Point-Based Value Iteration for Constrained POMDPs*

Dongho Kim **Jaesong Lee** **Kee-Eung Kim**
Department of Computer Science
KAIST, Korea
{dkim, jaesong, kekim}@cs.kaist.ac.kr

Pascal Poupart
School of Computer Science
University of Waterloo, Canada
ppoupart@cs.uwaterloo.ca

Abstract

Constrained partially observable Markov decision processes (CPOMDPs) extend the standard POMDPs by allowing the specification of constraints on some aspects of the policy in addition to the optimality objective for the value function. CPOMDPs have many practical advantages over standard POMDPs since they naturally model problems involving limited resource or multiple objectives. In this paper, we show that the optimal policies in CPOMDPs can be randomized, and present exact and approximate dynamic programming methods for computing randomized optimal policies. While the exact method requires solving a minimax quadratically constrained program (QCP) in each dynamic programming update, the approximate method utilizes the point-based value update with a linear program (LP). We show that the randomized policies are significantly better than the deterministic ones. We also demonstrate that the approximate point-based method is scalable to solve large problems.

1 Introduction

Partially observable Markov decision processes (POMDPs) are widely used for modeling stochastic sequential decision problems under partial or uncertain observations. The standard POMDP model has the reward function which encodes the immediate utility of executing actions in environment states, and the optimal policy is obtained by maximizing the long-term reward. However, since the utility depends on multiple objectives in practice, it is often required to manually balance different objectives into the single reward function until the corresponding optimal policy is satisfactory to the domain expert. In addition, application domains of POMDPs generally have well-established measures for evaluating systems, and the domain experts typically have a hard time understanding the concept of value functions.

Constrained POMDPs (CPOMDPs) concern the situation where there is one criterion (reward) to be maximized while making other criteria (costs) below the prescribed thresholds. Each criterion is represented using its own reward or cost function. A typical situation is a resource-limited agent, e.g., a battery-equipped robot whose goal is to accomplish as many tasks as possible given a finite amount of energy. In fact, many problems in practice can be naturally formulated using a set of constraints. For example, POMDP-based spoken dialogue systems [Williams and Young, 2007] have to successfully complete dialogue tasks while minimizing the length of dialogues. We can use the CPOMDP to represent these two criteria by assigning a constant reward of -1 for each dialogue turn and a cost of 1 for each unsuccessful dialogue. By bounding the aggregate cost, the optimal policy from the CPOMDP is guaranteed to achieve certain level of dialogue success rate, *i.e.*, task completion rate (TCR), the performance measure which dialogue system experts are more comfortable with than the value function in standard POMDPs. Another example is POMDP-based opportunistic spectrum access (OSA) [Zhao *et al.*, 2007] in wireless communications. OSA seeks to maximize the utilization of wireless spectrum by allowing secondary devices to communicate through the wireless channel that is already allocated to primary devices. Since the communication collision with the primary device is potentially dangerous, there are regulatory requirements on the maximum collision rate that secondary devices have to meet in order to be approved. Such requirements can be naturally modeled as cost constraints, while the communication bandwidth is the reward that should be maximized.

Despite these advantages, the CPOMDP has not received as much attention as its MDP counterpart, *i.e.*, constrained MDPs (CMDPs) [Altman, 1999], with the exception of the dynamic programming method for finding deterministic optimal policies [Isom *et al.*, 2008]. In this paper, we first present a motivating CPOMDP example where the best deterministic policy is suboptimal. We then present our exact and approximate algorithms for finding randomized optimal policies in CPOMDPs. The exact algorithm is only of theoretical interest, since it is based on solving minimax quadratically constrained programs (QCPs) to prune useless policies. The approximate algorithm is motivated by point-based value iteration (PBVI) [Pineau *et al.*, 2006] in standard POMDPs, where we collect the samples of *admissible costs* [Piunovskiy and Mao, 2000] in addition to belief points. It thereby solves

*This paper is a detailed version of the paper presented in the IJCAI-11 main technical program.

linear programs (LPs) instead of computationally demanding minimax QCPs. We demonstrate the scalability of our method on the constrained version of a POMDP problem with thousands of states.

2 Preliminaries

In this section, we briefly review the definitions of CMDPs and CPOMDPs. We also explain the suboptimality of deterministic policies for CPOMDPs through an example.

2.1 Constrained POMDPs

The standard, unconstrained POMDP is defined as a tuple $\langle S, A, Z, T, O, R, \gamma, b_0 \rangle$: S is the set of states s; A is the set of actions a; Z is the set of observations z; T is the transition function where $T(s, a, s')$ denotes the probability $P(s'|s, a)$ of changing to state s' from state s by taking action a; O is the observation function where $O(s, a, z)$ denotes the probability $P(z|s, a)$ of making observation z when executing action a and arriving in state s; R is the reward function where $R(s, a)$ denotes the immediate reward of executing action a in state s; $\gamma \in [0, 1)$ is the discount factor; b_0 is the initial belief where $b_0(s)$ is the probability that we start in state s.

The constrained POMDP (CPOMDP) is defined as a tuple $\langle S, A, Z, T, O, R, \{C_k\}_{k=1}^K, \{\hat{c}_k\}_{k=1}^K, \gamma, b_0 \rangle$ with the following additional components: $C_k(s, a) \geq 0$ is the cost of type k incurred for executing action a in state s, and $\hat{c}_k$ is the upper bound on the cumulative cost of type k.

Solving a CPOMDP corresponds to finding an optimal policy π:

$$\text{maximize } E_\pi \left[\sum_{t=0}^\infty \gamma^t R(s_t, a_t)\right]$$

subject to the cumulative cost constraints:

$$E_\pi \left[\sum_{t=0}^\infty \gamma^t C_k(s_t, a_t)\right] \leq \hat{c}_k \quad \forall k. \tag{1}$$

Since the state is not directly observable in POMDPs and CPOMDPs, we often use the notion of belief, which is the probability distribution b over the current states. $b' = \tau(b, a, z)$ denotes the successor of belief b upon executing a and observing z, which is computed using Bayes theorem:

$$b'(s') = O(s', a, z) \sum_s T(s, a, s') b(s) / P(z|b, a). \tag{2}$$

The constrained MDP (CMDP) is defined as a tuple $\langle S, A, T, R, \{C_k\}_{k=1}^K, \{\hat{c}_k\}_{k=1}^K, \gamma, b_0 \rangle$, assuming complete state observability.

2.2 Suboptimality of deterministic policies

It is well known that optimal policies for CMDPs may be randomized. In [Altman, 1999], it is shown that when a CMDP is feasible, the number of randomizations under an optimal stationary policy is related to the number of constraints. More specifically, if we define the number of randomizations $m(s, \pi^*)$ under an optimal policy π^* in state s as $|\{a|\pi^*(a|s) > 0\}| - 1$, the total number of randomizations is $m(\pi^*) = \sum_{s \in S} m(s, \pi^*) \leq K$ where K is the number of constraints.

It has also been shown that, under the special condition of non-atomic initial distribution and transition probabilities, searching in the space of deterministic policies is sufficient to find optimal policies in CMDPs with uncountable state spaces [Feinberg and Piunovskiy, 2002]. A probability distribution is defined to be non-atomic if its cumulative distribution function is continuous. Since a POMDP can be formulated as an MDP with the continuous belief space [Kaelbling *et al.*, 1998], we may regard a CPOMDP as a CMDP with an uncountable state space. The non-atomic condition is however not met. The initial distribution is atomic since the b_0 is the only possible initial state. The transition probabilities, which are defined as

$$p(b'|b, a) = \textstyle\sum_{z \in Z} p(b'|b, a, z) P(z|b, a)$$

where

$$p(b'|b, a, z) = \begin{cases} 1 & \text{if } \tau(b, a, z) = b', \\ 0 & \text{otherwise,} \end{cases}$$

are also atomic because the probability mass of the transition function is concentrated at a finite number of points $\tau(b, a, z_1), \ldots, \tau(b, a, z_{|Z|})$, *i.e.*, their cumulative distribution functions are not necessarily continuous. Therefore, the existence result of deterministic optimal policies for uncountable state CMDPs cannot be directly applied to CPOMDPs.

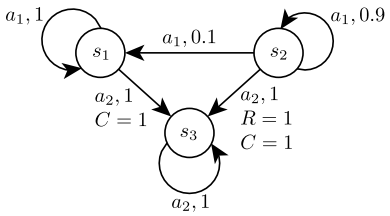

Figure 1: State transition diagram for the counter example. The edges are labeled with actions, followed by the corresponding transition probabilities, immediate rewards and costs.

We can construct examples of CPOMDPs where deterministic policies are suboptimal. The simplest case is the degenerate CPOMDP with perfectly observable states. It is equivalent to a finite-state CMDP which may not have any optimal policy that is deterministic. While the degenerate CPOMDP has a finite number of reachable belief states, the following example has infinitely many reachable beliefs.

Consider a CPOMDP with $S = \{s_1, s_2, s_3\}$, $A = \{a_1, a_2\}$, $Z = \{z\}$. The reward function is defined by assigning 1 for performing a_2 in s_2 and zero for all other cases. The cost function is defined by assigning 1 for performing a_2 in s_1 or s_2, and zero for all other cases. The transition probabilities are shown in Fig. 1, where action a_2 leads to the absorbing state s_3. Since there is only one observation, the agent cannot exactly figure out the current state. Therefore, given the initial belief $b_0 = [0, 1, 0]$, the set of reachable beliefs is $\{b_t = [1 - 0.9^t, 0.9^t, 0]\}_{t=0}^\infty$. Note that b_t is reached only at time step t, and that the agent has only one chance of receiving a non-zero reward $R(b_t, a_2) = 0.9^t$ by executing a_2 while this will incur a cost of 1.

Suppose $\gamma < \hat{c} < 1$. A deterministic policy cannot execute action a_2 earlier than $t = 1$ because executing it at $t = 0$ will violate the cumulative cost constraint. Hence, the maximum value achievable by a deterministic policy is 0.9γ with the

cumulative cost of γ. However, consider a randomized policy that executes action a_2 with probability $\hat{c}$ at $t = 0$, and then, if action a_1 was executed at $t = 0$, always executes action a_1 at $t \geq 1$. This policy achieves the value of $1 \cdot \hat{c} + 0 \cdot (1 - \hat{c}) = \hat{c} > 0.9\gamma$ with the cumulative cost of exactly $\hat{c}$.

3 Dynamic Programming for CMDP

In this section, we briefly review the dynamic programming (DP) approach for CMDPs proposed in [Piunovskiy and Mao, 2000] which provides the inspiration for our point-based DP method for CPOMDPs. The main idea of this approach is to build a new MDP model by including the cumulative cost corresponding to the constraints in the state, and assigning an immediate reward of negative infinity to the states which violate the cumulative cost constraints. For the sake of presentation, we assume only one constraint ($K = 1$). The state in the new MDP model is the pair (b_t, W_t) where b_t is the probability distribution on S at time step t and W_t is the expected cumulative cost up to time step $t-1$. The randomized action $\tilde{a}$ is defined as the probability distribution on $S \times A$. If the state is (b, W), then only those action $\tilde{a}$ are available for which the projection on S coincides with b:

$$\tilde{A}(b) = \{\tilde{a} \in \tilde{A} | \forall a, \textstyle\sum_s \tilde{a}(s, a) = \sum_s b(s)\}.$$

The transition function of the new model is defined as:

$$\begin{aligned} b_t(s') &= \textstyle\sum_{s,a} T(s, a, s') \tilde{a}_{t-1}(s, a) \\ W_t &= \textstyle\sum_{\tau=0}^{t-1} \gamma^\tau \sum_{s,a} C(s, a) \tilde{a}_\tau(s, a) \\ &= W_{t-1} + \gamma^{t-1} \textstyle\sum_{s,a} C(s, a) \tilde{a}_{t-1}(s, a) \quad t = 1, 2, \ldots \end{aligned} \tag{3}$$

where $\tilde{a}_t$ is the action executed at the time step t and $W_0 = 0$. Note that the transition function for the expected cumulative cost is not homogeneous. The reward function is defined as:

$$\tilde{R}(b, W, \tilde{a}) = \begin{cases} \sum_{s,a} R(s, a) \tilde{a}(s, a) & \text{if } W \leq \hat{c}, \\ -\infty & \text{otherwise.} \end{cases}$$

Finally, the Bellman equation is now given as:

$$V_t(b_t, W_t) = \max_{\tilde{a}_t \in \tilde{A}(b_t)} \tilde{R}(b_t, W_t, \tilde{a}_t) + \gamma V_{t+1}(b_{t+1}, W_{t+1})$$

where b_{t+1} and W_{t+1} are given as in Eqn. 3.

To make the transition function homogeneous, we introduce a new variable d_t, which is referred to as *admissible cost*, representing the expected cumulative cost that can be additionally incurred for the remaining time steps $\{t, t+1, \ldots\}$ without violating the cumulative cost constraint. Hence, d_t can be defined as $d_t = \frac{1}{\gamma^t}(\hat{c} - W_t)$. In other words, d_t is the difference between $\hat{c}$ (the maximum expected cumulative cost allowed) and W_t (the cumulative cost incurred so far) rescaled by $1/\gamma^t$. The transition function of the admissible cost can be defined by the following equation:

$$\begin{aligned} d_t &= \tfrac{1}{\gamma^t}(\hat{c} - W_t) \\ &= \tfrac{1}{\gamma^t}(\hat{c} - W_{t-1} - \gamma^{t-1} \textstyle\sum_{s,a} C(s, a) \tilde{a}_{t-1}(s, a)) \\ &= \tfrac{1}{\gamma}(d_{t-1} - \textstyle\sum_{s,a} C(s, a) \tilde{a}_{t-1}(s, a)). \end{aligned}$$

The initial admissible cost is $d_0 = \hat{c}$. Using the notion of admissible costs, the reward function and the Bellman equation can be rewritten as:

$$\tilde{R}(b, d, \tilde{a}) = \begin{cases} \sum_{s,a} R(s, a) \tilde{a}(s, a) & \text{if } d \geq 0, \\ -\infty & \text{otherwise,} \end{cases}$$

$$V(b_t, d_t) = \max_{\tilde{a}_t \in \tilde{A}(b_t)} \tilde{R}(b_t, d_t, \tilde{a}_t) + \gamma V(b_{t+1}, d_{t+1}).$$

4 Exact Dynamic Programming for CPOMDP

In [Isom *et al.*, 2008], a DP method was proposed to find *deterministic* policies in CPOMDPs. We briefly review the method in order to present our contribution.

For the sake of presentation, we shall refer to the value function of a CPOMDP as the joint function of cumulative reward and cost functions and assume only one constraint ($K = 1$) unless explicitly stated otherwise. The DP method by Isom *et al.* [2008] constructs the set of α-vector pairs for the value function, one for the cumulative reward function and the other for the cumulative cost function. Therefore, for each pair of vectors $\langle \alpha'_{i,r}, \alpha'_{i,c} \rangle$ in the value function V', the DP update should compute:[1]

$$\begin{aligned} \alpha^{a,z}_{i,r}(s) &= \tfrac{R(s,a)}{|Z|} + \gamma \textstyle\sum_{s' \in S} T(s, a, s') O(s', a, z) \alpha'_{i,r}(s') \\ \alpha^{a,z}_{i,c}(s) &= \tfrac{C(s,a)}{|Z|} + \gamma \textstyle\sum_{s' \in S} T(s, a, s') O(s', a, z) \alpha'_{i,c}(s') \\ V &= \cup_{a \in A} \oplus_{z \in Z} \{\langle \alpha^{a,z}_{i,r}, \alpha^{a,z}_{i,c} \rangle | \forall i\}, \end{aligned}$$

which in the worst case will generate $|A||V'|^{|Z|}$ pairs of vectors. To mitigate the combinatorial explosion, the method uses incremental pruning [Cassandra *et al.*, 1997] which interleaves pruning useless vectors with generating α-vectors for each action and observation. For the pruning step, we determine whether to include the newly created pair of vectors $\langle \alpha_r, \alpha_c \rangle$ in the value function using the following mixed integer linear program (MILP):

$$\max_{h, b, d^i} h \;\Bigg|\; \begin{aligned} &\alpha_c \cdot b \leq \hat{c}, \\ &(\alpha_r - \alpha_{i,r}) \cdot b \geq h - d^i M \quad \forall i, \\ &\alpha_{i,c} \cdot b \geq d^i \hat{c} \quad \forall i, \\ &d^i \in \{0, 1\} \quad \forall i, \\ &\textstyle\sum_{s \in S} b(s) = 1, \\ &b(s) \geq 0 \quad \forall s \in S, \\ &h \geq 0, \end{aligned} \tag{4}$$

where $\langle \alpha_{i,r}, \alpha_{i,c} \rangle$ is the i-th vector pair in the value function V, and M is a sufficiently large positive constant. $\langle \alpha_r, \alpha_c \rangle$ should be included if there exists a belief b where it is useful to represent the value function. For $\langle \alpha_r, \alpha_c \rangle$ to be useful at b, α_c should satisfy the cumulative cost constraint as stated in the first constraint in Eqn. 4, and α_r should have a higher cumulative reward than any other $\langle \alpha_{i,r}, \alpha_{i,c} \rangle \in V$ that satisfies the cumulative cost constraint $\alpha_{i,c} \cdot b \leq \hat{c}$ as stated in

[1] the cross-sum operator $\oplus$ is defined as $A \oplus B = \{a + b | a \in A, b \in B\}$ with the summation of pairs as $\langle a_1, a_2 \rangle + \langle b_1, b_2 \rangle = \langle a_1 + b_1, a_2 + b_2 \rangle$.

the second and third constraints. A variable $d^i \in \{0, 1\}$ indicates whether $\alpha_{i,c}$ violates the cumulative cost constraint at b. If $d^i = 1$, the third constraint $\alpha_{i,c} \cdot b \geq \hat{c}$ indicates that $\alpha_{i,c}$ violates the cumulative cost constraint at b and the second constraint is trivially satisfied. If $d^i = 0$, α_r must have a higher cumulative reward than $\alpha_{i,r}$ by satisfying the second constraint. If this program is feasible, we have found a belief where $\langle \alpha_r, \alpha_c \rangle$ is useful, hence the newly created vector will not be pruned.

However, this pruning algorithm has a number of issues. First, as described in the previous section, deterministic policies can be suboptimal in CPOMDPs. Hence, we have to consider randomized policies which involves taking a convex combination of α-vectors when checking for dominance. Second, the method will prune away every vector that violates the cumulative cost constraint in each DP update. This may lead to a suboptimal deterministic policy since it effectively ensures that *every* intermediate t-step policy should satisfy the cumulative cost constraint; satisfying the long-term cumulative cost constraint in Eqn. 1 does not necessarily mean that the constraint should be satisfied at every time step.

We therefore revise the MILP to the following minimax quadratically constrained program (QCP):

$$\min_{b} \max_{w_i, h} h \left| \begin{array}{l} \alpha_c \cdot b \geq b \cdot \sum_i w_i \alpha_{i,c} + h, \\ b \cdot \sum_i w_i \alpha_{i,r} - \alpha_r \cdot b \geq h, \\ \sum_{s \in S} b(s) = 1, \\ b(s) \geq 0 \quad \forall s \in S, \\ \sum_i w_i = 1, \\ w_i \geq 0 \quad \forall i. \end{array} \right. \tag{5}$$

The first and second constraints state that, if $h \geq 0$, there exists a convex combination of vectors which incurs less cumulative cost than α_c while achieving a higher cumulative reward than α_r at belief b. Hence, if we obtain a nonnegative h by maximizing it for belief b, then $\langle \alpha_r, \alpha_c \rangle$ is not useful at belief b because we have a better or equally performing randomized policy. Since we minimize the maximum h over the entire belief simplex, if the final solution h is nonnegative, then $\langle \alpha_r, \alpha_c \rangle$ is not useful at any belief b, and thus it can be pruned. Unfortunately, this minimax QCP is computationally demanding to solve, and thus we propose a point-based approximate method for CPOMDPs in the next section.

5 Approximate Method for CPOMDP

5.1 Point-based value iteration

The point-based value iteration (PBVI) algorithm [Pineau *et al.*, 2006] for the standard POMDP uses a finite set of reachable beliefs $B = \{b_0, b_1, \ldots, b_q\}$, instead of the entire belief simplex, for planning. Performing DP updates only at the beliefs $b \in B$ eliminates the need to solve linear programs (LPs) for pruning α-vectors in standard POMDPs.

We can adapt the point-based DP update to CPOMDPs in a simple way. For each $b \in B$, we enumerate the regressions[2] of α-vectors using Alg. 1. We then prune the dominated vectors using only $b \in B$. The complete point-based DP update is shown in Alg. 2. Since pruning is confined to B, we check the dominance of α-vectors for each $b \in B$, thereby reducing the minimax QCP in Eqn. 5 to the following LP:

$$\max_{w_i, h} h \left| \begin{array}{l} \alpha_c \cdot b \geq b \cdot \sum_i w_i \alpha_{i,c} + h, \\ b \cdot \sum_i w_i \alpha_{i,r} - \alpha_r \cdot b \geq h, \\ \sum_i w_i = 1, \\ w_i \geq 0 \quad \forall i, \end{array} \right. \tag{6}$$

which has the same formulation as the maximization problem in Eqn. 5 except that b is no longer a variable. The pruning algorithm using the above LP is shown in Alg. 3.

Although this algorithm is based on the enumeration algorithm for standard POMDPs [Monahan, 1982], we can easily modify it to perform incremental pruning. However, compared to the standard PBVI which maintains only one α-vector at each belief, this simple point-based algorithm still suffers from the potential combinatorial explosion in the number of α-vectors. This is mainly because, although we have collected the finite set of reachable beliefs, we have *not* collected any information on how much cost can be incurred while still satisfying the cumulative cost constraint at those beliefs.

5.2 PBVI with admissible cost

The main idea behind our approximate algorithm is to additionally obtain information on the cumulative cost for each collected belief. We define the cumulative cost and the admissible cost for the belief-action history as in [Piunovskiy and Mao, 2000], *i.e.*, $W_t = \sum_{\tau=0}^{t-1} \gamma^\tau C(b_\tau, a_\tau)$ and the ad-

Algorithm 1: regress V'

input : V'
output: $\{\langle \alpha_r^{a,*}, \alpha_c^{a,*} \rangle\}, \{\Gamma^{a,z}\}$
foreach $a \in A$ *and* $z \in Z$ **do**
  $\langle \alpha_r^{a,*}, \alpha_c^{a,*} \rangle \leftarrow \langle R(\cdot, a), C(\cdot, a) \rangle$
  $\Gamma^{a,z} \leftarrow \emptyset$
  foreach $\langle \alpha'_{i,r}, \alpha'_{i,c} \rangle \in V'$ **do**
    $\alpha_{i,r}^{a,z}(s) = \sum_{s' \in S} T(s, a, s') O(s', a, z) \alpha'_{i,r}(s')$
    $\alpha_{i,c}^{a,z}(s) = \sum_{s' \in S} T(s, a, s') O(s', a, z) \alpha'_{i,c}(s')$
    $\Gamma^{a,z} \leftarrow \Gamma^{a,z} \cup \langle \alpha_{i,r}^{a,z}, \alpha_{i,c}^{a,z} \rangle$

Algorithm 2: update $V = \tilde{H} V'$

input : B, V'
output: V
$V \leftarrow \emptyset$
$\{\langle \alpha_r^{a,*}, \alpha_c^{a,*} \rangle\}, \{\Gamma^{a,z}\} \leftarrow$ `regress(`V'`)`
foreach $a \in A$ **do**
  $\Gamma^a \leftarrow \langle \alpha_r^{a,*}, \alpha_c^{a,*} \rangle \oplus \bigoplus_{z \in Z} \gamma \Gamma^{a,z}$
$\Gamma \leftarrow \bigcup_{a \in A} \Gamma^a$
foreach $b \in B$ **do**
  $V \leftarrow V \cup$ `prune(`b, Γ`)`

[2] Regression refers to the multiplication of an α-vector by the dynamics of an action-observation pair.

Algorithm 3: prune

input : b, Γ
output: $\tilde{\Gamma}$
$\tilde{\Gamma} \leftarrow \emptyset$
foreach $\langle \alpha_r, \alpha_c \rangle \in \Gamma$ **do**
  Solve the LP (Eqn. 6) and get the solution h
  if $h < 0$ **then**
    $\tilde{\Gamma} \leftarrow \tilde{\Gamma} \cup \{\langle \alpha_r, \alpha_c \rangle\}$

missible cost is recursively defined as:

$$d_{t+1} = \frac{1}{\gamma}(d_t - C(b_t, a_t)). \tag{7}$$

Note that we use the expected cost $C(b_t, a_t)$ instead of the actually incurred cost when updating the admissible cost. This is because the policies in CPOMDPs are not defined to be contingent on actual costs, in the same way as the policies in standard POMDPs are not contingent on the received rewards. If we extend the definition of policies to be contingent on actual costs, we can adopt some of the approaches in MDP such as incorporating actual costs into the state space [Meuleau *et al.*, 2009], or using the sample path constraints [Ross and Varadarajan, 1989; 1991].

In order to use the notion of admissible cost in PBVI, we first sample pairs of beliefs and admissible costs, $B = \{(b_0, d_0), (b_1, d_1), \ldots, (b_q, d_q)\}$. For a belief-cost pair (b, d), the best action is obtained by solving the following LP:

$$\max_{w_i} b \cdot \sum_i w_i \alpha_{i,r} \left| \begin{array}{l} b \cdot \sum_i w_i \alpha_{i,c} \leq d, \\ \sum_i w_i = 1, \\ w_i \geq 0 \quad \forall i, \end{array} \right. \tag{8}$$

where the resulting coefficient w_i represents the probability of choosing the action corresponding to $\langle \alpha_{i,r}, \alpha_{i,c} \rangle$. Note that there exists a solution w_i with at most two non-zero components because the above LP contains $|V| + 2$ constraints and at least $|V|$ constraints must be active at extreme points. For CPOMDPs with K constraints, there always exists a solution that will have at most $K + 1$ non-zero components.[3]

The revised point-based DP update is described in Alg. 4. For each belief-cost point $(b, d) \in B$, we construct:

$$\alpha_r^{(b,d),a} = \alpha_r^{a,*} + \gamma \textstyle\sum_{z \in Z} \tilde{\alpha}_r^{a,z}$$

where $\tilde{\alpha}_r^{a,z}$ is the best convex combination of the value vectors with respect to the next belief and admissible cost, obtained by the LP in Eqn. 8. $\alpha_c^{(b,d),a}$ is obtained as the by-product of computing $\alpha_r^{(b,d),a}$. Finally, we once again use the LP in Eqn. 8 to find the best convex combination of the value vectors with respect to the current belief and admissible cost.

Note that each $\tilde{\alpha}_r^{a,z}$ is computed by distributing the admissible cost via $\frac{1}{\gamma}(d - C(b, a))P(z|b, a)$. Ideally, we should not impose such a constraint on each observation to obtain the best convex combination at b. However, this will lead to a local combinatorial explosion due to cross-summations, and we observed that distributing the admissible cost yielded sufficiently good α-vectors while ensuring that the admissible cost constraint is satisfied at b.

[3]We can guarantee at most $(K + 1)$ vector pairs if we use LP solver that always returns an extreme point, e.g., simplex method.

Algorithm 4: update $V = \tilde{H}V'$ (PBVI with admissible cost)

input : B, V'
output: V
$V \leftarrow \emptyset$
$\{\langle \alpha_r^{a,*}, \alpha_c^{a,*} \rangle\}, \{\Gamma^{a,z}\} \leftarrow$ `regress(`V'`)`
foreach $(b, d) \in B$ **do**
  foreach $a \in A$ **do**
    foreach $z \in Z$ **do**
      $d_z \leftarrow \frac{1}{\gamma}(d - C(b, a))P(z|b, a)$
      Solve the LP (Eqn. 8) with $\forall \langle \alpha_{i,r}, \alpha_{i,c} \rangle \in \Gamma^{a,z}$ and (b, d_z), and get the solution $\tilde{w}_i$.
      $\tilde{\alpha}_r^{a,z} \leftarrow \sum_i \tilde{w}_i \alpha_{i,r}$
      $\tilde{\alpha}_c^{a,z} \leftarrow \sum_i \tilde{w}_i \alpha_{i,c}$
    $\alpha_r^{(b,d),a} = \alpha_r^{a,*} + \gamma \sum_{z \in Z} \tilde{\alpha}_r^{a,z}$
    $\alpha_c^{(b,d),a} = \alpha_c^{a,*} + \gamma \sum_{z \in Z} \tilde{\alpha}_c^{a,z}$
  $\Gamma^{(b,d)} \leftarrow \bigcup_{a \in A} \{\langle \alpha_r^{(b,d),a}, \alpha_c^{(b,d),a} \rangle\}$
  Solve the LP (Eqn. 8) with $\Gamma^{(b,d)}$ and (b, d), and get the solution w_i.
  $V \leftarrow V \cup \{\langle \alpha_{i,r}, \alpha_{i,c} \rangle \in \Gamma^{(b,d)} | w_i > 0\}$

In summary, the algorithm does not depend on cross-summations and maintains at most $(K + 1)$ vector pairs for each belief, hence a total of at most $(K + 1)|B|$ vector pairs.

5.3 Policy execution

In the execution phase, the agent chooses its action with respect to the current belief and admissible cost. The overall procedure for the execution phase is shown in Alg. 5. Specifically, at time step t, the optimal randomized action is calculated by solving the LP in Eqn. 8 with (b_t, d_t), and obtaining the solution w_i. The agent selects a vector pair $\langle \alpha_{i,r}, \alpha_{i,c} \rangle$ by randomly choosing the index i with probability w_i, and then the current admissible cost d_t is *reset* to $d'_t = \alpha_{i,c} \cdot b$ since the agent decides to follow the policy corresponding to $\langle \alpha_{i,r}, \alpha_{i,c} \rangle$ which incurs the expected cumulative cost of $\alpha_{i,c} \cdot b$ for the remaining steps. The new admissible cost d'_t can be higher or lower than the original admissible cost d_t, but they will be the same in the expectation since actions are chosen randomly according to the w_i satisfying $d_t = \sum_i w_i(\alpha_{i,c} \cdot b_t)$. After executing the action associated with $\langle \alpha_{i,r}, \alpha_{i,c} \rangle$, the next admissible cost d_{t+1} is then calculated by Eqn. 7 and the next belief is computed by Eqn. 2 with the observation z_t from the environment.

6 Experiments

6.1 Randomized vs. deterministic policies

We first experimentally confirm that deterministic policies cannot represent optimal policies of CPOMDPs using the toy

Algorithm 5: Execution

input : $b = b_0$, $d = \hat{c}$
while *true* **do**
  Solve the LP (Eqn. 8) with (b, d)
  Randomly choose the index i with probability w_i
  Perform the action a_i corresponding to $\langle \alpha_{i,r}, \alpha_{i,c} \rangle$
  Receive observation z from the environment
  $d \leftarrow \alpha_{i,c} \cdot b$
  $d \leftarrow \frac{1}{\gamma}(d - C(b, a_i))$
  $b \leftarrow \tau(b, a, z)$

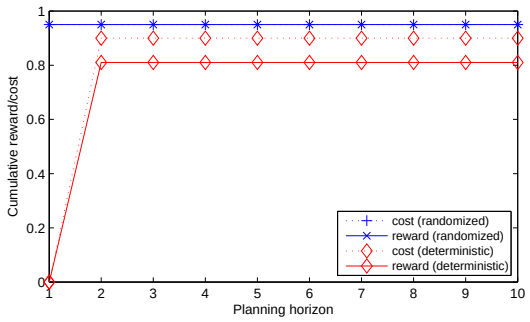

Figure 2: Results for the toy problem ($\hat{c} = 0.95$, $\gamma = 0.9$).

problem in Fig. 1. Fig. 2 shows the cumulative reward and cost of the deterministic and randomized policies obtained by the algorithms. As demonstrated in Sec. 2.2, the deterministic policy was suboptimal since it had to execute a_1 at $t = 0$ and a_2 at $t = 1$ in order to satisfy the cumulative cost constraint. Hence, the deterministic policy achieved the value of 0.9γ with the cumulative cost of γ. The randomized policy achieved the value of $\hat{c}$ while exactly satisfying the cumulative cost constraint at $\hat{c}$.

6.2 Quickest change detection

We compare the policies found by the exact and the approximate methods for CPOMDPs in the Quickest Change Detection (QCD) problem [Isom *et al.*, 2008]. The problem has 3 states consisting of PreChange, PostChange, and PostAlarm. The agent has to alarm as soon as possible after the state changes to PostChange, while bounding the probability of false alarm, *i.e.*, executing the alarm action when the state is PreChange. We use the discounted version of the problem with $\gamma = 0.95$, and set the false alarm probability constraint to $\hat{c} = 0.2$.

Fig. 3 compares the results of the exact and approximate methods for the discounted QCD problem. Due to the complexity of the MILP in Eqn. 4 and the minimax QCP in Eqn. 5, the exact methods using MILP and QCP pruning were not able to perform DP updates more than 6 and 5 time steps, respectively. The approximate method used 500 belief-cost sample pairs, and it was able to perform DP updates more than 10 time steps without difficulty. Furthermore, the policy from the approximate method performed close to the one from the exact method. Fig. 4 compares the planning time for

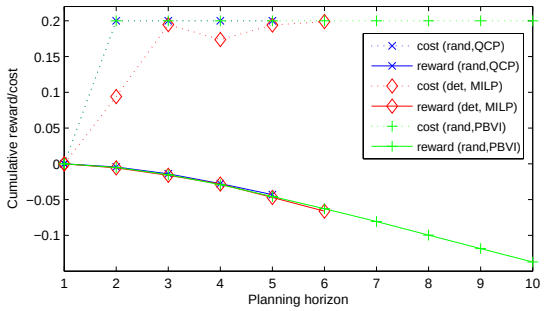

Figure 3: Results of the exact and approx. algorithms (PBVI with admissible cost) for the discounted QCD problem.

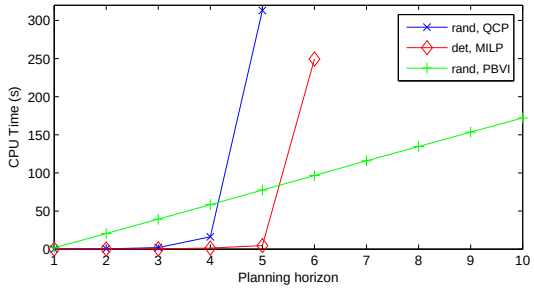

Figure 4: Planning time for the discounted QCD problem.

each method.[4] Note that the exact method takes large amount of time due to the combinatorial explosion in the number of α-vectors even though the useless ones are pruned, whereas the approximate method exhibits its running time linear in the planning horizon.

6.3 n-city ticketing

In order to demonstrate the usefulness of the CPOMDP formulation in spoken dialogue systems and the scalability of our approximate method, we show experimental results on the n-city ticketing problem [Williams *et al.*, 2005]. The problem models the dialogue manager agent which interacts with the user to figure out the origin and the destination among n cities for flight reservation. At each time step, the agent asks the user for the information about the origin and/or the destination, and submit the ticket purchase request once it has gathered sufficient information. However, due to the speech recognition errors, the observed user's response can be different from the true response. We denote the probability of speech recognition error as P_e which is incorporated into the observation probability of the model. We used a constant reward function of -1 for each time step till the terminal submit action, and a constant cost function of 1 for issuing a ticket with wrong origin or destination and 0 otherwise. Hence, the reward part represents the efficiency (dialogue length) and the cost part represents the accuracy (task completion rate).

[4] All the experiments were done on a Linux platform with the Intel Xeon 2.66GHz CPU and 32GB memory. All the algorithms were implemented in Matlab; MILPs and LPs were solved using CPLEX 12.1; minimax QCPs were solved using `fmincon` in Matlab for the outer minimization and CPLEX 12.1 for the inner maximization.

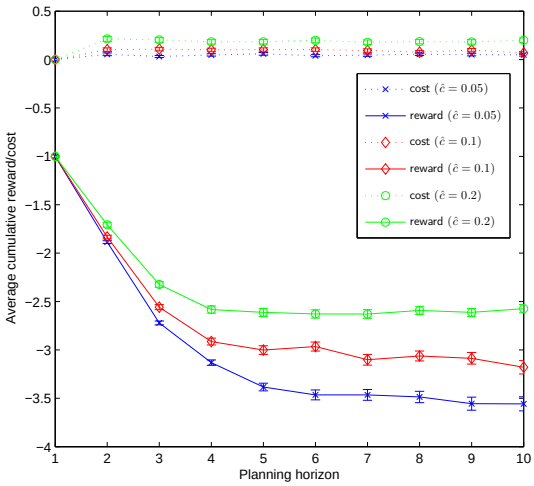

Figure 5: Results of PBVI with admissible cost for 3-City Ticketing problem ($P_e = 0.2$). Vertical axis represents the average cumulative reward/cost over 1000 simulations. Error bars represent 95% confidence intervals.

Fig. 5 shows the results from the approximate method using 50 belief-cost sample pairs for $n = 3$ with $P_e = 0.2$. This problem has $|S| = 1945$, $|A| = 16$, $|Z| = 18$, and $\gamma = 0.95$. Note that the policy uses more dialogue turns for smaller $\hat{c}$, since it needs more information gathering steps to be more accurate about the origin and the destination.

7 Conclusion

We showed that optimal policies in CPOMDPs can be randomized, and presented exact and approximate methods for finding randomized policies for CPOMDPs. Experimental results show that randomized optimal policies are better then deterministic ones, and our point-based method efficiently finds approximate solutions. Although we demonstrated CPOMDPs with one constraint, our algorithms naturally extend to multiple constraints and different discount factors for each reward or cost function.

Careful readers may note that the policy from our point-based method can violate the cost constraints because the cumulative cost function is constructed using the sampled beliefs in the same way PBVI approximates the value function. If such violation is a serious issue, we can use upper bound approximation techniques [Hauskrecht, 2000] for representing the cumulative cost functions to absolutely guarantee satisfying the cost constraints. However, in our experiments, using the lower bound representation (α-vectors) yielded fairly good results.

There are several future works worth pursuing. First, the proposed method can benefit from adopting state-of-the-art POMDP solvers with heuristic belief exploration. Second, it would be interesting to extend this approach to average reward and cost criterion models, since a lot of well-established measures are defined using such criterion in practice. Lastly, it is an open question whether we can extend this approach to factored representation for CPOMDPs.

Acknowledgments

This work was supported by the National Research Foundation (NRF) of Korea grant 2009-0069702 and by the Agency for Defense Development (ADD) of Korea grant 09-01-03-04.

References

[Altman, 1999] E. Altman. *Constrained Markov Decision Processes*. Chapman & Hall/CRC, 1999.

[Cassandra *et al.*, 1997] A. Cassandra, M. Littman, and N. Zhang. Incremental pruning: A simple, fast, exact method for partially observable Markov decision processes. In *Proceedings of UAI*, 1997.

[Feinberg and Piunovskiy, 2002] E. A. Feinberg and A. B. Piunovskiy. Nonatomic total rewards Markov decision processes with multiple criteria. *Journal of Mathematical Analysis and Applications*, 273(1):93–111, 2002.

[Hauskrecht, 2000] M. Hauskrecht. Value-function approximations for partially observable Markov decision processes. *Journal of Artificial Intelligence Research*, 13:33–94, 2000.

[Isom *et al.*, 2008] J. D. Isom, S. P. Meyn, and R. D. Braatz. Piecewise linear dynamic programming for constrained POMDPs. In *Proceedings of AAAI*, 2008.

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

[Meuleau *et al.*, 2009] N. Meuleau, E. Benazera, R. I. Brafman, E. A. Hansen, and Mausam. A heuristic search approach to planning with continuous resources in stochastic domains. *Journal of Artificial Intelligence Research*, 34:27–59, 2009.

[Monahan, 1982] G. E. Monahan. A survey of partially observable Markov decision-processes: Theory, models, and algorithms. *Management Science*, 28(1):1–16, 1982.

[Pineau *et al.*, 2006] J. Pineau, G. Gordon, and S. Thrun. Anytime point-based approximations for large POMDPs. *Journal of Artificial Intelligence Research*, 27:335–380, 2006.

[Piunovskiy and Mao, 2000] A. B. Piunovskiy and X. Mao. Constrained Markovian decision processes: the dynamic programming approach. *Operations Research Letters*, 27(3):119–126, 2000.

[Ross and Varadarajan, 1989] K. W. Ross and R. Varadarajan. Markov decision-processes with sample path constraints - the communicating case. *Operations Research*, 37(5):780–790, 1989.

[Ross and Varadarajan, 1991] K. W. Ross and R. Varadarajan. Multichain Markov decision-processes with a sample path constraint - a decomposition approach. *Mathematics of Operations Research*, 16(1):195–207, 1991.

[Williams and Young, 2007] J. D. Williams and S. Young. Partially observable Markov decision processes for spoken dialog systems. *Computer Speech and Language*, 21(2):393–422, 2007.

[Williams *et al.*, 2005] J. Williams, P. Poupart, and S. Young. Factored partially observable Markov decision processes for dialogue management. In *Proceedings of IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, 2005.

[Zhao *et al.*, 2007] Q. Zhao, L. Tong, A. Swami, and Y. X. Chen. Decentralized cognitive MAC for opportunistic spectrum access in ad hoc networks: A POMDP framework. *IEEE Journal on Selected Areas in Communications*, 25(3):589–600, 2007.