Asian Journal of Control, Vol. 16, No. 6, pp. 1735–1743, November 2014
Published online 21 March 2014 in Wiley Online Library (wileyonlinelibrary.com) DOI: 10.1002/asjc.874

POLICY GRADIENT APPROACH OF EVENT-BASED OPTIMIZATION AND ITS ONLINE IMPLEMENTATION

Li Xia

ABSTRACT

In the theory of event-based optimization (EBO), the decision making is triggered by events, which is different from the traditional state-based control in Markov decision processes (MDP). In this paper, we propose a policy gradient approach of EBO. First, an equation of performance gradient in the event-based policy space is derived based on a fundamental quantity called Q-factors of EBO. With the performance gradient, we can find the local optimum of EBO using the gradient-based algorithm. Compared to the policy iteration approach in EBO, this policy gradient approach does not require restrictive conditions and it has a wider application scenario. The policy gradient approach is further implemented based on the online estimation of Q-factors. This approach does not require the prior information about the system parameters, such as the transition probability. Finally, we use an EBO model to formulate the admission control problem and demonstrate the main idea of this paper. Such online algorithm provides an effective implementation of the EBO theory in practice.

Key Words: event-based optimization, discrete event system, Markov decision process, Q-factor, performance potential.

I. INTRODUCTION

Markov decision processes (MDP) are a well-known optimization framework in the performance optimization of stochastic systems, and have been richly studied in the literature [5,13,22]. In the framework of MDP, we have to select an action at every state, which is called state-based control. Nevertheless, in many practical systems, the action is selected only when certain event happens. We call this event-based control. For example, in a wireless sensor network for task surveillance, the sensor nodes communicate only when an intrusion event occurs in order to save energy [7]. In an intelligent building energy system, the controller adopts a new action of the building devices, such as the tuning of heating ventilaton and air conditioning (HVAC) and lighting, only when the event of the environment deviating from the occupants' comfort zone occurs [8,23].

How to choose a proper decision rule (or event-based policy) to make the associated system performance optimal is called the event-based optimization problem [4,5]. The events usually have their own physical meanings in practice, such as an intrusion in the wireless sensor network or an environment's deviation from the comfort zone in the intelligent building. In the mathematical formulation, we define an event as a set of state transitions that possess some common properties, where the state is usually defined with the underlying Markov systems. With this definition, we can formulate such an event-based optimization (EBO) problem using the basic framework of Markov systems, in which the system dynamics are driven by the underlying state transitions. Nevertheless, the series of event occurrences are usually not Markovian and we cannot directly formulate this optimization problem as an MDP. Therefore, the EBO problem does not fit the framework of the standard MDP theory and we have to resort to other new optimization approaches.

The theory of EBO was proposed recently [4,5,16,17,29], and it is intended to provide an effective approach to analyze and optimize the system performance of EBO problems. It extends the idea of the direct-comparison based approach in Markov systems [3,5] and directly compares the performance difference under two event-based policies. With the performance difference equation, the policy iteration is developed [4]. Nevertheless, the optimality of the policy iteration for EBO requires a condition, that is, the conditional probability of the system state given an event happening is irrelevant to the policy [4,29]. This requirement may not be satisfied in some practical systems.

Since the EBO theory is under development, only the basic theoretical framework of EBO has been proposed. More effort needs to be devoted to the detailed implementation of the related algorithms. For example, in a real system, some system parameters, such as the transition probability, are unknown, and the system state may be unobservable. We have to estimate or learn these parameters during the

Manuscript received April 1, 2013; revised July 31, 2013; accepted September 28, 2013.

The author is with the Center for Intelligent and Networked Systems (CFINS), Department of Automation, TNList, Tsinghua University, Beijing 100084, China (Tel: +86-10-62793029, e-mail: xial@tsinghua.edu.cn).

This work was supported in part by the National Natural Science Foundation of China (61203039, U1301254, 60736027), the National 111 International Collaboration Project (B06002), the Specialized Research Fund for the Doctoral Program of Higher Education (20120002120009), and the TNList Cross-Discipline Foundation.

implementation of optimization algorithms. A recursive algorithm with learning capability is desirable in such a scenario.

In this paper, we mainly discuss the two aforementioned problems. About the first problem of the restriction of policy iteration, we propose a policy gradient optimization approach instead of the policy iteration to approach to the local optimum of the EBO problem. This approach is based on the performance derivative equation of EBO, and it does not require the strict condition to guarantee its optimality. About the second problem of the algorithm implementation, we propose a definition called the Q-factor of EBO, which is similar to the Q-learning of reinforcement learning [18,24]. The Q-factor can be viewed as a fundamental quantity in the EBO approaches and it can be learned recursively based on a single sample path. Therefore, the EBO algorithm based on Q-factors can be implemented online, without knowing the full information of the system. We call it Q-factor based online learning implementation of EBO approaches. Finally, we discuss an example of the admission control problem. The EBO algorithm is implemented to control the admission probability when the event of customer arrival happens. The policy gradient approach with Q-factors can find the optimum of this problem. This example demonstrates the main idea of this paper.

II. FORMULATION AND BASIC THEORY OF EBO

The Markov model is a very general model, and it can be used to model any system after some tricks of state definition techniques [22]. The model of EBO is also built on the basic framework of the Markov model [4]. Consider a discrete time discrete state Markov chain $\boldsymbol{X} = \{X_0, X_1, \dots\}$, where X_t is the system state at time epoch t, $t = 0, 1, \dots$. The state space is assumed finite, and its size is S. Without loss of generality, we denote the state space as $\mathcal{S} := \{1, 2, \cdots, S\}$. The Markov chain will transit from the current state $X_t = i$ to the next state $X_{t+1} = j$ with probability p_{ij}, where $i, j \in \mathcal{S}$. The state transition probability matrix is denoted as $P = [p_{ij}]_{i,j\in\mathcal{S}}$.

In the real world, an event is always related to certain physical meaning, such as the invader intrusion in sensor networks or the blind turn-down in the energy-efficient building. In mathematical formulation, an event is defined as a set of state transitions that possess some common properties. Thus, we define event $e := \{\langle i, j\rangle : i, j \in \mathcal{S}$ and $\langle i, j\rangle$ possess common properties$\}$. These common properties usually have certain physical meaning, such as the invader appearance or the blind position being down. Since the state space is finite, the number of events is also finite, and we denote the event space as $\mathcal{E}$. Assume the size of $\mathcal{E}$ is V. For simplicity, we define $\mathcal{E} := \{1, 2, \cdots, V\}$. For the sake of completeness, we define a virtual event e_ϕ which means that no action can be selected at the state transitions $\langle i, j\rangle \in e_\phi$. That is, this event e_ϕ cannot trigger decision making, or the action at these transitions is fixed. For simplicity, we omit such events e_ϕ from the definition of the event space $\mathcal{E}$ since they have no effect on the system performance. The input state of an event e is defined as $I(e) := \{\text{all } i : \langle i, j\rangle \in e, \exists j\}$. The output state of event e is defined as $O(e) := \{\text{all } j : \langle i, j\rangle \in e, \exists i\}$. The output state of event e on the input state i is defined as $O_i(e) := \{\text{all } j : \langle i, j\rangle \in e\}$.

When an event e happens, we have to select an action a from the action space $\mathcal{A}$. We assume the action space $\mathcal{A}$ is finite. Different action a will affect the transition probability and system reward. When event e happens and we adopt an action a, the system state will transit from i to j with probability $p^a(j|i, e)$, where $e \in \mathcal{A}$, $a \in \mathcal{A}$, $i \in I(e)$, and $j \in O_i(e)$. Please note that we usually observe only the event e and that the underlying state i is unobservable. The action will also affect the system reward, and we rewrite it as $r(i, a) \in \mathcal{R}$, where $i \in \mathcal{S}$ and $a \in \mathcal{A}$. An event-based policy d determines the rule of action selection according to various events $e \in \mathcal{E}$. That is, d is a mapping from the event space $\mathcal{E}$ to the action space $\mathcal{A}$, $d : \mathcal{E} \to \mathcal{A}$. All of the possible event-based policies comprise the policy space $\mathcal{D}$ which is assumed finite. Here, we only consider the stationary and deterministic policy d since it has good applicability in practice. Please note, however, a general policy in EBO can be related to the history of events and actions since the event sequence is usually not Markovian.

Denote t_l as the time of the lth event happening, $l = 1, 2, \dots$. Therefore, E_{t_l} is denoted as the event happening at time t_l. A_{t_l} is denoted as the action adopted at time t_l for the lth event happening. During the period between t_l and t_{l+1}, there is no event happening. R_t is the system reward at time t and it is determined by $R_t = r(X_t, A_t)$. The action sequence is determined by the event-based policy, *i.e.*, $A_{t_l} = d(E_{t_l})$. When $t_l < t < t_{l+1}$, A_t denotes a virtual action "no control adopted" for the sake of notational consistency. The long-run average performance of the system is

$$\eta = E\{R_t\} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} r(X_t, A_t), \tag{1}$$

where we assume that the system is ergodic and η is irrelevant to the initial system state. Denote $\pi(i)$ as the steady state distribution at state i, $i \in \mathcal{S}$. The row vector of steady state distribution is denoted as $\pi := (\pi(1), \pi(2), \dots, \pi(S))$. The column vector of reward is denoted as $r := (r(1), r(2), \dots, r(S))^T$, where we omit the action-related factor for simplicity. From this, we have:

$$\eta = \pi r. \tag{2}$$

Different policy d will affect the system parameters. We can use η^d, π^d, r^d to replace the corresponding terms in (1) and

(2). The transition probability matrix is also rewritten as P^d to reflect the effect of policy. The goal of EBO is to find an optimal policy d^* from the policy space $\mathcal{D}$, which maximizes the long-run average performance. That is,

$$\begin{aligned} d^* &= \arg\max_{d\in\mathcal{D}} \eta^d \\ &= \arg\max_{d\in\mathcal{D}} \left\{ \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} r(X_t, d(E_t)) \right\}. \end{aligned} \tag{3}$$

The direct-comparison based approach is the main basis of EBO theory. For any two event-based policy d and h, $d, h \in \mathcal{D}$, their performance difference is quantified by the following difference equation [5,29]

$$\begin{aligned} \eta^h - \eta^d = \sum_{e=1}^{V} \pi^h(e) \sum_{i\in\mathcal{I}(e)} \pi^h(i|e) \Bigg\{ \sum_{j\in O_i(e)} [p^{h(e)}(j|i,e) \\ - p^{d(e)}(j|i,e)] g^d(j) + [r^h(i) - r^d(i)] \Bigg\}, \end{aligned} \tag{4}$$

where g is called the performance potential and it is defined as below

$$g(i) = E\left\{ \lim_{T\to\infty} \sum_{t=0}^{T} [r(X_t, A_t) - \eta] | X_0 = i \right\}. \tag{5}$$

The row vector g can be determined by solving the Poisson equation:

$$(I - P + \mathbf{1}\pi) g = r, \tag{6}$$

where $\mathbf{1}$ is an S-dimensional column vector with all elements 1. g can also be estimated based on the statistics of the system sample path (please see chapter 3.1 of [5]).

A policy iteration algorithm directly follows the difference equation (4), when the following condition is satisfied:

$$\pi^h(i|e) = \pi^d(i|e), \quad \forall e \in \mathcal{E}, i \in I(e), d, h \in \mathcal{D}. \tag{7}$$

The condition above indicates that the conditional probability $\pi(i|e)$ is irrelevant to the underlying event-based policy. With this condition, policy iteration is proposed for EBO and it is proven to converge to the global optimum within a finite number of iterations [5]. In practice, there are some categories of problems satisfying this condition [5,26,28].

Nevertheless, this condition (7) is restrictive and not all problems can satisfy it in practice. For the problems not satisfying this condition, the optimality of policy iteration cannot be guaranteed and we have to develop other optimization approaches. Moreover, in many practical systems, the system state is unobservable. We only observe the historical sequence of events and system rewards. Even some system parameters can be unknown to the decision maker. These above difficulties make the policy iteration based on (4) infeasible. Approximation techniques, such as reinforcement learning [18,24] or neuro-dynamic programming [2], may be utilized to handle these difficulties. How to solve EBO in such scenarios is the main content of our discussion in the next section.

III. POLICY GRADIENT APPROACH AND ONLINE IMPLEMENTATION

The policy gradient approach is an alternative approach when policy iteration is not applicable. It is intended to calculate or estimate the gradient of the system performance with respect to policy (random policy or parameterized policy) and apply the gradient-based optimization algorithm to find the local optima. It has been studied in the literature of perturbation analysis [14,19,32], MDP [3,10,21,31], and reinforcement learning [1,25]. Below, we study the policy gradient approach for the EBO problem.

First, we study the performance derivative equation based on (4). Consider a random policy d_δ^h that adopts policy d with probability $1 - \delta$ and adopts policy h with probability δ, where $d, h \in \mathcal{D}$ and $0 \le \delta \le 1$. Thus, d_δ^h determines a policy-varying direction from d to h in the policy space. One can see that, when $\delta = 0$, $d_\delta^h = d$. When $\delta = 1$, $d_\delta^h = h$. Consider the performance difference of the system under policy d and d_δ^h. Applying (4) and letting $\delta \to 0$, we can derive the following derivative equation:

$$\begin{aligned} \frac{\partial \eta^d}{\partial \delta} = \sum_{e=1}^{V} \pi^d(e) \sum_{i\in I(e)} \pi^d(i|e) \Bigg\{ \sum_{j\in O_i(e)} [p^{h(e)}(j|i,e) \\ - p^{d(e)}(j|i,e)] g^d(j) + [r^h(i) - r^d(i)] \Bigg\}. \end{aligned} \tag{8}$$

With respect to the above derivative equation, we see that it does not require Condition (7) since $\pi^h(i|e)$ is not needed in (8). For the theoretical calculation, all of the values of the terms in the right-hand-side of (8) are easy to obtain. Parameters $\pi^d(e)$, $\pi^d(i|e)$, and g^d can be numerically calculated or estimated from the sample path of the system under policy d. Parameters $p^a(j|i, e)$, r^h, and r^d are usually known to the decision maker. Therefore, it is feasible to execute (8) during the optimization process. With the information of the performance derivative, we can use gradient-based algorithms to find the local optima of the EBO problem. How to choose a proper gradient-based algorithm is not the emphasis of this paper, and there are many traditional algorithms in the literature [9].

Nevertheless, many optimization algorithms are required to be executed along with the online operation of the system. The system model may be a "black box" for the decision maker. For example, we do not know the exact values of the system parameters, such as the values of $p^h(j|i, e)$, $p^d(j|i, e)$, r^h, and r^d. Furthermore, in some situations, the system state is unobservable. The decision maker can only observe the sequence of events occurrence E_t and system reward $r(X_t, A_t)$. This makes the estimation of $\pi^d(i|e)$ and $g^d(i)$ impossible since we cannot observe state i.

The above situation exists in practice very commonly. How to solve this problem is meaningful to apply the optimization framework of EBO to practical situations. In the traditional policy iteration or the difference equation (4), we use the performance potential $g(i)$ as a fundamental quantity to evaluate the quality of action a, see the term $\sum_{j\in O_i(e)} p^a(j|i,e)g^d(j)$ in (4) where $a = h(e)$. Since the system state i is unobservable for the decision maker, we have to use other quantities to quantify the action quality instead of the performance potential $g(i)$. Notice that the event sequence is observable and the action adopted is controlled by the decision maker. So, one of the natural conjectures is whether we can use a quantity with an event index e to evaluate the performance quality of action selection. The answer is YES. $Q(e, a)$ is exactly such a quantity and it is defined as below:

$$Q(e,a) := \sum_{j\in S} p^a(j|e)g(j) + r(e,a), \tag{9}$$

where $p^a(j|e)$ is the probability that the system state transits to j with the condition of the current event being e and the action adopted being a. With the formulation of EBO in Section II, we have:

$$p^a(j|e) = \sum_{i\in I(e)} p(i|e)p^a(j|i,e), \tag{10}$$

where $p(i|e)$ is the conditional probability of state i given the current event e. Similarly, we have:

$$r(e,a) = \sum_{i\in I(e)} p(i|e)r(i,a). \tag{11}$$

When we investigate the system statistics at the steady state with policy d, we can also write $p(i|e)$ as $\pi^d(i|e)$. Therefore, the definition (9) of $Q(e, a)$ can be rewritten as:

$$Q^d(e,a) = \sum_{i\in I(e)} \pi^d(i|e)\left[\sum_{j\in O_i(e)} p^a(j|i,e)g^d(j) + r(i,a)\right], \tag{12}$$

where the superscript d indicates the above statistics is for the system dynamics under policy d. When Condition (7) holds, we substitute (12) into (4) and obtain:

$$\eta^h - \eta^d = \sum_{e=1}^{V} \pi^h(e)[Q^d(e,h(e)) - Q^d(e,d(e))]. \tag{13}$$

From the above difference equation, we see that $Q^d(e, a)$ is a fundamental quantity to quantify the relation between the system performance and the policy. Similar to the Q-learning, we also call $Q^d(e, a)$ as *Q-factor* of EBO. We can see that Q-factor $Q^d(e, a)$ quantifies the long-term contribution of the decision of selecting action a when event e happens. One intuitive idea is to always choose the action a that makes $Q^d(e, a)$ maximal, that is,

$$a = \arg\max_{a\in\mathcal{A}} Q^d(e,a). \tag{14}$$

The above action update scheme is equivalent to the policy iteration in EBO, and the corresponding optimality equation can be derived thereafter. Details can be found in [5,29]. Nevertheless, as we clarified before, the optimality of the above scheme requires Condition (7). When (7) does not hold, we alternatively have the derivative equation (8). In the definition of Q-factor (12), setting $a = h(e)$ and $a = d(e)$ and substituting them into (8), we have:

$$\frac{\partial\eta^d}{\partial\delta} = \sum_{e=1}^{V} \pi^d(e)\left[Q^d(e,h(e)) - Q^d(e,d(e))\right]. \tag{15}$$

Please note that the above equation does not require Condition (7). More importantly, the system state i and transition probability $p^a(j|i, e)$ do not appear in (15). Only the event e and the action a appear in (15). Therefore, (15) is exactly suitable to the situation where we only observe the event occurrence, the system state is unobservable, and the transition probability is unknown. To apply (15), we have to calculate or estimate the Q-factor $Q^d(e, a)$ based on the observation of the system sample path.

The left task is to obtain the value of $Q^d(e, a)$ by only observing the sequences of event occurrence and system rewards. A typical scenario is described as follows. At time t, if we observe an event E_t happening, we adopt an action $A_t = d(E_t)$ and get an instant reward $r(X_t, A_t)$, where X_t is the inner state and it is not observable. The time clock moves to $t + 1$ and the above process repeats. Below, we discuss how to estimate $Q^d(e, a)$ based on the observations on the system sample path.

First, combining with the definition (5) of $g(j)$, we see that (12) is equivalent to

$$Q^d(e,a) = E\left\{\lim_{T\to\infty}\sum_{t=0}^{T} r(X_t,A_t) - T\eta^d \,|E_0 = e, A_0 = a\right\}, \tag{16}$$

where the action sequence A_t, $t > 0$, is determined by a predefined policy d. Since $r(X_t, A_t)$ is statistically equal to η when t is large enough, we can truncate the summation in (16) to T to approximate $Q^d(e, a)$. Thus, we use $Q_T^d(e,a)$ as an estimate of $Q^d(e, a)$ when T is large enough. That is,

$$Q_T^d(e,a) = E\left\{\sum_{t=0}^{T} r(X_t,A_t) - T\eta^d \,|E_0 = e, A_0 = a\right\}. \tag{17}$$

For a predefined policy d, the term $T\eta^d$ is a constant. Notice that, if $Q^d(e, a)$ is a Q-factor of the EBO, then $Q^d(e, a) + c$ with any constant c is also a Q-factor of the EBO. This assertion can be verified in (13), (14), and (15). Therefore, we can add a constant $T\eta^d$ in (17) and derive the following estimate of the Q-factor:

$$\hat{Q}_T^d(e,a) = E\left\{\sum_{t=0}^{T} r(X_t, A_t) | E_0 = e, A_0 = a\right\}. \quad (18)$$

The above estimate of the Q-factor $Q^d(e, a)$ is very easy to implement on the sample path. When we observe an event e happening at the current time, we adopt an action a. Then, the system evolves according to policy d. We summate the system rewards in the succeeding T time epochs as a sample of $\hat{Q}_T^d(e,a)$. Repeating this process, when the number of samples is large enough, we can obtain $\hat{Q}_T^d(e,a)$ with high accuracy.

Nevertheless, there is another issue we need to solve. We want to know the value of the Q-factor $Q^d(e, a)$ at every action $a \in \mathcal{A}$. When the underlying policy d is a random policy, the system sample path can visit every action $a \in \mathcal{A}$ with every event $e \in \mathcal{E}$. When the underlying policy d is deterministic, however, only the action $a = d(e)$ can be visited on the sample path of policy d. In order to make the sample path visit every event-action pair (e, a), we have to adopt an ε-exploration technique similar to the Q-learning approach in reinforcement learning [15,18,24].

We define a d_ε policy to make the sample path visit all the possible event-action pair (e, a)'s. If d is deterministic, d_ε policy is defined as: when event e happens, we choose action $a = d(e)$ with probability $1 - \varepsilon$ and we choose all actions $a' \in \mathcal{A}$ evenly with probability $\varepsilon/|\mathcal{A}|$. When $\varepsilon \to 0$, the system statistics under policy d_ε is approaching that under policy d, while every event-action pair can be visited. The estimation of $Q^d(e, a)$ is summarized in the following algorithm.

Algorithm 1. Sequential online estimation algorithm of Q-factor $Q^d(e, a)$ in EBO.

Initialization
- Choose a large enough number T and a very small positive number $\varepsilon \to 0$. Set $\hat{Q}_T^d(e,a) = 0$, $I(e, a) = 0$, and $N(e, a) = 0$, for all $e \in \mathcal{E}$ and $a \in \mathcal{A}$.

Action Generation
- Run the system under policy d_ε, *i.e.*, when an event e is observed, choose action $a = d(e)$ with probability $1 - \varepsilon$, choose action $a' \in \mathcal{A}$ evenly with probability $\dfrac{\varepsilon}{|A|}$.

Estimation
- At the current time t, suppose the event observed is E_t, the action adopted is A_t, and the system reward observed is R_t. If $I(E_t, A_t) = 0$, set $I(E_t, A_t) = 1$ and $N(E_t, A_t) := N(E_t, A_t) + 1$. For every (e, a) pair, if $T > I(e, a) > 0$, update $I(e, a) := I(e, a) + 1$ and $\hat{Q}_T^d(e,a) := \hat{Q}_T^d(e,a) + R_t$; otherwise, set $I(e, a) = 0$.

Output
- When the sample path is long enough, stop and output $\hat{Q}_T^d(e,a) := \hat{Q}_T^d(e,a)/N(e,a)$ as an estimate of Q-factor $Q^d(e, a)$, for all $e \in \mathcal{E}$ and $a \in \mathcal{A}$.

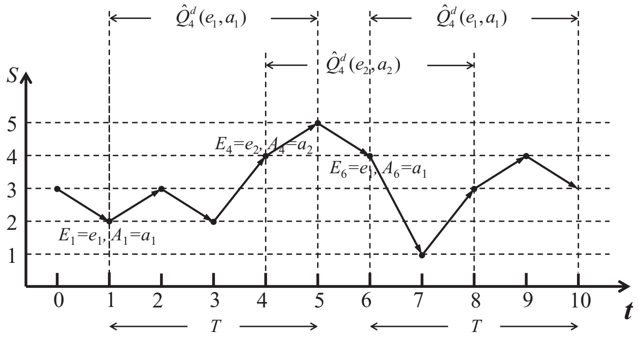

Fig. 1. Illustration of the online estimation of $Q(e, a)$'s with direct approach in Algorithm 1.

Fig. 1 illustrates the online estimation process of $Q(e, a)$ with the approach of Algorithm 1. Please note that we assume there are only two events e_1, e_2 and two actions a_1, a_2 in this EBO problem. The estimation length $T = 4$. During the period from time 0 to 10, there are only three event occurrences. That is, $E_1 = e_1$, $E_4 = e_2$, and $E_6 = e_1$. The events e_1 and e_2 are defined as $e_1 := \{\langle 2, 3\rangle, \langle 4, 1\rangle\}$ and $e_2 := \{\langle 4, 5\rangle\}$, respectively. From Fig. 1, we see that the number of event occurrences is much less than that of state transitions and the decision making in EBO is much rarer than that in MDP.

The above estimation algorithm is based directly on (18). Moreover, we can further develop a recursive learning estimation algorithm as follows. Based on Definitions (9) and (5), we can derive:

$$Q(e,a) = E\left\{\sum_{t=0}^{\tau-1}[r(X_t, A_t) - \eta] + \sum_{i\in\mathcal{S}} p^a(i|e) \sum_{j\in\mathcal{S}} p^{(\tau)}(j|i) \sum_{e'\in\mathcal{E}} p(e'|j) Q(e',a')\right\}\Bigg| E_0 = e, A_0 = a, \quad (19)$$

where τ is a random number indicating the time interval between two successive event occurrences, a' is the action generated by a predefined event-based policy for the next event, and $p^{(\tau)}(j|i)$ is the τ-step transition probability from i to j. This is a recursive formula of Q-factor in EBO. We can further develop the following recursive learning estimation algorithm for $Q^d(e, a)$, which is similar to the SARSA (state-action-reward-state-action) algorithm in reinforcement learning [24]. Define an update difference as:

$$\gamma_l := \sum_{t=t_l}^{t_{l+1}-1}[r(X_t, A_t) - \hat{\eta}] + \hat{Q}^d(E_{t_{l+1}}, A_{t_{l+1}}) - \hat{Q}^d(E_{t_l}, A_{t_l}), \quad (20)$$

where t_l is the occurrence time of the lth event, $\hat{\eta}$ is the current estimate of the long-run average performance and it is updated as

$$\hat{\eta} := \hat{\eta} + \frac{1}{t}[r(X_t, A_t) - \hat{\eta}]. \quad (21)$$

Then, the estimate $\hat{Q}(e, a)$ is updated as

$$\hat{Q}^d(E_{t_l}, A_{t_l}) := \hat{Q}^d(E_{t_l}, A_{t_l}) + \beta_l \gamma_l, \quad (22)$$

where β_l is the update step-size at time t_l and it can be $1/l$ or $1/l^2$ as an example. In order to visit all of the event-action pairs, we also have to adopt the aforementioned d_ε policy. Therefore, with (20), (21), and (22), we can directly have a recursive estimation algorithm for $Q^d(e, a)$ based on the sample path of EBO. Compared with the sequential estimation algorithm in Algorithm 1, the recursive algorithm is easy to implement since it needs less storage. The detailed description of the above recursive algorithm is omitted due to limited. Similar to Fig. 1, we use Fig. 2 to illustrate the recursive estimation process of $Q(e, a)$'s based on (22).

With the above discussion, we can estimate the Q-factor $Q^d(e, a)$ only based on the observation of a sample path under the current policy d, using either the sequential or recursive estimation algorithm. The distribution of $\pi^d(e)$ is easy to estimate with the observation of event sequences. Therefore, all of the terms in the derivative equation (15) are obtained based on the sample path. A typical gradient-based algorithm can be used as below. We choose a policy h from the policy space, which has a maximal value of derivative (15). The next policy can be updated as $d := d + \alpha(h - d)$, where $(h - d)$ can be understood as the steepest gradient direction in the policy space and α is an update step-size. Repeating the above procedure, the gradient-based algorithm can approach the local optimum of EBO. Other details of the gradient-based algorithm can be seen in [9].

Please note, this policy-gradient based approach does not require Condition (7) on the EBO problem. The underlying system state can be unobservable. We only have to observe the sequence of events, actions, and rewards to do online optimization. This provides a major advantage in applying the optimization framework of EBO to a wide range of practical problems. Below, we use an example of admission control to demonstrate the main idea of this paper.

IV. APPLICATION TO ADMISSION CONTROL

Admission control is an important problem existing in many service-oriented systems, such as call admission control in cellular networks [11] and traffic control in the Internet [20]. Queueing network is a widely-used model to formulate the admission control problem. In this section, we discuss the admission control in a semi-open Jackson network, which is studied in [27,28] using the policy-iteration type approach. As a comparison, in this paper, we use the policy gradient approach with Q-factors of EBO to explore this problem. We demonstrate that the online policy gradient approach of EBO is effective.

Consider a semi-open Jackson network with $M = 3$ servers [6,12]. Fig 3 is an illustrative example with 3 servers. The capacity of the network is $N = 6$. That is, when the total

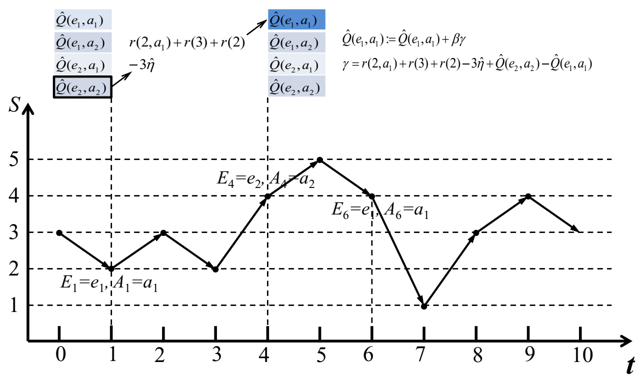

Fig. 2. Illustration of the online estimation of $Q(e, a)$ with recursive approach.

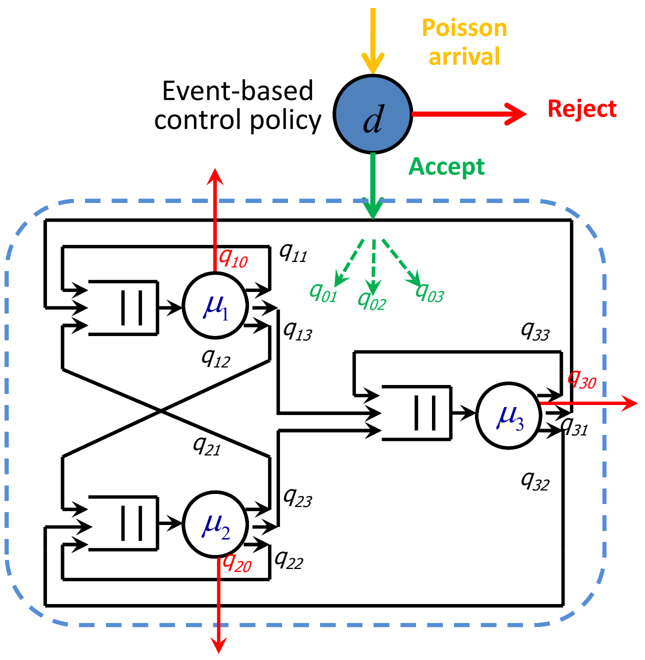

Fig. 3. Admission control of a semi-open Jackson network with 3 servers.

number of customers in the network reaches N, any newly arriving customer will be dropped directly (this is why we call it semi-open). The external customer arrival obeys a Poisson process with rate $\lambda = 50$. When an external customer arrives at the network, a decision maker controls the admission of the arriving customer. If the customer is rejected, it will leave the network without penalty. If it is admitted, the customer will enter server i with probability q_{0i}, $i = 1, 2, \dots, M$. When a customer finishes its service at server i, it will transit to server j with probability q_{ij} or leave the network with probability q_{i0}, $i, j = 1, 2, \dots, M$. The values of these routing probabilities are $q_{01} = 0.3$, $q_{02} = 0.4$, $q_{03} = 0.3$, $q_{10} = 0.2$, $q_{12} = 0.4$, $q_{13} = 0.4$, $q_{20} = 0.4$, $q_{21} = 0.3$, $q_{23} = 0.3$, $q_{30} = 0.3$, $q_{31} = 0.2$, $q_{32} = 0.5$, respectively. When a customer finishes its service at a server, it will receive a constant reward $R = 10$. Every customer sojourning in the network has to pay a cost $C = 100$ per unit time. The service time of servers obeys exponential distribution and the service rates are $\mu_1 = 40$, $\mu_2 = 50$, and $\mu_3 = 80$, respectively. The service discipline is first come first served. The number of customers at server i is denoted as n_i and the system state is denoted as $\boldsymbol{n} := (n_1, n_2, \dots, n_M)$. The state space is denoted as $\mathcal{S} := \{\text{all } n : \sum_{i=1}^{M} n_i \le N\}$.

The controller makes decision only when the event of customer arrival happens. The event is defined as $e_n := \left\{\langle \mathbf{n}, \mathbf{n}_{+i}\rangle, \langle \mathbf{n}, \mathbf{n}\rangle, \textit{with} \sum_{i=1}^{M} n_i = n\right\}$, where $\boldsymbol{n}_{+i} := (n_1, \dots, n_{i+1}, \dots, n_M)$ is called a neighboring state of $\boldsymbol{n}$. The definition of event e_n means that a customer arrives at the network and finds that the total number of customers in the network equals n, $n = 0, 1, \dots, N-1$. The possible action is either accepting or rejecting the arriving customer, denoted as $a = 1$ and $a = 0$, respectively. Suppose the system state $\boldsymbol{n}$ is unobservable for the controller. The controller observes only the sequences of events e, actions a, and rewards. We use $Q(e, a)$'s to evaluate the quality of action a when event e happens, where $e \in \mathcal{E} := \{e_0, e_1, \cdots, e_{N-1}\}$ and $a \in \mathcal{A} := \{0, 1\}$. In this paper, we consider a random admission policy $d := (b_0, b_1, \dots, b_{N-1})$. That is, when event e_n happens, we accept the arrival with probability b_n and reject it with probability $1 - b_n$, where $b_n \in \mathcal{R}[0,1]$ and $n = 0, 1, \dots, N-1$ ($n = N$ is omitted because any arrival at this situation is dropped directly due to the network capacity limit). We see that such a random policy usually explores all of the possible actions ($a = 1$ or $a = 0$). We do not have to use the randomization technique d_ε policy as mentioned before, except for the situation where $b_n = 0$ or $b_n = 1$. For these excepted situations, we will use d_ε policy with $\varepsilon = 0.01$.

We formulate this problem as an EBO problem and use the policy gradient approach to solve it. First, we use Algorithm 1 to estimate Q-factor $Q(e, a)$'s under the current policy d. By applying the derivative equation (15), we find a policy h with a maximal derivative. The next policy is updated as $d := d + \alpha(h - d)$, where the step-size α is set as $1/k$ with k being the index of the iterations. The parameter setting of the above procedure can be referred to other gradient-based approaches that are commonly used in practice. Here, we omit the details and focus on the online estimation of Q-factors and the effect of the policy gradient approach in this problem.

Table I. The mean and standard deviation of $Q(e_n, a)$ estimations when $a = 0$.

n	0	1	2	3	4	5
Mean	136.16	154.27	158.77	162.50	160.76	156.79
Std.	5.83	13.12	2.87	3.92	2.95	3.54

Table II. The mean and standard deviation of $Q(e_n, a)$ estimations when $a = 1$.

n	0	1	2	3	4	5
Mean	153.59	159.39	163.79	163.46	160.40	156.01
Std.	3.87	3.28	2.35	2.73	3.27	3.37

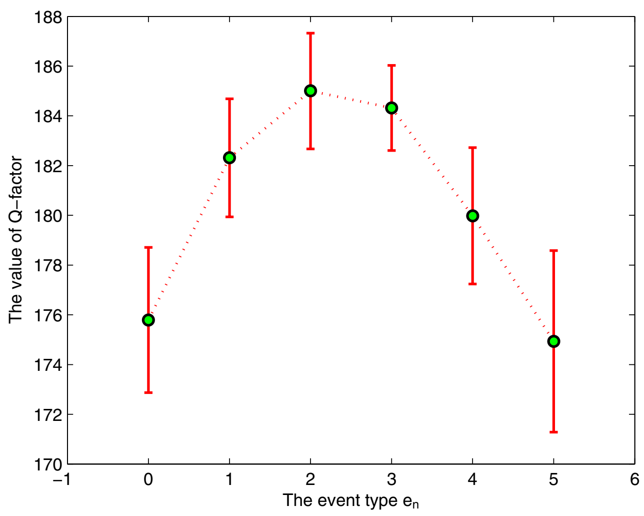

Fig. 4. A typical example of the mean and standard deviation curve of $Q(e_n, a)$ estimations.

The estimation length T of $Q(e, a)$ in Algorithm 1 is set as $T = 220$. The initial admission policy is chosen arbitrarily as $d_0 = (0.5, 0.8, 0.3, 0.6, 0.2, 0.4)$. Under the current policy, we run the simulation for $L = 220{,}000$ and replicate it 10 times. Some typical examples of the mean and standard deviation of estimations are illustrated in Tables I and II and in Fig. 4. We use the mean estimates of $Q(e, a)$'s and execute the above optimization process to obtain an updated policy. Run the system under the new policy and repeat this process. Finally, we find that the output policy is almost stable after 20 iterations. The optimization process is illustrated by the curves in Fig. 5. We see that the output policy is approaching $d^* = (1, 1, 1, 1, 0, 0)$, which is truly the optimal policy of this

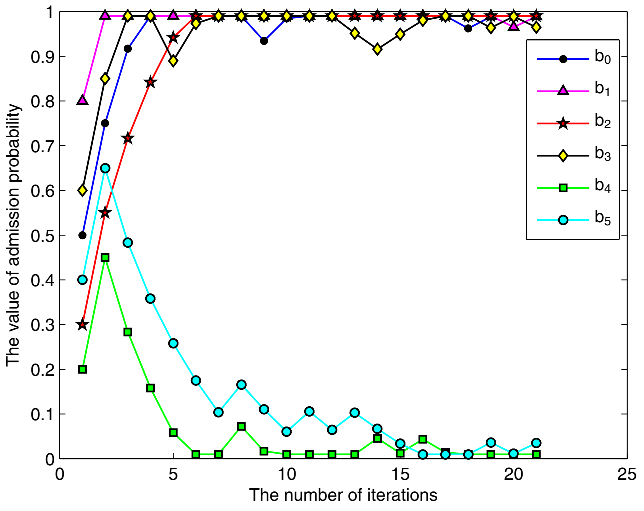

Fig. 5. The curve of admission probabilities during the evolution of policy gradient optimization.

admission control problem [27]. This policy has a threshold form, *i.e.*, when $n < 4$, all the arrivals are accepted; otherwise, all are rejected. It has been proven that we can always find the optimal policy from the threshold type policies [27,30]. From this example, we see that the policy gradient approach with Q-factors is effective for the EBO problems. We can estimate online the Q-factors of EBO and use the gradient-based approach to find the local optimum of EBO. This is an effective alternative approach of EBO when the policy iteration type approach presented in [4,5,29] is not applicable.

V. CONCLUSION

In this paper, we define the Q-factor $Q(e, a)$ of EBO to quantify the quality of action selections when the system state is unobservable. With the Q-factor of EBO, we derive the performance derivative equation of EBO. Therefore, when the policy iteration approach of EBO is inapplicable, we can use the policy gradient approach alternatively. We also propose some online estimation methods for the Q-factors in EBO. This makes the policy gradient approach of EBO online implementable. As a complementary approach to policy iteration of EBO, the online policy gradient approach makes the EBO optimization framework applicable to a wider range, without requiring restrictive optimality conditions. Future efforts may be devoted to the research of other algorithms with Q-factors in EBO, such as the analogs of the algorithms in reinforcement learning.

REFERENCES

1. Baxter, J. and P. Bartlett, "Infinite-horizon policy-gradient estimation," *J. Artif. Intell. Res.*, Vol. 15, pp. 319–350 (2001).
2. Bertsekas, D. P. and J. N. Tsitsiklis, *Neuro-Dynamic Programming*, Athena Scientific, Belmont, MA (1996).
3. Cao, X. R. and H. F. Chen, "Potentials, perturbation realization, and sensitivity analysis of Markov processes," *IEEE Trans. Autom. Control*, Vol. 42, pp. 1382–1393 (1997).
4. Cao, X. R., "Basic ideas for event-based optimization of Markov systems," *Discret. Event Dyn. Syst.-Theory Appl.*, Vol. 15, pp. 169–197 (2005).
5. Cao, X. R., Stochastic Learning and Optimization – A Sensitivity-Based Approach, Springer, New York (2007).
6. Chen, H. and D. D. Yao, *Fundamentals of Queueing Networks, Performance, Asymptotics, and Optimization*, Springer, New York (2001).
7. Chen, X., H. Bai, L. Xia, and Y. C. Ho, "Design of a sensor network for target detection," *Automatica*, Vol. 43, pp. 1713–1722 (2007).
8. Cheng, Z., L. Xia, Q. Zhao, Y. Zhao, F. Wang, and F. Song, "Integrated control of blind and lights in daily office environment," *Porceedings 2013 IEEE Conference Automation Scinience Eng.*, Madison, WI, USA (2013).
9. Chong, E. K. P. and S. Zak, *An Introduction to Optimization, Fourth Edition*, John Wiley & Sons, New York (2013).
10. Fang, H. T., H. F Chen, and X. R. Cao, "Recursive approaches for single sample path based Markov reward processes," *Asian J. Control*, Vol. 3, pp. 21–26 (2001).
11. Ghaderi, M. and R. Boutaba, "Call admission control in mobile cellular networks: a comprehensive survey," *Wirel. Commun. Mob. Comput.*, Vol. 6, pp. 69–93 (2006).
12. Gross, D., J. F. Shortle, J. M. Thompson, and C. M. Harris, *Fundamentals of Queueing Theory*, 4th Edition, Wiley, Hoboken (2008).
13. Guo, X. and O. Hernandez-Lerma, *Continuous-Time Markov Decision Processes: Theory and Applications*, Springer (2009).
14. Ho, Y. C. and X. R. Cao, *Perturbation Analysis of Discrete Event Systems*, Kluwer Academic Publishers, Norwell (1991).
15. Hwang, K. S., Y. J. Chen, W. C. Jiang, and T. F. Lin, "Continuous action generation of Q-learning in multi-agent cooperation," *Asian J. Control*, Vol. 15, No. 4, pp. 1011–1020 (2013).
16. Jia, Q. S., "On solving optimal policies for finite-stage event-based optimization," *IEEE Trans. Autom. Control*, Vol. 56, pp. 2195–2200 (2011).
17. Jia, Q. S., "On solving event-based optimization with average reward over infinite stages," *IEEE Trans. Autom. Control*, Vol. 56, pp. 2912–2917 (2011).
18. Kaelbling, L. P., M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *J. Artif. Intell. Res.*, Vol. 4, pp. 237–285 (1996).

19. Leahu, H., B. Heidergott, and A. Hordijk, “Perturbation analysis of waiting times in the G/G/1 queue,” *Discret. Event Dyn. Syst.-Theory Appl.*, Vol. 23, pp. 277–305 (2013).
20. Lima, S. R., P. Carvalho, and V. Freitas, “Admission control in multiservice IP networks: architectural issues and trends,” *IEEE Commun. Mag.*, Vol. 45, pp. 114–121 (2007).
21. Marbach, P. and J. N. Tsitsiklis, “Simulation-based optimization of Markov reward processes,” *IEEE Trans. Autom. Control*, Vol. 46, pp. 191–209 (2001).
22. Puterman, M. L., *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, John Wiley & Sons, New York (1994).
23. Sun, B., P. B. Luh, Q. S. Jia, Z. Jiang, F. Wang, and C. Song, “An integrated control of shading blinds, natural ventilation, and HVAC systems for energy saving and human comfort,” *Porceedings 2010 IEEE Conference Automation Scinience Eng.*, pp. 7–14 (2010).
24. Sutton, R. S. and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA (1998).
25. Sutton, R. S., D. A. McAllester, S. P. Singh, and Y. Mansour, “Policy gradient methods for reinforcement learning with function approximation,” *Adv. Neural Inform. Process. Syst.*, Vol. 12, pp. 1057–1063 (2000).
26. Wang, D. X. and X. R. Cao, “Event-based optimization for POMDP and its application in portfolio management,” *Proceedings 18th IFAC World Congr.*, Milano, Italy, pp. 3228–3233 (2011).
27. Xia, L., “Optimal control of customer admission to an open Jackson network,” *Proceedings 31st Chinese Control Conference*, Hefei, China, pp. 2362–2367 (2012).
28. Xia, L., “Event-based optimization of admission control in open queueing networks,” *Discret. Event Dyn. Syst.-Theory Appl.*, DOI 10.1007/s10626-013-0167-1 (2013).
29. Xia, L., Q. S. Jia, and X. R. Cao, “A Tutorial on Event-based optimization,—A new optimization framework,” *Discret. Event Dyn. Syst.-Theory Appl.*, DOI 10.1007/s10626-013-0170-6 (2013).
30. Xia, L. and B. Shihada, “Max-Min optimality of service rate control in closed queueing networks,” *IEEE Trans. Autom. Control*, Vol. 58, pp. 1051–1056 (2013).
31. Xu, Y. and X. Chen, “Optimization of dependently controlled jump rates of JLQG problem,” *Asian J. Control*, Vol. 11, pp. 432–439 (2009).
32. Yao, C. and C. G. Cassandras, “Using infinitesimal perturbation analysis of stochastic flow models to recover performance sensitivity estimates of discrete event systems,” *Discret. Event Dyn. Syst.-Theory Appl.*, Vol. 22, pp. 197–219 (2012).

Li Xia received the B.E. degree in automation in July 2002, and the Ph.D. degree in control science and engineering in July 2007, both from Tsinghua University, Beijing, China. From 2007 to 2009, he was a staff research member in IBM China Research. From 2009 to 2011, he was a postdoctoral research fellow in the King Abdullah University of Science and Technology (KAUST), Saudi Arabia. After 2011, he joined Tsinghua University and currently he is an associate professor with the Center for Intelligent and Networked Systems (CFINS), Department of Automation, Tsinghua University.

He was a visiting scholar in the Hong Kong University of Science and Technology and a visiting scholar in University of Waterloo, Canada. He is serving or has served as a reviewer of international journals including IEEE Transactions on Automatic Control, IEEE Transactions on Automation Science and Engineering, European Journal of Operational Research, IIE Transactions, Journal of Optimization Theory and Applications, etc. He is an IEEE member and a member of Operation Research Society of China. His research interests include the methodology research in stochastic analysis and optimization, queueing theory, discrete event systems, and the application research in building energy, scheduling and optimization of smart grid, wireless sensor networks, production systems, etc.