Space-Progressive Value Iteration: An Anytime Algorithm for a Class of POMDPs

Nevin L. Zhang and Weihong Zhang
{lzhang,wzhang}@cs.ust.hk

Department of Computer Science
Hong Kong University of Science & Technology
Clear Water Bay, Kowloon, Hong Kong, China

Abstract. Finding optimal policies for general partially observable Markov decision processes (POMDPs) is computationally difficult primarily due to the need to perform dynamic-programming (DP) updates over the entire belief space. In this paper, we first study a somewhat restrictive class of special POMDPs called almost-discernible POMDPs and propose an anytime algorithm called space-progressive value iteration(SPVI). SPVI does not perform DP updates over the entire belief space. Rather it restricts DP updates to a belief subspace that grows over time. It is argued that given sufficient time SPVI can find near-optimal policies for almost-discernible POMDPs. We then show how SPVI can be applied to more a general class of POMDPs. Empirical results are presented to show the effectiveness of SPVI.

1 Introduction

Partially observable Markov decision processes (POMDPs) provide a general framework for AI planning problems where effects of actions are nondeterministic and the state of the world is not known with certainty. Unfortunately, finding optimal policies for general POMDPs is computationally very difficult [8]. Despite of much recent progresses [6, 4, 5, 11], our ability to battle the computational complexity of general POMDPs is still limited. It is therefore advisable to study special classes of POMDPs and design special-purpose algorithms for them.

Several classes of special POMDPs have been previously investigated. For example, Hansen [4] has studied *memory-resetting* POMDPs, where there are actions that give perfect information about the current state of the world. Zhang and Liu [10] have examined *region-observable* POMDPs, where one always knows that the world must be in one set of a handful of possible states.

General POMDP problems are difficult to solve primarily due to the need to perform dynamic programming (DP) update over the entire belief space. Solving special POMDP problems is easier because they permit one to focus on a subspace. In fully observable Markov decision processes (MDPs), for instance, one needs to deal only with the set of extreme belief states. In memory resetting POMDPs, one needs to deal only with extreme belief states and those reachable from them within a limited number of steps.

This paper considers POMDPs where there are two types of actions that are informally said to be *information-rich* and *information-poor* respectively. An information-rich action, when executed, always gives one a good idea about the current state of the world. In other words, the possible belief states after an information-rich action comprise only a small subspace of the belief space. An information-poor action, on the other hand, provides little or no information. We call this class of POMDPs *near-discernible POMDPs* since one can get a good idea about the current state of the world by executing an information-rich action.

It is arguable that near-discernible POMDPs are general enough for many applications with a large flat state space. In some path planning problems, for instance, actions can be divided into those that gather information and those that affect the state of the world. Often actions in the first category are information-rich and those in the second are information-poor.

We propose an anytime algorithm for near-discernible POMDPs. The algorithm is called *space-progressive value iteration (SPVI)*. To avoid high complexity, SPVI does not perform DP updates over the entire belief space. Rather, it restricts DP updates and hence value iteration to a belief subspace and, when convergence is reached, expands the subspace if time permits. So, SPVI works in a fashion similar to the envelope algorithm for MDPs [3].

For technical reasons, we develop SPVI first for a subclass of near-discernible POMDPs called *almost-discernible* POMDPs. In this subclass, it is easier to discuss what initial belief subspace to begin with and how to expand a belief subspace. It is also possible to argue for optimality of the policies found.

SPVI has been evaluated on a number of test problems. Some of the problems are solvable by the exact algorithm described in [11], which is probably the most efficient exact algorithm. For those problems, SPVI was able to find near-optimal policies in much less time. For the other problems, SPVI was still able to find near-optimal policies within acceptable time.

2 Technical Background

A POMDP is a sequential decision model for an agent who acts in a stochastic environment with only partial knowledge about the state of the world. In a POMDP model, the environment is described by a set of states $\mathcal{S}$. The agent changes the states by executing one of a finite set of actions $\mathcal{A}$. At each point in time, the world is in one state s. Based on the information it has, the agent chooses and executes an action a. Consequently, it receives an *immediate reward* $r(s, a)$ and the world moves stochastically into another state s' according to a *transition probability* $P(s'|s, a)$. Thereafter, the agent receives an observation z from a finite set $\mathcal{Z}$ according to an *observation probability* $P(z|s', a)$. The process repeats itself.

Information that the agent has about the current state of the world can be summarized by a probability distribution over $\mathcal{S}$[1]. The probability distribution is called a *belief state* and is denoted by b. The set of all possible belief states is called the *belief space* and is denoted by $\mathcal{B}$. If the agent observes z after taking action a in belief state b,

its next belief state b' is updated as

$$b'(s') = \frac{\sum_s P(z|s',a)P(s'|s,a)b(s)}{\sum_{s'}\sum_s P(z|s',a)P(s'|s,a)b(s)}.$$

We will sometimes denote this new belief state by $\tau(b,a,z)$. The dominator is the probability of observing z after taking action a in belief state b and will be denoted by $P(z|b,a)$.

A *policy* prescribes an action for each possible belief state. In other words, it is a mapping from $\mathcal{B}$ to $\mathcal{A}$. Associated with policy π is its *value function* V^π. For each belief state b, $V^\pi(b)$ is the expected total discounted reward that the agent receives by following the policy starting from b, i.e. $V^\pi(b) = E_{\pi,b}[\sum_{t=0}^{\infty}\lambda^t r_t]$, where r_t is the reward received at time t and λ ($0 \leq \lambda < 1$) is the *discount factor*. It is known that there exists a policy π^* such that $V^{\pi^*}(b) \geq V^\pi(b)$ for any other policy π and any belief state b. Such a policy is called an *optimal policy*. The value function of an optimal policy is called the *optimal value function*. We denote it by V^*. For any positive number ϵ, a policy π is *ϵ-optimal* if $V^\pi(b) + \epsilon \geq V^*(b)$ for any belief state b. Without explicitly referring to ϵ, we will sometimes say that such a policy is *near-optimal*.

The *dynamic programming(DP) update operator* T maps a value function V to another value function TV that is defined as follows: for any b in $\mathcal{B}$,

$$TV(b) = \max_a[r(b,a) + \lambda\sum_z P(z|b,a)V(\tau(b,a,z))]$$

where $r(b,a) = \sum_s r(s,a)b(s)$ is the expected immediate reward for taking action a in belief state b.

Value iteration is an algorithm for finding ϵ-optimal policies. It starts with an initial value function V_0 and iterates using the formula: $V_n = TV_{n-1}$. Because T is a contraction mapping, V_n converges to V^* as n goes to infinity. Value iteration terminates when the *Bellman residual* $\max_b |V_n(b) - V_{n-1}(b)|$ falls below $\epsilon(1-\lambda)/2\lambda$. When it does, the so-called *V_n-improving* policy given below is ϵ-optimal: for any b in $\mathcal{B}$,

$$\pi(b) = \arg\max_a[r(b,a) + \lambda\sum_z P(z|b,a)V_n(\tau(b,a,z))].$$

Functions over the state space $\mathcal{S}$ are sometimes referred to as *vectors*. A set $\mathcal{V}$ of vectors *represents* a value function f in a subspace $\mathcal{B}'$ of belief states if $f(b) = \max_{\alpha\in\mathcal{V}}\alpha.b$ for any b in $\mathcal{B}'$. The notion $\alpha.b$ means the inner product of α and b. We will sometimes write $f(b)$ as $\mathcal{V}(b)$. A vector in a set is *extraneous in $\mathcal{B}'$* if, after its removal, the set represents that same value function in $\mathcal{B}'$. It is *useful in $\mathcal{B}'$* otherwise. If a value function f is representable by a set of vectors in $\mathcal{B}'$, then there is a minimum set that represents f in $\mathcal{B}'$. None of the vectors in this unique set are extraneous in $\mathcal{B}'$. A value function that is representable as a finite set of vectors in the entire belief space is said to be *piecewise linear and convex (PLC)*.

Since there are infinitely many possible belief states, value functions cannot be explicitly represented as a table. Sondik [9] has shown that if a value function V is PLC, then so is TV. Consequently, if the initial value function is PLC, every value function generated by value iteration is PLC and hence can be represented by a finite number of vectors.

3 Space-Progressive Value Iteration in Almost-Discernible POMDPs

In this section, we define almost-discernible POMDPs and outline space-progressive value iteration (SPVI). We also explain why SPVI is feasible and argue that given sufficient time it can find near-optimal policies.

3.1 Almost-Discernible POMDPs

For any action a, let $\mathcal{Z}_a$ be the set of observations that are possible after taking action a. For any z in $\mathcal{Z}_a$, define $\mathcal{S}_{az} = \{s : P(z|a, s)>0\}$. If z is observed after action a, the true state must be in the set $\mathcal{S}_{az}$ and one's belief state must be in $\mathcal{B}_{az} = \{b : b(s)=0\ \forall s \notin \mathcal{S}_{az}\}$.

If there is an integer k such that $|\mathcal{S}_{az}| \leq k$ for every z in $\mathcal{Z}_a$, then executing a would narrow the true state down to no more than k possibilities. When this is the case, we say that the action is *k-focal*. For convenience, we will use the term *information-opulent actions* to refer to actions that are k-focal for some small integer k.

We say that a POMDP is *k-discernible* if it consists of one or more k-focal actions and all other actions are *information-void* in the sense that they do not yield any observations. For convenience, we will use that term *almost-discernible POMDPs* to refer to POMDPs that are k-discernible for some small k. To simplify exposition, we assume that there is only one information-opulent action and will denote it by d. Note that after executing d, one's belief state must be in $\cup_{z\in\mathcal{Z}_d}\mathcal{B}_{dz}$, where the union is taken over all possible observations that d might produce.

Almost-discernible POMDPs are obviously a generation of memory-resetting POMDPs. They also generalize region-observable POMDPs: An almost-discernible POMDP is region-observable if all actions are information-opulent.

3.2 Space-Progressive Value Iteration

SPVI starts with the belief subspace $\cup_{z\in\mathcal{Z}_d}\mathcal{B}_{dz}$. It performs value iteration restricted to the subspace and, when convergence is reached, expands the subspace. SPVI continues subspace expansions as long as time permits. When time runs out, SPVI returns the best policy found so far [1].

Belief Subspace Representation In this paper, we consider only subspaces that are unions of belief simplexes. A *belief simplex* is specified by a list of *extreme points*. The simplex with extreme points b_1, b_2, ..., b_k consists of all belief states of the form $\sum_{i=1}^{k} \lambda_i b_i$ where $\lambda_i \geq 0$ and $\sum_{i=1}^{k} \lambda_i = 1$.

The initial belief subspace $\cup_{z\in\mathcal{Z}_d}\mathcal{B}_{dz}$ is the union of belief simplexes. To see this, define, for any state s, χ_s to be the extreme belief state such that $\chi_s(s')=1$ if and only if $s'=s$. It is clear that $\mathcal{B}_{dz}$ is a simplex with the following set of extreme points: $\{\chi_s : s \in S_{dz}\}$.

[1] The quality of a policy can be determined by simulation.

In the rest of this subsection, we consider a belief subspace $\mathcal{B}'$ that consists of m simplexes $\mathcal{B}'_1, \mathcal{B}'_2, \ldots, \mathcal{B}'_m$. We explain how to restrict DP update to $\mathcal{B}'$, how to guarantee the convergence of value iteration when restricted to $\mathcal{B}'$, and how to expand the belief subspace $\mathcal{B}'$ when convergence is reached.

Restricted DP Update DP update takes as input the minimum set of vector that represent a PLC value function V and computes the minimum set $\mathcal{U}$ of vectors that represents TV in the entire belief space. A simple and comparatively efficient algorithm for DP update is incremental pruning [10, 2]. It solves a series of linear programs. Each linear program involves a vector α and a set of other vectors $\mathcal{W}$. The purpose of the linear program is to determine whether there is a belief state b where α *dominates* vectors in $\mathcal{W}$, i.e. $\alpha.b > \beta.b$ for all $\beta \in \mathcal{W}$. The linear program is as follows:

Maximize: x.
Constraints:
$b.\alpha \geq x + b.\beta \; \forall \beta \in \mathcal{W}$
$\sum_s b(s) = 1, b(s) \geq 0 \; \forall s \in \mathcal{S}$

The vector α dominates vectors in $\mathcal{W}$ at some belief states if and only the optimal value of x is positive. We will refer to this linear program as $LP(\alpha, \mathcal{W}, \mathcal{B})$.

Restricting DP update to the belief subspace $\mathcal{B}'$ means to compute the minimum set $\mathcal{U}'$ of vectors that represents TV in that subspace. To do so, SPVI first does this in each belief simplex $\mathcal{B}'_j$, resulting in a set of vectors $\mathcal{U}'_j$. It then takes the union of those sets. Some vectors in the union might be extraneous in $\mathcal{B}'$. All such vectors can be identified and removed by checking for duplicates.

Restricting DP update to a simplex of belief states requires little change to incremental pruning. Let $\mathcal{B}'_j$ be the simplex with extreme points $b_1, b_2, \ldots, b_k$. To restrict DP update to $\mathcal{B}'_j$, all one has to do is to modify each linear program $LP(\alpha, \mathcal{W}, \mathcal{B})$ as follows:

Maximize: x.
Constraints:
$\sum_{i=1}^{k} \lambda_i b_i.\alpha \geq x + \sum_{i=1}^{k} \lambda_i b_i.\beta \; \forall \beta \in \mathcal{W}$
$\sum_{i=1} \lambda_i = 1, \lambda_i \geq 0 \; \forall 1 \leq i \leq k$

We will refer to this linear program as $LP(\alpha, \mathcal{W}, \mathcal{B}'_j)$.

Restricting DP update to a subspace reduces computational complexity for two reasons. First, the number of variables in $LP(\alpha, \mathcal{W}, \mathcal{B}'_j)$ is $k+1$, while that in $LP(\alpha, \mathcal{W}, \mathcal{B})$ is $|\mathcal{S}|+1$. When solving near-discernible POMDPs, k is always much smaller than $|\mathcal{S}|$. Consequently, the numbers of variables in linear programs are reduced. Second, one needs fewer vectors to represents TV in a subspace than in the entire belief space. This implies fewer linear programs and fewer constraints in linear programs.

Restricted Value Iteration Restricting value iteration to a belief subspace means to restrict DP update to the subspace at each iteration. Unlike unrestricted DP update, restricted DP update is not necessarily a contraction mapping. There is therefore an issue

of ensuring the convergence of restricted value iteration. A simple solution is to start SPVI with a value function that is sufficiently small so that it is upper bounded by the optimal value function and to maintain monotonicity during restricted value iteration.

For each simplex $\mathcal{B}'_j$, let $\mathcal{V}_j$ be the minimum set of vectors that represents V in the simplex. This was computed in the previous iteration. After DP update in $\mathcal{B}'_j$, we get another set $\mathcal{U}'_j$ of vectors. To maintain monotonicity, SPVI takes the union $\mathcal{V}_j \cup \mathcal{U}'_j$ and then prunes vectors that are extraneous in $\mathcal{B}'_j$ by solving linear programs of the form $LP(\alpha, \mathcal{W}, \mathcal{B}'_j)$.

To determine whether to terminate restricted value iteration, SPVI considers the quantity $\max_{j=1}^{m} \max_{b \in \mathcal{B}'_j} [\mathcal{U}'_j(b) - \mathcal{V}_j(b)]$, which can be computed by solving linear programs of the form $LP(\alpha, \mathcal{V}_j, \mathcal{B}'_j)$. It terminates restricted value iteration when the quantity drops below a predetermined threshold. In our experiments, it is set at $0.1(1-\lambda)/2\lambda$ — the threshold for Bellman residual that exact value iteration would use in order to find 0.1-optimal policies.

Belief Subspace Expansion When restricting value iteration to $\mathcal{B}'$, we are concerned only with values for belief states in $\mathcal{B}'$. However, values for some belief states in $\mathcal{B}'$ might depend on values for belief states outside $\mathcal{B}'$. Hence it is natural to, when convergence is reached in $\mathcal{B}'$, expand $\mathcal{B}'$ to include all such belief states.

Let V and U be the second last and the last value functions produced by restricted value iteration before it converges in $\mathcal{B}'$. Further let π be the V-improving policy. The value of U at a belief state b in $\mathcal{B}'$ depends on the values of V at belief states that are reachable from b be executing action $\pi(b)$.

For any belief simplex $\mathcal{B}'_j$, any action a, and any z in $\mathcal{Z}_a$, define $\tau(\mathcal{B}'_j, a, z) = \{\tau(b, a, z) : b \in \mathcal{B}'_j\}$. Suppose $\mathcal{B}'_j$ has k extreme points b_1, b_2, ..., b_k. It can be shown that $\tau(\mathcal{B}'_j, a, z)$ is a simplex with the following set of extreme points

$$\{\tau(b_i, a, z) : i \in \{1, 2, \ldots, k\} \text{ and } P(z|a, b_i) > 0)\}$$

Note that $\tau(\mathcal{B}'_j, a, z)$ has no more extreme points than $\mathcal{B}'_j$.

It is clear that $\cup_{z \in \mathcal{Z}_a} \tau(\mathcal{B}'_j, a, z)$ is the set of belief states reachable from inside $\mathcal{B}'_j$ by executing action a. Let $\mathcal{A}_j$ be the set of actions that π prescribes for belief states in $\mathcal{B}'_j$. Then $\cup_{j=1}^{m} \cup_{a \in \mathcal{A}_j} \cup_{z \in \mathcal{Z}_a} \tau(\mathcal{B}'_j, a, z)$ contains all the belief states whose values under V influence the values of U inside $\mathcal{B}'$.

SPVI expands $\mathcal{B}'$ by including more belief simplexes: for each j, each a in $\mathcal{A}_j$, and each z in $\mathcal{Z}_a$, it adds the simplex $\tau(\mathcal{B}'_j, a, z)$ to the collection of simplexes if it is not a subset of any existing simplexes.

3.3 Feasibility

SPVI would not be computationally feasible if the number of belief simplexes increases quickly. In this subsection, we argue that this is not the case thanks to the properties of almost-discernible POMDPs and the way the initial belief simplexes are chosen.

First note that when a is the information-opulent action d, the simplex $\tau(\mathcal{B}'_j, d, z)$ is a subset of $\mathcal{B}_{dz}$, which is in the collection to begin with. Consequently, it is not

added to the collection. Second, each $\mathcal{B}'_j$ is small in size. The information-void actions that the V-improving policy prescribes for belief states are usually only a fraction of all information-void actions. Moreover, each information-void action has only one possible observation, namely the void observation. All those point to the conclusion that the number of belief simplexes does not increase quickly.

Although SPVI is proposed as an anytime algorithm, we argue below that, given sufficient time, SPVI will eventually *converge* in the sense that underlying belief subspace will stop growing. Define a *history* to be an ordered sequence of action-observation pairs. For any belief subspace $\hat{\mathcal{B}}$ and any history h, let $\tau(\hat{\mathcal{B}}, h)$ be the set of belief states that are possible when the history h is realized starting from inside $\hat{\mathcal{B}}$. Using this notation, we can rewrite each initial belief simplex $\mathcal{B}_{dz}$ as $\tau(\mathcal{B}, [d, z])$, where $[d, z]$ is the one step history where z is observed after taking action d. Similarly $\tau(\mathcal{B}'_j, a, z)$ can be rewritten as $\tau(\mathcal{B}'_j, [a, z])$.

It is easy to see that each belief simplex that SPVI encounters can be written as $\tau(\mathcal{B}, h)$ for some history h. As already pointed, each initial belief simplex $\mathcal{B}_{dz}$ can be written as $\tau(\mathcal{B}, [d, z])$. Now if $\mathcal{B}'_j$ is $\tau(\mathcal{B}, h)$, then $\tau(\mathcal{B}'_j, a, z)$ can be written as $\tau(\mathcal{B}, h')$, where h'=$h[a, z]$ is obtained by appending the pair $[a, z]$ to the end of h.

Let $\tau(\mathcal{B}, h)$ be a particular belief simplex that SPVI encounters. We claim that all actions in h are information-void except for the first one. This is trivially true if $\tau(\mathcal{B}, h)$ is an initial belief simplex $\tau(\mathcal{B}, [d, z])$. Suppose the statement is true if the length of h is k. Consider expanding $\tau(\mathcal{B}, h)$. For each action a and each $z \in \mathcal{Z}_a$, we get a new simplex $\tau(\mathcal{B}, h[a, z])$. When a is d, the new simplex is a subset of the initial belief simplex $\mathcal{B}_{dz}$ and hence is not added to the collection of simplexes. When a is d, all actions in $h[a, z]$ are information-void except for the first one. So the claim follows by induction.

Intuitively, after a long sequence of information-void actions, one should be quite uncertain about the current state of the world. As such, good policies should prescribe the information-opulent action for belief states in $\tau(\mathcal{B}, h)$ when h is long. The V-improving policy intuitively should be good as the underlying subspace grows large and hence should prescribe only the information-opulent action for belief states in $\tau(\mathcal{B}, h)$ when h is long. When this happens, no new belief simplexes will be produced from $\tau(\mathcal{B}, h)$. Consequently, SPVI should eventually stop introducing new belief simplexes and hence converge.

It should be noted that the foregoing arguments do not constitute proofs. The conclusions drawn might not be true in all cases. However, they are indeed found to be true in the problems used in our experiments.

3.4 Optimality

Assume SPVI does eventually converge. In this subsection, we argue that SPVI can find near-optimal policies.

Let $\mathcal{B}'$ be the final belief subspace, V and U be the second last and the last value functions, and π be the V-improving policy. The way that SPVI expands a subspace and the fact that $\mathcal{B}'$ is the final subspace imply that $\mathcal{B}'$ is *closed* under π in the following sense: Starting from inside $\mathcal{B}'$, the policy does not lead to belief states outside $\mathcal{B}'$. Let $\mathcal{P}_{\mathcal{B}'}$ be the set of all policies under which $\mathcal{B}'$ is closed. In the following, we prove that π

is near-optimal in $\mathcal{B}'$ among all policies in $\mathcal{P}_{\mathcal{B}'}$, i.e. $V^{\pi}(b)$ is close to $\max_{\pi' \in \mathcal{P}_{\mathcal{B}'}} V^{\pi'}(b)$ for any belief state b in $\mathcal{B}'$.

The POMDP under discussion can be transformed into an MDP over the belief space $\mathcal{B}$. Denote this MDP by $\mathcal{M}$. Define another MDP $\mathcal{M}'$ that is the same as $\mathcal{M}$ except that its state space is $\mathcal{B}'$ and possible actions for each belief state in $\mathcal{B}'$ include only those that do not lead to belief states outside $\mathcal{B}'$. Let V' and π' be respectively the restrictions of V and π onto $\mathcal{B}'$. Imagine carrying out value iteration for $\mathcal{M}'$ starting from V'. Let U' be the value function obtained after the first iteration. The fact that $\mathcal{B}'$ is closed under π implies that π' is a legitimate policy for $\mathcal{M}'$ and is V'-improving. It also implies that $U'(b)=U(b)$ for any b in $\mathcal{B}'$. Hence $\max_{b \in \mathcal{B}'} |V'(b) - U'(b)| = \max_{b \in \mathcal{B}'} |V(b) - U(b)|$. If restricted value iteration was terminated when the latter quantity is small, then π' is a near-optimal policy for $\mathcal{M}'$. Together with the fact all policies in $\mathcal{P}_{\mathcal{B}'}$, when restricted to $\mathcal{B}'$, are policies for $\mathcal{M}'$, this allows us to conclude π is near-optimal (for the original POMDP) in $\mathcal{B}'$ among policies in $\mathcal{P}_{\mathcal{B}'}$.

The intuition that near-optimal policies should prescribe the information-opulent action for belief state in $\tau(\mathcal{B}, h)$ when h is long gives us some reason to believe that the set $\mathcal{P}_{\mathcal{B}'}$ contains near-optimal policies. If this is the case, the policy π is near-optimal in $\mathcal{B}'$ among all policies. Since one can always get into $\mathcal{B}'$ be taking the information-opulent action, π is near-optimal in the entire belief space.

As in the previous subsection, we wish to note that some of the foregoing arguments rely strongly on intuitions. The conclusion drawn might not be true in all cases. However, our experiments do indicate that SPVI can find near-optimal policies after a few subspace expansions.

4 Empirical results

We have tested SPVI on a number of problems. The results are encouraging. Due to space limit, we discuss only the results on a simple maze game. The layout of the maze is shown in Figure 1. There are 11 states: 10 locations plus one terminal state. An agent needs to move to location 9 and declare goal. There are six actions: four "move" actions plus `look` and `declare-goal`. The `look` action is information-opulent (2-focal) and all other actions are information-void.

The "move" actions allow the agent to move in each of the four nominal directions and have 80% chance of achieving their intended effects, i.e. moving the agent one step in certain direction. Moving against maze walls leaves the agent at its original location. These actions have no rewards. At locations 9 and 10, the `declare-goal` action yield rewards of +1 and -1 respectively and it moves the game into the terminal state. In all other states, it does not cause state transitions and has no reward. The `look` action does not have rewards and does not cause state transitions. It produces observations except in the terminal state. The observation it produces at a location is a sequence of four letters indicating, for each of the four directions, where there is a wall (w) or nothing (o). There are 4 pairs of locations that have the same observations. For example, locations 2 and 5 both have observation `owow`, meaning that there are walls in the South and North and the other two directions are open. Observations for the other 2 locations are unique.

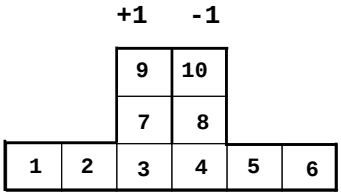

Fig. 1. Layout of test problem domain.

We have run SPVI on two versions of the maze problem: the original problem and a simpler version where locations 1 and 6 are deleted. The two versions will be referred to as Maze and Maze1 respectively. The point-based improvement technique described in Zhang and Zhang [11] is incorporated to speed up the convergence of restricted value iteration. The discount factor is set at 0.95. After restricted value iteration converges in a belief subspace, simulation is conducted to determine the quality of the policy found. The simulation consists of 1000 trials. Each trial starts from a random initial belief state and is allowed to run up to 20 steps. The average reward across all the trials is used as a measurement of policy quality.

The charts in Figure 2 depict quality of policies SPVI found in the belief subspaces against the time taken. Simulation time is not included. There is one data point for each belief subspace. What is shown for a subspace is not necessarily the quality of the policy found in that subspace. Rather, it is the quality of the best policy found so far. We present data this way for the following reason. As subspace grows, SPVI computes better and better value functions. However, as is well known, better value function does not necessarily imply better policy. In other words, the policy found in a larger subspace is not necessarily better than the one found in a smaller subspace. Knowing this fact, one naturally would want to keep the best policy so far.

In Maze1, SPVI converged after 11 expansions and the final number of simplexes is 100. In Maze, it converged after 22 expansions and the final number of simplexes is 432. These support our claims that the number of simplexes does not grow quickly and subspace expansion will eventually terminate.

Using the algorithm described in Zhang and Zhang [11], we were able to compute 0.1-optimal policies for Maze1 in 6,457 seconds. This provides us with a benchmark to judge how close the policies SPVI found are to the optimal. From Figure 2, we see that SPVI found a near-optimal policy for Maze1 in less than 50 seconds after two subspace expansions [2]. This is much less time than 6,457 seconds.

For Maze, the algorithm described in Zhang and Zhang [11] did not converge after 24 hours. On the other hand, SPVI was able to find a policy whose average reward is 0.46 in 13 seconds after only one subspace expansion and another policy whose average reward is 0.48 in 520 seconds after 6 subspace expansions. Since the optimal average

[2] One might notice that some of the policies found by SPVI appear to be better than the 0.1-optimal policy. This is probably due to randomness in simulation.

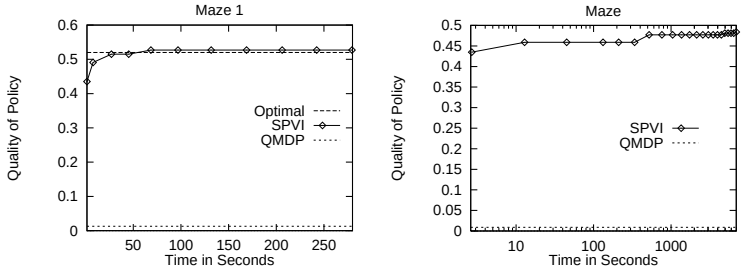

Fig. 2. Performance of SPVI in two almost-discernible POMDPs.

reward one can get in Maze should be less than that in Maze1 and the latter is 0.52, there are good reasons to believe the two policies found by SPVI are near-optimal.

QMDP [7] is a simple approximation method whose performance has been shown to be fairly good compared to other approximation methods [5]. It is computationally much cheaper than SPVI. Naturally, we are interested in how the two methods differ in terms of quality of policies they obtain. From Figure 2, it is clear that SPVI found much better policies than QMDP.

The poor quality of the QMDP approximation in our test problems can easily be explained. The symmetric nature of the domain mains that our agent can easily confuses the left and right halves. The only way to disambiguate the uncertainty is to move to either location 1 or location 6. Policies produced by QMDP simply do not have such sophisticated information-gathering strategy.

5 Near-Discernible POMDPs

The conditions that define almost-discernible POMDPs are rather restrictive. In this section, we relax those conditions to define a more general class of POMDPs called near-discernible POMDPs and show how SPVI can be adapted for such POMDPs.

We say that an action a is *information-rich* if, for each observation z in $\mathcal{Z}_a$, the probability $P(z|s,a)$ is close to zero except for a small number of states. On the other hand, if $P(z|s,a)$ is significantly larger than zero for a large number of states, then we say that a is *information-poor*. A POMDP is *near-discernible* if its actions are either information-rich or information-poor. To simplify exposition, we assume that there is only one information-rich action and we denote this action by d.

One difficulty of applying SPVI to near-discernible POMDPs is that the set $\mathcal{S}_{dz}$ can be large in cardinality. When this is the case, the belief simplex $\mathcal{B}_{dz}$ is also large and consequently the complexity of SPVI would be high if it starts with the belief subspace $\cup_{z\in\mathcal{Z}_d}\mathcal{B}_{dz}$. To overcome this difficulty, we introduce a number δ that close to zero and define $\mathcal{S}^{\delta}_{az} = \{s : P(z|a,s)>\delta\}$. By the definition of information-rich actions, the set $\mathcal{S}^{\delta}_{dz}$ should be small. We further define $\mathcal{B}^{\delta}_{az} = \{b : b(s)=0\forall s \notin \mathcal{S}^{\delta}_{az}\}$ and let SPVI starts with the belief subspace $\cup_{z\in\mathcal{Z}_d}\mathcal{B}^{\delta}_{dz}$. In our experiments, δ is set at 0.1.

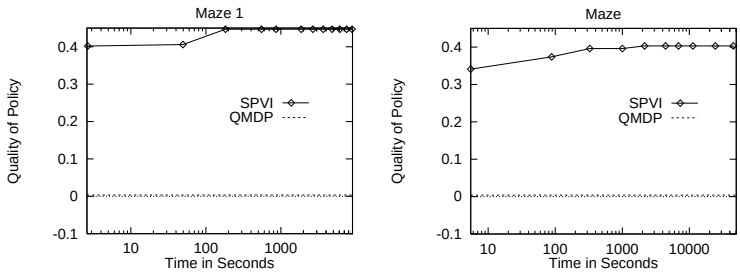

Fig. 3. Performance of SPVI in two near-discernible POMDPs.

The above way of choosing the initial belief subspace gives rise to another issue: new belief simplexes $\tau(\mathcal{B}'_j, d, z)$ generated by the information-rich action are not necessarily a subset of the initial belief simplex $\mathcal{B}^{\delta}_{dz}$. If belief subspaces are expanded in the same way as in almost-discernible POMDPs, the number of belief simplexes will increase quickly. To avoid this problem, new belief simplexes $\tau(\mathcal{B}'_j, d, z)$ generated by the information-rich action are simply discarded.

To evaluate the performance of SPVI in near-discernible POMDPs, we modified the two problems described in the previous section. The only changes are in the observation probability of the `look`. In the original problems, `look` produces, at each location, a string of four characters with probability 1. Let us say that the string is the ideal string for the location. In the modified problems, *look* produces, at each location, the ideal string for that location with probability around 0.8. With probability 0.05, it produces the void observation. Also with probability 0.05, it produces a string that is ideal for some other location and that differs from the ideal string for the current location by no more than 2 characters. After the modifications, `look` is no longer an information-opulent action. It produces the void observation with nonzero probability at all states.

The performance of SPVI in the two modified problems is shown in Figure 3. In modified Maze1, SPVI converged after 11 subspace expansions. In modified Maze, it was manually terminated after 9 subspace expansions. Intuitively, the maximum average rewards one can get in the modified problems should be less than those in the original problems. This and a quick comparison of Figures 2 and 3 suggest that the policies that SPVI found for the modified problems after two subspace expansions are near-optimal. They are much better than the policies found by QMDP.

6 Conclusions

Finding optimal policies for general partially observable Markov decision processes (POMDPs) is computationally difficult primarily due to the need to perform dynamic-programming (DP) updates over the entire belief space. In this paper, we first studied a somewhat restrictive class of special POMDPs called almost-discernible POMDPs and proposed an anytime algorithm called space-progressive value iteration (SPVI). SPVI does not perform DP updates over the entire belief space. Rather it restricts DP

updates to a belief subspace that grows over time. It was argued that given sufficient time SPVI can find near-optimal policies for almost-discernible POMDPs. We then showed how SPVI can be applied to more a general class of POMDPs called near-discernible POMDPs, which is arguably general enough for many applications. We have evaluated SPVI on a number of test problems. For those that are solvable by previous exact algorithms, SPVI was able to find near-optimal policies in much less time. For the others, SPVI was still able to find, with acceptable time, policies that are arguably near-optimal.

Acknowledgements. The authors are grateful to the three anonymous reviewers for their helpful comments. This work has been supported by Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. HKUST658/95E).

References

1. K. J. Astrom. Optimal control of Markov decision processes with incomplete state estimation. *Journal of Mathematical Analysis and Applications*, 10, 174-205, 1965.
2. A. R. Cassandra, M. L. Littman, and N. L. Zhang. Incremental pruning: a simple, fast, exact method for partially observable Markov decision processes. *Proceedings of Thirteenth Conference on Uncertainty in Artificial Intelligence*, 54-61, 1997.
3. T. Dean, L. P. Kaelbling, J. Kirman and A. Nicholson. Planning under time constraints in stochastic domains. *Artificial Intelligence*, Vol 76, Num 1-2, 35-74, 1995.
4. E. A. Hansen. Finite-memory controls of partially observable systems. PhD thesis, Depart of Computer Science, University of Massachusetts at Amherst, 1998.
5. M. Hauskrecht. Value-function approximations for partially observable Markov decision processes. *Journal of Artificial Intelligence Research*, 13, 33-94, 2000.
6. L. P. Kaelbling, M. L. Littman and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101, 99-134, 1998.
7. M. L. Littman, A. R. Cassandra and L. P. Kaelbling. Learning policies for partially observable environments: scaling up. *Proceedings of the Twelfth International Conference on Machine Learning*, 263-370, 1995.
8. C. H. Papadimitriou and J. N. Tsitsiklis(1987). The complexity of Markov decision processes. *Mathematics of Operations Research*, 12(3), 441-450, 1987.
9. E. J. Sondik. The optimal control of partially observable Markov processes. PhD thesis, Stanford University, 1971.
10. N. L. Zhang and W. Liu. A model approximation scheme for planning in stochastic domains. *Journal of Artificial Intelligence Research*, 7, 199-230, 1997.
11. N. L. Zhang and W. Zhang. Speeding up the convergence of value iteration in partially observable Markov decision processes. *Journal of Artificial Intelligence Research*, 14, 29-51, 2001.