, F. Charpillet, and S. Zilberstein. MAA*: A heuristic search algorithm for solving decentralized POMDPs. In *Proc. of Uncertainty in Artificial Intelligence*, 2005.

[19] 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 *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, 2007.

[20] N. Vlassis. *A Concise Introduction to Multiagent Systems and Distributed Artificial Intelligence*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2007.