Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence

# Information Gathering and Reward Exploitation of Subgoals for POMDPs

**Hang Ma** and **Joelle Pineau**
School of Computer Science
McGill University, Montreal, Canada

## Abstract

Planning in large partially observable Markov decision processes (POMDPs) is challenging especially when a long planning horizon is required. A few recent algorithms successfully tackle this case but at the expense of a weaker information-gathering capacity. In this paper, we propose *Information Gathering and Reward Exploitation of Subgoals* (IGRES), a randomized POMDP planning algorithm that leverages information in the state space to automatically generate "macro-actions" to tackle tasks with long planning horizons, while locally exploring the belief space to allow effective information gathering. Experimental results show that IGRES is an effective multi-purpose POMDP solver, providing state-of-the-art performance for both long horizon planning tasks and information-gathering tasks on benchmark domains. Additional experiments with an ecological adaptive management problem indicate that IGRES is a promising tool for POMDP planning in real-world settings.

## Introduction

Partially observable Markov decision processes (POMDPs) have emerged as a rich framework for planning under uncertainty due to their ability to capture a number of important planning aspects that appear in many real-world sequential decision tasks, such as the ability to handle stochastic actions, missing or noisy observations, and stochastic costs and rewards. The POMDP framework has been widely applied to various complex situations in practice thanks to its rich capabilities for planning in an uncertain environment (Cassandra 1998; Hsiao, Kaelbling, and Lozano-Pérez 2007; Pineau and Atrash 2007; Chades et al. 2012). Despite their mathematical expressivity, the difficulty of solving large POMDPs has limited their application in complex domains. For a POMDP problem modelled with $n$ states, we must reason in an $(n-1)$-dimensional continuous belief space. Moreover, the complexity of POMDP planning is affected by the length of planning horizon (Littman 1996). Practically speaking, solving real-world tasks in this framework involves two challenges: how to carry out intelligent information gathering in a large high dimensional belief space; and how to scale up planning with long sequences of actions and delayed rewards.

Recent point-based planning algorithms have made impressive improvements by addressing one or the other of the above challenges: PBVI (Pineau, Gordon, and Thrun 2003), HSVI2 (Smith and Simmons 2004; 2005), FSVI (Shani, Brafman, and Shimony 2007) and SARSOP (Kurniawati, Hsu, and Lee 2008) work very successfully in problems that require gathering information with a large state space, using bias expansion of belief sampling to yield good approximations of value functions; MiGS (Kurniawati et al. 2011) and some online solvers, e.g. PUMA (He, Brunskill, and Roy 2010), facilitate planning for problems that require long sequences of actions to reach a goal, by generating macro-actions and restricting the policy space. Other approaches, including a dynamic programming adaption to POMDPs called RTDP-Bel (Bonet and Geffner 2009) and a UCT-based online method called POMCP (Silver and Veness 2010) have demonstrated strong performance on some of the POMDP domains. However, we still lack methods that can tackle problems simultaneously requiring substantial information gathering and long planning horizons.

Here we propose a point-based POMDP solution method, called *Information Gathering and Reward Exploitation of Subgoals* (IGRES), that effectively tackles both challenges by incorporating elements from the two families of approaches: first, IGRES identifies potentially important states as subgoals; second, IGRES only gathers information and exploits rewards with macro-actions in the neighbourhood of those subgoals. Thus, IGRES is efficient in terms of computational time and space in the sense that it covers the belief space well with a much smaller size of belief points set for expensive backup operations while still maintaining good performance. Promising experimental results show that IGRES provides state-of-the-art performance on tasks that require significant information gathering and planning with long sequences of actions. We also show that IGRES can effectively tackle a new class of ecological adaptive management problems presented in the IJCAI 2013 data challenge track, thus providing evidence that IGRES is capable of solving tasks in useful real-world settings.

## Technical Background

A partially observable Markov decision process (POMDP) is a tuple $\langle \mathcal{S}, \mathcal{A}, \Omega, T, O, R, b_0, \gamma \rangle$ (Kaelbling, Littman, and Cassandra 1998), where $\mathcal{S}$ is a finite set of states, $\mathcal{A}$ is a finite set of actions, $\Omega$ is a finite set of observations, $T(s,a,s') = p(s'|s,a)$ is the transition function that maps each state and action to a probability distribution over states, $O(s',a,o) = p(o|s',a)$ is the observation function that maps a state and an action to a probability distribution over possible observations, $R(s,a)$ is the reward function, $b_0(s)$ is the initial belief state, and $\gamma$ is the discount factor.

A belief state $b \in \mathcal{B}$ is a sufficient statistic for the history, and is updated after taking action $a$ and receiving observation $o$ as follows:

$$b^{a,o}(s') = \frac{O(s',a,o)\sum_{s\in\mathcal{S}} T(s,a,s')b(s)}{p(o|a,b)}, \tag{1}$$

where $p(o|a,b) = \sum_{s\in\mathcal{S}} b(s) \sum_{s'\in\mathcal{S}} T(s,a,s')O(s',a,o)$ is a normalizing factor.

A policy is a mapping from the current belief state to an action. A value function $\mathcal{V}_\pi(b)$ specifies the expected reward gained starting from $b$ followed by policy $\pi$:

$$\mathcal{V}_\pi(b) = \sum_{s\in\mathcal{S}} b(s)R(s,\pi(b)) + \gamma \sum_{o\in\Omega} p(o|b,\pi(b))\mathcal{V}_\pi(b^{\pi(b),o}).$$

The goal of POMDP planning is to find an optimal policy that maximizes its value function denoted by $\mathcal{V}^*$. In general, $\mathcal{V}$ can be approximated arbitrarily closely by a piecewise-linear and convex function (Sondik 1978): $\mathcal{V}(b) = \max_{\alpha\in\Gamma}(b(s)\alpha(s))$, represented by a finite set of hyper-planes, $\Gamma$.

**Point-based Solvers** Information gathering in large state spaces and planning with long sequences of actions are two major challenges for planning in large POMDPs. Point-based approximations are among the most successful approaches to approximate the value function in large POMDPs. Solutions are computed by applying iterative value function backups over a small set of belief points (see Algorithm 1). To reduce computation, the choice of the candidate belief points at which to apply backups becomes crucial for this class of algorithms. PBVI (Pineau, Gordon, and Thrun 2003), one of the first point-based algorithm, samples only belief points in a reachable space $R(b_0)$ from an initial belief point $b_0$ as the representative set for backup operations rather than sampling from the high dimensional continuous $\mathcal{B}$. HSVI2 keeps lower and upper bounds on the value function and use heuristics to sample belief states that reduce the gap between bounds. FSVI (Shani, Brafman, and Shimony 2007) uses only upper bound to guide the belief point sampling, and works well for domains that require only simple information gathering actions. SARSOP (Kurniawati, Hsu, and Lee 2008) focuses sampling around the optimal reachable space, $R^*(b_0) \subseteq R(b_0)$, sampled from optimal sequences of actions. MiGS (Kurniawati et al. 2011) adopts the notion of generating macro-actions, which are sequences of actions, rather than single actions to expand the belief point set, which is one of the successful strategies to tackle problems with long planning horizons and delayed rewards.

**Algorithm 1** Backup($\Gamma$,$b$)

```
$\alpha_{a,o} \leftarrow \text{argmax}_{\alpha\in\Gamma} \sum_{s\in\mathcal{S}} \alpha(s) b^{a,o}(s), \forall a \in \mathcal{A}, o \in \Omega$;
$\alpha_a(s) \leftarrow R(s,a) + \gamma \sum_{o,s'} T(s,a,s')O(s',a,o)\alpha_{a,o}(s')$, $\forall a \in \mathcal{A}, s \in \mathcal{S}$;
$\alpha' \leftarrow \text{argmax}_{a\in\mathcal{A}} \sum_{s\in\mathcal{S}} \alpha'(s)b(s)$;
$\Gamma \leftarrow \Gamma \cup \{\alpha'\}$.
```

## Gathering Information & Exploiting Rewards

Our work attempts to bridge the gap between the two classes of point-based approaches, to produce a single POMDP solver with sufficient versatility to address both problems that require substantial information gathering actions, and problems that require long sequences of actions. To achieve this, IGRES tries to identify potentially important states as subgoals using a strategy similar to MiGS (Kurniawati et al. 2011). Then for each belief associated with a subgoal, IGRES generates another macro-action to gather information and exploit rewards.

### Capturing Important States

In general, actions that yield high rewards or gather significant information from the current belief state play an important role in constructing good policies (Cassandra, Kaelbling, and Kurien 1996; He, Brunskill, and Roy 2010). This suggests that states associated with high rewards or informative observations may also be important. Our algorithm leverages this structure by attempting to identify these potentially important states and use them as subgoals.

Specifically, we define two heuristic functions $h_r(s)$ and $h_i(s)$ of a state $s \in \mathcal{S}$, to describe its importance in terms of rewards exploitation and information gathering respectively. We calculate the importance of rewards exploitation for some state $s$:

$$h_r(s) = \max_{a\in\mathcal{A}} \frac{R(s,a) - R_{min}}{R_{max} - R_{min}}, \tag{2}$$

which captures the highest immediate reward we can get from state $s$ over all actions. We also calculate its information gain as:

$$h_i(s) = \max_{a\in\mathcal{A}} \sum_{o\in\Omega} \left( -\frac{1}{|\Omega|}\log(\frac{1}{|\Omega|}) + O(s,a,o)\log\left(O(s,a,o)\right) \right). \tag{3}$$

which measures for some $s$ over all actions the highest possible entropy of observation probabilities against a uniform distribution.

Then a state $s \in \mathcal{S}$ is sampled as subgoal using the softmax function:

$$p(s) \propto e^{\eta(\frac{h_r(s)}{\sum h_r(s)} + \lambda \frac{h_i(s)}{\sum h_i(s)})}, \tag{4}$$

where the pre-defined positive constant $\eta$ serves as a normalizer and a controller for the smoothness of our random sampling, and $\lambda$ balances between the two heuristic functions.

## Leveraging State Structure

In this section, we follow the algorithm of MiGS to exploit structure in the state space, that helps focus sampling in the belief space. We first consider the state graph $G_{\mathcal{S}}$. We associate with each edge $(\overline{ss'}, a)$ a weight that measures the difference between the expected total reward of state $s$ and destination $s'$ via action $a$: $w(\overline{ss'}, a) = \alpha(s) - \alpha(s') = R(s,a) + \gamma \sum_{s'' \in \mathcal{S}/\{s'\}} T(s,a,s') \times \sum_{o \in \Omega} O(s'', a, o)(\alpha(s'') - \alpha(s'))$, where $\alpha(s) = R(s,a) + \gamma \sum_{o,s'} T(s,a,s')O(s',a,o)\alpha(s')$ approximates the expected total reward. Since $s''$ can be reached from $s$ with one single action, we expect the difference between $\alpha(s'')$ and $\alpha(s)$ is small. By replacing $\alpha(s'')$ with $\alpha(s)$, we get:

$$w(\overline{ss'}, a) = \frac{-R(s,a) \cdot \mathbb{1}_{R(s,a) \leq 0}}{1 - \gamma + \gamma T(s,a,s')}, \tag{5}$$

which captures the cost of selecting action $a$ transitioning from $s$ to $s'$. Then we define the distance as $d_{\mathcal{S}}(s,s') = \min_{a \in \mathcal{A}: T(s,a,s') > 0} w(\overline{ss'}, a)$. We further extend the notion of distance $d_{\mathcal{S}}$ such that distance from $s$ to $s'$ is just the shortest path by the distance measure above. Now we have reduced the directed multigraph $G_{\mathcal{S}}$ to a weighted directed graph. If we assume strong connectivity of the state graph, an inward Voronoi partitioning (Erwig 2000) can then be used to partition the states based on $d_{\mathcal{S}}$ into a partitioning

$$\mathcal{K} = \{K_m | m \in \mathcal{M}\}, \tag{6}$$

where $\mathcal{M} \subseteq \mathcal{S}$ is a set of subgoals and $K_m = \{s \in \mathcal{S} | d_{\mathcal{S}}(s,m) < d_{\mathcal{S}}(s,m'), \forall m' \neq m \text{ and } m, m' \in \mathcal{M}\}$ is the set of states whose distance to subgoal $m$ is less than the distance to any other subgoal in $\mathcal{M}$.

We build a roadmap graph $G_{\mathcal{M}}$ where each node corresponds to a sampled subgoal. Edge $\overline{mm'}$ from subgoal $m$ to $m'$ is present if there exists a path from $m$ to $m'$ in graph $G_{K_m \cup K_{m'}}$. And this edge is labeled with a sequence of actions and a sequence of states according to the path. Then we know the edge $\overline{mm'}$ is also associated with the weight of the shortest path. In this way, we partition the state space into regions where the states in each region lead towards a subgoal inducing that region, and the connectivity between regions is also well structured by the edges of $G_{\mathcal{M}}$.

## Sampling Belief States Using Macro-actions

Now we give a description of the belief sampling strategy. Specifically, given an estimate of current state $s$, a new belief state $b'$ is sampled from the current belief state $b$ according to a macro-action $(a_1, a_2, \ldots, a_l)$ and a state sequence $(s_0, s_1, s_2, \ldots, s_l)$ where $s_0 = s$. To achieve this, we first generate an observation sequence $(o_1, o_2, \ldots, o_l)$ where $o_i$ is sampled with probability

$$p(o_i) \propto O(s_i, a_i, o_i) = p(o_i | s_i, a_i). \tag{7}$$

Using the updating rule of belief states (1), we immediately get a sequence of belief states $(b_0, b_1, b_2, \ldots, b_l)$ such that

$$b_1 = b_0^{a_1, o_1}; \quad b_i = b_{i-1}^{a_{i-1}, o_{i-1}}, \text{for } 2 \leq i \leq l, \tag{8}$$

where $b_0 = b$ and $b_l = b'$. As the core of the algorithm, two types of macro-actions, subgoal-oriented macro-actions and exploitation macro-actions, are generated, which split our planning strategy into subgoal transitioning phase and exploitation phase respectively.

**Generating Subgoal-Oriented Macro-actions** With the help of roadmap $G_{\mathcal{M}}$, we first extract a path transitioning to the closest $m \in \mathcal{M}$ from current state $s$. To do so, we store a list of all outgoing edges for each subgoal according to $G_{\mathcal{M}}$. If $s$ is not a subgoal, the path will be the only path from $s$ to the subgoal in the same partition; otherwise $s$ is a subgoal and the path is just the next outgoing edge in the list. From this path, we obtain a sequence of actions as our subgoal oriented macro-action and a sequence of states that are visited along the path. According to the belief sampling strategy (Eqn 7,8), we get a new belief state $b'$ associated with $m$. Since the subgoal $m \in \mathcal{M}$ is potentially important, next we introduce an exploitation macro-action which is restricted to the neighbourhood of $m$.

**Generating Exploitation Macro-actions** Given an estimate of current state $s$ at each iteration, we sample an action $a$ with probability $p(a)$ proportional to some heuristic function $\tilde{a}(s)$. Many heuristics can be adopted here, such as choosing the action that maximizes long term reward, or just uniformly at random. We define $\tilde{a}(s) = e^{\mu T(s,a,s)}$ to favor the action that could gather information explicitly without changing the estimate of state, which works well empirically. And we update the estimate of state as $s'$ with probability $p(s') \propto T(s,a,s')$. We stop exploiting with probability $(1 - p_{ex})$ at each iteration, where $p_{ex}$ is an exploitation probability parameter to control the length of this macro-action. At the end of the exploitation, we add an action that maximizes the reward from the current estimate of state if necessary, which is consistent with the heuristic function (Eqn 2) for choosing subgoals.

## Data Structures

IGRES maintains a belief tree $T_{\mathcal{R}}$ rooted at the initial belief node $u_{b_0}$ which consists of all belief states that are sampled. It also stores all belief states that are associated with subgoals in a set $\mathcal{R}_\delta$. Similar to MiGS, at each iteration, the algorithm chooses a belief state in the tree with probability proportional to $\frac{1}{|K_c(b)|}$, where the $c$-neighbourhood $K_c(b)$ is defined as: $K_c(b) = \{\hat{b} \in \mathcal{R} | d_{\mathcal{K}}(b, \hat{b}) \leq c\}$, to favor belief states that have fewer neighbours. A similar belief expansion strategy is adopted by PBVI, except that instead of $L_1$ distance, the algorithm uses the relaxed $\mathcal{K}$-distance $d_{\mathcal{K}}$ defined based on $\mathcal{K}$-partitioning (Eqn 6):

$$d_{\mathcal{K}}(b, b') = \sum_{m \in \mathcal{M}} \left| \sum_{s \in K(m)} b(s) - \sum_{s \in K(m)} b'(s) \right|, \tag{9}$$

which measures the differences of probability for partitions rather than single states. We also store the estimate of current state in each node. Then a new belief state is generated by the subgoal oriented macro-action and the estimate of current state is updated to be the subgoal state. To control the total number of belief states inserted in the tree, we calculate the $\mathcal{K}$-distance from the new belief state to the belief set $\mathcal{R}_\delta$

to decide whether we will exploit this subgoal or not. If we decide to exploit, we continue to do exploitation. The new belief states are inserted in the tree, and an $\alpha$-backup is done for each of the belief states along the path back to the root. The corresponding estimate of current state is also stored in each belief node so that if we choose to sample new beliefs from that node later, we could start immediately from the estimate of current state. This strategy enables sampling deeper belief points without maintaining a belief tree that is too large. After the value of the initial belief state is not improved for some number of rounds, new states are sampled and added to $\mathcal{M}$, and a new $G_{\mathcal{M}}$ is constructed accordingly.

**Algorithm 2** IGRES

```
Input: $b_0, G_{\mathcal{M}}, \delta$
Output: $\mathcal{V}(b_0)$
  $T_{\mathcal{R}} \leftarrow \{u_{b_0}\}, R_\delta \leftarrow \{b_0\}$;
  while number of rounds $\mathcal{V}(b_0)$ not improving $<$ limit do
      Sample node $u_b \in T_{\mathcal{R}}$ probability $\propto \frac{1}{|K_c(b)|}$;
      if $b = b_0$ then
          Sample $u_b.state$ according to $b(s)$;
      end if
      Current estimate of state $s \leftarrow u_b.state$;
      Generate subgoal oriented macro-action for $b$ with state estimate $s$, which gives updated belief $b'$ and $s'$;
      if $\min_{\hat{b} \in \mathcal{R}_\delta} d_{\mathcal{K}}(b', \hat{b}) > \delta$ then
          BackUpAtLeaf($b'$,$s'$,$T_{\mathcal{R}}$,$u_b$);          ▷ Algorithm 3.
          $R_\delta \leftarrow R_\delta \cup \{b'\}$;
          Generate exploitation macro-action for $b'$ with state estimate $s'$, which gives updated belief $b''$ and $s''$;
          BackupAtLeaf($b''$,$s''$,$T_{\mathcal{R}}$,$u_{b'}$);          ▷ Algorithm 3.
      end if
  end while
```

**Algorithm 3** BackupAtLeaf($b$,$s$,$T_{\mathcal{R}}$,$u^{parent}$)

```
  Create a node $u_b$;
  $u_b.state \leftarrow s$;
  Insert $u_b$ to $T_{\mathcal{R}}$ as a child of node $u^{parent}$;
  Backup for each belief inducing each node along the path from $u_b$ back to root $u_{b_0}$;          ▷ Algorithm 1.
```

**Analysis of $\mathcal{K}$-Distance Threshold** The $\mathcal{K}$-distance threshold $\delta$ is used to control the number of belief states associated with subgoals for backup operations.

**Theorem 1.** *If $d_{\mathcal{K}}(b, b') \leq \delta$, then $|\mathcal{V}^*(b) - \mathcal{V}^*(b')| \leq \frac{1}{1-\gamma}(\delta R_{max} - 2 l_{max} R_{min}), \forall b, b' \in \mathcal{B}$,*

where $l_{max}$ is the maximal number of states traversed along the path from any state to the subgoal in the same partition, which is less than the number of states in that partition. This theorem is derived from Theorem 1 in (Kurniawati et al. 2011) by relating the error to the size of each partition, which implies that $\mathcal{V}^*(b')$ can be approximated by $\mathcal{V}^*(b)$ with a small error controlled by the $\mathcal{K}$-distance threshold $\delta$ and an extra error depends on the number of states in each partition (details in tech report).

**Complexity Reduced by Macro-actions** Consider constructing a belief tree $T_R$ for a set of belief points $R$ sampled from the reachable space $R(b_0)$, then we have $\Theta((|\mathcal{A}||\Omega|)^h)$ beliefs for a planning horizon $h$. Let $l$ be the minimum length of a macro-action generated by IGRES, then the size of $T_R$ is reduced to $\mathcal{O}((d|\Omega|^l)^{\frac{h}{l}} = \mathcal{O}(d^{\frac{h}{l}}|\Omega|^h)$. The maximum branching number $d$ is the maximum degree of all subgoals in $G_{\mathcal{M}}$.

**Completeness of Planning** In the case where $\delta = 0$, IGRES inserts every sampled belief state. Then consider the fact that $G_{\mathcal{M}}$ is updated by adding without replacement new subgoals sampled from a finite $\mathcal{S}$. We can show that eventually, all states would be sampled as subgoals. As new edges are inserted to $G_{\mathcal{M}}$, all possible edges between each pair of states would finally be included in $G_{\mathcal{M}}$. For $p_{ex} > 0$, self-loop action sequences of any length could also be generated. Thus IGRES is planning in a complete policy space. Local exploration ensures completeness over the belief space.

# Experiments

We first consider classic POMDP benchmark problems of various sizes and types, and then present results for a real-world challenge domain.

## Benchmark Problems

POMDP as a model for planning differs largely from MDP due to the uncertainty of state information. Thus algorithms that sample belief states by using pure state information to generate macro-actions (e.g. MiGS) might not work well on some problems because they fail to take advantages of observation information to identify subgoals and sample more focused beliefs. On the other hand, algorithms that adapt certain heuristics to gather information (e.g. SARSOP) might improve very slowly when they maintain a large set of sampled beliefs or they might even get stuck locally on some problems because they fail to sample deeper belief points. IGRES constructs macro-actions considering both informative observations and rewarding actions, and gathers information and exploits subgoals in a probabilistic manner without making assumptions about the domain. Thus, IGRES has the potential to provide good solutions to problems which require either information gathering to compute good policies or long sequences of actions to exploit delayed rewards. To demonstrate this, we carried out experiments on both types of benchmark problems.

Tiger (Cassandra, Kaelbling, and Littman 1994) is the most classic POMDP problem. Noisy-tiger (He, Brunskill, and Roy 2010) is a modified version of Tiger problem where the noise of the *listen* action is increased to make the information gathering more difficult. RockSample (Smith and Simmons 2004) is another classic information gathering problem. Hallway2 introduced in (Littman, Cassandra, and Kaelbling 1995) is a robot navigation problem in a grid map of maze. Tag is a domain introduced in (Pineau, Gordon, and Thrun 2003) where a robot searches for a moving opponent. Underwater Navigation (Kurniawati, Hsu, and Lee 2008) models an autonomous underwater vehicle (AUV) navigating in a grid map. Homecare (Kurniawati, Hsu, and Lee 2008) is a domain similar to Tag but with much more complex dynamic, which models a robot tracking an elderly person. 3D-Navigation (Kurniawati et al. 2011) models an unmanned aerial vehicle (UAV) navigating in an 3D-indoor environment, which requires the vehicle localizing at specific landmarks to avoid dangerous areas before heading for the goal states.

We compared performance of IGRES to existing POMDP solvers RTDP-Bel (Bonet and Geffner 2009), HSVI2 (Smith and Simmons 2005), FSVI (Shani, Brafman, and Shimony 2007), SARSOP (Kurniawati, Hsu, and Lee 2008) and MiGS (Kurniawati et al. 2011) on all benchmark domains introduced above: RTDP-Bel is an adaptation of real time dynamic programming to POMDPs, and has demonstrated strong performance on several POMDP domains; and the rest are four point-based algorithms aiming at various POMDP domains respectively which we have mentioned in the second section of this paper. We performed these experiments on a computer with a 2.50GHz Intel Core i5-2450M processor and 6GB of memory. We ran the probabilistic solvers, MiGS and IGRES, 30 times each to compute policies on Homecare and 3D-Navigation, and 100 times each for the other domains. We ran each of the four other algorithms once for each domain until convergence or achieving good level of estimated values. Then we ran sufficient number of simulations to evaluate each policy computed by these four algorithms. The average reward with the 95% confidence intervals and the corresponding computation times are reported in Table 1.

The number of subgoals for IGRES is randomly picked roughly according to the size of each domain. We will address the effect of varying the size of subgoals in the later set of experiments. For the Tiger domains, we observe that while RTDP-Bel, HSVI2, SARSOP and IGRES show good performance, MiGS fails even though both problems are extremely simple. The reason is that MiGS fails to produce policies that take actions changing the probability distribution within state dimensions. MiGS performs poorly on RockSample for the similar reasons. We notice that for both RockSample problems there exist small gaps between the results of IGRES and the best solutions in the same amount of time. This class of problems only require performing single-step information-gathering actions at specific locations with fully observable movement actions, which the algorithms adapting certain deterministic heuristics might be able to take advantage of. On the other hand, IGRES generates macro-actions in a probabilistic manner. So the parameters of generating macro-actions should be tuned to match this pattern in order to reach the optimal performance. For example on RockSample(4,4) with 20 subgoals, IGRES returns 17.91 ± 0.12 in 10 seconds of computation time. For the Hallway2 domain, SARSOP and IGRES achieve best solutions due to their strong ability to gather useful information. For the Tag problem, RTDP-Bel and HSVI2 are not able to compute a good solution, while the other four algorithms achieve high rewards in short computation time. For Underwater Navigation task, IGRES is substantially faster than RTDP-Bel to compute the best solution. For the Homecare problem, IGRES is able to achieve the best rewards given same amount of computation time. For 3D-Navigation task, RTDP-Bel, HSVI2 and SARSOP are unable to achieve

| | Return | Time(s) |
|---|---:|---:|
| **Tiger** | | |
| $\lvert\mathcal{S}\rvert = 2, \lvert\mathcal{A}\rvert = 3, \lvert\Omega\rvert = 2$ | | |
| **RTDP-Bel** | $\mathbf{19.42 \pm 0.59}$ | 0.30 |
| **HSVI2** | $\mathbf{19.31 \pm 0.09}$ | <1 |
| FSVI* | N/A | |
| **SARSOP** | $\mathbf{18.59 \pm 0.61}$ | 0.09 |
| MiGS | $-19.88 \pm 0$ | 100 |
| **IGRES (# subgoals: 1)** | $\mathbf{19.41 \pm 0.59}$ | 1 |
| **Noisy-tiger** | | |
| $\lvert\mathcal{S}\rvert = 2, \lvert\mathcal{A}\rvert = 3, \lvert\Omega\rvert = 2$ | | |
| **RTDP-Bel** | $\mathbf{-13.67 \pm 0.28}$ | 1.22 |
| **HSVI2** | $\mathbf{-13.69 \pm 0.04}$ | <1 |
| FSVI* | N/A | |
| **SARSOP** | $\mathbf{-13.66 \pm 0.18}$ | 0.18 |
| MiGS | $-19.88 \pm 0$ | 100 |
| **IGRES (# subgoals: 1)** | $\mathbf{-13.67 \pm 0.18}$ | 1 |
| **RockSample(4,4)** | | |
| $\lvert\mathcal{S}\rvert = 257, \lvert\mathcal{A}\rvert = 9, \lvert\Omega\rvert = 2$ | | |
| **RTDP-Bel** | $\mathbf{17.94 \pm 0.12}$ | 10.7 |
| **HSVI2** | $\mathbf{17.92 \pm 0.01}$ | <1 |
| **FSVI** | $\mathbf{17.85 \pm 0.18}$ | 1 |
| SARSOP | $17.75 \pm 0.12$ | 0.7 |
| MiGS | $8.57 \pm 0$ | 100 |
| IGRES (# subgoals: 4) | $17.30 \pm 0.12$ | 10 |
| **RockSample(7,8)** | | |
| $\lvert\mathcal{S}\rvert = 12545, \lvert\mathcal{A}\rvert = 13, \lvert\Omega\rvert = 2$ | | |
| RTDP-Bel | $20.55 \pm 0.13$ | 103 |
| **HSVI2** | $\mathbf{21.09 \pm 0.10}$ | 100 |
| FSVI | $20.08 \pm 0.20$ | 102 |
| **SARSOP** | $\mathbf{21.35 \pm 0.13}$ | 100 |
| MiGS | $7.35 \pm 0$ | 100 |
| IGRES (# subgoals: 8) | $19.54 \pm 0.12$ | 100 |
| **Hallway2** | | |
| $\lvert\mathcal{S}\rvert = 92, \lvert\mathcal{A}\rvert = 5, \lvert\Omega\rvert = 17$ | | |
| RTDP-Bel | $0.237 \pm 0.006$ | 1004 |
| HSVI2 | $0.507 \pm 0.001$ | 250 |
| FSVI | $0.494 \pm 0.007$ | 280 |
| **SARSOP** | $\mathbf{0.530 \pm 0.008}$ | 200 |
| **MiGs** | $\mathbf{0.522 \pm 0.008}$ | 200 |
| **IGRES (# subgoals: 20)** | $\mathbf{0.530 \pm 0.008}$ | 200 |
| **Tag** | | |
| $\lvert\mathcal{S}\rvert = 870, \lvert\mathcal{A}\rvert = 5, \lvert\Omega\rvert = 30$ | | |
| RTDP-Bel | $-6.32 \pm 0.12$ | 372 |
| HSVI2 | $-6.46 \pm 0.09$ | 400 |
| **FSVI** | $\mathbf{-6.11 \pm 0.11}$ | 35 |
| **SARSOP** | $\mathbf{-6.08 \pm 0.12}$ | 30 |
| **MiGS** | $\mathbf{-6.00 \pm 0.12}$ | 30 |
| **IGRES (# subgoals: 20)** | $\mathbf{-6.12 \pm 0.12}$ | 30 |
| **Underwater Navigation** | | |
| $\lvert\mathcal{S}\rvert = 2653, \lvert\mathcal{A}\rvert = 6, \lvert\Omega\rvert = 103$ | | |
| **RTDP-Bel** | $\mathbf{750.07 \pm 0.28}$ | 338 |
| HSVI2 | $718.37 \pm 0.60$ | 400 |
| FSVI | $725.88 \pm 5.91$ | 414 |
| SARSOP | $731.33 \pm 1.14$ | 150 |
| MiGS | $715.50 \pm 1.37$ | 400 |
| **IGRES (# subgoals: 20)** | $\mathbf{749.94 \pm 0.30}$ | 50 |
| **Homecare** | | |
| $\lvert\mathcal{S}\rvert = 5408, \lvert\mathcal{A}\rvert = 9, \lvert\Omega\rvert = 928$ | | |
| RTDP-Bel** | N/A | |
| HSVI2 | $15.07 \pm 0.37$ | 2000 |
| FSVI*** | N/A | |
| **SARSOP** | $\mathbf{16.64 \pm 0.82}$ | 1000 |
| **MiGS** | $\mathbf{16.70 \pm 0.85}$ | 1600 |
| **IGRES (# subgoals: 30)** | $\mathbf{17.32 \pm 0.85}$ | 1000 |
| **3D-Navigation** | | |
| $\lvert\mathcal{S}\rvert = 16969, \lvert\mathcal{A}\rvert = 5, \lvert\Omega\rvert = 14$ | | |
| RTDP-Bel | $-93.03 \pm 0.01$ | 2115 |
| HSVI2 | $-91.98 \pm 0$ | 2000 |
| FSVI** | N/A | |
| SARSOP | $-99.97 \pm 0$ | 800 |
| **MiGS** | $\mathbf{(2.977 \pm 0.512) \times 10^4}$ | 150 |
| **IGRES (# subgoals: 163)** | $\mathbf{(3.272 \pm 0.193) \times 10^4}$ | 150 |

[1] For RTDP-Bel, FSVI and MiGS, we use the software packages provided by the authors of the papers. For HSVI2, we use the latest ZMDP version 1.1.7 (http://longhorizon.org/trey/zmdp/). For SARSOP, we use the latest APPL version 0.95 (http://bigbird.comp.nus.edu.sg/pmwiki/farm/appl/index.php?n=Main.Download).

* ArrayIndexOutOfBoundsException is thrown.

** Solver is not able to compute a solution given large amount of computation time.

*** OutOfMemoryError is thrown.

Table 1: Results of benchmark problems.[1]

a good solution because they fail to sample belief points that are far away from the existing set, and thus get trapped in the local area, whereas MiGS and IGRES easily overcome the effect of long planning horizon in this task. We conclude from these results that IGRES is able to successfully tackle both problems requiring information gathering, and problems with long planning horizons.

## The Ecological Adaptive Management Problem

Next, we apply IGRES to a class of ecological adaptive management problems (Nicol et al. 2013) that was presented as an IJCAI 2013 data challenge problem to the POMDP community. To the best of our knowledge, there has not been any published work on this challenge so far. The POMDPs in this domain represent networks for migratory routes by different shorebird species utilizing the East Asian-Australasian flyway, under uncertainty about the rate of sea level rise and its effect on shorebird populations. The goal is to select one node to perform protection action against a fixed amount of sea level rise in a weighted directed graph representing the flyway. The state space is a factored representation by the cross product of one fully observable population variable, the fully observable protection variable for each node in the graph, and one variable for sea level rise which is not observable. Five different bird species are considered (characterized by different POMDP sizes and parameterizations).

To solve this problem, we ran IGRES 30 times for each task to compute policies, and then ran 100 simulations to test each computed policy. Our results are generated on a 2.67GHz Intel Xeon W3520 computer with 8GB of memory. We also present results of the benchmark solutions computed by the original authors of the dataset using symbolic Perseus (Poupart 2005) on a more powerful computer (Nicol et al. 2013).

We observe in Table 2 that IGRES outperforms the benchmark solutions by achieving higher rewards for all species. Even though IGRES is not specially designed for solving the problem and does not directly exploit the factored state structure, it computes good solutions that yield high rewards in reasonable computation times for all these tasks.

We further experiment with IGRES on one of the tasks, varying parameters to demonstrate its behaviour as the number of subgoals changes. Figure 1 shows the behaviour of IGRES with different number of subgoals used given increasing computation time on Grey-tailed Tattler task. Given reasonable number of subgoals, IGRES is able to take advantage of subgoals that help sampling useful belief points. However, if the number of subgoals gets too large, a much larger set of beliefs would be sampled before the algorithm plans deep enough to compute a good solution, thus degrading the performance. On the other hand, if the number of subgoals is too small, some important states might be omitted from the set of subgoals, which also degrades the performance. Recall that the baseline performance for symbolic Perseus (as shown in Table 1) with 378 seconds of planning yielded a return of just 4520, well under the results shown in Figure 1. Thus, the performance indicates that IGRES is robust against changes in the number of subgoals for this type of problems and consistently samples important subgoal states to compute good solutions.

| | Return | Time(s) |
|---|---:|---:|
| **Lesser sand plover** | | |
| $\lvert\mathcal{S}\rvert = 108, \lvert\mathcal{A}\rvert = 3, \lvert\Omega\rvert = 36$ | | |
| symbolic Perseus* | 4675 | 10 |
| IGRES (# subgoals: 18) | $5037.72 \pm 8.82$ | 10 |
| **Bar-tailed godwit b.** | | |
| $\lvert\mathcal{S}\rvert = 972, \lvert\mathcal{A}\rvert = 5, \lvert\Omega\rvert = 324$ | | |
| symbolic Perseus* | 18217 | 48 |
| IGRES (# subgoals: 36) | $19572.41 \pm 39.35$ | 60 |
| **Terek sandpiper** | | |
| $\lvert\mathcal{S}\rvert = 2916, \lvert\mathcal{A}\rvert = 6, \lvert\Omega\rvert = 972$ | | |
| symbolic Perseus* | 7263 | 48 |
| IGRES (# subgoals: 72) | $7867.95 \pm 2.44$ | 60 |
| **Bar-tailed godwit m.** | | |
| $\lvert\mathcal{S}\rvert = 2916, \lvert\mathcal{A}\rvert = 6, \lvert\Omega\rvert = 972$ | | |
| symbolic Perseus* | 24583 | 58 |
| IGRES (# subgoals: 72) | $26654.06 \pm 38.60$ | 60 |
| **Grey-tailed tattler** | | |
| $\lvert\mathcal{S}\rvert = 2916, \lvert\mathcal{A}\rvert = 6, \lvert\Omega\rvert = 972$ | | |
| symbolic Perseus* | 4520 | 378 |
| IGRES (# subgoals: 72) | $4860.91 \pm 38.47$ | 60 |
| IGRES (# subgoals: 72) | $4927.17 \pm 38.14$ | 300 |

* Results from (Nicol et al. 2013).

Table 2: Results of adaptive management of migratory birds.

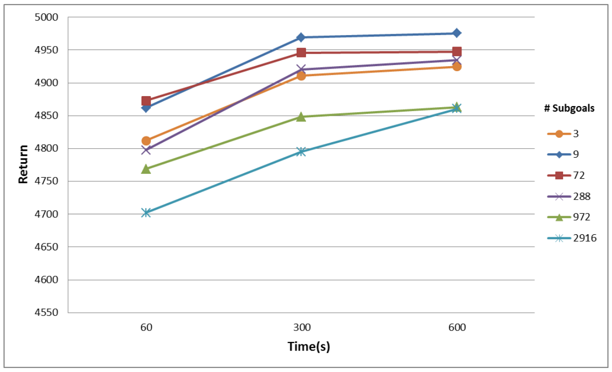

Figure 1: Performance of IGRES on Grey-tailed Tattler.

## Conclusion

In this paper, we present a new multi-purpose POMDP planning algorithm, IGRES, which exploits state information to identify subgoals that are essential for computing good policies, and automatically generates macro-actions to improve computational efficiency while maintaining good performance. Despite the simplicity of macro-actions generation, IGRES computes better solution at up to 10 times faster speed for some problems, and successfully generalizes to a wide set of POMDP tasks. We also present improved empirical performance for a set of real-world challenge tasks in ecological management, with significant potential impact. These promising results suggest that the notion of gathering information for exploitation of subgoals with macro-actions provides a new perspective to view POMDP problems, which advances application of POMDP planning for complex tasks in practice. Our code will be publicly released to help future efforts.[1]

## Acknowledgements

This work was supported by the Natural Sciences and Engineering Research Council (NSERC) through the Discovery Grants Program and the NSERC Canadian Field Robotics Network (NCFRN), as well as by the Fonds de Recherche du Quebec Nature et Technologies (FQRNT).

## References

Bonet, B., and Geffner, H. 2009. Solving POMDPs: RTDP-bel vs. point-based algorithms. In *International Jont Conference on Artifical Intelligence*.

Cassandra, A. R.; Kaelbling, L. P.; and Kurien, J. A. 1996. Acting under uncertainty: Discrete Bayesian models for mobile-robot navigation. In *International Conference on Intelligent Robots and Systems*.

Cassandra, A. R.; Kaelbling, L. P.; and Littman, M. L. 1994. Acting optimally in partially observable stochastic domains. In *National Conference on Artificial Intelligence*.

Cassandra, A. R. 1998. A survey of POMDP applications. In *AAAI 1998 Fall Symposium on Planning with Partially Observable Markov Decision Processes*.

Chades, I.; Carwardine, J.; Martin, T. G.; Nicol, S.; Sabbadin, R.; and Buffet, O. 2012. MOMDPs: A solution for modelling adaptive management problems. In *National Conference on Artificial Intelligence*.

Erwig, M. 2000. The graph Voronoi diagram with applications. *Networks* 36(3):156–163.

He, R.; Brunskill, E.; and Roy, N. 2010. Puma: Planning under uncertainty with macro-actions. In *National Conference on Artificial Intelligence*.

Hsiao, K.; Kaelbling, L. P.; and Lozano-Pérez, T. 2007. Grasping POMDPs. In *IEEE International Conference on Robotics and Automation*.

Kaelbling, L. P.; Littman, M. L.; and Cassandra, A. R. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101:99–134.

Kurniawati, H.; Du, Y.; Hsu, D.; and Lee, W. S. 2011. Motion planning under uncertainty for robotic tasks with long time horizons. *International Journal of Robotics Research* 30(3):308–323.

Kurniawati, H.; Hsu, D.; and Lee, W. S. 2008. SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *Robotics: Science and Systems*.

Littman, M. L.; Cassandra, A. R.; and Kaelbling, L. P. 1995. Learning policies for partially observable environments: Scaling up. In *International Conference on Machine Learning*.

Littman, M. L. 1996. *Algorithms for sequential decision-making*. Ph.D. Dissertation, Brown University.

Nicol, S.; Buffet, O.; Iwamura, T.; and Chadès, I. 2013. Adaptive management of migratory birds under sea level rise. In *International Joint Conference on Artificial Intelligence*.

Pineau, J., and Atrash, A. 2007. Smartwheeler: A robotic wheelchair test-bed for investigating new models of human-robot interaction. In *AAAI Spring Symposium: Multidisciplinary Collaboration for Socially Assistive Robotics*.

Pineau, J.; Gordon, G.; and Thrun, S. 2003. Point-based value iteration: An anytime algorithm for POMDPs. In *International Joint Conference on Artificial Intelligence*.

Poupart, P. 2005. *Exploiting structure to efficiently solve large scale partially observable Markov decision processes*. Ph.D. Dissertation, University of Toronto.

Shani, G.; Brafman, R. I.; and Shimony, S. E. 2007. Forward search value iteration for POMDPs. In *International Joint Conference on Artifical Intelligence*.

Silver, D., and Veness, J. 2010. Monte-carlo planning in large POMDPs. In *Advances in Neural Information Processing Systems*.

Smith, T., and Simmons, R. G. 2004. Heuristic search value iteration for POMDPs. In *Uncertainty in Artificial Intelligence*.

Smith, T., and Simmons, R. G. 2005. Point-based POMDP algorithms: Improved analysis and implementation. In *Uncertainty in Artificial Intelligence*.

Sondik, E. J. 1978. The optimal control of partially observable Markov processes over the infinite horizon: Discounted costs. *Operations Research* 26:282–304.

[1] The software package will be available at: http://cs.mcgill.ca/%7Ehma41/IGRES/.