Scaling Up Optimal Heuristic Search in Dec-POMDPs via Incremental Expansion

Matthijs T.J. Spaan
Inst. for Systems and Robotics
Instituto Superior Técnico
Lisbon, Portugal
mtjspaan@isr.ist.utl.pt

Frans A. Oliehoek
CSAIL
Massachusetts Inst. of Technology
Cambridge, MA 02139, USA
fao@csail.mit.edu

Christopher Amato
Aptima, Inc.
Woburn, MA 01801, USA
camato@aptima.com

Abstract

Planning under uncertainty for multiagent systems can be formalized as a decentralized partially observable Markov decision process. We advance the state of the art for optimal solution of this model, building on the Multiagent A* heuristic search method. A key insight is that we can avoid the full expansion of a search node that generates a number of children that is doubly exponential in the node's depth. Instead, we incrementally expand the children only when a next child might have the highest heuristic value. We target a subsequent bottleneck by introducing a more memory-efficient representation for our heuristic functions. Proof is given that the resulting algorithm is correct and experiments demonstrate a significant speedup over the state of the art, allowing for optimal solutions over longer horizons for many benchmark problems.

1 Introduction

Planning under uncertainty for multiagent systems is an important problem in artificial intelligence, as agents may often possess uncertain information while sharing their environment with other agents. Due to stochastic actions and noisy sensors, agents must reason about many possible outcomes and the uncertainty surrounding them. In cooperative systems, finding optimal joint plans is especially challenging when each agent must choose actions based solely on local knowledge due to nonexistent or noisy communication. Possible application domains include multi-robot teams, communication networks, load balancing, and other problems in which agents need to coordinate under uncertain conditions.

The decentralized partially observable Markov decision process (Dec-POMDP) is a formal model for such planning problems. In this paper we consider the optimal solution of Dec-POMDPs over a finite horizon. Unfortunately, optimal solution methods and even bounded approximations (ϵ-optimal solutions) [Rabinovich *et al.*, 2003] suffer from doubly-exponential complexity (NEXP-Complete); the search space for horizon $h + 1$ is exponentially larger than the one for horizon h. Even though the high worst-case complexity results preclude optimal methods from being applicable for some larger problems, there are several reasons to be interested in optimal solutions: 1) As approximate algorithms come with no guarantees, optimal methods are necessary as a tool to analyze the performance of approximate algorithms. 2) Most successful approximate algorithms (e.g., [Emery-Montemerlo *et al.*, 2004; Seuken and Zilberstein, 2007; Wu *et al.*, 2011]) are based on optimal solution methods, so algorithmic improvements to the latter are likely to transfer to the former. 3) Optimal techniques can give insight in the nature of problems and their solutions. For instance, previous work on optimal methods generated the insight that certain properties of the BroadcastChannel problem make it easier to solve [Oliehoek *et al.*, 2009]. 4) They are of interest for solving small problems that arise naturally or as part of a decomposition. Moreover, many problem instances are much easier to solve than the worst-case complexity suggests [Allen and Zilberstein, 2007], allowing optimal solutions to be practical.

We provide significant advances to the state of the art in optimal Dec-POMDP solution methods by extending Multiagent A* (MAA*) [Szer *et al.*, 2005] —which performs an A* search through the tree of possible partial joint policies— and derived methods with a new technique for incremental expansion of search tree nodes. Expanding a node in this context entails generating all possible children, which is a major source of intractability since the number of such children is doubly exponential in the depth of the node. In practice, however, only a small number of the generated nodes may actually be queried during the search. Our key insight is that if a method is able to *incrementally* generate children in order of their heuristic value, not all nodes need to expanded at once.

As with any A* method, our approach's performance depends on the tightness of the heuristic. In many problems the upper bound provided by the value function of the underlying MDP (Q_{MDP}) is not tight enough for heuristic search to be effective [Oliehoek *et al.*, 2008]. Other heuristics are tighter, such as those based on the underlying POMDP solution (Q_{POMDP}) or the value function resulting from assuming 1-step-delayed communication (Q_{BG}). However, they require storing values for all joint action-observation histories or representing them as a potentially exponential number of vectors. A crucial insight is that the number of values stored in a tree-based representation grows exponentially when moving forward in time, while the size of a vector-based representation grows in the opposite direction. We exploit this insight by introducing a hybrid representation that is more compact.

In this work, we integrate the incremental expansion idea in GMAA* with incremental clustering (GMAA*-IC), an MAA* extension that uses lossless history clustering for improved scalability [Oliehoek *et al.*, 2009]. The resulting algorithm is called GMAA*-ICE as it provides incremental clustering and expansion. We prove that GMAA*-ICE is correct and expands search nodes in the same order as the original method. We show the efficacy of our methods on a suite of benchmark problems, demonstrating a significant speedup over the state of the art. In many cases GMAA*-ICE provides the optimal solution over longer horizons than those previously solved. In particular, incremental expansion provides leverage in those problem domains in which history clustering is less effective.

The remainder of the paper is organized as follows. We begin in Sec. 2 with background on Dec-POMDPs. Sec. 3 introduces GMAA*-ICE, and in Sec. 4 we prove its correctness. The hybrid representation is introduced in Sec. 5 and Sec. 6 presents experimental results. Lastly, Sec. 7 presents conclusions and future work.

2 Background

A *decentralized partially observable Markov decision process (Dec-POMDP)* consists of a set of n agents, a finite set of states $\mathcal{S}$, a set $\mathcal{A} = \times_i \mathcal{A}_i$ of joint actions $\mathbf{a} = \langle a_1, \ldots, a_n \rangle$, a transition function specifying $\Pr(s'|s,\mathbf{a})$, a reward function $R(s,\mathbf{a})$, a set $\mathcal{O} = \times_i \mathcal{O}_i$ of joint observations $\mathbf{o} = \langle o_1, \ldots, o_n \rangle$, an observation function specifying $\Pr(\mathbf{o}|\mathbf{a},s')$, a planning horizon h, and an initial state distribution $\boldsymbol{b}^0$.

The goal of a Dec-POMDP is to find a decentralized deterministic joint policy $\boldsymbol{\pi} = \langle \pi_1, \ldots, \pi_n \rangle$. Each individual policy π_i maps from *local* observation-histories (OH) $\vec{o}_i^t = \left(o_i^1, \ldots, o_i^t\right)$ to actions: $\pi_i(\vec{o}_i^t) = a_i^t$. An individual policy π_i is a sequence of decision rules $\pi_i = (\delta_i^0, \delta_i^1, \ldots, \delta_i^{h-1})$, where δ_i^t maps from length-t OHs to actions. We will also consider action-observation histories (AOHs) $\vec{\theta}_i^t = \left(a_i^0, o_i^1, a_i^1, \ldots, a_i^{t-1}, o_i^t\right)$. The optimal joint policy $\boldsymbol{\pi}^*$ maximizes the expected cumulative reward. For a more detailed introduction to Dec-POMDPs see, e.g., [Oliehoek *et al.*, 2008; Seuken and Zilberstein, 2008].

We build upon GMAA*-Cluster [Oliehoek *et al.*, 2009], which in turn is based on MAA* [Szer *et al.*, 2005]. These methods search over partial, or *past*, joint policies $\boldsymbol{\varphi}^t$ that specify the joint policy up to stage t: $\boldsymbol{\varphi}^t = (\boldsymbol{\delta}^0, \boldsymbol{\delta}^1, \ldots, \boldsymbol{\delta}^{t-1})$, where $\boldsymbol{\delta}^i$ is the joint decision rule for the i-th stage. For such a $\boldsymbol{\varphi}^t$, we can compute a heuristic value $\widehat{V}$ by computing $V^{0\ldots(t-1)}(\boldsymbol{\varphi}^t)$, the actual expected reward over the first t stages, and adding a heuristic value $H(\boldsymbol{\varphi}^t)$ for the remaining stages. When the heuristic is *admissible* (i.e., a guaranteed overestimation), it is possible to perform standard A* search: select the node $\boldsymbol{\varphi}^t$ with the highest $\widehat{V}(\boldsymbol{\varphi}^t)$ and expand it by generating all child nodes $\boldsymbol{\varphi}^{t+1} = \langle \boldsymbol{\varphi}^t \circ \boldsymbol{\delta}^t \rangle$ that can be formed by appending a joint decision rule $\boldsymbol{\delta}^t$ to $\boldsymbol{\varphi}^t$.

Alternatively, it is possible to interpret this as enumeration of the joint policies $\boldsymbol{\beta}$ of a *collaborative Bayesian game (CBG)* $B(\boldsymbol{\varphi}^t)$ constructed for $\boldsymbol{\varphi}^t$ [Oliehoek *et al.*, 2008]. In this perspective, an AOH corresponds to a *type* and a decision rule corresponds to a joint policy $\boldsymbol{\beta}$ for the CBG: $\boldsymbol{\beta} \equiv \boldsymbol{\delta}^t$ with heuristic value given by

$$\widehat{V}(\boldsymbol{\beta}) = \sum_{\vec{\boldsymbol{\theta}}^t} \Pr(\vec{\boldsymbol{\theta}}^t|\boldsymbol{\varphi}^t,\boldsymbol{b}^0) \widehat{Q}(\vec{\boldsymbol{\theta}}^t,\boldsymbol{\beta}(\vec{\boldsymbol{\theta}}^t)), \tag{1}$$

where $\Pr(\vec{\boldsymbol{\theta}}^t|\boldsymbol{\varphi}^t,\boldsymbol{b}^0)$ is the probability of a joint AOH, called a joint *type*; $\widehat{Q}(\vec{\boldsymbol{\theta}}^t,\mathbf{a})$ is a heuristic payoff function for the CBG; and $\boldsymbol{\beta}^t(\vec{\boldsymbol{\theta}}^t) = \langle \beta_i(\vec{\theta}_i^t) \rangle_{i=1\ldots n}$ denotes the joint action that results from application of the individual BG-policies to the individual AOH $\vec{\theta}_i^t$ specified by $\vec{\boldsymbol{\theta}}^t$.

The valuation of a child node $\boldsymbol{\varphi}^{t+1} = \langle \boldsymbol{\varphi}^t \circ \boldsymbol{\beta} \rangle$ is given by

$$\widehat{V}(\boldsymbol{\varphi}^{t+1}) = V^{0\ldots(t-1)}(\boldsymbol{\varphi}^t) + \widehat{V}(\boldsymbol{\beta}), \tag{2}$$

where now the expected immediate reward for stage t is represented within the heuristic $\widehat{V}(\boldsymbol{\beta})$. When $\widehat{Q}$ faithfully represents the expected immediate reward, this reformulation is exactly equal to regular MAA* [Oliehoek *et al.*, 2008].

The redefinition of MAA* to work on CBGs is exploited by GMAA*-Cluster by clustering individual types in a CBG in such a way that the solution of the clustered CBG corresponds to a solution of the original CBG. This can result in great computational savings, since the number of $\boldsymbol{\beta}$ is exponential in the number of types. To avoid having to cluster an exponential number of types (corresponding to the number of OHs) for each CBG, GMAA*-IC performs *incremental clustering* by bootstrapping from the clustered CBG for the previous stage.

A remaining source of complexity in these methods is the full expansion of search nodes; when no clustering is possible, the number of $\boldsymbol{\delta}^t$ (used to form the children of a node $\boldsymbol{\varphi}^t$ at depth t in the search tree) is doubly exponential in t. In an attempt to counter this problem, for the last stage MAA* generates the child nodes one by one until a node is found with value equal to its parent's heuristic value. If this happens, no other siblings will have to be generated. Unfortunately, this method does not provide much leverage in practice, since it is unlikely that a child node will have the same heuristic value as its parent and, even if one does, there is no effective way to find such a child [Seuken and Zilberstein, 2008]. Therefore Seuken and Zilberstein [2008] argue that MAA* "can at best solve problems whose horizon is only 1 greater than those that can already be solved by naïve brute force search."

In this paper, we address these problems. That is, we provide efficient incremental expansion through a method that is able to select the highest ranked child at all stages. Moreover, we combine this approach with clustering of histories.

3 Incremental Expansion

Recently, new methods for solving CBGs have been developed [Kumar and Zilberstein, 2010; Oliehoek *et al.*, 2010] that can provide speedups of multiple orders of magnitude over brute force search (enumeration). Unfortunately, MAA* has not been able to profit from these methods: in order to guarantee optimality, it relies on expansion of *all* (child nodes corresponding to all) joint BG-policies $\boldsymbol{\beta}$ for the intermediate stages.[1] However, many of the expanded child nodes

[1] For the last stage, clearly it is possible to only generate the best child node of $\boldsymbol{\varphi}^{h-1}$ by appending the optimal solution of the CBG.

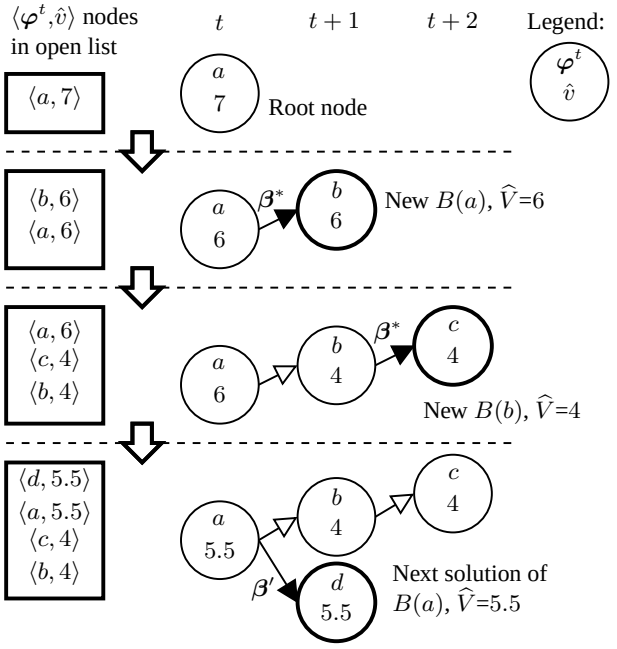


Figure 1: Illustration of incremental expansion, which shows the evolution of the open list ($\boldsymbol{\varphi}^t$ are indexed by letters).

may never be selected for further expansion. The key observation is the following:

Observation 1. *If we have a way to generate the children in increasing heuristic order and that heuristic is admissible, we do not have to expand all the children.*

We discuss this in more detail below, starting with a formalization of the relative heuristic values of two child nodes.

Lemma 1. *Given two joint BG policies $\boldsymbol{\beta}, \boldsymbol{\beta}'$ for a CBG $B^t(\boldsymbol{\varphi}^t)$, if $\widehat{V}(\boldsymbol{\beta}) \geq \widehat{V}(\boldsymbol{\beta}')$, then for the corresponding child nodes $\widehat{V}(\boldsymbol{\varphi}^{t+1}) \geq \widehat{V}(\boldsymbol{\varphi}^{t+1\prime})$.*

Proof. This holds directly by the definition of $\widehat{V}(\boldsymbol{\varphi}^t)$

$$\begin{aligned}\widehat{V}(\boldsymbol{\varphi}^{t+1}) &= V^{0\ldots(t-1)}(\boldsymbol{\varphi}^t) + \widehat{V}(\boldsymbol{\beta}) \\ &\geq V^{0\ldots(t-1)}(\boldsymbol{\varphi}^t) + \widehat{V}(\boldsymbol{\beta}') = \widehat{V}(\boldsymbol{\varphi}^{t+1\prime}),\end{aligned}$$

as given by (2). □

It follows directly that, if for $B(\boldsymbol{\varphi}^t)$ we use a CBG solver that can generate a sequence of policies $\boldsymbol{\beta}, \boldsymbol{\beta}', \ldots$ such that

$$\widehat{V}(\boldsymbol{\beta}) \geq \widehat{V}(\boldsymbol{\beta}') \geq \ldots$$

then, for the sequence of corresponding children

$$\widehat{V}(\boldsymbol{\varphi}^{t+1}) \geq \widehat{V}(\boldsymbol{\varphi}^{t+1\prime}) \geq \ldots.$$

Exploiting this knowledge, we can expand only the first child $\boldsymbol{\varphi}^{t+1}$, compute its $\widehat{V}(\boldsymbol{\varphi}^{t+1})$ and set the value of the parent node to $q.\hat{v} \leftarrow \widehat{V}(\boldsymbol{\varphi}^{t+1})$, since we know that all the unexpanded siblings will have $\widehat{V}$ lower or equal to that. As such, we can reinsert q into P to act as a *placeholder* for all its non-expanded children. To ensure that children are expanded before their parents, we break ties in a consistent manner, ranking children nodes higher in case of equal value. Fig. 1 illustrates incremental expansion.

We integrate incremental expansion in GMAA*-IC resulting in GMAA* *with incremental clustering and expansion* (GMAA*-ICE). It performs an A* search over nodes $q = \langle \boldsymbol{\varphi}^t, \hat{v}, \text{PH} \rangle$, where PH is a boolean indicating whether the node is a placeholder. At every iteration, the heuristically highest ranked q is selected from an open list P and expanded. When a new best full joint policy is found, the lower bound $\underline{\text{v}}^{GMAA}$ is updated. Each time a new CBG is constructed, it is built by extending the CBG for the parent node and then applying lossless clustering. However, rather than expanding all children, GMAA*-ICE requests only the next solution $\boldsymbol{\beta}$ of an *incremental CBG solver*, which is then used to construct a single child $\boldsymbol{\varphi}^{t+1} = \langle \boldsymbol{\varphi}^t \circ \boldsymbol{\beta} \rangle$.

For the incremental CBG solver, we use the BAGABAB algorithm [Oliehoek *et al.*, 2010], which performs a second (nested) A* search, but now over (partially specified) CBG policies.[2] The solver for $\boldsymbol{\varphi}^t$ is initialized with lower bound

$$\underline{\text{v}}^{CBG} = \underline{\text{v}}^{GMAA} - V^{0\ldots(t-1)}(\boldsymbol{\varphi}^t), \tag{3}$$

and, in case of the last stage $t = h - 1$, upperbound

$$\bar{\text{v}}^{CBG} = \widehat{V}(\boldsymbol{\varphi}^{h-1}) - V^{0\ldots(h-2)}(\boldsymbol{\varphi}^{h-1}). \tag{4}$$

This can be used to stop expanding when we find a lower bound equal to the upper bound $\bar{\text{v}}^{CBG} = V(\boldsymbol{\beta})$, as in the original A*. Note that each time when asking BAGABAB for a next solution, $\underline{\text{v}}^{CBG}$ is reset by re-evaluating (3), because $\underline{\text{v}}^{GMAA}$ may have changed since the last solution was delivered. Then it continues searching by selecting the heuristically best-ranked node from its own internal open list.

Additional details can be found in an expanded version of this paper [Spaan *et al.*, 2011].

4 Theoretical Guarantees

We shall now prove some properties of GMAA*-ICE. We say that two search algorithms are *search-equivalent* if they select exactly the same set of nodes to expand in the search tree. We will show that the IC and ICE variants are search-equivalent. To do so, we will talk about equivalence of the open lists maintained. The open list P maintained by IC only contains non-expanded nodes q. That of ICE, P^{IE}, contains both non-expanded nodes q and placeholders (previously expanded nodes), $\bar{q}$. We use Q and $\bar{Q}$ to denote the respective (ordered) subsets of P^{IE}. We think of these open lists as ordered sets of heuristic values and their associated nodes.

Definition 1. *P and P^{IE} are* equivalent, *$P \equiv \text{P}^{IE}$, when:*

1. *$Q \subseteq P$.*
2. *The q's have the same ordering: P.remove$(P \setminus Q) = Q$. (A.remove(B) removes the elements of B from A without changing A's ordering.)*

[2] In principle, GMAA*-ICE can use any CBG solver that is able to incrementally deliver all $\boldsymbol{\beta}$ in descending order of $\widehat{V}(\boldsymbol{\beta})$.

3. *Nodes not present in P^{IE} instead have a placeholder, $\forall q = \langle \boldsymbol{\varphi}^t, \hat{v}_q, \textit{false} \rangle \in (P \setminus Q) : \exists \bar{q} = \langle \boldsymbol{\varphi}^{t-1}, \hat{v}_{\bar{q}}, \textit{true} \rangle \in \bar{Q}$ such that: $\bar{q}$ is the parent of q ($\boldsymbol{\varphi}^t = \langle \boldsymbol{\varphi}^{t-1} \circ \boldsymbol{\beta} \rangle$), and $\bar{q}$ is ranked higher: $\hat{v}_{\bar{q}} \geq \hat{v}_q$.*
4. *There are no other placeholders.*

Let us write IT-IC(P) and IT-ICE(P^{IE}) for one iteration of the respective algorithms. Let IT-ICE* denote the operation that repeats IT-ICE as long as a placeholder $\bar{q}$ was selected.

Lemma 2. *If $P \equiv P^{IE}$, then executing IT-IC(P) and IT-ICE*(P^{IE}) will lead to new open lists that again are equivalent: $P' \equiv P^{IE\prime}$.*

Proof. When IT-ICE* selects a placeholder $\bar{q}$, it will generate child q' that was already present in P (due to property 3 and 4 of def. 1) and insert it at the proper location, thereby preserving properties 1 and 2. If there are remaining unexpanded children of $\bar{q}$, IT-ICE* will reinsert $\bar{q}$ with an updated heuristic value $\bar{q}.\hat{v} \leftarrow q'.\hat{v}$ which is guaranteed to upper bound the value of unexpanded siblings q'' since $q'.\hat{v} = \widehat{V}(q'.\boldsymbol{\varphi}) \geq \widehat{V}(q''.\boldsymbol{\varphi}) = q''.\hat{v}$ (preserving properties 3 and 4).

When IT-ICE* finally selects a non-placeholder q, it is guaranteed to be the same q as selected by IT-IC (due to property 1 and 2). Expansion in ICE will generate 1 child q' (again, inserted at the same relative location as in IC) and insert placeholder $\bar{q} = \langle q.\boldsymbol{\varphi}, q'.\hat{v}, \text{true} \rangle$ for the other siblings q'' (again preserving properties 3 and 4). □

Theorem 1. GMAA*-ICE *and* GMAA*-IC *are search-equivalent.*

Proof. This follows directly from the proof of Lemma 2: Both algorithms initialize with the same (equivalent) open list and therefore maintain equivalent open lists throughout search. At each point IT-ICE(P^{IE}) will either select a $\bar{q} = \langle \boldsymbol{\varphi}, \hat{v}, \text{true} \rangle$—then IC also expanded a node for $\boldsymbol{\varphi}$—or a q. In the last case, because of property 2 of def. 1 we know that the same q is selected by IT-IC(P). □

Note that Theorem 1 does not mean that the runtime and space requirements of GMAA*-ICE and GMAA*-IC are identical: for each expansion, GMAA*-ICE will only generate one q to be stored on the open list versus a number of children that is, in the worst case, doubly exponential in the depth of the selected node.[3] On the other hand, GMAA*-ICE may select a placeholder for further expansion.

We say that a search algorithm is complete if it searches until it finds an optimal solution.

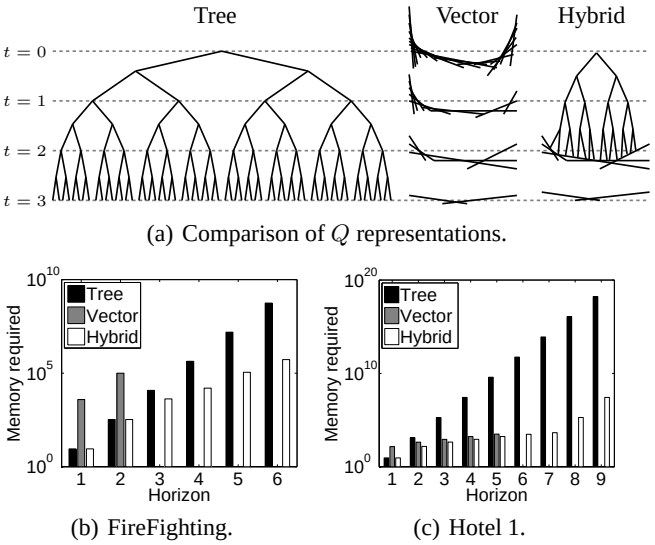
Corollary 1. *When using a heuristic of the form*

$$\widehat{Q}(\vec{\boldsymbol{\theta}}^t, \mathbf{a}) = \mathbf{E}[R(s, \mathbf{a}) \mid \vec{\boldsymbol{\theta}}^t] + \mathbf{E}[\widehat{V}(\vec{\boldsymbol{\theta}}^{t+1}) \mid \vec{\boldsymbol{\theta}}^t, \mathbf{a}], \qquad (5)$$

where $\widehat{V}(\vec{\boldsymbol{\theta}}^{t+1}) \geq Q_{\boldsymbol{\pi}^*}(\vec{\boldsymbol{\theta}}^{t+1}, \boldsymbol{\pi}^*(\vec{\boldsymbol{\theta}}^{t+1}))$ is an overestimation of the value of an optimal joint policy $\boldsymbol{\pi}^*$, GMAA*-ICE *is complete.*

Proof. Under the stated conditions, GMAA*-IC is complete [Oliehoek *et al.*, 2008; 2009]. Since GMAA*-ICE is search-equivalent to GMAA*-IC, it is also complete. □

[3] When a problem allows clustering, the number of child nodes grows less dramatically.



(a) Comparison of Q representations.

(b) FireFighting. (c) Hotel 1.

Figure 2: Hybrid heuristic representations. (a) Comparison of different representations. (b), (c) The number of real numbers stored for different representations of Q_{BG}.

5 Heuristic Representation

Previous research indicated that the upper bound provided by Q_{MDP} is often too loose for effective heuristic search [Oliehoek *et al.*, 2008]. However, for tighter heuristics such as Q_{POMDP} or Q_{BG} the space needed to store these heuristics grows exponentially with the horizon. There are two approaches to computing Q_{POMDP} or Q_{BG}. The first approach constructs a tree of all joint AOHs and their heuristic values, which is simple to implement, but requires storing a value for each $(\vec{\boldsymbol{\theta}}^t, \mathbf{a})$-pair (and their number grows exponentially with t, illustrated in Fig. 2(a)(left)). The second approach maintains a vector-based representation as is common for POMDPs (Fig. 2(a)(middle)). It also has exponential space complexity; even though pruning will provide leverage, in the worst case the number of maintained vectors grows exponentially with $h - t$, the number of stages-to-go.

In practice, we found that these space requirements become a bottleneck. To mitigate this problem we introduce a hybrid representation, as illustrated in Fig. 2(a)(right). The insight is that the exponential growth of the discussed representations is in opposite directions. Therefore, we can use the low-space-complexity side of both representations: the later stages use a vector-based representation (and later stages have fewer vectors), while the earlier stages use a history-based representation (and earlier stages have fewer histories). It is easy to compute a minimally-sized representation.

Figs. 2(b)-(c) illustrate the memory requirements for the "Tree", the "Vector", and the "Hybrid" representation for Q_{BG}, where missing "Vector" bars indicates those representations grew beyond limits. The vector-based Q_{BG} representation is computed using a variation of Incremental Pruning. The pruning performance depends on the problem and the complexity of the value function, which can increase suddenly, as for instance happens in Fig. 2(c). We see that the hybrid representation allows for very significant savings, allowing us to compute tight heuristics for longer horizons.

h	V^*	T_{IC}(s)	T_{ICE}(s)
Dec-Tiger			
2	-4.000000	≤ 0.01	≤ 0.01
3	5.190812	≤ 0.01	≤ 0.01
4	4.802755	$\S\, 0.27$	≤ 0.01
5	7.026451	$\dagger\, 21.03$	$\S \dagger\, 0.02$
6	$\mathbf{10.381625}$	$-$	46.43
7		$-$	$*$
FireFighting $\langle n_h = 3, n_f = 3 \rangle$			
2	-4.383496	≤ 0.01	≤ 0.01
3	-5.736969	$\S\, 0.11$	0.10
4	-6.578834	$\dagger\, 950.51$	1.00
5	$\mathbf{-7.069874}$	$-$	$\dagger\, 4.40$
6	$\mathbf{-7.175591}$	0.08	0.07
7		$\#$	$\#$
GridSmall			
2	0.910000	≤ 0.01	≤ 0.01
3	1.550444	$\S\, 0.10$	≤ 0.01
4	2.241577	$\dagger\, 1.77$	$\S \dagger \leq 0.01$
5	$\mathbf{2.970496}$	$-$	0.02
6	$\mathbf{3.717168}$	$-$	0.04
7		$\#$	$\#$

h	V^*	T_{IC}(s)	T_{ICE}(s)
Hotel 1			
2	10.000000	≤ 0.01	≤ 0.01
3	16.875000	≤ 0.01	≤ 0.01
4	22.187500	$\S \dagger \leq 0.01$	$\S \dagger \leq 0.01$
5	$\mathbf{27.187500}$	≤ 0.01	≤ 0.01
6	$\mathbf{32.187500}$	≤ 0.01	≤ 0.01
7	$\mathbf{37.187500}$	≤ 0.01	≤ 0.01
8	$\mathbf{42.187500}$	≤ 0.01	≤ 0.01
9	$\mathbf{47.187500}$	0.02	≤ 0.01
10		$\#$	$\#$
Recycling Robots			
5	16.486000	$\dagger \leq 0.01$	$\dagger \leq 0.01$
15	47.248521	$\S \leq 0.01$	≤ 0.01
18	$\mathbf{56.479290}$	≤ 0.01	$\S \leq 0.01$
20	$\mathbf{62.633136}$	≤ 0.01	≤ 0.01
30	$\mathbf{93.402367}$	0.08	0.05
40	$\mathbf{124.171598}$	0.42	0.25
50	$\mathbf{154.940828}$	2.02	1.27
60	$\mathbf{185.710059}$	9.70	6.00
70	$\mathbf{216.479290}$	$-$	28.66
80		$-$	$-$

h	V^*	T_{IC}(s)	T_{ICE}(s)
Cooperative Box Pushing			
2	17.600000	≤ 0.01	≤ 0.01
3	66.081000	$\S \dagger\, 0.11$	$\dagger \leq 0.01$
4	98.593613	$*$	$\S\, 313.07$
5		$\#$	$\#$
BroadcastChannel			
7	6.590000	$\dagger \leq 0.01$	$\dagger \leq 0.01$
10	9.290000	≤ 0.01	≤ 0.01
20	18.313228	≤ 0.01	≤ 0.01
25	22.881523	≤ 0.01	≤ 0.01
30	$\mathbf{27.421850}$	≤ 0.01	≤ 0.01
50	$\mathbf{45.501604}$	≤ 0.01	≤ 0.01
53	$\mathbf{48.226420}$	$\S \leq 0.01$	$\S \leq 0.01$
100	$\mathbf{90.760423}$	≤ 0.01	≤ 0.01
250	$\mathbf{226.500545}$	0.06	0.07
500	$\mathbf{452.738119}$	0.81	0.94
600	$\mathbf{543.228071}$	11.63	13.84
700	$\mathbf{633.724279}$	0.52	0.63
800		$-$	$-$
900	$\mathbf{814.709393}$	9.57	11.11
1000		$-$	$-$

Table 1: Experimental results comparing the computation times of GMAA*-IC (T_{IC}) and GMAA*-ICE (T_{ICE}), using the hybrid Q_{BG} representation. Bold entries highlight results for which no previous solution was known in literature. Legend: "$-$" are memory limit violations; "$*$" time limit overruns; "$\#$" memory or time overruns computing the heuristic; $\S$ indicates the maximum planning horizon when using the Q_{MDP} heuristic; and $\dagger$ shows the highest h using a tree-based Q_{BG} representation.

6 Experiments

We performed an empirical evaluation of GMAA*-ICE by comparing to GMAA*-IC, which is currently (one of) the fastest optimal solvers for finite-horizon Dec-POMDPs.[4] We tested on a suite of benchmark problems from the literature [Oliehoek *et al.*, 2009], using discount factor $\gamma = 1.0$.[5] We used Q_{BG} with a hybrid representation, and GMAA*-ICE uses BAGABAB [Oliehoek *et al.*, 2010] (with joint types ordered according to increasing probability). Experiments were run on an Intel iCore5 CPU running Linux, and we limited each search process to 2Gb of RAM and a maximum computation time of $3{,}600$s. Reported CPU-times are averaged over 10 independent runs and have a resolution of 0.01s. They concern only the MAA* search process, since computation of the heuristic can be amortized over multiple runs.[6]

The main results are listed in Table 1. It clearly shows that incremental expansion combined with the hybrid representation allows for significant improvements over the state of the art: for the vast majority of problems tested we provide results for longer horizons than any previously known (the bold entries). Thus, incorporating the hybrid representation into GMAA*-IC greatly increases its scalability, while adding the incremental expansion of GMAA*-ICE results in even more performance improvements. When comparing against GMAA*-IC, for Dec-Tiger we see that for $h = 5$ GMAA*-ICE achieves a speedup of 3 orders of magnitude, and it is also able to compute a solution for $h = 6$, unlike GMAA*-IC. For GridSmall we see a large speedup for $h = 4$ and very fast solutions for $h = 5, 6$, where GMAA*-IC runs out of memory. Similar positive results are obtained for Cooperative Box Pushing and FireFighting. An interesting counter-intuitive behavior can be observed for FireFighting, $h = 6$, which could be solved much faster than $h = 5$. Analysis reveals that the CBG instances encountered during the $h = 6$ search happen to cluster much better than the CBGs in the $h = 5$ search, which is possible because the heuristics vary with the horizon. Also, for BroadcastChannel we can see that the search is not necessarily monotonic in h.

Due to the hybrid representation we can compute Q_{BG} heuristics for all these problems and horizons, and as a consequence our results, *also for GMAA*-IC*, are much better. The $\S$ entries show the limits of running GMAA*-IC and GMAA*-ICE using Q_{MDP} instead of Q_{BG}: in most of these problems we can reach longer horizons with Q_{BG}. Furthermore, the $\dagger$ entries indicate the horizon to which we can solve a problem with a tree-based Q_{BG} representation.

The efficacy of a hybrid representation can be clearly seen for problems like GridSmall, Cooperative Box Pushing, FireFighting and Hotel 1 (for the latter two see Fig. 2(b) resp. 2(c)), where neither the tree nor the vector representation is able to provide a compact Q_{BG} heuristic for longer horizons. Apart from FireFighting, for these problems computing and storing Q_{BG} (or another tight heuristic) for longer horizons becomes the bottleneck for scaling further.

As a final note regarding Table 1, we see that only on the BroadcastChannel problem GMAA*-IC is (slightly) faster than GMAA*-ICE. Because this problem exhibits clustering

[4] The method by Amato *et al.* [2009] effectively focuses on state space reachability in problem structure.

[5] All problem definitions are available at `http://www.isr.ist.utl.pt/~mtjspaan/decpomdp`.

[6] The heuristics' computation time ranges from less than a second to many hours (for high h in some difficult problems).

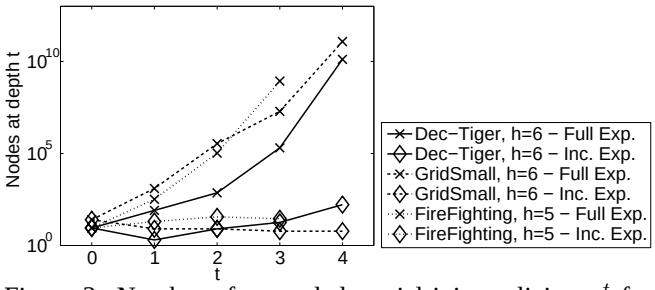


Figure 3: Number of expanded partial joint policies φ^t for intermediate stages $t = 0, \ldots, h-2$.

to a single joint type [Oliehoek *et al.*, 2009], the overhead of incremental expansion does not pay off (cf. footnote 3).

Summarizing our main results, we can conclude that 1) GMAA*-ICE outperforms GMAA*-IC leading to solutions of longer horizons in many problems, 2) both methods benefit from the improved heuristic representation, 3) in several problems computation and representation of the heuristic is the bottleneck that prevents from scaling further. The last point implies that our method may scale even further when the computation of the heuristic is further improved.

Finally, we have also investigated the impact of incremental expansion in terms of the number of nodes that are actually expanded for intermediate stages $t = 0, \ldots, h-2$. Fig. 3 shows the number of nodes expanded in GMAA*-ICE and the number that would be expanded for GMAA*-IC (which can be easily computed as they are search-equivalent). The plots confirm our initial hypothesis that in practice only a small number of child nodes are being queried.

7 Conclusions & Future work

Decentralized POMDPs offer a rich model for multiagent coordination under uncertainty. Optimal solution methods for Dec-POMDPs are of great interest; they are of practical value for smaller or decomposable problems and lie at the basis for most successful approximate methods [Emery-Montemerlo *et al.*, 2004; Seuken and Zilberstein, 2007; Wu *et al.*, 2011]. In this paper, we advance the state of the art by introducing an effective method for *incremental expansion* of nodes in the search tree. We proved that the resulting algorithm, GMAA*-ICE, is search-equivalent to GMAA*-IC and therefore complete. A new bottleneck, the amount of space needed for representation of the heuristic, was addressed by introducing a representation that is a hybrid between tree-based and vector-based representations.

We demonstrated our approach experimentally with and without incremental expansion, showing that its effect is complementary to clustering of histories. With just the new heuristic representation, optimal plans could be found for larger horizons than any known previous work for four benchmarks. In one case, horizons that are over an order of magnitude larger could be reached. By exploiting incremental expansion, GMAA*-ICE achieves further improvements in scalability. The combination of the hybrid representation and incremental expansion provides a powerful method for optimally solving Dec-POMDP over longer horizons.

Some possible extensions of this work are the following. First, we may consider improving the current CBG solver or try to adapt other CBG solvers, e.g., [Kumar and Zilberstein, 2010]. Second, incremental solvers for graphical CBGs may allow for further scaling of optimal solutions of Dec-POMDPs with multiple agents. Finally, future work should further consider improved heuristics and methods of computation, which can allow GMAA*-ICE to scale even further.

Acknowledgments

We would like to thank Anthony Cassandra for his pomdp-solve code (used for vector pruning), and the reviewers for their insightful suggestions. This work was funded in part by Fundação para a Ciência e a Tecnologia (ISR/IST pluriannual funding) through the PIDDAC Program funds and was supported by projects PTDC/EEA-ACR/73266/2006 and CMU-PT/SIA/0023/2009 (the latter under the Carnegie Mellon-Portugal Program). Research supported in part by AFOSR MURI project #FA9550-09-1-0538.

References

[Allen and Zilberstein, 2007] M. Allen and S. Zilberstein. Agent influence as a predictor of difficulty for decentralized problem-solving. In *AAAI*, 2007.

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

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

[Kumar and Zilberstein, 2010] A. Kumar and S. Zilberstein. Point-based backup for decentralized POMDPs: Complexity and new algorithms. In *AAMAS*, 2010.

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

[Oliehoek *et al.*, 2009] F. A. Oliehoek, S. Whiteson, and M. T. J. Spaan. Lossless clustering of histories in decentralized POMDPs. In *AAMAS*, 2009.

[Oliehoek *et al.*, 2010] F. A. Oliehoek, M. T. J. Spaan, J. Dibangoye, and C. Amato. Heuristic search for identical payoff Bayesian games. In *AAMAS*, 2010.

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

[Seuken and Zilberstein, 2007] S. Seuken and S. Zilberstein. Memory-bounded dynamic programming for DEC-POMDPs. In *IJCAI*, 2007.

[Seuken and Zilberstein, 2008] S. Seuken and S. Zilberstein. Formal models and algorithms for decentralized decision making under uncertainty. *JAAMAS*, 17(2):190–250, 2008.

[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 *Multi-agent Sequential Decision Making in Uncertain Domains*, 2011. Workshop at AAMAS11.

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

[Wu *et al.*, 2011] F. Wu, S. Zilberstein, and X. Chen. Online planning for multi-agent systems with bounded communication. *Artificial Intelligence*, 175(2):487–511, 2011.