Exploiting Separability in Multiagent Planning with Continuous-State MDPs (Extended Abstract) *

Jilles S. Dibangoye
Inria — CITI
Université de Lyon, France
jilles.dibangoye@inria.fr

Christopher Amato
University of New Hampshire
Durham, NH, USA
camato@cs.unh.edu

Olivier Buffet and **François Charpillet**
Inria — LORIA
Vandœuvre-lès-Nancy, France
{firstname.lastname}@inria.fr

Abstract

Decentralized partially observable Markov decision processes (Dec-POMDPs) provide a general model for decision-making under uncertainty in cooperative decentralized settings, but are difficult to solve optimally (NEXP-Complete). As a new way of solving these problems, we recently introduced a method for transforming a Dec-POMDP into a continuous-state deterministic MDP with a piecewise-linear and convex value function. This new Dec-POMDP formulation, which we call an *occupancy MDP*, allows powerful POMDP and continuous-state MDP methods to be used for the first time. However, scalability remains limited when the number of agents or problem variables becomes large. In this paper, we show that, under certain separability conditions of the optimal value function, the scalability of this approach can increase considerably. This separability is present when there is locality of interaction between agents, which can be exploited to improve performance. Unlike most previous methods, the novel continuous-state MDP algorithm retains optimality and convergence guarantees. Results show that the extension using separability can scale to a large number of agents and domain variables while maintaining optimality.

1 Introduction

There is a growing interest in research for solving multiagent problems represented as decentralized partially observable Markov decision processes (Dec-POMDPs) [Bernstein *et al.*, 2002; Amato *et al.*, 2013]. Dec-POMDPs can represent multiple agents, each of which acting based only upon its noisy information about the world state. This model encompasses a large range of practical problems, in which a group of decision-makers collaborates to optimize a common objective. Notable examples include: controlling a team of autonomous robots [Amato *et al.*, 2015], network congestion control, optimizing the production and distribution of energy resources [Jain *et al.*, 2009], or performing monitoring and assisting tasks by a set of sensors [Kumar and Zilberstein, 2009]. All these applications involve a group of decision-makers that operate under uncertainty and rely on noisy sensors.

The difficulty in solving these problems arises because decision-makers can neither see the global state of the process nor (explicitly) communicate their partial views with one another. Exact and approximate algorithms that can solve Dec-POMDPs exist [Bernstein *et al.*, 2009; Dibangoye *et al.*, 2009; Amato *et al.*, 2009; Aras and Dutech, 2010; Boularias and Chaib-draa, 2008; Oliehoek *et al.*, 2013], unfortunately the NEXP-hardness of the Dec-POMDP formalism has restricted their scalability [Bernstein *et al.*, 2002].

Recent advances in optimally solving Dec-POMDPs have recast them as continuous-state (deterministic) MDPs with a piecewise linear and convex optimal value function [Dibangoye *et al.*, 2013b], making it possible to apply MDP and POMDP methods. This reformulation is possible using the common assumption in Dec-POMDP methods that planning can be centralized while preserving decentralized execution. In this new formulation, called the *Occupancy-State MDP* (OMDP), states (called *occupancy states*) are represented by distributions over the underlying states and agent histories, and actions (called *decentralized decision rules*) are mappings from agent histories to agent actions. Because histories are used in the states (and actions), this transformation is lossless, but results in an exponentially larger problem. In this form, it is possible to apply a large variety of efficient continuous-state MDP and POMDP methods.

Unfortunately, in OMDPs, because histories are incorporated, the dimensionality of value functions and occupancy states grows exponentially with time. In fact, it is usually not possible to even maintain a complete and accurate representation of value functions or occupancy states. This is particularly true when the number of agents or other problem variables becomes very large. Moreover, methods that attempt to represent OMDPs more compactly using features rather than states and histories can only scale to medium-sized problems [Dibangoye *et al.*, 2013b; 2014b].

Yet, many practical applications have a structure that should allow greater scalability while preserving optimality.

[0]This paper was invited for submission to the Best Papers From Sister Conferences Track, based on a paper that appeared in the International Conference on Autonomous Agents and Multi-Agent Systems in 2014 (AAMAS-14).

In particular, the idea of exploiting the separability conditions, which occur when optimal value functions are the sum of linear functions over factors associated with a small subset of problem variables, may scale well for domains with large numbers of agents. These value functions are known as Additively Weakly-Separable and Linear (AWSL) functions, a property that is present in the optimal value functions of many practical applications [Kumar and Zilberstein, 2009; Nair *et al.*, 2005; Varakantham *et al.*, 2007]. In fact, the idea of exploiting separability conditions can be traced back to [Koller and Parr, 1999], who explored the use of this property as an approximation for accelerating dynamic programming in MDPs. Since then, several authors have used the approach to exploit locality of interaction between agents [Nair *et al.*, 2005; Kumar *et al.*, 2011].

This paper combines the benefits of transforming Dec-POMDPs into continuous-state MDPs and the separability conditions that are present due to locality of interaction in multiagent systems. Specifically, we target domains represented as ND-POMDPs [Nair *et al.*, 2005], which are a subclass of Dec-POMDPs that exhibit locality of interaction. The primary contribution is a demonstration that, with the locality of interaction, optimal value functions are AWSL functions of occupancy states. Even more importantly, we prove that AWSL functions depend on occupancy states only through marginal probability distributions over factors. The AWSL property permits us to introduce new value function representations that can accelerate both action selection and information tracking steps, thus enhancing performance by several orders of magnitude while still retaining accuracy and convergence guarantees. We demonstrate the scalability of the proposed approach on many ND-POMDP benchmark domains, showing the ability to optimally solve problems that include up to fifteen agents.

2 Background

In this section, we briefly discuss Dec-POMDPs, ND-POMDPs and the conversion of Dec-POMDPs into continuous-state MDPs.

2.1 Dec-POMDPs

A Dec-POMDP $M \equiv (S, A, Z, p, r, \eta^0, T)$ with N agents is:

- A finite set of states S
- A finite set of joint actions $A = A_1 \times A_2 \times \ldots \times A_N$ with A_i representing an action set for agent i
- A finite set $Z = Z_1 \times Z_2 \times \ldots \times Z_N$ of joint observations with Z_i representing an observation set for agent i
- A system dynamics model $p = \{p^{a,z} \colon a \in A, z \in Z\}$, where $p^{a,z}$ is a state transition matrix, and $p^{a,z}(s, s')$ is the probability of transitioning to state s' and receiving joint observation z after taking joint action a in state s
- A reward model $r = \{r^a \colon a \in A\}$, where r^a is a reward vector and $r^a(s)$ is the immediate reward given after executing joint action a in state s
- An initial probability distribution over states, η^0
- A planning horizon T

Dec-POMDPs execute over a number of time steps. At each step, each agent chooses an action and receives an observation while a joint reward is generated for the team. Each agent receives its own observations, but it does not typically receive the observations or actions of the other agents. As a result, each agent has to reason about what the other agents observed and plan to do in order to optimize a joint stream of rewards. This property is at the core of the high complexity of Dec-POMDPs. Therefore, choices for each agent — local policies — depend only upon local information of that agent. Hence, solving Dec-POMDPs requires determining N local policies, which jointly maximize the total expected stream of rewards starting in η^0.

2.2 ND-POMDPs

We now discuss a subclass of Dec-POMDPs, called a networked distributed POMDP (ND-POMDP), that displays locality of interaction (and thus separability) between the agents. That is, unlike general Dec-POMDPs, agents in ND-POMDPs interact only with a small subset of their neighbors. In the following, we write $[1\colon N] = \{1, \ldots, N\}$, and for a given subset $u \subseteq [1\colon N]$ referred to as a factor (or neighborhood), we denote $\bar{u}$ the complement of u. That is $\bar{u} = [1\colon N] \backslash u$. We also define $|u|$, the cardinality of u.

Definition 1. *An ND-POMDP is a Dec-POMDP M with the following properties:*

1. *A **factored state space** $S = S_0 \times S_1 \times \ldots \times S_N$, where S_0 denotes local states that agents cannot affect, and S_i represents a local-state set of agent i; We denote $s_u = (s_0, s_i)_{i \in u}$, $a_u = (a_i)_{i \in u}$, and $z_u = (z_i)_{i \in u}$, the state, action and observation relative to factor $u \subseteq [1\colon N]$.*
2. *A **multiplicative weakly-separable dynamics model** p, where there exist dynamics models $p_0, p_1, \ldots, p_N$ with:*

$$p_u^{a_u, z_u}(s_u, s'_u) \quad = \quad p_0(s_0, s'_0) \prod_{i \in u} p_i^{a_i, z_i}(s_i, s_0, s'_i)$$

for any factor $u \subseteq [1\colon N]$ and $s_u = (s_0, s_i)_{i \in u}$.

3. *An **additive weakly-separable reward model** r, where there exists reward models $r_{u_1}, r_{u_2}, \ldots, r_{u_M}$ such that:*

$$r(s, a) \quad = \quad \sum_{k=1}^{M} r_{u_k}(s_{u_k}, a_{u_k}),$$

where $u_k \subseteq [1\colon N]$, $s_{u_k} = (s_0, s_i)_{i \in u_k}$.

4. *A **multiplicative fully-separable distribution** η^0, where there exists independent distributions $\eta_0^0, \eta_1^0, \cdots, \eta_N^0$ such that:*

$$\eta^0(s) \quad = \quad \eta_0^0(s_0) \prod_{i=1}^{N} \eta_i^0(s_i).$$

Example Problem

To make the representation more concrete, we discuss a simple multi-sensor target tracking problem [Nair *et al.*, 2005] which was motivated by a real-world challenge [Lesser *et al.*, 2003].

In this problem (see Figure 1), five sensors collaborate to track a moving target on a network. We assume the target motion is stochastic and unaffected by the actions that the sensors can perform. Each sensor's possible actions include scanning in one of the four directions — *i.e.,* north, south, east and west, or turning off. Each time a sensor scans an area, it receives a noisy observation (*i.e.,* the observations can have

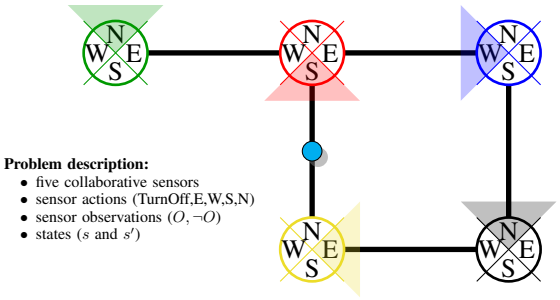

Figure 1: An ND-POMDP target tracking problem.

false positives and false negatives). The action of a given sensor does not affect the transitions or observations of the other sensors. In fact, sensors with no overlapping scanning areas have no direct influence with one another, while the reward function only depends on the actions of neighboring sensors. For example, scanning areas of the red and black sensors do not overlap, hence they cannot influence each other except indirectly through the yellow sensor. However, to track the target and receive a reward, two sensors with overlapping scanning areas (*e.g.,* the red and yellow sensors) must coordinate by scanning the same area simultaneously. Each sensor incurs a cost for scanning whether the target is present or not, but no cost for turning off. In the scenario depicted in Figure 1, there are six factors, factor u_0 captures the location of the moving target, which is not affected by the sensor decisions; and each of the remaining factors $u_1, \ldots, u_5$ involves two sensors connected by an arc, (*e.g.,* u_1 involves red and yellow sensors). In this problem, since only neighboring sensors influence one another directly, each sensor's decisions depend solely on its neighbors. These characteristics are often referred to as *locality of interaction* [Nair *et al.*, 2005], and appear in many real-world applications [Lesser *et al.*, 2003; Kumar and Zilberstein, 2009].

2.3 Policies and Value Functions

In ND-POMDPs and Dec-POMDPs, agents choose actions based on the past observations that have been seen. We formally define this concept (in the form of a policy) below.

A local decision rule at step t, denoted d_i^t, is a mapping from t-step action and observation histories of agent i, denoted $\theta_i^t = (a_i^0, z_i^1, \ldots, a_i^{t-1}, z_i^t)$, to local actions of agent i. A T-step local policy of agent i, denoted π_i, is a length-T sequence of local decision rules $\pi_i = (d_i^0, \ldots, d_i^{T-1})$.

A T-step joint policy, denoted π, is an N-tuple of T-step local policies $(\pi_1, \ldots, \pi_N)$, one for each agent. It is also a length-T sequence of joint decision rules $(d^0, \ldots, d^{T-1})$. A joint decision rule at step t, denoted d^t, is an N-tuple of local decision rules $(d_1^t, \ldots, d_N^t)$, one for each agent. A t-step joint action and observation history, denoted θ^t, is an N-tuple of local action and observation histories $(\theta_1^t, \ldots, \theta_N^t)$, one for each agent.

We consider finite-horizon Dec-POMDPs, where the optimality criterion is to maximize the expected sum of rewards over finite steps T. The value function at step t, for a joint policy π, denoted υ_π^t, maps state and joint history pairs to reals for any step t state s^t and joint history θ^t:

$$\upsilon_\pi^t(s^t, \theta^t) = \mathbb{E}[\sum_{\tau=t}^{T-1} r^{a^\tau}(s^\tau) \mid a^\tau = d^\tau(\theta^\tau), \pi],$$

An optimal joint policy π^*, starting at η^0, satisfies equation: $\pi^* \in \arg\max_\pi \upsilon_\pi^0(\eta^0)$. Value functions $\upsilon_{\pi^*}^0, \ldots, \upsilon_{\pi^*}^{T-1}$ are optimal value functions with respect to η^0. At first glance, these value functions exhibit no structural restrictions, but a recent analysis reveals that they are linear over some high-dimensional space.

2.4 Dec-POMDPs as Continuous-State MDPs

A common assumption in many Dec-POMDPs is that planning takes place in a centralized (offline) manner even though agents execute actions in a decentralized fashion (online). In such a planning paradigm, a centralized algorithm maintains, at each time step $t \in [1\colon T]$, the total available information — initial distribution η^0 and partial joint policy $(d^0, \ldots, d^{t-1})$ — it has about the process to be controlled. To summarize this information, Dibangoye et al. [2012; 2013a; 2013b] and Oliehoek [2013] introduced *sufficient statistics*[1] of the total available information for optimal decentralized decision-making.

Such a statistic can retain problem features that are important for calculating rewards. Informally, a sufficient statistic with respect to information state ι and $\check{M}$ is a statistic that summarizes ι and preserves the ability to find an optimal solution of $\check{M}$. Given a sufficient statistic with respect to the current information state and the problem at hand, no additional data about the current information state would provide any further information about the problem.

Theorem 1 ([Dibangoye *et al.*, 2013b]). *A t-step sufficient statistic with respect to information state ι^t, which we call an **occupancy state** and denote η^t, is a probability distribution over all states and joint histories, $\eta^t(s, \theta) = P(s, \theta | \iota^t)$, for any state s and joint history θ.*

The next-step occupancy state $F(\eta^t, d^t) = \eta^{t+1}$ depends on the current occupancy state η^t and joint decision rule d^t:

$$\eta^{t+1}(s', (\theta, a, z)) = \mathbf{1}_{\{a\}}(d^t(\theta)) \sum_{s \in S} \eta^t(s, \theta) \cdot p^{a,z}(s, s'),$$

where $\mathbf{1}_F$ is the indicator function, and for all states $s' \in S$, joint actions $a \in A$, joint observations $z \in Z$, and joint histories θ.

Definition 2. *Let $\check{M} \equiv (\triangle, D, F, R, \eta^0)$ be the MDP with respect to M, which we call the **occupancy Markov decision process**: where $\triangle = \{\triangle^t \colon t \in [0\colon T-1]\}$ is the set of occupancy states, $\triangle^t$ is the step t occupancy state set; and D, F, R, η^0 are identical to $\check{M}$ or eventually M.*

Relative to M, the occupancy-state MDP $\check{M}$ is a deterministic and continuous-state MDP. Note that the occupancy-state MDP is deterministic even though the original Dec-POMDP is stochastic because the occupancy-state is a distribution over states and (action-observation) histories. In the

[1] A statistic $T(X)$ is *sufficient* for the parameter Y precisely if the conditional probability of Y, given the statistic $T(X)$, does not depend on data X — *i.e.,* $P(Y|T(X), X) = P(Y|T(X))$.

occupancy-state MDP, the actions cannot be conditioned on specific observations that are seen because agents do not have access to other agents' observations during execution. Individual agents can still condition their actions on their own observations (but not those of the other agents) because decision rules are used which map these individual histories to actions. An optimal joint policy for $\check{M}$, together with the correct estimation of the occupancy states, will give rise to an optimal behavior for M and [Dibangoye *et al.*, 2013b]. One can solve $\check{M}$ and nevertheless provide an optimal solution for the original problem M [Oliehoek *et al.*, 2013; Dibangoye *et al.*, 2013b].

2.5 Solving Occupancy MDPs

POMDPs can be cast into continuous-state MDPs with piecewise-linear and convex optimal value functions [Smallwood and Sondik, 1973]. As we discuss next, because the occupancy MDP represents a deterministic and continuous-state MDP with a piecewise-linear convex value function, POMDP theory and algorithms can be used.

Lemma 1. *The **optimality equation** for any occupancy state η^t is written as follows: for all $t \in [0\colon T-1]$,*

$$v_*^t(\eta^t) = \max_{d^t} \ \left(R(\eta^t, d^t) + v_*^{t+1}(F(\eta^t, d^t))\right).$$

For $t = T$, we add a boundary condition $v_*^T(\cdot) = 0$.

Dibangoye et al. [2013b] proved that value functions $v_*^0, \ldots, v_*^{T-1}$, which are solutions of the optimality equations (Lemma 1), are piecewise-linear and convex functions of the occupancy states. That is, there exist finite sets of linear functions $\Lambda^0, \ldots, \Lambda^{T-1}$ such that: $v_*^t(\eta^t) = \max_{\alpha^t \in \Lambda^t} \langle \alpha^t, \eta^t \rangle$ (where notation $\langle \cdot, \cdot \rangle$ is the inner-product), for any arbitrary t-step occupancy state η^t.

The FB-HSVI Algorithm

Heuristic search value iteration (HSVI) is a leading POMDP algorithm which can converge to an optimal solution [Smith and Simmons, 2004]. By recasting Dec-POMDPs as occupancy MDPs, HSVI (as well as other POMDP algorithms) can be extended to solve Dec-POMDPs.

Dibangoye et al. [2013b] introduced feature-based HSVI (FB-HSVI), which is shown in Algorithm 1, to improve the efficiency of HSVI in occupancy MDPs. It uses a trial-based best-first search and finds an optimal path from a given initial occupancy state to one T-step occupancy state. It traverses the search space by creating trajectories of occupancy states, each of which starts with the initial occupancy state. For each visited occupancy state, such trajectories always follow the best joint decision rule (ties are broken arbitrarily) specified by the upper bounds $(\bar{v}_t)_{t \in \{0,\ldots,T\}}$. As the algorithm traverses the search space, it updates the upper bounds of the occupancy states along the way. Once the trajectories are finished, it maintains lower bounds $(\underline{v}_t)_{t \in \{0,\ldots,T\}}$ of visited occupancy states in reverse order.

FB-HSVI provably converges to optimal value functions for the initial occupancy state. As it seeks the occupancy states where the upper bound is the largest, and maintains both upper and lower bounds, it reduces the gap between bounds over the initial occupancy state at each iteration. Once the gap is zero, the algorithm has converged. Moreover, FB-HSVI guarantees termination after a finite number of iterations, although this number is (in the worst case) doubly exponential in the maximal length of a trajectory.

Algorithm 1: The FB-HSVI Algorithm.

```
function FB-HSVI()
    initialize v̲^t and v̄_t for all t ∈ {0, ··· , T − 1}.
    while ¬Stop(η_0, 0) do Explore(η^0, 0) ;
    return v̲^t and v̄_t
function Explore(η^t, g^t)
    η̃^t ← Compact(η^t).
    if ¬Stop(η̃^t, g^t) then
        d*^t ∈ arg max_{d^t} R(η̃^t, d^t) + v̄^{t+1}(F(η̃^t, d^t)).
        Update v̄^t.
        Explore(F(η̃^t, d*^t), R(η̃^t, d*^t) + g^t).
        Update v̲^t.
    return g^t
function Stop(η^t, g^t)
    return (v̄^t(η^t) > v̲^t(η^t)) ∨ (g^t + v̄^t(η^t) > v̲^t(η^0))
```

Key Limitations of FB-HSVI

While FB-HSVI performs well in many domains and has theoretical guarantees, its scalability is limited when the number of agents or problem variables is large. To better understand this, notice that the complexity of FB-HSVI depends essentially on two operations: the *decision rule selection*; and the *information tracking*. In either case, FB-HSVI is not geared to exploit the locality of interaction, and thus, it will typically have to consider decision rules and occupancy states over exponentially many variables, though multiple variables have little influence on one another.

3 Leveraging Separability

In this section, we discuss how locality of interaction through separability assumptions (Definitions 1) influences the structure of value functions and occupancy states.

3.1 Separable Value Functions

The primary contribution is a proof that the optimal value function is the sum of linear functions over factors, a property referred to as the additive weak separability and linearity. A formal definition of this property follows.

Definition 3. *Value function g is **additively weakly separable and linear**, if there exist linear functions $g_{u_1}, g_{u_2}, \ldots, g_{u_M}$ such that: $g(s, \theta) = \sum_{k=1}^{M} g_{u_k}(s_{u_k}, \theta_{u_k}), \quad u_1, \ldots, u_M \subseteq [1\colon N]$. Value function g is said to be **additively fully separable and linear**, if $u_k \cap u_{k'} = \emptyset$ for all $k, k' \in [1\colon M]$.*

An optimization problem with an additively fully separable and linear objective function g can be reduced to M independent optimization problems with lower dimensionalities. If g is not fully separable, we often search the whole N-dimensional space all at once. However, algorithms that exploit the weak separability when it is present have been particularly successful, notable examples include weighted

constraint satisfaction algorithms [de Givry *et al.*, 2005; Dechter, 1997; 1999]. In the following, we present the proof that optimal value functions are AWSL functions of the occupancy states. Before proceeding any further, we introduce short-hand notation $g_{u_k|\theta_{u_k}}$ to represent a function over states s_{u_k} s.t.: $g_{u_k|\theta_{u_k}}(s_{u_k}) = g_{u_k}(s_{u_k}, \theta_{u_k})$.

Theorem 2. *Value functions $(\upsilon^t_\pi)_{t\in[1:\,T-1]}$, for any joint policy π, are additively weakly separable and linear functions of occupancy states. That is, there exist vectors $(\alpha^t_{u_k|\theta_{u_k}})_{\theta,k\in[1:\,M]}$ s.t.*

$$\upsilon^t_\pi(\eta^t) = \sum_u \sum_{s_u} \sum_{\theta_u} \eta^t_{u|\theta_u}(s_u) \cdot \alpha^t_{u_k|\theta_{u_k}}(s_u),$$

where $\eta^t_{u|\theta_u}(s_u) = \sum_{s_{\bar{u}},\theta_{\bar{u}}} \eta^t(s,\theta)$ for any η^t and $u \subseteq [1:\,N]$.

The proof of this theorem can be seen in full version of the paper [Dibangoye *et al.*, 2014a].

This theorem demonstrates that value functions can be represented using a finite set of low-dimensional vectors, one $|S_u|$-length vector $\alpha_{u|\theta_u}$ for each joint history θ_u. This result extends a previous separability property of the value function for ND-POMDPs [Nair *et al.*, 2005], which stated that value functions of a specified joint policy can be decomposed into the sum of value functions over factors. Relative to the PWLC property of value function solutions of the optimality equations, the AWSL property provides a significant restrictive structure in the shape of value functions. It is nevertheless unclear how this property can improve efficiency of the FB-HSVI algorithm. In addition, this theorem yields interesting insights. It is worth noticing that this result holds even when there exists a unique factor $u = [1:\,N]$, that is, in general DecPOMDPs.

Corollary 1. *Value functions $(\upsilon^t_\pi)_{t\in[1:\,T-1]}$, for any joint policy π, are additively weakly separable and linear functions of occupancy states. That is, there exist vectors $(\alpha^t_{|\theta})_{\theta,t\in[0:\,T-1]}$ such that $\upsilon^t_\pi(\eta^t) = \sum_s \sum_\theta \eta^t_{|\theta}(s) \cdot \alpha^t_{|\theta}(s)$, where $\eta^t_{|\theta}(s) = \eta^t(s,\theta)$ for any arbitrary occupancy state η^t.*

Proof. The proof holds directly from Theorem 2 with a single factor $u = [1:\,N]$. □

While under the AWSL property, the value function representation is factored, *updating* bounds on the optimal value function requires reasoning about all factors at once. More precisely, no algorithm can update lower or upper bounds separately for each factor and still preserves optimality. However, *evaluating* bounds at a given separable occupancy state can be done separately for each factor while still preserving optimality — which saves significant time. This insight allowed us to extend standard representation of lower and upper bounds so we can evaluate separately for each factor — which enhances the generalization [Dibangoye *et al.*, 2014a].

3.2 Separable Sufficient Statistics

Another important result from Theorem 2 is a proof that value functions depend on occupancy states only through marginal probability distributions over factors. This is a significant result as it allows us to maintain marginal probability distributions independently from one another, which saves non-negligible time and memory, while preserving optimality.

Theorem 3. *For any ND-POMDP with factors $u_1, \ldots, u_M$, marginal occupancy states $(\eta_{u_k|\theta_{u_k}})_{u_k,\theta_{u_k}}$ collectively constitute a sufficient statistic of occupancy state η. Marginal occupancy state $\eta_{u|\theta_u}$ can be updated at each step to incorporate the latest action a_u and observation z_u, where:*

$$\eta_{u|\theta_u,a_u,z_u}(s'_u) = \sum_{s_u} \eta_{u|\theta_u}(s_u) \cdot p^{a_u,z_u}_u(s_u, s'_u).$$

The proof of this theorem can be seen in full version of the paper [Dibangoye *et al.*, 2014a].

This theorem permits us to circumvent unnecessary or redundant operations when maintaining the occupancy states. In particular, we can maintain marginal occupancy states independently from one another, and reuse pre-computed ones when it is possible. In the full version, we described a novel representation of bounds in the FB-HSVI algorithm based on the AWSL property. To this end, the marginal occupancy states $(\eta_{u_k|\theta_{u_k}})_{u_k,\theta_{u_k}}$ are collectively referred to as a **separable occupancy state**.

Similarly to the evaluation of bounds, one can independently *maintain* separable occupancy states, one factor at a time — which saves significant time and memory. Since the update of bounds depend on both bound evaluation and separable occupancy state update, we introduce a weighted constrained optimization program, that allows us to fully exploit the additive weak separability for the joint decision rule selection — which speed up this selection step. Finally, we extended the state-of-the-art FB-HSVI algorithm to incorporate these insights all together into a new algorithm called separable feature-based heuristic search value iteration (SFB-HSVI) [Dibangoye *et al.*, 2014a].

4 Experiments

We compare our extension of FB-HSVI for ND-POMDPs with the standard FB-HSVI algorithm [Dibangoye *et al.*, 2013b], a state-of-the-art exact algorithm for solving general Dec-POMDPs. We call our extension, the separable feature-based heuristic search value iteration (SFB-HSVI) algorithm. We could not compare to the global optimal algorithm (GOA), as it quickly runs out of memory even for the smallest benchmarks. We nonetheless compare with the state-of-the-art approximate algorithms for solving ND-POMDPs, including constraint based dynamic programming (CBDP) [Kumar and Zilberstein, 2009], and FANS [Marecki *et al.*, 2008]. CBDP constructs joint policies based on a small selection of distributions over states. We set the number of distributions to 5 as advised in Kumar and Zilberstein [2009]. FANS relies on various heuristics to build approximate joint policies. For each benchmark, we consider only the heuristic with the best performance.

The experiments using FB-HSVI and SFB-HSVI were run on a Mac with a 2.2GHz Intel Core i7 CPU, 1GB of RAM available, and a time limit of one thousand seconds. The other

T	Algorithms					
	CBDP		FANS		FB-HSVI	
	EV	CPU	EV	CPU	EV	CPU (ext.)	CPU (std.)
5-P domain — $\|S\| = 12$; $N = 5$, $\|Z_i\| = 2$, and $2 \leq \|A_i\| \leq 3$						
3	198.1	2	198.1	20	332.0	2.03	3.77
4	253.7	3	253.9	70	471.2	3.65	10.4
5	302.0	4	355.1	80	605.0	9.36	32.3
6	339.5	5	376.3	90	735.8	35.4	125
7	410.5	6	410.5	100	869.2	231.4	
10	558.6	9	569.4	400			
7-H domain — $\|S\| = 12$; $N = 7$, $\|Z_i\| = 2$, and $2 \leq \|A_i\| \leq 3$						
3	255.5	2	175.8	0.5	418.0	1.5	1.7
4	331.0	4	184.8	1.0	581.8	2.3	5.7
5	404.6	6	274.7	700	765.8	4.7	18.3
6	462.7	7	327.8	800	940.4	12.0	50.4
7	507.5	8	376.8	900	1082.8	40.4	162.6
8	561.4	9			1206.6	261	
10	658.1	10					
11-helix domain — $\|S\| = 49$; $N = 11$, $\|Z_i\| = 2$, and $2 \leq \|A_i\| \leq 4$						
3	328.8	20	255.0	135	554.4	3.1	
4	-	-			777.2	6.4	
5	-	-			1057.6	21.7	
6	-	-			1347.7	140.7	
7	-	-					
10	-	-					
15-3D domain — $\|S\| = 60$; $N = 15$, $\|Z_i\| = 2$, and $2 \leq \|A_i\| \leq 4$						
3	529.0	50	514.2	3000	814.0	4.6	
4	616.9	60			1167.0	7.9	
5	831.5	70			1587.1	22.4	
6	996.2	80			2008.0	78.3	
7	1124.7	90			2353.9	272.7	
10	1493.6	110					
15-Mod domain — $\|S\| = 16$; $N = 15$, $\|Z_i\| = 2$, and $2 \leq \|A_i\| \leq 4$						
3	515.9	60	367.6	200	814.0	2.0	
4	-	-			1142.5	3.5	
5	-	-			1553.2	8.6	
6	-	-			1971.2	26.6	
7	-	-			2336.5	103.8	

Table 1: Performance of FB-HSVI (extended and standard versions), CBDP, and FANS. Notations: EV = $v_\pi^0(\eta^0)$, CPU (sec.), ' ' = time (1000s) expired and '-' = no results available

experiments were conducted on a machine with a 2.4GHz Intel dual core CPU and 1GB of RAM available. To show scalability of SFB-HSVI with respect to the number of agents, we conducted the experiments on the largest ND-POMDP benchmarks based on the sensor network domain [Nair *et al.*, 2005; Marecki *et al.*, 2008; Kumar and Zilberstein, 2009], which range from five to fifteen agents. To highlight the necessity of exact solvers in contrast to approximate methods, we report value $v_\pi^0(\eta^0)$ relative to the best joint policy π each algorithm found. We also report running time in seconds for different planning horizons.

Results can be seen in Table 1. In all tested benchmarks, as depicted in column CPU (ext.), the SFB-HSVI algorithm can find an optimal joint policy for short planning horizons. In particular, it can optimally solve the largest benchmark (15-Mod) at planning horizon $T = 7$ in about one hundred seconds. The results show that the standard FB-HSVI algorithm can also find an optimal joint policy but only for medium-sized benchmarks. For instance, in 5-P and 7-H, both standard and extended FB-HSVI algorithms can find an optimal joint policy for $T \leq 6$. But SFB-HSVI is about three times faster than the standard FB-HSVI algorithm. Since the time required to compute an optimal joint policy increases with increasing planning horizons, the standard FB-HSVI algorithm always runs out of time before our extension, as illustrated in benchmark 5-P at $T = 6$, and benchmark 7-H at $T = 7$. In larger benchmarks 11-helix, 15-3D, and 15-Mod, which involve a dozen of agents, FB-HSVI quickly runs out of memory, as it cannot exploit the locality of interaction.

We further compare SFB-HSVI with approximate ND-POMDP solvers CBDP and FANS. Experiments demonstrate that, although approximate methods can scale up with respect to planning horizon, they often produce poor quality solutions. To illustrate this, consider benchmark 7-H at $T = 7$: CBDP takes 8 seconds and returns a joint policy with a value of 507.5; and FANS takes about 900 seconds and returns a joint policy with a value of 376.8; but, SFB-HSVI takes about 40 seconds to find an optimal joint policy with value 1082.6. Our extension provides solution quality three times higher than that of FANS, and two times higher than that of CBDP. It is worth noting that CBDP can improve solution quality by increasing the number of state distributions considered, but it cannot provide any guarantees since these distributions are not sufficient for optimal planning in ND-POMDPs.

To summarize, our experiments illustrate the scalability of SFB-HSVI with respect to the number of agents. Our algorithm optimally solves all ND-POMDP benchmarks with up to fifteen agents. These results also highlight the necessity of the exact algorithms, especially in critical domains where theoretical guarantees are required.

5 Conclusion

This paper has demonstrated that under a locality of interaction assumption, a property that is exploited in models such as ND-POMDPs, the optimal value functions are additively weakly separable and linear functions. This special structure can be utilized in the context of a recent method for transforming Dec-POMDPs into continuous-state MDPs, which has shown significant scalability gains over previous Dec-POMDP methods. This problem structure allows us to introduce a novel representation of lower and upper bounds of the optimal value functions. This representation has two properties: first, it preserves convergence to an optimal solution; but even more importantly, it significantly reduces the memory requirement of standard representations, thereby increasing scalability. Using this representation, we extended the state-of-the-art algorithm for solving Dec-POMDPs as continuous-state MDPs to optimally solve ND-POMDPs. The resulting algorithm is the first exact algorithm for ND-POMDPs that can solve problems with up to fifteen agents. In the future, we plan to explore applying the additive weak separability and linearity property to general factored Dec-POMDPs. Furthermore, the scalability with respect to the number of agents of our algorithm is encouraging, and we will pursue additional improvements to also scale up with respect to the planning horizon.

6 Acknowledgements

We thank A. Kumar for providing his software. Research supported in part by AFOSR MURI project #FA9550-09-1-0538.

References

[Amato *et al.*, 2009] C. Amato, J. S. Dibangoye, and S. Zilberstein. Incremental policy generation for finite-horizon DEC-POMDPs. In *ICAPS*, 2009.

[Amato *et al.*, 2013] C. Amato, G. Chowdhary, A. Geramifard, N. K. Ure, and M. J. Kochenderfer. Decentralized control of partially observable Markov decision processes. In *CDC*, 2013.

[Amato *et al.*, 2015] C. Amato, G. D. Konidaris, G. Cruz, C. A. Maynor, J. P. How, and L. P. Kaelbling. Planning for decentralized control of multiple robots under uncertainty. In *Proceedings of the International Conference on Robotics and Automation*, 2015.

[Aras and Dutech, 2010] R. Aras and A. Dutech. An investigation into mathematical programming for finite horizon decentralized POMDPs. *JAIR*, 37:329–396, 2010.

[Bernstein *et al.*, 2002] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein. The complexity of decentralized control of Markov decision processes. *Math. Oper. Res.*, 27(4), 2002.

[Bernstein *et al.*, 2009] D. S. Bernstein, C. Amato, E. A. Hansen, and S. Zilberstein. Policy iteration for decentralized control of Markov decision processes. *JAIR*, 34:89–132, 2009.

[Boularias and Chaib-draa, 2008] A. Boularias and B. Chaib-draa. Exact dynamic programming for decentralized POMDPs with lossless policy compression. In *ICAPS*, 2008.

[de Givry *et al.*, 2005] S. de Givry, F. Heras, M. Zytnicki, and J. Larrosa. Existential arc consistency: Getting closer to full arc consistency in weighted CSPs. In *IJCAI*, 2005.

[Dechter, 1997] R. Dechter. Bucket elimination: a unifying framework for processing hard and soft constraints. *Constraints*, 2(1):51–55, 1997.

[Dechter, 1999] R. Dechter. Bucket elimination: A unifying framework for reasoning. *Artif. Intell.*, 113(1-2):41–85, 1999.

[Dibangoye *et al.*, 2009] J. S. Dibangoye, A.-I. Mouaddib, and B. Chaib-draa. Point-based incremental pruning heuristic for solving finite-horizon DEC-POMDPs. In *AAMAS (1)*, pages 569–576, 2009.

[Dibangoye *et al.*, 2012] J. S. Dibangoye, C. Amato, and A. Doniec. Scaling up decentralized MDPs through heuristic search. In *UAI*, 2012.

[Dibangoye *et al.*, 2013a] J. S. Dibangoye, C. Amato, A. Doniec, and F. Charpillet. Producing efficient error-bounded solutions for transition independent decentralized MDPs. In *AAMAS*, 2013.

[Dibangoye *et al.*, 2013b] J. S. Dibangoye, C. Amato, O. Buffet, and F. Charpillet. Optimally solving Dec-POMDPs as continuous-state MDPs. In *IJCAI*, 2013.

[Dibangoye *et al.*, 2014a] J. S. Dibangoye, C. Amato, O. Buffet, and F. Charpillet. Exploiting separability in multiagent planning with continuous-state mdps. In *AAMAS*, 2014.

[Dibangoye *et al.*, 2014b] J. S. Dibangoye, O. Buffet, and F. Charpillet. Error-bounded approximations for infinite-horizon discounted decentralized POMDPs. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2014*, pages 338–353, 2014.

[Jain *et al.*, 2009] M. Jain, M. E. Taylor, M. Tambe, and M. Yokoo. Dcops meet the real world: Exploring unknown reward matrices with applications to mobile sensor networks. In *IJCAI*, pages 181–186, 2009.

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

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

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

[Lesser *et al.*, 2003] V. Lesser, C. Ortiz, and M. Tambe, editors. *Distributed Sensor Networks: A Multiagent Perspective (Edited book)*, volume 9. Kluwer Academic Publishers, May 2003.

[Marecki *et al.*, 2008] 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.

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

[Oliehoek *et al.*, 2013] F. A. Oliehoek, M. T. J. Spaan, C. Amato, and S. Whiteson. Incremental clustering and expansion for faster optimal planning in Dec-POMDPs. *JAIR*, 46:449–509, 2013.

[Oliehoek, 2013] F. A. Oliehoek. Sufficient plan-time statistics for decentralized POMDPs. In *IJCAI*, 2013.

[Smallwood and Sondik, 1973] R. D. Smallwood and E. J. Sondik. The optimal control of partially observable Markov decision processes over a finite horizon. *Operations Research*, 21(5):1071–1088, 1973.

[Smith and Simmons, 2004] T. Smith and R. Simmons. Heuristic search value iteration for POMDPs. In *UAI*, 2004.

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