Proceedings of the ASME 2010 Dynamic Systems and Control Conference
DSCC2010
September 12-15, 2010, Cambridge, Massachusetts, USA

DSCC2010-4047

A ROLLOUT CONTROL ALGORITHM FOR DISCRETE-TIME STOCHASTIC SYSTEMS

Andreas A. Malikopoulos
Propulsion System Research Lab
General Motors Global Research & Development
Warren, MI 48090
andreas.malikopoulos@gm.com

ABSTRACT

The growing demand for making autonomous intelligent systems that can learn how to improve their performance while interacting with their environment has induced significant research on computational cognitive models. Computational intelligence, or rationality, can be achieved by modeling a system and the interaction with its environment through actions, perceptions, and associated costs. A widely adopted paradigm for modeling this interaction is the controlled Markov chain. In this context, the problem is formulated as a sequential decision-making process in which an intelligent system has to select those control actions in several time steps to achieve long-term goals. This paper presents a rollout control algorithm that aims to build an online decision-making mechanism for a controlled Markov chain. The algorithm yields a lookahead suboptimal control policy. Under certain conditions, a theoretical bound on its performance can be established.

1. INTRODUCTION

Sequential decision models [1, 2] are mathematical abstractions of situations in which decisions must be made in several decision epochs while incurring a certain cost (or reward) at each epoch. Each decision may influence the circumstances under which future decisions will be made, and thus, the decision maker must balance his/her desire to minimize (maximize) the cost (reward) of the present decision against his/her desire to avoid future situations where high cost is inevitable.

A large class of sequential decision-making problems under uncertainty can be solved using dynamic programming (DP) [3]. However, the computational cost of DP in some instances may be prohibitive and can grow intractably as the size of the problem increases. As an alternative approach to address this issue, Approximate Dynamic Programming (ADP) [4] is employed, providing suboptimal control methods for deterministic and stochastic problems. Rollout algorithms and model predictive control are two major methods within ADP with properties founded on policy iteration. The main idea of rollout algorithms [5-10] is to obtain an improved policy starting from some other suboptimal policy using a one-time policy improvement. It has been proposed by Abramson [11] and by Tesauro and Galperin [12] in the context of game-playing computer programs. In the latter, a backgammon position is evaluated by simulating many games starting from that position and the results are averaged. Model predictive control [13-17] is a popular approach in a variety of control system design contexts, and in particular, in chemical process control. It was motivated by the desire to introduce nonlinearities and constraints into the linear-quadratic control framework, while obtaining a suboptimal but stable closed-loop system.

Other alternatives for approaching these problems have been primarily developed in the field of Reinforcement Learning (RL) [4, 18, 19]. RL has aimed to provide algorithms, founded on DP, for learning suboptimal control policies when analytical methods cannot be used effectively, or the system's state transition probabilities are not known [20]. Although many of these algorithms are eventually guaranteed to find sub-optimal policies in sequential decision-making problems under uncertainty, their use of the accumulated data acquired over the learning process is inefficient, and they require a significant amount of experience to achieve acceptable performance [21]. This requirement arises due to the formation of these algorithms in deriving control policies without learning the system dynamics en route, that is, they do not solve the system identification problem simultaneously. In addition, RL algorithms are suited to problems in which the system needs to achieve particular "goal" states, which imposes limitations in employing efficiently these algorithms to solve particular problems.

The Predictive Optimal Decision-making (POD) learning model [22, 23] has aimed to address the system identification problem for a completely unknown system by learning in real time the system's evolution over a varying and unknown finite time horizon. The POD model has been employed in various applications towards making autonomous intelligent systems that can learn to improve their performance over time in stochastic environments. In the cart-pole balancing problem [23], an inverted pendulum was made capable of realizing the balancing control policy and turning into a stable system. In a vehicle cruise control implementation [23], an autonomous cruise controller was developed to learn to maintain the desired vehicle's speed at different road grades. POD has also taken steps toward development autonomous intelligent propulsion systems realizing their optimal operation with respect to the driver's driving style [22, 24].

In this paper, a rollout control algorithm that aims to build an online decision-making mechanism for controlled Markov chains is presented. The algorithm can be combined with the POD model to yield a lookahead suboptimal control policy that assesses the system output with respect to alternative control actions, and selecting those that optimize specified performance criteria. A theoretical bound on its performance is proven in Theorem 4.1, thus establishing that, under certain conditions, the lookahead control policy exists.

The remainder of the paper proceeds as follows: Section 2 establishes the mathematical framework of the controlled Markov chain. Section 3 reviews briefly the Predictive Optimal Decision-making (POD) computational model that aims to learn the transition probabilities and associated costs. Section 4 introduces the rollout control algorithm and formulates the theoretical bound on its performance. Concluding remarks are presented in Section 5.

2. PROBLEM FORMULATION

The stochastic system model establishes the mathematical framework for the representation of dynamic systems that evolve stochastically over time [2, 25, 26], that is, when incurring a stochastic disturbance or noise at time *k*, w_k, in their portrayal. The one-dimensional model is given by an equation of the form

$$s_{k+1} = f_k(s_k, a_k, w_k),\ k = 0,1,... \tag{1}$$

where s_k is the system's state that belongs to some state space $\mathcal{S} = \{1,2,...,N\}$, $N \in \mathbb{N}$, f_k is a function that describes how the system's state is updated, a_k is the control action, and w_k is the disturbance at time *k*. The sequence $\{w_k, k \geq 0\}$ is treated as a stochastic process, and the joint probability distribution of the random variables $w_0, w_1, ..., w_k$ is unknown for each *k*. The system output is represented by

$$y_k = h_k(s_k, v_k),\ k = 0,1,... \tag{2}$$

where y_k is the observation or system's output, h_k is a function that describes how the system output is updated, and v_k is the measurement error or noise. The sequence $\{v_k, k \geq 0\}$ is also considered a stochastic process with unknown probability distribution. We are interested in deriving a control policy so that a given performance criterion is optimized over all admissible policies *Π*. An admissible policy consists of a sequence of functions $\pi = \{\mu_0, \mu_1, ...\}$, where μ_k maps states s_k into actions $a_k = \mu_k(s_k)$.

The system's state s_k depends upon the input sequence $a_0, a_1, ...$ as well as the random variables $w_0, w_1, ...$, Eq. (1). Consequently, s_k is a random variable; the system output $y_k = h_k(s_k, v_k)$ is a function of the random variables $s_0, s_1, ..., v_0, v_1, ...$, and thus, is also a random variable. Similarly, the sequence of control actions $a_k = \mu(s_k)$, $\{a_k, k \geq 0\}$, constitutes a stochastic process.

Suppose that the previous values of the random variables s_m and a_m, $m \leq k-1$ are known. Then the conditional distribution of s_{k+1} given these values will be

$$\begin{aligned} &\mathbb{P}^{\pi}_{s_{k+1}|s_k,a_k}(s_{k+1} \mid s_k, ..., s_0, a_k, ..., a_0) = \\ &= \mathbb{P}^{\pi}_{w_k|s_k,a_k}(w_k \mid s_{k-1}, ..., s_0, a_{k-1}, ..., a_0). \end{aligned} \tag{3}$$

The conditional probability distribution of s_{k+1} given s_k and a_k can be independent of the previous values of states and control actions, if it is guaranteed that for every control policy π, w_k is independent of the random variables s_m and a_m, $m \leq k-1$. Kumar and Varaiya [26] proved that this property is imposed under the assumption that the following random variables $s_0, w_0, w_1, ...,\ v_0, v_1, ...,$ are all independent. The latter imposes a condition directly to the basic random variables which eventually yields that the state s_{k+1} depends only on s_k and a_k. Moreover, the conditional probability distributions do not depend on the control policy π, and thus the superscript π can be dropped

$$\begin{aligned} &\mathbb{P}_{s_{k+1}|s_k,a_k}(s_{k+1} \mid s_k, ..., s_0, a_k, ..., a_0) = \\ &= \mathbb{P}_{s_{k+1}|s_k,a_k}(s_{k+1} \mid s_k, a_k). \end{aligned} \tag{4}$$

A stochastic process $\{s_k, k \geq 0\}$ satisfying Eq. (4) is called a *Markov Process*. If the state space is discrete, then the process is defined as a controlled Markov chain.

The discrete-time, stationary controlled Markov chain is a stochastic dynamic system specified by a five-tuple $\{\boldsymbol{S}, \boldsymbol{\mathcal{A}}, \boldsymbol{A}, \mathbf{P}, R\}$, where

(a) $\boldsymbol{S} = \{1, 2, ..., N\}$ is the finite state space;

(b) $\boldsymbol{\mathcal{A}}$ is the compact action space;

(c) $\boldsymbol{A}$ is a family $\{\boldsymbol{A}(i) \mid i \in \boldsymbol{S}\}$ of nonempty measurable subsets of $\boldsymbol{\mathcal{A}}$, where $A(i)$ denotes the set of feasible actions when the system is at state $i \in \boldsymbol{S}$, with the property that the set

$$\boldsymbol{K} := \{(i,a) \mid i \in \boldsymbol{S}, a \in \boldsymbol{A}(i)\},$$

of feasible pairs is measurable subset of $\boldsymbol{S} \times \boldsymbol{\mathcal{A}}$ and contains the graph of a measurable function form $\boldsymbol{S}$ to $\boldsymbol{\mathcal{A}}$;

(d) $\mathbf{P}$ is the stochastic kernel on $\boldsymbol{S}$ given $\boldsymbol{K}$, that is, the transition probability of the system from state $i \in \boldsymbol{S}$ to $j \in \mathbf{S}$; and

(e) R is the measurable one-stage cost function, $R : \boldsymbol{K} \to \mathbb{R}$.

The evolution of the system occurs at each of a sequence of stages $k = 0, 1, ...$, and is portrayed by the sequence of the random variables s_k and a_k corresponding to the system's state and control action. At each stage, the controller observes the system's state $s_k = i \in \boldsymbol{S}$, and executes an action $a_k = a$, from the feasible set of actions $\boldsymbol{A}(i) \subseteq \boldsymbol{\mathcal{A}}$ at this state. At the next stage, the system transits to the state $s_{k+1} = j \in \boldsymbol{S}$ imposed by the conditional probability $P(j \mid i, a)$, and a cost $R(j \mid i, a)$ is incurred. After the transition to the next state has occurred, a new action is selected, and the process is repeated. The completed period of time over which the system is observed is called the *decision-making horizon* and is denoted by M. The horizon can be either finite or infinite; in this paper, we consider finite-horizon decision-making problems.

A control policy π determines the probability distribution of state process $\{s_k, k \geq 0\}$ and the control process $\{a_k, k \geq 0\}$. Different policies will lead to different probability distributions. In optimal control problems, the objective is to derive the optimal control policy that minimizes (maximizes) the accumulated cost (reward) incurred at each state transition per decision epoch. If a policy π is fixed, the cost incurred by π when the process starts from an initial state s_0 and up to the time horizon M is

$$J^{\pi}(s_0) = \sum_{k=0}^{M-1} R_k(s_{k+1} = j \mid s_k = i, a_k), \ \forall i, j \in \boldsymbol{\mathcal{S}}, \forall a_k \in A(i). \tag{5}$$

The accumulated cost $J_{\pi}(s_0)$ is a random variable since s_k and a_k are random variables. Hence the expected accumulated cost of a control policy is given by

$$J^{\pi}(s_0) = \underset{\substack{s_k \in \boldsymbol{\mathcal{S}} \\ a_k \in A(s_k)}}{E} \left\{ \sum_{k=0}^{M-1} R_k(s_{k+1} = j \mid s_k = i, a_k(s_k)) \right\}$$

$$= \underset{\substack{s_k \in \boldsymbol{\mathcal{S}} \\ \mu_k \in A(s_k)}}{E} \left\{ \sum_{k=0}^{M-1} R_k(s_{k+1} = j \mid s_k = i, \mu_k(s_k)) \right\}$$

$$= \sum_{k=0}^{M-1} p(s_{k+1} = j \mid s_k = i, \mu_k(s_k)) \cdot R(s_{k+1} = j \mid s_k = i, \mu_k(s_k)), \tag{6}$$

where the expectation is taken with respect to the probability distribution of $\{s_k, k \geq 0\}$ and $\{a_k, k \geq 0\}$ determined by the control policy π. The optimal policy $\pi^* = \{\mu_0^*, \mu_1^*, ..., \mu_M^*\}$ can be derived by

$$\pi^* = \arg\min_{\pi \in \Pi} J^{\pi}(s_0). \tag{7}$$

3. ONLINE SELF-LEARNING IDENTIFICATION

The problem of making autonomous intelligent systems is formulated as sequential decision-making under uncertainly. In this context, an intelligent system (decision maker), e.g., advanced propulsion systems, robot, automated manufacturing system, etc, has to select those actions in several time steps (decision epochs) to achieve long-term goals efficiently. This problem involves two major sub-problems: (a) the system identification problem, and (b) the stochastic control problem. The first is exploitation of the information acquired from the system output to identify its behavior, that is, how a state representation can be built by observing the system's state transitions. The second is assessment of the system output with respect to alternative control policies, and selecting those that optimize specified performance criteria.

The Predictive Optimal Decision-making (POD) learning model [23] is intended to address the system identification problem for a completely unknown system by learning in real time the system dynamics over a varying and unknown finite time horizon. The model embedded in the self-learning controller is constituted by a state representation which attempts to provide an efficient process in realizing the state transitions that occurred in the Markov domain. The model considers systems that their evolution can be modeled as a controlled Markov chain under the assumptions that the Markov chain is homogeneous, ergodic, and irreducible.

The learning process of the POD model transpires while the system interacts with its environment. Taken in conjunction with assigning values of the control actions from the feasible action space, $\mathcal{A}$, this interaction portrays the progressive enhancement of the controller's "knowledge" of the system's evolution with respect to the control actions. More precisely, at each of a sequence of decision epochs $k = 0,1,2,...M$, a state $s_k \in \mathcal{S}$ is introduced to the controller, and on that basis the controller selects an action, $a_k = \mu(s_k)$. This state arises as a result of the system's evolution. One epoch later, as a consequence of this action, the system transits to a new state $s_{k+1} = j \in \mathcal{S}$, and receives a numerical cost, $R_k(s_{k+1} = j \mid s_k = i, a_k) \in \mathbb{R}$.

At each epoch, the controller implements a mapping from the Cartesian product of the state space and action space to the set of real numbers, $\mathcal{S} \times \mathcal{A} \to \mathbb{R}$, by means of the costs that it receives. Similarly, another mapping from the Cartesian product of the state space and action space to the closed set [0,1] is executed, $\mathcal{S} \times \mathcal{A} \to [0,1]$, i.e., the transition probability matrix, $\mathbf{P}(\cdot,\cdot)$. The latter essentially perceives the incidence in which particular states or particular sequences of states arise.

The POD model possesses a structure that enables a convergent behavior of the conditional probabilities infused by the POD state-space representation to the stationary distribution. This behavior is desirable in the effort towards making autonomous intelligent systems that can learn to improve their performance over time in stochastic environments. The convergence of POD to the stationary distribution of the Markov state transitions has been proven in [27], hence establishing POD as a robust model.

As the process is stochastic, however, it is still necessary for the controller to build a decision-making mechanism to derive the control policy. This policy is expressed by means of a mapping from states to probabilities of selecting the actions, resulting in the minimum expected accumulated cost.

4. ROLLOUT CONTROL ALGORITHM

The objective of the control algorithm is to evaluate in real time the optimal action at each epoch not only for the current state, but also for the next two subsequent states over the following epochs. The requirement of real-time implementation imposes a computational burden in allowing the algorithm to look further ahead in time, thus evaluating an action over additional succeeding states.

Suppose that the current state is s_k and the following state given an action $a_k \in \mathcal{A}(s_k)$, is s_{k+1}. The immediate cost incurred by this transition is $R(s_{k+1} \mid s_k, a_k)$. The minimum expected cost for the next two subsequent states is perceived in terms of the magnitude, $V(s_{k+1})$, and is equal to

$$V(s_{k+1}) = \min_{a_{k+1} \in A(s_{k+1})} \underset{s_{k+2} \in \mathcal{S}}{E} \left\{ R(s_{k+2} \mid s_{k+1}, a_{k+1}) \right\}. \quad (8)$$

For the problem of optimal control of uncertain systems, which is treated in a stochastic framework, all uncertain quantities are described by probability distributions and the expected value of the overall cost is minimized. In this context, the control policy $\bar{\pi}$ realized by the algorithm is based on the minimax control approach, whereby the worst possible values of the uncertain quantities within the given set are assumed to occur. This essentially assures that the control policy will result in at most a maximum overall cost. Consequently, being at state s_k the control algorithm provides the policy $\bar{\pi} = \{\bar{\mu}_0, \bar{\mu}_1, ..., \bar{\mu}_{M-1}\}$, in terms of the values of the controllable variables as

$$\bar{\pi}(s_k) = \arg\min_{\bar{\mu}_k(s_k) \in A(s_k)} \max_{s_{k+1} \in \mathcal{S}} \left[R(s_{k+1} \mid s_k, a_k) + V(s_{k+1}) \right]. \quad (9)$$

To evaluate the efficiency of the algorithm, the establishment of a performance bound in terms of the accumulated cost over the decision epochs is necessary. The following Lemma (see, e.g. [1]) aims to provide a useful step toward presenting the main result (Theorem 3.1).

Lemma 4. 1: Let $f : \mathcal{S} \to [-\infty, \infty]$ and $g : \mathcal{S} \times \mathcal{A} \to [-\infty, \infty]$ be two functions. If

$$\min_{a \in \mathcal{A}} \left(g(i,a) \right) > -\infty, \forall i \in \mathcal{S} \quad (10)$$

then we have

$$\min_{\mu(i) \in \mathcal{A}} \max_{i \in \mathcal{S}} [f(i) + g(i, \mu(i))] = \max_{i \in \mathcal{S}} [f(i) + \min_{a \in \mathcal{A}} g(i,a)], \quad (11)$$

where the function $\mu : \mathcal{S} \to \mathcal{A}$, maps the state into action, that is, $a = \mu(i)$, and $\mathcal{S}, \mathcal{A}$ are the state and action space respectively.

Assumption 4. 1: The minimum expected cost $V(s_{k+1})$, Eq. (8), incurred at the decision epoch $k+1$ is bounded, that is, $V(s_{k+1}) > -\infty, \forall s_{k+1} \in \mathcal{S}$.

Theorem 4.1: The accumulated cost $\tilde{J}_k(s_k)$ incurred by the lookahead control policy $\bar{\pi} = \{\bar{\mu}_0, \bar{\mu}_1, ..., \bar{\mu}_{M-1}\}$, namely,

$$\bar{\pi}(s_k) = \arg\min_{\bar{\mu}_k(s_k) \in A(s_k)} \max_{s_{k+1} \in \mathcal{S}} \left[R_k(s_{k+1} \mid s_k, a_k) + \min_{a_{k+1} \in A(s_{k+1})} \underset{s_{k+2} \in \mathcal{S}}{E} \left\{ R_{k+1}(s_{k+2} \mid s_{k+1}, a_{k+1}) \right\} \right], \quad (12)$$

is bounded by the accumulated cost $J_k(s_k)$ incurred by the minimax control policy $\pi = \{\mu_0, \mu_1, ..., \mu_{M-1}\}$, namely,

$$\pi = \arg\min_{\bar{\mu}_k(s_k) \in A(s_k)} \max_{s_{k+1} \in \mathcal{S}} \left[R_k(s_{k+1} \mid s_k, a_k) + J_{k+1}(s_{k+1}) \right] \quad (13)$$

with probability 1.

Proof: Suppose that the chain starts at a state $s_0 = i, i \in \mathcal{S}$ at time $k = 0$ and ends up at $k = M$. We consider the problem of finding a policy $\pi = \{\mu_0, \mu_1, ..., \mu_{M-1}\}$ with $\mu_k(s_k) \in \mathcal{A}$ for all $s_k \in \mathcal{S}$ and k that minimizes the cost function

$$J^{\pi}(s_k) = \max_{s_{k+1} \in \mathcal{S}} \left[R_{M-1}(s_M \mid s_{M-1}, a_{M-1}) + \sum_{k=0}^{M-2} R_k(s_{k+1} \mid s_k, a_k) \right]. \quad (14)$$

The DP algorithm for this problem takes the following form starting from the tail sub-problem

$$J_M(s_M) = \min_{\mu_M(s_M) \in A(s_M)} \max_{s_{M+1} \in \mathcal{S}} \left[R_M(s_{M+1} \mid s_M, a_M) \right] = R_M(s_M) \text{, and} \quad (15)$$

$$J_k(s_k) = \min_{\mu_k(s_k) \in A(s_k)} \max_{s_{k+1} \in \mathcal{S}} \left[R_k(s_{k+1} \mid s_k, a_k) + J_{k+1}(s_{k+1}) \right], \quad (16)$$

where $R_M(s_M)$ is the cost of the terminal decision epoch.

Following the steps of the DP algorithm proposed by Bertsekas [1], the optimal accumulated cost $J^{\pi^*}(s_0)$ starting from the last decision epoch and moving backwards is

$$J^{\pi^*}(s_0) = \min_{\mu_0(s_0) \in A(s_0)} ... \min_{\mu_{M-1}(s_{M-1}) \in A(s_{M-1})} \max_{s_0 \in \mathcal{S}} ... \max_{s_M \in \mathcal{S}} \left[R_{M-1}(s_M \mid s_{M-1}, a_{M-1}) + \sum_{k=0}^{M-2} R_k(s_{k+1} \mid s_k, a_k) \right]. \quad (17)$$

By applying Lemma 4. 1, we can interchange the min over $\mu_{\text{M-1}}$ and the max over $s_0, ..., s_{M-2}$. The necessary condition in Lemma 4. 1 is implied by Assumption 4. 1.

Equation (17) yields

$$J^{\pi^*}(s_0) = \min_{\mu_0(s_0) \in A(s_0)} ... \min_{\mu_{M-2}(s_{M-2}) \in A(s_{M-2})} \left[\max_{s_0 \in \mathcal{S}} ... \max_{s_{M-2} \in \mathcal{S}} \left[\sum_{k=0}^{M-3} R_k(s_{k+1} \mid s_k, a_k) + \max_{s_{M-1} \in \mathcal{S}} \left[R_{M-2}(s_{M-1} \mid s_{M-2}, a_{M-2}) + J_M(s_M) \right] \right] \right] \quad (18)$$

$$= \min_{\mu_0(s_0) \in A(s_0)} ... \min_{\mu_{M-2}(s_{M-2}) \in A(s_{M-2})} \left[\max_{s_0 \in \mathcal{S}} ... \max_{s_{M-2} \in \mathcal{S}} \left[\sum_{k=0}^{M-3} R_k(s_{k+1} \mid s_k, a_k) + J_{M-1}(s_{M-1}) \right] \right]. \quad (19)$$

By continuing backwards in similar way we obtain

$$J^{\pi^*}(s_0) = J_0(s_0) . \quad (20)$$

Consequently, an optimal policy for the minimax problem can be constructed by minimizing the right hand side of Eq. (14).

Performing the same task as we did with the DP algorithm by starting from the last epoch of the decision-making process and moving backwards, the accumulated cost, $\tilde{J}_k(s_k)$, incurred by the control policy $\bar{\pi} = \{\bar{\mu}_0, \bar{\mu}_1, ..., \bar{\mu}_{M-1}\}$ is

$$\tilde{J}_M(s_M) = = \min_{\bar{\mu}_M(s_M) \in A(s_M)} \max_{s_{M+1} \in \mathcal{S}} \left[R_M(s_{M+1} \mid s_M, a_M) + \min_{a_{M+1} \in A(s_{M+1})} \underset{s_{M+2} \in \mathcal{S}}{E} \{ R_{M+1}(s_{M+2} \mid s_{M+1}, a_{M+1}) \} \right]$$
$$= R_M(s_{M+1} \mid s_M, a_M) = R_M(s_M) = J_M(s_M), \quad (21)$$

$R_{M+1}(s_{M+2} \mid s_{M+1}, a_{M+1}) = 0$ since the terminal epoch is at $k = M$.

$$\tilde{J}_{M-1}(s_{M-1}) = = \min_{\bar{\mu}_{M-1}(s_{M-1}) \in A(s_{M-1})} \max_{s_M \in \mathcal{S}} \left[R_{M-1}(s_M \mid s_{M-1}, a_{M-1}) + \min_{a_M \in A(s_M)} \underset{s_{M+1} \in \mathcal{S}}{E} \{ R_M(s_{M+1} \mid s_M, a_M) \} \right] + \tilde{J}_M(s_M) \quad (22)$$

$$= \min_{\bar{\mu}_{M-1}(s_{M-1}) \in A(s_{M-1})} \max_{s_M \in \mathcal{S}} \left[R_{M-1}(s_M \mid s_{M-1}, a_{M-1}) \right] + \tilde{J}_M(s_M) , \quad (23)$$

Similarly, $R_M(s_{M+1} \mid s_M, a_M) = 0$ since the terminal epoch is at $k = M$. Consequently,

$$\tilde{J}_{M-1}(s_{M-1}) = \min_{\bar{\mu}_M(s_M) \in A(s_M)} \max_{s_{M+1} \in \mathcal{S}} \left[R_M(s_{M+1} \mid s_M, a_M) + \tilde{J}_M(s_M) \right] = J_{M-1}(s_{M-1}), \quad (24)$$

since $\tilde{J}_M(s_M)$ is a constant quantity.

$$\tilde{J}_{M-2}(s_{M-2}) = = \min_{\bar{\mu}_{M-2}(s_{M-2}) \in A(s_{M-2})} \max_{s_{M-1} \in \mathcal{S}} \left[R_{M-2}(s_{M-1} \mid s_{M-2}, a_{M-2}) + \min_{a_{M-1} \in A(s_{M-1})} \underset{s_M \in \mathcal{S}}{E} \{ R_{M-1}(s_M \mid s_{M-1}, a_{M-1}) \} \right] + +\tilde{J}_{M-1}(s_{M-1}). \quad (25)$$

However,

$$\min_{\bar{\mu}_{M-2}(s_{M-2})\in A(s_{M-2})} \max_{s_{M-1}\in \mathcal{S}}\left[R_{M-2}(s_{M-1}\mid s_{M-2},a_{M-2})+\min_{a_{M-1}\in A(s_{M-1})}\mathop{E}_{s_M\in\mathcal{S}}\left\{R_{M-1}(s_M\mid s_{M-1},a_{M-1})\right\}\right]\leq \min_{\mu_{M-2}(s_{M-2})\in A(s_{M-2})}\max_{s_{M-1}\in\mathcal{S}}\left[R_{M-2}(s_{M-1}\mid s_{M-2},a_{M-2})\right], \tag{26}$$

since the LHS of the inequality will minimize a cost which is not only maximum over the cost incurred when the chain transits from s_{M-2} to s_{M-1} but also minimum over the cost incurred when the chain transits from s_{M-1} to s_M. So, the LHS can be at most equal to the cost which is maximum over the transition from s_{M-2} to s_{M-1}.

Consequently, comparing the accumulated cost of the control policy $\tilde{J}_k(s_k)$ in Eq. (25) with the one resulted from the DP at the same decision epoch, namely,

$$J_{M-2}(s_{M-2})=\min_{\mu_{M-2}(s_{M-2})\in A(s_{M-2})}\max_{s_{M-1}\in\mathcal{S}}\left[R_{M-2}(s_{M-1}\mid s_{M-2},a_{M-2})+J_{M-1}(s_{M-1})\right] \tag{27}$$

we conclude that

$$\tilde{J}_{M-2}(s_{M-2})\leq J_{M-2}(s_{M-2}). \tag{28}$$

By continuing backward with similar arguments, we have

$$\tilde{J}_0(s_0)\leq J_0(s_0)=J^{\pi^*}(s_0). \tag{29}$$

Consequently, the accumulated cost resulting from the control policy $\bar{\pi}=\{\bar{\mu}_0,\bar{\mu}_1,...,\bar{\mu}_{M-1}\}$ is bounded by the accumulated cost of the optimal minimax control policy with probability 1.

□

5. CONCLUDING REMARKS

We presented the theoretical framework and a rollout control algorithm toward making autonomous intelligent systems that can learn their optimal operation in real time. The evolution of the system was modeled as a controlled Markov chain, and the task of deriving a control policy was formulated as a sequential decision-making problem under uncertainty. The algorithm comprises the decision-making mechanism that solves the stochastic control problem by utilizing accumulated data acquired as the system interacts with its environment. The solution of the algorithm has a theoretical performance bound that is superior to that of the solution provided by the one-step minimax control algorithm (Theorem 4.1).

The research presented here considered the approximate solution of a discrete optimization problem using procedures capable of magnifying the effectiveness of any given heuristic algorithm through sequential application. In particular, the problem was embedded within a dynamic programming framework, and a two-step rollout algorithm was introduced related to notions of policy iteration. Future research should explore the impact of the number of time steps that the algorithm can look forward in time on its performance bound.

REFERENCES

[1] Bertsekas, D. P., *Dynamic Programming and Optimal Control (Volumes 1 and 2)*, Athena Scientific, September 2001.

[2] Bertsekas, D. P. and Shreve, S. E., *Stochastic Optimal Control: The Discrete-Time Case*, 1st edition, Athena Scientific, February 2007.

[3] Bellman, R., *Dynamic Programming*. Princeton, NJ, Princeton University Press, 1957.

[4] Bertsekas, D. P. and Tsitsiklis, J. N., *Neuro-Dynamic Programming (Optimization and Neural Computation Series, 3)*, 1st edition, Athena Scientific, May 1996.

[5] Bertsekas, D. P., Tsitsiklis, J. N., and Wu, C., "Rollout Algorithms for Comninatorial Optimization," *Heuristics*, vol. 3, pp. 245-262, 1997.

[6] Bertsekas, D. P. and Castanon, D. A., "Rollout Algorithms for Stochastic Scheduling Problems," *Heuristics*, vol. 5, pp. 89-108, 1999.

[7] Secomandi, N., "Comparing Neuro-Dynamic Programming Algorithms for the Vehicle Routing Problem with Stochastic Demands," *Computers and Operations Research*, vol. 27, pp. 1201-1225, 2000.

[8] McGovern, A., Moss, E., and Barto, A., "Building a Basic Building Block Scheduler Using Reinforcement Learning and Rollouts," *Machine Learning*, vol. 49, pp. 141-160, 2002.

[9] Bertsimas, D. and Popescu, I., "Revenue Management in a Dynamic Network Environment," *Transportation Science*, vol. 37, pp. 257-277, 2003.

[10] Tu, F. and Pattipati, K. R., "Rollout Strategies for Sequential Fault Diagnosis," *IEEE Trans on Systems, Man and Cybernetics, Part A*, pp. 86-99, 2003.

[11] Abramson, B., "Expected-Outcome: A General Model of Static Evaluation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 182-193, 1990.

[12] Tesauro, G. and Galperin, G., "On-line Policy Improvement Using Monte-Carlo Search," *Advances in Neural Information Processing*, vol. 9, pp. 1068-1074, 1996.

[13] Morari, M. and Lee, J. H., "Model Predictive Control: Past, Present, and Future," *Computers and Chemical Engineering*, vol. 23, pp. 667-682, 1999.

[14] Mayne, D. Q., Rawlings, J. B., Rao, C. V., and Scokaert, P. O. M., "Constrained Model Predictive Control: Stability and Optimality," *Automatica*, vol. 36, pp. 789-814, 2000.

[15] Rawlings, J. B., "Tutorial Overview of Model Predictive Control," *Control Systems Magazine*, vol. 20, pp. 38-52, 2000.

[16] Findeisen, R., Imsland, L., Allgower, F., and Foss, B. A., "State and Output Feedback Nonlinear Model Predictive Control: An Overview," *European Journal of Control*, vol. 9, pp. 190-205, 2003.

[17] Qin, S. J. and Badgwell, T. A., "A Survey of Industrial Model Predictive Control Technology," *Control Engineering Practice*, vol. 11, pp. 733-764, 2003.

[18] Sutton, R. S. and Barto, A. G., *Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning)*, The MIT Press, March 1998.

[19] Gosavi, A., *Simulation-Based Optimization: Parametric Optimization Techniques and Reinforcement Learning*, 1st edition, Springer, June 30, 2003.

[20] Borkar, V. S., "A Learning Algorithm for Discrete-Time Stochastic Control," *Probability in the Engineering and Information Sciences*, vol. 14, pp. 243-258, 2000.

[21] Kaelbling, L. P., Littman, M. L., and Moore, A. W., "Reinforcement Learning: A Survey," *Journal of Artificial Intelligence Research*, vol. 4, 1996.

[22] Malikopoulos, A. A., Real-Time, Self-Learning Identification and Stochastic Optimal Control of Advanced Powertrain Systems, Ph.D. Dissertation, Department of Mechanical Engineering, University of Michigan, Ann Arbor, USA, 2008.

[23] Malikopoulos, A. A., Papalambros, P. Y., and Assanis, D. N., "A Real-Time Computational Learning Model for Sequential Decision-Making Problems Under Uncertainty," *ASME J. Dyn. Sys., Meas., Control*, vol. 131, 4, pp. 041010-(8), 2009.

[24] Malikopoulos, A. A., Assanis, D. N., and Papalambros, P. Y., "Real-Time Self-Learning Optimization of Diesel Engine Calibration," *ASME J. Eng. Gas Turbines Power*, vol. 131, 2, pp. 022803(7), 2009.

[25] Kushner, H. J., *Introduction to Stochastic Control*, Holt, Rinehart and Winston, 1971.

[26] Kumar, P. R. and Varaiya, P., *Stochastic Systems*, Prentice Hall, June 1986.

[27] Malikopoulos, A. A., "Convergence Properties of a Computational Learning Model for Unknown Markov Chains," *ASME J. Dyn. Sys., Meas., Control*, vol. 131, No. 4, pp. 041011(7), 2009.