Real-world challenges for multi-agent reinforcement learning in grid-interactive buildings

Kingsley Nweye, Bo Liu, Peter Stone and Zoltan Nagy

ARTICLE INFO
Keywords: grid-interactive buildings benchmarking reinforcement learning

ABSTRACT
Building upon prior research that highlighted the need for standardizing environments for building control research, and inspired by recently introduced challenges for real life reinforcement learning (RL) control, here we propose a non-exhaustive set of nine real-world challenges for RL control in grid-interactive buildings (GIBs). We argue that research in this area should be expressed in this framework in addition to providing a standardized environment for repeatability. Advanced controllers such as model predictive control (MPC) and RL control have both advantages and disadvantages that prevent them from being implemented in real-world problems. Comparisons between the two are rare, and often biased. By focusing on the challenges, we can investigate the performance of the controllers under a variety of situations and generate a fair comparison. As a demonstration, we implement the offline learning challenge in CityLearn, an OpenAI Gym environment for the easy implementation of RL agents in a demand response setting to reshape the aggregated curve of electricity demand by controlling the energy storage of a diverse set of buildings in a district, and study the impact of different levels of domain knowledge and complexity of RL algorithms. We show that the sequence of operations utilized in a rule-based controller (RBC) used for offline training affects the performance of the RL agents when evaluated on a set of four energy flexibility metrics. Longer offline learning from an optimized RBC leads to improved performance in the long run. RL agents that learn from a simplified RBC risk poorer performance as the offline learning period increases. We also observe no impact on performance from information sharing amongst agents. We call for a more interdisciplinary effort of the research community to address the real-world challenges, and unlock the potential of GIB controllers.

1. Introduction
Buildings account for approximately 40% of the global energy consumption and 30% of the associated greenhouse gas emissions, while also offering a 50–90% CO₂ mitigation potential [26]. Optimal decarbonization requires electrification of end-uses and concomitant decarbonization of electricity supply, efficient use of electricity for lighting, heating, ventilation and air conditioning (HVAC), and domestic hot water (DHW) generation, and upgrade of the thermal properties of buildings [24]. A major driver for grid decarbonization is the integration of renewable energy systems (RES) into the grid (supply) and, photovoltaic (PV) systems and solar-thermal collectors into residential and commercial buildings (demand), electric vehicles (EVs), with their storage capacity and inherent connectivity, hold a great potential for integration with buildings [27]. However, this grid-building integration must be carefully managed during operation to ensure reliability and stability of the grid [39, 14, 7] (Fig. 1).

Demand response (DR) as an energy-management strategy enables end-consumers to provide the grid with more flexibility by reducing their energy consumption through load curtailment, shifting their energy consumption over time, or generating and storing energy at certain times (Fig. 1). In exchange, consumers typically receive a reduction of their energy bill [35]. HVAC can contribute to load curtailment events by modifying the temperature set points, participating in load shifting by pre-heating or pre-cooling the buildings [3] (passive energy storage), or by directly storing thermal energy in an energy storage system (active energy storage). Thermostats with DR functionality can provide energy savings to residential customers by allowing electricity retailing companies to adjust set-points during peak-demand events. Widespread integration of communication technologies allows all involved systems (PV, HVAC, storage, EV, thermostats, etc.) to exchange information on their operation, leading to the concept of smart cities, allowing cities to achieve energy savings, and become more sustainable [5].

Advanced control systems can be a major driver for DR by automating the operation of energy systems, while adapting to individual characteristics of occupants and buildings. However, for DR to be effective, loads must be controlled in a responsive, adaptive and intelligent way. When all the electrical loads react simultaneously to the same price signals, aggregated electricity peaks could be shifted rather than shaved. Therefore, there is a need for more efficient
and effective ways of coordinating the response of all the technologies described above.

Advanced control algorithms such as MPC [10] and deep RL [41] have been proposed for a variety of building control applications. While both methods have their disadvantages, e.g., MPC requiring a model while RL being data intensive, spectacular applications and results have been presented in the past several years. In addition, recently, hybrid methods, based on physics constrained neural networks for models of cooperation or competition between agents which are dimensionally state and action spaces that require a notion of cooperation or competition between agents which are common characteristics of GIBs. With MARL, grid-level objectives such as peak shaving and ramp reduction can be optimized.

A major challenge for RL in DR is the ability to compare algorithm performance [39]. As argued in [43], a shared collection of representative environments [needs to be established in order to] systematically compare and contrast [...] building optimization algorithms.

Building on [43], and inspired by [13, 12], the purpose of this paper is twofold. First, we introduce and discuss specific real world challenges for GIBs that our community should be focusing on. Second, we demonstrate one particular challenge using the CityLearn gym environment [36].

This paper is organized as follows. Section 2 presents nine real world challenges for GIBs, while Section 3 provides background on RLs and CityLearn. In Section 4, we provide a framework towards addressing one of the introduced challenges and present our results from addressing said challenge using a case study data set. A discussion of the results and conclusion follow in Section 5 and Section 6.

2. Real-world challenges

Dulac-Arnold et al. provide nine real-world challenges for RL in [13] and prescribe a suite of environments in [12] that may be used to benchmark algorithms which, address the challenges. The environments in [12] are not suitable to evaluate GIBs, as they are based on small scale environments without the necessary domain knowledge or context. In the following, we present the nine challenges, in the context of GIBs and provide the description of [12] in italics.

C1: Being able to learn on live systems from limited samples: In this challenge, the controller is initialized randomly and has to learn to perform only based on the samples it observes. The sample size can be artificially reduced by presenting the controller only with a subset of the data, e.g., every three hours instead of every 15 min. The algorithms can be evaluated on how quickly in terms of time or sample number they converge, and how stable their exploration is. Conversely, we can evaluate the trade-off between data requirement and controller performance.

C2: Dealing with unknown and potentially large delays in the system actuators, sensors, or feedback The thermal dynamics of buildings are such that the effects of controller actions to adjust the HVAC systems are observed in delays. This has implications for, e.g., pre-cooling/heating of buildings to take advantage of the thermal mass of buildings. The controller need to implicitly and automatically learn the dynamics of the building. Challenge data sets with different thermal mass from light to heavy should be created, and the converged controller should be compared to understand the relationship between longer delays in feedback (higher thermal mass) and controller performance.

C3: Learning and acting in high-dimensional state and action spaces. This challenge addresses the scalability of a proposed controller. As buildings can inherently
have a large state-action space, controllers can be evaluated on specific subsets of them to understand how the performance changes. In the case of controlling multiple buildings (or multiple zones within a building), scalability refers to essentially increasing the number of buildings (or zones) and observing the control performance.

**C4:** Reasoning about system constraints that should never or rarely be violated. This is a central challenge, as building control problems are indeed often presented as balancing between reducing energy use while maintaining comfortable conditions. Other constraints in the energy system are operational, such as ensuring a minimum state of charge, maintaining operational temperatures within limits, etc. The algorithms should be evaluated on both the number of violations during the learning process and for the converged policy. Integration of constraint violation into the objective function is addressed in C6 below.

**C5:** Interacting with systems that are partially observable, which can alternatively be viewed as systems that are non-stationary or stochastic. This challenge has two parts. In the first part, observations can be modified to contain failures (sensor noise, missing data, etc.), which can be common in any real life systems, like buildings and HVAC systems. We then observe the performance of the algorithms for various levels of the failures (more noise, more missing data). In the second part, we can observe how a controller performs on a perturbed system. Perturbations can consist of retrofit measures on buildings (improving envelop or windows), improving equipment, changed occupant behavior or different climate. We can then judge the algorithms on their ability to perform their previously learned policy on the perturbed system.

**C6:** Learning from multiple or poorly specified objective functions. Energy management in buildings is inherently multi-objective, especially when considering multiple zones or multiple buildings. Another example is when there is a global objective (overall building energy use) as well as multiple local objectives (equipment operation). As mentioned in C4, constraints can be incorporated into the objective function directly. When evaluating the controller performance, the individual objectives should be separated to allow for a fair comparison.

**C7:** Being able to provide actions quickly, especially for systems requiring low latencies. Latency is a delay in executing a control action after acquiring a measurement due to long computational time. Latencies in real life systems can occur if the system dynamics are fast or computational times are long. A practical example for smart buildings and micro-grids is if the computation is taking place in the cloud, adding also data transfer to the execution time, which can be exacerbated by connectivity issues. To observe the impact of latency, time-step delays of various lengths should be included into the control execution and the impact on their performance should be evaluated.

**C8:** Training off-line from the fixed logs of an external behavior policy. The challenge here is to learn a control law from data generated by a suboptimal reference controller, e.g., an RBC, which is often available, essentially a system log. In addition to the control environment, data sets of various sizes, e.g., two weeks, one month and six months should be provided that are generated with a known reference RBC. Then, the controllers can be evaluated on the ability to improve these baselines.

**C9:** Providing system operators with explainable policies. Here we deviate from the description in [12] who propose to generate figures to improve the interpretability of the results. Rather, for the building context, what is needed is that the control actions can be explained simply to building managers. Advances in explainable artificial intelligence (AI) are needed, and algorithms that might perform suboptimally, yet are easier to explain are favored as they are more likely to get accepted, and thus implemented. A consensus between modelers and system operators on the standards and outcomes of a control law could be established to facilitate effective communication amongst invested parties.

Each of the aforementioned challenges require unique experimental designs within a simulation environment to adequately study and quantify the factors that affect their resolution. We demonstrate challenge C8 using the CityLearn environment [36] in Section 4.

### 3. Background

We provide a background on RL and MARL. Detailed introductions can be found in standard textbooks [34].

#### 3.1. Reinforcement Learning

In RL, an agent interacts with an environment to maximize the reward it receives. RL is usually formulated as an MDP. An MDP $\mathcal{M}$ is a tuple $\mathcal{M} = (S, A, T, \gamma, R)$. $S$ and $A$ are the state and action spaces for the agent. At time step $t$, the agent is located at a state $s_t \in S$. After taking an action $a_t \in A$, the agent will be transitioned to the next state $s_{t+1} \sim T(\cdot | s_t, a_t)$, where $T$ denotes the transition probability and is usually hidden from the agents. Moreover, the agent receives a scalar reward $r_t \sim R(s_t, a_t)$. The overall objective of RL is to find a policy $\pi: S \rightarrow A$ that maximizes the expected cumulative return:

$$\max_{\pi} \mathbb{E}_{s_t,a_t \sim \pi(\cdot | s_t)} \left[ \sum_{t=0}^{\infty} \gamma^t r_t \right]. \quad (1)$$

It has been shown that given any stationary policy $\pi$, the above objective will converge to a value based on which state
the agent starts from. Specifically, we have the value of a policy defined as:

\[ V^\pi(s) = \mathbb{E}_{s_0 \sim s} \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right] \tag{2} \]

where \( r(s_t, a_t) = r_t - R(s_t, a_t) \) and we use \( \mathbb{E}_\pi \) to denote that the expectation is taken over the trajectories sampled from the policy \( \pi \). Similarly, we can define the action-value function:

\[ Q^\pi(s, a) = \mathbb{E}_{s_0 \sim s, a_0 \sim a} \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right] \tag{3} \]

The RL objective in Eq. (1) is therefore equivalent to

\[ \max_{\pi} V^\pi(s), \forall s. \tag{4} \]

To optimize the above objective, there are typically two types of RL algorithms: value-based and policy-based. The value-based algorithms are based on the well-known Bellman equation of the action-value function. Denote the optimal action-value function as \( Q^* \), then it is known that for \( Q^* \), it satisfies

\[ Q^*(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim T(\cdot|s, a), a'} \max_{a'} Q^*(s', a'). \tag{5} \]

By minimizing the difference between the left and right-hand sides of the above equation, we reach the Q-learning algorithm [42].

### 3.2. Multiagent Reinforcement Learning

MARL extends RL to the setup involving multiple agents. The general MARL framework includes the cooperative setup, the competitive setup and the mixture of the two. In this work, we focus on the cooperative setup because the main objective is to coordinate buildings to flatten the electricity demand curve, which is a shared objective for all agents. To summarize, the MARL problem we consider in this work is also formulated as an MDP represented by the tuple \( M = (S, A, T, \gamma, R) \). The major differences are: 1) the action space now includes the joint actions of all agents, i.e. \( A = A_1 \times A_2 \times \cdots \times A_n \), where \( A_i \) is the action space of the \( i \)-th agent. 2) the state space \( S = S^1 \times S^2 \times \cdots \times S^n \), where \( S^i \) is the observation of the \( i \)-th agent. The pipeline of RL and MARL are summarized in Fig. 2. We refer the reader to [25] for a comprehensive discussion on RL and MARL algorithms.

In principle, a multi-agent problem can be regarded as a single-agent problem where a centralized agent chooses actions for all agents. However, it is both computationally expensive and costly to deploy and train a centralized agent in practice, as the state and action space grow dramatically with the number of agents [33]. A centralized control architecture also, decreases the robustness of the system to malicious attacks [45]. Therefore, decentralized algorithms that learn a decision module for each agent is a more practical approach. On the other hand, a fully decentralized algorithm where agents are not aware of other agents’ policies might result in poor coordination.

#### 3.3. CityLearn

CityLearn is an OpenAI Gym environment for the easy implementation of RL agents in a DR setting to reshape the aggregated curve of electricity demand by controlling the energy storage of a diverse set of buildings in a district [36, 37, 16]. Its main objective is to facilitate and standardize the evaluation of RL agents, such that it can be used to benchmark different algorithms. CityLearn includes energy models of air-to-water heat pumps, electric heaters, chilled water (CHW), DHW and electricity energy storage devices as shown in Fig. 3. In each building, the air-to-water heat pump is used to meet the cooling demand and an electric heater is used to meet DHW heating demand. Buildings could also possess a combination of CHW, DHW and electricity storage devices to offset cooling. DHW heating and electricity demand from the grid. CHW and DHW storage capacities are represented as a multiple of the hours the storage devices can satisfy the maximum annual cooling or DHW demand if fully charged. All these devices, together with other electric equipment and appliances (non-shiftable loads) consume electricity from the main grid. PV systems may be included in the buildings’ energy systems to offset part of this electricity consumption by allowing the buildings to generate their own electricity.

The RL agents control the storage of CHW, DHW and electricity by deciding how much cooling, heating and electrical energy to store or release at any given time. CityLearn guarantees that, at any time, the heating and cooling energy demand of the building are satisfied regardless of the actions of the controller by utilizing pre-computed energy loads of the buildings, which include space cooling, dehumidification, appliances, DHW, and solar generation. The backup controller guarantees that the energy supply devices prioritize satisfying the energy demand of the building before storing any additional energy.

CityLearn has been used extensively as a reference environment to demonstrate incentive-based DR [6], collaborative DR [17], coordinated energy management [30, 22], or benchmarking RL algorithms [9, 32].
Table 1
Chilled water (CHW), domestic hot water (DHW) and electricity (ELE) storage and photovoltaic (PV) system capacities per building. The unit of measurement for CHW and DHW storage capacities is the hours of maximum annual hourly cooling and DHW demand that can be satisfied on full charge.

<table>
<thead>
<tr>
<th>ID</th>
<th>CHW Stg. (h)</th>
<th>DHW Stg. (h)</th>
<th>ELE Stg. (kWh)</th>
<th>PV (kW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>140</td>
<td>120</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>3</td>
<td>80</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>0</td>
<td>50</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1.5</td>
<td>0</td>
<td>75</td>
<td>40</td>
</tr>
<tr>
<td>5</td>
<td>3.5</td>
<td>1.5</td>
<td>50</td>
<td>25</td>
</tr>
<tr>
<td>6</td>
<td>1.5</td>
<td>3</td>
<td>30</td>
<td>20</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>2</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>3</td>
<td>30</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>3</td>
<td>35</td>
<td>0</td>
</tr>
</tbody>
</table>

4. Offline Learning Challenge (C8)

Here, we provide a framework for studying C8. Specifically, we compare two RL control approaches, (1) independent, uncoordinated soft actor-critic (SAC) agents (see Section 4.2.1), and (2) the MARLISA algorithm for coordinating the agents (see Section 4.2.2) in the CityLearn environment using the nine-building data set described in Section 4.1. We investigate the agents’ behavior with respect to varied periods of offline training on an RBC. Our central hypothesis is that a longer offline training period results in better performance, since the agents will have more existing knowledge of what ideal actions could resemble by the time they come online.

4.1. Data Set

We use the CityLearn Challenge 2021 data set, [40]. It consists of nine Department of Energy (DOE) prototype buildings: one medium office (ID=1), one fast-food restaurant (ID=2), one standalone retail (ID=3), one strip mall retail (ID=4), and five medium multifamily buildings (ID=5–9) [8]. The energy demand for each building has been pre-simulated in EnergyPlus using 2014–2017 actual meteorological year weather data for Austin, TX. Their cooling, DHW and electricity storage capacities, as well as PV capacities, are shown in Table 1.

4.2. Agent & Reward Design

4.2.1. Independent SAC agents

To control environments that have continuous states and actions, tabular Q-learning is not practical, as it suffers the curse of dimensionality. Actor-critic RL methods use artificial neural networks to generalize across the state-action space. The actor network maps the current states to the actions that it estimates to be optimal. Then, the critic network evaluates those actions by mapping them, together with the states under which they were taken, to the Q-values.

SAC is a model-free off-policy RL algorithm [18]. As an off-policy method, SAC can reuse experience and learn from fewer samples. SAC is based on three key elements: an actor-critic architecture, off-policy updates, and entropy maximization for efficient exploration and stable training.

SAC learns three different functions: the actor (policy), the critic (soft Q-function), and the value function \( V \). For more details about SAC, we refer the reader to [19].

\[
r_{i}^{SAC}(t) = \min(0, e_{i}(t))
\]  

(6)

The network architecture and algorithm hyperparameters utilized in the SAC agent are summarized in Table 2 and are the provided CityLearn defaults for the data set being used.

We use the reward \( r_{i}^{SAC}(t) \) (Eq. (6)) for the independent SAC RL agents. It is a single-agent reward whose value only depends on the net electricity consumption \( e_{i}(t) \) of the agent \( i \) at time step \( t \). \( e_{i}(t) < 0 \) if the building is consuming more electricity than it generates, and \( e_{i}(t) > 0 \) if the building is self-sufficient at that time and generates excess electricity.

4.2.2. MARLISA RL Agents

MARLISA is built on the SAC algorithm and allows for coordination of the agents through reward sharing, collective...
The agents predict their own future electricity consumption and share this information with each other, following a leader-follower schema. In an iterative process, each agent converges to selecting an action before the action is implemented.

\[ r_i^{\text{MARL}}(t) = -\text{sign}(e_i(t)) \cdot 0.01 \cdot e_i(t)^2 \cdot \min \left( 0, \sum_{i=0}^{n} e_i(t) \right) \] (7)

The same network architecture and algorithm hyperparameters utilized in the SAC agents and described in Table 2 are used in the MARLISA agents.

4.2.3. RBC

We assumed no detailed knowledge of the energy profile of each building and developed two variations of RBC sequence of operation (SOO) where, RBCBasic (Algorithm 1) mimics a simplified logic and RBCOptimized (Algorithm 2) is informed by domain knowledge. For both SOOs, the input is the hour of the day, \( h \) and time step, \( t \) and, the output is the charge/discharge action, \( a(t) \) for chilled water, DHW or electricity storage. The action values in Algorithm 1 are arbitrarily chosen to mimic a poorly tuned controller while those in Algorithm 2 are selected by performing a grid search on different combinations of hourly values to determine a combination that provides the best performance when evaluated on the metrics presented in Section 4.5. The RBCs are tuned to act greedily in every building and use the storage capacity to reduce energy consumption by storing more energy during the night (when the coefficient of performance of the heat pumps is higher) and release it during the day. We also use the RBC to normalize the RL agents’ performance metrics.

**Algorithm 1: RBCBasic sequence of operation.**

Input: \( h, t \)
Output: \( a(t) \)
if \( 1 \leq h \leq 6 \) then
\[ a(t) = 0.05532; \]
else if \( 7 \leq h \leq 15 \) then
\[ a(t) = -0.044; \]
else if \( 16 \leq h \leq 18 \) then
\[ a(t) = -0.02; \]
else if \( 19 \leq h \leq 22 \) then
\[ a(t) = -0.024; \]
else
\[ a(t) = 0.034; \]
end

4.3. Action-Space Design

The action space per building is determined by the number of available energy storage systems to control, including the CHW, DHW and electricity storage systems. Hence, the action space is bounded at \( n = 3 \) for a district of \( n \) buildings that each possess the three storage systems. The action value is bounded between -1 and 1 where positive and negative values are charge and discharge control actions respectively.

4.4. State-Space Design

The available state space is made up of 27 observable temporal, weather, district, and building variables which, are summarized in Table 3. The storage system state of charge (SOC) states are conditionally available in each building. Meanwhile, the RBC controllers utilize only the hour state in determining the control action. The states are transformed to aid the learning process by applying cyclical transformation to the month and hour states, one-hot encoding to the day state and min-max normalization to all other states.

4.5. Performance Metrics/Cost Functions

We evaluate the agents’ performance on a set of cost functions that quantify the collective district’s energy flexibility as follows:

**Average Daily Peak** is the average of all the daily peaks of the 365 days of the year and is calculated using the net energy demand of the whole district of buildings defined as

\[ \text{ADP} = \frac{\sum_{d=0}^{364} \max(Q_{d(1+i)}, \ldots, Q_{d(1+d+i-1)})}{365} \] (8)

where \( d \) is the day of the year and \( i \) is the number of time steps in a day. In our application, \( i = 24 \) for an hourly resolution.

**Load Factor** is the difference between 1 and the ratio of average monthly demand to monthly peak demand defined as

\[ 1 - \text{Load Factor} = \left( \sum_{m=0}^{11} 1 - \frac{\sum_{k=m+1}^{k=x(m+1)-1} Q_t}{k \times \max(Q_t, \ldots, Q_{k(x(m+1)-1)})} \right) \times \frac{1}{12} \] (9)
**Table 3**
The unified state space for all agents.

<table>
<thead>
<tr>
<th>State</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Temporal</strong></td>
<td></td>
</tr>
<tr>
<td>Month</td>
<td>-</td>
</tr>
<tr>
<td>Day</td>
<td>-</td>
</tr>
<tr>
<td>Hour</td>
<td>-</td>
</tr>
<tr>
<td><strong>Weather</strong></td>
<td></td>
</tr>
<tr>
<td>Dry-bulb temperature</td>
<td>°C</td>
</tr>
<tr>
<td>Dry-bulb temperature (+6 hr)</td>
<td>°C</td>
</tr>
<tr>
<td>Dry-bulb temperature (+12 hr)</td>
<td>°C</td>
</tr>
<tr>
<td>Dry-bulb temperature (+24 hr)</td>
<td>°C</td>
</tr>
<tr>
<td>Relative humidity (6 hr)</td>
<td>%</td>
</tr>
<tr>
<td>Relative humidity (12 hr)</td>
<td>%</td>
</tr>
<tr>
<td>Relative humidity (24 hr)</td>
<td>%</td>
</tr>
<tr>
<td>Diffuse solar</td>
<td>W/m²</td>
</tr>
<tr>
<td>Diffuse solar (6 hr)</td>
<td>W/m²</td>
</tr>
<tr>
<td>Diffuse solar (12 hr)</td>
<td>W/m²</td>
</tr>
<tr>
<td>Diffuse solar (24 hr)</td>
<td>W/m²</td>
</tr>
<tr>
<td>Direct solar</td>
<td>W/m²</td>
</tr>
<tr>
<td>Direct solar (6 hr)</td>
<td>W/m²</td>
</tr>
<tr>
<td>Direct solar (12 hr)</td>
<td>W/m²</td>
</tr>
<tr>
<td>Direct solar (24 hr)</td>
<td>W/m²</td>
</tr>
<tr>
<td><strong>District</strong></td>
<td></td>
</tr>
<tr>
<td>Net electricity consumption</td>
<td>kWh</td>
</tr>
<tr>
<td>Carbon intensity</td>
<td>kg CO₂/kWh</td>
</tr>
<tr>
<td><strong>Building</strong></td>
<td></td>
</tr>
<tr>
<td>Indoor dry-bulb temperaure</td>
<td>°C</td>
</tr>
<tr>
<td>Indoor relative humidity</td>
<td>%</td>
</tr>
<tr>
<td>Non-shiftable load</td>
<td>kWh</td>
</tr>
<tr>
<td>Solar generation</td>
<td>W</td>
</tr>
<tr>
<td>Chilled water stg. SOC</td>
<td>-</td>
</tr>
<tr>
<td>Domestic hot water stg. SOC</td>
<td>-</td>
</tr>
<tr>
<td>Electricity stg. SOC</td>
<td>-</td>
</tr>
</tbody>
</table>

where \( Q_t \) is the net electric consumption at time step \( t \) in the \( m \)th month and \( k \) is the total number of time steps per month. \( k = 730 \) in our application where we use an hourly time step resolution.

**Net Electricity Demand** is given by

\[
\text{Net Electricity Demand} = \sum_{t=0}^{n-1} \max(0, Q_t) \quad (10)
\]

i.e., the sum of positive net electricity demand because the objective is to minimize the energy consumed in the district, not to profit from the excess generation, i.e., island operation is incentivized.

**Ramping** is the difference in net electric consumption at two consecutive time steps defined as

\[
\text{Ramping} = \sum_{t=1}^{n-1} |Q_t - Q_{t-1}| \quad (11)
\]

where \( Q_t \) is the net electric consumption at time step \( t \) and \( n \) is the total number of time steps such that \( 0 \leq t < n \).

### 4.6. Experimental Design

We vary the offline training period and the RBC SOO during offline training to test our hypothesis. For one training episode, the initial 744 (two weeks), 4,344 (six months) or 8,760 (one year) time steps of states are used for offline training of the RL algorithms while selecting actions from either RBCBasic or RBCOptimized algorithms before switching online to train on actions selected from the SAC or MARLISA agents algorithms for the remainder of the 35,040 time steps (4 years). Hence, the RL agents considered in totality include:

1. \( \text{SAC}_{\text{RBC Basic}} \)
2. \( \text{SAC}_{\text{RBC Optimized}} \)
3. \( \text{MARLISA}_{\text{RBC Basic}} \)
4. \( \text{MARLISA}_{\text{RBC Optimized}} \)

With these combinations, we study the impact of simpler vs comparatively more complex algorithms (independent SAC vs MARLISA) and the value of less or more detailed domain knowledge (\( \text{RBC Basic} \) vs \( \text{RBC Optimized} \)).

The simulations are run for one epoch, where an epoch is a period of 35,040 time steps that represent the number of hours in years 2014–2017. We simulate each combination of offline training period and RL agent three times in CityLearn using the nine-building data set described in Section 4.1, initialized with different random seeds. The results are averaged over the three runs.

The source code used to produce this work is available in [15].

### 4.7. Results

#### 4.7.1. Performance Metrics

Fig. 4 shows the distinction in the district level performance metrics when all storage systems are controlled by either RBCBasic or RBCOptimized during the entire four-year simulation period. RBCOptimized outperforms RBCBasic when evaluated on the average daily peak, load factor and ramping metrics. Both RBC algorithms perform similarly in terms of net electric consumption with RBCOptimized achieving very little advantage in minimizing the net electric consumption in the long run.

Fig. 5 shows the district level performance metrics for the varied offline training periods and RL agents outlined in Section 4.6. The metrics are normalized with respect to the RBC used for offline training (dashed black line), where superior and inferior performance of the RL agents is indicated by values less than one and values greater than one, respectively. The detailed domain knowledge of RBCOptimized causes superior performance compared to both SAC RBCOptimized and MARLISA RBCOptimized agents. Consequently, longer offline training with RBCOptimized results in delayed convergence but better performance in the long run. On the other hand, the simplified SOO utilized in RBCBasic leads to inferior performance compared to the RL agents, such that longer trained RL agents suffer from poorer performance compared to those trained for a shorter period. The net electric consumption for RBCOptimized-trained RL...
agents is noteworthy, as variations in offline training period show negligible difference in performance. Interestingly, the shortest offline training period of 2 weeks results in an initially large improvement in the net electric consumption metric immediately after the RL agents come online but within the first year, worsens and approaches the lower performance six-month and one-year trained agents.

Between the SAC and MARLISA RL algorithms, average daily peak and load factor are unaffected by algorithm complexity when the agents are trained using the same RBC. The ramping metric for MARLISA$_{\text{RBC Optimized}}$ shows poor initial performance for shorter offline training periods, but improves over time. In comparison, the SAC$_{\text{RBC Optimized}}$ agents are able to maintain nearly the same ramping performance as RBC$_{\text{Optimized}}$.

4.7.2. District Electricity Consumption

In Fig. 6, we show the district’s net electric consumption profile for the four offline trained RL agents, as well its electric consumption without PV installation and energy storage control for a selected period. The 2014 profile is the following seven days after offline training for six months and, the same period is shown in 2015. In 2014, the two-week and six-month trained RL agents are already online while the agents trained for one year are still being trained offline hence, represents net electric consumption under RBC control. For each RL agent, the six-month trained agents behave like the two-week trained agents immediately after coming online and as a result both variations of training period have the same net electric consumption six months into the simulation. For all RL agents in 2014, the one-year training setup still offline has higher net electric consumption early in the morning and late at night, but lower net electric consumption during midday compared to already online scenarios. By the same period in 2015, the net electric consumption profile is almost equal irrespective of RBC domain knowledge, RL algorithm complexity and offline training period. Overall, in comparison to the baseline i.e. no control and PV, there is significant energy flexibility in the form of peak shaving provided by solar generation and control of energy storage systems between late morning and afternoon.

5. Discussion

5.1. Advanced Building Controllers

Advanced building controllers are needed to improve upon the industry standard of pre-determined set-points, that do not take into account predictions or allow optimizing the operational sequence [41]. MPC has been developed in the petrochemical industry in the 1970s and applied across many industries since then [28], MPC requires the development of a mathematical model for the plant to be controlled, which works well for replicable systems (cars, planes). The uniqueness of buildings and their energy systems, and the engineering costs incurred when developing and calibrating a model made it such that, despite all advances, MPCs have not been adopted in the building industry [23, 31]. RL algorithms have been considered to address the shortcomings of MPC by potentially being model-free. However, RL approaches can be more data intensive and more time-consuming compared to MPC approaches. Comparisons, if even performed, are often biased toward one type of algorithm, and therefore relatively meaningless. The challenges introduced here specifically focus on the breadth of applications rather than on one specific problem. This allows for a fair comparison. Of course, while we argue in the context of RL, the challenges can be used for comparisons between algorithm classes.

A promising approach in MARL is centralized training with decentralized execution (CTDE), CTDE assumes that the learning of each agent’s policy can depend on the global state (the aggregation of all agents’ observation in our case), but during executing, agents work independently. By doing so, it is possible for the agents to cooperate according to some learned heuristics so that during execution they do not need to know what others’ observations are. A CTDE version of MARLISA has been found to provide more smooth trajectories compared to the basic MARLISA algorithm [17]. Of course, advances in algorithm complexity must be weighted against data and communication requirements and potential privacy issues.

5.2. Environment Standardization

We emphasize the need for standardizing computational environments, such as the COMprehensive Building simulator (COBS) [46], Sinergym [21], BOPTEST [1], the Advanced Controls TestBed (ACTB), or CityLearn [36] using a common interface, e.g., OpenAI Gym [2], and releasing data sets and implementations open source. This can help spark a development rush similar to the one that the ImageNet data set sparked for the deep learning community [20]. However, in contrast to ImageNet’s development, a more in-depth collaboration and exchange between researchers in the built environment and computer science would be beneficial to transfer domain knowledge from buildings to controller design on the one hand and facilitate transitioning theoretical findings of algorithms into practice on the
Figure 5: Energy flexibility performance metrics evaluated on results from CityLearn simulations of varied offline training period and RL agents. Offline training periods include 744 (two weeks), 4344 (six months), 8760 (one year) time steps indicated by the blue, orange, and green lines respectively. The RL agents include SAC\textsubscript{RBC\_Basic}, SAC\textsubscript{RBC\_Optimized}, MARLISA\textsubscript{RBC\_Basic}, MARLISA\textsubscript{RBC\_Optimized}. Each metric is normalized with respect to the RBC used for offline training (dashed black line) which is indicated in the subscript of the RL agent’s name.

Figure 6: Comparison between electricity consumption without control and PV (dashed black line) and net electricity with RL agent-controlled CHW, DHW and electric storage systems and, PV at the district level for varied offline training periods. The offline training periods include 744 (two weeks), 4344 (six months), 8760 (one year) timesteps indicated by the blue, orange, and green lines respectively. The RL agents include SAC\textsubscript{RBC\_Basic}, SAC\textsubscript{RBC\_Optimized}, MARLISA\textsubscript{RBC\_Basic}, MARLISA\textsubscript{RBC\_Optimized}. The following seven days after six months of simulation in 2014 are shown (left) and the same time period in the subsequent year of 2015 is shown (right).
other. Common venues or guest invitation to each other’s venues could be established: ACM’s BuildSys/e-energy and the ASHRAE/IBPSA communities should explore common pathways for knowledge exchange to ultimately unlock the built environment’s potential to reduce greenhouse gas emissions.

5.3. Offline Learning Challenge (C8)

Our central hypothesis in addressing C8 is that a longer offline training period results in better performance, since the agents will have more existing knowledge of what ideal actions could resemble by the time they come online. We find this hypothesis to be true and governed by certain design choices. Our experiments reveal that the SOO utilized in the RBC for offline training determines the performance of the RL agents when evaluated on a set of energy flexibility metrics. Longer offline training from an optimized RBC will lead to slower convergence upon coming online, but superior energy flexibility in the long run. RL agents that learn from a simplified RBC risk poorer performance as the offline learning period increases.

The optimized RBC is able to significantly outperform the RL controllers in reducing the district average daily peak, load factor and ramping. This shows significant energy flexibility potential from improving existing RBCs in practice over installation of more complex controllers. Nevertheless, RBC systems are unable to respond to perturbations in the control environment (C5), an ability RL controllers possess, which may affect the overall performance of the controller in satisfying the control objective. We shall address C5 in our future work.

We do not observe any significant differences between the performance of the SAC and MARLISA RL algorithms when evaluated on the four performance metrics. This suggests that the simpler SAC algorithm is sufficient and the added complexity and cost of information sharing amongst agents could be avoided.

Our experiments show negligible difference in net electricity consumption irrespective of offline learning period, RBC SOO and RL algorithm complexity. We provide an explanation in the context of the RBC design. Both RBCBasic and RBCOptimized are designed to charge the storage systems at night and early in the morning to take advantage of higher heat pump coefficient of performance (COP). Their logic can also be beneficial in a residential DR program that incentivizes electricity consumption during periods of lower demand. However, in the absence of such DR setup in the simulation environment, this design is most beneficial to the CHW storage whose energy is delivered by a heat pump. The DHW and electrical storage charging demands are directly met by the grid and offset by available solar generation. Solar generation is intermittently available during the day, hence, these storage devices could potentially benefit from ‘free’ charging during the RBC’s hours of discharge control action.

A challenge that is presented in offline training is the possibility of a homogeneous offline data set that is non-exploratory which, may lead to a non-generalized policy that performs poorly on live systems. The work by Yarats et al. highlights the importance of the diversity of exploratory data used in offline training on the performance of the RL agents [44]. Our results corroborate this observation as longer offline training on the fixed log of a tuned RBC yields preferable results.

6. Conclusion

We have introduced a set of challenges to study real world GIBs. While there are many research challenges that remain in this realm, we highlight the need for an organized move forward of the community in addressing both fundamental computational challenges, but in a way that applies to the larger problems in the built environment. As an example, we studied the off-line learning challenge (C8) for two levels of domain knowledge, RL algorithm complexity and four performance metrics. It is not our intention to imply that the list above is an exhaustive list of challenges. Rather, by highlighting typical real world problems, our aim is to inspire researchers to define and share their environments and the problems they are addressing with these challenges as a standard framework.

Acronyms

AI artificial intelligence.
CHW chilled water.
COP coefficient of performance.
CTDE centralized training with decentralized execution.
DHW domestic hot water.
DOE Department of Energy.
DR demand response.
EV electric vehicle.
GIB grid-active building.
HVAC heating, ventilation and air conditioning.
MARL multi-agent reinforcement learning.
MDP Markov Decision Process.
MG Markov Game.
MPC model predictive control.
PV photovoltaic.
RBC rule based controller.
RES renewable energy system.

Nweye et al.: Preprint submitted to Elsevier
RL reinforcement learning.

SAC soft actor-critic.

SARL single-agent reinforcement learning.

SOC state of charge.

SOO sequence of operation.

References


Highlights

Real-world challenges for multi-agent reinforcement learning in grid-interactive buildings
Kingsley Nweye, Bo Liu, Peter Stone, Zoltan Nagy

- Proposing real-world control challenges for grid-interactive buildings.
- Algorithms should be compared on their performance on these challenges.
- Studying off-line learning challenge using CityLearn.
- The performance of RL controller depends strongly on the quality of the RBC controller.
Graphical Abstract

Real-world challenges for multi-agent reinforcement learning in grid-interactive buildings

Kingsley Nweye, Bo Liu, Peter Stone, Zoltan Nagy

<table>
<thead>
<tr>
<th>9 Real-World Challenges</th>
<th>CityLearn Environment for benchmarking</th>
<th>Challenge C8 demonstration</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1: Learn from live systems</td>
<td></td>
<td>An advanced RL agent (MARLISA)</td>
</tr>
<tr>
<td>C2: Deal with unknown delays</td>
<td></td>
<td>- can improve a bad rule based controller (RBC)</td>
</tr>
<tr>
<td>C3: Learning in high-dimensional spaces</td>
<td></td>
<td>- cannot improve an optimized RBC</td>
</tr>
<tr>
<td>C4: Full system constraints</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C5: Partially observable systems</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C6: Learning multiple objectives</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C7: Provide quick actions</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>C8: Train off-line from fixed logs</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C9: Provide explainable policies</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Nweye et al, 2022
Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐ The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: