Lebesgue-Sampling-Based Optimal Control Problems With Time Aggregation

Yan-Kai Xu and Xi-Ren Cao, *Fellow, IEEE*

***Abstract*—We formulate the Lebesgue-sampling-based optimal control problem. We show that the problem can be solved by the time aggregation approach in Markov decision processes (MDP) theory. Policy-iteration-based and reinforcement-learning-based methods are developed for the optimal policies. Both analytical solutions and sample-path-based algorithms are given. Compared to the periodic-sampling scheme, the Lebesgue sampling scheme improves system performance.**

***Index Terms*— Aggregation, Markov decision processes (MDPs), performance potentials, reinforcement learning.**

I. Introduction

In a digital control system, the states are usually sampled periodically with a fixed-length sampling interval [5]. Because true continuous control with digital computers is impossible, this periodic sampling schema is widely used in computer control systems in engineering. The problems with such a sampling scheme are relatively easy to analyze. However, implementing sampling and determining control actions are usually expensive. To save computer power and resources, varying sampling intervals may be used. One such alternative approach is to sample the system whenever the signal (e.g., state) passes some prespecified levels. This type of sampling is natural when digital sensors are used, and it is closer to human behavior as a controller. This is called Lebesgue sampling [4], event-based sampling [3], or event-triggered sampling [6]. A comparison between periodic and Lebesgue sampling for one-dimensional systems can be found in [4], among others, which shows that with impulsive control, Lebesgue sampling may reduce the sampling frequency to achieve the same performance as periodic sampling. A brief explanation for this is that with periodic sampling, in some cases, at the next sampling time, the system state needs to be measured and a new control action has to be chosen and applied even if the system state has not been changed much, and in some other cases, no new action can be applied even if the system state already has a big change because the next sampling time has not been reached.

Recent works in this area focus on sampling schemes [15], [20] and applications to networked control systems [14], [16], manufacturing systems [18], communication systems [17], and so on. The controls adopted in these papers are often simple, e.g., impulsive control [11], on–off control [23], or heuristic PID control [3]. The optimal control problem with Lebesgue sampling is not generally formulated and well studied, probably because of the difficulty involved in analyzing this type of problem.

In this paper, we formulate the optimal control problem with Lebesgue sampling and provide a solution to the problem. The solution is based on the recently developed time aggregation (TA) approach in discrete-time Markov decision processes (MDPs) [10]. In this paper, we extend the TA approach to the Lebesgue-sampling-based control problems. There are two main differences between this problem and the problem considered in [10]: Lebesgue sampling deals with continuous time and continuous states, while in [10], discrete time and discrete states are considered. In addition, with Lebesgue sampling, the control determined at a sampling point is continuously applied to the system until the next sampling point, while in [10], the action at an embedded point only applies to the embedded point itself. We show that even with these additional features, the principle in [10] applies to our problem as well.

The fundamental idea is as follows. At the sampling points, the system states form an embedded discrete-time Markov chain. With a properly defined cost function, this embedded Markov chain has the same performance as the original continuous-time dynamic system. Thus, the problem becomes to optimize the performance of this embedded chain. The difficulty comes from the fact that this "properly defined" cost function depends on the controls at other states, and therefore the conventional MDP solution methods cannot be applied. However, we can show that the optimization problem of the embedded chain can be changed to another equivalent MDP problem, and therefore standard solution techniques such as policy iteration and reinforcement learning [24] can be used to solve the problem. Lebesgue-sampling-based control can also be viewed as a special case of the event-based optimization formulated in [9].

After presenting the theoretical analysis based on the above ideas, we provide some examples—analytical, numerical, and sample-path-based (with learning)—to illustrate the results. The examples show that Lebesgue sampling does obtain better performance than periodic sampling with the same sampling frequency.

Manuscript received November 26, 2008; revised August 11, 2009; April 05, 2010; May 28, 2010; and June 28, 2010; accepted June 28, 2010. Date of publication September 07, 2010; date of current version May 11, 2011. This work was supported by the Hong Kong UGC under Grants 610806, 610808, and 610809 and by the National Natural Science Foundations of China under Grants 60574064 and 60736027. Recommended by Associate Editor C.-H. Chen.

Y.-K. Xu is with the Beijing GeoScience Center, Schlumberger Ltd., Beijing 100084, China, and also with the Center for Intelligent and Networked Systems (CFINS), Tsinghua University, Beijing 100084, China (e-mail: xuyankai@gmail.com).

X.-R. Cao is with the Department of Automation and Department of Finance, Shanghai Jiao-Tong University, Shanghai 200240, China, and also with the Institute of Advanced Study, The Hong Kong University of Science and Technology, Hong Kong (e-mail: eecao@ust.hk).

Digital Object Identifier 10.1109/TAC.2010.2073610

This paper is organized as follows. In Section II, we formulate the optimal control problem with Lebesgue sampling. In Section III, we study this optimal control problem with the time aggregation approach; an equivalent MDP is found so that the standard optimization methods can be applied to solve the problem. A policy iteration algorithm is developed in this section. In Section IV, analytical solutions are given for some special cases. In Section V, sample-path-based learning approaches, including both the Q-factor-based policy iteration and the State-Action-Reward-State-Action (SARSA) algorithm, are proposed for general cases. In Section VI, results on periodic-sampling-based optimal control model are briefly reviewed, for a comparison to Lebesgue sampling. In Section VII, several examples are given to illustrate the results.

II. Problem Formulation

Consider a one-dimensional continuous-time nonlinear control system

$$\mathrm{d}x = \mu(x,u)\mathrm{d}t + \sigma \mathrm{d}v \tag{1}$$

where $x = x(t) \in \mathcal{R}$ is the system state at time t, $u = u(t) \in U \subseteq \mathcal{R}^n$ is the control variable (or action) at time t, with U denoting a control set, $v = v(t)$ is a Wiener process (under a probability measure denoted as $\mathcal{P}$), and σ is a constant. $\mu(\cdot,\cdot) : \mathcal{R} \times U \to \mathcal{R}$ is a scalar function of (x,u), $x \in \mathcal{R}$ and $u \in U$. The cost function associated with control variable u is denoted as a Lebesgue measurable function $f^u(x)$. The goal of the optimal control problem is to determine a control law $u(t)$, $t \in [0,\infty)$, which may depend on $x(t)$, that minimizes the long-run average performance defined as

$$\eta^u = \lim_{T\to\infty} \frac{1}{T} \mathrm{E}\left\{ \int_0^T f^u\left(x(t)\right)\mathrm{d}t | x(0) \right\} \tag{2}$$

where "E" denotes the expectation with respect to $\mathcal{P}$. We assume that the system under $u(t)$ is stable [21], and that the performance η^u does not depend on the initial state $x(0)$.

With digital technology, we need to take samples of the system state $x(t)$. With Lebesgue sampling, we define a finite event set: $\mathcal{D} = \{1,\cdots,D\}$. Each event $d \in \mathcal{D}$ corresponds to a prespecified state value denoted as x_d. The set of state values corresponding to the event set $\mathcal{D}$ is denoted as $\mathcal{X}_{\mathcal{D}} = \{x_d : d \in \mathcal{D}\} \subset \mathcal{R}$. For convenience, we also call $\mathcal{X}_{\mathcal{D}}$ the set of events. Without loss of generality, we assume $x_1 < x_2 < \cdots < x_D$ and $x_1 < 0 < x_D$. If at some time t, the system state "reaches" x_d (meaning $x(t) = x_d$ and $x(t-) \neq x_d$), we say that event d occurs at t.

Now, we consider a sample path $\omega = \{x(t), t \geq 0\}$. We assume that the sample path starts from an event, i.e., $x(0) \in \mathcal{X}_{\mathcal{D}}$. Let $t_0 = 0$, and for $i = 1,2,\ldots$, define

$$t_i = \min\left\{t : t > t_{i-1}, x(t) \in \mathcal{X}_{\mathcal{D}}, x(t) \neq x(t_{i-1})\right\}. \tag{3}$$

t_i is the ith sampling point. Denote the event at t_i as $d_i \in \mathcal{D}$. This definition excludes the case $d_{i+1} = d_i$, and $d_{i+1} - d_i \in \{-1,1\}$; $\{d_i, i = 0,1,2,\cdots\}$ forms an embedded chain.

An alternative is to define the occurrence time epoch of events as follows:

$$t_i = \min\left\{t : t > t_{i-1}, x(t) \in \mathcal{X}_{\mathcal{D}}, x(t-) \neq x(t)\right\}. \tag{4}$$

This definition allows $d_{i+1} = d_i$. However, this definition may not properly define a discrete-time embedded chain. For example, if $\mu(x,u) = \mu$ is a constant in the system dynamic (1), then the state process becomes a Brownian motion with drift coefficient μ and diffusion coefficient σ. In this case, we have the following theorem (for a proof, see Appendix A).

Theorem 2.1: With definition (4) and $\mu(x,u) = \mu$, we have $\mathrm{E}\{t_{i+1} - t_i\} = 0$.

From this theorem, events occur infinitely often in any small time interval. Thus, with definition (4), the resulting embedded chain is infinitely dense and cannot be handled. Therefore, we adopt (3) instead of (4) to obtain a proper discrete-time embedded chain.

In Lebesgue-sampling-based (or event-based) formulation, control action is updated only when an event occurs. The action adopted at time t_i is denoted as u_i, which remains unchanged until t_{i+1}, i.e., $u(t) = u_i$ for $t \in [t_i, t_{i+1})$. Thus, the system dynamic becomes

$$\mathrm{d}x = \mu(x,u_i)\mathrm{d}t + \sigma\mathrm{d}v, \qquad t_i \leq t < t_{i+1}. \tag{5}$$

The control action u_i can be determined according to an admissible control law (or called a policy) denoted as $u_i = \mathbf{u}(d_i)$, $d_i \in \mathcal{D}$, $u_i \in U$, $i = 0,1,\ldots$ Action u_i is determined by event d_i, thus $\mathbf{u}$ is an event-based policy. We assume that the number of available actions is finite, i.e., U is a finite set. The problem becomes to find an event-based policy $\mathbf{u}(\cdot)$ so that the performance (2) is minimized, subject to the system dynamic (5).

III. Time Aggregation Approach

The system in an optimal control problem is often modeled as a Markov process [26], [27]. Thus, the optimization methods for Markov systems can be applied. In this paper, we solve the optimal control problem with Lebesgue sampling by using the time aggregation approach developed in [10] (also see [9]).

Consider the system (5) under a policy $u = \mathbf{u}(d)$, $d \in \mathcal{D}$. The state process $x(t)$, $t \in [0,\infty)$, of the system described in (5) is not a Markov process because the control variable at $t \in [t_i, t_{i+1})$, $u(t) = u_i = \mathbf{u}(d_i)$, depends on the state at time $t_i \leq t$. However, $x(t)$ is a semi-Markov process because at any sampling point t_i, $i = 0,1,\ldots$, the future of the system depends only on $x(t_i) = x_{d_i}$, and the embedded chain $\{d_i, i = 0,1,2,\cdots\}$ is a Markov chain with state space $\mathcal{D}$. Let $P^{\mathbf{u}} = \{p^{\mathbf{u}(d)}(d'|d)\}_{d,d'\in\mathcal{D}}$ denote the transition probability matrix of this embedded Markov chain under policy $\mathbf{u}$. From (3), for a stable system, we have

$$p^{\mathbf{u}(d)}(d'|d) \begin{cases} > 0, & d' = d-1 \text{ or } d+1 \\ = 0, & \text{otherwise} \end{cases} \tag{6}$$

for $d = 2,3,\cdots,D-1$, and $p^{\mathbf{u}(1)}(2|1) = 1$, $p^{\mathbf{u}(D)}(D-1|D) = 1$. Obviously, this embedded Markov chain is an irreducible *periodic* chain with period 2 under any policy $\mathbf{u}$. For such a periodic chain, there is a unique invariant probability (row) vector $\pi^{\mathbf{u}}$ satisfying $\pi^{\mathbf{u}}P^{\mathbf{u}} = \pi^{\mathbf{u}}$ and $\pi^{\mathbf{u}}e = e$, where $e = (1,1,\ldots,1)^{\mathrm{T}}$ is a D-dimensional (column) vector with all components 1, with "T" denoting transpose.

Most results in MDPs in the literature are developed for aperiodic chains because the proofs of these results for periodic chains are technically involved. In this paper, we will verify that the policy iteration approach and reinforcement learning algorithms we are going to use for our problem are valid even for periodic chains.

Consider the embedded Markov chain $\{d_i, i = 0, 1, 2, \cdots\}$. Our first step is to find a cost function for the embedded Markov chain $\{d_i, i = 0, 1, 2, \cdots\}$ so that its average performance is the same as that of original system (2) and (5). The sample path of the system is divided into segments by the embedded points. Let $\zeta_i = \{x(t), t_i \le t < t_{i+1}\}$ denote the ith segment. The sample path can be written as $\omega = \{\zeta_0, \zeta_1, \cdots\}$. Consider a segment ζ_i starting from event d_i with action u_i, and define the quantities

$$h_f^{u_i}(\zeta_i) = \int_{t_i}^{t_{i+1}} f^{u_i}(x(t))\,\mathrm{d}t \tag{7}$$

and

$$H_f^u(d) = \mathrm{E}\left\{h_f^{u_i}(\zeta_i)|d_i = d, u_i = u\right\}. \tag{8}$$

Then, $H_f^u(d)$ is the expected cost received on a segment starting from event d with action u. For any policy $\mathbf{u}$, let

$$H_f^{\mathbf{u}} = \left[H_f^{\mathbf{u}(1)}(1), H_f^{\mathbf{u}(2)}(2), \cdots, H_f^{\mathbf{u}(D)}(D)\right]^{\mathrm{T}}. \tag{9}$$

We will use the notation h_1 and H_1 to denote the above quantities for the constant cost function $f^u(x) = 1$, for all $x \in \mathcal{R}$ and all $u \in U$. Thus, $h_1^{u_i}(\zeta_i) = t_{i+1} - t_i$ is the length of the ith segment, which depends on u_i implicitly, and

$$H_1^u(d) = \mathrm{E}\{t_{i+1} - t_i|d_i = d, u_i = u\} \tag{10}$$

is the average length of the segment starting from event d with action u. We also have $H_1^{\mathbf{u}} = [H_1^{\mathbf{u}(1)}(1), \cdots, H_1^{\mathbf{u}(D)}(D)]^{\mathrm{T}}$. From stability, it is clear that $|H_1^u(d)| < \infty$. We also assume that $|H_f^u(d)| < \infty$, which is satisfied, e.g., for any bounded function f.

Applying the strong law of large numbers, we obtain the performance of the system under policy $\mathbf{u}$ (cf. [12, Eq. (19)])

$$\begin{aligned}\eta^{\mathbf{u}} &= \lim_{T\to\infty} \frac{\int_0^T f^u(x(t))\,\mathrm{d}t}{T} = \lim_{l\to\infty} \frac{\frac{1}{l}\sum_{i=0}^{l-1} h_f^{u_i}(\zeta_i)}{\frac{1}{l}\sum_{i=0}^{l-1} h_1^{u_i}(\zeta_i)}\\ &= \frac{\pi^{\mathbf{u}} H_f^{\mathbf{u}}}{\pi^{\mathbf{u}} H_1^{\mathbf{u}}} = \pi^{\mathbf{u}} \frac{H_f^{\mathbf{u}}}{\bar{t}^{\mathbf{u}}} \qquad \text{w.p.1}\end{aligned} \tag{11}$$

where $\bar{t}^{\mathbf{u}} = \pi^{\mathbf{u}} H_1^{\mathbf{u}}$ is the average length of segments, or the average sampling interval, under policy $\mathbf{u}$. If we define a cost vector for the embedded Markov chain as

$$\bar{f}^{\mathbf{u}} = \frac{H_f^{\mathbf{u}}}{\bar{t}^{\mathbf{u}}} \tag{12}$$

then the embedded chain has the same performance $\eta^{\mathbf{u}} = \pi^{\mathbf{u}} \bar{f}^{\mathbf{u}}$ as the original system (2) and (5) under policy $\mathbf{u}$. Therefore, the optimal control problem of Lebesgue sampling system (5) with performance (2) is equivalent to the optimization of the embedded Markov chain with transition probability matrix $P^{\mathbf{u}}$ in (6) and cost function $\bar{f}^{\mathbf{u}}$ defined in (12). From a sample path point of view, it looks as if the total cost on a segment is "aggregated" onto the embedded point that starts the segment, thus we call this approach "time aggregation." This approach was proposed in [10] for discrete-time Markov chains; in this paper, we extend it to continuous-time systems.

However, we have a major obstacle here: According to (12), the cost at event d of the equivalent embedded chain, $\bar{f}^{\mathbf{u}}(d)$, depends on the average length $\bar{t}^{\mathbf{u}}$, which depends on the actions taken at events other than d. This violates the formulation of the standard MDPs (in which the cost at state x depends on the action taken at x, not on the policy [19]), and there is no simple solution to such a problem. The situation is similar to the problem studied in [10], where a discrete-time Markov chain is assumed to be controllable (in terms of transition probabilities) only when it is in a subset of the state space. The same obstacle was encountered, and it was solved by constructing another equivalent MDP problem that can be solved by the standard methods in MDPs.

There are, however, two major differences between our current problem and that in [10]. First, the problem in this paper is with continuous time, while that in [10] is with discrete time. Second, in this paper, action u_i is applied to the system during the whole segment ζ_i, while in [10], the action at an embedded point only affects the transition at this point and does not affect the transitions inside the segment. In the following, we will develop the time aggregation approach for our current problem, which can be viewed as an extension of the results in [10].

From (6) and (9), the dth components of $H_1^{\mathbf{u}}$ and $H_f^{\mathbf{u}}$ and the dth row of $P^{\mathbf{u}}$ depend only on action u, which is taken when event d occurs, and we denote them as $H_1^u(d)$, $H_f^u(d)$, and $p^u(d'|d)$, $d, d' \in \mathcal{D}$. Define a new cost function for the embedded chain

$$r_\delta^u(d) = H_f^u(d) - \delta H_1^u(d), \qquad d \in \mathcal{D} \tag{13}$$

where δ is a real parameter. Set $r_\delta^{\mathbf{u}} = (r_\delta^{\mathbf{u}(1)}(1), r_\delta^{\mathbf{u}(2)}(2), \ldots, r_\delta^{\mathbf{u}(D)}(D))^{\mathrm{T}}$. From (11), the average performance of the embedded chain with the newly defined cost function $r_\delta^{\mathbf{u}}$ is

$$\lambda_\delta^{\mathbf{u}} := \pi^{\mathbf{u}} r_\delta^{\mathbf{u}} = \pi^{\mathbf{u}} H_1^{\mathbf{u}}(\eta^{\mathbf{u}} - \delta). \tag{14}$$

We have the following simple theorem, which plays a crucial role in developing policy iteration algorithms on the newly defined MDP for our optimal control problem.

Theorem 3.1:

i) Policy $\mathbf{u}'$ is better than $\mathbf{u}$ for the embedded Markov chain with cost function (13) and $\delta = \eta^{\mathbf{u}}$ if and only if $\mathbf{u}'$ is better than $\mathbf{u}$ for the embedded Markov chain with cost function (12).

ii) Policy $\mathbf{u}^*$ is optimal for the embedded Markov chain with cost function (12) if and only if $\mathbf{u}^*$ is optimal for the embedded Markov chain with cost function (13) and $\delta = \eta^{\mathbf{u}^*}$.

Proof:

i) From (14), with $\delta = \eta^{\mathbf{u}}$, we have $\lambda_\delta^{\mathbf{u}'} = \pi^{\mathbf{u}'} H_1^{\mathbf{u}'}(\eta^{\mathbf{u}'} - \eta^{\mathbf{u}})$ and $\lambda_\delta^{\mathbf{u}} = 0$. Because $\pi^{\mathbf{u}'} H_1^{\mathbf{u}'} > 0$ for any $\mathbf{u}'$, we conclude that $\eta^{\mathbf{u}'} < \delta = \eta^{\mathbf{u}}$ if and only if $\lambda_\delta^{\mathbf{u}'} = \pi^{\mathbf{u}'} H_1^{\mathbf{u}'}(\eta^{\mathbf{u}'} - \eta^{\mathbf{u}}) < 0 = \lambda_\delta^{\mathbf{u}}$.

ii) From (14), with $\delta = \eta^{\mathbf{u}^*}$, we have $\lambda_\delta^{\mathbf{u}} = \pi^{\mathbf{u}} H_1^{\mathbf{u}}(\eta^{\mathbf{u}} - \eta^{\mathbf{u}^*})$ and $\lambda_\delta^{\mathbf{u}^*} = 0$. Therefore, $\eta^{\mathbf{u}} \geq \delta = \eta^{\mathbf{u}^*}$ for all policy $\mathbf{u}$ if and only if $\lambda_\delta^{\mathbf{u}} \geq 0 = \lambda_\delta^{\mathbf{u}^*}$ for all $\mathbf{u}$. ■

Note that the newly defined optimization problem also does not satisfy the standard MDP formulation. In Theorem 3.1.i, we fix a policy $\mathbf{u}$ and compare any other policy $\mathbf{u}'$ to this particular policy $\mathbf{u}$. Therefore, $\eta^{\mathbf{u}}$ is a constant in the comparison, and the cost function $r_\delta^u(d) = H_f^u(d) - \eta^{\mathbf{u}} H_1^u(d)$, $d \in \mathcal{D}$, depends only on action d (not on other actions). A similar explanation holds for Theorem 3.1.ii.

When $\delta = \eta^{\mathbf{u}}$, the cost function (13) is $r_\delta^{\mathbf{u}} = r_{\eta^{\mathbf{u}}}^{\mathbf{u}} = H_f^{\mathbf{u}} - \eta^{\mathbf{u}} H_1^{\mathbf{u}}$ and the performance is $\lambda_{\eta^{\mathbf{u}}}^{\mathbf{u}} = 0$. Let $g^{\mathbf{u}}$ be the performance potential vector [9] of the embedded Markov chain with policy $\mathbf{u}$ and cost function $r_{\eta^{\mathbf{u}}}^{\mathbf{u}}$. The potential indicates the contribution of a state to long-run average performance under a given policy. The definition of performance potential leads to the Poisson equation [9] for periodic transition probability matrix $P^{\mathbf{u}}$ and cost function $r_{\eta^{\mathbf{u}}}^{\mathbf{u}}$. The Poisson equation is

$$(I - P^{\mathbf{u}})g^{\mathbf{u}} = r_{\eta^{\mathbf{u}}}^{\mathbf{u}} - \lambda_{\eta^{\mathbf{u}}}^{\mathbf{u}}.$$

Since $\lambda_{\eta^{\mathbf{u}}}^{\mathbf{u}} = 0$ as mentioned before, Poisson equation reduces to

$$(I - P^{\mathbf{u}})g^{\mathbf{u}} = H_f^{\mathbf{u}} - \eta^{\mathbf{u}} H_1^{\mathbf{u}} \tag{15}$$

where I is the identity matrix, and $g^{\mathbf{u}}$ is the performance potential vector [9] of the embedded Markov chain with policy $\mathbf{u}$ and cost function $r_{\eta^{\mathbf{u}}}^{\mathbf{u}}$. It is well known that if $g^{\mathbf{u}}$ is a solution to (15), then $g^{\mathbf{u}} + ce$ (c is any real number) is also a solution to (15). Especially, one solution $g^{\mathbf{u}}$ takes the following form [9]

$$g^{\mathbf{u}} = (I - P^{\mathbf{u}} + e\pi^{\mathbf{u}})^{-1}\left(H_f^{\mathbf{u}} - \eta^{\mathbf{u}} H_1^{\mathbf{u}}\right). \tag{16}$$

The performance potentials $g^{\mathbf{u}}(d)$, $d = 1, \ldots, D$, can also be estimated from a sample path. We choose any event d^* as a reference event. For any event $d \in \mathcal{D}$, we define the stopping time $\tau_{d^*}(d) = \min\{i : i > 0, d_i = d^* | d_0 = d\}$ on a Markov chain $\{d_i, i = 0, 1, \cdots\}$ under policy $\mathbf{u}$. $\tau_{d^*}(d) < \infty$ for recurrent Markov chain $P^{\mathbf{u}}$. We have the following.

Theorem 3.2: For periodic Markov chains, the performance potential satisfies

$$g^{\mathbf{u}}(d) = \mathrm{E}\left\{\sum_{i=0}^{\tau_{d^*}(d)-1} r_\delta^{u_i}(d_i) | d_0 = d\right\} \tag{17}$$

where $u_i = \mathbf{u}(d_i)$ and $\delta = \eta^{\mathbf{u}}$. (See Appendix B for a proof.)

From (17), sample-path-based algorithms can be developed on the state process $x(t)$ with (8), (10), and (13). These algorithms are the same as those in the literature, except the cost function $H_f^{u_i}(d)$ is estimated by its sample path values $h_f^{u_i}(\zeta_i)$ [cf. (46)].

With Theorem 3.1, the standard techniques, such as policy iteration, reinforcement learning, etc., can be used to solve the MDP (6) with cost (13), which is equivalent to the MDP (6) with cost (12), and which is again equivalent to the original optimization problem (2) and (5). Next, we propose the following *policy iteration algorithm* for an optimal policy of the MDP (6) with cost (13) [which is also the optimal event-based control policy of the system (5) with performance (2)]. Let $\mathbf{u}_k$ be the policy used in the kth iteration, and $\mathbf{u}^*$ be the optimal policy.

Algorithm 3.1: Policy Iteration

```
1) Guess an initial policy $u = \mathbf{u}_0(d), \forall d \in \mathcal{D}$; set $k = 0$.
2) (Policy evaluation) Obtain $\bar{f}^{\mathbf{u}_k}$ by (12). Obtain $\eta^{\mathbf{u}_k}$ by
   $\eta^{\mathbf{u}_k} = \pi^{\mathbf{u}_k}\bar{f}^{\mathbf{u}_k}$. Obtain the potential $g^{\mathbf{u}_k}$ by (16) or (17).
3) (Policy improvement) Choose
   $$\mathbf{u}_{k+1}(d) \in \arg\min_{u \in U}\left\{\sum_{d' \in \mathcal{D}} p^u(d'|d)g^{\mathbf{u}_k}(d') + H_f^u(d) - \eta^{\mathbf{u}_k} H_1^u(d)\right\}, \text{ for all } d \in \mathcal{D}. \quad (18)$$
   If at an event $d$, action $\mathbf{u}_k(d)$ attains the minimum, then
   set $\mathbf{u}_{k+1}(d) = \mathbf{u}_k(d)$.
4) If $\mathbf{u}_{k+1} = \mathbf{u}_k$ componentwisely ($\mathbf{u}_{k+1}(d) = \mathbf{u}_k(d)$ for
   all $d \in \mathcal{D}$), then set $\mathbf{u}^* = \mathbf{u}_k$ and the algorithm stops;
   otherwise set $k := k + 1$ and go to step 2.
```

By Theorem 3.1.i and the policy iteration theory [9], [19], if the algorithm does not stop, at each iteration the performance of the embedded Markov chain with cost function (12) improves. When there are only a finite number of policies, the iteration procedure must stop. By Theorem 3.1.ii, when the iteration stops, it reaches the optimal performance of the embedded Markov chain with cost function (12).

Finally, note that there is a slight difference between the problem (6) and (13) and the formulation of a standard MDP problem: In (6) and (13), the cost function of action u at the kth iteration $r_{\eta^{\mathbf{u}_k}}^u(d) = H_f^u(d) - \eta^{\mathbf{u}_k} H_1^u(d)$ depends on $\eta^{\mathbf{u}_k}$, which changes every iteration, while in the standard MDP formulation, the cost function of any action u is fixed in all iterations. This difference does not change the policy iteration algorithm and the proof for its convergence.

Sample-path-based algorithms will be presented in Section V.

IV. Analytical Solutions

The three quantities used in policy iteration—$H_1^u(d)$, $H_f^u(d)$, and $p^u(d'|d)$—can be obtained analytically by solving differential equations or can be estimated from sample paths of the original system (5). In this section, we first discuss the analytical approach. We derive the differential equations that the quantities satisfy.

1) $H_1^u(d)$: First, suppose $t_0 = 0$ and $x(0) = x \in (x_{d-1}, x_{d+1})$, $d = 2, 3, \ldots, D-1$. u_0 is the action taken since $t_0 = 0$, until the end of this segment. Let [cf. (3)]

$$t_1 = \min\{t : t > 0, x(t) = x_{d-1} \text{ or } x_{d+1}\}.$$

This is the first passage time from any state $x \in (x_{d-1}, x_{d+1})$ to reach the set $\{x_{d-1}, x_{d+1}\}$. Let $q_1(x, u) = \mathrm{E}\{t_1 | x(0) = x, u_0 = u\}$ for all $x \in (x_{d-1}, x_{d+1})$, and $q_1(x, u) = 0$ for any other x.

First, from the backward Kolmogorov equation [13], we have

$$\mathcal{A}q_1(x,u) = \frac{\sigma^2}{2}\frac{\partial^2 q_1(x,u)}{\partial x^2} + \mu(x,u)\frac{\partial q_1(x,u)}{\partial x} \tag{19}$$

where $\mathcal{A}$ is the infinitesimal generator, which is defined by

$$\mathcal{A}q_1(x,u) = \lim_{t\to 0^+}\frac{\mathcal{P}_t q_1(x,u) - q_1(x,u)}{t} \tag{20}$$

where $\mathcal{P}_t$ is a transition operator

$$\begin{aligned}\mathcal{P}_t q_1(x,u) &= \mathrm{E}\{q_1(x(t),u)\,|x(0)=x,u_0=u\}\\ &= \int_{y\in\mathcal{R}} \mathbf{p}(x,t,\mathrm{d}y)q_1(y,u)\\ &= \int_{y\in(x_{d-1},x_{d+1})} \mathbf{p}(x,t,\mathrm{d}y)q_1(y,u)\end{aligned} \tag{21}$$

with $\mathbf{p}(x,t,\mathrm{d}y)$ being the transition function from state x to a Borel set $\mathrm{d}y$ in time t, determined by the system dynamics (5). We can easily verify that $\mathcal{A}q_1(x,u) = -1$ (cf. [13, p. 193]). Thus

$$\frac{\sigma^2}{2}\frac{\partial^2 q_1(x,u)}{\partial x^2} + \mu(x,u)\frac{\partial q_1(x,u)}{\partial x} = -1. \tag{22}$$

With boundary conditions $q_1(x_{d-1},u) = q_1(x_{d+1},u) = 0$, we can solve (22) to obtain $H_1^u(d) = q_1(x_d,u)$.

2) $H_f^u(d)$: Let [cf. (7) and (8)]

$$q_f(x,u) = \mathrm{E}\left\{\int_{t_0}^{t_1} f^{u_0}(x(t))\,\mathrm{d}t|x(0)=x,u_0=u\right\} \tag{23}$$

denote the expected cost on a segment starting from a state $x \in (x_{d-1},x_{d+1})$, $1<d<D$, under action u, and $q_f(x,u)=0$ for any $x \notin (x_{d-1},x_{d+1})$. Similar to (22), we have

$$\frac{\sigma^2}{2}\frac{\partial^2 q_f(x,u)}{\partial x^2} + \mu(x,u)\frac{\partial q_f(x,u)}{\partial x} = -f^u(x). \tag{24}$$

With boundary conditions $q_f(x_{d-1},u) = q_f(x_{d+1},u) = 0$, we can solve it to obtain $H_f^u(d) = q_f(x_d,u)$.

3) $p^u(d'|d)$: Let $q_p(x,u)$ denote the probability that from state $x \in (x_{d-1},x_{d+1})$, $1<d<D$, state x_{d+1} is reached before state x_{d-1} under action u. Similar to (22), we have

$$\frac{\sigma^2}{2}\frac{\partial^2 q_p(x,u)}{\partial x^2} + \mu(x,u)\frac{\partial q_p(x,u)}{\partial x} = 0. \tag{25}$$

With boundary conditions $q_p(x_{d-1},u) = 0$ and $q_p(x_{d+1},u)=1$, we can solve it to obtain $p^u(d+1|d) = q_p(x_d,u)$ and $p^u(d-1|d) = 1-q_p(x_d,u)$.

The solutions to (22), (24), and (25) are as follows (see, e.g., [13, p. 195]). Let

$$z(x,u) = e^{-\int^x [2\mu(s,u)/\sigma^2]\mathrm{d}s} \tag{26}$$

where $\int^x$ denotes an indefinite integral. We define a scale function

$$\begin{aligned}Z(x,u) &= \int^x z(y,u)\mathrm{d}y\\ &= \int^x \left\{e^{-\int^y [2\mu(s,u)/\sigma^2]\mathrm{d}s}\right\}\mathrm{d}y\end{aligned} \tag{27}$$

and the speed density $c(x,u) = 1/\sigma^2 z(x,u)$. Then, for $x_{d-1}\le x\le x_{d+1}$, $1<d<D$, the solutions to the three (22), (24), and (25) are

$$q_p(x,u) = \frac{Z(x,u)-Z(x_{d-1},u)}{Z(x_{d+1},u)-Z(x_{d-1},u)} \tag{28}$$

$$\begin{aligned}q_f(x,u) = 2\Bigg\{&q_p(x,u)\int_x^{x_{d+1}} [Z(x_{d+1},u)-Z(s,u)]c(s,u)f^u(s)\mathrm{d}s\\ &+[1-q_p(x,u)]\int_{x_{d-1}}^{x} [Z(s,u)-Z(x_{d-1},u)]\\ &\times c(s,u)f^u(s)\mathrm{d}s\Bigg\}\end{aligned} \tag{29}$$

and $q_1(x,u)$ can be obtained from the above equation by setting $f^u(s)=1$. For the cases with $d=1$ and $d=D$, the boundary conditions are different, but there are no conceptual difficulties. With these solutions, we can implement policy iteration Algorithm 3.1 to obtain an optimal policy of the system (5) with performance (2).

Generally, the integration in the scale function (27) may not have a closed form, so $q_1(x,u)$, $q_f(x,u)$, and $q_p(x,u)$, [and therefore, $H_1^u(d)$, $H_f^u(d)$, and $p^u(d'|d)$] may not be calculated analytically, but can be at least obtained numerically.

Example: We consider a special case with $\mu(x,u)=\mu(u)$ (it is called the state-independent case, which is often used in the formulation of manufacturing flow control problems; see, e.g., [1]) and a quadratic cost function $f^u(x) = mx^2 + u^{\mathrm{T}}Nu$, where m is a positive number and N is a positive definite matrix. We show that for this case, (28) and (29) have elementary functional forms.

After careful calculation, we have the following results. For all $x_{d-1}\le x\le x_{d+1}$, $1<d<D$, if $\mu(u)\ne 0$, we have

$$q_p(x,u) = \frac{e^{-2\mu(u)x/\sigma^2} - e^{-2\mu(u)x_{d-1}/\sigma^2}}{e^{-2\mu(u)x_{d+1}/\sigma^2} - e^{-2\mu(u)x_{d-1}/\sigma^2}} \tag{30}$$

$$\begin{aligned}q_1(x,u) = &\,q_p(x,u)\left[-\frac{\sigma^2}{2\mu^2(u)}\left(1-e^{-2\mu(u)(x_{d+1}-x)/\sigma^2}\right)\right.\\ &\left.+\frac{1}{\mu(u)}(x_{d+1}-x)\right] + (1-q_p(x,u))\\ &\times\left[-\frac{\sigma^2}{2\mu^2(u)}\left(1-e^{-2\mu(u)(x_{d-1}-x)/\sigma^2}\right)\right.\\ &\left.+\frac{1}{\mu(u)}(x_{d-1}-x)\right]\end{aligned} \tag{31}$$

and

$$\begin{aligned}q_f(x,u)=&u^{\mathrm{T}}Nuq_1(x,u)-\frac{mq_p(x,u)}{\mu(u)}\\&\times\Bigg[\frac{\sigma^2x_{d+1}^2}{2\mu(u)}-\frac{\sigma^4x_{d+1}}{2\mu^2(u)}+\frac{\sigma^6}{4\mu^3(u)}\\&-\left(\frac{\sigma^2x^2}{2\mu(u)}-\frac{\sigma^4x}{2\mu^2(u)}+\frac{\sigma^6}{4\mu^3(u)}\right)\\&\times e^{-2\mu(u)(x_{d+1}-x)/\sigma^2}-\frac{1}{3}\left(x_{d+1}^3-x^3\right)\Bigg]\\&-\frac{m\left(1-q_p(x,u)\right)}{\mu(u)}\\&\times\Bigg[\frac{\sigma^2x_{d-1}^2}{2\mu(u)}-\frac{\sigma^4x_{d-1}}{2\mu^2(u)}+\frac{\sigma^6}{4\mu^3(u)}\\&-\left(\frac{\sigma^2x^2}{2\mu(u)}-\frac{\sigma^4x}{2\mu^2(u)}+\frac{\sigma^6}{4\mu^3(u)}\right)\\&\times e^{-2\mu(u)(x_{d-1}-x)/\sigma^2}-\frac{1}{3}\left(x_{d-1}^3-x^3\right)\Bigg].\quad(32)\end{aligned}$$

If $\mu(u)=0$, then

$$q_p(x,u)=\frac{x-x_{d-1}}{x_{d+1}-x_{d-1}}\quad(33)$$
$$q_1(x,u)=-\frac{1}{\sigma^2}\left[x^2-(x_{d+1}+x_{d-1})x+x_{d+1}x_{d-1}\right]\quad(34)$$

and

$$\begin{aligned}q_f(x,u)=&-\frac{m}{6\sigma^2}\left[x^4-(x_{d+1}^2+x_{d-1}^2)(x_{d+1}+x_{d-1})x\right.\\&\left.+x_{d+1}^3x_{d-1}+x_{d+1}^2x_{d-1}^2+x_{d+1}x_{d-1}^3\right]\\&+u^{\mathrm{T}}Nuq_1(x,u).\quad(35)\end{aligned}$$

When $d=1$, since the system is stable and $x_1<0$ as assumed before, we have $\mu(u)>0$, and then we have $q_p(x_1,u)=1$, $q_1(x_1,u)=(x_2-x_1)/\mu(u)$, and

$$\begin{aligned}q_f(x_1,u)=-\frac{1}{\mu(u)}\Bigg\{&m\left[\frac{\sigma^2\left(x_2^2-x_1^2\right)}{2\mu(u)}-\frac{\sigma^4(x_2-x_1)}{2\mu^2(u)}\right]\\&-\frac{m}{3}\left(x_2^3-x_1^3\right)-u^{\mathrm{T}}Nu(x_2-x_1)\Bigg\}.\quad(36)\end{aligned}$$

When $d=D$, $\mu(u)<0$. We have $q_p(x_D,u)=0$, $q_1(x_D,u)=(x_{D-1}-x_D)/\mu(u)$ and

$$\begin{aligned}q_f(x_D,u)=-\frac{1}{\mu(u)}\Bigg\{&m\left[\frac{\sigma^2\left(x_{D-1}^2-x_D^2\right)}{2\mu(u)}-\frac{\sigma^4(x_{D-1}-x_D)}{2\mu^2(u)}\right]\\&-\frac{m}{3}\left(x_{D-1}^3-x_D^3\right)-u^{\mathrm{T}}Nu(x_{D-1}-x_D)\Bigg\}.\quad(37)\end{aligned}$$

In this special case, we apply (30)–(37) instead of (28) and (29) to calculate the quantities $H_1^u(d)$, $H_f^u(d)$, and $p^u(d'|d)$ for all $d,d'\in\mathcal{D}$, $u\in U$. If the size of event set is not very large, we may calculate performance potential $g^{\mathbf{u}}$ by using (16), without simulation and observation. Therefore, for the state-independent case, we can find an optimal policy for the optimal control problem by policy iteration analytically. □

V. Sample-Path-Based Algorithms

When a closed-form solution cannot be obtained, we may develop sample-path-based algorithms in which the quantities needed in policy iteration are estimated by observing/analyzing the system's behavior over a sample path.

We apply the Q-learning approach to our problem. Given a policy $\mathbf{u}$, define a Q-factor [2] for every event–action pair (d,u) as (note that with the cost function $r_{\eta^{\mathbf{u}}}^u(d)=H_f^u(d)-\eta^{\mathbf{u}}H_1^u(d)$, the corresponding performance is $\lambda_{\eta^{\mathbf{u}}}^{\mathbf{u}}=0$)

$$Q(d,u)=\sum_{d'=1}^{D}p^u(d'|d)g^{\mathbf{u}}(d')+r_{\eta^{\mathbf{u}}}^u(d)\quad(38)$$

where $g^{\mathbf{u}}$ and $\eta^{\mathbf{u}}$ are the performance potential and the long-run average performance under the given policy $\mathbf{u}$. From the policy improvement (18), we choose

$$\hat{\mathbf{u}}(d)\in\arg\left\{\min_{u'\in U}Q(d,u')\right\}\qquad\forall d\in\mathcal{D}\quad(39)$$

as the action taken at event d in the next iteration. Therefore, if we can estimate the Q-factors, we can update the policy using (39).

However, it is impossible to estimate $Q(d,u)$ on a sample path if the pair (d,u) does not appear on the path at all. Therefore, if a sample path is under a deterministic policy that maps an event to one action, Q-learning approach is not useful because the sample path only contains one event–action pair for each event. Thus, this approach applies only to random policies. The standard way to implement the Q-learning approach is to use the ϵ-greedy policy [24]. At event d, denote $\hat{\mathbf{u}}(d)$ as an action satisfying (39), which is called a greedy action, and $\hat{\mathbf{u}}$ is called a greedy policy. The greedy action (or policy) may not be unique. Given a small real number $0<\epsilon<1$. From any greedy policy $\hat{\mathbf{u}}$, we can construct an ϵ-greedy policy $\hat{\mathbf{u}}_\epsilon$ as follows: At event d, with probability $1-\epsilon$, we choose a greedy action $\hat{\mathbf{u}}(d)$, and with probability ϵ, we choose any other action randomly, usually with an equal probability $\epsilon/(|U|-1)$ ($|U|$ is the number of actions), so that all the actions may be chosen for any event d. When ϵ is small, an ϵ-greedy policy is close to the greedy policy determined by (39), yet it explores all the possible pairs of (d,u).

Suppose that we are given a sample path under an ϵ-greedy policy $\hat{\mathbf{u}}_\epsilon$. To get $Q(d,u)$, we need to estimate $r_\delta^u(d)=H_f^u(d)-\delta H_1^u(d)$ with $\delta=\eta^{\hat{\mathbf{u}}_\epsilon}$. At the lth event, for all $d\in\mathcal{D}$, $u\in U$, we have the estimates shown in (40)–(43), shown at the bottom of the next page, where $1_{d,u}(d_i,u_i)=1$ if $d_i=d$ and $u_i=u$; $1_{d,u}(d_i,u_i)=0$ otherwise. From the strong law of large numbers, we have $\eta_l^{\hat{\mathbf{u}}_\epsilon}$ and $r_{\delta,l}^u(d)$ converge to $\eta^{\hat{\mathbf{u}}_\epsilon}$ and $r_\delta^u(d)$ with probability 1 as l goes to infinity, respectively.

Recall the stopping time $\tau_{d^*}(d)=\min\{i:i>0,d_i=d^*|d_0=d\}$ under policy $\hat{\mathbf{u}}_\epsilon$. From (17) and (38), we can easily derive an equation for the Q-factors under the ϵ-greedy policy $\hat{\mathbf{u}}_\epsilon$

$$Q(d,u)=\mathrm{E}\left\{\sum_{i=0}^{\tau_{d^*}(d)-1}r_\delta^{u_i}(d_i)|d_0=d,u_0=u\right\}\quad(44)$$

with $\delta = \eta^{\hat{\mathbf{u}}_\epsilon}$. When ϵ is small, this is close to the Q-factor of the greedy policy $\hat{\mathbf{u}}$.

Next, we develop an algorithm that estimates $Q(d,u)$ according to (44) by observing and analyzing a sample path of the system under $\hat{\mathbf{u}}_\epsilon$. Consider such a sample path with $d_0 = d^*$. Define regenerative points: $\beta_0 = 0$, and $\beta_{j+1} = \min\{i : i > \beta_j, d_i = d^*\}$, $j = 0,1,\ldots$. We call the period between the two regenerative points β_j and β_{j+1}, $\{d_i : \beta_j \le i < \beta_{j+1}\}$, the jth regenerative period. Define $\gamma_j(d,u)$ as follows:

$$\gamma_j(d,u) = \begin{cases} \min\{i : \beta_j \le i < \beta_{j+1}, \\ \quad d_i = d, u_i = u\}, & \text{if the above set is not } \emptyset \\ \beta_{j+1} - 1, & \text{otherwise} \end{cases} \tag{45}$$

which denotes the first occurrence time of pair (d,u) in the jth regenerative period. Also, let $\chi_j(d,u) = 1$ if $\{i : \beta_j \le i < \beta_{j+1}, d_i = d, u_i = u\} \ne \emptyset$; $\chi_j(d,u) = 0$ otherwise. By (44), we have the following estimate of $Q(d,u)$ at the lth event, d_l:

$$Q_l(d,u) = \begin{cases} \frac{1}{\sum_{j=0}^{Y_l-1}\chi_j(d,u)} \sum_{j=0}^{Y_l-1} \chi_j(d,u) \\ \quad \times \sum_{i=\gamma_j(d,u)}^{\beta_{j+1}-1} r_{\delta,i}^{u_i}(d_i), & \text{if } \sum_{j=0}^{Y_l-1}\chi_j(d,u) \ne 0 \\ 0, & \text{otherwise} \end{cases} \tag{46}$$

where Y_l is the number of regenerative periods obtained up to d_l, and $r_{\delta,i}^{u_i}(d_i)$ is given by (43). We have the following result on the convergence of this estimate. (See Appendix C for a proof.)

Theorem 5.1: For every $d \in \mathcal{D}$ and $u \in U$

$$\lim_{l\to\infty} Q_l(d,u) = Q(d,u), \qquad \text{w.p.1.} \tag{47}$$

Applying (46) to estimate $Q(d,u)$ directly may require lots of memory: When calculating $Q_l(d,u)$ at d_l, all the costs along the sample path, $r_{\delta,i}^{u_i}(d_i)$, $i = 0,1,\ldots,l$, are used. Then, we have to store all the quantities along the whole sample path. This storage resource requirement is very large and increasing as the length of sample path increases, so it limits application of the algorithm. We propose a recursive estimation that is equivalent to (46), but does not require the large storage. Define $\varphi_l(d,u) = \sum_{i=0}^{l-1} 1_{d,u}(d_i,u_i)$ be the number of visits to pair (d,u) up to d_l, then we have the following estimates:

$$\varphi_l(d,u) = \varphi_{l-1}(d,u) + 1_{d,u}(d_{l-1},u_{l-1}) \tag{48}$$

$$H_{f,l}^u(d) = \frac{\varphi_{l-1}(d,u)}{\varphi_l(d,u)} H_{f,l-1}^u(d) + \frac{1_{d,u}(d_{l-1},u_{l-1})}{\varphi_l(d,u)} h_f^{u_{l-1}}(\zeta_{l-1}) \tag{49}$$

$$H_{1,l}^u(d) = \frac{\varphi_{l-1}(d,u)}{\varphi_l(d,u)} H_{1,l-1}^u(d) + \frac{1_{d,u}(d_{l-1},u_{l-1})}{\varphi_l(d,u)} h_1^{u_{l-1}}(\zeta_{l-1}) \tag{50}$$

$$\eta_l^{\hat{\mathbf{u}}_\epsilon} = \frac{t_{l-1}}{t_l}\eta_{l-1}^{\hat{\mathbf{u}}_\epsilon} + \frac{1}{t_l} h_f^{u_{l-1}}(\zeta_{l-1}) \tag{51}$$

with initial values $\varphi_0(d,u) = 0$, $H_{f,0}^u(d) = 0$, $H_{1,0}^u(d) = 0$ for all $d \in \mathcal{D}$, $u \in U$, and $\eta_0^{\hat{\mathbf{u}}_\epsilon} = 0$. The cost function $r_{\delta,l}^u(d)$ is calculated by (43). We verify that the above values are the same as (40)–(42), and these estimates are updated at every event.

As shown in (46), Q-factors are estimated by regenerative periods. We propose a recursive algorithm in which Q-factors are updated at every regenerative point. Recall Y_l is the number of regenerative periods up to d_l, then β_{Y_l} is the last regenerative point before d_l. Define $\psi_{Y_l}(d,u) = \sum_{j=0}^{Y_l-1}\chi_j(d,u)$ to be the number of regenerative periods, in which event–action pair (d,u) happens, among these Y_l regenerative periods. Then, we have the following estimate of $Q(d,u)$ at regenerative point β_{Y_l}:

$$\psi_{Y_l}(d,u) = \psi_{Y_l-1}(d,u) + \chi_{Y_l-1}(d,u) \tag{52}$$

$$Q_{Y_l}(d,u) = \frac{\psi_{Y_l-1}(d,u)}{\psi_{Y_l}(d,u)} Q_{Y_l-1}(d,u) + \frac{\chi_{Y_l-1}(d,u)}{\psi_{Y_l}(d,u)} \sum_{i=\gamma_{Y_l-1}(d,u)}^{\beta_{Y_l}-1} r_{\delta,i}^{u_i}(d_i) \tag{53}$$

with initial values $\psi_0(d,u) = 0$, $Q_0(d,u) = 0$ for all $d \in \mathcal{D}$, $u \in U$. Apparently, $Q_{Y_l}(d,u) = Q_l(d,u)$ for all $l = 0,1,\ldots$. However, with (48)–(53), we need store only $r_{\delta,i}^{u_i}(d_i)$, $\beta_{Y_l-1} \le i < \beta_{Y_l}$. This requirement of storage is much smaller than that

$$H_{f,l}^u(d) = \begin{cases} \frac{1}{\sum_{i=0}^{l-1} 1_{d,u}(d_i,u_i)} \sum_{i=0}^{l-1} 1_{d,u}(d_i,u_i) h_f^{u_i}(\zeta_i), & \text{if } \sum_{i=0}^{l-1} 1_{d,u}(d_i,u_i) \ne 0 \\ 0, & \text{otherwise} \end{cases} \tag{40}$$

$$H_{1,l}^u(d) = \begin{cases} \frac{1}{\sum_{i=0}^{l-1} 1_{d,u}(d_i,u_i)} \sum_{i=0}^{l-1} 1_{d,u}(d_i,u_i) h_1^{u_i}(\zeta_i), & \text{if } \sum_{i=0}^{l-1} 1_{d,u}(d_i,u_i) \ne 0 \\ 0, & \text{otherwise} \end{cases} \tag{41}$$

$$\eta_l^{\hat{\mathbf{u}}_\epsilon} = \frac{1}{\sum_{i=0}^{l-1} h_1^{u_i}(\zeta_i)} \sum_{i=0}^{l-1} h_f^{u_i}(\zeta_i) \tag{42}$$

$$r_{\delta,l}^u(d) = H_{f,l}^u(d) - \eta_l^{\hat{\mathbf{u}}_\epsilon} H_{1,l}^u(d) \tag{43}$$

of (46). Furthermore, recursive algorithms are more effective for implementation in practice.

From the above estimates, Algorithm 3.1 becomes the following.

Algorithm 5.1

```
1) (Initialize) Guess an initial (deterministic) policy $\hat{\mathbf{u}}_0$.
   Choose a reference event $d^*$. Set a large integer $L$ as the
   simulation length, a small number $0 < \epsilon < 1$, and $k := 0$.
2) (Policy evaluation) Set $t_0 = 0$, $d_0 = d^*$ (i.e.,
   $x(0) = x_{d_0} = x_{d^*}$). At each event $d_i$, $i = 0, 1, \ldots$,
   choose action $u_i = \hat{\mathbf{u}}_{k,\epsilon}(d_i)$ according to the $\epsilon$-greedy
   policy $\hat{\mathbf{u}}_{k,\epsilon}$ (constructed by the greedy policy $\hat{\mathbf{u}}_k$). Run
   the system up to $d_L$, estimate $H^u_{f,L}(d)$, $H^u_{1,L}(d)$, $\eta_L^{\hat{\mathbf{u}}_{k,\epsilon}}$,
   $r^u_{\delta,L}(d)$ and $Q_{Y_L}(d,u), \forall d \in \mathcal{D}$, $u \in U$ by (43) and
   (48)–(53) recursively.
3) (Policy improvement) Choose
   $$\hat{\mathbf{u}}_{k+1}(d) \in \arg\left\{\min_{u' \in U} Q_{Y_L}(d,u')\right\} \qquad \forall d \in \mathcal{D}. \qquad (54)$$
   If at an event $d$, action $\hat{\mathbf{u}}_k(d)$ attains the minimum, then
   set $\hat{\mathbf{u}}_{k+1}(d) = \hat{\mathbf{u}}_k(d)$.
4) If $\hat{\mathbf{u}}_{k+1} = \hat{\mathbf{u}}_k$ component-wisely, then the algorithm
   stops; otherwise set $k := k + 1$ and go to step 2.
```

The quantities $Q_{Y_L}(d,u)$ estimated in the policy evaluation step are for the ϵ-greedy policies. There are two types of errors in the algorithm. One is the stochastic error due to the finiteness of L, and the other is due to the difference between the ϵ-greedy policy and greedy policy. We first consider the second type of error. We expect that when ϵ is small, these errors are also small. As shown in [9, Ch. 5], when the errors are small enough, and policy space is finite as assumed before, the improved greedy policies determined in the policy improvement step are exactly the same as they would have been had the correct quantity $Q(d,u)$ been used. Therefore, if ϵ is small enough and there is no stochastic error, Algorithm 5.1 stops at the optimal policy $\mathbf{u}^*$ in a finite number of iterations. Finally, we can choose L large enough so that the probability that this algorithm does not converge can be less than any given small number. For a formal discussion, see [9, Ch. 5].

Many reinforcement learning algorithms can be developed with the above formulation. For example, we can combine the policy evaluation step and the policy improvement step together to improve the efficiency of the algorithm [9]. An alternative is to use the so-called SARSA algorithm [24], which explores the temporal difference (TD) defined as

$$w_i = Q(d_{i+1}, u_{i+1}) + h^{u_i}_f(\zeta_i) - \eta^{\hat{\mathbf{u}}_\epsilon} h^{u_i}_1(\zeta_i) - Q(d_i, u_i). \quad (55)$$

In the following algorithm, we choose a sequence ϵ_i, $i = 0, 1, \ldots$, diminishing to zero as i increases. Thus, the ϵ-greedy policies used in the algorithm converge to the greedy policy.

Algorithm 5.2: SARSA

```
1) (Initialize) Guess a set of initial $Q$-factors $Q(d,u)$
   for all $d \in \mathcal{D}$ and $u \in U$ (or guess an initial policy
   and run a sample path of the system to estimate its
   $Q$-factors). Set $\eta = 0$, $i = 0$, and $t_0 = 0$. Choose an
   initial event $d_0$ (i.e. $x(0) = x_{d_0}$), an initial action by
   $u_0 \in \arg\{\min_{u' \in U} Q(d_0, u')\}$, a large integer $K$ as the
   simulation length, a diminishing sequence $0 < \epsilon_i < 1$,
   and a sequence of step sizes $\alpha_i$ satisfying
   $$\alpha_i > 0 \quad \sum_{i=0}^{\infty} \alpha_i = \infty \quad \sum_{i=0}^{\infty} \alpha_i^2 < \infty. \qquad (56)$$
2) (Simulation) Run the system under action $u_i$, until
   time $t_{i+1}$ when the next event $d_{i+1}$ occurs. Record
   $h^{u_i}_f(\zeta_i)$ and $h^{u_i}_1(\zeta_i)$ of this segment.
3) (Determine actions) Determine a greedy action at
   event $d_{i+1}$ according to
   $$\hat{\mathbf{u}}(d_{i+1}) \in \arg\left\{\min_{u' \in U} Q(d_{i+1}, u')\right\}.$$
   Choose action $u_{i+1} = \hat{\mathbf{u}}(d_{i+1})$ with probability $1 - \epsilon_{i+1}$,
   and choose any other action $u_{i+1} = u' \neq \hat{\mathbf{u}}(d_{i+1})$,
   $u' \in U$, with probability $\epsilon_{i+1}/(|U| - 1)$.
4) (Update) Update $\eta$ by
   $$\eta = \frac{t_i}{t_{i+1}}\eta + \frac{1}{t_{i+1}} h^{u_i}_f(\zeta_i) \qquad (57)$$
   and $Q$-factors by
   $$Q(d_i, u_i) = Q(d_i, u_i) + \alpha_i w_i \qquad (58)$$
   where $w_i$ is the temporal difference in (55).
5) If $i = K$, algorithm terminates. Otherwise, set $i := i + 1$
   and go to step 2.
```

In the algorithm, whenever an event occurs, we update one of the Q-factors and determine the next action according to the updated Q-factors. The idea is the change of the Q-factors at each event is usually very small, and this small change in Q-factors may not change the greedy actions, thus the system in fact runs under the same ϵ-greedy policy for many segments, and therefore the iterative process is hopefully stable. However, the convergence of the SARSA algorithm for the average-performance criterion problem is not guaranteed. With properly chosen ϵ_i and step-size α_i, Algorithm 5.2 may converge to an optimal policy.

VI. Periodic-Sampling-Based Control

In order to compare the performance of the Lebesgue sampling approach to that of the periodic sampling approach, we briefly review some results of the periodic-sampling-based optimal control problem. Details can be found in [5]. We consider only the linear quadratic case. Denote the sampling time as t_i, $i = 0, 1, 2, \ldots$, with an equal sampling interval $\Delta = t_{i+1} - t_i$ for all i. The system dynamic is

$$\mathrm{d}x = (ax + bu_i)\mathrm{d}t + \sigma \mathrm{d}v, \qquad t_i \le t < t_{i+1} \qquad (59)$$

where u_i is the control adopted at time t_i, which remains unchanged in $[t_i, t_{i+1})$, and $a \in \mathcal{R}$ and $b \in \mathcal{R}^{1\times n}$ are parameters.

The cost function is $f^u(x) = mx^2 + u^TNu$. Let $x_i = x(t_i)$. Our goal is to find a feedback control law $\mathbf{u}(x)$, $x \in \mathcal{R}$, to minimize the performance (2).

From (59), if $a \neq 0$, we have

$$x_{i+1} = Ax_i + Bu_i + \xi \tag{60}$$

where $A = e^{a\Delta}$, $B = -(b/a)(1 - e^{a\Delta})$, and $\xi = \sigma\int_0^\Delta e^{a(\Delta-s)}\mathrm{d}v$ is an i.i.d. random variable with mean zero and variance $Var(\xi) = (\sigma^2/2a)(e^{2a\Delta} - 1)$.

Let $F^u(x) = \mathrm{E}\{(1/\Delta)\int_0^\Delta f^u(x(s))\mathrm{d}s|x_0 = x\}$. This is the expected cost on a sampling period starting from x. After some calculation, we have

$$F^u(x) = Gx^2 + xRu + u^\mathrm{T}Vu + J \tag{61}$$

where $G = (m/2a\Delta)(e^{2a\Delta} - 1)$, $R = (mb/a^2\Delta)(e^{2a\Delta} - 1) - (2mb/a^2\Delta)(e^{a\Delta} - 1)$, $V = (mb^\mathrm{T}b/2a^3\Delta)(e^{2a\Delta} - 1) - (2mb^\mathrm{T}b/a^3\Delta)(e^{a\Delta} - 1) + (mb^\mathrm{T}b/a^2) + N$, and $J = (m\sigma^2/4a^2\Delta)(e^{2a\Delta} - 1) - (m\sigma^2/2a)$. The optimal control law is $\mathbf{u}(x) = -Lx$, where $L = (1/2)(B^\mathrm{T}BS + V)^{-1}(2AB^\mathrm{T}S + R^\mathrm{T})$, and S satisfies the algebraic Riccati equation

$$S = G + A^2S - \frac{1}{4}(2ABS + R)\left(B^\mathrm{T}BS + V\right)^{-1}(2AB^\mathrm{T}S + R^\mathrm{T}). \tag{62}$$

Solving (62), we get S and L. The corresponding optimal performance is $\eta = Var(\xi)S + J = (\sigma^2S/2a)(e^{2a\Delta} - 1) + J$.

If $a = 0$ (a special state-independent case considered in Section IV), we have $A = 1$, $B = b\Delta$, and $\xi = \sigma v$ is a random variable with a normal distribution with zero mean and variance $Var(\xi) = \sigma^2\Delta$. $G = m$, $R = mb\Delta$, $V = (1/3)mb^\mathrm{T}b\Delta^2 + N$, and $J = (1/2)m\sigma^2\Delta$. We obtain the optimal control policy by solving Riccati equation (62). The corresponding optimal performance is $\eta = \sigma^2\Delta S + J$.

If the system dynamic is not linear, or the cost function is not quadratic, or the control set $U \neq \mathcal{R}^n$, the optimal control problems in general do not have an analytical solution. However, it can be solved with approximate approaches, e.g., by approximate dynamic programming [22], [26] or policy iteration with discretized state space [27].

VII. Numerical Examples and Comparison

Now, we give a few numerical examples to show the results of our approach and to compare them to those with equal-length sampling. In Examples 7.1–7.3, we use the analytical approach of policy iteration derived in Sections III and IV, and in Example 7.4, we apply the sample-path-based approach SARSA introduced in Section V.

Example 7.1: Consider a linear quadratic optimal control problem with $\mathrm{d}x = u\mathrm{d}t + \sqrt{2}\mathrm{d}v$, $f^u(x) = x^2$ and control set $U = \{-5, -4, \cdots, 4, 5\}$. In this example, the variance of the system state is minimized. The event set is $\mathcal{D} = \{1, 2, \ldots, 7\}$, and we have $\mathcal{X}_\mathcal{D} = \{-3, -2, \cdots, 2, 3\}$.

We first calculate $H_1^u(d)$, $H_f^u(d)$, and $p^u(d'|d)$, for all $d, d' \in \mathcal{D}$ and $u \in U$, from (30)–(37), and then calculate the long-run average performance $\eta^\mathbf{u}$ and the performance potentials $g^\mathbf{u}$ to implement policy iteration Algorithm 3.1. A policy $\mathbf{u}$ can be written as a column vector $\mathbf{u} = [\mathbf{u}(1), \ldots, \mathbf{u}(D)]^\mathrm{T}$. The initial policy is chosen as the hollow circles in Fig. 1: $\mathbf{u}_0 = [1\,1\,1\,0\,-1\,-1\,-1]^\mathrm{T}$. After one iteration, the system performance reduces from 2.2778 to 0.2943, and the optimal policy is obtained, which is shown as the solid circle in Fig. 1: $\mathbf{u}^* = [5\,5\,5\,0\,-5\,-5\,-5]^\mathrm{T}$. Under the optimal policy $\mathbf{u}^*$, the average length of a sampling interval is $\bar{t}^{\mathbf{u}^*} = 0.3476$ s. As shown in [4], if the impulsive control (which requires an infinitely large control power) is applied, the performance is $1/6 = 0.1667$. In this problem, the controls are finite ($|u| \leq 5$), and the system state cannot reach 0 immediately when control is added, as the impulsive control does. Therefore, the performance is not as good as that in [4], with the cost function $f^u(x) = x^2$, which implies free energy cost even for an infinite power. If the control variable can be very large, then the optimal performance can be close to 0.1667.

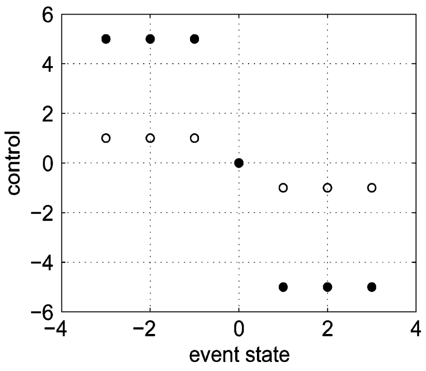


Fig. 1. Lebesgue-sampling-based control policy.

With the periodic sampling approach, to make a fair comparison, we choose the sampling interval be the same as that of the Lebesgue-sampling-based approach, i.e., $\Delta = \bar{t}^{\mathbf{u}^*} = 0.3476$ s. From Section VI, if we allow to use the continuous and unbounded control set $U = \mathcal{R}$, the solution to the Riccati equation is $S = 1/(2\sqrt{3})$, the theoretical optimal control is $u = -3.647x$, and the optimal performance is $\eta = 1.5773\Delta = 0.5483$. In this example, however, the control set is discretized to be a finite set $U = \{-5, \cdots, 4, 5\}$. Thus, we do not have analytical solutions and need to apply the sample-path-based approach [27] to get an optimal policy. We have a discrete-time control problem (60), (61). The continuous state space is discretized into a finite number of states $\{\kappa : \kappa = -M, \cdots, 0, \cdots, M - 1, M\}$, and each κ corresponds to an interval $(\kappa\iota - (\iota/2), \kappa\iota + (\iota/2)]$, with $\iota = 0.01$ and $M = 300$. When the continuous state x_i falls into the interval $(\kappa\iota - (\iota/2), \kappa\iota + (\iota/2)]$, we say that a discrete state κ is reached. ι is small enough so that the values of states in the same interval are very close, and M is large enough so that the probability of visiting regions $(-\infty, -M\iota - (\iota/2)]$ and $(M\iota + (\iota/2), \infty)$ is close to zero. Thus, the error caused by discretization can be negligible. We approximate the original continuous-state system by a discrete-state system and then implement policy-iteration-based numerical algorithms. For a discrete state κ, we accumulate the costs of the next 100 steps to estimate performance potential $g(\kappa)$. At each iteration, we run the system 2×10^4 steps and then update the control policy. In Fig. 2, the dashed line is the initial control law

$$\mathbf{u}_0(x) = \begin{cases} 5, & x \in \left(-\infty, -\frac{9}{4}\right) \\ -\lfloor 2x + 0.5 \rfloor, & x \in \left[-\frac{9}{4}, \frac{9}{4}\right) \\ -5, & x \in \left[\frac{9}{4}, +\infty\right) \end{cases} \tag{63}$$

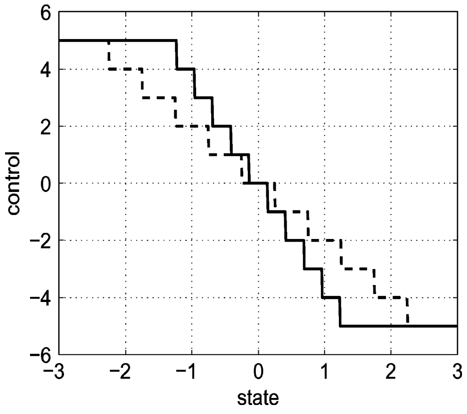


Fig. 2. Periodic-sampling-based control policy.

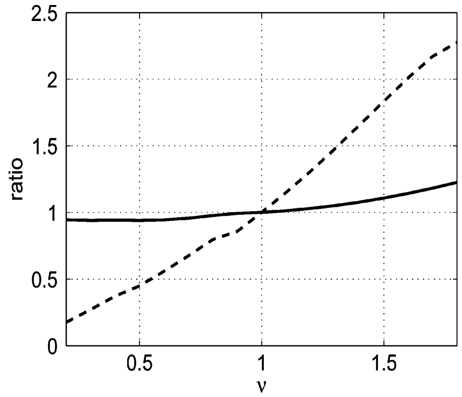


Fig. 3. Improvement ratio.

where $\lfloor x \rfloor$ denotes the largest integer that is not larger than x. After four iterations, an optimal policy is obtained, which is shown as the solid line in Fig. 2. The optimal performance is 0.5825, which is slightly larger than the theoretical value for the continuous-state problem. Overall, with the same average length of a sampling interval, the optimal performance of the approach with Lebesgue sampling is about 49.5% better than the approach with periodic sampling. For periodic sampling method, performance η is linear to the sampling interval ($\eta = 1.5773\Delta$ as mentioned before), then in order to achieve the same performance as Lebesgue sampling, periodic sampling technique has to sample the system state twice faster than Lebesgue sampling.

It is worth noting that the optimal policy obtained by Lebesgue sampling is of the min–max type, which drives the system to the origin as quickly as possible. It is natural because the cost for control energy is zero in this example. The larger the control variable is, the faster the system state reaches zero, and the better performance the system obtains. However, the control law by periodic sampling is not of the type. It is more conservative to avoid overshooting because once overshooting happens (say the system state becomes very large), nothing can be done before the next sampling instant, and therefore it may result in a large state variance. With Lebesgue sampling, whenever the system reaches a high level, e.g., $x(d) = 1$ or -1, the system will detect it, and an appropriate control can be (timely) applied to correct the error. This explains why the Lebesgue sampling approach may be better.

Example 7.2: In this example, we consider the same system as in Example 7.1, except the cost function changes to $f^u(x) = x^2 + Nu^2$, for $N = 0, 0.1, 0.2, \ldots, 1$.

In this example, we need to balance the system variance and the control energy. Using the same approaches and the same initial policies as in Example 7.1, we obtain the optimal control policies and their performances for all possible N. With Lebesgue sampling, optimal policies are always obtained in three iterations. When $N > 0$, the optimal policy may not be of the min–max type any more. For example, when $N = 0.3$, the optimal policy is $\mathbf{u}^* = [5, 3, 2, 0, -2, -3, -5]^{\mathrm{T}}$. The optimal performances and the corresponding average lengths of a sampling interval are shown in Table I. With the same sampling intervals, the optimal performances with periodic sampling are also obtained, including both theoretical and simulation-based (for the case with finite control set) results. The percentages of improvement of Lebesgue sampling compared to periodic sampling are listed in the last column in Table I. As N increases, the percentage of improvement gets smaller because the cost on control energy becomes more important.

TABLE I
RESULTS OF EXAMPLE 7.2

N	η (Leb. Samp.)	$\bar{t}^{\mathbf{u}^*} = \Delta$	η (Per. Samp.) Theor.	Simu.	Improvement Percentage
0	0.2943	0.3476	0.5483	0.5825	49.5%
0.1	0.7991	0.3936	1.0656	1.0959	27.1%
0.2	1.0547	0.4245	1.3519	1.4057	25.0%
0.3	1.2834	0.4275	1.5504	1.6258	21.1%
0.4	1.4695	0.4520	1.7435	1.8369	20.0%
0.5	1.6095	0.4520	1.8901	1.9999	19.5 %
0.6	1.7337	0.4598	2.0316	2.1514	19.4%
0.7	1.8447	0.4612	2.1556	2.2978	19.7%
0.8	1.9531	0.4612	2.2698	2.4268	19.5%
0.9	2.0615	0.4612	2.3772	2.5479	19.1%
1	2.1699	0.4612	2.4788	2.6642	18.6%

Example 7.3: Consider the same system as Example 7.2, with a fixed $N = 0.3$, except that the event set changes to $\mathcal{X}_{\mathcal{D}} = \{-3, -1-\nu, -\nu, 0, \nu, 1+\nu, 3\}$, with $\nu \in [0.2, 1.8]$.

In this example, the intervals between two event values are not constant. The case with $\nu = 1$ is the same as that in Example 7.2. With Lebesgue sampling, we optimize the system for all possible values of ν. Results are shown in Fig. 3. The solid line represents the ratio of the optimal performance with all possible ν versus the optimal performance with $\nu = 1$; the dashed line represents the ratio of the mean lengths of sampling intervals versus the mean length with $\nu = 1$. From the figure, it is obvious that the slope of the solid line is much smaller than that of the dashed line. When ν increases slightly, performance becomes a little worse, and the mean length of a sampling interval increases significantly. It means lots of computational resources are saved, with a little cost on the system performance. This is not surprising, as with the "uneven" event set $\mathcal{X}_{\mathcal{D}}$, more emphasis is put to more important events, i.e., the events apart from the origin.

A question naturally arises: How do we determine the event set so that the optimal performance is the best? This problem remains unsolved.

Example 7.4: In this example, we consider the system $\mathrm{d}x = (-0.1x + u)\mathrm{d}t + \sqrt{2}\mathrm{d}v$ with $f^u(x) = x^2 + 0.3u^2$, and the other parameters are the same as those in Example 7.1.

In this case, a closed-form solution for Lebesgue-sampling-based approach is not available. Thus, we apply the SARSA Algorithm 5.2 to obtain an optimal policy. In simulation, the time scale is set to be 0.001 s. Choose $\epsilon_i = 1/(i/5000 + 2)$. At each event, the ϵ-greedy policy

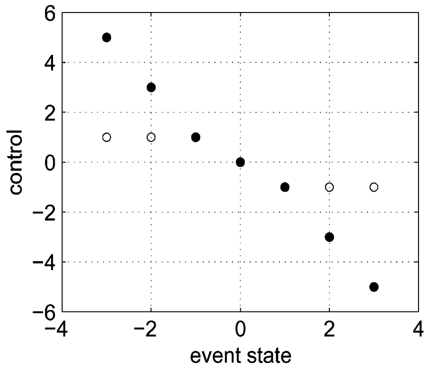


Fig. 4. Control policy by Algorithm 5.2.

picks up randomly any possible actions. Set the simulation length $K = 2\,000\,000$. The initial policy is the same as in Example 7.1, and the initial Q-factors are chosen to be all zero. Set the step-size $\alpha_i = 1/\varphi_{i+1}(d_i, u_i)$, where $\varphi_{i+1}(d_i, u_i)$ defined in (48) is the number of visits to event–action pair (d_i, u_i) up to event d_{i+1}. Let $d_0 = 4$, and initial action $u_0 = 0$. The "optimal" policy obtained by Algorithm 5.2 is shown as the solid circle in Fig. 4. The performance of this policy is 1.2116, and the average length of a sampling interval is 0.4789 s. With periodic sampling and using the same sampling interval, the theoretical optimal control is $u = -1.1744x$ and the optimal performance is 1.4823. With the same finite control set and the same discretized state space as in Example 7.1, the simulation-based algorithm gives the optimal performance 1.5556. Lebesgue sampling is 20% better than periodic sampling.

VIII. Conclusion

In this paper, we formulate the optimal control problem with Lebesgue sampling and show that it can be solved by solving an equivalent MDP problem. Both policy-iteration and reinforcement-learning algorithms are developed. Policy iteration can be implemented analytically, numerically, or based on sample paths.

Compared to the periodic sampling approach, with the same average length of a sampling interval, Lebesgue-sampling-based policy may have a considerably better performance. How to choose the event set to achieve the best performance and under what conditions Lebesgue sampling performs better remain open problems.

The paper considers only the one-dimensional case of state space. The same idea applies for the multiple-dimensional case, and some technical issues arise because we can only aggregate one component at a time. This extension is promising and will be done in our future work.

Lastly, the sensors used in Lebesgue sampling may work slightly differently from those for periodic sampling. For example, a pressure sensor may turn the air pressure into voltage, and a temperature sensor may turn the temperature to the length of mercury. In periodic sampling, the voltage or length is turned into digital signals periodically. In Lebesgue sampling, a signal is triggered when the voltage or length reaches a set of specific values. Because the values, denoted as $e_1, e_2 \cdots$, are predetermined, the sensor may only need to send out the identification numbers of these values, e.g., 1, 2, $\cdots$, and the exact values are not needed.

Appendix A
Proof of Theorem 2.1

When $\mu(x, u) = \mu$, the system is a Brownian motion with drift coefficient μ and diffusion coefficient σ. A Brownian motion can be approximated by a random walk with the time interval approaching zero. First, we have a lemma about random walk.

Lemma A.1: A random walker wanders among $K+1$ points denoted as $0, 1, \cdots, K$. At any point k, $0 < k < K$, he jumps to $k+1$ with probability p, or to $k-1$ with probability $q = 1-p$. Suppose that the random walker starts from point n, and let f_n be the expected number of steps for him to reach, for the first time, the set $\{0, K\}$. If $p = q = 0.5$, then $f_n = n(K-n)$; if $p \neq q$, we have

$$f_1 = \frac{K-1}{p\left(1-\left(\frac{q}{p}\right)^K\right)} - \frac{\frac{q}{p} - \left(\frac{q}{p}\right)^K}{(p-q)\left(1-\left(\frac{q}{p}\right)^K\right)} \tag{64}$$

and

$$f_n = f_1 \frac{1-\left(\frac{q}{p}\right)^n}{1-\frac{q}{p}} + \frac{\frac{q}{p} - \left(\frac{q}{p}\right)^n}{(p-q)\left(1-\frac{q}{p}\right)} - \frac{n-1}{p-q}. \tag{65}$$

Proof: From the definition, we have $f_n = 1 + pf_{n-1} + qf_{n+1}$. With boundary conditions $f_0 = 0$ and $f_K = 0$, the results can be easily verified. ■

Consider a drifted Brownian motion with drift coefficient μ and diffusion coefficient σ. We approximate it with a discrete-time random walk in a standard way. The time interval of each step is Δt, and the size of one jump at each step is Δx. We have

$$\begin{aligned} \Delta x &= \sigma\sqrt{\Delta t} \\ p &= (1+\mu\sqrt{\Delta t})/2 \\ q &= (1-\mu\sqrt{\Delta t})/2. \end{aligned} \tag{66}$$

This random walk converges to the drifted Brownian motion as $\Delta t \to 0$.

When this discrete-time random walk starts from any event level x_d, it may reach the set $\{x_{d-1}, x_d, x_{d+1}\}$ in a finite number of steps. At the next step, it first reaches $x_d - \Delta x$ with probability q or $x_d + \Delta x$ with probability p. We consider the case when $p = q = 0.5$ for simplicity. Suppose that it reaches $x_d + \Delta x$ in the next step. After that, it will reach the set $\{x_d, x_{d+1}\}$. Let t_p be the expected time length of jumping from x_d to $x_d + \Delta x$, and then reaching the set $\{x_d, x_{d+1}\}$. This procedure looks like a random walk in Lemma A.1, with $n = 1$ and $K = (x_{d+1} - x_d)/\Delta x$. From the lemma, we have that the expected number of steps to reach $\{x_d, x_{d+1}\}$ is $f_1 = K - 1 = ((x_{d+1} - x_d)/\sigma\sqrt{\Delta t}) - 1$. The time interval of each step is Δt, so the total expected time is $t_p = (1 + f_1)\Delta t = (1/\sigma)(x_{d+1} - x_d)\sqrt{\Delta t}$. Obviously, $t_p \to 0$ as $\Delta t \to 0$. Next, if starting from x_d the random walk reaches $x_d - \Delta x$ in the next step, we define t_q be the expected time of jumping from x_d to $x_d - \Delta x$ and then reaching the set $\{x_{d-1}, x_d\}$. With the similar derivation, we have $t_q \to 0$ as $\Delta t \to 0$. The expected time from x_d to the set $\{x_{d-1}, x_d, x_{d+1}\}$ is $pt_p + qt_q$, which also goes to zero as $\Delta t \to 0$.

For the case when $p \neq q$, we can calculate t_p and t_q from Lemma A.1 and (66). The same result holds: The expected time

$pt_p + qt_q \to 0$ as $\Delta t \to 0$. Therefore, for a drifted Brownian motion, $\mathrm{E}\{t_{i+1} - t_i\} = \lim_{\Delta t \to 0}(pt_p + qt_q) = 0$. We omit the details here. This completes the proof of the theorem.

Appendix B
Proof of Theorem 3.2

For a periodic chain, $\lim_{l\to\infty}(P^{\mathbf{u}})^l$ does not exist. To prove the theorem, we first construct an equivalent aperiodic Markov chain to implement sample-path-based algorithms. Let

$$\tilde{P}^{\mathbf{u}} = \varepsilon I + (1-\varepsilon)P^{\mathbf{u}} \tag{67}$$

where $0 < \varepsilon < 1$. $\tilde{P}^{\mathbf{u}} = \{\tilde{p}^{\mathbf{u}(d)}(d'|d)\}_{d,d' \in \mathcal{D}}$ is an ergodic transition probability matrix defined by policy $\mathbf{u}$. We use the symbol $\tilde{}$ to denote the quantities associated with $\tilde{P}^{\mathbf{u}}$.

It is easy to verify that the invariant probability row vector of $\tilde{P}^{\mathbf{u}}$ is the same as that of $P^{\mathbf{u}}$: $\tilde{\pi}^{\mathbf{u}} = \pi^{\mathbf{u}}$, and therefore the long-run average performance $\tilde{\eta}^{\mathbf{u}} = \eta^{\mathbf{u}}$. Define the cost function of the aperiodic chain $\tilde{P}^{\mathbf{u}}$ as $\tilde{r}^{\mathbf{u}}_{\delta} = H^{\mathbf{u}}_f - \delta H^{\mathbf{u}}_1$ with $\delta = \tilde{\eta}^{\mathbf{u}}$. Then, we have $\tilde{r}^{\mathbf{u}}_{\delta} = r^{\mathbf{u}}_{\delta}$ and $\tilde{\lambda}^{\mathbf{u}}_{\delta} = \lambda^{\mathbf{u}}_{\delta} = 0$ with $\delta = \eta^{\mathbf{u}}$. Let $\tilde{g}^{\mathbf{u}}$ be the performance potential of the aperiodic MDP $(\tilde{P}^{\mathbf{u}}, r^{\mathbf{u}}_{\delta})$, satisfying the Poisson equation

$$(I - \tilde{P}^{\mathbf{u}})\tilde{g}^{\mathbf{u}} = H^{\mathbf{u}}_f - \eta^{\mathbf{u}} H^{\mathbf{u}}_1. \tag{68}$$

From (67), we have

$$P^{\mathbf{u}} = (\tilde{P}^{\mathbf{u}} - \varepsilon I)/(1-\varepsilon). \tag{69}$$

Substituting (69) into (15) (for $g^{\mathbf{u}}$), and from (68), we have

$$(I - \tilde{P}^{\mathbf{u}})\left[g^{\mathbf{u}} - (1-\varepsilon)\tilde{g}^{\mathbf{u}}\right] = 0. \tag{70}$$

Since $\tilde{P}^{\mathbf{u}}$ is a transition matrix of an ergodic chain, so $(I - \tilde{P}^{\mathbf{u}})$ is a singular matrix with rank $D-1$, and e is an eigenvector of $(I - \tilde{P}^{\mathbf{u}})$. Therefore, the solution to (70) is

$$g^{\mathbf{u}} - (1-\varepsilon)\tilde{g}^{\mathbf{u}} = ce \tag{71}$$

where c is any real constant.

Consider a sample path generated according to $\tilde{P}^{\mathbf{u}}$: $\tilde{\varpi} = \{\tilde{d}_0, \tilde{d}_1, \ldots\}$. We first choose any event d^* as a reference event. For any event $d \in \mathcal{D}$, we define the stopping time $\tilde{\tau}_{d^*}(d) = \min\{i : i > 0, \tilde{d}_i = d^* | \tilde{d}_0 = d\}$ on $\tilde{\varpi}$. Since the chain is ergodic, we have [9]

$$\tilde{g}^{\mathbf{u}}(d) = \mathrm{E}\left\{\sum_{i=0}^{\tilde{\tau}_{d^*}(d)-1} r^{u_i}_{\delta}(\tilde{d}_i) | \tilde{d}_0 = d\right\} \tag{72}$$

where $u_i = \mathbf{u}(\tilde{d}_i)$ and $\delta = \eta^{\mathbf{u}}$.

Now consider a sample path generated according to the periodic $P^{\mathbf{u}}$ as $\varpi = \{d_0, d_1, \ldots\}$. Note that the sample path $\tilde{\varpi}$ can be constructed from ϖ according to (67). Specifically, for any sample path $\varpi = \{d_0, d_1, \ldots\}$, we set $\tilde{\varpi} = \{\tilde{\varpi}_0, \tilde{\varpi}_1, \ldots\}$, where $\tilde{\varpi}_i = \{d_i, d_i, \ldots, d_i\}$ is a sequence of k_i consecutive visits to state d_i, with k_i being a geometrically distributed integer with parameter ϵ. From (67), $\tilde{\varpi}$ is a sample path of the aperiodic chain $\tilde{P}^{\mathbf{u}}$. The cost accumulated on $\tilde{\varpi}_i$ is $\Phi(\tilde{\varpi}_i) = k_i r^{u_i}_{\delta}(d_i)$, and its mean is

$$\mathrm{E}\left\{\Phi(\tilde{\varpi}_i)|d_i\right\} = \frac{1}{1-\varepsilon} r^{u_i}_{\delta}(d_i). \tag{73}$$

On the sample path ϖ, define the stopping time $\tau_{d^*}(d) = \min\{i : i > 0, d_i = d^* | d_0 = d\}$. Construct a sample path $\tilde{\varpi}$ from ϖ. By Wald's equation [25], we can rewrite (72) as (recall that the performance with $r^{\mathbf{u}}_{\delta}$ is zero)

$$\begin{aligned}
\tilde{g}^{\mathbf{u}}(d) &= \mathrm{E}\left\{\sum_{i=0}^{\tau_{d^*}(d)-1} \sum_{k=0}^{k_i-1} r^{u_i}_{\delta}(d_i) | d_0 = d\right\} \\
&= \mathrm{E}\left\{\sum_{i=0}^{\tau_{d^*}(d)-1} \Phi(\tilde{\varpi}_i) | d_0 = d\right\} \\
&= \mathrm{E}\left\{\sum_{i=0}^{\tau_{d^*}(d)-1} \mathrm{E}\left\{\Phi(\tilde{\varpi}_i)|d_i\right\} | d_0 = d\right\} \\
&= \frac{1}{1-\varepsilon}\mathrm{E}\left\{\sum_{i=0}^{\tau_{d^*}(d)-1} r^{u_i}_{\delta}(d_i) | d_0 = d\right\}.
\end{aligned} \tag{74}$$

Substituting (74) into (71), we have

$$g^{\mathbf{u}}(d) = \mathrm{E}\left\{\sum_{i=0}^{\tau_{d^*}(d)-1} r^{u_i}_{\delta}(d_i) | d_0 = d\right\} + ce \tag{75}$$

which leads to (17) (with an additive constant). Theorem 3.2 is proved.

Appendix C
Proof of Theorem 5.1

Rewrite (46) as

$$\begin{aligned}
Q_l(d,u) &= \frac{1}{\sum_{j=0}^{Y_l-1}\chi_j(d,u)} \sum_{j=0}^{Y_l-1}\chi_j(d,u) \sum_{i=\gamma_j(d,u)}^{\beta_{j+1}-1} r^{u_i}_{\delta}(d_i) \\
&+ \frac{1}{\frac{1}{Y_l}\sum_{j=0}^{Y_l-1}\chi_j(d,u)} \frac{1}{Y_l}\sum_{j=0}^{Y_l-1}\chi_j(d,u) \\
&\times \sum_{i=\gamma_j(d,u)}^{\beta_{j+1}-1}\left[r^{u_i}_{\delta,i}(d_i) - r^{u_i}_{\delta}(d_i)\right].
\end{aligned} \tag{76}$$

By (44) and the strong law of large numbers, the first term on the right-hand side of (76) converges to $Q(d,u)$ w.p.1 as $l \to \infty$. The first fraction in the second term on the right-hand side of (76), $1/(1/Y_l \sum_{j=0}^{Y_l-1}\chi_j(d,u))$, converges to $1/\mathcal{P}(\chi_j(d,u) = 1)$, w.p.1 as $l \to \infty$. $\mathcal{P}(\chi_j(d,u)=1)$ is the probability that the pair (d,u) appears in a regenerative period. Let Ω_l denote the second part

$$\Omega_l = \frac{1}{Y_l}\sum_{j=0}^{Y_l-1}\chi_j(d,u)\sum_{i=\gamma_j(d,u)}^{\beta_{j+1}-1}\left[r^{u_i}_{\delta,i}(d_i) - r^{u_i}_{\delta}(d_i)\right].$$

Define $\phi(d',u') = \{i : d_i = d', u_i = u'$, and $\exists j, \gamma_j(d,u) \le i \le \beta_{j+1} - 1\}$. Then, we can rearrange the sum in Ω_l as follows:

$$\Omega_l = \sum_{d' \in \mathcal{D}, u' \in U} \frac{\Theta_{(d',u')}}{Y_l} \frac{1}{\Theta_{(d',u')}} \sum_{i \in \phi(d',u')}\left[r^{u'}_{\delta,i}(d') - r^{u'}_{\delta}(d')\right] \tag{77}$$

where $\Theta_{(d',u')}$ is the number of visits to the event–action pair (d',u') before the (Y_l+1)th regenerative point along the sample path. Note that $\Theta_{(d',u')}$ is a function of (d,u), and we omit (d, u) here for simplicity. Then, Ω_l is also a function of (d, u). It is obvious that the number of elements in set $\phi(d', u')$ is no more than $\Theta_{(d',u')}$. Let $\mathcal{L}_d(u)$ be the probability of applying action u when event is d, according to policy $\hat{\mathbf{u}}_\epsilon$. Then, we have $\Theta_{(d',u')}/Y_l \rightarrow \pi(d')\mathcal{L}_{d'}(u')\mathrm{E}[\tilde{\tau}_{d^*}(d^*)]$ (which is the mean number of visits to (d', u') in a regenerative period) w.p.1 as $l \rightarrow \infty$, and

$$\lim_{\Theta_{(d',u')}\rightarrow\infty} \frac{1}{\Theta_{(d',u')}} \sum_{i\in\phi(d',u')} \left[r^{u'}_{\delta,i}(d') - r^{u'}_{\delta}(d')\right] = 0, \quad \text{w.p.1} \tag{78}$$

follows directly from

$$\begin{aligned}\limsup_{\Theta_{(d',u')}\rightarrow\infty} &\left| \frac{1}{\Theta_{(d',u')}} \sum_{i\in\phi(d',u')} \left[r^{u'}_{\delta,i}(d') - r^{u'}_{\delta}(d')\right]\right| \\ &\leq \limsup_{\Theta_{(d',u')}\rightarrow\infty} \frac{1}{\Theta_{(d',u')}} \sum_{n=1}^{\Theta_{(d',u')}} \left|r^{u'}_{\delta,i_n}(d') - r^{u'}_{\delta}(d')\right| \\ &= \lim_{n\rightarrow\infty} \left|r^{u'}_{\delta,i_n}(d') - r^{u'}_{\delta}(d')\right| = 0, \qquad \text{w.p.1}\end{aligned}$$

where i_n is the time epoch of the nth visit to event–action pair (d', u'). The last two equalities follow from $\lim_{l\rightarrow\infty} r^u_{\delta,l}(d) = r^u_\delta(d)$. Since $\Theta_{(d',u')} \rightarrow \infty$ w.p.1 as $l \rightarrow \infty$, (47) follows directly from (76)–(78).

References

[1] H. Abou-Kandil, O. D. Smet, G. Freiling, and G. Jank, "Flow control in a failure-prone multi-machine manfacturing system," in *Proc. INRIA/IEEE Symp. Emerg. Technol. Factory Autom.*, 1995, vol. 2, pp. 575–583.

[2] J. Abounadi, D. Bertsekas, and V. S. Borkar, "Learning algorithms for Markov decision processes with average cost," *SIAM J. Control Optim.*, vol. 40, no. 3, pp. 681–698, 2001.

[3] K. E. Arzen, "A simple event-based PID controller," in *Proc. IFAC World Cong.*, Beijing, China, 1999, vol. 18, pp. 423–428.

[4] K. J. Astrom and B. M. Bernhardsson, "Comparison of Riemann and Lebesgue sampling for first order stochastic systems," in *Proc. 41th IEEE Conf. Decision Control*, Las Vegas, NV, USA, December 2002.

[5] K. J. Astrom, K. Johan, and B. Wittenmark, *Computer-Controlled Systems*, 3rd ed. Englewood Cliffs, NJ: Prentice-Hall, 1997.

[6] B. Bernhardsson, "Event triggered sampling," in *Research Problem Formulations in the DICOSMOS Project*, M. Torngren and M. Sanfridson, Eds. Lund, Sweden: Lund Inst. Technol. Press, 1998.

[7] D. P. Bertsekas and T. N. Tsitsiklis, *Neuro-Dynamic Programming*. Belmont, MA: Athena Scientific, 1996.

[8] M. S. Branicky, V. S. Borkar, and S. Mitter, "A unified framework for hybrid control," *IEEE Trans. Autom. Control*, vol. 43, no. 1, pp. 31–45, Jan. 1998.

[9] X. R. Cao, *Stochastic Learning and Optimization—A Sensitivity-Based Approach*. New York: Springer, 2007.

[10] X. R. Cao, Z. Ren, S. Bhatnagar, M. Fu, and S. Marcus, "A time aggregation approach to Markov decision processes," *Automatica*, vol. 38, pp. 929–943, 2002.

[11] S. DeWeerth, L. Nielsen, C. Mead, and K. J. Astrom, "A neuron-based pulse servo for motion control," in *Proc. IEEE Int. Conf. Robot. Autom.*, Cincinnati, OH, 1990, vol. 3, pp. 1698–1703.

[12] J. P. Forestier and P. Varaiya, "Multilayer control of large Markov chains," *IEEE Trans. Autom. Control*, vol. AC-23, no. 2, pp. 298–305, Apr. 1978.

[13] S. Karlin and H. M. Taylor, *A Second Course in Stochastic Processes*. San Diego, CA: Academic, 1981.

[14] R. McCann, A. K. Gunda, and S. D. Damugatla, "Improved operation of networked control systems using Lebesgue sampling," in *Proc. Ind. Appl. Conf.*, 2004, vol. 2, pp. 1211–1216.

[15] M. Miskowicz, "The event-triggered sampling optimization criterion for distributed networked monitoring and control systems," in *Proc. IEEE Int. Conf. Ind. Technol.*, Maribor, Slovenia, 2003, pp. 1083–1088.

[16] M. Miskowicz and S. Kuta, "Application-driven flow control in distributed monitoring and control systems," in *Proc. IEEE Int. Conf. Ind. Technol.*, Maribor, Slovenia, 2003, pp. 421–425.

[17] C. De Persis, "N-bit stabilization of n-dimensional nonlinear systems in feedforward form," *IEEE Trans. Autom. Control*, vol. 30, no. 3, pp. 299–311, Mar. 2005.

[18] N. Persson and F. Gustafsson, "Event based sampling with application to vibration analysis in pneumatic tires," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Salt Lake City, UT, 2001, pp. 3885–3888.

[19] M. L. Puterman, *Markov Decision Processes*. New York: Wiley, 1994.

[20] M. Rabi and J. S. Baras, "Sampling of diffusion processes for real-time estimation," in *Proc. IEEE Conf. Decision Control*, Atlantis, Bahamas, Dec. 2004, vol. 4, pp. 4163–4168.

[21] S. Sastry, *Nonlinear Systems: Analysis, Stability, and Control*. New York: Springer-Verlag, 1999.

[22] J. Si, A. G. Barto, W. B. Powell, and D. Wunsch, *Handbook of Learning and Approximate Dynamic Programming*. Hoboken, NJ: Wiley-IEEE Press, 2004.

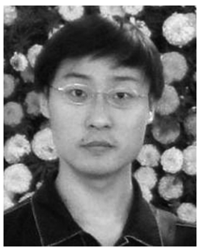
[23] H. Sira-Ramirez, "A geometric approach to pulse-width modulated control in nonlinear dynamical systems," *IEEE Trans. Autom. Control*, vol. 34, no. 2, pp. 184–187, Feb. 1989.

[24] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

[25] A. Wald, "Fitting of straight lines if both variables are subject to error," *Ann. Math. Stat.*, vol. 11, pp. 284–300, 1940.

[26] P. J. Werbos, "Approximate dynamic programming for real-time control and neural modeling," in *Handbook of Intelligent Control*, D. A. White and D. A. Sofge, Eds. New York: Van Nostrand Reinhold, 1992.

[27] K. J. Zhang, Y. K. Xu, X. Chen, and X. R. Cao, "Policy iteration based feedback control," *Automatica*, vol. 44, no. 4, pp. 1055–1061, 2008.



Yan-Kai Xu received the doctoral degree in automatic control from the Center for Intelligent and Networked Systems (CFINS), Tsinghua University, Beijing, China, in 2008.

He currently works for the Beijing GeoScience Center, Schlumberger Ltd., Beijing, China, as a Project Engineer. His research interests include optimization and control of stochastic systems, discrete event dynamic systems, and machine learning.

Xi-Ren Cao (S'82–M'84–F'96) received the M.S. and Ph.D. degrees from Harvard University, Cambridge, MA, in 1981 and 1984, respectively.

From 1984 to 1986, he was a Research Fellow with Harvard University. From 1986 to 1993, he worked as Consultant Engineer/Engineering Manager with Digital Equipment Corporation, Marlboro, MA. From 1993 to 2010, he was with the Hong Kong University of Science and Technology (HKUST), Hong Kong, where he served as a Reader/Professor/Chair Professor. Since July 2010, he has been a Chair Professor with Shanghai Jiao Tong University, Shanghai, China, and an Affiliate Member of the Institute for Advanced Study, Hong Kong University of Science and Technology. He owns three patents in data communications and telecommunications and has published three books in the area of performance optimization and discrete-event dynamic systems. His current research areas include and financial engineering, stochastic learning and optimization, performance analysis of economic systems, and discrete-event dynamic systems.

Dr. Cao has been a Fellow of the International Federation of Automatic Control (IFAC) since 2008. He has been the Chairman of IEEE Fellow Evaluation Committee of IEEE Control System Society, Editor-in-Chief of *Discrete Event Dynamic Systems: Theory and Applications*, and Associate Editor at Large of the IEEE TRANSACTIONS OF AUTOMATIC CONTROL. He has served on the Board of Governors of the IEEE Control Systems Society and on the Technical Board of IFAC. He received the Outstanding Transactions Paper Award from the IEEE Control System Society in 1987, the Outstanding Publication Award from the Institution of Management Science in 1990, the Outstanding Service Award from IFAC in 2008, and the National Natural Science Award (2nd class), China, in 2009.