Optimally Solving Dec-POMDPs as Continuous-State MDPs

Jilles Steeve Dibangoye
Inria / Université de Lorraine
Nancy, France
jilles.dibangoye@inria.fr

Christopher Amato
CSAIL / MIT
Cambridge, MA, USA
camato@csail.mit.edu

Olivier Buffet and **François Charpillet**
Inria / Université de Lorraine
Nancy, France
{firstname.lastname}@inria.fr

Abstract

Optimally solving decentralized partially observable Markov decision processes (Dec-POMDPs) is a hard combinatorial problem. Current algorithms search through the space of full histories for each agent. Because of the doubly exponential growth in the number of policies in this space as the planning horizon increases, these methods quickly become intractable. However, in real world problems, computing policies over the full history space is often unnecessary. True histories experienced by the agents often lie near a structured, low-dimensional manifold embedded into the history space. We show that by transforming a Dec-POMDP into a continuous-state MDP, we are able to find and exploit these low-dimensional representations. Using this novel transformation, we can then apply powerful techniques for solving POMDPs and continuous-state MDPs. By combining a general search algorithm and dimension reduction based on feature selection, we introduce a novel approach to optimally solve problems with significantly longer planning horizons than previous methods.

1 Introduction

Decentralized partially observable Markov decision processes (Dec-POMDPs) have been studied as a general model for decision making under uncertainty in cooperative multiagent systems [Bernstein *et al.*, 2002; Szer *et al.*, 2005; Boularias and Chaib-draa, 2008; Amato *et al.*, 2009; Bernstein *et al.*, 2009; Aras and Dutech, 2010; Dibangoye *et al.*, 2011; Spaan *et al.*, 2011; Oliehoek *et al.*, 2013]. While a number of algorithms have been developed, current optimal algorithms cannot scale beyond small benchmark problems. This is not unexpected given the worst-case NEXP complexity [Bernstein *et al.*, 2002], but many real-world problems have structure that should allow greater scalability.

Current optimal algorithms for general Dec-POMDPs search through the space of solutions (or policies) which map histories of actions that have been taken and observations that have been seen to actions [Hansen *et al.*, 2004; Szer *et al.*, 2005; Boularias and Chaib-draa, 2008; Amato *et al.*, 2009; Spaan *et al.*, 2011; Oliehoek *et al.*, 2013]. These approaches typically proceed by iteratively growing these policies from either the first step (using heuristic search) [Szer *et al.*, 2005; Spaan *et al.*, 2011; Oliehoek *et al.*, 2013] or the last step (using dynamic programming) [Hansen *et al.*, 2004; Boularias and Chaib-draa, 2008; Amato *et al.*, 2009] until policies for the full problem horizon are constructed. As the horizon increases, the doubly exponential growth in the number of possible policies causes solution methods to quickly become intractable. Methods for increasing scalability by compressing policies [Boularias and Chaib-draa, 2008] and histories [Oliehoek *et al.*, 2009] have begun to be explored, but scalability remains limited for many problems.

In contrast, significant progress has been made in the size of problems solved as fully and partially observable Markov decision processes (MDPs and POMDPs). One reason for progress in MDPs has been the use of approximate dynamic programming and function approximation [Powell, 2007; De Farias and Van Roy, 2003] to represent the state of the system (and value function) more concisely. For POMDPs, efficient algorithms have been developed by recasting problems as belief MDPs that utilize probability distributions over states of the system, namely *belief states* [Smallwood and Sondik, 1973]. This belief MDP is a continuous-state MDP with a piecewise linear convex value function, allowing algorithms for POMDPs to scale to large problems while sometimes retaining performance bounds [Shani *et al.*, 2012].

To take advantage of the advances in solvers for POMDPs and MDPs, we solve a Dec-POMDP by recasting it as a continuous-state MDP. The state space consists of all reachable probability distributions over states of the system and histories of the agents (which we term *occupancy states*) and the action space consists of all decision rules mapping histories to actions. A primary result of this paper is a demonstration that the occupancy state is sufficient for optimal planning in Dec-POMDPs. Then, we show that the value function is a piecewise linear and convex function of occupancy states. As a result, POMDP algorithms can for the first time be applied directly to Dec-POMDPs. This is a significant theoretical advance for planning in Dec-POMDPs. However, given that both occupancy states and decision rules are defined over the full history space, scalability remains limited. To increase scalability, we replace the full history space by a low-dimensional feature set using a lossless dimension reduction technique. This reduction allows scalability in the hori-

zon that is significantly greater than previous state-of-the-art optimal algorithms.

The remainder of this paper is organized as follows. We first present the necessary background in Section 2. We then discuss, in Section 3, the transformation of a Dec-POMDP into a continuous-state MDP together with theoretical results. Next, we introduce, in Section 4, our algorithm, *feature-based heuristic search value iteration* (FB-HSVI), which encodes the occupancy state using a low-dimensional feature space and can generate an optimal Dec-POMDP policy. We conclude with experimental results on several benchmarks.

2 Background and related work

We first review relevant models and provide a short overview of optimal Dec-POMDP algorithms.

2.1 MDPs, POMDPs and Dec-POMDPs

An MDP is a tuple $(S, A, \boldsymbol{P}, \boldsymbol{R})$ where S is a set of states, A is a set of actions, $\boldsymbol{P}^a$ is a $|S| \times |S|$ stochastic matrix denoting the probability of transiting from state s to state s' using action a, $\boldsymbol{R}^a$ is a $|S| \times 1$ reward vector defining the reward of executing action a in state s. In solving an MDP, the goal is to produce a *policy* π (a mapping from states to actions) to be executed at each time step t that maximizes some measure of accumulated reward — here we focus on the *expected total reward* over planning horizon T.

A POMDP is a tuple $(S, A, Z, \boldsymbol{P}, \boldsymbol{O}, \boldsymbol{R}, b^0)$ where $S, A, \boldsymbol{P}, \boldsymbol{R}$ are the same as in an MDP, Z is a set of observations and set $\boldsymbol{O}$ consists of $|S| \times |S|$ observation matrices $\boldsymbol{O}^{az}$, one for each observation z and action a. While the agent is unable to observe the true state of the world, it can nonetheless maintain a belief state $b^t[s] = P(s^t = s \mid b^0, a^0, z^1, \cdots, a^{t-1}, z^t)$. Belief b^0 defines the initial belief state. We shall refer to the past sequence of actions and observations the agent experienced up to time step t as $\theta^t = (a^0, z^1, \cdots, a^{t-1}, z^t)$, a t-length *history*. POMDPs can be solved as MDPs with continuous states that are the belief states in the POMDP. In this formulation a policy is a mapping from belief states to actions.

A Dec-POMDP is a tuple $(S, \otimes_i A_i, \otimes_i Z_i, \boldsymbol{P}, \boldsymbol{O}, \boldsymbol{R}, b^0)$. These quantities are the same as a POMDP, but now each agent has its own action and observation sets. The transitions, observations and rewards depend on the actions chosen by all agents. Because each agent only has access to its own local observations, the goal is to maximize a common value function while executing policies that depend on solely each agent's own histories. Due to this decentralized nature of information, Dec-POMDPs cannot naïvely be transformed into POMDPs (or MDPs). In fact, by casting the Dec-POMDP model into a continuous MDP with a piecewise linear convex value function, as we demonstrate in Section 3.2, we provide the first evidence that POMDP methods also directly apply in Dec-POMDPs. Because a central belief state cannot be calculated by the agents, previous work has typically represented a T-step policy for each agent as a *policy tree* which maps each history to an action. Details are discussed in Section 3.1.

2.2 Optimal solutions for Dec-POMDPs

One class of Dec-POMDP solution methods is based on dynamic programming [Howard, 1960]. Here, a set of T-step policy trees, one for each agent, is generated from the bottom up [Hansen *et al.*, 2004]. On each step, all t-step policies are generated that build off policies from step $t+1$. Any policy that has lower value than some other policy for all states and possible policies of the other agents is then pruned (with linear programming). This generation and pruning continues until the desired horizon is reached and trees with the highest value at the initial state are chosen. More efficient dynamic programming methods have been developed, reducing the number trees generated [Amato *et al.*, 2009] or compressing policy representations [Boularias and Chaib-draa, 2008].

Trees can also be built from the top down using heuristic search [Szer *et al.*, 2005]. In this case, a search node is a set of partial policies for the agents up to a given horizon, t. These partial policies can be evaluated up to that horizon and then a heuristic (such as the MDP or POMDP value) can be added. The resulting heuristic values are over-estimates of the true value, allowing an A*-style search through the space of possible policies for the agents, expanding promising search nodes to horizon $t+1$ from horizon t. A more general search framework was also developed [Oliehoek *et al.*, 2008]. Recent work has included clustering probabilistically equivalent histories [Oliehoek *et al.*, 2009] and incrementally expanding nodes in the search tree [Spaan *et al.*, 2011], greatly improving scalability of the original algorithm.

While current methods attempt to limit the number of policies considered, they rely on explicit policy representations that consider the full histories of each agent. Furthermore, even though these algorithms use an offline centralized planning phase, they have not been able to identify a concise sufficient statistic that allows for greater scalability.

3 Dec-POMDPs as continuous-state MDPs

We now discuss the transformation of a Dec-POMDP into a continuous-state MDP, starting with the required definitions. This approach utilizes the commonly used offline planning phase to centralize the available information as a distribution over the state and agent histories from the perspective of a central planner. Note that the central planner does not observe the agent's actions or observations during execution but knows what policies they are executing. The actions can then be selected in the form of decision rules which condition actions on specific histories seen by each agent.

3.1 Preliminary definitions

Let Θ_i^t and Θ^t be sets of agent i's and joint histories at time step $t = 0, 1, \cdots, T-1$, respectively. A *local policy* for agent i is an ordered sequence of *local decision rules*, $\pi_i \equiv \pi_i^0 \pi_i^1 \cdots \pi_i^{T-1}$. A local decision rule $\pi_i^t \colon \Theta_i^t \mapsto A_i$ is a mapping from local histories $\theta_i^t \in \Theta_i^t$ to local actions $\pi_i^t(\theta_i^t) \in A_i$.

A *separable policy* π is a tuple of n local policies, $\pi \equiv (\pi_1, \pi_2, \cdots, \pi_n)$, one for each agent $i = 1, 2, \cdots, n$. We denote π_i the local policy of agent i, and $\pi_{-i} \equiv$

$(\pi_1, \cdots, \pi_{i-1}, \pi_{i+1}, \cdots, \pi_n)$ that of its teammates. A *separable decision rule* at time step t is a tuple of n local decision rules at time step t, $\pi^t \equiv (\pi_1^t, \pi_2^t, \cdots, \pi_n^t)$, one local decision rule for each agent. One can consider a separable policy as a sequence of separable decision rules, $\pi \equiv (\pi^0, \pi^1, \cdots, \pi^{T-1})$, one for each time step. We use notation $\pi^{t:l}$ (where $l \geq t$) to represent an ordered sequence of separable decision rules $(\pi^t, \cdots, \pi^l)$, namely a (partial) policy.

The *occupancy state* of the process under the control of separable policy $\pi^{0:t-1}$ starting at initial probability distribution b^0 is the probability distribution over states of the system and joint histories, $\boldsymbol{\eta}^t = b^0 \boldsymbol{P}^{\pi^{0:t-1}}$. Like $\pi^{t:l}$, we shall use notation $\boldsymbol{\eta}^{t:l}$ instead of $(\boldsymbol{\eta}^t, \cdots, \boldsymbol{\eta}^l)$. Here, occupancy state $\boldsymbol{\eta}^t$ is a $(|S| \times |\Theta^t|) \times 1$ stochastic vector, and $\boldsymbol{P}^{\pi^{t:l}}$ is a $(|S| \times |\Theta^t|) \times (|S| \times |\Theta^{l+1}|)$ stochastic matrix, where each entry $\boldsymbol{P}^{\pi^{t:l}}[s^t, \theta^t; s^{l+1}, \theta^{l+1}]$ is the conditional probability of being in (s^{l+1}, θ^{l+1}) if the initial state and history is (s^t, θ^t) and agents follow separable policy $\pi^{t:l}$. For the sake of simplicity, we use notation $\boldsymbol{P}^{\pi^t}$ instead of $\boldsymbol{P}^{\pi^{t:t}}$. Note that given a policy and initial state distribution, the occupancy state can be calculated from $\boldsymbol{P}$ and $\boldsymbol{O}$ in the Dec-POMDP.

3.2 Formulation and properties

The first property of the occupancy state is that it summarizes all past information of the agents. The proof can be derived by expanding out the occupancy state representation $\boldsymbol{\eta}^t = b^0 \boldsymbol{P}^{\pi^{0:t-1}}$. While we utilize action-observation histories, it can also be shown that occupancy states depending on just observation histories and states are also sufficient when using deterministic policies (which contain an optimal policy) [Oliehoek, 2013].

Theorem 1. *The future occupancy state depends only upon the present occupancy state and separable decision rule:*

$$\boldsymbol{\eta}^t \quad = \quad b^0 \prod_{l=0}^{t-1} \boldsymbol{P}^{\pi^l} \quad = \quad \boldsymbol{\eta}^{t-1} \boldsymbol{P}^{\pi^{t-1}} \tag{1}$$

This theorem shows that occupancy states describe a Markov decision process: the *occupancy MDP model*. Equation (1) defines the transition function of this model. Not surprisingly, the occupancy MDP model is deterministic. A complete representation of the occupancy MDP model is obtained by also specifying the reward function over occupancy states. We denote $\boldsymbol{R}^{\pi^t}$ the reward vector associated to separable decision rule π^t, where entries are defined by: $\forall s^t \in S, \theta^t \in \Theta^t$, $\boldsymbol{R}^{\pi^t}[s^t, \theta^t] = \boldsymbol{R}^{\pi^t(\theta^t)}[s^t]$. That is, the reward for choosing a decision rule at an occupancy state is defined as the expected value of rewards over underlying states and histories. We are now ready to formally define the occupancy MDP model that corresponds to the Dec-POMDP model. If we let $\triangle$ be the occupancy state space, and $\boldsymbol{A}$ be the separable decision rule space, then we define tuple $(\triangle, \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{R}, b^0)$ to be the occupancy MDP model that corresponds to the Dec-POMDP model.

Solving the occupancy MDP determines policy π^* that maximizes the expected cumulative reward over horizon T:

$$\pi^* \quad = \quad \arg\max_\pi \ \sum_{t=0}^{T-1} (\boldsymbol{R}^{\pi^t})^\top (b^0 \boldsymbol{P}^{\pi^{0:t-1}})$$

Similar to the traditional MDP model, the t-th value function v^t under the control of a policy is defined as follows: $\forall t = 0, 1, \cdots, T-1, \forall \boldsymbol{\eta}^t \in \triangle$

$$v^t(\boldsymbol{\eta}^t) \quad = \quad \max_{\pi^t \in \boldsymbol{A}} \ (\boldsymbol{R}^{\pi^t})^\top \boldsymbol{\eta}^t + v^{t+1}(\boldsymbol{\eta}^t \boldsymbol{P}^{\pi^t}), \tag{2}$$

where $v^T(\cdot) = 0$. The quantity at the right hand side of max operator of Equation (2) is referred to as the Q-value function defined by: $\forall \boldsymbol{\eta}^t \in \triangle$, and $\forall \pi^t \in \boldsymbol{A}$, $Q^{v^{t+1}}(\boldsymbol{\eta}^t, \pi^t) = (\boldsymbol{R}^{\pi^t})^\top \boldsymbol{\eta}^t + v^{t+1}(\boldsymbol{\eta}^t \boldsymbol{P}^{\pi^t})$.

Notice Equation (2) is Bellman's optimality equation for the occupancy MDP model. This leads to the following result.

Corollary 1. *Optimizing the occupancy MDP model is equivalent to optimizing the Dec-POMDP.*

This follows from expanding Equation (2) for each step of the problem and noting that due to the deterministic update of $\boldsymbol{\eta}$, the resulting policy is a sequence of separable decision rules that maximizes the expected value for the Dec-POMDP.

We can also show that the value function is a piecewise linear and convex (PWLC) function of the occupancy states.

Theorem 2. *The t-th value function $v^t : \triangle \mapsto \mathbb{R}$ solution of the Equation (2) is a PWLC function of the occupancy states.*

Similar to belief MDPs, the proof holds by induction given that v^T, $\boldsymbol{R}^{\pi^t}$ and $\boldsymbol{P}^{\pi^t}$ are linear and all operators involved in Equation (2) preserve the PWLC property. With Theorems 1 and 2 as a background, one can exploit advances in belief MDPs to solve occupancy MDPs. However, given that both occupancy states and decision rules are defined over the entire history space, the ability to scale up remains limited. In the remainder of this paper, we will use Dec-POMDPs and occupancy MDPs interchangeably.

4 Optimally solving occupancy MDPs

In this section, we first present an outline of a novel algorithm for solving Dec-POMDPs. We also propose a dimension reduction framework that replaces the high-dimensional history space by a lower-dimensional feature set, with no risk of losing: (PWLC) the PWLC property of the value function over occupancy states in the lower-dimensional feature set; and (LOSSLESS) the sufficiency of occupancy states in the lower-dimensional feature set.

4.1 The FB-HSVI algorithm

Our algorithm, termed feature-based heuristic search value iteration (FB-HSVI), exploits the PWLC structure of the value function within a general heuristic search algorithm. Many algorithms for solving belief MDPs take advantage of the PWLC property of the value function. One such heuristic search algorithm that preserves the ability to theoretically converge to an ϵ-optimal solution is heuristic search value iteration (HSVI) [Smith and Simmons, 2004]. We extend HSVI to solve the occupancy MDP, but other POMDP algorithms would also be applicable.

HSVI proceeds by creating trajectories, based on upper and lower bounds over the exact value function, denoted $\bar{v}$ and $\underline{v}$, respectively. Each such trajectory starts from the initial probability distribution b^0. HSVI always executes the best action

Algorithm 1: The FB-HSVI algorithm

initialize $\underline{v}$, and $\bar{v}$ and $\boldsymbol{\eta}^0 = b^0$
while $\bar{v}^0(\boldsymbol{\eta}^0) - \underline{v}^0(\boldsymbol{\eta}^0) > \epsilon$ **do**
  Explore($\boldsymbol{\eta}^0, \bar{v}, \underline{v}$)

Explore($\boldsymbol{\eta}^t, \bar{v}, \underline{v}$)
if $\bar{v}^t(\boldsymbol{\eta}^t) - \underline{v}^t(\boldsymbol{\eta}^t) > \epsilon/T$ **then**
  $\bar{\pi}^t \leftarrow \arg\max_{\pi^t \in \boldsymbol{A}} Q^{\bar{v}^{t+1}}(\boldsymbol{\eta}^t, \pi^t)$
  Update $(\bar{v}^t, \boldsymbol{\eta}^t, \bar{\pi}^t)$
  Explore(Compress($\boldsymbol{\eta}^t \boldsymbol{P}^{\bar{\pi}^t}$), $\bar{v}, \underline{v}$)
  Update $(\underline{v}, \boldsymbol{\eta}^t, \bar{\pi}^t)$

specified by the upper bound, and then selects the next belief state that maximizes the gap between the bounds. When the trajectory is finished the beliefs are backed up in reverse order. HSVI uses the following ideas: *value function guarantees*: value function representations for the lower bound $\underline{v}$ and upper bound $\bar{v}$ maintain valid bounds on the optimal value function v^*; *point-based backups*: value functions are improved by using backup operations only over a subset of the entire belief simplex; *trajectory-based heuristic search*: the upper bound guides search through trajectories of beliefs, focusing on regions where the bounds need improvement.

FB-HSVI, as described in Algorithm 1, uses the HSVI algorithmic framework, inheriting the above characteristics. FB-HSVI differs from HSVI in many aspects, including: (1) *efficient point-based backup*: we use an efficient implementation of the backup that typically circumvents the systematic enumeration of all separable decision rules [Dibangoye *et al.*, 2012; 2013]; (2) *low-dimensional feature set representations*: we replace the high-dimensional history space by a lower dimensional feature set, allowing for compact representations (represented with the Compress operation in the algorithm); and (3) *bound updates*: we take advantage of the determinism of occupancy MDPs to update forward the upper bound and backward the lower bound. These ideas together allow FB-HSVI to: (1) scale up to Dec-POMDP problems of unprecedented horizon size; (2) allow value function generalization over unvisited occupancy states; and (3) speed up the convergence to the optimal value function.

4.2 Lossless linear dimension reduction

In the following, we describe the dimension reduction framework that permits compact representations. This consists of defining metrics for considering histories to be equivalent. Then, we build upon this framework to introduce a feature selection algorithm that we use in practice. We introduce finite-order feature selection (FOFS) that operates incrementally during algorithm execution. That is, histories can be generated for the next step using forward search (as in Algorithm 1) with the current histories. The next-step histories can then be compressed by the methods discussed below, providing a reduced set of histories to be utilized in the algorithm. FOFS seeks to find representative sets of shorter horizon histories to replace the possible histories at a given step without any loss in solution quality.

Using feature mappings

The (local) *feature* associated with local history θ_i given occupancy state $\boldsymbol{\eta}^t$, denoted ϕ_{θ_i}, is a function mapping all pairs of states and histories of the other agents to reals, such that $\forall s, \forall \theta_{-i}\colon \phi_{\theta_i}[s,\theta_{-i}] = p(s,\theta_{-i},\theta_i|\boldsymbol{\eta}^t)$. Each feature ϕ_{θ_i} has a *label* associating it with a local history, ℓ_i or $L(\theta_i)$. Furthermore, we denote $\mathcal{C}_{\ell_i}$ the cluster of local histories with label ℓ_i. One way to make explicit the relationship between features and occupancy states is to consider occupancy in matrix form, where columns correspond to labels and rows are states and histories of the other agents.

In selecting features, we combine two criteria. The first criterion permits us to group together local histories that are *probabilistically equivalent* and treat them as a single local history. For agent i, we say that two local histories θ_i and θ_i' are probabilistically equivalent when $\phi_{\theta_i} \propto \phi_{\theta_i'}$. Intuitively, it can be seen that probabilistically equivalent histories provide the same information about the environment (including the other agents) from a single agent's perspective. It has been shown that it is lossless to condition action selection only upon label $L(\theta_i)$ instead of θ_i [Oliehoek *et al.*, 2009]. This allows a lossless compression of features with values generated as follows: $\forall s, \forall \theta_{-i}\colon \phi_{\ell_i}[s, L(\theta_{-i})] = \sum_{\theta_i \in \mathcal{C}_{\ell_i}} \phi_{\theta_i}[s,\theta_{-i}]$.

The second criterion enhances the generalization of the value function over unvisited occupancy states while preserving important information. This approach can be thought of as having a "sliding window" of history information. We can consider labels ℓ_i to be any length local history in set Θ_i. To reduce the space of labels, we define a finite-order label $\ell_{i,k}$ or $L^k(\ell_i)$ to be label ℓ_i's k rightmost (most recent) actions and observations. In addition, we denote $\mathcal{C}_{\ell_{i,k}}$ the cluster of labels with identical finite-order label $L^k(\ell_i)$. This allows a (possibly lossy) compression of features with values defined as follows $\forall s, \forall \ell_{-i}\colon \phi_{\ell_{i,k}}[s, L^k(\ell_{-i})] = \sum_{\ell_i \in \mathcal{C}_{\ell_{i,k}}} \phi_{\ell_i}[s,\ell_{-i}]$. Obviously, there is always a parameter k from 0 to $t-1$ so that features $\phi_{\ell_{i,k}}$ are lossless (such as $k = t-1$). Next, we present an algorithm that finds such a parameter k.

Algorithmic details

To guarantee the dimension reduction preserves the PWLC property, we consider linear dimension reductions [Boularias and Chaib-draa, 2008]. More specifically, we use clustering based on probabilistic equivalence and finite-order labels and generate a linear dimension reduction accordingly.

In order to ensure the sufficiency of occupancy states, we introduce the concept of *Hajnal lossless* compression. Informally, this property gives insight on how to select features that preserve the ability to eventually find an optimal policy. In defining this property, we need to bound the regret of searching for the optimal policy over the restricted space instead of the original high-dimensional history space. The regret depends on distance metrics that are used to measure the *closeness* between exact and approximate occupancy states.

Define metrics d and D as follows: $\forall \boldsymbol{\eta}_x^t, \forall \boldsymbol{\eta}_y^t$, $d(\boldsymbol{\eta}_x^t, \boldsymbol{\eta}_y^t) = \sum_{s\in S}\sum_{\theta\in\Theta} \left(P(s,\theta\,|\boldsymbol{\eta}_x^t) - P(s,\theta\,|\boldsymbol{\eta}_y^t)\right)^+$, where $(o)^+ = o$ if $o > 0$ and zero otherwise, and $D(\boldsymbol{\eta}_x^{0:T-1}, \boldsymbol{\eta}_y^{0:T-1}) = \sum_{t=0}^{T-1} d(\boldsymbol{\eta}_x^t, \boldsymbol{\eta}_y^t)$. Metric d, known as the *Hajnal measure*, has many applications in the theory of

ergodic chains [Paz, 1971]. Informally, $d(\boldsymbol{\eta}_x^t, \boldsymbol{\eta}_y^t)$ is the minimal *quantity* of probability that would have to be *re-assigned* in order to transform exact occupancy state $\boldsymbol{\eta}_x^t$ to occupancy state $\boldsymbol{\eta}_y^t$. Similarly, $D(\boldsymbol{\eta}_x^{0:T-1}, \boldsymbol{\eta}_y^{0:T-1})$, referred to as the *variational Hajnal measure* between sequences of occupancy states, is the minimal *cumulated quantity* of probability by which both sequences might differ.

The following theorem bounds the regret of searching for the optimal policy over the restricted space instead of the original high-dimensional history space. The proof is similar to the performance guarantee of the policy search algorithm (Theorem 1) [Bagnell *et al.*, 2003].

Theorem 3. *Let policies π and $\widehat{\pi}$ be the solution of the occupancy state MDP model when optimized on exact and approximate occupancy states, respectively. Then,*

$$\upsilon^{\widehat{\pi}}(\boldsymbol{\eta}^0) \quad \geq \quad \upsilon^{\pi}(\boldsymbol{\eta}^0) - T \cdot D(\boldsymbol{\eta}^{0:T-1}, \widehat{\boldsymbol{\eta}}^{0:T-1})||r||,$$

where $||r|| = \max_{s,a} \boldsymbol{R}^a[s] - \min_{s,a} \boldsymbol{R}^a[s]$.

Proof. If we let $\beta = \upsilon^{\pi}(\boldsymbol{\eta}^0) - \upsilon^{\widehat{\pi}}(\boldsymbol{\eta}^0)$, then we have that:

$$\begin{aligned}
\beta &= \textstyle\sum_{t=0}^{T-1} (\boldsymbol{R}^{\pi^t})^\top \boldsymbol{\eta}^t - \upsilon^{\widehat{\pi}}(\boldsymbol{\eta}^0) \\
&= \textstyle\sum_{t=0}^{T-1} [(\boldsymbol{R}^{\pi^t})^\top \boldsymbol{\eta}^t + \upsilon^{\widehat{\pi}^{t:T-1}}(\boldsymbol{\eta}^t) - \upsilon^{\widehat{\pi}^{t:T-1}}(\boldsymbol{\eta}^t)] - \upsilon^{\widehat{\pi}}(\boldsymbol{\eta}^0) \\
&= \textstyle\sum_{t=0}^{T-1} [(\boldsymbol{R}^{\pi^t})^\top \boldsymbol{\eta}^t + \upsilon^{\widehat{\pi}^{t+1:T-1}}(\boldsymbol{\eta}^t \boldsymbol{P}^{\pi^t}) - \upsilon^{\widehat{\pi}^{t:T-1}}(\boldsymbol{\eta}^t)] \\
&= \textstyle\sum_{t=0}^{T-1} [\upsilon^{\pi^t, \widehat{\pi}^{t+1:T-1}}(\boldsymbol{\eta}^t) - \upsilon^{\widehat{\pi}^{t:T-1}}(\boldsymbol{\eta}^t)]
\end{aligned}$$

Now, let $\beta^t = \upsilon^{\pi^t, \widehat{\pi}^{t+1:T-1}}(\boldsymbol{\eta}^t) - \upsilon^{\widehat{\pi}^{t:T-1}}(\boldsymbol{\eta}^t)$, and α^t be the hyperplane in value function $\upsilon^{\pi^t, \widehat{\pi}^{t+1:T-1}}$ that is maximal at $\boldsymbol{\eta}^t$, and $\widehat{\alpha}^t$ be the hyperplane in $\upsilon^{\widehat{\pi}^{t:T-1}}$ that is maximal at $\widehat{\boldsymbol{\eta}}^t$. Thus, given that $\widehat{\alpha}^t$ is maximal at $\widehat{\boldsymbol{\eta}}^t$, then we have that:

$$\begin{aligned}
\beta^t &= (\alpha^t)^\top \cdot \boldsymbol{\eta}^t - (\widehat{\alpha}^t)^\top \cdot \boldsymbol{\eta}^t \\
&= (\alpha^t)^\top \cdot \boldsymbol{\eta}^t - (\widehat{\alpha}^t)^\top \cdot \boldsymbol{\eta}^t + (\alpha^t)^\top \cdot \widehat{\boldsymbol{\eta}}^t - (\alpha^t)^\top \cdot \widehat{\boldsymbol{\eta}}^t \\
&\leq (\alpha^t)^\top \cdot \boldsymbol{\eta}^t - (\widehat{\alpha}^t)^\top \cdot \boldsymbol{\eta}^t + (\widehat{\alpha}^t)^\top \cdot \widehat{\boldsymbol{\eta}}^t - (\alpha^t)^\top \cdot \widehat{\boldsymbol{\eta}}^t \\
&= (\alpha^t - \widehat{\alpha}^t) \cdot (\boldsymbol{\eta}^t - \widehat{\boldsymbol{\eta}}^t) \;\leq\; T d(\boldsymbol{\eta}^t, \widehat{\boldsymbol{\eta}}^t) ||r||
\end{aligned}$$

The first inequality holds since $(\widehat{\alpha}^t)^\top \cdot \widehat{\boldsymbol{\eta}}^t \geq (\alpha^t)^\top \cdot \widehat{\boldsymbol{\eta}}^t$, where $\widehat{\boldsymbol{\eta}}^t$ and $\boldsymbol{\eta}^t$ need to be losslessly (de-)compressed to fit $\widehat{\alpha}^t$ and α^t, respectively. The last inequality holds using the Hölder inequality and Hajnal measure. By replacing β^t in the last expression of β and definition of D, the result holds. □

The bound (Theorem 3) depends on how *close*, in the Hajnal sense, exact and approximate occupancy states are. When the linear dimension reduction is lossless, that is $D(\boldsymbol{\eta}^{0:T-1}, \widehat{\boldsymbol{\eta}}^{0:T-1}) = 0$, the bound implies that there exists a policy in the restricted space that achieves performance as good as any policy in the high-dimensional history space.

It is easy to check whether or not a linear dimension reduction is Hajnal lossless. In fact, checking the property over the entire sequence $\boldsymbol{\eta}^{0:T-1}$ of occupancies consists of checking that $d(\boldsymbol{\eta}^t, \widehat{\boldsymbol{\eta}}^t) = 0$ holds for any occupancy $\boldsymbol{\eta}^t$ in that sequence. We call this a *local Hajnal lossless compression*. We are now ready to introduce our feature selection algorithm, namely *the finite-order feature selection* (FOFS), that automatically selects informative features based on the Hajnal lossless compression.

FOFS first compresses occupancy states based on probabilistic equivalence as in [Oliehoek *et al.*, 2009]. Given occupancy state $\boldsymbol{\eta}^t$, for each agent i, we replace histories with the associated labels, thereby replacing features ϕ_{θ_i} by corresponding feature ϕ_{ℓ_i}. Let $\tilde{\boldsymbol{\eta}}^t$ be the occupancy state with probabilistically equivalent histories removed. Clearly, there is no risk of loosing either property (PWLC) or property (LOSSLESS) by planning over $\tilde{\boldsymbol{\eta}}^t$ instead of $\boldsymbol{\eta}^t$ since this compression is lossless and linear. However, $\tilde{\boldsymbol{\eta}}^t$ is still defined over a high-dimensional label set. This significantly limits the generalization of the value function over unvisited occupancy states and highlights the necessity of a dimension reduction transformation.

FOFS continues by compressing occupancy $\tilde{\boldsymbol{\eta}}^t$ into $\widehat{\boldsymbol{\eta}}^t$ using finite-order labels, starting with $k = 0$. This compression is linear so it preserves property (PWLC), but it can be lossy. When local Hajnal lossless compression holds, that is $\forall s, \forall \ell\colon P(s, \ell\,|\tilde{\boldsymbol{\eta}}^t) - P(s, L^k(\ell)|\widehat{\boldsymbol{\eta}}^t) = 0$, FOFS returns $\widehat{\boldsymbol{\eta}}^t$ with labels ℓ replaced with finite-order labels $L^k(\ell)$. Otherwise, it re-computes $\widehat{\boldsymbol{\eta}}^t$ with increased length $k = k+1$. The algorithm iterates until the local Hajnal lossless compression holds. In the worst case, FOFS terminates with $k = t-1$, and returns a lossless representation $\widehat{\boldsymbol{\eta}}^t$ of the original occupancy state $\boldsymbol{\eta}^t$. When $k < t-1$, occupancy state $\widehat{\boldsymbol{\eta}}^t$ is defined as a probability distribution over states and l-length histories where $l \leq k < t-1$, and thus the compression is a dimension reduction.

In both cases, to compress an occupancy state for a given criterion, FOFS proceeds by clustering histories (or labels) of one agent at a time assuming the other agents' labels are fixed (as full extensions of previous step histories initially and then as their compressed length). It repeats this procedure for each agent alternatively until no more clustering can occur.

5 Experiments

We ran FB-HSVI on a Mac OSX machine with 2.4GHz Dual-Core Intel and 2GB of available RAM. We solved the backup using constraint optimization with the toulbar2 solver. We initialized the upper bound as the value function of the underlying MDP; and the lower bound as the value function of the blind policy. We evaluate three variants of FB-HSVI(ρ): the first variant, FB-HSVI(0), is FB-HSVI without compression or efficient backups; the second variant, FB-HSVI(1) utilizes $k = \infty$ (full histories), but incorporates efficient point-based backups; and FB-HSVI(2) uses both efficient backups and histories compressed with FOFS.

We evaluated our algorithms on five benchmark problems from the literature: multi-agent tiger; recycling-robot; grid-small; cooperative box-pushing; and mars rovers. These are the largest and most difficult benchmarks from the literature. For each benchmark, we compared our algorithms with state-of-the-art exact solvers: GMAA*-ICE [Spaan *et al.*, 2011], IPG [Amato *et al.*, 2009], MILP [Aras and Dutech, 2010], and LPC [Boularias and Chaib-draa, 2008]. IPG and LPC perform dynamic programming, GMAA*-ICE performs heuristic search and MILP is a mixed integer linear programming method. Results for GMAA*-ICE (provided by Matthijs Spaan), IPG, MILP and LPC were conducted on different machines. Because of this, the timing results are not directly comparable to the other methods, but are likely to

T	MILP	LPC	IPG	ICE	FB-HSVI(ρ)			$\upsilon_\epsilon(\boldsymbol{\eta}^0)$
					0	1	2	
The multi-agent tiger problem ($\|S\| = 2, \|Z\| = 4, \|A\| = 9, K = 3$)								
2	–	0.17	0.32	0.01	0.05	0.03	**0.03**	−4.00
3	4.9	1.79	55.4	0.01	2.17	0.06	**0.40**	5.2908
4	72	534	2286	108	9164	2.66	**1.36**	4.8027
5				347		22.2	**9.65**	7.0264
6						171.3	**24.42**	10.381
7							**33.11**	**9.9935**
8							**41.21**	**12.217**
9							**58.51**	**15.572**
10							**65.57**	**15.184**
The recycling-robot problem ($\|S\| = 4, \|Z\| = 4, \|A\| = 9, K = 1$)								
2	–	–	0.30	36	0.03	0.02	**0.01**	7.000
3	–	–	1.07	36	0.05	0.47	**0.10**	10.660
4	–	–	42.0	72	0.85	0.65	**0.30**	13.380
5	–	–	1812	72	1.52	0.87	**0.34**	16.486
10					5.06	2.83	**0.52**	31.863
30					62.8	37.9	**1.13**	93.402
70						78.1	**2.13**	216.47
100						259	**2.93**	**308.78**
The grid-small problem ($\|S\| = 16, \|Z\| = 4, \|A\| = 25, K = 4$)								
2	0.65	–	0.18	36	116.1	0.1	**0.1**	0.9100
3	1624	–	4.09	1512	3024	6.09	**0.73**	1.5504
4		–	77.4	242605		12.85	**1.39**	2.2415
5						148.2	**2.40**	2.9704
6						319.8	**7.12**	3.7171
7						645.1	**58.25**	**4.4657**
8							**65.12**	**5.2319**
9							**71.38**	**5.9878**
The cooperative box-pushing problem ($\|S\| = 100, \|Z\| = 16, \|A\| = 25, K = 3$)								
2	–	–	1.07	36	0.1	0.1	**0.1**	17.600
3	–	–	6.43	540	2026	0.64	**0.457**	66.081
4	–	–	1138	2596		3.16	**0.622**	98.593
5						16.72	**5.854**	**107.72**
6						274.6	**70.67**	**120.67**
7						462.4	**74.40**	**156.42**
8						751.5	**95.38**	**191.22**
9							**105.7**	**208.19**
10							**168.5**	**220.45**
The mars-rovers problem ($\|S\| = 256, \|Z\| = 81, \|A\| = 36, K = 3$)								
2	–	–	83	1.0	0.21	0.09	**0.10**	5.80
3	–	–	389	1.0	2.84	0.21	**0.23**	9.38
4				103	104.2	1.73	**0.47**	10.18
5						6.38	**0.82**	**13.26**
6						8.16	**3.97**	**18.62**
7						11.13	**5.81**	**20.90**
8						35.49	**22.8**	**22.47**
9						57.47	**26.5**	**24.31**
10						316.2	**62.7**	**26.31**

Table 1: Experiments comparing the computation times (in seconds) of all exact solvers. Time limit violations ($1000s$) are indicated by " ", and "–" indicate unknown values.

only differ by a small constant factor.

The results can be seen in Table 1. For each algorithm we reported the computation time, which includes the time to compute heuristic values when appropriate since all algorithms do not use the same heuristics. We also reported the ϵ-optimal expected cumulative reward $\upsilon_\epsilon(\boldsymbol{\eta}^0)$ (where $\epsilon = 0.01$) at the initial occupancy state. Furthermore, we also reported K the maximum parameter k found by FOFS for each benchmark (representing the maximum history length used). Table 1 clearly shows that FB-HSVI(2) allows for significant improvement over the state-of-the-art solvers: for all tested benchmarks we provide results for longer horizons than have been solved previously (the bold entries).

There are several reasons for FB-HSVI(2)'s performance: first, we search in the space of policies mapping lower-dimensional features to actions, whereas the other solvers search in the space of policies mapping histories to actions; we use a value function mapping occupancy states to reals allowing it to generalize the value function over unvisited occupancies whereas all other solvers use value functions mapping partial policies to reals; finally, we replace the brute-force backup by an efficient constraint optimization approach.

To better understand FB-HSVI, we compare three variants $\rho = 0, 1, 2$ presented earlier. For $\rho = 0$, we quickly run out of time since we explicitly enumerate all separable decision rules. For $\rho = 1$, we scale up to larger horizons but convergence slows down because the value function is defined over the full history space. For $\rho = 2$, we enhance the generalization of the value function over unvisited occupancy states by means of finite-order labels. This circumvents unnecessary backups and speeds up convergence.

6 Conclusion

This paper explores new theory and algorithms for solving Dec-POMDPs. This theory builds upon a novel transformation of Dec-POMDPs to continuous-state and deterministic MDPs, generalizing previous theory and algorithmic results. In particular, we demonstrate two important results: (1) the distribution over both the states and the histories, namely occupancy states, are sufficient for optimal planning in Dec-POMDPs; (2) the value function is piecewise linear and convex over the occupancy states. With this new formulation, POMDP methods can, for the first time, be directly applied to Dec-POMDPs. This permits us to no longer maintain an explicit policy representation and allows us to exploit structure in the value function. We also describe a novel algorithm for this representation, extending to large state and action spaces by dimension reduction and constraint optimization. This algorithm, termed feature-based heuristic search value iteration or FB-HSVI, was shown to scale up to large planning horizons, reducing the computation time by multiple orders of magnitude over previous approaches.

In the future, we plan to explore applying this theory and algorithm to subclasses of Dec-POMDPs and larger teams of agents. For instance, we have already shown that occupancy states over just states (and not agent histories) can be used in transition and observation independent Dec-MDPs to greatly increase scalability [Dibangoye *et al.*, 2012; 2013]. We would like to explore using occupancy states over observation histories (rather than action-observation histories), which were shown to be sufficient (along with action-observation histories) simultaneous to this work [Oliehoek, 2013]. In addition, we think occupancy states together with graphical models could help exploit the locality of interaction among agents statically (as in ND-POMDPs [Nair *et al.*, 2005; Kumar and Zilberstein, 2009]) or dynamically (as in [Canu and Mouaddib, 2011]). Furthermore, the scalability of FB-HSVI is encouraging and we will pursue additional methods to automatically compute compact representations.

7 Acknowledgements

We would like to thank Matthijs Spaan for providing results for GMAA*-ICE as well as Frans Oliehoek and the reviewers for their helpful comments. Research supported in part by AFOSR MURI project #FA9550-09-1-0538.

References

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

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

[Bagnell *et al.*, 2003] J. Andrew Bagnell, Sham Kakade, Andrew Ng, and Jeff Schneider. Policy search by dynamic programming. In *NIPS*, volume 16, 2003.

[Bernstein *et al.*, 2002] Daniel S. Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, 27(4), 2002.

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

[Boularias and Chaib-draa, 2008] Abdeslam Boularias and Brahim Chaib-draa. Exact dynamic programming for decentralized POMDPs with lossless policy compression. In *ICAPS*, pages 20–27, 2008.

[Canu and Mouaddib, 2011] Arnaud Canu and Abdel-Illah Mouaddib. Collective decision under partial observability - a dynamic local interaction model. In *IJCCI (ECTA-FCTA)*, pages 146–155, 2011.

[De Farias and Van Roy, 2003] D. P. De Farias and B. Van Roy. The linear programming approach to approximate dynamic programming. *Operations Research*, 51(6):850–865, 2003.

[Dibangoye *et al.*, 2011] Jilles S. Dibangoye, Abdel-Illah Mouaddib, and Brahim Chaib-draa. Toward error-bounded algorithms for infinite-horizon DEC-POMDPs. In *AAMAS*, pages 947–954, 2011.

[Dibangoye *et al.*, 2012] Jilles S. Dibangoye, Christopher Amato, and Arnaud Doniec. Scaling up decentralized MDPs through heuristic search. In *UAI*, pages 217–226, 2012.

[Dibangoye *et al.*, 2013] Jilles S. Dibangoye, Christopher Amato, Arnaud Doniec, and François Charpillet. Producing efficient error-bounded solutions for transition independent decentralized MDPs. In *AAMAS*, 2013.

[Hansen *et al.*, 2004] Eric A. Hansen, Daniel S. Bernstein, and Shlomo Zilberstein. Dynamic programming for partially observable stochastic games. In *AAAI*, pages 709–715, 2004.

[Howard, 1960] Ronald A. Howard. *Dynamic Programming and Markov Processes*. The M.I.T. Press, 1960.

[Kumar and Zilberstein, 2009] Akshat Kumar and Shlomo Zilberstein. Constraint-based dynamic programming for decentralized POMDPs with structured interactions. In *AAMAS*, pages 561–568, 2009.

[Nair *et al.*, 2005] Ranjit Nair, Pradeep Varakantham, Milind Tambe, and Makoto Yokoo. Networked distributed POMDPs: A synthesis of distributed constraint optimization and POMDPs. In *AAAI*, pages 133–139, 2005.

[Oliehoek *et al.*, 2008] Frans A. Oliehoek, Matthijs T. J. Spaan, and Nikos A. Vlassis. Optimal and approximate Q-value functions for decentralized POMDPs. *JAIR*, 32:289–353, 2008.

[Oliehoek *et al.*, 2009] Frans A. Oliehoek, Shimon Whiteson, and Matthijs T. J. Spaan. Lossless clustering of histories in decentralized POMDPs. In *AAMAS*, pages 577–584, 2009.

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

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

[Paz, 1971] Azaria Paz. *Introduction to probabilistic automata (Computer science and applied mathematics)*. Academic Press, Inc., Orlando, FL, USA, 1971.

[Powell, 2007] Warren B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality (Wiley Series in Probability and Statistics)*. Wiley-Interscience, 2007.

[Shani *et al.*, 2012] Guy Shani, Joelle Pineau, and Robert Kaplow. A survey of point-based POMDP solvers. *JAAMAS*, pages 1–51, 2012.

[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] Trey Smith and Reid Simmons. Heuristic search value iteration for POMDPs. In *UAI*, pages 520–527, 2004.

[Spaan *et al.*, 2011] Matthijs T. J. Spaan, Frans A. Oliehoek, and Christopher Amato. Scaling up optimal heuristic search in Dec-POMDPs via incremental expansion. In *IJCAI*, pages 2027–2032, 2011.

[Szer *et al.*, 2005] Daniel Szer, Francois Charpillet, and Shlomo Zilberstein. MAA*: A heuristic search algorithm for solving decentralized POMDPs. In *UAI*, pages 568–576, 2005.