Journal of Process Control 20 (2010) 1038–1048

Contents lists available at ScienceDirect

Journal of Process Control

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

Approximate dynamic programming approach for process control

Jay H. Lee*, Weechin Wong

311 Ferst Dr. NW, School of Chemical and Biomolecular Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0100, USA

ARTICLE INFO

Article history:
Received 7 December 2009
Received in revised form 31 May 2010
Accepted 11 June 2010

Keywords:
Stochastic process control
Stochastic dynamic programming
Approximate dynamic programming
Dual control
Constrained control

ABSTRACT

We assess the potentials of the approximate dynamic programming (ADP) approach for process control, especially as a method to complement the model predictive control (MPC) approach. In the artificial intelligence (AI) and operations research (OR) research communities, ADP has recently seen significant activities as an effective method for solving Markov decision processes (MDPs), which represent a type of multi-stage decision problems under uncertainty. Process control problems are similar to MDPs with the key difference being the *continuous* state and action spaces as opposed to discrete ones. In addition, unlike in other popular ADP application areas like robotics or games, in process control applications first and foremost concern should be on the safety and economics of the on-going operation rather than on efficient learning. We explore different options within ADP design, such as the pre-decision state vs. post-decision state value function, parametric vs. nonparametric value function approximator, batch-mode vs. continuous-mode learning, and exploration vs. robustness. We argue that ADP possesses great potentials, especially for obtaining effective control policies for stochastic constrained nonlinear or linear systems and continually improving them towards optimality.

1. Introduction

Model predictive control (MPC) is a technique in which the current control action is obtained by minimizing on-line, a cost criterion defined on a finite time interval. Nominal deterministic trajectories of future disturbance signals and uncertainties are necessarily assumed in order to obtain an optimization problem amenable to on-line solution via math programming. The solution generates a control sequence from which the first element is extracted and implemented. The procedure is repeated at the next time instant. Owing to its ability to handle constrained, multivariable control problems in an optimal manner, MPC has become the de-facto advanced process control solution for the process industries today.

MPC is by now considered to be a mature technology owing to the plethora of research and industrial experiences during the past three decades. Despite this, it has some fundamental limitations, which prevents it from being a panacea for all process control problems. One well-known limitation is the potentially exorbitant on-line computation required for solving a large-scale, and potentially non-convex math program that scales with the dimension of the state as well as the length of prediction horizon. Recent developments [1] have made some headway in tackling this problem although nontrivial computational challenges still exist.

The second limitation arises from the fact that the deterministic formulation adopted by MPC is inherently limited in addressing uncertainty in a closed-loop optimal fashion. Its open-loop optimal control formulation used to find the control moves at each sample time means the fact that information about future uncertainty will be revealed, this being generally beneficial for control performance, is not considered. Most of the past attempts at ameliorating the impact of uncertainty has been reflected in robust MPCs formulations based on the objective of minimizing the worst-case scenarios [2] at the expense of overly conservative policies. Multi-scenario formulations [1] have also been developed but the number of scenarios is limited and they do not give closed-loop optimal policies in general. Stochastic programming-based methodologies [3] allow for recourse actions at the computational expense of enumerating an exponentially growing number of scenarios. Chance constrained optimization formulation has also been studied extensively by a number of authors [4,5].

In this paper, we examine the possibility of lessening or removing the above-mentioned limitations by combining MPC with an approach called “approximate dynamic programming (ADP).” ADP is a technique that surfaced from the research on reinforcement learning in the artificial intelligence (AI) community [6,7]. It has its theoretical foundations in the traditional dynamic programming by Richard Bellman [8] but its computational bottlenecks, termed as “the curse of dimensionality” by Bellman himself, are relieved through ideas such as intelligent sampling of the state space

* Corresponding author. Tel.: +1 404 385 2148; fax: +1 404 894 2866.
E-mail addresses: jay.lee@chbe.gatech.edu (J.H. Lee), weechin.wong@chbe.gatech.edu (W. Wong).

0959-1524/$ – see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jprocont.2010.06.007

through simulations and function approximation. ADP, due to its root in AI, has mainly been studied in the context of Markov decision processes (MDPs), which involve discrete finite state/action spaces and probabilistic transitions. Hence, its application to process control problems, which typically involve continuous state/action spaces, is not straightforward. In addition, the characteristics of process control problems are somewhat different from those of robotics, games, and resource allocation problems. For example, in process control applications, the idea of "learning by mistakes" for the sake of efficient learning, may not be tolerated as mistakes often bring unacceptable consequences in terms of safety and economics. Hence, extension of ADP to process control may require significant care and possibly some new tools.

Design of an ADP algorithm involves a variety of choices, including type of function approximator, pre-decision vs. post-decision formulation, batch vs. continuous updating of the value table, and exploration vs. robustness trade-off. We will visit these issues, carefully examining the implications of these choices in the context of designing a learning algorithm for process control applications. In addition, we will also consider the complementary nature or synergies between ADP and MPC.

It is to be noted that most previous published works on ADP, as specialized for process control problems, are based on the pre-decision state formulation and are better suited for deterministic problems. For stochastic problems, such as those treated in the examples of this paper, the post-decision state formulation (see Section 3.3) confers immediate practical benefits since it allows the efficient use of off-the-shelf optimization solvers found in all MPC technology.

The rest of the paper is organized as follows. In Section 2, we will briefly review the basics of MDP, ADP and also present a mathematical representation of the system we consider for control. In Section 3, we will examine the various options and choices and their implications for process control applications. In Section 4, we will present a few examples, including those involving both linear and nonlinear stochastic systems. In Section 5, we conclude the paper and discuss other control-related areas where ADP can potentially be useful in the process industries.

2. Background

2.1. Markov decision processes and approximate dynamic programming

Markov decision processes (MDPs) provide a framework for modeling real world processes that have a stage-wise structure. The stage can denote a time epoch or other quantities like location, processing step, etc. At any stage, the system is recognized as being in a state (designated as s), which is a set of attributes that aid decision-making. The set of all possible states is called state space (designated as S). Starting in state s belonging to S, there is a set of actions from which the decision-maker must choose. The set of all possible actions is called action space (A) and an element of the action space is denoted by a. When action a is taken in states, and the system transitions to the next stage, it ends up in a unique next state $s' \in S$ in the absence of any uncertainty. However, for stochastic problems, there is a set of possible next states for each state-action pair. The probability of transition to a particular next state in this case is governed by a state transition probability function, P. In the process, reward $r(s, a, s')$ is received, which is determined by the reward function$r$. The dependence of r on s' is often suppressed by taking a weighted average over all possible states at the next stage. At each stage, actions are taken so that the sum of stage-wise rewards is maximized. In the presence of uncertainty, the expected sum of rewards is maximized. When infinite stages are present, i.e., extremely large time horizon, the future rewards are often discounted using a discount factor γ. When the number of stages is infinite, the problem is called an infinite horizon MDP as opposed to a finite horizon MDP for finite number of stages. In most applications, a stage symbolizes a time epoch. Therefore, the term time epoch or time step is often used synonymously with 'stage'.

More formally, MDP is defined by a tuple (S, A, P, R, γ) where S is a set of states, A is a set of actions, P: $S \times A \times S \rightarrow [0,1]$ is a set of transition probabilities that describe the dynamic behavior of the modeled environment, $R : S \times A \times S \rightarrow \mathbb{R}$ denotes a reward model that determines the stage-wise reward when action$a$ is taken in state$s$ leading to next state s' and $\gamma \in [0,1)$ is the discount factor used to discount future rewards. A γ value close to 0, places very little weightage on future rewards, while γ close to 1 results in very little discounting.

One of the fundamental properties of the MDPs is that the transition and reward functions associated with the stage-wise transition of state are independent of the past states and actions. Referred to as the Markov property, this memory-less feature enables the decomposition of the overall optimization problem into separate stage-wise problems. This is accomplished by using a recursive relationship between the value of being in a state at any stage.

An important notion in this regard is the so called value function denoted by$V(s)$, which is defined as the (often discounted) sum of rewards over a time horizon which can be either finite or infinite (shown below) and discussed hereafter:

$$V^{\pi}(s) = E\left[\sum_{t=0}^{\infty}\gamma^t r(s_t, \pi(s_t))|s_0 = s\right] \tag{1}$$

where$t$ denotes the time epoch, s_t is the state at time t and π: $S \rightarrow A$, is the policy that dictates the choice of action for a given state at time t.

The goal is to find an optimal policy that maximizes the value function for all $s \in S$. This is achieved by solving the Bellman equation [8] for finite or infinite horizon problems. The optimal policy can be derived via dynamic programming. Let $a^*(s)$ be the optimal action to be taken when the system is in state s, independent of time t. $V^*(s)$ is called the optimal value function and is obtained as the solution to the (Bellman equation) (2), which must be solved for all $s \in S$:

$$V^*(s) = \max_{a \in A}\left\{r(s,a) + \gamma\sum_{s' \in S}p(s'|s,a)V^*(s')\right\} \quad \forall s \tag{2}$$

$$a^*(s) = \pi^*(s) \triangleq \arg\max_{a \in A}\left\{r(s,a) + \gamma\sum_{s' \in S}p(s'|s,a)V^*(s')\right\} \tag{3}$$

where symbol $p(.)$ denotes the probability of a quantity. It is well-known [9] that for infinite horizon problems, a stationary optimal policy of the form in (3) exists, where $V^*(s)$ is the average discounted infinite horizon reward obtained when the optimal policy is followed starting from s until infinity [9]. This implies that the state to action mapping in the form of optimal policy is independent of the time epoch. The existence of stationary optimal policy is conditioned on the properties of model elements. One of the sufficient conditions is that there be a finite action space A_s corresponding to each state $s \in S$, maximum attainable stage-wise reward is finite and discount factor $\gamma \in [0,1)$. The alternative sets of sufficient conditions for existence of a stationary optimal policy for discounted infinite horizon MDPs can be found in [9].

It must be noted that the set of Bellman equations also called optimality equation is difficult to solve analytically because of the presence of the max operator. One of the popular solution methods is called value iteration [9]: starting with an arbitrary value function