This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2021.3094719

Learning-based Predictive Control via Real-time Aggregate Flexibility

Tongxin Li, *Student Member, IEEE,* Bo Sun, *Member, IEEE,* Yue Chen, *Member, IEEE,* Zixin Ye, *Student Member, IEEE,* Steven H. Low, *Fellow, IEEE,* and Adam Wierman, *Member, IEEE*

***Abstract*—Aggregators have emerged as crucial tools for the coordination of distributed, controllable loads. To be used effectively, an aggregator must be able to communicate the available flexibility of the loads they control, as known as the aggregate flexibility to a system operator. However, most of existing aggregate flexibility measures often are slow-timescale estimations and much less attention has been paid to real-time coordination between an aggregator and an operator. In this paper, we consider solving an online optimization in a closed-loop system and present a design of *real-time* aggregate flexibility feedback, termed the *maximum entropy feedback* (MEF). In addition to deriving analytic properties of the MEF, combining learning and control, we show that it can be approximated using reinforcement learning and used as a penalty term in a novel control algorithm – the *penalized predictive control* (PPC), which modifies vanilla model predictive control (MPC). The benefits of our scheme are (1). *Efficient Communication*. An operator running PPC does not need to know the exact states and constraints of the loads, but only the MEF. (2). *Fast Computation*. The PPC often has much less number of variables than an MPC formulation. (3). *Lower Costs* We show that under certain regularity assumptions, the PPC is optimal. We illustrate the efficacy of the PPC using a dataset from an adaptive electric vehicle charging network and show that PPC outperforms classical MPC.**

***Index Terms*—Aggregate flexibility, closed-loop control systems, online optimization, model predictive control, reinforcement learning, electric vehicle charging**

I. Introduction

The uncertainty and volatility of renewable sources such as wind and solar power has created a need to exploit the flexibility of distributed energy resources (DERs) and aggregators have emerged as dominate players for coordinating these loads [1], [2]. An aggregator can coordinate a large pool of DERs and be a single point of contact for independent system operators (ISOs) to call on for flexibility. This enables ISOs to minimize cost, respond to unexpected fluctuations of renewables, and even mitigate failures quickly and reliably. Typically, an ISO communicates a time-varying signal to an aggregator, e.g., a desired power profile, that optimizes ISO objectives and the aggregator coordinates with the DERs to collectively respond to the time-varying signal as faithfully as possible, e.g., by shaping their aggregate power consumption to follow ISO's power profile, while satisfying DER constraints. These constraints are often private to the loads, e.g., satisfying energy demands of electric vehicles before their deadlines. They limit the flexibility available to the aggregator so the aggregator must also communicate with the ISO by providing a feedback signal that quantifies its available flexibility. This feedback provides ISO with crucial information for determining the signal it sends to the aggregator. Thus the aggregator and the ISO form a closed-loop control system to manage the aggregate flexibility of DERs.

This paper focuses on the design of this closed-loop system and, in particular, the design of real-time feedback signal from the aggregator to the ISO quantifying the available flexibility. The design of *the aggregate flexibility feedback* signal is complex and has been the subject of significant research over the last decade, e.g., [3], [4], [5], [6], [7], [8], [9], [10], [11]. Any feedback design must balance a variety of conflicting goals. Given the scale, complexity and privacy of the load constraints, it may neither be possible nor desirable to communicate precise information about every load. Instead, aggregate flexibility feedback must be a concise summary of a system's constraints and it must limit the leakage about specific load constraints. On the other hand, the feedback sent by an aggregator needs to be informative enough that it allows the ISO to achieve operational objectives, e.g., minimize cost, and, most importantly, containing feasibility information of the whole system with respect to the private load constraints. Moreover, a design for a flexibility feedback signal must be general enough to be applicable for a wide variety of controllable loads, e.g., electric vehicles (EVs), heating, ventilation, and air conditioning (HVAC) systems, energy storage units, thermostatically controlled loads, residential loads, and pool pumps. It is impractical to design different feedback signals for each load, so the same design must work for all DERs.

The challenge and importance of the design of flexibility feedback signals has led to the emergence of a rich literature. In many cases, the literature focuses on specific classes of controllable loads, such as EVs [12], heating, ventilation, and air conditioning (HVAC) systems [13], [14], energy storage units [10], thermostatically controlled loads [4] or residential

Tongxin Li and Steven Low acknowledge the support received from National Science Foundation (NSF) through grants CCF 1637598, ECCS 1931662 and CPS ECCS 1932611. Bo Sun is supported by Hong Kong Research Grant Council (RGC) General Research Fund (Project 16207318). Adam Wierman's research is funded by NSF (AitF-1637598 and CNS-1518941), Amazon AWS and VMware.

Li, Low and Wierman are with the Computing + Mathematical Sciences Department, California Institute of Technology, Pasadena, CA 91125 USA (e-mails: {tongxin, slow, adamw}@caltech.edu)

Sun is with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong (e-mail: bsunaa@connect.ust.hk)

Chen is with the Department of Mechanical and Automation Engineering, the Chinese University of Hong Kong, Hong Kong SAR, China. (e-mail: yuechen@mae.cuhk.edu.hk)

Ye is with the Electrical Engineering Department, California Institute of Technology, Pasadena, CA 91125 USA (e-mail: zyye@caltech.edu)

loads and pool pumps [5], [15]. In the context of these applications, a variety of approaches have been suggested, e.g., convex geometric approximations via virtual battery models [4], [6], hyper-rectangles [8] and graphical interpretations [10]; scheduling based aggregation [16], [17]; linear combination of demand bit curves [14]; and probability-based characterization [5], [15]. These approaches have all yielded some success, especially in terms of quantifying available aggregate flexibility (see Section I-B for more detail on related work). However, nearly all prior work only focused on slower-timescale estimations and does not meet the goal of providing *real-time* aggregate flexibility feedback. The fast-changing environment and the uncertainties of the DERs, however, demand real-time flexibility feedback. For example, in an EV charging facility, it is notoriously challenging to predict future EV arrivals and their battery capacities. With on-site solar generation, the aggregator's dynamical system can be time-varying and non-stationary, so it is crucial that real-time feedback be defined and approximated for it to be used in online feedback-based applications. Furthermore, most of the existing frameworks are designated for specific tasks, such as managing HVAC systems [13], [14], and therefore may not be applicable to other applications. Reinforcement learning (RL), especially, deep RL, has been used widely as approximation tools in smart grid applications. Joint pricing and EV charging scheduling for a single EV charger is considered in [18] using state–action–reward–state–action (SARSA). But it is unclear how the proposed method in [18] can be extended to allow multiple chargers. Q-learning is used to estimate the residual energy in an energy storage system at the end of each day in [19] and determine the aggregate action for thermostatically controlled loads (TCLs) [20]. The authors in [21] combine evolution strategies and model predictive control (MPC) to coordinate heterogeneous TCLs. Most existing studies, including the aforementioned works typically use RL for a "central controller" (which is an operator in our context). Instead we use it for the aggregator to learn flexibility representations.

To the best of our knowledge, no paper has focused on the design of real-time coordination between an aggregator and a system operator that achieves the goals laid out above, except for some preliminary results in [22], [23]. Those results rely on a novel design of a real-time feedback signal that can be used to quantify the aggregate flexibility and coordinate real-time control. In this paper, we extend the design of the feedback signal to a more general dynamic system with time-varying and non-stationary constraints, and we mainly focus on how to apply the real-time feedback to practical applications (e.g., EV charging) in power systems. Towards this goal, we propose a reinforcement learning based approach to approximate this feedback and further incorporate the feedback into a penalized predictive control (PPC) scheme. On the theory side, we prove the optimality of the proposed PPC scheme, and through extensive numerical tests, we validate the superior empirical performance of PPC over classic benchmarks, such as MPC.

A. Contributions.

In summary, to complement previous research, this paper considers a closed-loop control model formed by a system operator (central controller) and an aggregator (local controller) and propose a novel design of *real-time aggregate flexibility* feedback, called the *maximum entropy feedback* (MEF) that quantifies the flexibility available to an aggregator. Based on the definition of MEF, we design a reward function, which allows MEF to be efficiently learned by model-free RL algorithms. Our main contributions are:

1) We introduce a model of the real-time closed-loop control system formed by a system operator and an aggregator. This work is the first to close the loop and both define a concise measure of aggregate flexibility and show how it can be used by the system operator in an online manner to optimize system objectives while respecting the constraints of the aggregator's loads.
2) Within this model we define the "optimal" real-time flexibility feedback as the solution to an optimization problem that maximizes the entropy of the feedback vector. The use of entropy in this context is novel and to the best of our knowledge, this article is among the first to rigorously define a notion for *real-time* aggregate flexibility with provable properties. In particular we show that the exact MEF allows the system operator to maintain feasibility and enhance flexibility.
3) Furthermore, we propose a novel combination of control and learning by integrating model predictive control (MPC) and the defined MEF. Using the MEF as a penalty term, we introduce an algorithm called the *penalized predictive control* (PPC), which only requires the system operator to receive the MEF at each time, *without* knowing the states and dynamics of the aggregator. We also prove that, under certain regularity conditions, the actions given by PPC are optimal.
4) Finally, we demonstrate the efficacy of the proposed scheme using real EV charging data from Caltech's ACN-Data [24]. Our experiments show that by sending simple action signals generated by the PPC, a system operator is able to coordinate with an EV charging aggregator to satisfy almost all EV charging demands, while only knowing the MEF learned by a model-free off-policy RL algorithm. The PPC is also showed to achieve lower cost than MPC, which in addition needs to have access to the complete state of the loads.

B. Related literature.

The growing importance of aggregators for the integration of controllable loads and the challenge of defining and quantifying the flexibility provided by aggregators has led to the emergence of a rich literature. Broadly, this work can be separated into three approaches.

Convex geometric approximation. The idea of representing the set of aggregate loads as a virtual battery model dates back to [3], [4]. In [6], flexibility of an aggregation of thermostatically controlled loads (TCLs) was defined as the Minkowski sum of individual polytopes, which is approximated by the homothets of a virtual battery model using linear programming. The recent paper [8] takes a different approach and defines the aggregate flexibility as upper and lower bounds

so that each trajectory to be tracked between the bounds is disaggregatable and thus feasible. However, convex geometric approaches cannot be extended to generate real-time flexibility signals because the approximated sets cannot be decomposed along the time axis. In [11], a belief function of setpoints is introduced for real-time control. However, feasibility can only be guaranteed when each setpoint is in the belief set and this may not be the case for systems with memory.

Scheduling algorithm-driven analysis. Scheduling algorithms that enable the aggregation of loads have been studied in depth over the past decade. The authors of [25], [26] introduced a decentralized algorithm with a real-time implementation for EV charging to track a given load profile. The authors of [27] considered the feasibility of matching a given power trajectory and show that causal optimal policies do not exist. In this work, aggregate flexibility was implicitly considered as the set of all feasible power trajectories. Three heuristic causal scheduling policies were compared and the results were extended to aggregation of deferrable loads and storage in [16]. Furthermore, decentralized participation of flexible demand from heat pumps and EVs was addressed in [17]. Notably, the flexibility signals that have emerged from this literature generally are applicable only to specific policies and DERs.

Probability-based characterization. There is much less work on probabilistic methods. The aggregate flexibility of residential loads was defined based on positive and negative pattern variations by analyzing collective behaviour of aggregate users [5]. A randomized and decentralized control architecture for systems of deferrable loads was proposed in [15], with a linear time-invariant system approximation of the derived aggregate nonlinear model. Flexibility in this work was defined as an estimate of the proportion of loads that are operating. Our work falls into this category, but differs from previous papers in that entropy maximization for a closed-loop control system yields an interpretable signal that can be informative for operator objectives in real-time, as well as guarantee feasibility of the private constraints of loads (if the signal is accurate). In our previous work [23], we study the problem of real-time coordination of an aggregator and a system operator under the paradigm of a control framework and provide regret analysis assuming feasibility predictions are available.

Other approaches. Beyond the works described above, there are many other suggestions for metrics of aggregate flexibility, e.g., graphical-based measures [28] and data-driven approaches [28]. Most of these, and the approaches described above, are evaluated on the aggregator side only, and much less attention has been paid to the question of real-time coordination between an ISO and an aggregator that controls decentralized loads.

The assessment and enhancement of aggregate flexibility are often considered independent of the operational objectives. For instance, in a reserve market, an aggregator will report to the ISO a day in advance an offline notion of aggregated flexibility based on forecast for the ISO to compute a energy and reserve schedule for the following day, e.g., [3], [29], [8], [7], with notable exceptions, such as [12], which considered charging and discharging of EV fleets batteries for tracking a sequence of automatic generation control (AGC) signals. However, this approach has several limitations. First, in large-scale systems, knowing the exact states of each load is not realistic. Second, classical flexibility representations often rely on a precise state-transition model on the aggregator's side. Third, traditional ISO market designs, such as a day-ahead energy market, often make use of ex ante estimates of future system states. The forecasts of the future states can sometime be far from reality, because of either an inaccurate model is used, or an uncertain event occurs. In contrast, a real-time energy market [30], [31] provides more robust system control when facing uncertainty in the environment, e.g., from fast-changing renewable resources or human behavioral parameters. This further highlights the need for real-time flexibility feedback, and serves to differentiate the approach in our paper. Below we present the notation frequently used in the remainder of this paper.

NOMENCLATURE TABLE

C. System Operator (Central Controller)

T	Total number of time slots.
t	Time index.
u_t	Operator action.
c_t	Cost function.
C_T	Cumulative costs.
ψ_t	Operator function.
β_t	Tuning parameter.

D. Aggregator (Local Controller)

x_t	Aggregator state.
p_t	Real-time aggregate flexibility feedback.
X_t	Set of feasible states.
U_t	Set of feasible actions.
S	Set of feasible action trajectories.
f_t	State transition function.
P	Set of flexibility feedback.
ϕ_t	Aggregator function.

E. EV Charging Example

N	Total number of accepted charging sessions.
j	Index of charging sessions.
u_t	Aggregate substation power level.
s_t	Charging decision vector.
$s_t(j)$	Scheduled energy.
$a(j)$	Arrival time.
$d(j)$	Departure time.
$e(j)$	Total energy to be delivered.
$r(j)$	Peak charging rate.
$d_t(j)$	Remaining charging time.
$e_t(j)$	Remaining energy demand.
Δ	Time unit.

Notation and Conventions. We use $\mathbb{P}(\cdot)$ and $\mathbb{E}(\cdot)$ to denote the probability distribution and expectation of random variables. The (differential) entropy function is denoted by $\mathbb{H}(\cdot)$. To distinguish random variables and their realizations, we follow the convention to denote the former by capital

letters (e.g., U) and the latter by lower case letters (e.g., u). Furthermore, we denote the length-t prefix of a vector u by $u_{\leq t} := (u_1, \ldots, u_t)$. Similarly, $u_{<t} := (u_1, \ldots, u_{t-1})$ and $u_{a\to b} := (u_a, \ldots, u_b)$. The concatenation of two vectors u and v is denoted by (u, v). Given two vectors $u, v \in \mathbb{R}^N$, we write $u \preceq v$ if $u_i \leq v_i$ for all $i = 1, \ldots, N$. For $x \in \mathbb{R}$, denote $[x]_+ := \max\{0, x\}$. The set of non-negative real numbers is denoted by $\mathbb{R}_+$.

The rest of the paper is organized as follows. We present our closed-loop control model in Section II. We define real-time aggregate flexibility, called the MEF, and prove its properties in Section III. An RL-based approach for estimating the MEF is provided in Section IV. Combining MEF and model MPC, we propose an algorithm, termed the PPC in Section V-B. Numerical results are given in Section VI. Finally, we conclude this paper in Section VII.

II. Problem Formulation

In this paper, we consider a real-time control problem involving two parties – a *load aggregator* and an independent system operator (*ISO*), or simply called an *operator* that interact over a discrete time horizon $[T] := \{1, \ldots, T\}$.

A. Load aggregator

A *load aggregator* is a device, often considered as a local controller that controls a fleet of controllable loads. In this part, we formally state the model of an aggregator and its objective. Let x_t denote the *aggregator state* at time t that takes value in a certain set $\mathsf{X} \subseteq \mathbb{R}^m$. To this end, the aggregator receives an *action* $u_t \in \mathsf{U}$ where $\mathsf{U} \subseteq \mathbb{R}$ denotes a closed and bounded set of actions at each time t from a *system operator*, which will be formally defined in Section II-B. The action space U and state space X are prefixed and known as common knowledge to both the aggregator and the system operator. The goal of the aggregator is to accomplish a certain task over the horizon $[T]$, e.g., delivering energy to a set of EVs by their deadlines while minimizing the costs, subject to system constraints. Mathematically, the constraints are represented by two collections of *time-varying* and *time-coupling* sets $\{\mathsf{X}_t(x_{<t}, u_{<t}) \subseteq \mathsf{X} : t \in [T]\}$ and $\{\mathsf{U}_t(x_{<t}, u_{<t})_t \subseteq \mathsf{U} : t \in [T]\}$. For notational simplicity, we denote $\mathsf{X}_t(x_{<t}, u_{<t})$ by X_t and $\mathsf{U}_t(x_{<t}, u_{<t})$ by U_t in the remaining contexts. The states and actions must satisfy $x_t \in \mathsf{X}_t$ and $u_t \in \mathsf{U}_t$ for all $t \in [T]$. The decision changes the aggregator state x_t according to a *state transition function* f_t:

$$x_{t+1} = f_t(x_t, u_t), \;\; x_t \in \mathsf{X}_t, \; u_t \in \mathsf{U}_t, \tag{1}$$

where f_t represents the transition of the state x_t. The initial state x_1 is assumed to be the origin without loss of generality. The aggregator state x_t and decision u_t need to be chosen from two time-varying sets X_t and U_t. We make the following model assumptions:

***Assumption* 1.** *The dynamic $f_t(\cdot, \cdot) : \mathsf{X}_t \times \mathsf{U}_t \to \mathsf{X}_{t+1}$ is a Borel measurable function for $t \in [T]$. The time-varying and time-coupling sets $\{\mathsf{U}_t : t \in [T]\}$ and $\{\mathsf{X}_t : t \in [T]\}$ are Borel sets in $\mathbb{R}$ and $\mathbb{R}^m$.*

The aggregator has flexibility in its actions u_t for accomplishing its task and, we assume for this paper, is indifferent to these decisions as long as the task is accomplished by time T. At each time t, based on its current state x_t, the aggregator needs to send *flexibility feedback*, p_t, a probability density function, from a collection of feedback signals P, to the system operator, which describes the flexibility of the aggregator for accepting different actions u_t. We formally define p_t and P in Section III-A. Designing p_t is one of the central problems considered in this paper (see Section III for more details). Below we state the aggregator's goal in the real-time control system.

***Aggregator's Objective*.** *The goal of the aggregator is two-fold: (1). Maintain the feasibility of the system and guarantee that $x_t \in \mathsf{X}_t$ and $u_t \in \mathsf{U}_t$ for all $t \in [T]$. (2). Generate flexibility feedback p_t and send it to the operator at time $t \in [T]$.*

***Remark* 1.** *We assume that the action space U is a continuous set in $\mathbb{R}$ only for simplicity of presentation. The results and definitions in the paper can be extended to discrete setting by changing the integrals to summations, and replacing the differential entropy functions by discrete entropy functions, e.g., see the definition of maximum entropy feedback (Definition III.1) and Lemma 2. In practical systems e.g., an electric system consisting of an EV aggregator and an operator, U often represents the set of power levels and when the gap between power levels is small, U can be modeled as a continuous set.*

B. System operator

A system operator is a central controller that operates the power grid. Knowing the flexibility feedback p_t from the aggregator, the operator sends an action u_t, chosen from U to the aggregator at each time $t \in [T]$. Each action is associated with a cost function $c_t(\cdot) : \mathsf{U} \to \mathbb{R}_+$, e.g., the aggregate EV charging rate increases load on the electricity grid. The system's objective is stated as follows.

***Operator's Objective*.** *The goal of the system operator is to provide an action $u_t \in \mathsf{U}$ at time $t \in [T]$ to the aggregator so as to minimize the cumulative system costs given by $C_T(u_1, \ldots, u_T) := \sum_{t=1}^T c_t(u_t)$.*

C. Real-time operator-aggregator coordination

Overall, considering the aggregator and operator's objectives, the goal of the closed-loop system is to solve the following problem in *real-time*, by coordinating the operator and aggregator via $\{p_t : t \in [T]\}$ and $\{u_t : t \in [T]\}$:

$$\min_{u_1, \ldots, u_T} C_T(u_1, \ldots, u_T) \tag{2a}$$

$$\text{subject to } \forall t = 1, \ldots, T:$$

$$x_{t+1} = f_t(x_t, u_t) \tag{2b}$$

$$x_t \in \mathsf{X}_t, \tag{2c}$$

$$u_t \in \mathsf{U}_t \tag{2d}$$

i.e., the operator aims to minimize its cost C_T in (2a) while the load aggregator needs to fulfill its obligations in the form of

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2021.3094719

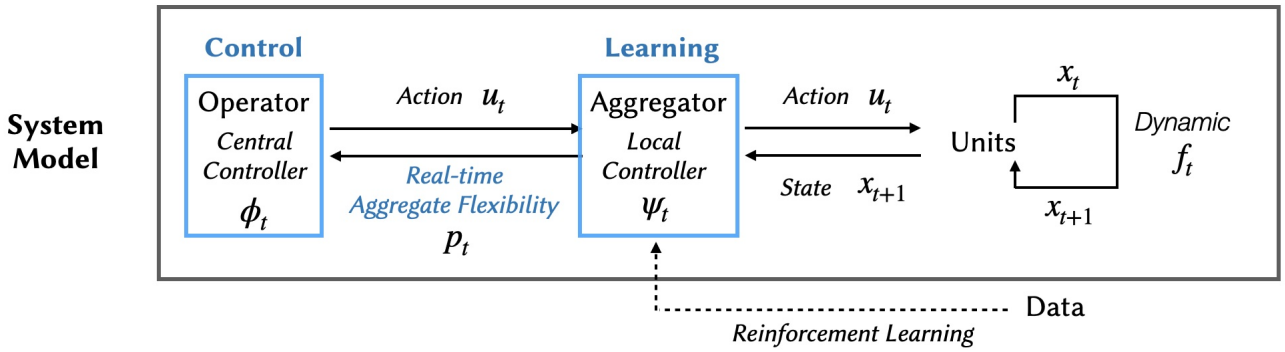


Fig. 1. System model: A feedback control approach for solving an online version of (2). The operator implements a control algorithm and the aggregator uses reinforcement learning to generate real-time aggregate flexibility feedback.

constraints (2b)-(2d). This is an offline problem that involves global information at all times $t \in [T]$.

Remark 2. *For simplicity, we describe our model in an offline setting where the cost and the constraints in the optimization problem* (2) *are expressed in terms of the entire trajectories of states and actions. The goal of the closed-loop control system is, however, to solve an online optimization via operator-aggregator coordination.*

The challenges are: (i) the aggregator and operator need to solve the online version of (2) jointly, and (ii) the cost function C_T is private to the operator and the constraints (2b)-(2d) are private to the operator. It is impractical for the aggregator to communicate the constraints to the operator because of privacy concerns or computational effort. Moreover, in an online setting, even the aggregator will not know the constraints that involve future information, e.g., future EV arrivals in an EV charging station. Formally, at each time $t \in [T]$, we assume that the operator and aggregator have access to the following information respectively:

1) *An operator knows the costs* $(c_1, \ldots, c_t)$ *and feedback* $(p_1, \ldots, p_t)$*, but not the future costs* $(c_{t+1}, \ldots, c_T)$ *and feedback* $(p_{t+1}, \ldots, p_T)$.
2) *An aggregator knows the state transition functions* $(f_1, \ldots, f_T)$*, the initial state* x_1 *and actions* $(u_1, \ldots, u_t)$.

System's Goal. *Overall, the goal of a aggregator-operator system is to jointly solve the online version of* (2a)-(2d) *whose partial information is known to an aggregator and an operator respectively.*

D. Necessities of combining learning and control

With the assumptions above, on the one hand the aggregator cannot solve the problem independently because it does not have cost information (since the costs are often sensitive and only of the operator's interests) from the operator and even if the aggregator could, it may not have enough power to solve an optimization to obtain an action. On the other hand, the operator has to receive flexibility information from the aggregator in order to act. Well-known methods in pure learning or control cannot be used for this problem directly. From a learning perspective, the aggregator cannot simply use reinforcement learning and transmit parameters of a learned Q-function or an actor-critic model to the operator because the aggregator does not know the costs. From a control perspective, although model predictive control (MPC) is widely used for EV charging scheduling in practical charging systems [32], [24], it requires precise state information of electric vehicle supply equipment (EVSE). Thus, to solve the induced MPC problem, the system operator or aggregator needs to solve an online optimization at each time step that involves hundreds or even thousands of variables. This not just a complex problem, but the state information of the controllable units is potentially sensitive. This combination makes controlling sub-systems using precise information impractical for a future smart grid [22], [23] In this work, we explore a solution where the system operator and the aggregator jointly solve an online version of (2) in a closed loop, as illustrated in Figure 1.

The real-time operator-aggregator coordination illustrated in Figure 1 combines learning and control approaches. It does not require the aggregator to know the system operator's objective in (2a), but only the action u_t at each time $t \in [T]$ from the operator. In addition, it does not require the system operator to know the aggregator constraints in (2b), but only a feedback signal p_t (to be designed) from the aggregator. After receiving flexibility feedback p_t, which could be generated by machine learning algorithms, the system operator outputs an action u_t using a causal *operator function* $\phi_t(\cdot) : \mathsf{P} \to \mathsf{U}$. Knowing the state x_t, the aggregator generates its feedback p_t using a causal *aggregator function* $\psi_t(\cdot) : \mathsf{X} \to \mathsf{P}$ where P denotes the domain of flexibility feedback that will be formally defined in Section III-A. By an "online feedback" solution, we mean that these functions (ϕ_t, ψ_t) use only information available locally at time $t \in [T]$.

In summary, the closed-loop control system in our model proceeds as follows. At each time t, the aggregator learns or computes a length-$|\mathsf{U}|$ vector p_t based on previously received action trajectory $u_{<t} = (u_1, \ldots, u_{t-1})$, and sends it to the sys-

for $t \in [T]$ **do**

Operator (Central Controller)

Generate actions using the PPC:

$$u_t = \phi_t(p_t)$$
$$C_t = C_{t-1} + c_t(u_t)$$

Aggregator (Local Controller)

Update system state:

$$x_{t+1} = f_t(x_t, u_t)$$

Compute estimated MEF:

$$p_{t+1} = \psi_t(x_{t+1})$$

end

Return Total cost C_T;

Algorithm 1: Closed-loop online control framework of a system operator (central controller) and an aggregator (local controller).

tem operator.[1] The system operator thencomputes a (possibly random) action $u_t = \phi_t(p_t)$ based on the flexibility feedback p_t and sends it to the aggregator. The operator chooses its signal u_t in order to solve the time-t problem in an online version of (2), so the function ϕ_t denotes the mapping from the flexibility feedback p_t to an optimal solution of the time-t problem. See V-B for examples. The aggregator then computes the next feedback p_{t+1} and the cycle repeats; see Algorithm 1. The goal of this paper is to provide concrete constructions of an aggregator function ψ (as an MEF generator; see Section III) and an operator function ϕ (via the PPC scheme; see Section V-B).

In the sequel, we demonstrate our system model using an EV charging application, as an example of the problem stated in (2).

E. An EV Charging Example

Consider an aggregator that is an EV charging facility with N accepted users. Each user j has a private vector $(a(j), d(j), e(j), r(j)) \in \mathbb{R}^4$ where $a(j)$ denotes its arrival (connecting) time; $d(j)$ denotes its departure (disconnecting) time, normalized according to the time indices in $[T]$; $e(j)$ denotes the total energy to be delivered, and $r(j)$ is its peak charging rate. Fix a set of N users with their private vectors $(a(j), d(j), e(j), r(j))$, the *aggregator state* x_t at time $t \in [T]$ is a collection of length-2 vectors $(d_t(j), e_t(j) : a(j) \le t \le d(j))$ for each EV that has arrived and has not departed by time t. Here $e_t(j)$ is the remaining energy demand of user j at time t and $d_t(j)$ is the remaining charging time. The decision $s_t(j)$ is the energy delivered to each user j at time t, determined by a *scheduling policy* π_t such as the well-known earliest-deadline-first, least-laxity-first, etc. Let $s_t := (s_t(1), \ldots, s_t(N))$ and we have $s_t = \pi_t(u_t)$ where u_t in this example is the aggregate substation power level, chosen from a small discrete set U. The aggregator decision $s_t(j) \in \mathbb{R}_+$ at each time t updates the state, in particular $e_t(j)$ such that

$$e_t(j) = e_{t-1}(j) - s_t(j) \tag{3a}$$
$$d_t(j) = d_{t-1}(j) - \Delta \tag{3b}$$

where Δ denotes the time unit and we assume that there is no energy loss. The laws (3a)-(3b) are examples of the generic transition functions $f_1, \ldots, f_T$ in (1).

Suppose, in the context of demand response, the system operator (a local utility company, or a building management) sends a signal u_t that is the aggregate energy that can be allocated to EV charging. The aggregator makes charging decisions $s_t(j)$ to track the signal u_t received from the system operator as long as they will meet the energy demands of all users before their deadlines. Then the constraints in (2b)-(2d) are the following constraints on the charging decisions s_t, as a function of u_t:

$$s_t(j) = 0 \ , \ t < a(j), \ j = 1, \ldots, N, \tag{4a}$$
$$s_t(j) = 0 \ , \ t > d(j), \ j = 1, \ldots, N, \tag{4b}$$
$$\sum_{j=1}^{N} s_t(j) = u_t, \ t = 1, \ldots, T, \tag{4c}$$
$$\sum_{t=1}^{T} s_t(j) = e(j), \ j = 1, \ldots, N, \tag{4d}$$
$$0 \le s_t(j) \le r(j), \ t = 1, \ldots, T \tag{4e}$$

In above, constraint (4c) ensures that the aggregator decision s_t tracks the signal u_t at each time $t \in [T]$, the constraint (4d) guarantees that EV j's energy demand is satisfied, and the other constraints say that the aggregator cannot charge an EV before its arrival, after its departure, or at a rate that exceeds its limit. Inequalities (4a)-(4e) above are examples of the constraints in (1). Together, for this EV charging application, (3a)-(3b) and (4a)-(4e) exemplify the dynamic system in (1).

The system operator's objective to minimize the cumulative costs $C_T(u) := \sum_{t=1}^{T} c_t u_t$ where $u = (u_1, \ldots, u_T)$ are substation power levels, as outlined in Section II-B. The cost c_t depends on multiple factors such as the electricity prices and injections from an installed rooftop solar panel. Overall, the EV charging problem is formulated below, as a specific example of the generic optimization (2a)-(2d):

$$\min_{u_1, \ldots, u_T} \sum_{t=1}^{T} c_t u_t \tag{5a}$$
$$(3a) - (3b) \text{ and } (4a) - (4e) \tag{5b}$$

III. Definitions of Real-time Aggregate Flexibility: Maximum Entropy Feedback

In this section we propose a specific function ψ_t in the class defined by (6) for computing flexibility feedback to quantify its future flexibility. We will justify our proposal by showing that the proposed ψ_t has several desirable properties for solving

[1] We will omit $u_{<t}$ in the notation when it is not essential to our discussion and simplify the probability vector as p_t. Note that in (7c) we slightly abuse the notation and use p_t to denote a conditional distribution. This is only for computational purposes and the information sent from an aggregator to an operator at time $t \in [T]$ is still a length-$|\mathsf{U}|$ probability vector, conditioned on a fixed $u_{<t}$.

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2021.3094719

an online version of (2) using the real-time feedback-based approach described in Section II.

A. Definition of Flexibility Feedback p_t

A major challenge in our problem is that the operator has access to neither the feasible set nor the dynamics directly. Therefore, a notion termed *aggregate flexibility* has to be designed. It is often a "simplified" summary of the constraints in (2b)-(2d), as we reviewed in Section I-B. Notably, existing aggregate flexibility definitions (for instance, in [3], [4], [5], [6], [7], [8], [9], [10]) all focus on the offline version of (2). It remains unclear that first, *what is the right notion of real-time aggregate flexibility? i.e., what is the right form of the flexibility feedback p_t?* Second, *how can this p_t be used by an operator?*

In the following, we present a design of the flexibility feedback p_t, which is first proposed in our previous work [22] for discrete U and [23] for continuous U. It quantifies future flexibility that will be enabled by an operator action u_t. The feedback p_t therefore is a surrogate for the aggregator constraints (2b) to guide the operator's decision. Let $u := (u_1, \dots, u_T)$. Specifically, define the set of all *feasible action trajectories* for the aggregator as:

$$\mathsf{S} := \left\{ u \in \mathsf{U}^T : u \text{ satisfies (2b)} - \text{(2d)} \right\}.$$

The following property of the set S is useful, whose proof can found in Appendix A.

***Lemma* 1.** *The set of feasible action trajectories S is Borel measurable.*

Existing aggregate flexibility definitions focus on approximating S such as finding its convex approximation (see Section I-B for more details). Our problem formulation needs a *real-time* approximation of this set S, i.e., decompose S along the time axis $t = 1, \dots, T$. Throughout, we assume that S is non-empty. Next, we define the space of flexibility feedback p_t. Formally, we let P denote a set of density functions $p_t(\cdot) : \mathsf{U} \to [0,1]$ that maps an action to a value in $[0,1]$ and satisfies

$$\int_{u \in \mathsf{U}} p(u) \mathrm{d}u = 1.$$

Fix x_t at time $t \in [T]$. The aggregator function $\psi_t(\cdot) : \mathsf{X} \to \mathsf{P}$ at each time t outputs:

$$\psi_t(x_t) = p_t(\cdot | u_{<t}) \tag{6}$$

such that $p_t(\cdot|u_{<t}) : \mathsf{U} \to [0,1]$ is a conditional density function in P. We refer to p_t as *flexibility feedback* sent at time $t \in [T]$ from the aggregator to the system operator. In this sense, (6) does not specify a specific aggregator function ψ_t, but a class of possible functions ψ_t. Every function in this collection is *causal* in that it depends only on information available to the aggregator at time t. In contrast to most aggregate flexibility notions in the literature [3], [4], [5], [6], [7], [8], [9], [10], the flexibility feedback here is specifically designed for an online feedback control setting.

B. Maximum entropy feedback

The intuition behind our proposal is using the conditional probability $p_t(u_t|u_{<t})$ to measure the resulting future flexibility of the aggregator if the system operator chooses u_t as the signal at time t, given the action trajectory up to time $t-1$. The sum of the conditional entropy of p_t thus is a measure of how informative the overall feedback is. This suggests choosing a conditional distribution p_t that maximizes its conditional entropy. Consider the optimization problem:

$$F \;:=\; \max_{p_1, \dots, p_T} \; \sum_{t=1}^{T} \mathbb{H}(U_t | U_{<t}) \;\text{ subject to } U \in \mathsf{S} \tag{7a}$$

where the variables are conditional density functions:

$$p_t \;:=\; p_t(\cdot|\cdot) := \mathbb{P}_{U_t|U_{<t}}(\cdot|\cdot), \qquad t \in [T], \tag{7b}$$

$U \in \mathsf{U}$ is a random variable distributed according to the joint distribution $\prod_{t=1}^{T} p_t$ and $\mathbb{H}(U_t|U_{<t})$ is the differential conditional entropy of p_t defined as:

$$\mathbb{H}(U_t|U_{<t}) := \int_{u_{\le t} \in \mathsf{U}^t} \Big(-\prod_{\ell=1}^{t} p_\ell(u_\ell|u_{<\ell}) \Big) \log p_t(u_t|u_{<t}) \mathrm{d}u_{\le t}. \tag{7c}$$

By definition, a quantity conditioned on "$u_{<1}$" means an unconditional quantity, so in the above, $\mathbb{H}(U_1|U_{<1}) := \mathbb{H}(U_1) := \mathbb{H}(p_1)$.

The chain rule shows that $\sum_{t=1}^{T} \mathbb{H}(U_t|U_{<t}) = \mathbb{H}(U)$. Hence (7) can be interpreted as maximizing the entropy $\mathbb{H}(U)$ of a random trajectory U sampled according to the joint distribution $\prod_{t=1}^{T} p_t$, conditioned on U satisfying $U \in \mathsf{S}$, where the maximization is over the collection of conditional distributions $(p_1, \dots, p_T)$.

***Definition* III.1** (Maximum entropy feedback)**.** *The flexibility feedback $p_t^* = \psi_t^*(u_{<t})$ for $t \in [T]$ is called the maximum entropy feedback (MEF) if $(p_1^*, \dots, p_T^*)$ is the unique optimal solution of* (7)*.*

***Remark* 3.** *Even though the optimization problem* (7) *involves variables p_t for the entire time horizon $[T]$, the individual variables p_t in* (7c) *are conditional probabilities that depend only on information available to the aggregator at times t. Therefore the maximum entropy feedback p_t^* in Definition III.1 is indeed causal and in the class of p_t^* defined in* (6)*. The existence of p_t^* is guaranteed by Lemma 2 below, which also implies that p_t^* is unique.*

We demonstrate Definition III.1 using a toy example.

***Example* III.1** (Maximum entropy feedback p^*)**.** *Consider the following instance of the EV charging example in Section II-E. Suppose the number of charging time slots is $T = 3$ and there is one customer, whose private vector is $(1,3,1,1)$ and possible energy levels are 0 (kWh) and 1 (kWh), i.e., $\mathsf{U} \equiv \{0,1\}$. Since there is only one EV, the scheduling algorithm u (disaggregation policy) assigns all power to this single EV. For this particular choices of x and u, the set of feasible trajectories is $\mathsf{S} = \{(0,0,1),(0,1,0),(1,0,0)\}$, shown in Figure 2 with the corresponding optimal conditional distributions given by* (7)*.*

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2021.3094719

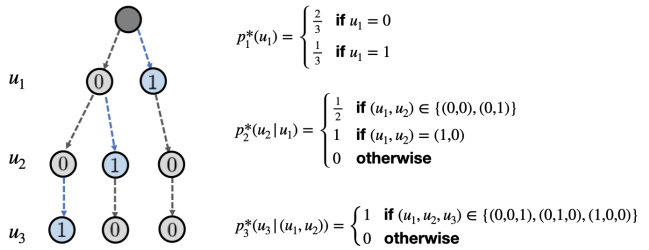


Fig. 2. Feasible trajectories of power signals and the computed maximum entropy feedback in Example III.1.

C. Properties of p_t^*

We now show that the proposed maximum entropy feedback p_t^* has several desirable properties. We start by computing p_t^* explicitly. Given any action trajectory $u_{\leq t}$, define the set of *subsequent* feasible trajectories as:

$$\mathsf{S}(u_{\leq t}) := \left\{ v_{>t} \in \mathsf{U}^{T-t} : v \text{ satisfies (2b)} - \text{(2d)}, v_{\leq t} = u_{\leq t} \right\}.$$

As a corollary The size $|\mathsf{S}(u_{\leq t})|$ of the set of subsequent feasible trajectories is a measure of future flexibility, conditioned on $u_{\leq t}$. Our first result justifies our calling p_t^* the optimal flexibility feedback: p_t^* is a measure of the future flexibility that will be enabled by the operator's action u_t and it attains a measure of system capacity for flexibility. By definition, $p_1^*(u_1|u_{<1}) := p_1^*(u_1)$.

***Lemma* 2.** *Let $\mu(\cdot)$ denote the Lebesgue measure. The MEF as optimal solutions of the maximization in (7a)-(7c) are given by*

$$p_t^*(u|u_{<t}) \equiv \frac{\mu\left(\mathsf{S}((u_{<t},u))\right)}{\mu\left(\mathsf{S}(u_{<t})\right)}, \quad \forall (u_{<t},u_t) \in \mathsf{U}^t. \tag{8}$$

Moreover, the optimal value of (7a)-(7c) *is equal to* $\log \mu(\mathsf{S})$.

***Remark* 4.** *When the denominator $\mu\left(\mathsf{S}(u_{<t})\right)$ is zero, the numerator $\mu\left(\mathsf{S}((u_{<t},u))\right)$ has also to be zero. For this case, we set $p_t^*(u|u_{<t}) = 0$ and this does not affect the optimality of* (7a)-(7b).

The proof can be found in Appendix B. The volume $\mu(\mathsf{S})$ is a measure of flexibility inherent in the aggregator. We will hence call $\log \mu(\mathsf{S})$ the *system capacity*. Lemma 2 then says that the optimal value of (7) is the system capacity, $F = \log \mu(\mathsf{S})$. Moreover the maximum entropy feedback $(p_1^*, \ldots, p_T^*)$ is the unique collection of conditional distributions that attains the system capacity in (7). This is intuitive since the entropy of a random trajectory x in S is maximized by the uniform distribution q^* in (21) induced by the conditional distributions $(p_1^*, \ldots, p_T^*)$.

Lemma 2 implies the following important properties of the maximum entropy feedback.

***Corollary* III.1** (Feasibility and flexibility)**.** *Let $p_t^* = p_t^*(\cdot|u_{<t})$ be the maximum entropy feedback at each time $t \in [T]$.*

1) For any action trajectory $u = (u_1, \ldots, u_T)$, if

$$p_t^*(u_t|u_{<t}) > 0 \quad \text{for all } t \in [T]$$

then $u \in \mathsf{S}$.

2) For all $u_t, u_t' \in \mathsf{U}$ at each time $t \in [T]$, if

$$p_t^*(u_t|u_{<t}) \geq p_t^*(u_t'|u_{<t})$$

then $\mu\left(|\mathsf{S}((u_{<t},u_t))\right) \geq \mu\left(\mathsf{S}((u_{<t},u_t'))\right)$.

The proof is provided in Appendix C. We elaborate the implication of Corollary III.1 for our online feedback-based solution approach.

***Remark* 5** (Feasibility and flexibility)**.** *Corollary III.1 says that the proposed optimal flexibility feedback p_t^* provides the right information for the system operator to choose its action u_t at time t.*

1) (Feasibility) Specifically, the first statement of the corollary says that if the operator always chooses an action u_t with positive conditional probability $p_t^*(u_t) > 0$ for each time t, then the resulting action trajectory is guaranteed to be feasible, $u \in \mathsf{S}$, i.e., the system will remain feasible at every *time $t \in [T]$ along the way.*

2) (Flexibility) Moreover, according to the second statement of the corollary, if the system operator chooses an action u_t with a larger $p_t^*(u_t)$ value at time t, then the system will be more flexible going forward than if it had chosen another signal u_t' with a smaller $p_t^*(u_t')$ value, in the sense that there are more feasible trajectories in $\mathsf{S}((u_{<t},u_t))$ going forward.

As noted in Remark 2, despite characterizations that involve the whole action trajectory u, such as $u \in \mathsf{S}$, these are *online* properties. This guarantees the feasibility of the online closed-loop control system depicted in Figure 1, and confirms the suitability of p_t^* for online applications.

IV. Approximating Maximum Entropy Feedback via Reinforcement Learning

For real-world applications, computing the maximum entropy feedback (MEF) could be computationally intensive. Thus, instead of computing it precisely, it is desirable to approximate it. In this section, we discuss the use of model-free reinforcement learning (RL) to generate an *aggregator function* ψ. For practical implementation, we switch to the case when U is a discrete set and reuse the notation P to denote a probability simplex that contains all possible discrete MEF:

$$\mathsf{P} := \left\{ p \in \mathbb{R}^{|\mathsf{U}|} : p(u) \geq 0, u \in \mathsf{U}; \sum_{u \in \mathsf{U}} p(u) = 1 \right\}. \tag{9}$$

We demonstrate that RL can be used to train a generator that outputs approximate MEF, given the state of the system. To be more precise, the learned aggregator function $\psi : \mathsf{X} \to \mathsf{P}$ outputs an estimate of the MEF given the state x_t at each time $t \in [T]$, where X is the state space and P is the set of all possible MEF. Note that the aggregator does not know the cost functions, so it cannot directly use an RL algorithm and transmit the learned Q-function or actor-critic model to the operator. Moreover, even if the aggregator knows the cost functions, generating actions using RL needs to solve two contradicting tasks of both optimizing rewards and penalizing

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2021.3094719

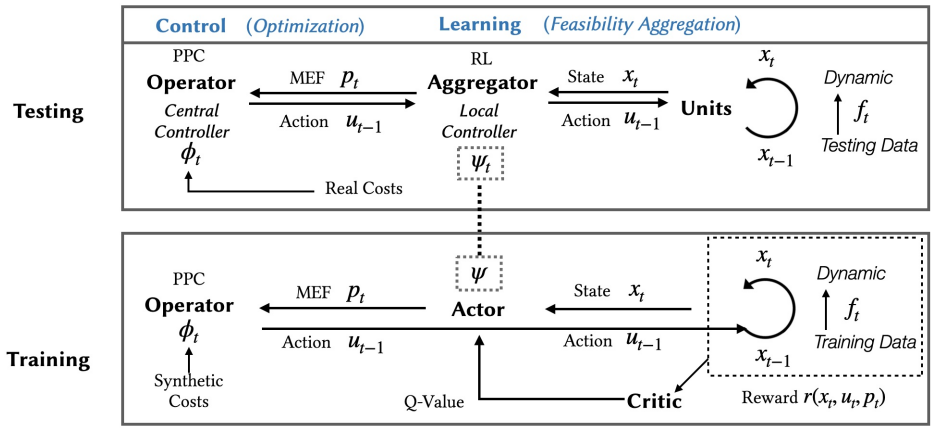


Fig. 3. Learning and testing architecture for learning aggregator functions.

feasibility violations, which makes the design of reward function and reward clipping a challenging goal. In our approach, we separate the tasks of enforcing feasibility and minimizing costs. We generate MEF as feasibility signals via reinforcement learning methods, and optimize the operator's objective via a MPC-based method (introduced in Section V-B). It is also worth noting that a number of effective heuristics may be available such as a greedy approximation in [22] and other gradient-based or density estimation [33] methods. We leave to future work the question of finding an optimal approximation algorithm.

A. Offline learning of aggregator functions

To learn an aggregator function ψ for estimating MEF, we use an actor-critic architecture [34] with separate policy and value function networks to enable the learning of policies on continuous action and state spaces. The actor-critic architecture is presented in Figure 3, which shows the information update between actor and critic networks. Note that in practical actor-critic algorithms, typically the policy, Q-function(s) and value function(s) are modeled using deep neural networks and the parameters are updated using policy iteration via stochastic gradient descent. We omit those details in Figure 3.

B. Training process

During the training process, the data used for defining training dynamics are the episodes $(\mathsf{U}_t, \mathsf{X}_t, f_t)_{t=1}^T$. For example, for the EV charging application in Section II-E, the training data of each episode (day) consist of historical private vectors $(a(j), d(j), e(j), r(j))$ specified by the users visited the charging station on the corresponding day. Among actor-critic-based RL algorithms, off-policy actor-critic methods, such as deep deterministic policy gradient (DDPG) [35] and soft actor-critic (SAC) [36] are known to attain better data efficiency in many applications. Below we take SAC, a maximum entropy deep RL algorithm, as an example to demonstrate the offline learning of an aggregator function ψ. In particular, for learning ψ, the objective of SAC is to maximize both the expected return and the expected entropy of the policy:

$$J(\psi) = \sum_{t=1}^{T} \mathbb{E}_{(\bar{x}_t, p_t)\sim\rho_\psi} \left[r(\bar{x}_t, p_t) + \alpha \mathbb{H}(\psi(\cdot|x_t)) \right] \tag{10}$$

where $\bar{x}_t := (x_t, u_t)$; ρ_ψ denotes the state-action marginals of the trajectory distribution induced by a policy ψ and $r(\bar{x}_t, p_t)$ is a customized reward function. To estimate MEF, we need to determine a reward function $r(\bar{x}_t, p_t)$ in (10). We adopt the following reward function that incorporates the constraints and the definition of MEF:

$$r(\bar{x}_t, p_t) = \mathbb{H}(p_t) + \sigma g(\bar{x}_t; \mathsf{X}_t, \mathsf{U}_t) \tag{11}$$

where the first term is critical and it maximizes the entropy of the probability distribution p_t, based the definition of the MEF in Definition III.1; $g(\bar{x}_t) = g(x_t, u_t)$ is a function that rewards the state and action if they satisfy the constraints $x_t \in \mathsf{X}_t$ and $u_t \in \mathsf{U}_t$. The reward function is independent of the cost functions, which are synthetic costs in the training stage. A concrete example of $g(\bar{x}_t)$ is given in Section VI. We clip the output MEF given by the policy to make sure it is a probability vector in the probability defined in simplex (9). In Figure 4, a training curve is given and it displays the changes of rewards regarding to the number of training episodes.

C. Testing process

With a trained aggregator function ψ that tries to optimize $J(\psi)$ in (10), we test the closed-loop system on new episodes defined by testing data, as shown in Figure 3. The trained aggregator function (parameterized by a deep neural network) is used as a "black box" function that maps each state x_t to feedback p_t.[2] Note that the real costs used in the testing process may not be same as the synthetic costs used in the training process, because the aggregator has no access to the costs as assumed in Section II.

In the sequel, with the learned MEF, we introduce a closed-loop framework that combines model predictive control (MPC)

[2]In our model, in general the aggregator functions $\psi_1, \ldots, \psi_T$ can be time-dependent. In the offline learning process presented in this section, we use a single function to generate feedback.

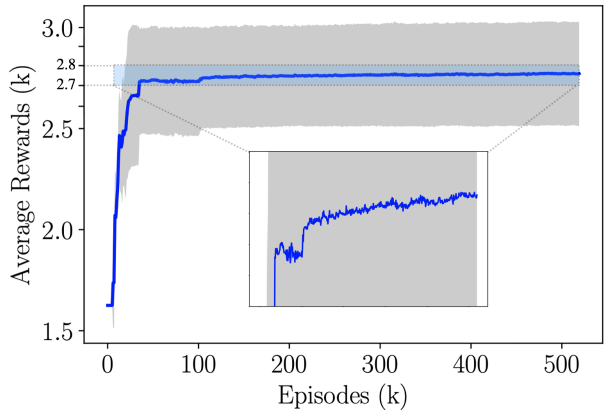


Fig. 4. Average rewards (defined in (11)) in the training stage with a tuning parameter $\beta = 6 \times 10^3$. Shadow region measures the variance.

and RL to coordinate a system operator and an aggregator in real-time. It is worth noting that the learned MEF may be different from the exact MEF provided in Definition III.1. However, later we show in Section VI that with the learned MEF, the constraints on the aggregator's side can almost be satisfied with a reasonable tuning parameter. In the EV charging example described in Section II-E, this means the EV's batteries are fully charged; see Figure 6 for details.

V. Penalized Predictive Control

Consider the system model in Section II. In this setting, the operator seeks to minimize the cost in an online manner, i.e., at time $t \in [T]$ the operator only knows the objective functions $c_1, \ldots, c_t$ and the flexibility feedback $p_1, \ldots, p_t$. The task of the operator is to, given the maximum entropy feedback, design a sequence of *operator functions* $\phi_1, \ldots, \phi_T$ to generate actions $u_1, \ldots, u_T$ that are always feasible with respect to the constraints *and* that minimize the cumulative cost.

A. Key Idea: Maximum entropy feedback as a penalty term

There is in general a trade-off between ensuring future flexibility and minimizing the current system cost in predictive control. The action u_t guaranteeing the maximal future flexibility, i.e., having the largest $p_t^*(u_t|u_{<t})$ may not be the one that minimizes the current cost function c_t and vice versa. Therefore, the online algorithm for the central controller must balance future flexibility and current cost. The key idea is to use MEF as a penalty term in the offline optimization problem. Note that Corollary III.1 guarantees that the online agent can always find a feasible action $u \in \mathsf{S}$. Indeed, knowing the MEF p_t^* for every $t \in [T]$ is equivalent to knowing the set of all admissible sequences of actions S. To see this, consider the unique maximum entropy feedback $(p_1^*, \ldots, p_T^*)$ guaranteed by Lemma 2 and let $q(u) = \prod_{t=1}^T p_t^*(u_t|u_{<t})$ denote the joint distribution of the action trajectory u. Then (8) implies that the joint distribution q is the uniform distribution over the set S of all feasible trajectories:

$$q(u) := \begin{cases} 1/\mu(\mathsf{S}) & \text{if } u \in \mathsf{S} \\ 0 & \text{otherwise} \end{cases}. \tag{12}$$

Data: Sequentially arrived cost functions and MEF
Result: Actions $u = (u_1, \ldots, u_T)$
for $t = 1, \ldots, T$ **do**
  Choose an action u_t by minimizing:

$$u_t = \phi_t(p_t) := \arg\inf_{u_t \in \mathsf{U}} (c_t(u_t) - \beta_t \log p_t(u_t|u_{<t})) \tag{14}$$

end
Return u;

Algorithm 2: Penalized Predictive Control (PPC).

Using this observation, the constraints (2b)-(2d) in the offline optimization can be rewritten as a penalty in the objective of (2a). We present a useful lemma that both motivates our online control algorithm and builds up the optimality analysis in Section V-D.

***Lemma* 3.** *The offline optimization* (2a)-(2d) *is equivalent to the following unconstrained minimization for any* $\beta > 0$*:*

$$\inf_{u \in \mathsf{U}^T} \sum_{t=1}^T (c_t(u_t) - \beta \log p_t^*(u_t|u_{<t})) \tag{13}$$

The proof of Lemma 3 can be found in Appendix E. It draws a clear connection between MEF and the offline optimal, which we exploit in the design of an online system operator in the next section.

B. Algorithm: Penalized Predictive Control via MEF

Our proposed design, termed penalized predictive control (PPC), is a combination of model predictive control (MPC) (*c.f.* [37]), which is a competitive policy for online optimization with predictions, and the idea of using MEF as a penalty term. This design makes a connection between the MEF and the well-known MPC scheme. The MEF as a feedback function, only contains limited information about the dynamical system in the local controller's side. (It contains only the feasibility information of the current and future time slots, as explained in Section III). The PPC scheme therefore is itself a novel contribution since it shows that, even if *only* feasibility information is available, it is still possible to incorporate the limited information to MPC as a *penalty term*.

We present PPC in Algorithm 2, where we use the following notation. Let $\beta_t > 0$ be a *tuning parameter* in predictive control to trade-off the flexibility in the future and minimization of the current system cost at each time $t \in [T]$. The next corollary follows whose proof is in Appendix C.

***Corollary* V.1** (Feasibility of PPC)**.** *When* $p_t = p_t^*$ *for all* $t \in [T]$*, the MEF defined in Definition III.1, the sequence of actions* $u = (u_1, \ldots, u_T)$ *generated by the PPC in* (14) *always satisfies* $u \in \mathsf{S}$ *for any sequence of tuning parameters* $(\beta_1, \ldots, \beta_T)$.

C. Framework: Closed-loop control between local and central controllers

Given the PPC scheme described above, we can now formally present our online control framework for the distant

central controller and local controller (defined in Section II). Recall that an overview of the closed-loop control framework has been given in Algorithm 1, where ϕ denotes an operator function and ψ is an aggregator function. To the best of our knowledge, this paper is the first to consider such a closed-loop control framework with limited information communicated in real-time between two geographically separate controllers seeking to solve an online control problem. We present the framework below.

At each time $t \in [T]$, the local controller first efficiently generates estimated MEF $p_t \in \mathsf{P}$ using an aggregator function ψ_t trained by a reinforcement learning algorithm. After receiving the current MEF p_t and cost function c_t (future w MEF and costs if predictions are available), the central controller uses the PPC scheme in Algorithm 2 to generate an action $u_t \in \mathsf{U}$ and sends it back to the local controller. The local controller then updates its state $x_t \in \mathsf{X}$ to a new state x_{t+1} based on the system dynamic in (1) and repeats this procedure again. In the next Section, we use an EV charging example to verify the efficacy of the proposed method.

D. Optimality Analysis

To end our discussion of PPC we focus on optimality. For the ease of analysis, we assume that the action space U is the set of real numbers $\mathbb{R}$; however, as noted in Remark 1, our system and the definition of MEF can also be made consistent with a discrete action space.

To understand the optimality of PPC we focus on standard regularity assumptions for the cost functions and the time-varying constraints. We assume cost functions are strictly convex and differentiable, which is common in practice. Further, let $\mu(\cdot)$ denote the Lebesgue measure. Note that the set of subsequent feasible action trajectories $\mathsf{S}(u_{\leq t})$ is Borel-measurable for all $t \in [T]$, implied by the proof of Corollary 1. We also assume that the measure of the set of feasible actions $\mu(\mathsf{S}(u_{\leq t}))$ is differentiable and strictly logarithmically-concave with respect to the subsequence of actions $u_t = (u_1, \ldots, u_t)$ for all $t \in [T]$, which is also common in practice, e.g., it holds in the case of inventory constraints $\sum_{t=1}^{T} \|u_t\|_2 \leq B$ with a budget $B > 0$. Finally, recall the definition of the set of subsequent feasible action trajectories:

$$\mathsf{S}(u_{\leq t}) := \Big\{ v_{>t} \in \mathsf{U}^{T-t} : v \text{ satisfies (2b)} - \text{(2d)}, v_{\leq t} = u_{\leq t} \Big\}.$$

Putting the above together, we can state our assumption formally as follows.

***Assumption* 2.** *The cost functions $c_t(u) : \mathbb{R} \to \mathbb{R}_+$ are differentiable and strictly convex. The mappings $\mu(\mathsf{S}(u_{\leq t})) : \mathbb{R}^t \to \mathbb{R}_+$ are differentiable and strictly logarithmically-concave.*

Given regularity of the cost functions and time-varying constraints, we can prove optimality of PPC.

***Theorem* V.1** (Existence of optimal actions)**.** *Let $\mathsf{U} = \mathbb{R}$. Under Assumption 1 and 2, there exists a sequence $b_1, \ldots, b_T$ such that implementing* (14) *with $\beta_t = b_t$ and $p_t = p_t^*$ at each time $t \in [T]$ ensures $u = (u_1^*, \ldots, u_T^*)$, i.e., the generated actions are optimal.*

Crucially, Theorem V.1 shows that there exists a sequence of "good" tuning parameters so that the PPC scheme is able to generate optimal actions under reasonable assumptions. However, note that the assumption of $\mathsf{U} = \mathbb{R}$ is fundamental. When the action space U is discrete or U is a high-dimensional space, it is impossible to generate the optimal actions because, in general, fixing t, the differential equations in the proof of Theorem V.1 (see Appendix F) do not have the same solution for all $\beta_t > 0$. Therefore a detailed regret analysis is necessary in such cases, which is a challenging task for future work.

VI. Numerical Results

In this section, we present experimental results for the case of online EV charging, introduced in Section II-E as an example of our system model (see Section II). The notation used in this section, if not defined, can be found in Section II-E.

A. Experimental setups

In the following, we present settings of parameters and useful metrics in our experiments.

1) Dataset and hardware: We use real EV charging data from ACN-Data [24], which is a dataset collected from adaptive EV charging networks (ACNs) at Caltech and JPL. The detailed hardware setup for that EV charging network structure can be found in [38].

2) Error Metrics: Recall the EV charging example in optimization (5a)-(5b). We first introduce two error metrics to measure the EV charging constraint violations. Note that the constraints (4a), (4b) and (4e) are hard constraints depending only on the scheduling policy, but not the actions and energy demands. Therefore they can be automatically satisfied in our experiments by fixing a scheduling policy satisfying them such as least laxity first. Violations may happen on constraint (4c) and (4d). To measure the violation of (4c), we use the (normalized) mean squared error (MSE) as the tracking error:

$$\mathsf{MSE} := \sum_{k=1}^{L} \sum_{t=1}^{T} \Big| \sum_{j=1}^{N} s_t^{(k)}(j) - u_t^{(k)} \Big|^2 / (L \times T \times \xi), \qquad (15)$$

where $u_t^{(k)}$ is the t-th power signal for the k-th test and $s_t^{(k)}(j)$ is the energy scheduled to the j-th charging session at time t for the k-th test. To better approximate real-world cases, we consider an additional *operational constraints* for the operator (central controller) and require that $u_t \leq \xi$ (kWh) for every $t \in [T]$. The total number of tests is L and the total number of charging sessions is N. Additionally, define the mean percentage error with respect to the undelivered energy corresponding to (4d) as

$$\mathsf{MPE} := 1 - \sum_{k=1}^{L} \sum_{t=1}^{T} \sum_{j=1}^{N} s_t^{(k)}(j) \Big/ \Big((L \times T) \cdot \sum_{j=1}^{N} e_j \Big), \qquad (16)$$

where e_j is the energy request for each charging session $j \in [N]$; $s_t^{(k)}(j)$ is the energy scheduled to the j-th charging session at time t for the k-th test.

3) Hyper-parameters: The detailed parameters used in our experiments are shown in Table I.

TABLE I
HYPER-PARAMETERS IN THE EXPERIMENTS.

Parameter	Value
System Operator	
Number of power levels $\lvert\mathsf{U}\rvert$	10
Cost functions $c_1,\ldots,c_T$	Average LMPs
Operator function ϕ	Penalized Predictive Control
Tuning parameter β	1×10^3 - 1×10^6
EV Charging Aggregator	
Number of Chargers W	54
State space X	$\mathbb{R}_+^{108}$
Action space	$[0,1]^{10}$
Time interval Δ	12 minutes
Private vector $(a(j),d(j),e(j),r(j))$	ACN-Data [24]
Power rating	150 kW
Scheduling algorithm π	Least Laxity First (LLF)
Laxity	$d_t(j)-e_t(j)/r(j)$
RL algorithm	Soft Actor-Critic (SAC) [36]
Optimizer	Adam [39]
Learning rate	$3\cdot10^{-4}$
Discount factor	0.5
Relay buffer size	10^6
Number of hidden layers	2
Number of hidden units per layer	256
Number of samples per minibatch	256
Non-linearity	ReLU
Reward function	$\sigma_1=0.1,\ \sigma_2=0.2,\ \sigma_3=2$
Temperature parameter	0.5

a) Control spaces: For the experimental results presented in this section, the control state space is $\mathsf{X}=\mathbb{R}_+^{2\times W}$ where W is the total number of charging stations and a state vector for each charging station is $(e_t,[d(j)-t]^+)$, i.e., the remaining energy to be charged and the remaining charging time if it is being used (see Section II-E); otherwise the vector is an all-zero vector. The control action space is $\mathsf{U}=\{0,15,30,\ldots,150\}$ (unit: kW) with $|\mathsf{U}|=10$, unless explicitly stated. The scheduling policy π is fixed to be least-laxity-first (LLF).

b) RL spaces: The RL action space[3] of the Markov decision process used in the RL algorithm is $[0,1]^{10}$. The outputs of the neural networks are clipped into the probability simplex (space of MEF) P afterwards.

c) RL rewards: We use the following specific reward function for our EV charging scenario, as a concrete example of (11):

$$\begin{aligned} r_{\mathrm{EV}}(\bar{x}_t,p_t) =& \mathbb{H}(p_t) \\ &+\sigma_1\sum_{i=1}^{N'}\|u_t(i)\|_2 \\ &-\sigma_2\sum_{i=1}^{N'}\left(\mathbf{I}(a(j_i)\le t\le a(j_i)+\Delta)\left[e(i)-\sum_{t=1}^{T}u_t(i)\right]_+\right) \\ &-\sigma_3\left|\phi_t(p_t)-\sum_{i=1}^{N'}u_t(j)\right| \end{aligned} \tag{17}$$

where σ_1,σ_2 and σ_3 are positive constants; N' is the number of EVs being charged; ϕ_t is the operator function, which is specified by (14); $\mathbf{I}(\cdot)$ denotes an indicator function and $a(j_i)$ is the arrival time of the $i-th$ EV in the charging station with j_i being the index of this EV in the total accepted charging sessions $[N]$. The entropy function $\mathbb{H}(p_t)$ in the first term is a greedy approximation of the definition of MEF (see Definition III.1). The second term is to further enhance charging performance and the last two terms are realizations of the last term in (11) for constraints (4c) and (4d). Note that The other constraints in the example shown in Section II-E can automatically be satisfied by enforcing the constraints in the fixed scheduling algorithm π. With the settings described above, in Figure 4 we show a typical training curve of the reward function in (17). We observe policy convergence with respect to a wide range of choices of the hyper-parameters σ_1,σ_2 and σ_3. In our experiments, we do not optimize them but fix the constants in (17) as $\sigma_1=0.1$, $\sigma_2=0.2$ and $\sigma_3=2$.

d) Cost functions: We consider the specific form of costs in (5a). In the RL training process, we train an aggregator function ψ using linear price functions $c_t=1-t/24$ where $t\in[0,24]$ (unit: Hrs) is the time index and we test the trained system with real price functions $c_1,\ldots,c_T$ being the average locational marginal prices (LMPs) on the CAISO (California Independent System Operator) day-ahead market in 2016 (depicted at the bottom of Figure 7).

e) Tuning parameters: In PPC defined in Algorithm 2, there is a sequence of tuning parameters $(\beta_1,\ldots,\beta_T)$. In our experiments, we fix $\beta_t=\beta$ for all $t\in[T]$ where $\beta>0$ is a universal tuning parameter that can be varied in our experiments.

B. Experimental results

1) Sensitivity of β: We first show how the changes of the tuning parameter β affect the total cost and feasibility. Figure 5 compares the results by varying β. The agents are trained on data collected from Nov. 1, 2018 to Dec. 1, 2019 and the tests are performed on data from Dec. 2, 2019 to Jan. 1, 2020 . Weekends and days with less than 30 charging sessions are removed from both training and testing data. For charging performance, we show in Figure 6 the battery states of each session after the charging cycle ends, tested with tuning parameters $\beta=2\times10^3,4\times10^3$ and 6×10^3 respectively. The results indicate that with a sufficiently large tuning parameter, the charging actions given by the PPC is able to satisfy EVs' charging demands and in practice, there is a trade-off between costs and feasibility depending on the choice of tuning parameters.

2) Charging curves: In Figure 7, substation charging rates (in kW) are shown. The charging rates generated by the PPC correspond to a trajectory $(\sum_j s_1(j)/\Delta,\ldots,\sum_j s_T(j))\Delta)$, which is the aggregate charging power given by the PPC for all EVs at each time $t=1,\ldots,T$. The agent is trained on data collected at Caltech from Nov. 1, 2018 to Dec. 1, 2019 and tested on Dec. 16, 2019 at Caltech using real LMPs on the CAISO day-ahead market in 2016. We use a tuning parameter $\beta=4\times10^3$ for both training and testing. The figure highlights that, with a suitable choice of tuning parameter, the operator is able to schedule charging at time slots where prices are lower and avoid charging at the peak of prices, as desired. In particular, it

[3]Note that the RL action space (consisting of p_t's) and state space (consisting of x_t's) referred here are the standard definitions in the context of RL and they are different from the "control action space" U and "control state space" X defined in Section II.

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2021.3094719

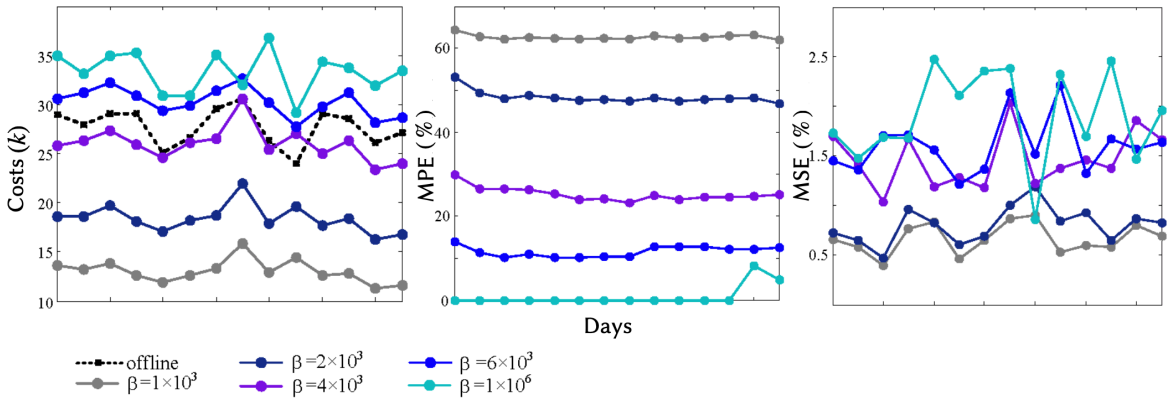


Fig. 5. Trade-offs of cost and charging performance. The dashed curve in the left figure corresponds to offline optimal cost. The tested days are selected (with no less than 30 charging sessions, i.e., $N \geq 30$) from Dec. 2, 2019 to Jan. 1, 2020.

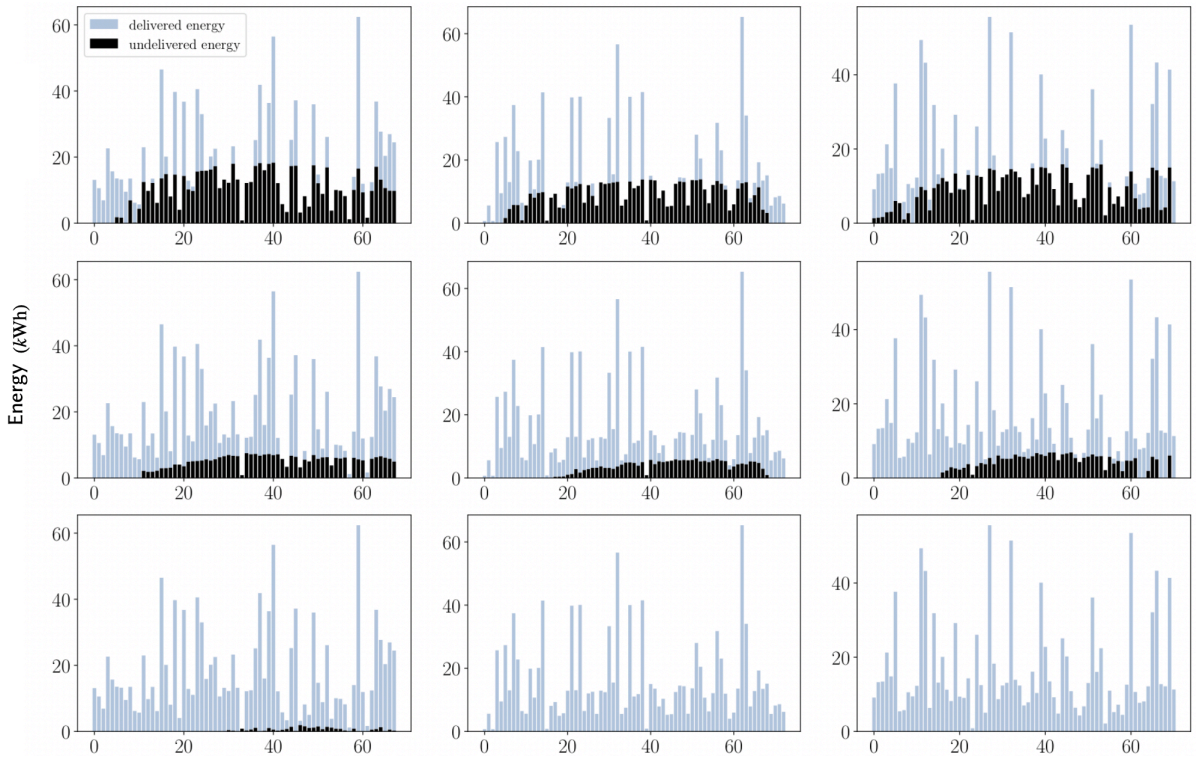


Fig. 6. Charging results of EVs controlled by PPC with tuning parameters $\beta = 2\times10^3$ (top), 4×10^3 (mid) and 6×10^3 (bot) for selected days (with no less than 30 charging sessions, i.e., $N \geq 30$) from Dec. 2, 2019 to Jan. 1, 2020. Each bar represents a charging session.

achieves a lower cost compared with the commonly used MPC scheme described in (18a)-(18f).The offline optimal charging rates are also provided.

3) Comparison of PPC and MPC: In Figure 8 we show the changes of the cumulative costs by varying the mean percentage error (MPE) with respect to the undelivered energy defined in (16). There are in total $K = 14$ episodes tested for days selected from Dec. 2, 2019 to Jan. 1, 2020 (days with less than 30 charging sessions are removed, i.e. we require, $N \geq 30$). Note that $0 \leq \mathsf{MPE} \leq 1$ and the larger MPE is, the higher level of constraint violations we observe. We allow constraint violations and modify parameters in the MPC and PPC to obtain varying MPE values. For the PPC, we vary the tuning parameter β to obtain the corresponding costs and MPE. For the MPC in our tests, we solve the following optimization at each time for obtaining the charging decisions $s_t = (s_t(1),\ldots,s_t(N'))$:

$$s_t = \arg\min_{s_t} \sum_{\tau=t}^{t'} c_\tau\Big(\sum_{i=1}^{N'} s_\tau(i)\Big) \text{ subject to :} \tag{18a}$$

$$s_\tau(i) = 0\ ,\ \tau < a(i),\ i = 1,\ldots,N', \tag{18b}$$

$$s_\tau(i) = 0\ ,\ \tau > d(i),\ i = 1,\ldots,N', \tag{18c}$$

$$\sum_{i=1}^{N'} s_\tau(i) = u_t,\ \tau = t,\ldots,t', \tag{18d}$$

$$\sum_{\tau=1}^{T} s_\tau(i) = \gamma\cdot e(i),\ i = 1,\ldots,N', \tag{18e}$$

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2021.3094719

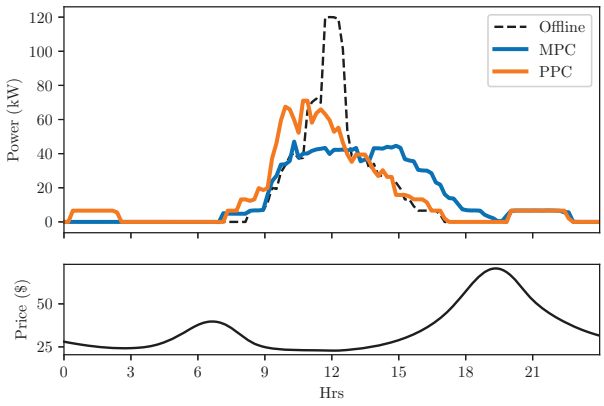


Fig. 7. Substation charging rates generated by the PPC (orange) in the closed-loop control shown in Algorithm 1, together with the MPC generated (blue) and global optimal (dashed black) charging rates.

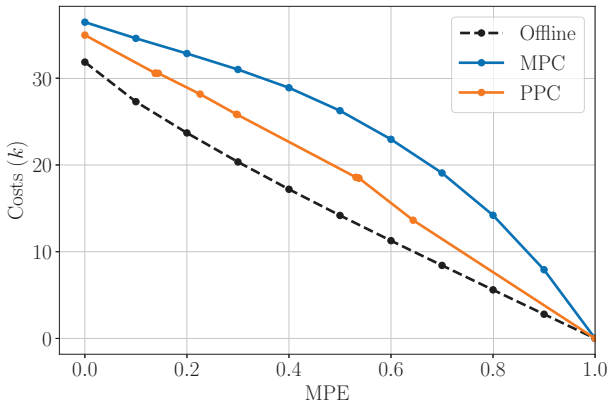


Fig. 8. Cost-energy curves for the offline optimization in (2a)-(2d) (for the example in Section II-E), MPC (defined in (18a)-(18f)) and PPC (introduced in Section V-B).

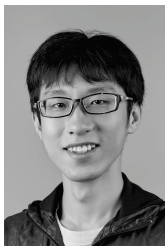
$$0 \le s_\tau(i) \le r(i),\ \tau = t,\dots,t' \tag{18f}$$

where at time t, the integer N' denotes the number of EVs being charged at the charging station and the time horizon of the online optimization is from $\tau = t$ to t', which is the latest departure time of the present charging sessions; $a(i)$ and $d(i)$ are the arrival time and departure time of the i-th session; $\gamma > 0$ relaxes the energy demand constraints and therefore changes the MPE region for MPC. The offline cost-energy curve is obtained by varying the energy demand constraints in (4d) in a similar way. We assume there is no admission control and an arriving EV will take a charger whenever it is idle for both MPC and PPC. Note that this MPC framework is widely studied [40] and used in EV charging applications [32]. It requires the precise knowledge of a 108-dimensional state vector of 54 chargers at each time step. We observe that with *only* feasibility information, PPC outperforms MPC for all $0 \le \text{MPE} \le 1$. The main reason that PPC outperforms vanilla MPC is that PPC utilizes MEF as its input, which is generated by a pre-trained aggregator function. Therefore the MEF may contain useful future feasibility information that vanilla MPC does not know, despite that it is trained and tested on separate datasets.

VII. Concluding Remarks

This paper formalizes and studies the closed-loop control framework created by the interaction between a system operator and an aggregator. Our focus is on the feedback signal provided by the aggregator to the operator that summarizes the real-time availability of flexibility among the loads controlled by the aggregator. We present the design of an maximum entropy feedback (MEF) signal based on entropic maximization. We prove a close connection between the MEF signal and the system capacity, and show that when the signal is used the system operator can perform online cost minimization while provably respecting the private constraints of the loads controlled by the aggregator and satisfying optimality under certain regularity assumptions. Further, we illustrate the effectiveness of these designs using simulation experiments of an EV charging facility.

There is much left to explore about this MEF signal presented in this work. In particular, computing it is computationally intensive and we use reinforcement learning for approximating the MEF. Improving the learning design and developing other approximations are of particular interest. Further, exploring the use of flexibility feedback for operational objectives beyond cost minimization and capacity estimation is an important goal. Finally, exploring the application of the defined real-time aggregate flexibility in other settings, such as multi-aggregator systems, frequency regulation and real-time pricing, is exciting.

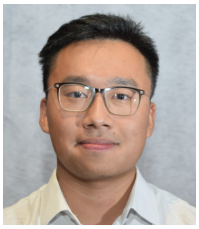


Tongxin Li is a doctoral student in Computing & Mathematical Sciences at the California Institute of Technology (Caltech). His research interests focus primarily on control, learning and optimization in cyber-physical systems and artificial intelligence techniques that impact the sustainability and resilience of smart grids. Prior to joining Caltech in 2017, he received a B.Sc. in mathematics, a B.Eng. in information engineering and an MPhil in information engineering from The Chinese University of Hong Kong.

Bo Sun received his B.E. in Electronic and Information Engineering from Harbin Institute of Technology, Harbin, China, in 2013, and his Ph.D. in Electronic and Computer Engineering from the Hong Kong University of Science and Technology (HKUST), Hong Kong, 2018. His research interests are on online optimization and stochastic models in real-world networked systems.

This article has been accepted for publication in IEEE Transactions on Smart Grid. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSG.2021.3094719

Yue Chen is an Assistant Professor with the Department of Mechanical and Automation Engineering, the Chinese University of Hong Kong. She received her B.E. and Ph.D. degrees in Electrical Engineering from Tsinghua University, Beijing, China, in 2015 and 2020, respectively; and her B.S. degree in Economics from Peking University, Beijing, China, in 2017. From 2018 to 2019, she was a visiting student at California Institute of Technology, Pasadena, CA, USA. Her research interests include optimization, game theory, mathematical economics, and their application in smart grid and integrated energy systems.



Z ixin Ye is a master student in Electrical Engineering at Caltech, USA. He is also an incoming PhD student at the University of Melbourne, Australia. His primary interest lies in network optimization and game theory for cyber-physical systems. Before joining Caltech, he received a B. Eng. in Electrical Engineering from the University of Queensland, Australia.

Steven H. Low is the Gilloon Professor of the Department of Computing & Mathematical Sciences and the Department of Electrical Engineering at Caltech and an honorary professor of the University of Melbourne, Australia. He has also held honorary and professorship in Australia, China and Taiwan. Before that, he was with AT&T Bell Laboratories, Murray Hill, NJ, and the University of Melbourne, Australia. He was a co-recipient of IEEE best paper awards and is an IEEE Fellow. His research on communication networks has been accelerating more than 1TB of Internet traffic every second since 2014. His research on smart electric vehicle charging is providing large scale, cost effective and sustainable EV charging, to workplaces, from high schools to Fortune Global 500 companies. He was a member of the Networking and Information Technology Technical Advisory Group for the US President's Council of Advisors on Science and Technology (PCAST) in 2006. He received his B.S. from Cornell and PhD from Berkeley, both in EE.

Adam Wierman is a Professor in the Department of Computing and Mathematical Sciences at the California Institute of Technology. He received his Ph.D., M.Sc. and B.Sc. in Computer Science from Carnegie Mellon University in 2007, 2004, and 2001, respectively, and has been a faculty at Caltech since 2007. He is a recipient of multiple awards, including the ACM SIGMETRICS Rising Star award, the IEEE Communications Society William R. Bennett Prize, and multiple teaching awards. He is a coauthor of papers that have received best paper awards at a wide variety of conferences across computer science, power engineering, and operations research including ACM Sigmetrics, IEEE INFOCOM, IFIP Performance, and IEEE PES.

REFERENCES

[1] D. S. Callaway and I. A. Hiskens, "Achieving controllability of electric loads," *Proceedings of the IEEE*, vol. 99, no. 1, pp. 184–199, 2010.

[2] S. Burger, J. P. Chaves-Ávila, C. Batlle, and I. J. Pérez-Arriaga, "A review of the value of aggregators in electricity systems," *Renewable and Sustainable Energy Reviews*, vol. 77, pp. 395–405, 2017.

[3] H. Hao and W. Chen, "Characterizing flexibility of an aggregation of deferrable loads," in *53rd IEEE Conference on Decision and Control*. IEEE, 2014, pp. 4059–4064.

[4] H. Hao, B. M. Sanandaji, K. Poolla, and T. L. Vincent, "Aggregate flexibility of thermostatically controlled loads," *IEEE Transactions on Power Systems*, vol. 30, no. 1, pp. 189–198, 2014.

[5] I. A. Sajjad, G. Chicco, and R. Napoli, "Definitions of demand flexibility for aggregate residential loads," *IEEE Transactions on Smart Grid*, vol. 7, no. 6, pp. 2633–2643, 2016.

[6] L. Zhao, W. Zhang, H. Hao, and K. Kalsi, "A geometric approach to aggregate flexibility modeling of thermostatically controlled loads," *IEEE Transactions on Power Systems*, vol. 32, no. 6, pp. 4721–4731, 2017.

[7] D. Madjidian, M. Roozbehani, and M. A. Dahleh, "Energy storage from aggregate deferrable demand: Fundamental trade-offs and scheduling policies," *IEEE Transactions on Power Systems*, vol. 33, no. 4, pp. 3573–3586, 2018.

[8] T. Chen, N. Li, and G. B. Giannakis, "Aggregating flexibility of heterogeneous energy resources in distribution networks," in *2018 Annual American Control Conference (ACC)*. IEEE, 2018, pp. 4604–4609.

[9] N. Sadeghianpourhamami, N. Refa, M. Strobbe, and C. Develder, "Quantitive analysis of electric vehicle flexibility: A data-driven approach," *International Journal of Electrical Power & Energy Systems*, vol. 95, pp. 451–462, 2018.

[10] M. P. Evans, S. H. Tindemans, and D. Angeli, "A graphical measure of aggregate flexibility for energy-constrained distributed resources," *IEEE Transactions on Smart Grid*, 2019.

[11] A. Bernstein, J.-Y. Le Boudec, M. Paolone, L. Reyes-Chamorro, and W. Saab, "Aggregation of power capabilities of heterogeneous resources for real-time control of power grids," in *2016 Power Systems Computation Conference (PSCC)*. IEEE, 2016, pp. 1–7.

[12] G. Wenzel, M. Negrete-Pincetic, D. E. Olivares, J. MacDonald, and D. S. Callaway, "Real-time charging strategies for an electric vehicle aggregator to provide ancillary services," *IEEE Transactions on Smart Grid*, vol. 9, no. 5, pp. 5141–5151, 2017.

[13] H. Hao, Y. Lin, A. S. Kowli, P. Barooah, and S. Meyn, "Ancillary service to the grid through control of fans in commercial building hvac systems," *IEEE Transactions on smart grid*, vol. 5, no. 4, pp. 2066–2074, 2014.

[14] T. Wei, Q. Zhu, and N. Yu, "Proactive demand participation of smart buildings in smart grid," *IEEE Transactions on Computers*, vol. 65, no. 5, pp. 1392–1406, 2015.

[15] S. P. Meyn, P. Barooah, A. Bušić, Y. Chen, and J. Ehren, "Ancillary service to the grid using intelligent deferrable loads," *IEEE Transactions on Automatic Control*, vol. 60, no. 11, pp. 2847–2862, 2015.

[16] A. Subramanian, M. J. Garcia, D. S. Callaway, K. Poolla, and P. Varaiya, "Real-time scheduling of distributed resources," *IEEE Transactions on Smart Grid*, vol. 4, no. 4, pp. 2122–2130, 2013.

[17] D. Papadaskalopoulos, G. Strbac, P. Mancarella, M. Aunedi, and V. Stanojevic, "Decentralized participation of flexible demand in electricity markets—part ii: Application with electric vehicles and heat pump systems," *IEEE Transactions on Power Systems*, vol. 28, no. 4, pp. 3667–3674, 2013.

[18] S. Wang, S. Bi, and Y. J. A. Zhang, "Reinforcement learning for real-time pricing and scheduling control in ev charging stations," *IEEE Transactions on Industrial Informatics*, 2019.

[19] Y. Wang, X. Lin, and M. Pedram, "A near-optimal model-based control algorithm for households equipped with residential photovoltaic power generation and energy storage systems," *IEEE Transactions on Sustainable Energy*, vol. 7, no. 1, pp. 77–86, 2015.

[20] B. J. Claessens, D. Vanhoudt, J. Desmedt, and F. Ruelens, "Model-free control of thermostatically controlled loads connected to a district heating network," *Energy and Buildings*, vol. 159, pp. 1–10, 2018.

[21] B. Chen, W. Yao, J. Francis, and M. Bergés, "Learning a distributed control scheme for demand flexibility in thermostatically controlled loads," in *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. IEEE, 2020, pp. 1–7.

[22] T. Li, S. H. Low, and A. Wierman, "Real-time flexibility feedback for closed-loop aggregator and system operator coordination," in *Proceed-*

ings of the Eleventh ACM International Conference on Future Energy Systems, 2020, pp. 279–292.

[23] T. Li, Y. Chen, B. Sun, A. Wierman, and S. H. Low, "Information aggregation for constrained online control," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 5, no. 2, pp. 1–35, 2021.

[24] Z. J. Lee, T. Li, and S. H. Low, "Acn-data: Analysis and applications of an open ev charging dataset," in *Proceedings of the Tenth ACM International Conference on Future Energy Systems*. ACM, 2019, pp. 139–149.

[25] Z. Ma, D. S. Callaway, and I. A. Hiskens, "Decentralized charging control of large populations of plug-in electric vehicles," *IEEE Transactions on control systems technology*, vol. 21, no. 1, pp. 67–78, 2011.

[26] L. Gan, U. Topcu, and S. H. Low, "Optimal decentralized protocol for electric vehicle charging," *IEEE Transactions on Power Systems*, vol. 28, no. 2, pp. 940–951, 2012.

[27] A. Subramanian, M. Garcia, A. Dominguez-Garcia, D. Callaway, K. Poolla, and P. Varaiya, "Real-time scheduling of deferrable electric loads," in *2012 American Control Conference (ACC)*. IEEE, 2012, pp. 3643–3650.

[28] E. C. Kara, J. S. Macdonald, D. Black, M. Bérges, G. Hug, and S. Kiliccote, "Estimating the benefits of electric vehicle smart charging at non-residential locations: A data-driven approach," *Applied Energy*, vol. 155, pp. 515–525, 2015.

[29] X. Chen, E. Dall'Anese, C. Zhao, and N. Li, "Aggregate power flexibility in unbalanced distribution systems," *IEEE Transactions on Smart Grid*, vol. 11, no. 1, pp. 258–269, 2019.

[30] M. Marzband, A. Sumper, J. L. Domínguez-García, and R. Gumara-Ferret, "Experimental validation of a real time energy management system for microgrids in islanded mode using a local day-ahead electricity market and minlp," *Energy Conversion and Management*, vol. 76, pp. 314–322, 2013.

[31] P. Siano and D. Sarno, "Assessing the benefits of residential demand response in a real time distribution energy market," *Applied Energy*, vol. 161, pp. 533–551, 2016.

[32] Z. J. Lee, D. Chang, C. Jin, G. S. Lee, R. Lee, T. Lee, and S. H. Low, "Large-scale adaptive electric vehicle charging," in *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. IEEE, 2018, pp. 1–7.

[33] S. Singh, A. Uppal, B. Li, C.-L. Li, M. Zaheer, and B. Póczos, "Nonparametric density estimation under adversarial losses," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 225–10 236.

[34] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE transactions on systems, man, and cybernetics*, no. 5, pp. 834–846, 1983.

[35] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[36] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[37] E. F. Camacho and C. B. Alba, *Model predictive control*. Springer Science & Business Media, 2013.

[38] Z. J. Lee, G. Lee, T. Lee, C. Jin, R. Lee, Z. Low, D. Chang, C. Ortega, and S. H. Low, "Adaptive charging networks: A framework for smart electric vehicle charging," *arXiv preprint arXiv:2012.02636*, 2020.

[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[40] U. Rosolia, X. Zhang, and F. Borrelli, "Data-driven predictive control for autonomous systems," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 259–286, 2018.

APPENDIX A
PROOF OF LEMMA 1

Proof. We first define a set $f^{-1}(\mathsf{X}_t)$ denoting the inverse image of the set X_t for actions: $f^{-1}(\mathsf{X}_t)(u_{<t}) := \{u \in \mathsf{U} : f(x_{t-1}, u) \in \mathsf{X}_t\}$. The inverse image $f^{-1}(\mathsf{X}_t)$ depends only on the past actions $u_{<t}$ since the states $x_{<t}$ are determined by $u_{<t}$ and a pre-fixed initial state x_1 via the dynamics in (1). Note that X_t and the dynamic f are Borel measurable. Therefore the inverse image $f^{-1}(\mathsf{X}_t)$ is also a Borel set, implying that the intersection $\mathsf{U}_t \bigcap f^{-1}(\mathsf{X}_t)$ is also Borel measurable. The set of feasible action trajectories S can be reprised as

$$\mathsf{S} := \left\{ u \in \mathsf{U}^T : u_t \in \mathsf{U}_t \bigcap f^{-1}(\mathsf{X}_t)(u_{<t}), \forall t \in [T] \right\},$$

which is a Borel measurable set of all *feasible* sequences of actions. □

APPENDIX B
PROOF OF LEMMA 2

Proof of Lemma 2. We prove the statement by induction. It is straightforward to verify the results hold when $T = 1$. We suppose the lemma is true when $T = m$. Suppose $T = m+1$. Let

$$F(u) := \max_{p_2,\ldots,p_T} \sum_{t=2}^{T} \mathbb{H}(U_t | \mathbf{U}_{2:t-1}; U_1 = u)$$

denote the optimal value corresponding to the time horizon $t \in [T]$, given the first action $U_1 = u$. By the definition of conditional entropy, we have

$$F = \max_{p_1} \int_{u \in \mathsf{U}} p_1(u) F(u) \mathrm{d}u + \mathbb{H}(p_1).$$

By the induction hypothesis, $F(u) = \mu(\mathsf{S}(u))$. Therefore,

$$\begin{aligned} F &= \max_{p_1} \int_{u \in \mathsf{U}} p_1(u) \log \mu(\mathsf{S}(u)) \, \mathrm{d}u + \mathbb{H}(p_1) \\ &= \max_{p_1} \int_{u \in \mathsf{U}} p_1(u) \log \left(\frac{\mu(\mathsf{S}(u))}{p_1(u)} \right) \mathrm{d}u \end{aligned}$$

whose optimizer p_1^* satisfies (8) and we get $F = \mu(\mathsf{S})$. The lemma follows by finding the optimal conditional distributions $p_1^*, \ldots, p_T^*$ inductively. □

APPENDIX C
PROOF OF COROLLARY III.1

Proof of Corollary III.1. Lemma 2 shows that the value of the density function corresponding to choosing $u_t = u$ in the MEF is proportional to the measure of $\mathsf{S}((u_{<t}, u))$, completing the proof of interpretability. According to the explicit expression in (8) of the MEF, the selected action u always ensures that $\mu((\mathsf{S}((u_{<t}, u))) > 0$ and therefore the set $\mathsf{S}((u_{<t}, u)$ is non-empty. This guarantees that the generated sequence u is always in S. □

APPENDIX D
PROOF OF COROLLARY V.1

Proof of Corollary V.1. The explicit expression in Lemma 2 ensures that whenever $p_t^*(u_t | u_{<t}) > 0$, then there is always a feasible sequence of actions in $\mathsf{S}(u_{<t})$. Now, if the tuning parameter $\beta_t > 0$, then the optimization (14) guarantees that $p_t^*(u_t | u_{<t}) > 0$ for all $t \in [T]$; otherwise, the objective value in (14) is unbounded. Corollary III.1 guarantees that for any sequence of actions $u = (u_1, \ldots, u_T)$, if $p_t^*(u_t | u_{<t}) > 0$ for all $t \in [T]$, then $u \in \mathsf{S}$. Therefore, the sequence of actions u given by the PPC is always feasible. □

$$u_t^* = \arg\min_{u_t \in \mathsf{U}} \left(c_t(u_t) + \underbrace{\min_{u_{t+1:T}} \left(\sum_{\tau=t+1}^{T} c(u_\tau) - \log \mu\left(\mathsf{S}(u^*_{t-k:t-1}, u_{t:T})\right) \right)}_{f(u_t)} \right). \tag{19}$$

APPENDIX E
PROOF OF THEOREM 3

Proof of Lemma 3. We note that the offline optimization (2a)-(2d) is equivalent to

$$\inf_{u \in \mathsf{U}^T} \sum_{t=1}^{T} c_t(u_t) - \beta \log q(u) \tag{20}$$

for any $\beta > 0$ and $q(u)$ is a uniform distribution on S:

$$q(u) := \begin{cases} 1/\mu(\mathsf{S}) & \text{if } u \in \mathsf{S} \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

where $\mu(\cdot)$ is the Lebesgue measure. Further, decomposing the joint distribution $q(u) = \prod_{t=1}^{T} p_t^*(u_t|u_{<t})$ into the conditional distributions given by (7a)-(7c), the objective function (20) becomes

$$\begin{aligned} &\sum_{t=1}^{T} c_t(u_t) - \beta \log \left(\prod_{t=1}^{T} p_t^*(u_t|u_{<t}) \right) \\ =&\sum_{t=1}^{T} \left(c_t(u_t) - \beta \log p_t^*(u_t|u_{<t}) \right), \end{aligned}$$

which implies the lemma. □

APPENDIX F
PROOF OF THEOREM V.1

Proof. Define the following optimal *cost-to-go function*, which computes the minimal cost given a subsequence of actions:

$$\begin{aligned} V_t^{\mathsf{OPT}}(u_{<t}) :=& \min_{u_{t:T} \in \mathsf{U}^{T-t+1}} \left(\sum_{\tau=t}^{T} c(u_\tau) - \beta \log p_t^*(u_{t:T}|u^*_{t-k:t-1}) \right) \\ =& \min_{u_t \in \mathsf{U}} \Bigg(c(u_t) - \log p_t^*(u_t|u^*_{t-k:t-1}) + \\ & \min_{u_{t+1:T} \in \mathsf{U}^{T-t}} \left(\sum_{\tau=t+1}^{T} c(u_\tau) - \log p_{t+1}^*(u_{t+1:T}|u^*_{t-k:t}) \right) \Bigg) \\ =& \min_{u_t \in \mathsf{U}} \left(c(u_t) - \log p_t^*(u_t|u^*_{t-k:t-1}) + V_{t+1}^{\mathsf{OPT}}(u_{\le t}) \right). \end{aligned}$$

Let $\mu(\cdot)$ denote the Lebesgue measure. Based on the definition of the optimal cost-to-go functions defined above and applying Lemma 3, we obtain the following expression of the optimal action u_t^* at each time $t \in [T]$:

$$\begin{aligned} u_t^* =& \arg\min_{u_t \in \mathsf{U}} \left(c_t(u_t) - \beta \log p_t^*(u_t|u^*_{t-k:t-1}) + V_{t+1}^{\mathsf{OPT}}(u_{t-k+1:t}) \right) \\ =& \arg\min_{u_t \in \mathsf{U}} \Bigg(c_t(u_t) - \beta \log \frac{\mu\left(\mathsf{S}(u^*_{t-k:t-1}, u_t)\right)}{\mu\left(\mathsf{S}(u^*_{t-k:t-1})\right)} \\ & + \min_{u_{t+1:T}} \Big(\sum_{\tau=t+1}^{T} \Big(c(u_\tau) - \beta \log \frac{\mu\left(\mathsf{S}(u^*_{t-k:t-1}, u_{t:\tau})\right)}{\mu\left(\mathsf{S}(u^*_{t-k:t-1}, u_{t:\tau-1})\right)} \Big) \Big) \Bigg) \\ =& \arg\min_{u_t \in \mathsf{U}} \Bigg(c_t(u_t) - \beta \log \frac{\mu\left(\mathsf{S}(u^*_{t-k:t-1}, u_t)\right)}{\mu\left(\mathsf{S}(u^*_{t-k:t-1})\right)} \\ & + \min_{u_{t+1:T}} \Big(\sum_{\tau=t+1}^{T} c(u_\tau) + \beta \log \frac{\mu\left(\mathsf{S}(u^*_{t-k:t-1}, u_t)\right)}{\mu\left(\mathsf{S}(u^*_{t-k:t-1}, u_{t:T})\right)} \Big) \Bigg), \end{aligned}$$

which implies (19) and when $u_{<t} = u^*_{<t}$, the solution of the PPC in Algorithm 2 satisfies

$$\begin{aligned} u_t =& \arg\min_{u_t \in \mathsf{U}} \left(c_t(u_t) - \beta \log p_t(u_t|u^*_{t-k:t-1}) \right) \\ =& \arg\min_{u_t \in \mathsf{U}} \left(c_t(u_t) - \beta \log \frac{\mu\left(\mathsf{S}(u^*_{t-k:t-1}, u_t)\right)}{\mu\left(\mathsf{S}(u^*_{t-k:t-1})\right)} \right) \\ =& \arg\min_{u_t \in \mathsf{U}} \left(c_t(u_t) + \beta \underbrace{\log\left(1/\mu\left(\mathsf{S}(u^*_{t-k:t-1}, u_t)\right)\right)}_{g(u_t)} \right). \end{aligned} \tag{22}$$

Since the cost functions $c_t(u)$ and the measure $\log(1/\mu(\mathsf{S}(u)))$ are strictly convex, the inner minimization in (19) is a convex minimization and hence $f(u_t)$ is convex. Therefore, u_t^* in (19) is unique. Denoting by c' and f' the corresponding derivatives of a given cost function c and the function f defined in (19), we have

$$c_t'(u_t^*) + f'(u_t^*) = 0.$$

Furthermore, the unique solution of the PPC scheme satisfies

$$c_t'(u_t) + \beta g'(u_t) = 0$$

where g' is the derivative of the function g defined in (22). Choosing $\beta = b_t = f'(u_t^*)/g'(u_t^*)$ implies that $u_t = u_t^*$ for all $t \in [T]$. □