2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Optimal Control of Discrete-Time Multivariate Point Processes with Finite-Time Steady State

A. Agung Julius, Yunshi Wen, and Ruixuan Yan

Abstract— **Multivariate point processes (MPP) are widely used to model the occurrence of multiple interrelated events in complex systems. They are used in a variety of fields to analyze data and define models that can make predictions about future events. In this paper, we consider finite-state MPP, which are products of finite state automata (FSA) and MPP. Specifically, the events in the MPP trigger state transitions in the FSA, while the intensities of the point processes are defined as functions of the FSA state and the history of the MPP. Further, we assume that some of the event types are controllable, i.e., they are not random but can be triggered. We formulate an optimal control problem for such system, which can then be expressed as optimal control for a Markov Decision Process (MDP) with infinite states.**

When the system has appropriate finite-time steady state properties, we use the concept of stochastic bisimulation of MDP to reduce the MDP into a finite state one, thereby allowing us to use standard optimal control techniques to calculate the optimal policy. We demonstrate the effectiveness of our method on a simplified sleep-wake cycle model, for the problem of optimally scheduling naps to maximize the length of wakefulness intervals.

I. Introduction

In recent years, multivariate point processes (MPPs) [1] have emerged as a framework for representing multivariate event data in continuous time. MPPs provide a mathematical structure for modeling the occurrence of events, where each event is assigned conditional intensity rate that quantifies its frequency of occurrence at any given time, taking into account the prior occurrence of other event labels [1], [2]. MPPs have gained significant popularity across several domains, including finance, social networks, and healthcare. In finance, events may represent buying or selling of stocks [3], while in social networks, they may represent user activity such as posts, likes, shares, or subscriptions [4]. In biology, events can represent falling asleep or waking up [5].

Existing research on MPPs can be divided into two categories, MPP modeling (MPPM) and MPP control (MPPC) [6]. MPPM is concerned with training machine learning models for analyzing and understanding the generation mechanisms of event streams, which is particularly useful in predicting the occurrence of events or understanding the cause of deleterious events' occurrence to prevent their occurrence. Some earlier works for MPPM make simplifying assumptions due to the difficulty of modeling the conditional intensity rate, such as Poisson networks [7], models assuming piece-wise constant intensity rates, models assuming the generation process as a non-homogeneous Poisson process [8], and multivariate Hawkes processes [9]. More recently, generalized algorithms for MPPM have been proposed as well, such as representing MPPs as graphical event models (GEMs) [10]–[12].

While the problem of MPPM has been extensively researched, the synthesis of control policies for MPPs has not received much attention. In the control domain, [13] formulate the problem of optimally triggering an action in a mean field games setting. However, only univariate event (i.e., one event type) is considered. The state-of-the-art approaches for MPPC are mainly accomplished by stochastic optimal control approaches with the assumption that the dynamics of MPPs are modeled as stochastic differential equations (SDEs) [14], [15]. Nevertheless, the drawbacks in the following aspects prevent their application in a broader range. Modeling the intensity rates as particular differential equations may limit their application when the dynamics do not follow an ideal functional form. Also, the objective functions for the stochastic optimal control problem need to be carefully designed to guarantee the feasibility of the proposed problem. Recent advancement in overcoming the above limitations by using reinforcement learning approaches has been proposed in [16], with the goal of finding the optimal policy for event generation using a policy gradient method. However, the policy gradient approach lacks global optimality guarantee.

This paper aims to tackle the issue of intervening the occurrence of events by appropriately triggering some controllable events, so as to maximize a given objective (e.g., delaying the occurrence of some events). Our contributions in this paper are in (i) formulating a generalized framework for MPP, called finite state MPP, which are products of finite state automata and MPP, (ii) formulating the above problem in the Markov Decision Process (MDP) framework, in which we seek to find the optimal policy for triggering these events, (iii) showing that, under some finite-time steady state properties, the MDP can be reduced to an equivalent finite-state MDP using the concept of stochastic bisimulation, and (iv) demonstrating that standard optimal control algorithms for finite state MDP, which yield globally optimal policies, can be effectively used on the reduced model in an example that is based on a simplified model of the sleep-wake process.

Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, New York 12180. Email: {juliua2,weny2,yanr5}@rpi.edu

979-8-3503-0123-6/23/$31.00 ©2023 IEEE

II. Mathematical Preliminaries and Problem Formulation

A. Multivariate Point Processes (MPP)

Consider N events modeled as multivariate point processes (MPP). We also have N clocks, $\{c_i\}_{i=1,\cdots,N}$, associated with these events. Assume that time is discretized. Thus, for each $i \in \{1,\cdots,N\}$, $c_i : \mathbb{N} \to \mathbb{N}$, where $c_i(k)$ is the length of the time interval between timestep k and the last time Event i happened **before** timestep k. For all $i \in \{1,\cdots,N\}$ and $k \in \mathbb{N}$, we define the binary-valued random variable $h_i(k)$ as

$$h_i(k) = \begin{cases} 1, & \text{if Event } i \text{ happens at timestep } k, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Therefore, $c_i(k)$ is also a random variable that evolves over time according to

$$c_i(k+1) = \begin{cases} c_i(k)+1, & \text{if } h_i(k) = 0, \\ 1, & \text{if } h_i(k) = 1. \end{cases} \tag{2}$$

We assume that for all timestep k, any random variables $h_i(k)$ and $h_j(k)$, where $i \neq j$, $i,j \in \{1,\cdots,N\}$, are independent. The dynamics of the clock variables is illustrated in Fig. 1.

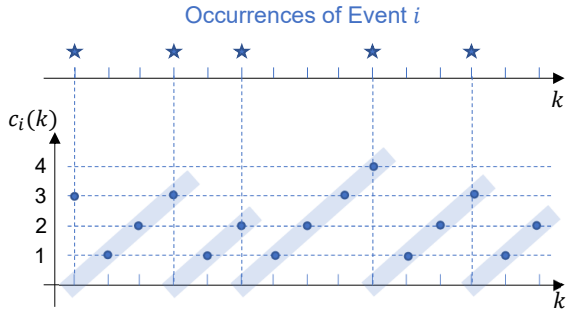

Fig. 1. Illustration of the dynamics of the clock variable of Event i. (Top) An event stream showing the occurrences of Event i. (Bottom) The value of the clock $c_i(k)$ corresponding to the event stream.

We assume that there are M other events, whose occurrences can be controlled. We refer to these events as Controlled Events, or C-Events in short. To these events, we associate the clocks $\{v_i\}_{i=1,\cdots,M}$, where $v_i(k)$ is the length of the time interval between timestep k and the last time the C-Event i happened **before** timestep k. For ease of discussion later, for all $i \in \{1,\cdots,M\}$ and $k \in \mathbb{N}$, we also introduce the binary-valued variable $u_i(k)$ as

$$u_i(k) = \begin{cases} 1, & \text{if C-Event } i \text{ happens at timestep } k, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Analogous to Eq. (2), we define the evolution of the controlled event clocks as

$$v_i(k+1) = \begin{cases} v_i(k)+1, & \text{if } u_i(k) = 0, \\ 1, & \text{if } u_i(k) = 1. \end{cases} \tag{4}$$

Note that neither $v_i(k)$ nor $u_i(k)$ are random; they are control variables.

Notation 1: The set of all events is denoted as

$$\text{Ev} \triangleq \{\text{Event 1, } \cdots, \text{Event } N, \text{ C-Event 1}, \cdots, \text{ C-Event M}\}. \tag{5}$$

Notation 2: The vector of all clock values is denoted as $x(k) \in \mathbb{N}^{N+M}$, where

$$\begin{aligned} x(k) &\triangleq [c(k), v(k)] \\ &= \left[\underbrace{c_1(k),\cdots,c_N(k)}_{c(k)}, \underbrace{v_1(k),\cdots,v_M(k)}_{v(k)}\right]. \end{aligned} \tag{6}$$

The vector of all $u_i(k)$ is denoted as

$$u(k) \triangleq [u_1(k),\cdots,u_M(k)] \in \{0,1\}^M.$$

Each event of the point processes is assumed to depend on the point processes and the controlled events. Specifically, let $p_i(k)$ denote the probability that Event i occurs at timestep k, we assume that for all $i \in \{1,\cdots,N\}$,

$$p_i(k) = F_i(x(k)) = F_i(c(k), v(k)). \tag{7}$$

Here, F_i is a known function that maps the clock values to the interval $[0,1]$.

Next, we discuss a finite-time steady state property of the transition probabilities. However, before that, we introduce the following notation that will simplify the discussion.

Notation 3: Given a vector $x \in \mathbb{N}^n$, where $x = [x_1, x_2, \cdots, x_i, \cdots, x_n]$, the operation $\texttt{Sat}_i(x, Z)$ replaces the i-th element of x with $Z \in \mathbb{N}$. That is,

$$\texttt{Sat}_i(x, Z) = [x_1, x_2, \cdots, x_{i-1}, Z, x_{i+1}, \cdots, x_n]. \tag{8}$$

Definition 1 (finite-time steady state): An Event i of a multivariate point process described above is said to have finite-time steady state if there exists a horizon $H_i \in \mathbb{N}$, such that for any positive integers $c \triangleq \{c_j\}_{j\in\{1,\cdots,N\}}$ and $v \triangleq \{v_j\}_{j\in\{1,\cdots,M\}}$, if $c_i \geq H_i$ then

$$F_k(c, v) = F_k(\texttt{Sat}_i(c, H_i), v),$$

for all $k \in \{1,\cdots,N\}$. The same definition also applies to controlled events. The multivariate point process itself is said to have finite-time steady state is all of its $N+M$ events have finite-time steady state.

Intuitively, an Event i having finite-time steady state means if it has not happened longer than H_i timesteps, its influence on the events in the MPP (including Event i itself) does not change with any additional timestep of Event i not happening.

B. Generalization: Finite State MPP

The formulation described in the previous subsection can be generalized by adding a finite state automaton to the MPP, forming a Finite State MPP. First, we state the standard definition of finite state automata.

Definition 2: A finite state automaton (FSA) is a tuple $\mathfrak{A} = (S, \Sigma, \tau, s_0)$, where S is the (finite) set of states, Σ is the input alphabet, $\tau : S \times \Sigma \to S$ is the deterministic transition function, where $s' = \tau(s, \sigma)$ means the state would switch from $s \in S$ to $s' \in S$ upon the arrival of the input

symbol $\sigma \in \Sigma$. The symbol $s_0 \in S$ denotes the initial state, where the FSA is assumed to start.

A Finite State MPP is setup by taking the product of an FSA $\mathfrak{A} = (S, \Sigma, \tau, s_0)$ with an MPP, where

- the state of the FSA evolves according to the set of events that happens, i.e., the input alphabet $\Sigma = 2^{\mathrm{Ev}}$,
- the probabilities of the occurrence of the events are determined by the clocks and the state of the FSA.

Thus, the probability that Event i occurs at timestep k is given by

$$p_i(k) = \hat{F}_i(c(k), v(k), s), \tag{9}$$

$i \in \{1, \cdots, N\}$, where $\hat{F}_i$ is a known function that maps $(c(k), v(k), s)$ to the interval $[0, 1]$. Note that $\hat{F}_i$ generalizes F_i in Eq. (7) by allowing $p_i(k)$ to depend not only on the clock variables but also on the state of the FSA.

The concept of finite-time steady state that we discussed in the previous section can be generalized to Finite State MPP in the following sense.

Definition 3: An Event i of a Finite State MPP is said to have finite-time steady state if there exists $H_i \in \mathbb{N}$, such that for any state $s \in S$, positive integers $c \triangleq \{c_j\}_{j\in\{1,\cdots,N\}}$, and $v \triangleq \{v_j\}_{j\in\{1,\cdots,M\}}$, if $c_i \geq H_i$ then

$$\hat{F}_k(c, v, s) = \hat{F}_k(\mathtt{Sat}_i(c, H_i), v, s),$$

for all $k \in \{1, \cdots, N\}$. As before, we also apply this definition on the controlled events. The Finite State MPP itself is said to have finite-time steady state is all of its $N+M$ events have finite-time steady state.

We note that an MPP is a special case of Finite State MPP. I.e., an MPP is a Finite State MPP with only one state. In the rest of the paper, we will formulate our work for Finite State MPP.

C. Finite State MPP as Markov Decision Process

Definition 4 (Markov Decision Process): A Markov Decision Process (MDP) is defined as a tuple $\mathfrak{M} = (X, A, R, T)$, where X is the state space, A is the action space, $R : X \times A \times X \to \mathbb{R}$ is the reward function, and $T : X \times A \times X \to [0, 1]$ is the transition probabilities. We refer the reader to the standard literature (e.g., [17], [18]) for the semantics of MDP.

The Finite State MPP can be formulated as an MDP $\mathfrak{M}_P = (\mathbb{N}^{N+M} \times S, \{0, 1\}^M, R, T)$. The state space is the product of the respective state spaces of the MPP and FSA, where the state is given by

$$(x(k), s(k)) = (c(k), v(k), s(k)).$$

The action space is $\{0, 1\}^M$ and action is $u(k)$ of the MPP. The transition probabilities T of the MDP is defined as follows. For any $(\bar{X}, \bar{S})$ and $(\bar{X}', \bar{S}') \in \mathbb{N}^{N+M} \times S$, where $\bar{X} = [C, V]$, $\bar{X}' = [C', V'] \in \mathbb{N}^{N+M}$ and $U \in \{0, 1\}^M$,

$$\begin{aligned} &T((\bar{X}, \bar{S}), U, (\bar{X}', \bar{S}')) \\ &\triangleq \Pr\left\{x(k+1) = \bar{X}', s(k+1) = \bar{S}' \middle| \right. \\ &\left. x(k) = \bar{X}, s(k) = \bar{S}, u(k) = U\right\} \qquad (10) \\ &= \mathcal{S}(C_i, U, C_i', \bar{S}, \bar{S}') \prod_{i=1}^{N} \mathcal{T}_i(\bar{S}, \bar{X}, U, \bar{X}') \prod_{i=1}^{M} \mathcal{W}_i(\bar{X}, U, \bar{X}'), \qquad (11) \end{aligned}$$

where

$$\mathcal{T}_i(\bar{S}, \bar{X}, U, \bar{X}') = \begin{cases} \hat{F}_i(\bar{S}, \bar{X}), & \text{if } C_i' = 1, \\ 1 - \hat{F}_i(\bar{S}, \bar{X}), & \text{if } C_i' = C_i + 1, \\ 0, & \text{else,} \end{cases} \tag{12}$$

$$\mathcal{W}_i(\bar{X}, U, \bar{X}') = \begin{cases} 1, & \text{if } V_i' = 1 \text{ and } U_i = 1, \\ 1, & \text{if } V_i' = V_i + 1 \text{ and } U_i = 0, \\ 0, & \text{else,} \end{cases} \tag{13}$$

and

$$\mathcal{S}(C_i, U, C_i', \bar{S}, \bar{S}') = \begin{cases} 1, & \text{if } \tau(\bar{S}, \mathrm{E}) = \bar{S}', \\ 0, & \text{else.} \end{cases} \tag{14}$$

Here, the set of events E are all events that happen in the time step of the transition. That is, we define the indices

$$I_u \triangleq \{i \mid C_i' = 1\},\ I_c \triangleq \{i \mid U_i = 1\},$$

and

$$\mathrm{E} = \bigcup_{i\in I_u} \{\text{Event } i\} \cup \bigcup_{i\in I_c} \{\text{C-Event } i\}.$$

Notation 4: To express the state-action sequence and reward sequence, we use the notations

$$\xi \triangleq (x(0), s(0)), u(0), (x(1), s(1)), u(1), \cdots, \tag{15}$$

$$\rho_\xi(k) \triangleq R((x(k), s(k)), u(k), ((x(k), s(k))). \tag{16}$$

The reward function of the MDP will be specified later, depending on the desired control objective. However, here we present a finite-time steady state property of the reward function.

Definition 5 (finite-time steady state reward): The MDP for Finite State MPP is said to have finite-time steady state reward if

- For all Event i, $i \in \{1, \cdots, N\}$ there exists a horizon $H_i \in \mathbb{N}$, such that for any state s and s' in S, and any positive integers $c \triangleq \{c_j\}_{j\in\{1,\cdots,N\}}$, $c' \triangleq \{c_j'\}_{j\in\{1,\cdots,N\}}$, $v \triangleq \{v_j\}_{j\in\{1,\cdots,M\}}$, and $v' \triangleq \{v_j'\}_{j\in\{1,\cdots,M\}}$, and any action u, if $c_i \geq H_i$ then

$$\begin{aligned} &R\left((c, v, s), u, (c', v', s')\right) \\ &= R\left((\mathtt{Sat}_i(c, H_i), v, s), u, (c', v', s')\right), \\ &R\left((c', v', s'), u, (c, v, s)\right) \\ &= R\left((c', v', s'), u, (\mathtt{Sat}_i(c, H_i), v, s)\right). \end{aligned}$$

- For all C-Event i, $i \in \{1, \cdots, M\}$ there exists a horizon $K_i \in \mathbb{N}$, such that for any state s and s' in S, and any positive integers $c \triangleq \{c_j\}_{j\in\{1,\cdots,N\}}$,

$c' \triangleq \{c'_j\}_{j\in\{1,\cdots,N\}}$, $v \triangleq \{v_j\}_{j\in\{1,\cdots,M\}}$, and $v' \triangleq \{v'_j\}_{j\in\{1,\cdots,M\}}$, and any action u, if $v_i \geq K_i$ then

$$\begin{aligned}&R\left((c,v,s),u,(c',v',s')\right)\\&= R\left(c,(\texttt{Sat}_i(v,K_i),s),u,(c',v',s')\right),\\&R\left((c',v',s'),u,(c,v,s)\right)\\&= R\left((c',v',s'),u,(c,\texttt{Sat}_i(v,K_i),s)\right).\end{aligned}$$

Intuitively, this property means if an Event or C-Event has not occurred within its horizon length, the reward value does not change with any additional timestep that Event or C-Event not occurring.

D. Problem Formulation

To define the problem, we use the standard definition of policies. A policy for an MDP $\mathfrak{M} = (X, A, R, T)$ is a function $\pi : X \to A$.

Problem 1: Given an MDP $\mathfrak{M} = (X, A, R, T)$, find the policy π that maximizes the discounted accumulated reward

$$J(\pi, X_0) = \sum_{k=0}^{\infty} E\left[\rho_\xi(k)\right]\gamma^k, \tag{17}$$

where $\gamma \in (0, 1)$ is the discount factor and the probability distribution of ξ is defined such that

$$x(0) = X_0, \tag{18}$$

$$u(k) = \pi(x(k)), \forall k \in \{0, 1, \cdots\}. \tag{19}$$

Assumption: In this paper, we consider MDP that are defined based on Finite State MPP with finite-time steady state properties described in Definitions 3 and 5.

Remark 1: In this paper, we assume that policies are deterministic. That is, $\pi(\cdot)$ returns a single action. More generally, policies of MDP can be stochastic, where $\pi(\cdot)$ returns a probability distribution over the set of actions A. However, it can be proven (e.g., see [18]) that for the objective function J in Eq. (17), the optimizing policy π can be assumed to be deterministic without any loss of performance.

Remark 2: In the problem definition, we define the objective function J, and therefore also the optimal policy, to be dependent on the initial state X_0. However, because of the principle of optimality (e.g., see [18]), it is well known that there exists an optimal policy π^* that maximizes J for any initial state $X_0 \in X$.

III. Stochastic Bisimulation to Reduce MDP

The problem defined in the previous section is the classical optimal control for MDP, for which there are known iterative algorithms that yield the exact optimal solutions in the finite models case (e.g., see [17], [18]). Here, finite models mean those with finitely many states and actions. Such algorithms include model-based ones (e.g., policy iteration and value iteration (e.g., see [17], [18]), which yield the optimal policy in finitely many steps) and simulation-based ones (e.g., tabular Reinforcement Learning (e.g., see [19]). For the cases where the number of states is infinite, such as the setup that we presented in the previous section, typically the approach is to have a finite parameterization of the policy and perform optimization on the finitely many parameters. In general, the optimization is gradient-based and therefore there is no guarantee on global optimality, unless other properties such as convexity are assumed.

One concept that allows for exact calculation of the optimal policy for infinite states MDP is *factored MDP* [20], [21]. The main idea is to construct a finite state MDP that is, in some sense, equivalent to the infinite state one and calculate the optimal policy on the reduced MDP. Formally, this is achieved using the concept of *stochastic bisimulation*, which is defined below. But first, we overload the notation of transition probability as follows.

Notation 5: Given an MDP $\mathfrak{M} = (X, A, R, T)$. For any countable set of states $\mathcal{X} \subset X$, state $x \in X$, and action $u \in A$, we define

$$T(x, u, \mathcal{X}) \triangleq \sum_{x'\in\mathcal{X}} T(x, u, x'). \tag{20}$$

Next, we recall that for a given set X, a relation $\mathcal{R} \subset X \times X$ is called an equivalence relation if it is

1) Reflexive: For all $x \in X$, $(x, x) \in \mathcal{R}$.
2) Symmetric: For all $x, x' \in X$, if $(x, x') \in \mathcal{R}$ then $(x', x) \in \mathcal{R}$.
3) Transitive: For all $x, x', x'' \in X$, if $(x, x') \in \mathcal{R}$ and $(x', x'') \in \mathcal{R}$ then $(x, x'') \in \mathcal{R}$.

Notation 6: For a given set X and an equivalence relation $\mathcal{R} \subset X \times X$, we define

$$\mathcal{R}(x) \triangleq \{x' \in X \mid (x, x') \in \mathcal{R}\}. \tag{21}$$

Definition 6 (Stochastic Bisimulation [20]): Given an MDP $\mathfrak{M} = (X, A, R, T)$, the equivalence relation $\mathcal{R} \subset X \times X$ is a stochastic bisimulation if the following are true.

1) Reward equivalence: For all $x, x', x'' \in X$ and $u \in A$, if $(x, x') \in \mathcal{R}$, then

$$R(x'', u, x) = R(x'', u, x'), \tag{22}$$

$$R(x, u, x'') = R(x', u, x''). \tag{23}$$

2) Transition equivalence: For all $x, x', x'' \in X$ and $u \in A$, if $(x, x') \in \mathcal{R}$, then

$$T(x, u, \mathcal{R}(x'')) = T(x', u, \mathcal{R}(x'')). \tag{24}$$

Since a stochastic bisimulation $\mathcal{R}$ is an equivalence, it induces a partition of X, the states of the MDP.

Notation 7: We denote the partition of X induced by $\mathcal{R}$ as $\Gamma_\mathcal{R}(X)$. The partition function $\gamma_\mathcal{R} : X \to \Gamma_\mathcal{R}(X)$ is then defined such that for all $x, x' \in X$: $(x, x') \in \mathcal{R}$ if and only if

$$\gamma_\mathcal{R}(x) = \gamma_\mathcal{R}(x'). \tag{25}$$

The partition induced by $\mathcal{R}$ is then used to define the factored MDP.

Definition 7 (Factored MDP [20]): Given an MDP $\mathfrak{M} = (X, A, R, T)$ and a stochastic bisimulation $\mathcal{R}$, the factored MDP generated by $\mathcal{R}$ is $\mathfrak{M}_\mathcal{R} = (\Gamma_\mathcal{R}(X), A, R_\mathcal{R}, T_\mathcal{R})$, where for all $\chi, \chi' \in \Gamma_\mathcal{R}(X)$ and $u \in A$,

$$R_\mathcal{R}(\chi, u, \chi') = R(x, u, x'), \tag{26a}$$

$$T_\mathcal{R}(\chi, u, \chi') = T(x, u, x'), \tag{26b}$$

where

$$\gamma_{\mathcal{R}}(x) = \chi, \gamma_{\mathcal{R}}(x') = \chi'. \tag{27}$$

Note that reward equivalence and transition equivalence properties of $\mathcal{R}$ ensure that any x, x' that satisfy Eq. (27) also satisfy Eq. (26).

Given a state-action sequence ξ of $\mathfrak{M}$ as in Eq. (15), we slightly abuse the notation of $\gamma_{\mathcal{R}}$ by defining

$$\gamma_{\mathcal{R}}(\xi) \triangleq \gamma_{\mathcal{R}}\left(x(0)\right), u(0), \gamma_{\mathcal{R}}\left(x(1)\right), u(1), \cdots, \tag{28}$$

$$\rho_{\gamma_{\mathcal{R}}(\xi)}(k) \triangleq R(\gamma_{\mathcal{R}}\left(x(k)\right), u(k), \gamma_{\mathcal{R}}\left(x(k+1)\right)). \tag{29}$$

Then, the optimal control problem for the factored MDP can be defined as follows.

Problem 2: Given a factored MDP $\mathfrak{M}_{\mathcal{R}} = (\Gamma_{\mathcal{R}}(X), A, R_{\mathcal{R}}, T_{\mathcal{R}})$, find the policy $\pi_{\mathcal{R}} : \Gamma_{\mathcal{R}}(X) \to A$ that maximizes the discounted accumulated reward

$$J_{\mathcal{R}}(\pi, \chi_0) \triangleq \sum_{k=0}^{\infty} E\left[\rho_{\gamma_{\mathcal{R}}(\xi)}(k)\right] \gamma^k, \tag{30}$$

where $\gamma \in (0,1)$ is the discount factor and the probability distribution of $\gamma_{\mathcal{R}}(\xi)$ is defined such that

$$\gamma_{\mathcal{R}}\left(x(0)\right) = \chi_0, \tag{31}$$

$$u(k) = \pi_{\mathcal{R}}(\gamma_{\mathcal{R}}\left(x(k)\right)), \forall k \in \{0, 1, \cdots\}. \tag{32}$$

The following theorem is central in solving the optimal control problem for the original MDP (i.e., Problem 1) by way of solving the optimal control problem of the factored MDP (i.e., Problem 2).

Lemma 1: (from [20]) Given a factored MDP $\mathfrak{M}_{\mathcal{R}} = (\Gamma_{\mathcal{R}}(X), A, R_{\mathcal{R}}, T_{\mathcal{R}})$, if a policy $\pi_{\mathcal{R}} : \Gamma_{\mathcal{R}}(X) \to A$ maximizes the objective $J_{\mathcal{R}}$ in Problem 2 then its implementation of the original MDP, defined as $\pi : X \to A$,

$$\pi \triangleq \pi_{\mathcal{R}} \circ \gamma_{\mathcal{R}}, \tag{33}$$

maximizes the objective J in Problem 1. Here $\circ$ denotes function composition.

IV. MDP Factorization of MPP with Finite-Time Steady State

In this section, we discuss how MDPs of Finite State MPP with finite-time steady state can be factorized into finite-state MDP. This would then allow us to use Lemma 1 and apply standard optimal control algorithms for finite models. Note that finite-time steady state property does not imply that the MDP has finitely many states, as the clock values can still go unboundedly. Rather, the increments of the clocks beyond a certain horizon (i.e., H_i in Definition 3) do not change the transition probabilities.

Consider the Finite State MPP described in Section II-B. The main idea is to construct a stochastic bisimulation relation $\mathcal{R}$ that lumps together clock values beyond the horizon, as illustrated in Fig. 2 for one and two clocks. Thus, given two states of the MDP, (c, v, s) and (c', v', s'), both in $\mathbb{N}^N \times \mathbb{N}^M \times S$, we define $((c, v, s), (c', v', s')) \in \mathcal{R}$ if and only if

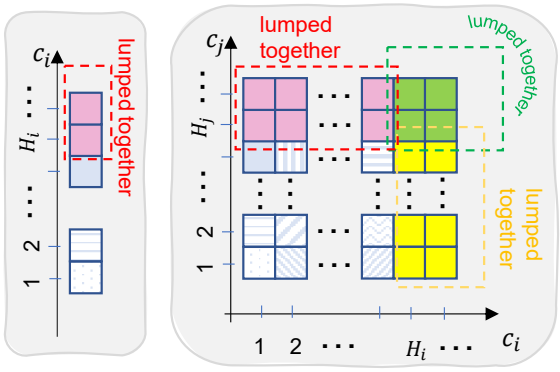

Fig. 2. Illustration of partitioning the clock values. The values of the clock c_i beyond the horizon H_i are lumped together. (Left) Illustration for one clock. (Right) Illustration for two clocks.

$$\bigwedge_{i=1}^{N} \left((c_i = c_i') \vee \left((c_i \geq H_i) \wedge (c_i' \geq H_i)\right)\right) \ldots$$

$$\bigwedge_{i=1}^{M} \left((v_i = v_i') \vee \left((v_i \geq K_i) \wedge (v_i' \geq K_i)\right)\right) \wedge (s = s').$$

The symbols $\wedge$ and $\vee$ denote the logical "AND" and "OR" operations, respectively.

We shall proceed to show that $\mathcal{R}$ is a stochastic bisimulation relation. First, we need to introduce some notations and a lemma.

Notation 8: For any $c = [c_1\ c_2\ \cdots\ c_N] \in \mathbb{N}^N$ and index set $I \subset \{1, 2, \cdots N\}$, let us define the following operations. $\texttt{Reset}_I(c)$ replaces the values of c at the indices in I with 1. That is, if $d = \texttt{Reset}_I(c)$ then

$$d_i = \begin{cases} 1, & i \in I, \\ c_i, & i \notin I. \end{cases}$$

$\texttt{Inc}_I(c)$ increments the values of c at the indices in I by 1. That is, if $d = \texttt{Inc}_I(c)$ then

$$d_i = \begin{cases} c_i + 1, & i \in I, \\ c_i, & i \notin I. \end{cases}$$

We can define analogous operations for $v \in \mathbb{N}^M$.

Lemma 2: The relation $\mathcal{R}$ is closed under the operations $\texttt{Reset}_I(\cdot)$ and $\texttt{Inc}_I(\cdot)$. That is, suppose that $((c, v, s), (c', v', s')) \in \mathcal{R}$ then for any index set I,

$$((\texttt{Inc}_I(c), v, s), (\texttt{Inc}_I(c'), v', s')) \in \mathcal{R},$$

$$((c, \texttt{Inc}_I(v), s), (c', \texttt{Inc}_I(v'), s')) \in \mathcal{R},$$

$$((\texttt{Reset}_I(c), v, s), (\texttt{Reset}_I(c'), v', s')) \in \mathcal{R},$$

$$((c, \texttt{Reset}_I(v), s), (c', \texttt{Reset}_I(v'), s')) \in \mathcal{R}.$$

Proof: Straightforward from the definition of $\mathcal{R}$. ■

Theorem 1: The relation $\mathcal{R}$ as defined above is a stochastic bisimulation relation.

Proof: Take any $(c, v, s), (c', v', s')$, and (c'', v'', s'') all in $\mathbb{N}^N \times \mathbb{N}^M \times S$ and any action u. Suppose that $((c, v, s), (c', v', s')) \in \mathcal{R}$. We need to show that both

Reward Equivalence and Transition Equivalence hold.

(Reward Equivalence) We need to show that

$$R((c'',v'',s''),u,(c,v,s)) = R((c'',v'',s''),u,(c',v',s')), \tag{34}$$

$$R((c,v,s),u,(c'',v'',s'')) = R((c',v',s'),u,(c'',v'',s'')). \tag{35}$$

Since $((c,v,s),(c',v',s')) \in \mathcal{R}$, we know that $s = s'$. Thus,

$$R((c'',v'',s''),u,(c,v,s)) = R((c'',v'',s''),u,(c,v,s')).$$

If $c' = c$ and $v' = v$ then Eq. (34) is implied immediately. If $c' \neq c$ then for any element i wherein they are different, we have

$$c_i \geq H_i,\ c'_i \geq H_i. \tag{36}$$

Similarly, if $v' \neq v$ then for any element i wherein they are different, we have

$$v_i \geq K_i,\ v'_i \geq K_i. \tag{37}$$

From here, Eq. (34) follows as a consequence of the finite-time steady state reward property in Definition 5. A similar argument can be formed to prove Eq. (35).

(Transition Equivalence) We need to show that

$$T((c,v,s),u,\mathcal{R}((c'',v'',s''))) = T((c',v',s'),\ u,\mathcal{R}((c'',v'',s'')). \tag{38}$$

As before, we infer that $s = s'$. Thus,

$$T((c,v,s),u,\mathcal{R}((c'',v'',s''))) = T((c,v,s'),u,\mathcal{R}((c'',v'',s''))). \tag{39}$$

If $c' = c$ and $v' = v$ then Eq. (38) is implied immediately. Using the definition of the transition probability in Eq. (11), we can conclude that for any $(\bar{c},\bar{v},\bar{s})$, if

$$\rho \triangleq T((c,v,s),u,(\bar{c},\bar{v},\bar{s})) > 0,$$

then $(\bar{c},\bar{v})$ can be obtained from applying a pair of reset and increment operations on (c,v). Suppose that applying the same operations on (c',v') yields $(\hat{c},\hat{v})$, then Lemma 2 implies that

$$((\bar{c},\bar{v},\bar{s}),(\hat{c},\hat{v},\hat{s})) \in \mathcal{R}.$$

Further, the transition probability defined in Eq. (11) implies

$$T((c',v',s'),u,(\hat{c},\hat{v},\hat{s})) = \rho.$$

It follows that Eq. (38) is true. ■

V. Example: Simplified Sleep-Wake Process Model

In this section, we apply the theory that we discussed in the previous sections to an example of the sleep-wake process. The body's internal clock includes circadian rhythms, which are 24-hour cycles that perform vital functions and processes in the background. Among the most significant and recognized circadian rhythms is the sleep-wake cycle.

In this example, we use a Finite State MPP as a simplified model of the sleep-wake cycle. It has two random events, *sleep* and *wake*, and a controlled event *nap*. The clocks are denoted as c_{sleep}, c_{wake}, and c_{nap}, respectively. The FSA part has 6 states, i.e.,

$$S = \{\text{asleep}, \text{awake}, \text{napping}\} \times \{\text{recent\_nap}, \text{recent\_sleep}\}.$$

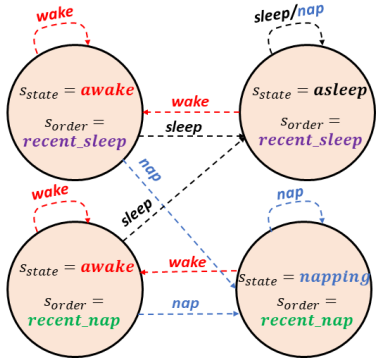

Fig. 3. Illustration of the finite state automaton in the sleep-wake cycle.

The first part of the discrete state, denoted as s_{state}, indicates whether the subject is asleep, awake, or napping, respectively. The second part, denoted as s_{order}, indicates whether the most recent non-awake period was asleep or napping, which influences the likelihood of the *sleep* happening. The finite state automaton for the sleep-wake cycle is shown in Fig.3. Note that only 4 discrete states are shown, due to the other 2 being not reachable (i.e., (asleep,recent_nap) and (napping, recent_sleep)).

The probabilities of occurrence of the events *sleep* and *wake* depend on the clock values and the discrete state, as summarized in the tables below.

$p_{sleep}(k)$	Conditions
0	$s_{state}(k)$ = asleep or napping
0	$s_{state}(k)$ = awake and $c_{wake}(k) < 8$
$e^{c_{wake}(k)-16}$	$s_{state}(k)$ = awake, $s_{order}(k)$ = recent_sleep, $8 \leq c_{wake}(k) \leq 16$
$e^{c_{wake}(k)-18}$	$s_{state}(k)$ = awake, $s_{order}(k)$ = recent_nap, $8 \leq c_{wake}(k) \leq 18$
1	else

$p_{wake}(k)$	Conditions
0	$s_{state}(k)$ = awake
$e^{c_{sleep}(k)-8}$	$s_{state}(k)$ = asleep, $c_{sleep}(k) < 8$
$e^{c_{nap}(k)-2}$	$s_{state}(k)$ = napping, $c_{nap}(k) < 2$
1	else

We assume that the event *nap* can only be triggered in the awake state. This is done by coding the transition probability function not to change the discrete state and the clocks if *nap* is triggered in the asleep or napping state. Also, if *sleep* and *nap* occur at the same time, then *nap* is ignored. Note that the model parameters are tuned such that if *nap* is never triggered (i.e., a no-nap policy) the typical trajectory has approximately 16 hours of awake and 8 hours of sleeping alternatingly. One such trajectory is shown in Fig. 4. When napping occurs, the model assumes that the subject wakes up randomly between 1 and 2 hours later, but no longer than 2 hours later.

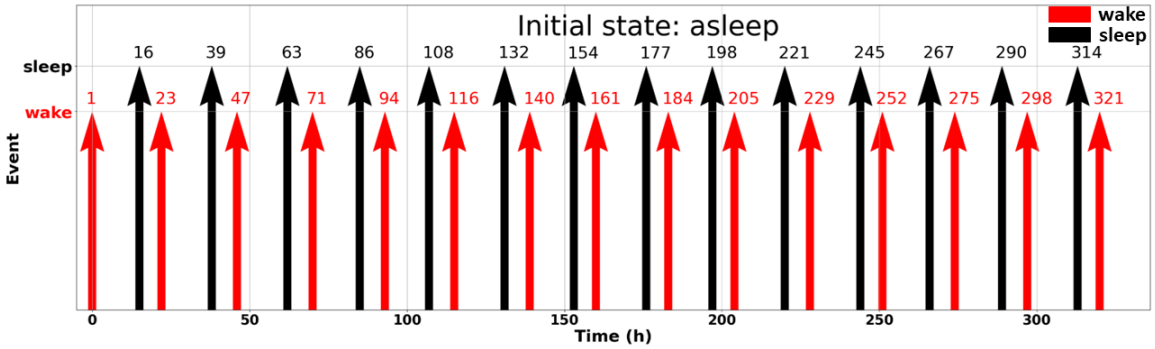

Fig. 4. Simulation trajectory for the *sleep* and *wake* events under no-nap policy. Observe that the lengths of sleep intervals (i.e., between wake and sleep events) are approximately 8 hours.

Reward function: In this example, we assume that we want to maximize the total (and discounted) lengths of the awake intervals. This is done by setting the reward function as 1, if $s_{state}(k) =$ awake, and 0 otherwise. The discount factor γ is chosen to be 0.99.

The probability tables above show that all events have a finite-time steady state. Specifically, the horizons are given by

$$H_{sleep} = 8, H_{wake} = 18, H_{nap} = 2.$$

The reward function also satisfies the finite-time steady state reward property. Therefore, Theorem 1 allows us to reduce the overall MDP to one with $8 \times 18 \times 2 \times 6 = 1728$ states.

We use the standard tabular policy iteration algorithm to calculate the optimal policy. The optimal policy is to trigger "*nap*" only in 8 states (of the factored MDP), which are defined by the following condition: $s_{state}(k)$ = awake, $s_{order}(k) =$ recent_sleep, $c_{sleep}(k) = 8^+$, $c_{nap}(k) = 2^+$, $12 \leq c_{wake}(k) \leq 15$, OR $s_{state}(k)$ = awake, $s_{order}(k) =$ recent_nap, $c_{sleep}(k) = 8^+$, $c_{nap}(k) = 2^+$, $14 \leq c_{wake}(k) \leq 17$.

The optimal policy can be interpreted as the subject taking naps to preempt natural sleep, thereby shortening the non-awake intervals (2 hours vs 8 hours). The policy also reveals the optimal timing for triggering *nap*. Triggering too late would increase the likelihood of failure to preempt *sleep*. Triggering too early would shorten the awake intervals. A sample trajectory obtained from running the optimal policy is shown in Fig. 5. Here, we can see that in most days the subject could preempt sleeping by taking short naps, except on Day 3 where *sleep* happened early.

VI. Conclusions and Future Research

We formulated an MDP formulation for optimal control of finite state automata that are driven by multivariate point processes. Generally, the MDP would involve infinitely many states, as some of the state components are the clock values of the events, which can grow unboundedly. In the special cases where the system has appropriate finite-time steady state properties, we show that the concept of stochastic bisimulation of MDP can be used to reduce the MDP into a finite state one, thereby allowing us to use standard optimal control techniques to calculate the optimal policy.

In the future, we believe generalizing the strict equivalence that is demanded by stochastic bisimulation to an approximate one is a potentially fruitful research direction. This idea can be coupled with a possibility to bound the sub-optimality of the policy calculated under the approximate equivalence relation. In particular, earlier work by Ferns, Panangaden, Precup and others may be useful in this effort.

Finally, we note that although our algorithm can find the optimal policy for this simplified sleep-wake process model, the result, which assumes that naps can completely preempt and replace sleep, is not realistic/practical. This is mainly because the simplified model does not capture *sleep homeostasis*, i.e., the quantity of "sleep debt". In the future, this model can be refined, e.g., by including multiple states of awake, each representing different amount of sleep debt. Here, nap and sleep will clear different amount of sleep debt, leading to a more realistic model and optimal policy.

Acknowledgment

The authors would like to thank Debarun Bhattacharjya, Ronny Luss, Tengfei Ma, and Achille Fokoue from IBM Research who introduced us to multivariate point process models and inspired the research presented in this paper. This work was supported by the Army Research Office through Grants W911NF-17-1-0562 and W911NF-22-10039 and by the National Science Foundation (NSF) through Grant DMS-2037357.

References

[1] O. Aalen, O. Borgan, and H. Gjessing, *Survival and event history analysis: a process point of view*. Springer Science & Business Media, 2008.

[2] D. J. Daley and D. Vere-Jones, *An Introduction to the Theory of Point Processes, Volume I: Elementary Theory and Methods*. Springer, 2003.

[3] E. Bacry, I. Mastromatteo, and J.-F. Muzy, "Hawkes processes in finance," *Market Microstructure and Liquidity*, vol. 1, no. 01, p. 1550005, 2015.

[4] M. Farajtabar, Y. Wang, M. Gomez Rodriguez, S. Li, H. Zha, and L. Song, "COEVOLVE: A joint point process model for information diffusion and network co-evolution," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 28, 2015, pp. 1954–1962.

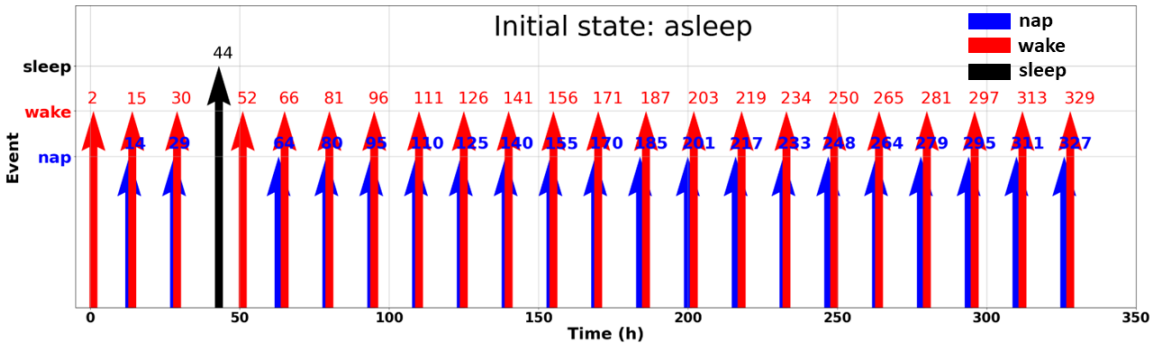

Fig. 5. Simulation trajectory for the *sleep*, *wake*, and *nap* events under optimal policy. Observe that under this policy the lengths of sleep intervals, mostly between nap and wake events, become shorter than before.

[5] N. Pini, M. Lucchini, W. P. Fifer, and R. Barbieri, "A point process framework for the characterization of fetal sleep states," in *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. IEEE, 2020, pp. 612–615.

[6] P. Brémaud, "Bang-bang controls of point processes," *Advances in Applied Probability*, vol. 8, no. 2, pp. 385–394, 1976. [Online]. Available: http://www.jstor.org/stable/1425910

[7] S. Rajaram, T. Graepel, and R. Herbrich, "Poisson-networks: A model for structured poisson processes," in *International Workshop on Artificial Intelligence and Statistics*. PMLR, 2005, pp. 277–284.

[8] J. Goulding, S. Preston, and G. Smith, "Event series prediction via non-homogeneous poisson process modelling," in *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 2016, pp. 161–170.

[9] K. Zhou, H. Zha, and L. Song, "Learning triggering kernels for multi-dimensional hawkes processes," in *International conference on machine learning*. PMLR, 2013, pp. 1301–1309.

[10] A. Gunawardana and C. Meek, "Universal models of multivariate temporal point processes," in *Artificial Intelligence and Statistics*. PMLR, 2016, pp. 556–563.

[11] D. Bhattacharjya, D. Subramanian, and T. Gao, "Proximal graphical event models," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 31, p. 8147–8156, 2018.

[12] D. Bhattacharjya, T. Gao, and D. Subramanian, "Ordinal historical dependence in graphical event models with tree representations," in *Proceedings of the Conference on Artificial Intelligence (AAAI)*, 2021, pp. 6759–6767.

[13] M. Zhou and M. Huang, "Mean field games with poisson point processes and impulse control," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 3152–3157.

[14] J. Kim, B. Tabibian, A. Oh, B. Schölkopf, and M. Gomez-Rodriguez, "Leveraging the crowd to detect and reduce the spread of fake news and misinformation," in *Proceedings of the eleventh ACM international conference on web search and data mining*, 2018, pp. 324–332.

[15] Y. Wang, E. Theodorou, A. Verma, and L. Song, "A stochastic differential equation framework for guiding online user activities in closed loop," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2018, pp. 1077–1086.

[16] U. Upadhyay, A. De, and M. Gomez-Rodrizuez, "Deep reinforcement learning of marked temporal point processes," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, ser. NIPS'18. Red Hook, NY, USA: Curran Associates Inc., 2018, p. 3172–3182.

[17] D. P. Bartsekas, *Dynamic Programming and Optimal Control*. Athena Scientific, 2007.

[18] F. L. Lewis, D. L. Vrabie, and V. L. Syrmos, *Optimal Control*. John Wiley and sons, Inc., 2012.

[19] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. A Bradford Book, 2018.

[20] R. Givan, T. Dean, and M. Greig, "Equivalence notions and model minimization in markov decision processes," *Artificial Intelligence*, vol. 147, pp. 163–223, 2003.

[21] N. Ferns, P. Panangaden, and D. Precup, "Metrics for finite markov decision processes," in *UAI 2004*, 2004, pp. 162–169.