Approximability of Constant-horizon Constrained POMDP

Majid Khonji[1,2*], **Ashkan Jasour**[2] and **Brian Williams**[2]
[1]EECS Department, KU Center for Autonomous Robotic Systems, Khalifa University, Abu Dhabi, UAE
[2]Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA
majid.khonji@ku.ac.ae, {mkhonji, jasour, williams}@mit.edu

Abstract

Partially Observable Markov Decision Process (POMDP) is a fundamental framework for planning and decision making under uncertainty. POMDP is known to be intractable to solve or even approximate when the planning horizon is long (i.e., within a polynomial number of time steps). Constrained POMDP (C-POMDP) allows constraints to be specified on some aspects of the policy in addition to the objective function. When the constraints involve bounding the probability of failure, the problem is called Chance-Constrained POMDP (CC-POMDP). Our first contribution is a reduction from CC-POMDP to C-POMDP and a novel Integer Linear Programming (ILP) formulation. Thus, any algorithm for the later problem can be utilized to solve any instance of the former. Second, we show that unlike POMDP, when the length of the planning horizon is constant, (C)C-POMDP is NP-Hard. Third, we present the first Fully Polynomial Time Approximation Scheme (FPTAS) that computes (near) optimal deterministic policies for constant-horizon (C)C-POMDP in polynomial time.

1 Introduction

A fundamental area in artificial intelligence involves planning under uncertainty. *Partially observable markov decision process* (POMDP) provides a model for optimal planning under actuator and sensor uncertainty, where the goal is to find policies that maximize some measure of expected utility [Kaelbling *et al.*, 1998; Sondik, 1971]. POMDP model is quite general and used in many areas such as reinforcement learning [Kaelbling *et al.*, 1996] and robotics [Kress-Gazit *et al.*, 2009]. The model extends a well-researched framework of *markov decision process* (MDP) to scenarios where an agent cannot accurately observe the underlying state of the environment [Howard, 1960; Puterman, 2014]. In many real-world applications, a single measure of performance is not sufficient to capture all requirements (e.g., a battery-operated unmanned aerial vehicle (UAV) tasked to maximize its surveillance coverage while keeping energy consumption below a given threshold). An extension, often called *constrained* POMDP (C-POMDP), encodes additional constraints on the system to capture those requirements [Poupart *et al.*, 2015]. Another approach is to bound the probability of constraint violation, modeled as *chance-constrained* POMDP (CC-POMDP) [Santana *et al.*, 2016]. Unsurprisingly, modeling constraints as negative rewards in the objective function leads to models that are over-sensitive to the particular value chosen, and to policies that are overly risk-taking or overly risk-averse [Undurti and How, 2010].

The focus in the literature has been primarily on the fully observable *constrained* MDP (C-MDP), for which non-trivial theoretical results [Altman, 1999; Feinberg and Shwartz, 1996] and efficient algorithms are known (e.g., [Trevizan *et al.*, 2016], [Ono *et al.*, 2012]). On the other hand, the state of the art for (C)C-POMDP is less mature. Despite recent algorithmic developments, there has been relatively little effort devoted to the theoretical aspects of (C)C-POMDP. Current methods span from extensions of dynamic programming [Isom *et al.*, 2008], point-based value iteration [Kim *et al.*, 2011], approximate linear programming [Poupart *et al.*, 2015], on-line search [Undurti and How, 2010], to heuristic forward search for CC-POMDP [Santana *et al.*, 2016]. These methods generally compromise on either the optimality or the feasibility of the problem, in which they may obtain policies that are sub-optimal or in some cases violate the constraints. Another issue is that some of these methods produce randomized policies [Poupart *et al.*, 2015]. Arguably, users rarely trust stochasticity in decision making in safety-critical systems [Dolgov and Durfee, 2005; Santana *et al.*, 2016].

In general, POMDP is known to be NP-Hard [Papadimitriou and Tsitsiklis, 1987]. A stronger hardness result states that there is no α-approximation for POMDP unless P=NP [Lusena *et al.*, 2001], where α characterizes the worst-case ratio between the approximate solution and an optimal solution. In this context, α can be any arbitrary value with a polynomial number of bits (hence exponentially small). The hardness result assumes the planning horizon has a polynomial number of time steps. If we assume a constant length, POMDP can be easily solved in polynomial time using a standard search algorithm [Russell and Norvig, 2016].

This work provides the first approximability study for

*Contact Author

(C)C-POMDP. We show that when the planning horizon is constant, unlike POMDP, (C)C-POMDP is still NP-hard even when the environment is fully observable. Despite the hardness, we show that one can obtain close-to-optimal solutions in polynomial time within a user-defined accuracy parameter ϵ. It is worth noting that in many practical applications (e.g., self-driving vehicle), the planning horizon is reasonably small (within 10 seconds). Thus, a better theoretical understanding of the *constant-horizon* case is both relevant to practice and paves the way for future development of fast heuristics without compromising optimality.

The paper is structured as follows. Sec. 2 presents definitions and relevant background. Sec. 3 provides a novel *integer linear programming* (ILP) formulation and a reduction from CC-POMDP into C-POMDP. Sec. 4 presents our hardness result, namely, (C)C-(PO)MDP is NP-Hard even when planning horizon is two. In Sec. 5, we provide a novel *fully polynomial time approximation scheme* FPTAS for the two problems, which is the best possible algorithm in theory in terms of approximation ratio.

2 Problem Definition

We consider a series of discretized time steps $\mathcal{T} = \{1, \ldots, h\}$ with a fixed interval. Through out this work, we assume h is a constant. By this assumption, we do not require our algorithm to scale in h. This is motivated by the fact that there is no α-approximation for POMDP for polynomial h, unless P=NP [Lusena *et al.*, 2001].

Definition 2.1 (C-POMDP). *Constrained partially observable markov decision process problem is a tuple $M = \langle \mathcal{S}, \mathcal{A}, \mathcal{O}, T, O, U, b_0, h, U', C' \rangle$, $\mathcal{S}, \mathcal{A}$ and $\mathcal{O}$ are finite sets of discrete states, actions and observations; $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$ is a probabilistic transition function between states, i.e., $T(s,a,s') = \Pr(s' \mid a, s)$, where $s, s' \in \mathcal{S}$ and $a \in \mathcal{A}$; $O : \mathcal{O} \times \mathcal{S} \to [0,1]$ is a probabilistic observation function such that $O(o,s) = \Pr(o \mid s)$, where $o \in \mathcal{O}$ and $s \in \mathcal{S}$; $U : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_+$ is a non-negative reward function; $b_0 : \mathcal{S} \to [0,1]$ is an initial belief state, a probability distribution over the state space; h is the execution horizon; $U' : \mathcal{S} \times \mathcal{A} \to \mathbb{R}_+$ is a secondary non-negative penalty function; and $C' \in \mathbb{R}_+$ is an upper bound on the total expected penalty. The objective is to compute a conditional plan (or a policy) that maximizes the cumulative expected reward while keeping the cumulative expected constraint penalty at most C'.*

Definition 2.2 (CC-POMDP). *Chance constrained partially observable markov decision process problem is a tuple $M = \langle \mathcal{S}, \mathcal{A}, \mathcal{O}, T, O, U, b_0, h, \mathcal{R}, \Delta \rangle$, where $\mathcal{S}, \mathcal{A}, \mathcal{O}, T, O, U, b_0, h$ are defined as in Def. 2.1; $\mathcal{R} \subset \mathcal{S}$ is a subset that represents risky states (wherein the agent cannot perform any further action); and Δ is the risk budget, a threshold on the probability of failure. The objective is to compute a conditional plan that maximizes the cumulative expected reward such that the probability of ending up in risky states is at most Δ.*

In the following, we set up notation and provide formal definitions for the notions of *conditional plan* and *approximation algorithm*. We show later how to calculate expected reward, risk, and penalty in Sec. 3.

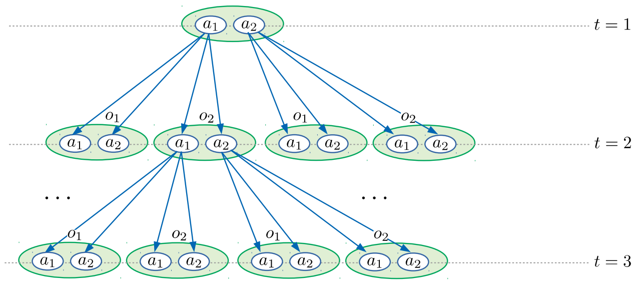

Figure 1: The precedence relationship between action sequences is represented by directed edges. Hyper-nodes, represented by shaded circles, are observation sequences $\widetilde{\mathcal{O}}$, while white circles are action sequences $\widetilde{\mathcal{A}}$, represented by the last action of the sequence (for $h = 3$, $\mathcal{A} = \{a_1, a_2\}$, and $\mathcal{O} = \{o_1, o_2\}$).

Define the set of all possible histories of length k, represented by interleaving action-observation sequences, as

$$\mathcal{Q}_k \triangleq \begin{cases} \{q = \langle a_q^1, o_q^2, a_q^3, \ldots, a_q^{k-1}, o_q^k \rangle \\ \quad \mid a_q^{2t-1} \in \mathcal{A}, o_q^{2t} \in \mathcal{O}, t \in \mathcal{T}\}, & \text{if } k \text{ is even,} \\ \{q = \langle a_q^1, o_q^2, a_q^3, \ldots, o_q^{k-1}, a_q^k \rangle \\ \quad \mid a_q^{2t-1} \in \mathcal{A}, o_q^{2t} \in \mathcal{O}, t \in \mathcal{T}\}, & \text{if } k \text{ is odd.} \end{cases}$$

Notice when k is even, a sequence $q \in \mathcal{Q}_k$ ends with an observation, whereas if k is odd, the sequence ends with an action. For convenience, we define a function $\tau : \mathbb{N}^+ \to \mathcal{T}$ that maps indices k onto time steps in $\mathcal{T}$ as follows:

$$\tau(k) \triangleq \begin{cases} \frac{k}{2} & \text{if } k \text{ is even,} \\ \frac{k+1}{2} & \text{if } k \text{ is odd.} \end{cases}$$

Let $\widetilde{\mathcal{O}} \triangleq \bigcup_{\substack{k \text{ is even,} \\ \tau(k) \in \mathcal{T}}} \mathcal{Q}_k$ be the set of all possible sequences that end with an observation. Similarly, define $\widetilde{\mathcal{A}} \triangleq \bigcup_{\substack{k \text{ is odd,} \\ \tau(k) \in \mathcal{T}}} \mathcal{Q}_k$ to be the set of all possible sequences that end with an action. For a sequence $q \in \widetilde{\mathcal{O}}$ and action $a \in \mathcal{A}$ (resp. observation $o \in \mathcal{O}$), we write qa (resp. qo) to denote the concatenation $q \| \langle a \rangle$ (resp. $q \| \langle o \rangle$). Also, for any two sequences $q, q' \in \widetilde{\mathcal{A}} \cup \widetilde{\mathcal{O}}$ we write qq' to denote $q \| q'$. Let $|q|$ be the length of sequence q. We refer to the empty sequence by 0.

Given a sequence $q \in \widetilde{\mathcal{O}} \cup \widetilde{\mathcal{A}}$, let $\mathcal{O}(q)$ and $\mathcal{A}(q)$ respectively denote the set of observation and action indices along the sequence. We write o_q^i and a_q^j to denote the i-th observation and j-th action, respectively, for $i \in \mathcal{O}(q), j \in \mathcal{A}(q)$. For convenience, we write $q - k$ to denote the sequence minus the last k elements (where an element is either an action or observation). For example, $q = a_1 o_2 a_3$, $q - 1 = a_1 o_2$, $q - 2 = a_1$, and so on. Next, define a *precedence relationship* between action sequences in $\widetilde{A}$ by a directed tree graph $\mathcal{G}^a = (\widetilde{\mathcal{A}}, \mathcal{E}^a)$ (depicted in Fig. 1) such that $(q, q') \in \mathcal{E}^a$ if and only if $q' = qoa$ for some $o \in \mathcal{O}$ and $a \in \mathcal{A}$[1].

[1]Note that $q \in \widetilde{\mathcal{A}}$ (resp. $q' \in \widetilde{\mathcal{O}}$) represents a sequence, while $a \in \mathcal{A}$ (resp. $o \in \mathcal{O}$) represents a single action (resp. observation).

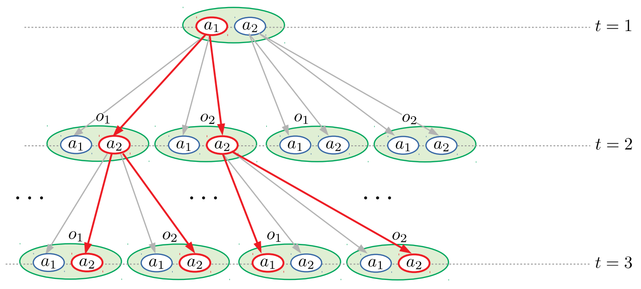

Figure 2: A conditional plan is represented by red circles over action sequences, represented by the last action. ($h=3$, $\mathcal{A}=\{a_1,a_2\}$, and $\mathcal{O}=\{o_1,o_2\}$.)

We define a binary decision variable x_q for each sequence such that $x_q = 1$ indicates the last action in q is selected, and $x_q = 0$ otherwise. We write $\mathbf{x} \triangleq (x_q)_{q\in\widetilde{\mathcal{A}}}$ to denote the vector of all decision variables.

Definition 2.3. $\mathbf{x} \in \{0,1\}^{|\widetilde{\mathcal{A}}|}$ *is a conditional plan if,*

$$\sum_{a\in\mathcal{A}} x_{qa} \le 1, \qquad \forall q \in \widetilde{\mathcal{O}} \cup \{0\}, \tag{1}$$

$$x_q \ge x_{q'}, \qquad \forall (q,q') \in \mathcal{E}^a. \tag{2}$$

Cons. (1) enforces that at most one action is selected for each observation (see the red circles in Fig. 2), while Cons. (2) maintains the precedence relationship, that is, if an action is selected then all its ancestors (defined by the precedence relationship $\mathcal{G}^a$) must be selected as well. The two constraints enforce a tree structure, also known as a *policy tree*.

We note that the definition above does not enforce a full-horizon policy tree. We deliberately highlight this definition because unlike POMDP where there always exists a full-horizon solution, such solution may not exist for (C)C-POMDP due to the restriction imposed by the constraint (see Sec. 3.3 below). In other words, the highest reward actions may consume all the risk budget before reaching the horizon. Thus, one may opt to prioritize objective optimization over policy completeness. In many practical applications, $x_q = 0$ can be interpreted as a safe maneuver. For example, an autonomous underwater glider, tasked to maximize expected science return while bounding the risk of collision [Timmons and Williams, 2015], can execute a *safe-extract* action anytime (by changing its buoyancy so that it floats to the surface for extraction).

We note that one can construct a scenario in which a full-horizon solution exists for CC-POMDP but the current state-of-the-art approach returns *infeasible* [Santana *et al.*, 2016]). Therefore we find it important to highlight this new aspect of the constrained version of POMDP that hasn't been discussed before. The results of this paper can be extended to enforce full-horizon policies. Due to the paucity of space, we omit the details and refer interested readers to the full version of the paper.

The subject of approximation algorithms is well-studied in the theoretical computer science community [Vazirani, 2013]. In the following, we define some standard terminology for approximation algorithms. Consider a maximization problem Π with non-negative objective function $f(\cdot)$, let F be a feasible solution to Π and $F^\star$ be an optimal solution to Π. $f(F)$ denotes the objective value of F. Let $\textsc{Opt} = f(F^\star)$ be the optimal objective value of $F^\star$. A common definition of approximate solutions is α-approximation, where α characterizes the approximation ratio between the approximate solution and an optimal solution.

Definition 2.4 ([Vazirani, 2013])**.** *For $\alpha \in [0,1]$, an α-approximation to maximization problem Π is an algorithm that obtains a feasible solution F for any instance such that $f(F) \ge \alpha \cdot \textsc{Opt}$.*

In particular, *polynomial-time approximation scheme* (PTAS) is a $(1-\epsilon)$-approximation algorithm to a maximization problem, for any $\epsilon > 0$. The running time of a PTAS is polynomial in the input size for every fixed ϵ, but the exponent of the polynomial might depend on $1/\epsilon$. In other words, PTAS allows to trade the approximation ratio against the running time. An even stronger notion is a *fully polynomial-time approximation scheme* (FPTAS), which requires the running time to be polynomial in both input size and $1/\epsilon$.

3 Novel Formulation for (C)C-POMDP

In this section, we present a novel ILP formulation for CC-POMDP and C-POMDP, and show the similarity between the two problems. We observe that the later is in fact more general. In other words, any instance of CC-POMDP can be reduced to an instance of C-POMDP. The reduction implies that any algorithm for C-POMDP can be also used to solve CC-POMDP. It also implies that because CC-POMDP is NP-Hard (as we show in Sec. 4), C-POMDP must be NP-Hard as well. The key idea for the new ILP formulation is that we can calculate rewards and risks (penalties for C-POMDP) independently from decision variable $\mathbf{x}$, hence we get *linear* constraints.

3.1 Calculating Reward

We show how to compute the cumulative expected reward of a conditional plan. The cumulative expected penalty for C-POMDP is obtained in a similar way, thus we omit for brevity. The reward of the last action in a sequence $q \in \widetilde{\mathcal{A}}$ is denoted by u_q. Essentially, the reward is the product of the probability of sequence q occurring, denoted by $\rho(q)$, and the sum of expected reward of the last action $a_q^{|q|}$ in that sequence, denoted by $v(q)$:

$$u_q \triangleq \rho(q)\cdot v(q). \tag{3}$$

As shown by [Aras *et al.*, 2007], we can recursively compute $\rho(q)$ by the following

$$\begin{aligned}\rho(q) &\triangleq \Pr\Big(\bigwedge_{i\in\mathcal{O}(q)} o_q^i \;\Big|\; b_0 \bigwedge_{j\in\mathcal{A}(q)} a_q^j\Big)\\ &= \prod_{i\in\mathcal{O}(q)} \Pr\Big(o_q^i \;\Big|\; b_0 \bigwedge_{\substack{j\in\mathcal{A}(q)\\ j<i}} a_q^j\Big)\\ &= \prod_{i\in\mathcal{O}(q)} \Pr(o_q^i \mid b_q^{i-1}),\end{aligned}$$

for $q \in \widetilde{A} \cup \widetilde{\mathcal{O}}$, $|q| > 1$, where b_q^{i-1} is the prior belief state after action a_q^{i-1} in q, defined by

$$b_q^{i-1}(s) \triangleq \sum_{s' \in \mathcal{S}} T(s', a_q^{i-1}, s) \cdot b_q^{i-2}(s').$$

The posterior belief b_q^i is computed as

$$b_q^i(s) \triangleq \frac{O(o_q^i, s) \cdot b_q^{i-1}(s)}{\Pr(o_q^i \mid b_q^{i-1})}, \quad \forall s \in \mathcal{S},$$

where $\Pr(o_q^i \mid b_q^{i-1}) \triangleq \sum_{s \in \mathcal{S}} b_q^{i-1}(s) \cdot O(o_q^i, s)$. Notice that by the definition of a sequence, if $i \in \mathcal{O}(q)$ corresponds to an observation then $(i-1) \in \mathcal{A}(q)$ corresponds to an action. For $|q| = 1$, the probability $\rho(q) \triangleq 1$ because such state is always reachable. Hence, the expected reward $v(q)$ of sequence $q \in \widetilde{\mathcal{A}}$ is computed by

$$v(q) \triangleq \sum_{s \in \mathcal{S}} b_{q-1}(s) \cdot U(s, a_q^{|q|}). \tag{4}$$

The total expected reward of a conditional plan $\mathbf{x}$ is given by $\sum_{q \in \widetilde{\mathcal{A}}} u_q x_q$.

Remark 1. *Given a sequence $q \in \widetilde{\mathcal{A}}$, the reward u_q can be computed independently from a conditional plan $\mathbf{x}$.*

3.2 Calculating Risk

We adopt the notion of *execution risk* from [Santana *et al.*, 2016; Santana and Williams, 2015], denoted by $er(0)$, to compute the probability of ending up in risky states starting from the initial sequence 0. The chance constraint requires that

$$er(0) \leq \Delta. \tag{5}$$

For a given belief b, let $r(b) \triangleq \sum_{s \in \mathcal{R}} b(s)$ be the expected risk of failure. We write the execution risk as a function of a conditional plan $\mathbf{x} \in \{0,1\}^{|\widetilde{\mathcal{A}}|}$ using the recursion in Eqn. (6) (see [Santana *et al.*, 2016] for a systematic derivation). $\widetilde{\Pr}(o \mid \bar{b}_{q-1})$ is the observation probability, and $\bar{b}_{q-1}(s)$ and $\widetilde{b}_q(s)$ are *safe* prior and posterior beliefs, respectively, defined by,

$$\widetilde{\Pr}(o \mid \bar{b}_{q-1}) \triangleq \sum_{s' \in \mathcal{S}} O(o, s') \bar{b}_{q-1}(s'),$$

$$\bar{b}_{q-1}(s) \triangleq \frac{\sum_{s' \in \mathcal{S} \setminus \mathcal{R}} T(s', a, s) \widetilde{b}_{q-2}(s')}{1 - r(\widetilde{b}_{q-2})},$$

$$\widetilde{b}_q(s) \triangleq \frac{O(o_q^{|q|}, s) \cdot \bar{b}_q(s)}{\widetilde{\Pr}(o \mid \bar{b}_{q-1})},$$

for $q \in \widetilde{\mathcal{O}}$, and where $\widetilde{b}_q^0 = b_0$. The reason we distinguish safe prior and posterior from the standard definitions is that reaching history q implies, roughly speaking, that none of the past states were in $\mathcal{R}$, otherwise no action could have been taken to proceed.

Next, we solve the recursive Eqn. (6) and rearrange the terms to obtain

$$\begin{aligned} er(0) = r(b_0) + \sum_{q \in \widetilde{\mathcal{A}}: \tau(|q|) < h} \Bigg(\sum_{o \in \mathcal{O}} \Big(\prod_{i \in \mathcal{O}(qo)} \big(1 - r(\widetilde{b}_{qo}^{i-2})\big) \\ \cdot \widetilde{\Pr}(o_{qo}^i \mid \bar{b}_{qo}^{i-1}) \Big) \cdot r(\widetilde{b}_{qo}) \Bigg) \cdot x_q \\ + \sum_{q \in \widetilde{\mathcal{A}}: \tau(|q|) = h} \Bigg(\Big(\prod_{i \in \mathcal{O}(q)} \big(1 - r(\widetilde{b}_q^{i-2})\big) \\ \cdot \widetilde{\Pr}(o_q^i \mid \bar{b}_q^{i-1}) \Big) \cdot r(\bar{b}_q) \Bigg) \cdot x_q, \end{aligned} \tag{7}$$

where $\bar{b}_q^i$ and $\widetilde{b}_q^j$ are the belief states after the i-th action and j-th observation, respectively. For $q \in \widetilde{\mathcal{A}}$, write the coefficient of x_q from Eqn. (7) as

$$r_q \triangleq \begin{cases} \sum_{o \in \mathcal{O}} \Big(\prod_{i \in \mathcal{O}(qo)} \big(1 - r(\widetilde{b}_{qo}^{i-2})\big) \cdot \widetilde{P}(o_{qo}^i \mid \bar{b}_{qo}^{i-1}) \Big) \\ \quad \cdot r(\widetilde{b}_{qo}), \quad \text{if } \tau(|q|) < h, \\ \Big(\prod_{i \in \mathcal{O}(q)} \big(1 - r(\widetilde{b}_q^{i-2})\big) \cdot \widetilde{P}(o_q^i \mid \bar{b}_q^{i-1}) \Big) \\ \quad \cdot (1 - r(\widetilde{b}_{q-1})) \cdot r(\bar{b}_q), \quad \text{if } \tau(|q|) = h. \end{cases} \tag{8}$$

By Eqns. (7) and (8), the chance Cons. (5) can be written as

$$\sum_{q \in \widetilde{\mathcal{A}}} r_q \cdot x_q \leq \Delta - r(b_0).$$

Remark 2. *For a given sequence $q \in \widetilde{\mathcal{A}}$, the risk r_q can be computed independently from a conditional plan $\mathbf{x}$.*

3.3 ILP Formulation for CC-POMDP

Chance-constrained POMDP

Given a precomputed $(u_q, r_q)_{q \in \widetilde{\mathcal{A}}} \in \mathbb{R}_+^{2|\widetilde{\mathcal{A}}|}$ from Eqns. (3) and (8), a precedence graph $\mathcal{G}^a = (\widetilde{\mathcal{A}}, \mathcal{E}^a)$, $\widetilde{\mathcal{O}}$, $C \triangleq \Delta - r(b_0), C \in \mathbb{R}_+$, CC-POMDP can be formulated by the following integer linear program:

$$\text{(ILP)} \quad \max_{x_q \in \{0,1\}} \sum_{q \in \widetilde{\mathcal{A}}} u_q \cdot x_q$$

$$\text{Subject to} \sum_{a \in \mathcal{A}} x_{qa} \leq 1, \qquad \forall q \in \widetilde{\mathcal{O}} \cup \{0\}, \tag{9}$$

$$x_q \geq x_{q'}, \qquad \forall (q, q') \in \mathcal{E}^a, \tag{10}$$

$$\sum_{q \in \widetilde{\mathcal{A}}} r_q \cdot x_q \leq C, \tag{11}$$

Constrained POMDP

In CC-POMDP, a secondary objective function $U'(s,a)$ is given as a constraint with an upper bound $C' \in \mathbb{R}_+$ on the total expected value. We can use the above ILP formulation for C-POMDP with the following changes in the coefficients: r_q is replaced by a penalty r'_q, calculated in the same way as u_q replacing $U(s,a)$ by $U'(s,a)$ in Eqn. (4) (as shown in Sec. 3.1). Thus Cons. (11) is replaced by

$$er(q) \triangleq \begin{cases} r(b_0) + (1 - r(b_0)) \sum_{a\in\mathcal{A}} \sum_{o\in\mathcal{O}} \widetilde{\Pr}(o \mid \bar{b}_a) \cdot er(ao) & \text{if } q = 0, \\ r(\widetilde{b}_q) \cdot x_{q-1} + (1 - r(\widetilde{b}_q)) \sum_{a\in\mathcal{A}} \sum_{o\in\mathcal{O}} \widetilde{\Pr}(o \mid \bar{b}_{qa}) \cdot er(qao), & \text{if } q \in \widetilde{O} \text{ and } \tau(|q| + 2) \le h, \\ r(\widetilde{b}_q) \cdot x_{q-1} + (1 - r(\widetilde{b}_q)) \sum_{a\in\mathcal{A}} er(qa), & \text{if } q \in \widetilde{O} \text{ and } \tau(|q| + 1) = h, \\ r(\widetilde{b}_q) \cdot x_q, & \text{if } q \in \widetilde{\mathcal{A}} \text{ and } \tau(|q|) = h, \end{cases} \tag{6}$$

$$\sum_{q\in\widetilde{\mathcal{A}}} r'_q x_q \le C'. \tag{12}$$

With the above formulation, we observe the similarity between the two problems. We highlight that C-POMDP is actually more general, because the parameters have higher degrees of freedom as we show below.

First, it is worth noting that one can slightly generalize C-POMDP by allowing the penalty function to be a function of state and action history instead of a single action, namely, $U' : \mathcal{S} \times \widetilde{\mathcal{A}} \to \mathbb{R}_+$ (denoted by C-POMDP*). Any instance $I = \big((u_q, r_q)_{q\in\widetilde{\mathcal{A}}}, C\big)$ of CC-POMDP can be easily reduced to an instance $I' = \big((u'_q, r'_q)_{q\in\widetilde{\mathcal{A}}}, C'\big)$ of C-POMDP* as follows: set $C' = C$ and define $U'(s, q) \triangleq \frac{r_q}{\rho(q)}, q \in \widetilde{\mathcal{A}}$. Following the same lines in Sec. 3.1 we obtain $r'_q = r_q$ (and $u'_q = u_q$). Any solution to such instance of C-POMDP* is also feasible to that of CC-POMDP with the same objective value.

Second, we note that C-POMDP (with the standard transition function) can be also seen as a generalization of CC-POMDP even for the fully observable case (C-MDP) . Given an instance I of CC-POMDP, we can compute an instance I' of C-(PO)MDP as follows: set the state space $\mathcal{S}'$ and observation space $\mathcal{O}'$ of C-POMDP to be equal to the set of observation sequences of CC-POMDP, $\mathcal{S}' = \mathcal{O}' = \widetilde{\mathcal{O}} \cup \{0\}$; set $O(q, q') = 1$ if $q = q'$ for $q, q' \in \mathcal{S}'$, and 0 otherwise (hence fully observable); set $T(q, a, q') = \frac{1}{|\mathcal{O}|}$ if $q' = qao$ and $o \in \mathcal{O}$, and 0 otherwise; $U(q, a) = u_q$ and $U'(q, a) = r_q$ for $q \in \mathcal{S}', a \in \mathcal{A}$. Following the steps in Sec. 3.1 we obtain $r'_q = r_q$ and $u'_q = u_q$.

4 Hardness of CC-POMDP

In this section we provide our main hardness result. The result is tightly focused on a highly restricted subset of CC-POMDP instances, with a planning horizon of two and a fully observable environment. This provides an argument that, in general, exact algorithms are infeasible for most CC-POMDP instances.

Theorem 4.1. CC-POMDP *is NP-Hard even for the fully obervable case ($\mathcal{O} = \mathcal{S}$), time horizon $h = 2$, and a single action $\mathcal{A} = \{a\}$.*

Proof. We present a reduction from the (weakly) NP-Hard knapsack problem (KP) to CC-POMDP.

Definition 4.2 (KP)**.** *Given a set of items $\{1, \dots, n\}$, each associated with a value $v_i \in \mathbb{R}_+$ and a weight $w_i \in \mathbb{R}_+$, $i = 1, \dots, n$, and a total capacity $P \in \mathbb{R}_+$; find a subset of items that achieve the highest aggregate value subject to the capacity constraint. More precisely,*

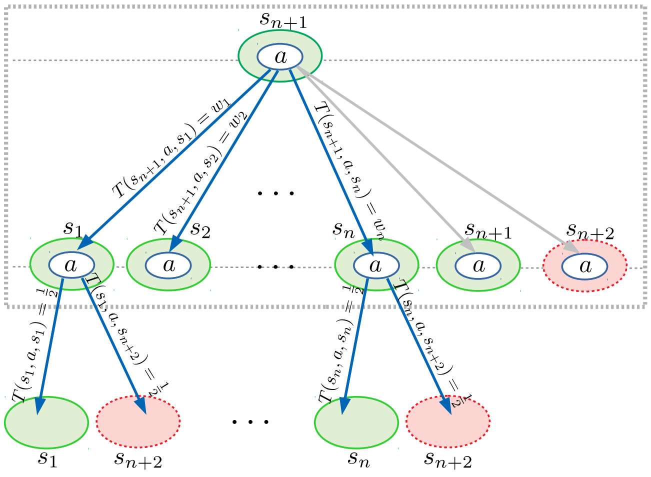
s_{n+1}
a
$T(s_{n+1}, a, s_1) = w_1$
$T(s_{n+1}, a, s_2) = w_2$
$T(s_{n+1}, a, s_n) = w_n$
s_1
s_2
s_n
s_{n+1}
s_{n+2}
$T(s_1, a, s_1) = \frac{1}{2}$
$T(s_1, a, s_{n+2}) = \frac{1}{2}$
$T(s_n, a, s_n) = \frac{1}{2}$
$T(s_n, a, s_{n+2}) = \frac{1}{2}$
s_1
s_{n+2}
s_n
s_{n+2}

Figure 3: A gadget for the reduction from KP to CC-POMDP. Blue arrows represent the state transition function, red dashed circles represent the risky state s_{n+2}, and green circles represent safe states. The subtree within the gray box represents the precedence relationship between action nodes.

$$\text{(KP)} \quad \max_{x_i\in\{0,1\}} \sum_{i=1}^{n} v_i \cdot x_i \quad \text{s.t.} \quad \sum_{i=1}^{n} w_i \cdot x_i \le P.$$

W.l.o.g. we assume $P \le 1$ and $\sum_{i=1}^{n} w_i = 1$ (which can be easily achieved via normalization). We present a reduction from an instance I of KP to instance I' of CC-POMDP as follows. Let $\mathcal{S} = \{s_1, s_2, \dots, s_n, s_{n+1}, s_{n+2}\}$, $\mathcal{R} = \{s_{n+2}\}$, $\mathcal{O} = \mathcal{S}$, $\mathcal{A} = \{a\}$, and $\Delta = \frac{P}{2}$. Since the system is fully observable ($O(s_i, s_j) = 1$ for $i = j$, and 0 otherwise), we have $b_0(s_{n+1}) = 1$, and $b_0(s) = 0$ for $s \in \mathcal{S} \backslash \{s_{n+1}\}$.

Based on the above, $r(b_0) = 0$, $C = \Delta - r(b_0) = \frac{P}{2}$. Define the state transition function as follows, for $s, s' \in \mathcal{S}$, $a \in \mathcal{A}$:

$$T(s, a, s') \triangleq \begin{cases} w_i & \text{if } s = s_{n+1}, s' = s_i \in \mathcal{S}\backslash\mathcal{R}, \\ \frac{1}{2} & \text{if } s \in \mathcal{S}\backslash\{s_{n+1}, s_{n+2}\}, s' \in \mathcal{R}, \\ \frac{1}{2} & \text{if } s = s' \in \mathcal{S}\backslash\mathcal{R} \\ 1 & \text{if } s = s' = s_{n+2}, \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$

Define the reward function as

$$U(s, a) \triangleq \begin{cases} \frac{v_i}{w_i} & \text{if } s = s_i \in \mathcal{S}\backslash\{s_{n+1}, s_{n+2}\}, \\ 0 & \text{otherwise.} \end{cases} \tag{14}$$

See Fig. 3 for a pictorial illustration of instance I'.

The precedence graph is given by $\mathcal{G}^a = (\widetilde{\mathcal{A}}, \mathcal{E}^a)$, where $\widetilde{\mathcal{A}} = \{a, as_1a, as_2a, \dots, as_na, as_{n+1}a, as_{n+2}a\}$, and $\mathcal{E}^a$ is defined as shown in Fig. 3 (the subtree within the box). Following the steps in Secs. 3.1–3.2,

$$\widetilde{\Pr}(o_i \mid \bar{b}_a) = \Pr(o_i \mid b_a) = b_a(s_i) = \bar{b}_a(s_i)$$
$$= T(s_{n+1}, a, s_i) = \begin{cases} w_i & \text{if } i = 1, \dots, n, \\ 0 & \text{otherwise;} \end{cases}$$
$$\widetilde{b}_{ao_i}(s_i) = b_{ao_i}(s_i) = \begin{cases} 1 & \text{if } i = 1, \dots, n, \\ 0 & \text{otherwise;} \end{cases}$$
$$\bar{b}_{ao_ia}(s) = b_{ao_ia}(s) = \begin{cases} \frac{1}{2} & \text{if } s \in \{s_i, s_{n+2}\}, \\ 0 & \text{otherwise,} \end{cases}$$

where $o_i = s_i, i = 1, \dots, n+2$. Hence, we obtain $r_q, q \in \widetilde{\mathcal{A}}$, as follows:

$$r_a = \sum_{o \in \mathcal{O}} (1 - r(b_0)) \cdot \widetilde{P}(o \mid b_0) \cdot r(\widetilde{b}_{ao}) = 0,$$
$$r_{ao_ia} = (1 - r(b_0)) \cdot \widetilde{P}(o_i \mid \bar{b}_a) \cdot (1 - r(\widetilde{b}_{ao_i})) \cdot r(\bar{b}_{ao_ia})$$
$$= \tfrac{w_i}{2}, \quad \text{for } i = 1, \dots, n,$$
$$r_{ao_{n+1}a} = r_{ao_{n+2}a} = 0.$$

We also obtain u_q as follows:

$$u_a = \rho(a) \cdot \sum_{s \in \mathcal{S}} b_0(s) U(s, a) = 0,$$
$$u_{ao_ia} = \Pr(o_i \mid b_a) \cdot \sum_{s \in \mathcal{S}} b_{ao_i}(s) \cdot U(s, a)$$
$$= w_i U(s_i, a) = w_i \tfrac{v_i}{w_i} = v_i, \text{ for } i = 1, \dots, n,$$
$$u_{ao_{n+1}a} = u_{ao_{n+2}a} = 0.$$

Now, any optimal solution $\mathbf{x}^*$ for instance I', of value denoted by $\textsc{Opt}(I')$, must satisfy $x^*_a = 1$; otherwise, $\textsc{Opt}(I') = 0$ (by Cons. (10)), which is smaller than that of the feasible solution $x'_a = 1, x'_{as_1a} = 1$, $x'_q = 0$, for $q \neq as_1a$, contradicting the optimality of $\mathbf{x}^*$. Let $V(\mathbf{x})$ and $V(\mathbf{x}')$ denote the objective values of solution $\mathbf{x} \in \{0,1\}^n$ for $\mathsf{KP}(I)$ and $\mathbf{x}' \in \{0,1\}^{|\widetilde{\mathcal{A}}|}$ for $\mathsf{CC\text{-}POMDP}(I')$, respectively.

Given a non-trivial feasible solution $\mathbf{x}'$ for $\mathsf{CC\text{-}POMDP}(I')$ (such that $x'_a = 1$, $x'_{as_{n+1}a} = x'_{as_{n+2}a} = 0$), one can obtain a solution $\mathbf{x}$ for $\mathsf{KP}(I)$ such that $x_i \triangleq x'_{as_ia}$. By Cons. (11),

$$\sum_{q \in \widetilde{\mathcal{A}}} r_q \cdot x'_q \le C \iff \sum_{i=1}^{n} \tfrac{w_i}{2} x'_{as_ia} + 0x'_a + 0x'_{as_{n+1}a}$$
$$+ 0x'_{as_{n+2}a} \le \tfrac{P}{2} \iff \sum_{i=1}^{n} w_i x_i \le P. \qquad (15)$$

Hence, solution $\mathbf{x}$ is feasible for $\mathsf{KP}(I)$. Note that the objective value of $\mathbf{x}'$ is equivalent to that of $\mathbf{x}$:

$$\sum_{q \in \widetilde{\mathcal{A}}} u_q \cdot x'_q = \sum_{i=1}^{n} v_i \cdot x_i, \qquad (16)$$

Hence, $V(\mathbf{x}) = \textsc{Opt}(I')$. It remains to show that $\mathbf{x}$ is optimal to $\mathsf{KP}(I)$. Suppose that there exists a solution $\mathbf{y}$ for $\mathsf{KP}(I)$ such that $V(\mathbf{y}) > \textsc{Opt}(I')$, then we can find $\mathbf{x}'$ for $\mathsf{CC\text{-}POMDP}(I')$ such that $x'_{as_ia} \leftarrow y_i$, $x'_a = 1$, $x'_{as_{n+1}a} = x'_{as_{n+2}a} = 0$, which is a feasible solution by (15) and Cons. (10), and $V(\mathbf{x}') > \textsc{Opt}(I')$ by (16), contradicting the optimality of $\mathbf{x}^*$. Therefore $\textsc{Opt}(I) = \textsc{Opt}(I')$.

Note that if there exists an optimal algorithm for CC-POMDP, then we can use it to obtain optimal solutions for KP. Since KP is NP-Hard, we conclude that such algorithm does not exist, unless P=NP. □

5 Algorithm

In this section, we provide an FPTAS that is applicable to both C-POMDP and CC-POMDP. We showed in Sec. 4 that CC-POMDP is NP-Hard, which implies that the FPTAS is the best polynomial time algorithm in theory in terms of approximation ratio (unless P=NP). In fact, we consider a generalization of the two problems by allowing arbitrary values for r_q, u_q, C in the ILP formulation. The new problem is also a generalization of the so-called *precedence constrained knapsack problem* (PKP) under tree topologies [Johnson and Niemi, 1983; Kellerer *et al.*, 2004].

Definition 5.1 (PKP). *Given a directed out-tree graph $\mathcal{G}^p = (\mathcal{V}^p, \mathcal{E}^p)$, where nodes represent items i, each of which has a value $u_i \in \mathbb{R}_+$ and a weight $r_i \in \mathbb{R}_+$, and edges correspond to the precedence order between items such that an item can be packed into the knapsack only if its parent is also packed (precedence relationship). The goal is to find a subset of items that maximizes the total value while satisfies the precedence relationship and the total weight is at most C.*

PKP can be formulated using the above ILP with $|\mathcal{O}| = 1$ (whereas r_q, u_q, C are allowed to be arbitrary non-negative numbers). When $|\mathcal{O}| > 1$, we call this generalization *multiple-choice precedence constrained knapsack problem* (MPKP).

Definition 5.2 (MPKP). *Given a directed And-Or graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where nodes are partitioned into And nodes and Or nodes. Contrary to* PKP*, only an And node i is associated with a value $u_i \in \mathbb{R}_+$ and a weight $r_i \in \mathbb{R}_+$. Edges correspond to the precedence order between And-Or nodes such that a node is selected only if its parent is also selected (precedence relationship). In addition, for an Or node, at most one of its children is allowed to be selected (disjunction condition). The goal is to find a subset of nodes that maximizes the total value of And nodes while satisfies the precedence relationship between And-OR nodes, the disjunction condition at Or nodes, and the total weight of And nodes is at most C.*

In the following, we present an FPTAS that solves MPKP. The algorithm is a non-trivial extension of the FPTAS for the PKP problem [Johnson and Niemi, 1983; Kellerer *et al.*, 2004] to account for *And-Or* graphs. The algorithm can also solve both C-POMDP and CC-POMDP since MPKP is a generalization of the two problems. First, we set up notation needed to describe the algorithm. For simplicity, we use the terminologies of CC-POMDP, namely, we say "reward" instead of "value" and "risk" instead of "weight". Define an *And-Or* graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the set of nodes $\mathcal{V} \triangleq \widetilde{A} \cup (\widetilde{\mathcal{O}} \cup \{0\})$ is a partition such that $\widetilde{A} \cap (\widetilde{\mathcal{O}} \cup \{0\}) = \varnothing$, and

$$\mathcal{E} \triangleq \Big\{(q, q') \mid q \in \mathcal{V} \text{ s.t. } \tau(|q|) < h, \\ \big((\text{if } q \in \widetilde{\mathcal{O}} \cup \{0\}, q' = qa, a \in \mathcal{A}) \\ \text{or } (\text{if } q \in \widetilde{\mathcal{A}}, q' = qo, o \in \mathcal{O})\big)\Big\}.$$

We call nodes in $\widetilde{\mathcal{A}}$ *And* nodes, while nodes in $\widetilde{\mathcal{O}} \cup \{0\}$ are called *Or* nodes. The tree is rooted at node 0.

Based on $\mathcal{E}$, we order nodes in $\mathcal{V}$ following the standard depth-first search order. More precisely, let $\mathcal{G}'(j, i) \subseteq \mathcal{G}$ represent the subtree induced by j-th node in that order, the first i children of j and all their successors, and all vertices in $\mathcal{V}$ such that their indices are lower than j. In other words, $\mathcal{G}'(j, i)$ is the expanded subtree using the depth first search at node j after fully expanding its i-th child. For simplicity, we say *node* j instead of the j-th node in the depth first search order. With a slight abuse of notation, we write $\mathcal{V} = \{0, 1, \ldots, m\}$, where $j \in \mathcal{V}$ refers to a sequence in $\widetilde{\mathcal{A}} \cup \widetilde{\mathcal{O}} \cup \{0\}$ in that order. Let $n = |\widetilde{\mathcal{A}}|$ (and $m - n = |\widetilde{\mathcal{O}}|$). In the following, we use subscript j instead of q to refer to nodes in $\mathcal{V}$. We write r_j and u_j to denote the risk and reward of the respective sequence. Let $\delta(j) \subseteq \mathcal{V}$ denote the set of children of j in $\mathcal{G}$. As a slight practical generalization, we can allow $|\delta(j)|$ to be different for each j.

5.1 Pseudo Polynomial-time Algorithm

We provide an exact algorithm to solve MPKP based on a novel dynamic program. The running time is polynomial in $\mathcal{L}$, defined as the set of all possible objective values. In general the size of $\mathcal{L}$ is exponential, but it could be polynomial for certain applications, e.g., when $u_j = 1$ for all $j \in \widetilde{\mathcal{A}}$ then $|\mathcal{L}| = n$. We show in the subsection below that with a proper rounding scheme, one can obtain an FPTAS for any large $\mathcal{L}$.

Define a 3D table $\mathcal{D}(j, i, \ell)$, where each cell corresponds to the minimum weight feasible solution within the subtree $\mathcal{G}'(j, i)$ that achieve an objective value of at least $\ell \in \mathcal{L}$, according to following rules. For $\ell \in \mathcal{L}$,

R1: $j \in \mathcal{O} \cup \{0\}$ and $i = 0$,

$$\mathcal{D}(j, 0, \ell) \triangleq \infty.$$

R2: $j \in \delta(0) \subseteq \widetilde{\mathcal{A}}$ and $i = 0$,

$$\mathcal{D}(j, 0, \ell) \triangleq \begin{cases} r_j & \text{if } u_j \geq \ell, \\ \infty & \text{otherwise.} \end{cases}$$

R3: $j \in \widetilde{\mathcal{A}} \cup \widetilde{\mathcal{O}} \cup \{0\}$ and $i > 0$,

$$\mathcal{D}(j, i, \ell) \triangleq \min\Big\{\mathcal{D}(j, i-1, \ell), \mathcal{D}(j_i, |\delta(j_i)|, \ell)\Big\},$$

where $j_i \in \delta(j)$ is the i-th child of j.

R4: $j \in \widetilde{\mathcal{A}} \backslash \delta(0)$ and $i = 0$,

$$\mathcal{D}(j, 0, \ell) \triangleq \begin{cases} \mathcal{D}(k', t-1, [\ell - u_j]^+) + r_j, \\ \quad \text{if } \mathcal{D}(k', t-1, [\ell - u_j]^+) + r_j \leq C \\ \infty, \text{otherwise,} \end{cases}$$

where $[\beta]^+ = \beta$ if $\beta \geq 0$, and $[\beta]^+ = 0$ if $\beta < 0$; j is a child of k, and k is the t-th child of k'. Notice that by the definition of $\mathcal{G}$, $j \in \widetilde{\mathcal{A}}$ implies that $k \in \widetilde{\mathcal{O}}$ and $k' \in \widetilde{\mathcal{A}}$.

Algorithm 1 MPKP-FPTAS$[(u_j, r_j)_{j \in \widetilde{\mathcal{A}}}, C, \mathcal{G}, \epsilon]$

```
Input: Rewards and risks $(u_j, r_j)_{j \in \widetilde{\mathcal{A}}}$; capacity $C$; And-Or
    graph $\mathcal{G}$; accuracy parameter $\epsilon$
Output: $(1-\epsilon)$-solution $\mathbf{x}'$ to MPKP
1: Let $K = \frac{\epsilon P}{n}$
2: $\bar{u}_j \leftarrow \lfloor \frac{u_j}{K} \rfloor$ for all $j \in \widetilde{\mathcal{A}}$
3: $\mathbf{x}' \leftarrow$ MPKP-Exact$\Big[\{1, 2, \ldots \lfloor \frac{nP}{K} \rfloor\}, (\bar{u}_j, r_j)_{j \in \widetilde{\mathcal{A}}}, C, \mathcal{G}\Big]$
4: return $\mathbf{x}'$
```

By the condition in R4, the process always maintains feasibility with respect to the capacity Cons. (11). Note that the precedence constraints are systematically maintained throughout the entire process. The optimum solution value corresponds to the maximum value $\bar{\ell} \in \mathcal{L}$ for which $\mathcal{D}(0, |\delta(0)|, \bar{\ell}) < \infty$. Through out the above steps, one can compute the corresponding binary vector $\mathbf{x}'$ such that $\sum_{j \in \widetilde{\mathcal{A}}} u_j x'_j = \bar{\ell}$. We denote the algorithm that computes $\mathbf{x}'$ for any instance $I = ((u_j, r_j)_{j \in \widetilde{\mathcal{A}}}, C, \mathcal{G}, \widetilde{\mathcal{A}}, \widetilde{\mathcal{O}})$ by MPKP-Exact$[\mathcal{L}, (u_j, r_j)_{j \in \widetilde{\mathcal{A}}}, C, \mathcal{G}]$. The running time of MPKP-Exact$[\mathcal{L}, (u_j, r_j)_{j \in \widetilde{\mathcal{A}}}]$ is $O(n|\mathcal{L}|)$. (In the context of constant horizon (C)C-POMDP, $n = \sum_{t=1}^{h} |\mathcal{A}|^t |\mathcal{O}|^{t-1}$.) Hence, we state the following lemma.

Lemma 5.3. MPKP-Exact$[\mathcal{L}, (u_j, r_j)_{j \in \widetilde{\mathcal{A}}}, C, \mathcal{G}]$ *obtains optimal solutions for* MPKP *in* $O(n|\mathcal{L}|)$.

5.2 FPTAS

In this section we present our $(1 - \epsilon)$-approximation algorithm for MPKP. We follow a standard rounding technique for knapsack [Vazirani, 2013]. The basic idea is to round all rewards u_j such that the set of all possible objective values $\mathcal{L}$ has polynomial cardinality, while closely approximates the actual set of all possible rewards. Then, we envoke MPKP-Exact$[\cdot]$ with the new $\mathcal{L}$. More precisely, let $P \triangleq \max_{j \in \widetilde{\mathcal{A}}} u_j$ be the maximum reward. Note that nP is a trivial upper bound on OPT. Now, divide all rewards u_j by a factor $K \triangleq \frac{\epsilon P}{n}$ and then round down. As a result, the set of all possible rewards $\mathcal{L} = \{0, 1, 2, \ldots \lfloor \frac{nP}{K} \rfloor\}$ has a polynomial length, $|\mathcal{L}| = \lfloor \frac{n^2}{\epsilon} \rfloor + 1$. Theorem 5.4 shows such rounding yields a good approximation. The pseudocode is provided in Alg. 1, denoted by MPKP-FPTAS.

Theorem 5.4. MPKP-FPTAS *is a fully polynomial time approximation scheme for* MPKP *and runs in* $O(\frac{1}{\epsilon} n^3)$.

A proof can be obtained following a standard rounding scheme [Vazirani, 2013]. Due to the lack of space, we omit our proof.

Remark. *One can speedup the algorithm in practice using a smaller set* $\mathcal{L}$ *with a tighter upper bound on the optimal. For example, instead of* nP*, one can use the objective value of the linear relaxation of* MPKP*, such that* $x_j \in [0, 1]$*, as an upper bound, which can be obtained using a standard LP solver.*

Corollary 5.5. MPKP-FPTAS *is a fully polynomial time approximation scheme for* (C)C-POMDP *and runs in* $O(\frac{1}{\epsilon} |\mathcal{A}|^{3h} |\mathcal{O}|^{3(h-1)})$.

6 Conclusion

In this work, we provide a fundamental study for constant-horizon (C)C-POMDP problem. A reduction is given from CC-POMDP problem to C-POMDP, and a novel ILP formulation that can be solved using standard ILP solvers. We show that CC-POMDP (consequently, C-POMDP) is NP-Hard even for the fully observable case and a planning horizon of two. We provide an FPTAS that can solve a generalization of constant-horizon C-POMDP called MPKP (and hence solve (C)C-POMDP) in polynomial time, within a bounded optimally gap given as input. We believe our results provide fundamental insights into the problem and can lead to future development of faster heuristics for (C)C-POMDP without compromising optimality.

References

[Altman, 1999] Eitan Altman. *Constrained Markov decision processes*, volume 7. CRC Press, 1999.

[Aras *et al.*, 2007] Raghav Aras, Alain Dutech, François Charpillet, et al. Mixed integer linear programming for exact finite-horizon planning in decentralized pomdps. In *ICAPS*, pages 18–25, 2007.

[Dolgov and Durfee, 2005] Dmitri Dolgov and Edmund Durfee. Stationary deterministic policies for constrained mdps with multiple rewards, costs, and discount factors. In *International Joint Conference on Artificial Intelligence*, volume 19, page 1326, 2005.

[Feinberg and Shwartz, 1996] Eugene A Feinberg and Adam Shwartz. Constrained discounted dynamic programming. *Mathematics of Operations Research*, 21(4):922–945, 1996.

[Howard, 1960] Ronald A Howard. Dynamic programming and markov processes. 1960.

[Isom *et al.*, 2008] Joshua D Isom, Sean P Meyn, and Richard D Braatz. Piecewise linear dynamic programming for constrained pomdps. In *AAAI*, volume 1, pages 291–296, 2008.

[Johnson and Niemi, 1983] David S Johnson and KA Niemi. On knapsacks, partitions, and a new dynamic programming technique for trees. *Mathematics of Operations Research*, 8(1):1–14, 1983.

[Kaelbling *et al.*, 1996] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

[Kaelbling *et al.*, 1998] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

[Kellerer *et al.*, 2004] Hans Kellerer, Ulrich Pferschy, and David Pisinger. *Knapsack Problems*. Springer, 2004.

[Kim *et al.*, 2011] Dongho Kim, Jaesong Lee, Kee-Eung Kim, and Pascal Poupart. Point-based value iteration for constrained pomdps. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[Kress-Gazit *et al.*, 2009] Hadas Kress-Gazit, Georgios E Fainekos, and George J Pappas. Temporal-logic-based reactive mission and motion planning. *IEEE transactions on robotics*, 25(6):1370–1381, 2009.

[Lusena *et al.*, 2001] Christopher Lusena, Judy Goldsmith, and Martin Mundhenk. Nonapproximability results for partially observable markov decision processes. *Journal of artificial intelligence research*, 14:83–103, 2001.

[Ono *et al.*, 2012] Masahiro Ono, Yoshiaki Kuwata, and J Balaram. Joint chance-constrained dynamic programming. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 1915–1922. IEEE, 2012.

[Papadimitriou and Tsitsiklis, 1987] Christos H Papadimitriou and John N Tsitsiklis. The complexity of markov decision processes. *Mathematics of operations research*, 12(3):441–450, 1987.

[Poupart *et al.*, 2015] Pascal Poupart, Aarti Malhotra, Pei Pei, Kee-Eung Kim, Bongseok Goh, and Michael Bowling. Approximate linear programming for constrained partially observable markov decision processes. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[Puterman, 2014] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[Russell and Norvig, 2016] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.

[Santana and Williams, 2015] Pedro Henrique Santana and Brian C Williams. Dynamic execution of temporal plans with sensing actions and bounded risk. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[Santana *et al.*, 2016] Pedro Santana, Sylvie Thiébaux, and Brian Williams. Rao*: an algorithm for chance constrained pomdps. In *Proc. AAAI Conference on Artificial Intelligence*, 2016.

[Sondik, 1971] Edward J Sondik. The optimal control of partially observable markov decision processes. *PhD the sis, Stanford University*, 1971.

[Timmons and Williams, 2015] Eric Timmons and Brian C Williams. Enumerating preferred solutions to conditional simple temporal networks quickly using bounding conflicts. In *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[Trevizan *et al.*, 2016] Felipe Trevizan, Sylvie Thiébaux, Pedro Santana, and Brian Williams. Heuristic search in dual space for constrained stochastic shortest path problems. In *Twenty-Sixth International Conference on Automated Planning and Scheduling*, 2016.

[Undurti and How, 2010] Aditya Undurti and Jonathan P How. An online algorithm for constrained pomdps. In *2010 IEEE International Conference on Robotics and Automation*, pages 3966–3973. IEEE, 2010.

[Vazirani, 2013] Vijay V Vazirani. *Approximation algorithms*. Springer Science & Business Media, 2013.