Digital Object Identifier: 10.1109/TSG.2020.3014055

An Edge-Cloud Integrated Solution for Buildings Demand Response Using Reinforcement Learning

Xiangyu Zhang, *Member, IEEE*, Dave Biagioni, *Member, IEEE*, Mengmeng Cai, *Student Member, IEEE*, Peter Graf, *Member, IEEE*, and Saifur Rahman, *Life Fellow, IEEE*

***Abstract*—Buildings, as major energy consumers, can provide great untapped demand response (DR) resources for grid services. However, their participation remains low in real-life. One major impediment for popularizing DR in buildings is the lack of cost-effective automation systems that can be widely adopted. Existing optimization-based smart building control algorithms suffer from high costs on both building-specific modeling and on-demand computing resources. To tackle these issues, this paper proposes a cost-effective edge-cloud integrated solution using reinforcement learning (RL). Beside RL's ability to solve sequential optimal decision-making problems, its adaptability to easy-to-obtain building models and the off-line learning feature are likely to reduce the controller's implementation cost. Using a surrogate building model learned automatically from building operation data, an RL agent learns an optimal control policy on cloud infrastructure, and the policy is then distributed to edge devices for execution. Simulation results demonstrate the control efficacy and the learning efficiency in buildings of different sizes. A preliminary cost analysis on a 4-zone commercial building shows the annual cost for optimal policy training is only 2.25% of the DR incentive received. Results of this study show a possible approach with higher return on investment for buildings to participate in DR programs.**

***Index Terms*—Demand Response, Reinforcement Learning, Smart Building, Air-Conditioning, Cloud computing**

NOMENCLATURE

General Math and Indices

$\mathbb{N}$	Gaussian Distribution
$i \in \mathcal{N}$	Index of building thermal zones and its complete set, used as superscript for building related variables
N	Cardinality of $\mathcal{N}$
$p(A)$	Probability of event A
$t \in \mathcal{T}$	Index of control steps and its complete set, used as subscript for building related variables

Building Control Related

η	Binary flag indicating if energy saving is considered
$\mathcal{C}_d, \mathcal{C}_e$	Cost function for building discomfort and energy consumption
$\mathcal{E}_t^i$	Air-conditioning unit energy consumption for Zone i at t
$\mathcal{F}^i$	Thermal dynamics model for Zone i
$\mathcal{P}$	Demand response power limit in kW
σ	Standard deviation of building thermal model temperature prediction error
τ^i	Maximum tolerable temperature for Zone i
P^i	Rated power for the air-conditioning unit for Zone i in kW. $\mathbf{P}$ is the vector form for P^i of all zones
T_t^i	Indoor temperature for Zone i at t
T_{eco}^i	Economic cooling temperature for Zone i, below which cooling is discouraged. $\mathbf{T_{eco}}$ is the vector format for T_{eco}^i of all zones
u_t^i	Binary control signal for the air-conditioning unit for Zone i at t

RL Related

γ	Discount factor for the Markov decision process
λ^i	Coefficient to map energy consumption to equivalent thermal comfort for Zone i
$\mathbf{a_t}, \mathcal{A}$	Action chose by RL agent at step t and the complete legal action set. There is $\mathbf{a_t} = [u_t^1, u_t^2, ..., u_t^N] \in \mathcal{A}$
$\mathbf{s_t}$	State at step t for the formulated Markov decision process
π	RL control policy, which helps RL agent determines action at each step. Specifically, π^* is used to present an optimal control policy
φ^i	Thermal comfort margin function for Zone i
ξ^i	Zone priority factor for Zone i
e_t^i	Penalty for energy consumption for Zone i at t
$f(x)$	Non-linear function used in φ^i, $f(x) = \beta^{-x}$ is used in this study.
m_t^i	Thermal comfort margin improvement for Zone i at t
r_t	Total reward at t
v_t^i	Penalty for violating temperature comfort upper bound for Zone i at t

X. Zhang, D. Biagioni and P. Graf are with the Computational Science Center in U.S. National Renewable Energy Laboratory, Golden, CO, 80401 USA. (e-mail: {Xiangyu.Zhang, Dave.Biagioni, Peter.Graf}@nrel.gov)

M. Cai and S. Rahman are with the Virginia Tech Advanced Research Institute, Arlington, VA, 22203 USA. (e-mail:{mmcai, srahman}@vt.edu)

This work was authored in part by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. Funding provided by the Assessment of Reinforcement Learning for Model NREL Problems Project funded by the National Renewable Energy Laboratory's Laboratory Directed Research and Development program. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.

I. INTRODUCTION

WITH the development of Smart Grid and Internet of Things (IoT) technologies, implementing smart building controls to provide grid services has become feasible recently. Buildings, from both residential and commercial sectors, accounted for around 40% of total U.S. energy consumption in 2018 [1]. When proper control is widely adopted,

buildings can provide precious untapped potential for supporting the grid, solving the "Last Mile" problem in the smart grid paradigm. Demand response (DR) is a common approach to incentivize buildings to be grid responsive. But automated building modeling and control that facilitates buildings to participate in DR events are challenging.

In reality, the most commonly used method for making buildings demand responsive is direct load control (DLC) [2], [3]. Via remotely controlled relay devices, large appliances in buildings are directly controlled by the utility. This open-loop control, though being beneficial to the grid [4], may cause building operational issues. To implement a more building-centric control, Model Predictive Control (MPC), which repeatedly solves an optimal control problem over a look-ahead horizon, has become the state-of-the-art. Research on MPC for building control has been popular since the early 2010s, though at that time being grid-interactive was not yet included in control objectives. Ma *et al.* [5] propose using MPC on building cooling systems with thermal storage by leveraging predictive knowledge. Similarly, [6] tests a MPC controller in a real building and achieves a saving of 17%-24%. Later, many MPC-based variants were also developed: Cai *et al.* [7] propose a distributed MPC in a multi-agent control framework using the Alternating Direction Method of Multipliers (ADMM) to solve the optimal control problem. To minimize the error brought by simplified building models, Chen *et al.* [8] propose non-linear MPC using a trained input-convex neural network to represent the building model. To introduce grid-interactive efficient buildings (GEB), [9] uses MPC for building air-conditioning (AC) systems coordination and control in order to provide grid frequency regulation, and [10] implements a DR power limited coordinated control for multiple AC units. Although MPC has become the workhorse for optimal control, there are a few drawbacks when using it in buildings: 1) Alessandro *et al.* [11] point out that traditional MPC suffers from a main drawback of "need to solve a mathematical program *online* to compute the control action". Indeed, with MPC, optimization problems are solved repeatedly on-the-fly, which makes MPC unlikely to be implemented using edge devices (i.e., cost-effective embedded systems with limited computing power) [12]. This means building owners need to pay for expensive real-time computing resources (i.e., a powerful computer) and the license fee for a commercial solver. 2) In addition to high implementation and maintenance cost, MPC requires a building model to be relatively simple (e.g., linear or at least convex) so that the optimization problem can be solved efficiently within control intervals. However, this model simplification, inevitably, will make the control performance sub-optimal. Additionally, obtaining such an accurate but simple model requires high modeling labor cost since domain expertise is required to acquire a model and simplify it appropriately, especially when considering massive real-life adoption in buildings and each building needs customization. As a result, this paper focuses on searching for a more cost-effective alternative, which possess the merits of a simple controller (e.g., PID controller) concerning cost and implementation but also resembles MPC in handling constraints and taking optimization into consideration [13].

Due to many breakthroughs in the early 2010s, deep learning has thrived recently and thus many applications to energy-related problems have appeared. In various aspects of building operation, deep learning approaches including recurrent neural network (RNN), convolutional neural network (CNN) and extreme deep learning approaches are used and compared with traditional time-series forecasting methods for their performance on building load prediction [14]–[17]. Additionally, reinforcement learning (RL) has been used for designing optimal controllers due to its outstanding performance for solving sequential decision-making optimal control problems, such as storage arbitrage [18], emergency control [19], electricity market bidding [20] and more. In contrast to MPC, RL does not require on-demand computation and is capable of taking non-linear models and environment uncertainty into consideration easily. Due to these merits, using RL for building control has gradually become popular. Previously on using RL for building control, Wen *et al.* [21] propose using RL to solve an energy management system rescheduling problem at device-level and Mocanu *et al.* [22] demonstrate using deep Q-network (DQN) and deep policy gradient (DPG) for optimizing building energy. Although these existing works successfully demonstrate the feasibility of using RL for building control problems, how to obtain the building/device model and dynamics in real life are not clear. Theoretically, it is possible to train an RL optimal control policy by directly controlling real buildings and thus building models are not needed. Nonetheless, this is undesirable due to the poor control policy and aggressive exploration behavior at the early learning phase. In addition, learning from real time environments might take three years of training data for an RL controller to outperform a rule-based controller according to [23]. As a result, most existing works requires a surrogate simulated model. To obtain such a model, in [24], a high fidelity EnergyPlus model is used to train a control policy for the AC system. However, in real-life applications, such a model that precisely represents the building thermal dynamics is not easily accessible due to the high modeling cost. To tackle this dilemma, Zhang *et al.* [25] propose using a crude EnergyPlus model from the building design phase to pre-train an RL agent and then deploy it in the real building for continuing learning. But this does not work for legacy buildings, which account for the vast majority, because there are not corresponding EnergyPlus models for these existing buildings that are not recently built.

To sum up, though successful stories of building RL controllers exist, there are still the following knowledge gaps:

a) The majority of existing works, including some high impact papers published in recent years [23], [26], [27], still use basic RL algorithms such as tabular Q-learning, which might not represent the state-of-the-art RL algorithms and could not reflect an up-to-date learning cost.

b) Most of existing studies assume there is a model for the building already, either a theoretical model or a model requiring domain expertise to build. How to obtain such a model in real life considering massive building deployment is not addressed.

c) Most intelligent control algorithms are targeting recently built large buildings, and there are few control algorithms

designed for small and medium sized commercial buildings (referred as SMCB hereinafter. E.g., small office buildings, doctor's clinic, etc.) [28], even though they account for 90% of total commercial buildings in the U.S. [29].

Therefore, to address the above-mentioned issues, this paper proposes an edge/cloud integrated RL-based solution that makes intelligent DR control in SMCB affordable to most building owners due to its low commission and system maintenance cost. By leveraging the state-of-the-art scalable RL algorithms and cloud computing, we conduct a techno-economic analysis of the proposed framework, demonstrating the control efficacy and training cost of the proposed controller.

The contributions of this paper are summarized as follows:

1) A building DR control framework that connects utility company, building and cloud service provider in a synergistic manner is proposed. With the framework commissioned, building model learning, RL controller training and real-time RL control execution can be streamlined and fully automated with little human configuration in an end-to-end manner.

2) With the AC system DR control problem in SMCB formulated as a Markov Decision Process (MDP), an innovative nonlinear reward structure is designed to properly guide the coordination among multiple AC units in a building and to balance the trade-off between discomfort and energy usage.

3) Scalable state-of-the-art RL algorithms that are suitable for cloud computing are leveraged for RL policy training, and their control efficacy is studied in simulation. The learning efficiency of RL control policy for buildings of different sizes is also investigated, based on experiments on a supercomputer.

4) Based on the pricing plan of one of the most popular cloud service providers, the preliminary cost analysis for training an RL controller for a SMCB is presented to show the financial feasibility of the proposed framework.

In all, the cost reduction based on the proposed framework is expected to trigger higher DR program participation rate among SMCB in real world scenarios.

II. DR Control and the Proposed Framework

In incentive-based DR [30] programs, customers reduce power under a limit $\mathcal{P}$ as an obligation for receiving incentives. Lower $\mathcal{P}$ means more load reduction and economic gain but meanwhile user discomfort increases. Typically, buildings are divided into areas with relatively independent thermal dynamics called thermal zones. In SMCB, multiple roof-top units (RTUs) are used for air-conditioning for their corresponding thermal zones. When DR events occur, due to their cyclic operation, these RTUs should be coordinated so that the total power consumption is constantly kept below $\mathcal{P}$. Depending on the DR programs, the electricity price during events can be either the same as normal hours (through capacity reservation), or a critical peak pricing adder will be charged to further discourage energy usage. In the case of increased price, building owners want to decrease energy usage while in the other case, they want to use as much energy as possible to minimize discomfort. In this study, both scenarios are discussed.

Mathematically, this paper aims at developing a controller to solve the above-mentioned SMCB RTU coordination problem as shown in (1). In this formulation, only AC RTUs are considered as controllable devices. Other appliances in buildings such as lighting and plug loads are not considered here because their control during a DR event is comparatively straightforward (namely turning off unnecessary load during events).

$$\begin{aligned} \underset{u_t^i \in \{0,1\}}{\text{minimize}} \quad & \sum_{i \in \mathcal{N}} \sum_{t \in \mathcal{T}} \mathcal{C}_d \left(T_t^i, \tau^i\right) + \eta \cdot \mathcal{C}_e(\mathcal{E}^i \left(u_t^i\right)) \\ \text{subject to} \quad & \sum_{i \in \mathcal{N}} P^i \cdot u_t^i \leq \mathcal{P} \left(\forall t\right), \\ & T_{t+1}^i = \mathcal{F}^i \left(T_t^i, u_t^i, \varrho_t^i\right) \left(\forall t, \forall i\right). \end{aligned} \tag{1}$$

Although (1) is a typical resource allocation problem in smart building control, the proposed edge/cloud integrated solution leverages RL to avoid high implementation cost of existing methods using the proposed framework shown in Fig. 1, which implements an end-to-end learning and fully automates three major tasks of the building automation for DR: building modeling, controller training and real-time execution.

1) *Building modeling*: Unlike optimization-based approaches, RL algorithms do not place any constraint on the form of building model $\mathcal{F}^i$. In fact, $\mathcal{F}^i$ does not even need to have an explicit mathematical expression. Therefore, on the proposed framework, a data-driven building thermal model learning module is used to avoid building-to-building configuration and to fully automate the learning process. This module learns $\mathcal{F}^i$ directly from operation data collected by edge devices, such as smart thermostats. Besides low modeling cost, another advantage of using a data-driven building model is that it can also be easily trained periodically to adapt to model drift. According to Fig. 1, $\mathcal{F}^i$ will be trained once data collected at the edge is received at the cloud.

2) *Controller training*: On the proposed framework, cloud computing is utilized to train the controller due to its powerful computing power and relatively low cost. Once a learned building thermal model $\mathcal{F}^i$ is obtained, it will be integrated into a building simulator, in the form of an OpenAI Gym environment [33]. Based on this Gym environment, an RL agent is trained to obtain an optimal control policy. Since the training happens during control system commissioning instead of a DR event, there is no need for on-demand computation, RL training can be scheduled to run when the cloud resource is cheap to further reduce training cost.

3) *DR control execution*: When the trained optimal control policy, implemented in a neural network, is ready, it is then downloaded to edge devices in the building, and will be activated for real-time execution when DR events start. During an event, the optimal control actions are determined by evaluating the output of the policy network given system state at that control interval. Such feed-forward network evaluation requires very limited computing power and is able to be easily carried out on edge devices in buildings. Over time, the policy remains optimal until the building model becomes inaccurate (e.g., caused by a change of occupancy behavior or building operation) and a new control policy can be re-trained by repeating tasks in the above two sections.

Among these modules on the proposed framework, Module 1, a building model learning module, will not be presented

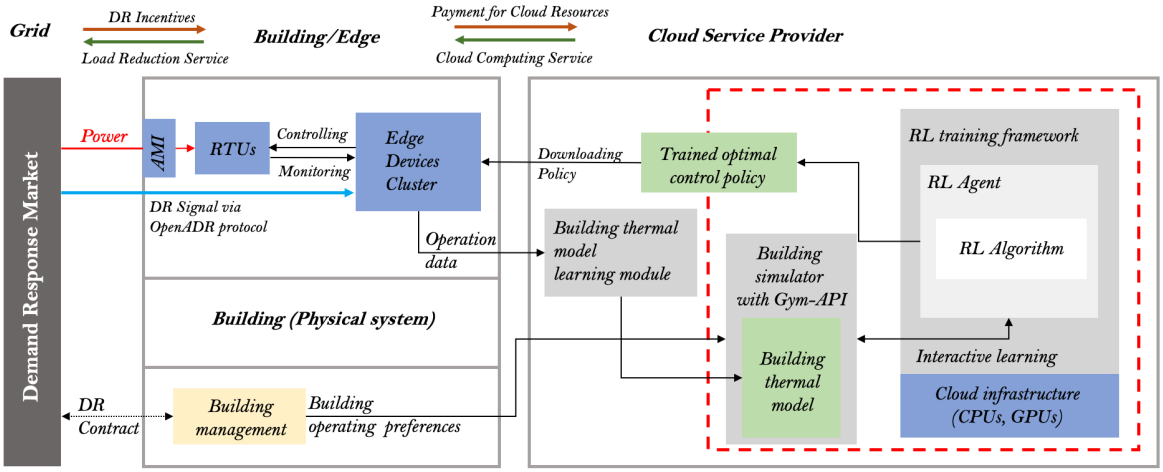


Fig. 1. Overview of the proposed paradigm, which fully automates building modeling, control policy learning and real-time execution. Building modeling module based on the Internet of Things and supervised learning are studied in [31] and [32]. The control policy learning module on cloud (shown in red dashed box) will be focused on in this paper.

in detail since it has been investigated and validated in our previous work [31] based on a real building experiment. Similarly, Module 3 will not be discussed since it is more of a software/hardware/communication implementation problem. So, in this paper, the RL controller training module, as shown in the red dashed box in Fig. 1, will be focused on. Details on how to formulate an RL problem for the RTU coordinated control in SMCB are explained in the following sections.

III. A Reinforcement Learning Approach

RL, derived from Markov Decision Process (MDP), can be used to solve a sequential stochastic optimal control problem: at Step t, an agent at a state ($\mathbf{s_t} \in \mathcal{S}$) takes an action ($\mathbf{a_t} \in \mathcal{A}$) in an environment, then receives a reward ($r_{t+1} = r(\mathbf{s_t}, \mathbf{a_t}) \in \mathcal{R}$) and moves to the next state with probability ($\mathcal{P}r^{\mathbf{a}}_{\mathbf{ss'}} = p[\mathbf{s_{t+1}} = \mathbf{s'}|\mathbf{s_t} = \mathbf{s}, \mathbf{a_t} = \mathbf{a}]$). The agent makes its decision based on a policy $\pi(\mathbf{a}|\mathbf{s}) = p[\mathbf{a_t} = \mathbf{a}|\mathbf{s_t} = \mathbf{s}]$. An optimal policy $\pi*$ ensures the agent receives a maximum cumulative reward $G^*_t = \sum_{n=t}^{T} \gamma^{n-t} r_n$ over a period of time called an episode. If the MDP $< \mathcal{S}, \mathcal{A}, \mathcal{P}r, \mathcal{R}, \gamma >$ is known, $\pi*$ can be obtained by dynamic programming. However, in real-world engineering problems, the transition probability $\mathcal{P}r^{\mathbf{a}}_{\mathbf{ss'}}$ and reward structure $\mathcal{R}^{\mathbf{a}}_{\mathbf{s}}$ are unknown. As a result, many RL algorithms are devised to train an agent to approximate $\pi*$ by learning from its interactive experience with the environment.

A. Agent, Actions and Environment

To coordinate RTUs during DR events, the RL agent determines u^i_t at each step. The action $\mathbf{a_t} = [u^1_t, u^2_t, ..., u^N_t] \in \mathcal{A}$ is a combination of multiple RTU control signals and $\mathcal{A}$, the legal actions set, is defined in (2). Since actions are selected from $\mathcal{A}$, the DR power limit constraint is always satisfied.

$$\mathcal{A} = \left\{ \mathbf{a} | \mathbf{P} \cdot \mathbf{a_t} = \sum_{i \in \mathcal{N}} P^i \cdot u^i_t < \mathcal{P}, u^i_t \in \{0, 1\} \right\} \quad (2)$$

According to Fig. 1, $\mathcal{F}^i$ is the kernel for the OpenAI Gym environment, representing the dynamic of the controlled system. $\mathcal{F}^i$ determines next step building condition based on current condition and RTU control signals, as shown in (3). $\left(\dot{T}^i_t\right)^*$ for $* \in \{+, -\}$ are the temperature gradients predicted by the data-driven thermal models, which map a given feature vector to the temperature gradient (e.g., in form of various supervised learning models). The feature vector input usually contains environmental variables (e.g., outdoor temperature/humidity), however, for simplicity, typical values for these environmental variables are used in this study due to the similarity of the outdoor environments during DR events. In (3), T^i_{t+1} is sampled from a normal distribution to consider uncertainty from disturbance. σ is the prediction error standard deviation and it can be obtained from the thermal model learning module.

$$T^i_{t+1} \sim \mathbb{N}\left(T^i_t + \left(1 - u^i_t\right) \cdot \left(\dot{T}^i_t\right)^+ + u^i_t \cdot \left(\dot{T}^i_t\right)^-, \sigma^2\right) \quad (3)$$

B. Rewards

To align the reward structure with the control objectives, the reward is devised to meet the following requirements:

Req. a) Keeping $T^i_t < \tau^i$ if possible ($i \in \mathcal{N}\, and\, t \in \mathcal{T}$);

Req. b) If $T^i_t > \tau^i$, the agent should take immediate actions to ameliorate the thermal condition in this zone;

Req. c) If T_i is about to exceed τ_i, Zone i should have higher priority to be cooled than Zone j with $T^j_t < T^i_t$;

Req. d) If $\eta = 1$ in (1), the agent should maintain thermal comfort using as little energy as possible.

To this end, three reward components are proposed:

1) Thermal comfort margin reward

A non-linear function is used to quantify the thermal comfort margin, as shown in (4) (Positive margin if $T < \tau^i$ and negative if otherwise). ξ^i is the importance factor of Zone i, it can be set proportional to the occupants' number.

$$\varphi^i(T) = \xi^i \cdot \int_0^{\tau^i - T} f(x)\, dx \quad (4)$$

The reward related to thermal comfort margin of Zone i at time t is defined as (5), representing the increment of comfort margin with respect to that of the last step:

$$m_t^i = \varphi^i(T_{t+1}^i) - \varphi^i(T_t^i) \tag{5}$$

A decreasing exponential function, $f(x) = \beta^{-x}$, is used in this study. Therefore, for decreasing indoor temperature for ΔT, there is $\varphi^i(T_1 - \Delta T) - \varphi(T_1) > \varphi^i(T_2 - \Delta T) - \varphi(T_2)$, given $T_1 > T_2$. As a result, for thermal zones with the same cooling speed, the one with higher temperature has higher priority to be cooled (***Req. b*** & ***c***).

2) Threshold violation penalty

Considering ***Req. c***, a violation penalty, defined in (6), is added to the reward if the temperature just exceeded τ^i because the agent did not cool the room in time.

$$v_t^i = \begin{cases} C < 0 & \left(T_t^i < \tau^i \,\&\, T_{t+1}^i > \tau^i\right) \\ 0 & (\text{else}) \end{cases} \tag{6}$$

3) Energy usage penalty

If energy-saving is being considered, an energy usage penalty is added to discourage unnecessary cooling. Thus, a parameter λ^i is introduced to bridge the trade-off between the energy consumption and the thermal comfort. To facilitate properly choosing λ^i, pragmatically, T_{eco}^i which represents the lowest temperature the building owner is willing to paying the cooling for, is used. For Zone i being cooled from T_t^i to $T_t^i - \Delta T$, the reward is $r_t^i = \varphi(T_t^i - \Delta T) - \varphi(T_t^i) - \lambda^i \mathcal{E}_t^i$. An appropriately chosen λ^i ensures if $T_t - \Delta T \geq T_{eco}$, there is $r_t \geq 0$; and the reward $r_t < 0$ if $T_t \leq T_{eco}$. Because there is $\varphi(T_t^i - \Delta T_t) - \varphi(T_t^i) > \varphi(T_{eco}^i - \Delta T_{eco}) - \varphi(T_{eco}^i)$ if $T_t^i > T_{eco}^i$ (non-linearity of φ), and because $\left(\dot{T}_t^i\right)^-$ decreases with the decrease of T_t^i, setting λ^i according to (7) can guarantee T_{eco} is the dividing point of positive and negative r_t. The superscript i in (7) is omitted here for simplicity.

$$\begin{aligned} \lambda &= \lim_{\Delta t \to 0} \frac{\varphi(T_{eco}) - \varphi(T_{eco} + \Delta T)}{\mathcal{E}} \\ &= \lim_{\Delta t \to 0} \frac{\xi}{P \cdot \Delta t} \int_{\tau - Teco - \Delta T}^{\tau - Teco} \beta^{-x} dx \\ &= -\frac{\xi \cdot \beta^{Teco - \tau}}{P \cdot \ln \beta} \cdot \lim_{\Delta t \to 0} \frac{1 - \beta^{\Delta T}}{\Delta t} \\ &= \frac{\xi \cdot \beta^{T_{eco} - \tau}}{P} \cdot \lim_{\substack{T = T_{eco} \\ \Delta t \to 0}} \frac{\Delta T}{\Delta t} \end{aligned} \tag{7}$$

The last limit term in (7) is the decreasing temperature gradient, which can be obtained from the data-driven building model (i.e., $\left(\dot{T}_t^i\right)^-$). With the calculated λ^i, the energy consumption penalty is calculated using $e_t^i = -\lambda^i \cdot \mathcal{E}_t^i(u_t^i) = -\lambda^i \cdot P^i \cdot u_t^i \cdot \Delta t$.

Finally, putting the three components together, the single step reward at Step t is $r_t = \sum_{i \in \mathcal{N}} (m_t^i + v_t^i + e_t^i \cdot \eta)$.

C. States and Control Horizon

The state is defined as $\mathbf{s_t} = [T_t^1, T_t^2, ..., T_t^N, t] \in \left(\mathbb{R}^N \otimes \mathbb{Z}^1\right)$ in this study. Augmenting $[T_t^1, T_t^2, ..., T_t^N]$ with the current step is because the same/similar temperatures might occur at different stage of a DR event and the optimal control might vary. The control horizon is the total number of control steps, namely DR duration divided by the length of control interval. As mentioned earlier, outdoor environment variables (e.g., temperature and humidity) are not included in the state for simplicity since this study assumes DR events usually happen with similar outdoor environment and thus typical and fixed values are used. If not the case, outdoor environment changes should also be considered in the building thermal simulator and be included in the state representation.

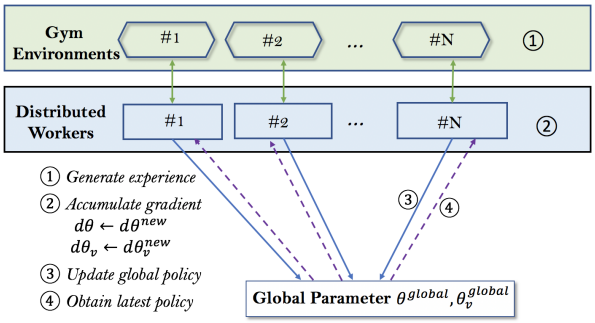


Fig. 2. Illustration of A3C algorithm. Solid blue arrows show asynchronous gradient update (i.e., $\theta^{global} = \theta^{global} + \alpha \cdot d\theta^j$) and purple dashed arrows represent policy update for each worker (i.e., $\theta^j = \theta^{global}$).

IV. RL Algorithms and Learning Architectures

In this study, two state-of-the-art asynchronous RL algorithms and architectures are used for training due to their scalability for cloud computing and desirable learning efficiency.

A. A3C

A3C [34] stands for *asynchronous advantage actor-critic* algorithm, a parallel learning policy-gradient method. It can effectively scale learning on a worker set $\mathcal{W}$. As shown in Fig. 2, each worker interacts with and explores its own Gym environment. Worker $j \in \mathcal{W}$ learns from its experience (i.e., in tuple format $< \mathbf{s_t}, \mathbf{a_t}, r_t, \mathbf{s_{t+1}} >$) and accumulates the gradients for the policy (π) and the value (V) function, see (8) and (9). θ^j and θ_v^j are Worker j's copy of neural network parameters for policy and value network. R is the cumulative discounted reward. During training, global parameters will be updated asynchronously using $d\theta^j$ and $d\theta_v^j$ (i.e., $\theta^{global} \leftarrow \theta^{global} + \alpha \cdot d\theta^j, \forall j \in \mathcal{W}$ and similarly for θ_v^{global}). On the other hand, Worker j will fetch the policy with the most updated global parameters and continue for further experience collection ($\theta^j = \theta^{global}$ and $\theta_v^j = \theta_v^{global}$). For more detail about the A3C algorithm, please refer to *Algorithm 3* in [34]. In previous studies, A3C has shown great performance in many benchmark problems and one special advantage is its relatively low hardware requirements (single multi-core CPU rather than GPU).

$$d\theta^j \leftarrow d\theta^j + \nabla_{\theta'} \log \pi\left(\mathbf{a_t}|\mathbf{s_t}; \theta^j\right)\left(R - V\left(\mathbf{s_t}; \theta_v^j\right)\right) \tag{8}$$

$$d\theta_v^j \leftarrow d\theta_v^j + \partial\left(R - V\left(\mathbf{s_t}; \theta_v^j\right)\right)^2 / \partial \theta_v^j \tag{9}$$

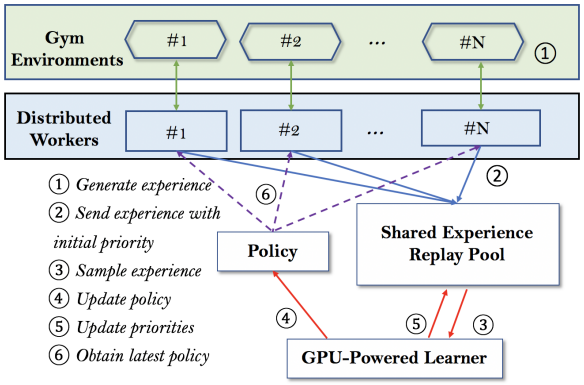


Fig. 3. Illustration of Ape-X Framework. Solid blue arrows show experience pooling and dashed purple arrows represent policy update.

B. Ape-X

Ape-X is a high-throughput architecture with prioritized experience replay [35]. Though Ape-X also achieves high learning efficiency by scaling the training tasks to multiple workers, unlike A3C, it decouples environment interaction and policy learning (i.e. gradient calculation): multiple CPU-based actors generate experience while a GPU-based learner learns the policy from the most significant data in the prioritized experience replay, as exemplified in Fig. 3. On this learning architecture, the GPU learner uses a deep Q-network (DQN) or deep deterministic policy gradient (DDPG) for discrete and continuous control problems, respectively. This framework has proven to train an agent to achieve good control performance with much less training time. Thus, in this study, Ape-X is used when the action space is large, and more exploration is needed to converge to an optimal policy. Using Ape-X, we can obtain higher learning efficiency compared with using A3C, but at the expense of more computing resources.

V. Case Study

To evaluate the proposed framework, this section studies the RL controller's control efficacy, parameter sensitivity and learning efficiency.

A. Baseline Control

Three controlling algorithms are introduced as baselines. Two of them are easy-to-implement algorithms without real-time computation requirement and one of them is the optimization-based model predictive control, representing the state of the art advanced building control technique.

1) Hysteresis Control by Set Point Setback (SPC)

Algorithm 1 shows the hysteresis control at each step. It does not consider multi-unit coordination and energy-saving, but is the most commonly used practice in real-life applications.

2) Greedy Rule-Based Controller (RBC)

Algorithm 2 shows a greedy rule-based controller which chooses actions from the legal action set $\mathcal{A}$ to cool zones with highest temperature at the current step.

Algorithm 1 for set point setback controller

Input: $[T_t^1, T_t^2, ..., T_t^N]$, $[T_{set}^1, T_{set}^2, ..., T_{set}^N]$

for $i \in \mathcal{N}$ **do**
  if $T_t^i \geq T_{set}^i + 1$ **then**
    $u_{t+1}^i = 1$
  else if $T_t^i \leq T_{set}^i$ **then**
    $u_{t+1}^i = 0$
  else
    $u_{t+1}^i = u_t^i$
  end if
end for
return $\mathbf{a_{t+1}} = [u_{t+1}^1, u_{t+1}^2, ..., u_{t+1}^N]$

Algorithm 2 Greedy Rule-Based Control

Input: $[T_t^1, T_t^2, ..., T_t^{\mathcal{N}}]$, $[\tau^1, \tau^2, ..., \tau^{\mathcal{N}}]$

Calculate linear margin: $\mathbf{K} = [\kappa^i = \tau^i - T_t^i, \forall i \in [1, \mathcal{N}]]$
Identify $\hat{i} = \arg\min_{i \in [1,\mathcal{N}]} \kappa^i$
Generate possible actions: $\mathcal{A}_{pos} = [\mathbf{a} | \mathbf{a}[\hat{i}] = 1 \,\&\, \mathbf{a} \in \mathcal{A}]$
if Energy saving considered **then**
  Remove actions that cools low temperature rooms $\mathcal{A}_{pos} = [\mathbf{a} | T_t^i > T_{eco}^i \; \forall \mathbf{a}[i] = 1 \,\&\, \mathbf{a} \in \mathcal{A}_{pos}]$
end if
if $\|\mathcal{A}_{pos}\| = 1$ **then**
  $\mathbf{a}_{t+1} = \mathbf{a} \in \mathcal{A}_{pos}$
else
  Greedily select action that cools most number of zones $\mathcal{A}_{pos} = [\mathbf{a} | \mathbf{a} \cdot \mathbf{1} \geq \mathbf{a}\prime \cdot \mathbf{1}, \forall \mathbf{a}\prime \in \mathcal{A}_{pos} \,\&\, \mathbf{a} \in \mathcal{A}_{pos}]$
  if $\|\mathcal{A}_{pos}\| = 1$ **then**
    $\mathbf{a}_{t+1} = \mathbf{a} \in \mathcal{A}_{pos}$
  else
    Select action that cools zones with higher temperature $\mathcal{A}_{pos} = [\mathbf{a} | \mathbf{a} \cdot \mathbf{T} \geq \mathbf{a}\prime \cdot \mathbf{T}, \forall \mathbf{a}\prime \in \mathcal{A}_{pos} \,\&\, \mathbf{a} \in \mathcal{A}_{pos}]$
    if $\|\mathcal{A}_{pos}\| = 1$ **then**
      $\mathbf{a}_{t+1} = \mathbf{a} \in \mathcal{A}_{pos}$
    else
      Random action selection: $\mathbf{a}_{t+1} = \mathbf{a} \in \mathcal{A}_{pos}$
    end if
  end if
end if
return $\mathbf{a}_{t+1}$

3) Model Predictive Control (MPC)

A model predictive controller, whose objective function shares the same components with the RL reward structure proposed in Section III-B (see Appendix A for details of optimization problem formulation used in MPC), is devised. The MPC controller solves an optimization problem every time step repeatedly, and the length of optimizing/planning horizon is reduced every step since the DR duration is finite, as shown in Fig. 4.

B. Simulation Setting and RL Parameter Selection

In this study, a building that has four thermal zones is cooled by four RTUs ($N = 4$) with $\mathbf{P} = [8.5, 7.0, 12.0, 4.5]$ (unit: kW). Thermal models for four zones used here is

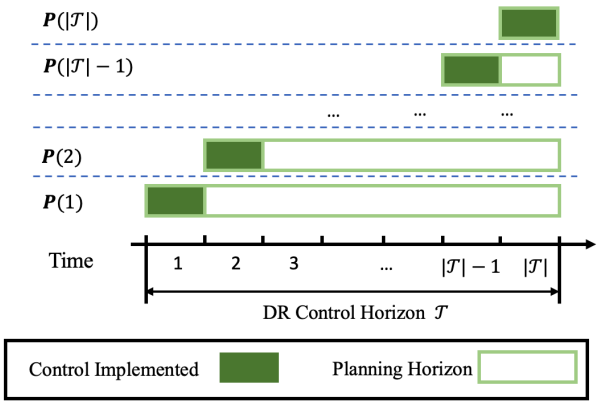


Fig. 4. MPC controller repeatedly solves $|\mathcal{T}|$ optimization problems (denote as $\boldsymbol{P}$(*)) before each control step over the DR horizon. At the beginning of each step, the optimal action for the first step will be implemented and the indoor temperature at the end of each step will be used as initial temperature for the problem at the next step.

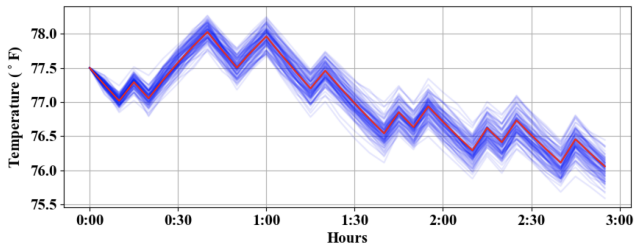


Fig. 5. Thermal model considering prediction error (as shown in (3)). Red curve shows the temperature profile predicted while blue curves shows possible 'actual' temperature profile with prediction error considered, given the same T_0 and control sequence.

obtained from [31], which is trained using one month of smart thermostats' data collected from a real building. Assuming a typical DR event lasts three hours with a control interval of 5 minutes ($\Delta t = 300$) for a total number of 36 intervals ($\|\mathcal{T}\| = 36$). Uncertainty in (3) simulates the randomness in the thermal model, illustrated in Fig. 5. $\mathcal{P}$ is set to be 13 kW. In each episode, initial zone temperatures are uniformly sampled from [75.0, 78,0] (unit: °F, same unit for temperature hereinafter) with 0.5 interval (emulating actual thermostat readings). τ for four zones are set to be [78.0, 78.0, 78.0, 77.0] (Zone 4 requires slightly lower temperature). If energy saving is considered, there is $\mathbf{T_{eco}} = [76.0, 76.0, 76.0, 75.0]$. Other simulator related parameters are configured as follow: the non-linear margin function is $f(x) = 2^{-x}$, importance factor $\xi^i = 2.0$, and the violation penalty is $C = -1000$. The energy saving coefficients λ^i are configured according to (7), see Table I. Fig. 6 takes one thermal zone as an example and shows by setting λ according to Table I, the reward becomes negative when $T_0 < T_{eco}$ to avoid unnecessary cooling.

TABLE I
ENERGY SAVING COEFFICIENT λ FOR FOUR ZONES

Zone ID	1	2	3	4
λ (10e-5)	3.6259	1.2476	3.2448	5.3410

Once the optimal control policy π^* is learnt, it is expected to be downloaded to edge devices in the building and the optimal control action at each step will be determined by $\mathbf{a}_{t+1} = \pi^*(\mathbf{s_t})$ on edge devices during a DR event.

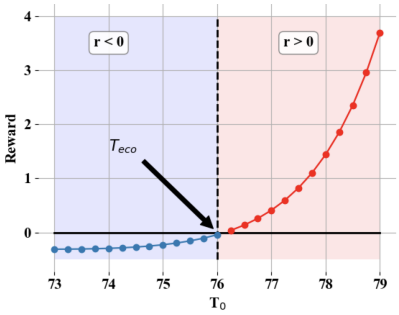


Fig. 6. One-step reward for a single thermal zone. T_0 represents the temperature at the beginning of the control interval. It shows when T_0 is below T_{eco}, cooling is discouraged due to the negative reward.

C. Performance Study

1) RL Control Efficacy

Fig. 7 shows multi-zone T^i profiles over three hours of DR events in 300 simulations using the trained π^*. Initial temperatures of the four thermal zones are randomly sampled from the same distribution used in the training process. Fig. 7 demonstrates that $T_t^i < \tau^i$ $(\forall i \in \mathcal{N}, \forall t \in \mathcal{T})$, which implies an effective control under various indoor conditions. It is worth noting that the RL agent also learned knowledge that is not explicitly given: In Zone 2, even without cooling, the temperature can naturally reach its equilibrium at $77°F$, due to good insulation and cooling impact from neighboring zones. Since $\tau^2 > 77$, the agent learnt from its experience that it would be more effective to spend the cooling capacity on zones other than Zone 2. This can be observed in Fig. 7 that for other zones, even with lower temperature (e.g. $76°F$), they will be pre-cooled to avoid a possible conflict of cooling demand later, but the pre-cooling does not happen in Zone 2.

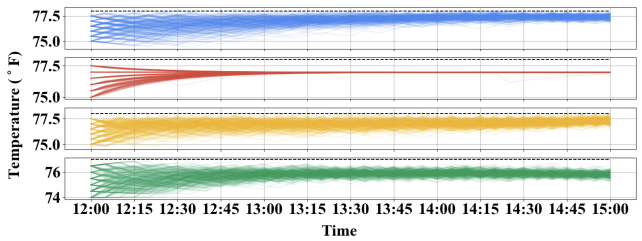
Comparing (a) and (b) in Fig. 7, it can be seen that the cooling need for Zone 1 and 3 are inelastic demand, while that of Zone 4 is elastic. This means when energy saving is considered, it is more cost-effective to keep T_t^4 at a higher level. Only until the last hour, the agent decides to further cool Zone 4 to gain extra reward.

Fig. 8 depicts the change of $p(u^i = 1)$ in a state trajectory: [78.2, 77.9, T^3, 76.2, 0], with T^3 : 74.5 → 78.2. It can be seen that the cooling priority switches from Zone 1 to Zone 3 as the temperature in Zone 3 increases. $p(u^3 = 1)$ exceeds $p(u^1 = 1)$ even before $T^3 > T^1$. This is mainly because there is $(T_t^3)^+ > (T_t^1)^+$ and $(T_t^3)^- < (T_t^1)^-$ in this temperature range, so it is more effective to cool Zone 3 first before T^3 gets too high. This shows the agent makes a more comprehensive decision rather than just simply cooling the hottest zone.

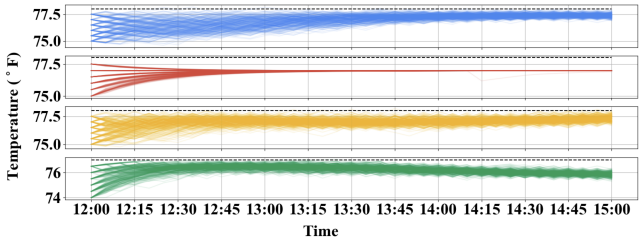
2) Comparison with Baseline Controllers

The control performance of the proposed RL controller (RLC) is compared with the above-mentioned three baseline controllers (SPC, RBC and MPC in Section V-A).

First, we compare RLC with SPC and RBC, all of which do not require on-demand computation and can be easily implemented in real-life applications. The comparison is conducted



(a) Energy saving not considered



(b) Energy saving considered

Fig. 7. Indoor temperature of four thermal zones (Zone 1-4 from top to bottom) during 3-hour DR events, each curve represents one of the 300 simulations. Black dashed lines are τ.

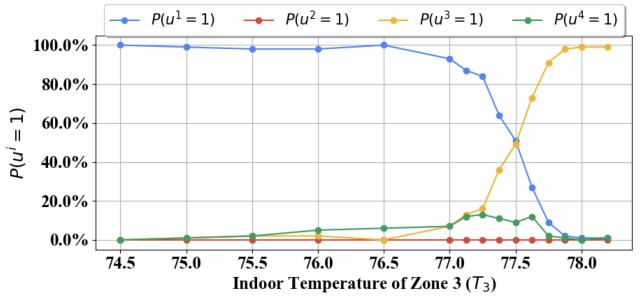


Fig. 8. Values of $P(u^i = 1)$ change along the state trajectory of [78.2, 77.9, T^3, 76.2]. $P(u^i = 1)$ represents the probability that RTU i is ON.

from the perspective of average zone temperature T^i_{avr} (comfort), average energy consumption $\mathcal{E}_{avr}$ (efficiency of control), maximum power consumption $\mathcal{P}_{peak}$ (DR requirement) and average RTU toggle counts n_{toggle} (proper usage of RTUs). The set points used in SPC are set to be $\tau^i - 1$ since the thermostat's deadband is $[\tau^i - 1, \tau^i]$. The comparisons based on 300 simulations are shown in Table II. Table II shows that $T^i_{avr} < \tau^i (\forall i \in \mathcal{N})$ for all three controllers. Due to lack of coordination, SPC fails at meeting $\mathcal{P}_{peak} \leq \mathcal{P}$. Both RBC and RLC can limit $\mathcal{P}_{peak}$ and they have similar T^i_{avr}, but RLC has three advantages over RBC: *a)* when energy saving is considered, due to proper planning, RLC yields smaller $\mathcal{E}_{avr}$ than RBC; *b)* Due to the greedy feature, RBC has higher n_{toggle} (which accelerate the device aging); *c)* RBC cannot prioritize a specific zone while RLC can achieve that, which is discussed in the next section.

Similarly, to compare the control performance of RLC and MPC, 300 simulations with randomized initial conditions are conducted. Table III shows that with RLC, the number of thermal comfort violation is far less than that of the MPC. In addition, Fig.9 shows the comparison between the objective values (i.e., minimized cost) of two controllers over the 300 identical simulations: the majority of data points are below the

TABLE II
CONTROL PERFORMANCE FOR RLC AND THREE BASELINE CONTROLLER

Controller		**SPC**	**RBC**		**RLC**	
Energy saving considered		N/A	Yes	No	Yes	No
$T^i_{avr}(^\circ F)$	1	77.423	77.125	77.098	76.939	77.036
	2	76.887	76.807	76.753	76.880	76.872
	3	77.415	77.181	77.166	77.026	76.920
	4	76.529	75.878	75.847	76.420	75.782
$\mathcal{E}_{avr}$ (kWh)		31.235	36.677	37.402	34.205	37.464
$\mathcal{P}_{peak}$ (kW)		25.0	13.0	13.0	13.00	13.0
n_{toggle}		6.834	23.844	24.087	17.848	17.441

TABLE III
AVERAGE THERMAL COMFORT UPPER BOUND VIOLATION COUNT IN 300 RANDOMIZED SIMULATIONS

Controller	RLC	MPC
Average violation count	0.07	2.70

diagonal, indicating that RLC can in general achieve lower cost than MPC. The average costs for RLC and MPC are 7.038 and 7.537 respectively, using RLC can achieve 6.6 % of objective function reduction. The better performance of RLC mainly comes with its ability to effectively handle a more complicated system model, where disturbance is considered. On the other hand, with the simplified model, the linear MPC's performance becomes sub-optimal and leads to a more frequent comfort bound violation.

3) Zone Priority

The proposed method can implement zone prioritization by setting different importance factor ξ^i in (4). Fig. 10 shows a comparison when Zone 3 is prioritized ($\xi^3 = 2 \cdot \xi^i (i \in 1, 2, 4)$). In Fig.10, if Zone 3 is prioritized, T^3 is lower than that of the equal priority case; meanwhile, T^1 and T^4 are higher since more resource are allocated to cool Zone 3. Admittedly, the priority setting here is relative and cannot be quantified.

4) Choice of Environment Setting

To study the sensitivity of agent behavior to the choice of $f(x) = \beta^{-x}$ in (4), several values for β are discussed. $\beta = 0$ is not considered because a linear $f(x)$ cannot prioritize cooling higher temperature zones. Thus, two policies trained by $\beta = 2$

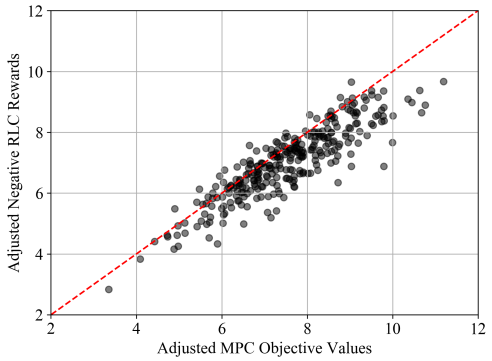


Fig. 9. Comparison of adjusted control costs using the proposed RLC and an MPC in 300 identical trials. Values are adjusted by disregarding thermal violation penalty, which has been compared in Table III (i.e., (18) in Appendix.A).

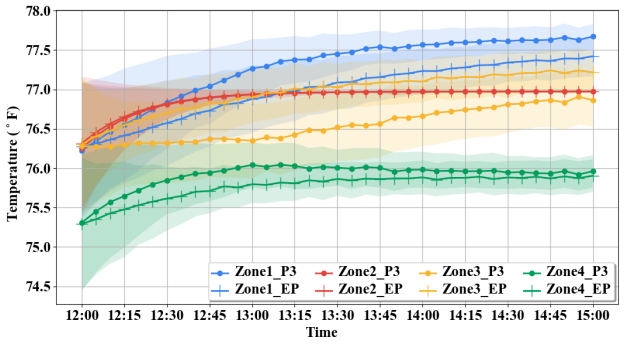


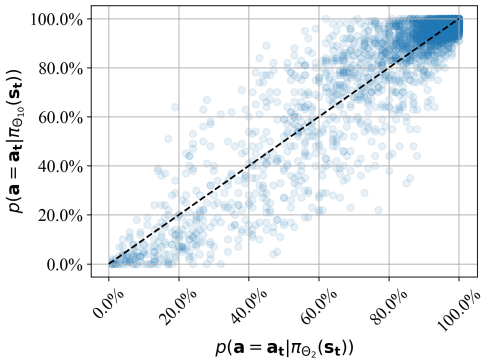
Fig. 10. Mean and standard deviation of indoor temperature for 300 simulations in four zones over the 3-hour DR event. Curves with '·' represent temperature profiles when Zone 3 is prioritized [P3]; Curves with '+' represent temperature profile when all zones have the equal priority[EP] (same as Fig. 7(a).) Shaded areas represent standard deviation.

and $\beta = 10$ (moderate and strong non-linearity), written as $\pi_{\theta_2}(\mathbf{s})$ and $\pi_{\theta_{10}}(\mathbf{s})$ are compared. To compare policy behavior, the validity of the relationship shown in (10) is investigated.

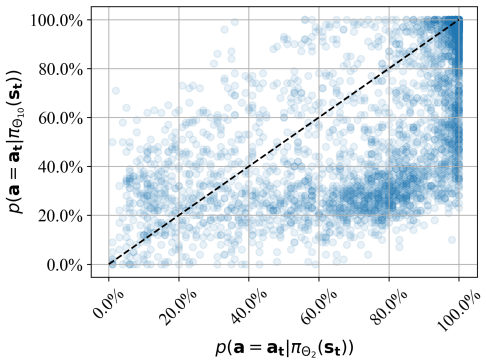
$$p(\mathbf{a} = \mathbf{a_t}|\pi_{\theta_2}(\mathbf{s_t})) \approx p(\mathbf{a} = \mathbf{a_t}|\pi_{\theta_{10}}(\mathbf{s_t})) \tag{10}$$

Both $\pi_{\theta_2}(\mathbf{s})$ and $\pi_{\theta_{10}}(\mathbf{s})$ are tested in 300 simulations each, and all state-action pairs $\{\mathbf{s_t}, \mathbf{a_t}\}$, from 300 control trajectories are put into sets of $\mathbf{Traj}_{\theta_2}$ and $\mathbf{Traj}_{\theta_{10}}$. Fig. 11 compares the probability of actions chosen by two policies for all $\{\mathbf{s_t}, \mathbf{a_t}\} \in \mathbf{Traj}_{\theta_2} \bigcup \mathbf{Traj}_{\theta_{10}}$. Fig. 11 (a) shows that when energy-saving is not considered, (10) is valid since all $(p(\mathbf{a} = \mathbf{a_t}|\pi_{\theta_2}(\mathbf{s_t})), p(\mathbf{a} = \mathbf{a_t}|\pi_{\theta_{10}}(\mathbf{s_t})))$ are distributed along the diagonal line, which implies strong similarity between two policies' behavior. In addition, the average temperature trajectories of both $\pi_{\theta_2}(\mathbf{s})$ and $\pi_{\theta_{10}}(\mathbf{s})$ from the 300 simulations also closely match with each other. If energy-saving is considered, however, (10) is not valid as shown in Fig. 11 (b). One observation is that for many $\mathbf{s_t}$, there are $p(\mathbf{a} = \mathbf{a_t}|\pi_{\theta_2}(\mathbf{s_t})) \approx 100\%$, but $p(\mathbf{a} = \mathbf{a_t}|\pi_{\theta_{10}}(\mathbf{s_t}))$ scatters from 20% to 100%. To analyze the difference, individual instances are investigated and two representative states are presented: Fig. 12 illustrates the probability distribution from $\pi_{\theta_2}(\mathbf{s})$ and $\pi_{\theta_{10}}(\mathbf{s})$, given $\mathbf{s_1}$ and $\mathbf{s_2}$. In the case of $\mathbf{s_1}$, the main difference between π_{θ_2} and $\pi_{\theta_{10}}$ is whether Zone 4 should be cooled. π_{θ_2} chooses not to cool Zone 4 because when energy-saving is considered, the agent want keep the temperature as high/closer to τ as possible. But when $\beta = 10$ is used in (7), the energy-saving consideration becomes insignificant because λ is too small and thus $\pi_{\theta_{10}}$ does not have preference among '1000' and '1001'. In the case of $\mathbf{s_2}$, $T^i(i \in \mathcal{N})$ are relatively low, π_{θ_2} determines to pre-cool Zone 3 (with highest temperature) while $\pi_{\theta_{10}}$ has a wide spread of action probability. This is again due to the strong non-linearity of $f(x) = 10^{-x}$; the reward for cooling at lower temperature are negligible, so $\pi_{\theta_{10}}$ decides to randomly choose an action until some zones reach a higher temperature and then cooling those to obtain much larger reward.

To summarize, when energy-saving is not considered, the agent behavior is not sensitive to β's value. But when energy-saving is considered, we observe that using $\beta = 2$ better aligns the policy behavior with the desired control objectives.



(a) Energy saving not considered



(b) Energy saving considered

Fig. 11. $p(\mathbf{a} = \mathbf{a_t}|\pi_{\theta_2}(\mathbf{s_t}))$ vs. $p(\mathbf{a} = \mathbf{a_t}|\pi_{\theta_{10}}(\mathbf{s_t}))$. Each dot represents one instance of $\{\mathbf{s_t}, \mathbf{a_t}\} \in \mathbf{Traj}_{\theta_2} \bigcup \mathbf{Traj}_{\theta_{10}}$.

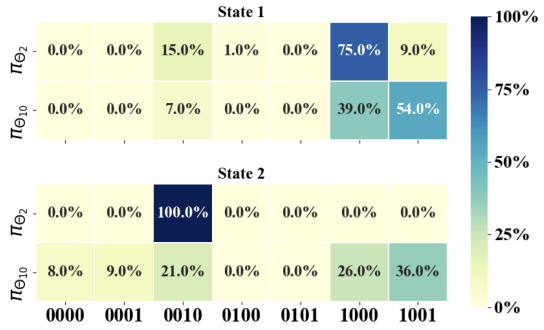
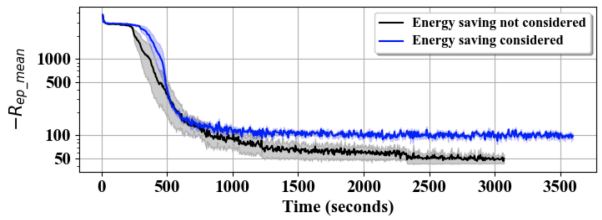


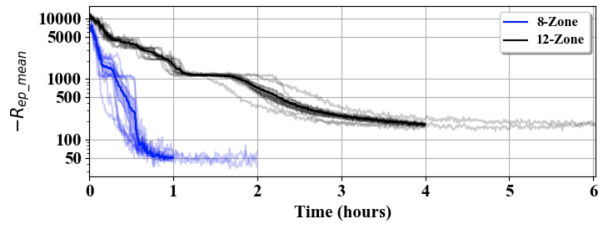
Fig. 12. Probability distribution for all legal actions based on $\pi_{\theta_2}(s)$ and $\pi_{\theta_{10}}(s)$. Two subplots share the same X labels, which are the binary representations for AC ON/OFF signal (e.g., '0010' stands for only AC unit in Zone 3 is ON.). There are $s_1 = [77.5, 74.3, 76.5, 76.2, 0]$ and $s_2 = [73.5, 74.3, 76.5, 73.2, 0]$.

D. Learning Efficiency and Cost Analysis

According to (2), action space size grows exponentially with the number of RTUs (known as the curse of dimensionality). A larger action space means longer exploration and learning time. In this section, the learning time for RL policy of three buildings with different sizes are summarized. The first building is the 4-zone building previously used; then, for simplicity, the 4-zone thermal model is replicated to create an 8-zone and a 12-zone building. Testing up to 12 zones is because 72.1% of total commercial buildings in the U.S. are less than 10,000 square feet [29] (assuming each thermal zone is on average 800 square feet). The 'cloud' computing



(a) Learning curves for 4-zone training. Curves show the change of negative average episodic reward ($-R_{ep\_mean}$) with wall time. Curves are average results from multiple runs, solid lines are the mean while the shaded areas represent 0.5 std. A3C is used.



(b) Learning curves for 8-zone and 12-zone training. Curves derived from ten trials for each case (solid lines are average and faded lines are individual trials). Ape-X is used. Two trials for each case run longer than others to show no further improvement.

Fig. 13. Learning curves for three cases. Energy-saving is not considered in the 8-zone and 12-zone cases.

resource used is the High Performance Computing system, named Eagle, at the U.S. Department of Energy's National Renewable Energy Laboratory (NREL). Eagle computing node has dual Intel 18-core processors. One Eagle node is used for training here, though multi-node training can be used for further acceleration.

For the 4-zone case, two learning curves are shown in Fig. 13a. In both cases, the average episodic reward ($-R_{ep\_mean}$ used in plot to log-scale the y-axis) converges to a value much larger than the violation penalty ($C = -1000$ in (6)). This means π^* can constantly maintain $T^i < \tau^i$. Comparatively, if energy-saving is considered, the reward converges to a lower value due to the extra energy consumption penalty. Fig. 13b shows the convergence for both 8-zone and 12-zone cases, also implying π^* are able to coordinate RTUs during DR events without breaching the τ^i. Table IV shows the size of the action spaces for the three problems. Due to a larger state and action space for control problems with 8 and 12 zones, we use Ape-X to solve them, with a GPU to accelerate the gradient computation. Neural network structures used for three problems are also shown in Table IV ([32, 32] means two hidden layers and each with 32 neurons). The activation functions for this fully connected policy network are all tanh functions. Learning rate for A3C and Apex-DQN are 1e-4 and 5e-4 respectively. Average numbers of training episodes for three cases are shown in Table IV (one episode means one rollout of control horizon), they represent how much experiences need to be collected in three cases until control policies are converged.

Finally, a preliminary cost estimation for training the optimal control policy π^* is shown in Table IV, considering the equivalent Amazon Web Service (AWS) EC2 instances. Since RL training does not require to be conducted on-demand, the EC2 spot instance price [36] is used, which is much cheaper

TABLE IV
LEARNING EFFICIENCY TESTS FOR BUILDINGS OF DIFFERENT SIZES

N	4	8	12
Action Space Size	7	83	785
$\mathcal{P}$ (kW)	13	26	35
RL Aglo/Framework	A3C	Ape-X	Ape-X
Network Structure	[32, 32]	[64, 64]	[64, 64]
Equivalent EC2 Instance	c5.9xlarge	g4dn.8xlarge	g4dn.8xlarge
Unit Cost [36] ($/Hour)	0.842	1.197	1.523
Training Episodes (1e6)	0.329	0.734	2.704
Training Time (Hour)	0.633	0.735	4.000
Estimated Total Cost ($)	0.533	0.880	6.092

TABLE V
ESTIMATED MONTHLY INCENTIVES

	May	**Jun**	**Jul**	**Aug**	**Sep**	**Oct**	**Sum**
$\mathcal{I}_u$	3.35	9.03	19.61	24.24	16.00	4.83	/
$\mathcal{I}_{sum}$	24.72	66.64	144.72	178.89	118.08	35.65	568.70

than the on-demand computing instances. The training time in Table IV is the average time it takes to first reach to the final converged values over 10 trials as shown in Fig. 13. Take the 4-zone building as an example, it consumes 20.38 kW on average during non-DR hours (simulated with set point of 75 $^\circ F$); Using this baseline, this building provides 7.38 kW load reduction. Assuming the reduction potential is the same for all months, Table V shows the unit incentive ($\mathcal{I}_u$, \$/kW-Month, using SDGE CBP rate [37]) and the estimated monthly and annual incentive received ($\mathcal{I}_{sum}$, \$). RL policy might need weekly re-training due to model drift of building thermal dynamics, so the monthly model training cost is $\$0.533\times4 = \$2.132$. Compared with $\mathcal{I}_{sum}$, the policy learning cost is trivial ($\$2.132 \times 6/\$568.70 = 2.25\%$). Since low-cost edge device and cloud computing are used, other costs such as hardware investment and maintenance are also minimized. Moreover, transfer learning can be utilized to warm-start the training based on a pre-trained controller from a similar building, which will further reduce the training cost when compared with the 'training from scratch' approach.

VI. CONCLUSION

In this paper, an automated building DR control framework is presented that allows large scale deployments in a cost-effective way. Leveraging RL, this framework reduces the costs in two ways: First, labor costs for building modeling is avoided since RL is capable of learning from a non-linear or even mathematically non-expressible data-driven models; second, costly on-demand computation is circumvented using offline training of RL and no real-time remote communication is required during DR execution. Experiments show that the trained policy can properly coordinate multiple RTUs, keeping the indoor temperature within comfort band. Comparison with MPC and other heuristic baseline controllers also shows a superior control of the proposed framework. Finally, the cost for policy training is evaluated and a preliminary analysis reveals that training such a controller is cheap, which is expected to make intelligent DR control affordable and trigger higher DR participation rates among SMCB.

APPENDIX A
MODEL PREDICTIVE CONTROL

To compare the proposed RL controller with MPC, here we construct an MPC optimization problem whose objective function is equivalent to the RL reward structure introduced in Section III-B (i.e., $r_t = \sum_{i\in\mathcal{N}}(m_t^i + v_t^i + e_t^i)$). Below are the details for formulating the optimization problem.

A. Thermal comfort margin term (m_t^i)

According to (4) and (5), the thermal comfort margin improvement between two control steps is:

$$m_t^i = \frac{\xi^i}{ln\beta}(\beta^{(T_t^i-\tau^i)} - \beta^{(T_{t+1}^i-\tau^i)}) \tag{11}$$

To make the problem easily solved by a commercial solver, we approximate (11) by (12), in which $h(*)$ is a convex piecewise function with quadratic component that approximate the original exponential function.

$$\hat{m}_t^i = \frac{\xi^i}{ln\beta}(h(T_t^i - \tau^i) - h(T_{t+1}^i - \tau^i)) \tag{12}$$

B. Violation penalty term (v_t^i)

This penalty term in (6) will be disregarded in the objective due to the conditional step function's non-linearity and non-convexity. Instead, an explicit constraint will be added to guarantee occupants' thermal comfort:

$$T_t^i \le \tau^i + \psi \quad (\forall t, \forall i) \tag{13}$$

In most of the cases, $\psi = 0$ in (13): strictly limit indoor temperatures below the maximum tolerable temperature. However, due to the system disturbance, the constraints above still might be violated after implement the optimal action yield by **P**(t) (**P**(t) is the optimization problem solved at Step t, see Fig. 4 for example). This might lead to an infeasible initial temperature to **P**(t+1). In this case, $\psi > 0$ is set for the first step in **P**(t+1) to relax the constraint and continue the MPC computation. When T_t^i returns below τ^i, ψ will again set back to zero.

C. Linearizing building thermal dynamics

The building temperature gradient prediction model, which can be in any format of a supervised learning model, is linearized as follow, in which μ^i, ν^i, ω^i and ζ^i are known:

$$\begin{aligned}(\dot{T}_t^i)^+ &= (\mu^i \cdot T_t^i + \nu^i)\cdot\Delta t \\ (\dot{T}_t^i)^- &= (\omega^i \cdot T_t^i + \zeta^i)\cdot\Delta t\end{aligned} \tag{14}$$

With (14), building thermal model becomes:

$$\begin{aligned}T_{t+1}^i &= T_t^i + \left(1-u_t^i\right)\cdot\left(\dot{T}_t^i\right)^+ + u_t^i\cdot\left(\dot{T}_t^i\right)^- \\ &= (1+\mu^i\cdot\Delta t)T_t^i + (\zeta^i - \nu^i)\Delta t\cdot u_t^i + \nu^i\cdot\Delta t \\ &\quad + (\omega^i - \mu^i)\Delta t\cdot u_t^i\cdot T_t^i\end{aligned} \tag{15}$$

Due to the bilinear term in (15), a new variable $y_t^i = u_t^i\cdot T_t^i$ is introduced. Because of this, two constraints based on the big-M method are included in the optimization problem:

$$-M\cdot u_t^i \le y_t^i \le M\cdot u_t^i \tag{16}$$

$$-(1-u_t^i)\cdot M \le y_t^i - T_t^i \le (1-u_t^i)\cdot M \tag{17}$$

D. Optimization Problem

Corresponding to the RL reward structure, the objective function for **P**(t*) in MPC is shown in (18). The goal is to maximize thermal comfort margin over the control horizon with minimum energy consumption.

$$\begin{aligned}\mathcal{C}(t^*) &= \sum_{i\in\mathcal{N}}\sum_{t\in\{t^*,\dots,|\mathcal{T}|\}}(-\hat{m}_t^i + \lambda^i\cdot P^i\cdot u_t^i\cdot\Delta t) \\ &= \sum_{i\in\mathcal{N}}(\frac{\xi^i}{ln\beta}(h(T_{|\mathcal{T}|}^i - \tau^i) - h(T_{t^*}^i - \tau^i))) \\ &\quad + \sum_{i\in\mathcal{N}}\sum_{t\in\{t^*,\dots,|\mathcal{T}|\}}\lambda^i\cdot P^i\cdot u_t^i\cdot\Delta t\end{aligned} \tag{18}$$

Combine all together, (19) shows the equivalent optimization problem **P**(t*) to be solved at each step. $T_{t^*}^{i,act}$ is the actual indoor temperature at t^* as initial values to **P**(t*).

$$\begin{aligned}\underset{u_t^i\in\{0,1\}}{\text{minimize}} \quad & (18) \\ \text{subject to} \quad & (13), (15)-(17), \\ & \sum_{i\in\mathcal{N}} P^i\cdot u_t^i \le \mathcal{P}\ (\forall t\in\{t^*,...,|\mathcal{T}|\}) \\ & T_{t^*}^i = T_{t^*}^{i,act} \quad (\forall i)\end{aligned} \tag{19}$$

REFERENCES

[1] U.S. Energy Information Administration. How much energy is consumed in U.S. residential and commercial buildings? Accessed: Dec. 3rd, 2019. [Online]. Available: https://www.eia.gov/tools/faqs/faq.php?id=86&t=1.

[2] San Diego Gas & Electric. AC Saver (Summer Saver). Accessed: May. 30th, 2020. [Online]. Available: https://www.sdge.com/residential/savings-center/rebates/your-heating-cooling-systems/summer-saver-program.

[3] Appalachian Power. Residential Peak Reduction Program. Accessed: May. 30th, 2020. [Online]. Available: https://www.appalachianpower.com/save/residential/programs/VAResidentialPeakReductionProgram.aspx.

[4] Nerea Ruiz, Iñigo Cobelo, and José Oyarzabal. A direct load control model for virtual power plant management. *IEEE Transactions on Power Systems*, 24(2):959–966, 2009.

[5] Yudong Ma, Francesco Borrelli, Brandon Hencey, Brian Coffey, Sorin Bengea, and Philip Haves. Model predictive control for the operation of building cooling systems. *IEEE Transactions on control systems technology*, 20(3):796–803, 2011.

[6] Samuel Privara, Jan Širokỳ, Lukáš Ferkl, and Jiří Cigler. Model predictive control of a building heating system: The first experience. *Energy and Buildings*, 43(2-3):564–572, 2011.

[7] Jie Cai, Donghun Kim, Rita Jaramillo, James E Braun, and Jianghai Hu. A general multi-agent control approach for building energy system optimization. *Energy and Buildings*, 127:337–351, 2016.

[8] Yize Chen, Yuanyuan Shi, and Baosen Zhang. Optimal control via neural networks: A convex approach. *arXiv preprint arXiv:1805.11835*, 2018.

[9] Mohammed M Olama, Teja Kuruganti, James Nutaro, and Jin Dong. Coordination and control of building hvac systems to provide frequency regulation to the electric grid. *Energies*, 11(7):1852, 2018.

[10] Xiangyu Zhang, Manisa Pipattanasomporn, and Saifur Rahman. A self-learning algorithm for coordinated control of rooftop units in small-and medium-sized commercial buildings. *Applied energy*, 205:1034–1049, 2017.

[11] Alessandro Alessio and Alberto Bemporad. A survey on explicit model predictive control. In *Nonlinear model predictive control*, pages 345–369. Springer, 2009.

[12] Daniele Corona and Bart De Schutter. Adaptive cruise control for a smart car: A comparison benchmark for mpc-pwa control methods. *IEEE Transactions on Control Systems Technology*, 16(2):365–372, 2008.

[13] XI Yu-Geng, LI De-Wei, and Lin Shu. Model predictive control—status and challenges. *Acta Automatica Sinica*, 39(3):222–236, 2013.

[14] Mengmeng Cai, Manisa Pipattanasomporn, and Saifur Rahman. Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques. *Applied energy*, 236:1078–1088, 2019.

[15] Daniel L Marino, Kasun Amarasinghe, and Milos Manic. Building energy load forecasting using deep neural networks. In *IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society*, pages 7046–7051. IEEE, 2016.

[16] Chengdong Li, Zixiang Ding, Dongbin Zhao, Jianqiang Yi, and Guiqing Zhang. Building energy consumption prediction: An extreme deep learning approach. *Energies*, 10(10):1525, 2017.

[17] Cheng Fan, Fu Xiao, and Yang Zhao. A short-term building cooling load prediction method using deep learning algorithms. *Applied energy*, 195:222–233, 2017.

[18] Hanchen Xu, Xiao Li, Xiangyu Zhang, and Junbo Zhang. Arbitrage of energy storage in electricity markets with deep reinforcement learning. *arXiv preprint arXiv:1904.12232*, 2019.

[19] Qiuhua Huang, Renke Huang, Weituo Hao, Jie Tan, Rui Fan, and Zhenyu Huang. Adaptive power system emergency control using deep reinforcement learning. *IEEE Transactions on Smart Grid*, 2019.

[20] Hanchen Xu, Hongbo Sun, Daniel Nikovski, Shoichi Kitamura, Kazuyuki Mori, and Hiroyuki Hashimoto. Deep reinforcement learning for joint bidding and pricing of load serving entity. *IEEE Transactions on Smart Grid*, 10(6):6366–6375, 2019.

[21] Zheng Wen, Daniel O'Neill, and Hamid Maei. Optimal demand response using device-based reinforcement learning. *IEEE Transactions on Smart Grid*, 6(5):2312–2324, 2015.

[22] Elena Mocanu, Decebal Constantin Mocanu, Phuong H Nguyen, Antonio Liotta, Michael E Webber, Madeleine Gibescu, and Johannes G Slootweg. On-line building energy optimization using deep reinforcement learning. *IEEE transactions on smart grid*, 10(4):3698–3708, 2018.

[23] Lei Yang, Zoltan Nagy, Philippe Goffin, and Arno Schlueter. Reinforcement learning for optimal control of low exergy buildings. *Applied Energy*, 156:577–586, 2015.

[24] Tianshu Wei, Yanzhi Wang, and Qi Zhu. Deep reinforcement learning for building hvac control. In *Proceedings of the 54th Annual Design Automation Conference 2017*, page 22. ACM, 2017.

[25] Zhiang Zhang, Adrian Chong, Yuqi Pan, Chenlu Zhang, and Khee Poh Lam. Whole building energy model for hvac optimal control: A practical framework based on deep reinforcement learning. *Energy and Buildings*, 199:472–490, 2019.

[26] Zhijin Cheng, Qianchuan Zhao, Fulin Wang, Yi Jiang, Li Xia, and Jinlei Ding. Satisfaction based q-learning for integrated lighting and blind control. *Energy and Buildings*, 127:43–55, 2016.

[27] Bingnan Jiang and Yunsi Fei. Smart home in smart microgrid: A cost-effective energy ecosystem with intelligent hierarchical agents. *IEEE Transactions on Smart Grid*, 6(1):3–13, 2014.

[28] Donghun Kim, JE Braun, J Cai, and DL Fugate. Development and experimental demonstration of a plug-and-play multiple rtu coordination control algorithm for small/medium commercial buildings. *Energy and Buildings*, 107:279–293, 2015.

[29] U.S. Energy Information Administration. Commercial Building Energy Consumption Survey 2012, Table B6. Building size, number of buildings. 2016. Accessed: Jan. 3rd, 2020. [Online]. Available: https://www.eia.gov/consumption/commercial/data/2012/bc/cfm/b6.php.

[30] Mohamed H Albadi and Ehab F El-Saadany. A summary of demand response in electricity markets. *Electric power systems research*, 78(11):1989–1996, 2008.

[31] Xiangyu Zhang, Manisa Pipattanasomporn, Tao Chen, and Saifur Rahman. An iot-based thermal model learning framework for smart buildings. *IEEE Internet of Things Journal*, 199:472–490, 2019.

[32] Xiangyu Zhang, Mengmeng Cai, Manisa Pipattanasomporn, and Saifur Rahman. A power disaggregation approach to identify power-temperature models of hvac units. In *2018 IEEE International Smart Cities Conference (ISC2)*, pages 1–6. IEEE, 2018.

[33] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[34] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

[35] Dan Horgan, John Quan, David Budden, Gabriel Barth-Maron, Matteo Hessel, Hado Van Hasselt, and David Silver. Distributed prioritized experience replay. *arXiv preprint arXiv:1803.00933*, 2018.

[36] Amazon Web Services. Amazon EC2 Spot Instances Pricing. Accessed: Jun. 3rd, 2020. [Online]. Available: https://aws.amazon.com/ec2/spot/pricing/.

[37] San Diego Gas Electric. Schedule CBP (Capacity Bidding Program). Accessed: Feb. 1st, 2020. [Online]. Available: https://www.sdge.com/sites/default/files/Capacity\%20Bidding\%20Program\%20Tariff_0.pdf.