Computers and Chemical Engineering 56 (2013) 101–113

Contents lists available at SciVerse ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

MILP based value backups in partially observed Markov decision processes (POMDPs) with very large or continuous action and observation spaces

Rakshita Agrawal[a], Matthew J. Realff[a], Jay H. Lee[b,*]

[a] School of Chemical and Biomolecular Engineering, Georgia Institute of Technology, Atlanta, GA 30022, USA
[b] Department of Chemical and Biomolecular Engineering, Korea Advanced Institute of Science and Technology, Daejeon, Republic of Korea

ARTICLE INFO

Article history:
Received 22 February 2013
Received in revised form 24 April 2013
Accepted 25 April 2013
Available online 27 May 2013

Keywords:
Markov decision processes
Dynamic programming
Mathematical programming
Partial observation
Network reliability

ABSTRACT

Partially observed Markov decision processes (POMDPs) serve as powerful tools to model stochastic systems with partial state information. Since the exact solution methods for POMDPs are limited to problems with very small sizes of state, action and observation spaces, approximate point-based solution methods like PERSEUS have gained popularity. In this work, a mixed integer linear program (MILP) is developed for calculation of exact value updates (in PERSEUS and similar algorithms), when the POMDP has very large or continuous action space. Since the solution time of the MILP is very sensitive to the size of the observation space, the concept of post-decision belief space is introduced to generate a more efficient and flexible model. An example involving a flow network is presented to illustrate the concepts and compare the results with those of the existing techniques.

1. Introduction

POMDP describes a discrete-time stochastic control process when the states of the environment are partially observed. At any time, the system is in one of the discrete states $s \in S$ where S is a set of all permissible states and is called state space. By taking an action a, the system transitions probabilistically to the next state $s' \in S$ according to known probability $p(s'|s,a)$ and accrues a reward $r(s,a)$. The next state s' is not completely observed but an observation o may be made, which is probabilistically related to the state s' and action a by $p(o|s',a)$ through stochastic system dynamics. The sets of all permissible actions and observations are called action and observation space, respectively. The objective is to find a control policy mapping the probability distribution across the states (called *belief state*) to action such that the expected value of the total rewards accrued over a time horizon is maximized.

Many real-world decision/control problems are characterized by probabilistic state transitions, partially observed states, and a reward/cost functions to be maximized/minimized over a finite or infinite time horizon. Therefore POMDPs have appeared in various applications including machine maintenance, structural inspection, elevator control, robotics, dialog control, and even population control in fishery industries (Cassandra, 1998). The main obstacle to wider use, however, has been the lack of computationally efficient algorithms for the value update.

Exact solution for POMDP requires solving the Bellman's optimality equation (Bellman, 1957) formulated with respect to the belief state. Since the exact solution methods for POMDPs are limited to problems with very small sizes of state, action and observation spaces, approximate solution methods have gained popularity. A saving grace for POMDP is that the optimal value function is piecewise linear and convex. Hence, the point based methods, which consider a fixed or evolving set of prototype belief points instead of considering the entire belief simplex, have been popular. A particular point based method called PERSEUS (Matthijs & Vlassis, 2005) favorably makes use of the piecewise linear and convex (PWLC) structure of the value function to speed up convergence. In this work, POMDPs with very large or continuous action space are considered. In the current form of PERSEUS and many other point based methods, presence of continuous actions or very large action space makes it practically impossible to compute the value backups exactly.

There is some literature that considers POMDPs with very large or continuous action spaces. Among the available POMDP solution methods, policy search methods are better equipped at handling continuous action spaces. An example is Pegasus (Ng & Jordan, 2000), which estimates the value of a policy by simulating trajectories using a fixed random seed, and adapts its policy in order to maximize the value. Pegasus can handle continuous action spaces

* Corresponding author. Tel.: +82 42 350 3926; fax: +82 42 350 3910.
E-mail addresses: jayhlee@kaist.ac.kr, eva@kt.dtu.dk (J.H. Lee).

0098-1354/$ – see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.compchemeng.2013.04.020

at the cost of a sample complexity that is polynomial in the size of the state space. Baxter and Bartlett (2001) propose a policy gradient method that searches in the space of randomized policies; this method can also handle continuous actions. The main disadvantages of policy search methods are the need to choose a particular policy class and the fact that they are prone to local optima. Thrun (1998) and Matthijs and Vlassis (2005) consider sampling techniques to keep the active size of the action space relatively small for continuous or very large action spaces. In the Monte Carlo POMDP (MC-POMDP) method of Thrun (1998), real-time dynamic programming is applied on a POMDP with a continuous state and action space. In that work, beliefs are represented by sets of samples drawn from the state space, while the values of the *Q*-functions, defined over belief state and action ($Q(b,a)$), are approximated by nearest-neighbor interpolation from a (growing) set of prototype values and are updated by online exploration and the use of sampling-based Bellman backups. In contrast with PERSEUS, the MC-POMDP method does not exploit the PWLC structure of the value function. Both methods are problem dependent and may lead to loss of solution quality in certain applications.

Alternatively, by the use of mathematical programming, exact value backups may be ensured in the presence of large or continuous action and/or observation space. This paper seeks to make two contributions: (i) First, mathematical programming models are developed to compute exact value backups associated with PERSEUS in presence of very large of continuous action spaces. (ii) Alternative formulation around post decision belief state is developed to allow for more efficient and flexible computation of the value updates in the presence of large sized observation space. The two algorithms are illustrated by example problems and results are compared with those from traditional techniques.

The paper is organized as follows: In Section 2, we describe POMDPs and the point based solution method PERSEUS which is the underlying algorithm we are using to solve POMDPs. We also motivate the problem by using an illustrative flow network problem and describe its formulation as POMDP. In Section 3, we develop the mathematical program to compute the value backups for PERSEUS. In Section 4, we introduce the notion of post decision belief state and reformulate value backups around it. In Section 5, we discuss application of developed mathematical programs to two illustrative problems and show results in terms of convergence times and solution quality.

2. POMDP description and motivating example

2.1. POMDP description

POMDP corresponds to a tuple (S, A, Θ, T, OP, R) where S is a set of states, A is a set of actions, Θ is a set of observations, T: $S\times A\times S\rightarrow[0,1]$ is a set of transition probabilities that describe the dynamic behavior of the modeled environment, OP: $S\times A\times\Theta\rightarrow[0,1]$ is a set of observation probabilities that describe the relationships among observations, states and actions, and R: $S\times A\times S\rightarrow R^1$ denotes a reward model that determines the reward when action a is taken in state s leading to next state s'. The dependence of reward function on s' is usually suppressed by taking a weighted average over all possible next states ($r(s,a)=\sum_{s'}p(s'|s,a)R(s,a,s')$). Symbols s, s', o and a are used to denote current state, next state, observation and action and belong to sets S, S, Θ and A, respectively. Since the state s at each time is not observed directly but indirectly through o, one does not know the current state with certainty. Hence, the information about the state is represented by belief state $b(s)$, which represents the probability of being in state s at a given time. $b(s)$ for all $s\in S$ must be defined at each time and the vector containing the entire probability distribution over the state space is called the belief state vector to be denoted by b hereafter. $0\leq\gamma<1$ is the discount rate that discounts the future rewards. The goal is to find a control policy mapping the belief state vector to a that maximizes the discounted sum of rewards over a time horizon, which can be either finite or infinite (in our case) as shown in (1).

$$\Pi^*=arg\left(\max_{a_t}\sum_{t=1}^{\infty}\gamma^{t-1}r(s_t,a_t)\right) \tag{1}$$

Infinite horizon POMDP with discounting is used in all illustrations, γ being the discounting factor. Equivalent models can be derived for finite horizon POMDPs with little difficulty.

The optimal policy for the above is defined by the following so called Bellman's optimality equation written with respect to the belief state vector.

$$V(b)=\max_{a\in A}\left\{\sum_{s\in S}r(s,a)b(s)+\gamma\sum_{o\in O}p(o|b,a)V(b^{a,o})\right\} \tag{2}$$

$$b^{a,o}(s')=\frac{\sum_{s\in S}p(s'|s,a)p(o|s',a)b(s)}{\sum_{s'\in S}\sum_{s\in S}p(s'|s,a)p(o|s',a)b(s)} \tag{3}$$

$$p(o|b,a)=\sum_{s'\in S}\sum_{s\in S}p(s'|s,a)p(o|s',a)b(s) \tag{4}$$

whereas the belief state $b(s)$ represents the probability of being in state s at a given time, $b^{a,o}(s')$ is the belief state at the next time period, which is reached by taking an action a and making an observation o.

Typically, the above is solved through iteration in the following manner:

$$V_{n+1}(b)=\max_{a\in A}\left\{\sum_{s\in S}r(s,a)b(s)+\gamma\sum_{o\in O}p(o|b,a)V_n(b^{a,o})\right\} \tag{5}$$

The above equation is very difficult to handle from a numerical standpoint as $V(b)$ is an arbitrary function of the continuous belief state vector b. Note that the right side involves maximization of an expectation quantity.

2.2. Point based solution method for POMDP

A saving grace for POMDP is that the optimal value function is known to be piecewise linear and convex. Based on this property, point based methods like PERSEUS have become popular.

Adopting the POMDP notation from (Matthijs & Vlassis, 2005), in the point based methods, the value function that appears in the value backup of Eq. (5) takes the form of

$$V_n(b^{a,o})=\max_i\sum_{s'\in S}\alpha_n^i(s')b^{a,o}(s') \tag{6}$$

In the above, α_n^i, $\quad i=1,2,\ldots|V_n|$ is the set of gradient vectors that characterizes the value function at nth iteration (denoted by V_n). The dimensionality of the gradient vectors is $|S|$, the size of the underlying state space. $\alpha^i{}_n\ (s')$ represents the scalar element of the gradient vector $\alpha^i{}_n$ corresponding to the state s'. Similar to the fully observable Markov decision processes (FO-MDP or simply MDP), for every iteration, the computation time is proportional to $|A|$ when enumeration of all actions is used. This is attributed to the max operation in (5). Additionally, the size of all possible gradient vectors at the $n+1$th iteration is $|V_n||A||O|$, where $|V_n|$ is the number of gradient vectors that characterize V_n. Therefore, POMDP with large action and observation spaces prove to be a challenge. It is