




Article

# An Adaptive Energy Orchestrator for Cyberphysical Systems Using Multiagent Reinforcement Learning <sup>†</sup>

Alberto Robles-Enciso <sup>1,\*</sup> , Ricardo Robles-Enciso <sup>2</sup>  and Antonio F. Skarmeta Gómez <sup>1</sup> 

<sup>1</sup> Department of Information and Communications Engineering, University of Murcia, 30100 Murcia, Spain; skarmeta@um.es

<sup>2</sup> Department of Information Technologies and Communications, Technical University of Cartagena, 30202 Cartagena, Spain; ricardo.robles@edu.upct.es

\* Correspondence: alberto.roblese@um.es

<sup>†</sup> In Proceedings of the ALGO CLOUD, 5 September 2023, Amsterdam, The Netherlands.

## Highlights:

### What are the main findings?

- A proof of concept of a smart energy management system in a smart home. Using the reinforcement learning technique, we optimise energy use, avoiding non-renewable sources as much as possible and predicting the critical moments of lower energy production.
- Multiagent reinforcement learning individually manages each of the smart home services (lights, fridge, etc.) so that it is possible to dynamically switch off or shift the period of operation to different slots depending on the energy production.

### What is the implication of the main finding?

- The use of intelligent techniques to manage the services of a smart home allows optimisation of energy use, exploiting peak times of possibly unused energy generation (e.g., PV solar) while minimising the use of non-renewable (petrol generator) or costly (grid peak time, batteries) energy sources.
- By dynamically determining an optimal schedule for smart home services by prioritising green energy sources, a reduction in carbon emissions is implicitly achieved, since less prioritised (carbon-based) energy sources are exceptionally used.

**Abstract:** Reducing carbon emissions is a critical issue for the near future as climate change is an imminent reality. To reduce our carbon footprint, society must change its habits and behaviours to optimise energy consumption, and the current progress in embedded systems and artificial intelligence has the potential to make this easier. The smart building concept and intelligent energy management are key points to increase the use of renewable sources of energy as opposed to fossil fuels. In addition, cyber-physical systems (CPSs) provide an abstraction of the management of services that allows the integration of both virtual and physical systems in a seamless control architecture. In this paper, we propose to use multiagent reinforcement learning (MARL) to model the CPS services control plane in a smart house, with the purpose of minimising, by shifting or shutdown services, the use of non-renewable energy (fuel generator) by exploiting solar production and batteries. Furthermore, our proposal dynamically adapts its behaviour in real time according to current and historic energy production, thus being able to handle occasional changes in energy production due to meteorological phenomena or unexpected energy consumption. In order to evaluate our proposal, we have developed an open-source smart building energy simulator and deployed our use case. Finally, several simulations with different configurations are evaluated to verify the performance. The simulation results show that the reinforcement learning solution outperformed the priority-based and the heuristic-based solutions in both power consumption and adaptability in all configurations.

**Keywords:** Internet of Things; edge computing; cyber-physical system; reinforcement learning; energy management



**Citation:** Robles-Enciso, A.; Robles-Enciso, R.; Skarmeta Gómez, A.F. An Adaptive Energy Orchestrator for Cyberphysical Systems Using Multiagent Reinforcement Learning. *Smart Cities* **2024**, *7*, 3210–3240. <https://doi.org/10.3390/smartcities7060125>

Academic Editors: Guillermo del Campo Jiménez and Asun Santamaría

Received: 25 September 2024

Revised: 24 October 2024

Accepted: 28 October 2024

Published: 29 October 2024



**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

With the rapid evolution of computer systems, high demand, and performance requirements, the manufacture of embedded devices has advanced dramatically over the past decades, reducing costs to historic lows while providing acceptable performance. Low-cost embedded devices have enabled the development of new technologies, greatly encouraging and simplifying the deployment of sensors and devices among us. Together with the successful adoption of embedded systems in our society, new paradigms such as cloud computing, Internet of Things, and edge computing have emerged that aim to modernise [1] the behaviour of digital interactions in our daily habits. In addition, the requirement to enable devices to operate independently has led to the development of new artificial intelligence techniques [2,3] to increase the reasoning capabilities of devices.

One of the approaches of increasing interest in Industry 4.0 is cyber-physical systems [4]. Cyber-physical systems (CPSs) are computer control systems that deeply integrate computation and physical components to provide an abstraction of control components [5–7]. The integration of CPS in smart systems facilitates the optimisation of power consumption [8], which makes them particularly useful for the design of zero-energy buildings [9], smart-grids, and smart-cities [10]. Energy management is a fundamental aspect of all intelligent systems in order to reduce carbon emissions [11]. It is also important to increase the use of renewable energy sources rather than fossil fuels, either by increasing the installation of clean energy sources or by shifting consumption from fossil to clean production schedules. Furthermore, energy optimisation has proved to be an issue of great relevance in Europe recently [12] due to the past energy crisis caused by rising gas prices [13].

The aim of this work is the intelligent energy management of a smart building [14–16], as it is an important objective of the European energy strategy [17]. Among the artificial intelligence methods, reinforcement learning (RL) approaches are widely used as a machine learning control technique [18]. There are several proposals in the literature that use reinforcement learning to manage energy consumption by shifting energy consumption [19,20] to reduce final consumption [21], battery usage [22–24], or energy cost [25], and some of them even use fuzzy reasoning techniques [26,27] and user-preference rules [28]. Nevertheless, there is no proposal that combines both CPS and RL for the energy management and optimisation of a zero-energy isolated house with fixed, shiftable, and optional energy consumptions associated with different services. As discussed in the next section on related works, many articles focus on the large-scale use of their proposals (smart grid, smart building) rather than houses, which neglects an important and promising use case. Moreover, our proposal includes the control of not only typical devices (e.g., HVAC, refrigerator, and lights) but also cyber-physical systems and even direct digital services that have a consumption associated with their operation (e.g., game servers, containers, and dashboards). In addition, as there is no direct connection to the grid, energy decisions are even more critical because the aim is to avoid completely draining the batteries at all costs, in which case, a generator would be used as an emergency measure.

Therefore, the main contributions of this work are the following:

- A smart home use case was designed with a set of services deployed, which include both virtual services and those associated with physical devices.
- Three standard control algorithms for CPS services (basic, priority, and greedy) are presented.
- A control system for CPSs based on reinforcement learning that responds to the status of power generation in real time is proposed.
- An open-source dynamic and extensible smart-building energy simulator was developed to deploy the proposal.
- The performance of the proposal was compared with the basic, priority, and greedy alternatives, showing that the proposed solution is superior in performance and flexibility to the other algorithms.

The rest of this paper is organized as follows. In Section 2, we explore the state of the art of energy management in buildings. In Section 3, we introduce the energy orchestrator with its main components. In Section 4, we present our proposals, including a reinforcement learning approach to energy management. Then, in Section 5, we evaluate the proposed algorithms in different scenarios and analyse the results. Finally, the conclusions and future work are presented in Section 6.

## 2. Background

Energy management in buildings and smart houses plays a crucial role in optimizing power consumption and enhancing overall efficiency, as evidenced by the variety of solutions proposed by researchers. In this section, we present an overview of the state of the art of energy management algorithms in buildings/smart houses, categorizing them based on their underlying methodology.

### 2.1. Data-Analytics-Based Algorithms

Data-based algorithms leverage well-known numerical and data mining techniques such as time series analysis, machine learning, and pattern recognition to extract meaningful insights from the information collected. By analysing historical and real-time data, these algorithms are able to detect simple patterns, enabling accurate forecasting, anomaly detection, and optimisation of energy.

Time series analysis is a classical method used to analyse patterns within a sequence of data points collected over a time interval. Techniques such as autoregressive integrated moving average (ARIMA) models, seasonal decomposition of time series (STL), and Fourier analysis are commonly employed to identify trends, seasonality, and other temporal patterns. In [29], Nepal et al. propose a hybrid model comprising ARIMA and a clustering technique (K-means) to forecast the electrical load of university buildings. They claim that combining clustering and ARIMA increases forecasting performance compared with using ARIMA alone. Also, Panapongpakorn and Banjerdpongchai [30] use ARIMA together with neural networks to implement a complex long-term load forecasting system and compare the performance using statistical criteria of three versions of their hybrid models to determine which is best for a micromanagement system. Furthermore, Di Silvestre and Riva Sanseverino [31] design an optimisation energy storage model using Fourier analysis for optimal power dispatch problems in smart grids and conclude that the results of different applications on a medium-sized system show that the proposed variable representation is quite effective and constitutes the first step towards a new way of formulating variables in dispatch problems.

Model Predictive Control (MPC) is a well-established method for constrained control. It utilizes a mathematical model of the building and its energy systems to predict future consumption and optimize control actions. MPC can consider various parameters, including occupant comfort, weather conditions, electricity prices, and equipment constraints, to achieve optimal performance. One notable work in this area is in [32], which provides an in-depth overview of MPCs and presents a clear and robust framework summarizing the necessary steps to formulate the control problem, especially for the control and management of thermal needed in buildings. In addition, in [33], they use MPC logic, based on weather forecasts, for the analysis of loads in an off-grid domestic system. Their control strategy has the objective of minimising final costs while maintaining optimal environmental comfort in the house, thus optimizing the use of renewable sources. They also compare their system with a standard rule-based system to conclude that their proposal reduces the use of fossil-based power and outperforms the rule-based control. Likewise, Zhou et al. [34] present a similar approach to the previous one, intended for a predictive energy management strategy of a smart community, but in this scenario, they use the information obtained from intelligent transport systems to estimate the upcoming occupancy of the building to anticipate peaks in consumption due to the charging of electric vehicles (EVs).

Support Vector Regression (SVR) is a machine learning-based regression method especially useful for load forecasting. It is particularly effective in handling non-linear relationships between input variables and power consumption. SVR utilizes a subset of training data called support vectors to construct a regression model, enabling accurate prediction of future demand. An example of this is the works in [35,36], which propose the use of SVR to estimate the energy consumption in the first case of a laboratory and in the second case of buildings in a specific region of China. Both papers conclude that with sufficient historical information, SVR is able to identify patterns and accurately predict future demands, enabling effective electricity management and optimisation. In certain cases, such as highly non-linear sample space, the use of standard SVR is not sufficient to build accurate models. For this reason, Zhong et al. [37] propose to modify the common modelling strategy in order to use multiple mappings to transform between feature spaces and finally approach the optimal feature space, since a single mapping cannot guarantee that the corresponding feature space is optimal.

Bayesian networks (BNs) are probabilistic graphical models that utilize directed acyclic graphs (DAGs) to represent and reason about uncertain knowledge and dependencies. In [38], they present a BN technique applied to select the most energy-efficient primary HVAC unit, rather than using a traditional system based on a designer's knowledge and experience. They show that after properly training the BN, the result is similar to traditional systems but with higher efficiency, and the study also proves the feasibility and capability of data-driven building design. Similarly, Shoji et al. [39] briefly describe how to use BNs to learn user behaviour patterns so as to prioritise the operation of some household appliances under power consumption restrictions. Moreover, Amayri et al. [40] propose a Bayesian network-based approach to determine occupancy sensing in buildings. By leveraging sensor data and historical occupancy patterns, the algorithm intelligently adjusts heating systems (hot water production), resulting in substantial energy savings.

Clustering methods such as k-means clustering and hierarchical clustering are used to group similar patterns together. By identifying clusters or segments in data, these techniques facilitate the creation of energy profiles and enable the development of customized management strategies for different user groups or building types. Thereby, Ayenew et al. [41] use K-mean to understand and predict the aggregate load of substations in an urban area of Ethiopia, where frequent power outages are caused by overloaded transmission and distribution systems. With their proposed solution, they are able to identify intermediate and super-peak demand hours and potential customers for price-based demand load shifting management, showing that an increase in electricity prices at peak hours leads to a reduction in electricity demand, and therefore, the load can be reduced, improving the reliability and stability of the grid. In the same fashion, Ayub et al. [42] implement a load scheduling algorithm based on K-mean clustering and integer linear programming to schedule consumers' appliances for the following day. They evaluate the proposal using a publicly available real dataset, and the simulation results show that the approach works better, thus increasing the confidence and continuity of the system.

## 2.2. Optimisation-Based Algorithms

Optimisation techniques are widely used in many decision problems, including energy management, where the best possible result for a given defined problem. Research has focused on various optimisation approaches, such as linear programming, MILP, dynamic programming, and genetic algorithms.

Linear programming (LP) is a frequently used technique for solving optimisation problems. It involves formulating an objective function and a set of linear constraints to optimize resource allocation and power usage. Umetani et al. [43] develop an LP-based algorithm on a time-space network model for scheduling the charge and discharge of electric vehicles within a limited computation time. After comparing their proposal with the results obtained with an exact MILP solver, they show that it is possible to use heuristic methods based on LP to solve complex optimisation problems quickly and with results very

close to the optimal one. Similarly, Bio Gassi and Baysal [44] formulate a complex LP model combining electricity, heating, and cooling to control a microgrid energy management system with particular consideration of storage systems, such as batteries, as they are an important aspect [45] of this type of problem.

Mixed-Integer Linear Programming (MILP) extends linear programming by allowing decision variables to take discrete values. MILP is particularly useful for problems that involve binary decisions or discrete choices and is much more widely used in optimisation systems than LP. A notable work that makes use of MILP in power management is [46], which presents the modelling and design of a modular energy system and its integration into a grid-connected PV–wind–battery-based hybrid microgrid. The scheduling model is a power generation-side strategy, defined as an MILP by taking into account two stages for proper charging of the storage units based on 24-h ahead forecast data. Following the same approach, Javadi et al. [47] design a Home Energy Management System (HEMS) based on MILP for optimal self-scheduling in the presence of photovoltaic power generation and batteries.

Dynamic programming (DP) is a method for solving optimisation problems by breaking them down into smaller subproblems and recursively solving them. Tischer and Verbic [48] use dynamic programming for smart home control, which is equipped with a fuel cell used for heat and power cogeneration, a PV system, an electric car, a battery, and a storage unit for thermal energy. Through simulations they show that their algorithm obtains good results even though its computational speed is slow, they also discuss how to decrease the computation time by approximating the influence of random variables. In addition, DP is quite useful for solving battery charging problems in home scenarios [49] and electric vehicles [50], as it provides a way to choose the most appropriate way to charge batteries while minimising costs and maximising lifetime.

Genetic algorithms (GA) are nature-inspired optimisation techniques; these algorithms employ principles of natural selection, crossover, and mutation to search for optimal or near-optimal solutions in complex optimisation problems with low computational requirements. The use of GA is more common in the scheduling and shifting of consumption in houses or buildings, as can be seen in the works of [51,52]. Additionally, Arabali et al. [53] propose a GA-based strategy to meet the controllable heating, ventilation, and air conditioning (HVAC) load with a hybrid-renewable generation and electricity storage system.

### 2.3. Artificial Intelligence-Based Algorithms

Artificial intelligence (AI) techniques have revolutionized energy management techniques by leveraging large datasets and complex algorithms. These techniques enable intelligent decision-making, anomaly detection, and adaptive control in a direct and precise manner.

Decision trees are simple yet effective machine learning algorithms used in management for tasks such as consumption prediction, fault diagnosis, and load classification, they are based on the recursive partitioning of data according to different attributes, enabling the generation of rules that guide the decision management system. Ferrández-Pastor et al. [54] propose a web-based system for automated forecasting of electricity consumption for multiple instances (buildings) using a decision tree approach that includes conditions based on available variables (electricity consumption, building condition, outdoor temperature, etc.). They indicate a decrease in forecasting error (up to 1.75 times) due to the usage of a decision tree for forecasting model selection. Additionally, in [55], a similar work is presented but with a special focus on IoT systems, where the benefits of these proposals can be completely exploited. They include, besides the estimate of future consumption, the weather forecast, as the scheme includes generation by solar panels and wind turbines, and conclude that IoT infrastructure architecture should be a requirement for such systems in order to perform at their best.

Gradient boosting algorithms, such as XGBoost (extreme gradient boosting) and LightGBM (Light Gradient Boosting Machine), are algorithms that iteratively build an ensemble of weak predictive models, each focused on minimising the errors made by the



previous models. Gradient boosting algorithms excel at capturing complex patterns and achieving high prediction accuracy. As can be seen in [56], in the context of consumption forecasting, there are several different models (for example, LightGBM, CatBoost, and XGBoost) that can be used to develop solutions. In the aforementioned work, each of them is examined and tested with different hyperparameters to determine which one is the most suitable or at least which one works best for a particular type of data, concluding that XGBoost performs best when trained on the selected dataset. Moreover, Lu et al. [57] design a short-term prediction of consumption in an intake tower employing an improved extreme gradient boosting model and compare the performance of the method with five benchmark models, showing that the mean absolute percentage error of the proposed model is 4.85%, the lowest of all models. Also, in [58], the three common gradient boosting methods mentioned above are used in order to implement a control and forecasting solution in a solar power plant in a smart grid, and after proper training and comparison using various methodologies, they demonstrate how gradient boosting methods offer considerable advantages such as high accuracy and fast learning.

Among all the techniques already cited, artificial neural networks (ANNs) are probably one of the most important in current research, not only in energy management but also in many other areas [59]. In the context of electricity management, the ANNs are useful because they can be trained using historical data to predict future demand with high accuracy. There are a large number of architectures and classes of artificial neural networks, including fully connected networks, radial basis function networks, feedforward networks, convolutional networks, recurrent networks, and many more. An example of a fully connected network (three-layer backpropagation architecture) is shown in [60], where the authors develop a system to optimally regulate the equivalent factor, a critical scaling factor that determines the proportion of power consumed from fuel and the battery, of plug-in hybrid electric vehicles. On the other hand, a deep recurrent neural network (RNN) is presented in [61] for short-term building energy predictions, serving as a basis for any other control algorithms that use these predictions in making decisions to optimise electricity consumption or reduce the use of non-renewable sources. Another promising approach is the use of convolutional networks to extract features from data, rather than images, to implement management systems. For example, in [62], the authors propose to use a deep residual convolutional neural network (CNN) to extract spatial features from electricity data to predict future loads. They also implemented a Long Short-Term Memory (LSTM) network to learn the temporal information and a Gated Recurrent Unit (GRU), performing several experiments with each model and evaluating them with additional Coefficient of Variation of the RMSE (CV-RMSE) metrics.

Reinforcement learning (RL) has gained significant attention in energy management, especially those involving HVAC (Heating, Ventilating, and Air Conditioning) systems, due to its ability to easily learn optimal control policies through interaction with the environment [63–65]. In fact, RL is a very popular technique in current research, and there is a growing number of works that use it to dynamically solve all kinds of control problems [66,67]. It was generally used to solve problems in vehicle control and motion-based systems, but its potential was gradually realised in other areas where the actions of an intelligent system need to be optimised, standing out for being a very dynamic method that can adapt easily. In particular, the approaches used for electricity management take advantage of the adaptive capabilities of RL, based on the rewards received for each action, to iteratively learn the best decisions for any possible state. For example, in [68] an RL algorithm is used for real-time scheduling of a microgrid, considering the uncertainty of the load demand, renewable energy, and electricity price, with the aim of finding an optimal scheduling strategy to minimise the daily operating cost of the microgrid. Also, in [20], it is used in the same way to intelligently schedule the power usage of a house, but in this case, a feedforward neural network (FNN) is also added to increase the performance of the proposal by allowing the RL agent to use the FNN to predict upcoming energy consumption data. On the other hand, reinforcement learning can also be used

for the electric management of electric vehicles (EV/HEVs), as has been carried out in the other works mentioned above. Weihan Li et al. propose, in [69], an energy management strategy based on deep reinforcement learning for a hybrid battery system in electric vehicles, characterising the battery cells electrically and thermally; they aim to minimise the energy loss and increasing both the electrical and thermal safety level of the whole system. Furthermore, they compare their deep Q-learning-based solution (using neural networks) with a classical Q-learning algorithm, and they report that in the HEV scenario, the deep-based method obtains better results.

#### 2.4. Hybrid Algorithms

Integrated approaches combine multiple algorithmic techniques to harness their synergistic benefits. For example, Fan et al. [70] combine Bayesian networks and deep reinforcement learning to optimize the reliability of gas supply in natural gas pipeline networks, Bisen et al. [71] proposed a hybrid model based on convolutional neural networks and a modified k-mean clustering method for mobile edge computing, and Kuppusamy et al. [72] implement an energy management system solution based on a modified J48 decision tree algorithm with fuzzy logic principles.

#### 2.5. Our Contributions

In conclusion, energy management algorithms in buildings and smart houses have evolved significantly with advancements in IoT, machine learning, and data analytics. The discussed papers highlight various technologies, each with its own advantages and disadvantages, employed in energy management algorithms, including IoT, machine learning, data analytics, and hybrid approaches. These algorithms contribute to optimizing consumption, reducing costs, and improving sustainability in buildings and smart houses.

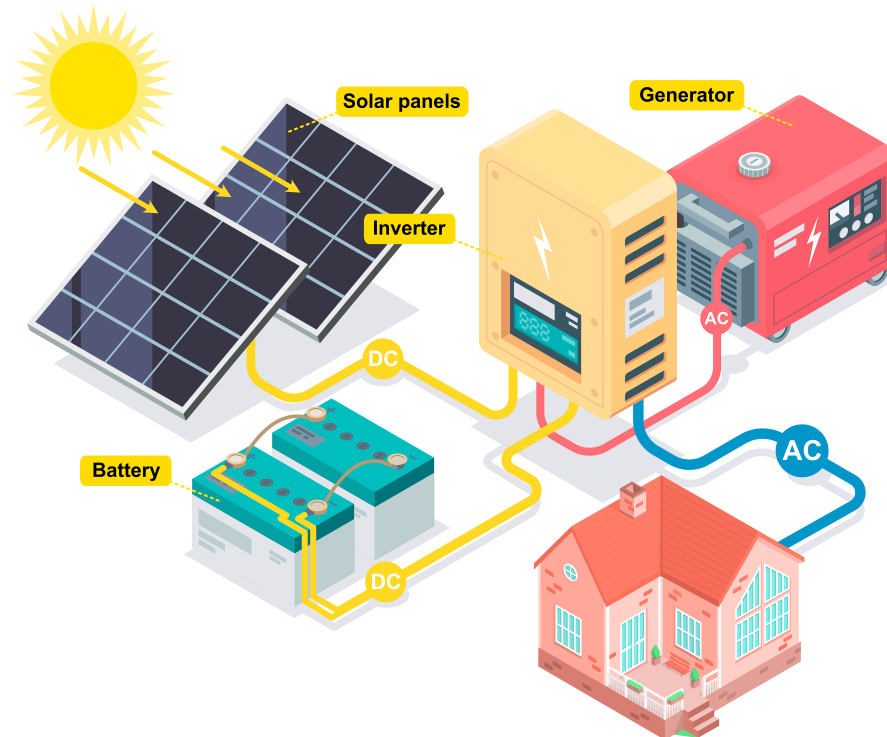
Among all the methods mentioned, those based on reinforcement learning are particularly promising and easy to implement in real-life settings. As presented above, there are a wide variety of techniques for addressing optimisation problems applied to decision systems, but most of them need to be designed with high precision and detailed knowledge of the scenario in which they are to be applied, which limits their flexibility and portability. Only those based on artificial intelligence or machine learning can be applied in dynamic scenarios where the possible states are not known in detail, and among them, reinforcement learning stands out for its flexibility and ability to learn on the fly without the need to define the agent's behaviour beforehand.

For that reason, the present work focuses on RL and applies it to a specific environment for managing devices, CPSs, and services in an isolated smart home with batteries, solar panels, a generator, and no connection to the grid. Thus, the algorithm will dynamically detect, using the rewards of each action taken, which are the best decisions for each moment taking into account the state of energy production and with the aim of optimising the use of electricity and avoiding the use of the gasoline power generator. One of the key aspects of our proposal is the integration of CPSs into the system so that both virtual services and physical devices are considered equally and managed by a multiagent reinforcement learning system. Furthermore, considering a scenario without a grid connection is also important, as none of the analysed proposals contemplated the possibility of a system being isolated from the energy grid. As a result, they did not address situations involving critical energy demands and the possibility of a power outage.

### 3. Energy Orchestrator

Our proposed system consists of an isolated house that has no external energy sources (grid) and therefore relies on solar panels, batteries, and a generator. The main source of energy of the system is the solar panel array which produces electricity when there is enough sunlight and stores the excess energy in a battery rack. When solar production decreases, the inverter supplies the energy needs of the system by discharging the batteries. Occasionally, this causes the batteries to discharge completely, reducing the battery lifespan

and forcing an abrupt shutdown of all system services. To prevent a no-power state, a generator is available which can exceptionally be used when the energy level of the batteries is very low to power the system until the batteries are sufficiently charged. Figure 1 illustrates the energy sources and the components of the energy supply system.



**Figure 1.** Energy supply system.

For this reason, energy management is crucial to avoid draining the batteries overnight and to take advantage of peak solar generation; therefore, we design a control plane for the services that will adjust the status of the services according to energy production and consumption. The control plane is deployed over the system as a service orchestrator focusing on energy management.

The services managed by the orchestrator are both virtual (*video streaming*) and physical (*fridge, pool pump*) services, and are therefore modelled as cyber-physical services with a defined power consumption per hour (*Wh*) and an execution rule (*on/off timetable and maximum run time*).

In addition, CPS services can be separated into two groups depending on whether the orchestrator manages them intelligently. The non-manageable services are those that the orchestrator cannot dynamically control their operation, so they are restricted to controlling their on/off switching according to their defined daily execution schedule (*execution rule*). On the contrary, manageable services are those that the orchestrator can manage intelligently, shutting them down when needed (*low energy or unexpected load*) and shifting their execution (*peak solar generation*) according to the current energy status, the service priority, the service execution schedule, and the maximum time allowed for the execution of the service.

Furthermore, the orchestrator also considers the existence of unexpected events that make the energy supply more complex and force it to take actions to reduce or shift energy consumption. For example, unexpected events can be a reduction in solar production on cloudy days, the punctual connection of a high power demand device to the system's power grid, or the complete draining of the batteries. Figure 2 summarises the proposed system components and some example services.



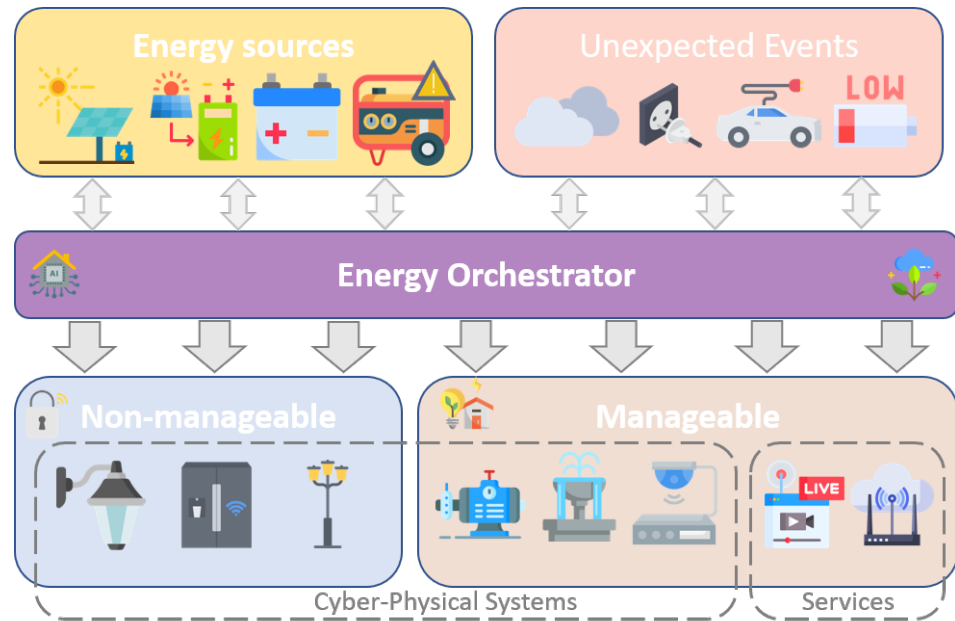


Figure 2. Proposed system.

#### 4. Proposed Approaches

In this section, we present our energy orchestrator approaches. Figure 3 provides an overview of the architecture of the proposed control algorithms. The energy orchestrator receives every minute information about the state of energy generation and consumption of the services provided by the simulator or real-world sensors. At each information update, the orchestrator generates a set of events that it sends to the control plane, where the main interface of the service controller receives it.

The service controller will collect event information and then make control decisions on the services in each timeslot according to the behaviour of the specific controller implementation. Timeslots are user-defined time intervals, e.g., 60 min, where the control algorithm will only collect information for decision-making at the beginning of the next timeslot. In the following subsections, we explain the proposed methods in detail.

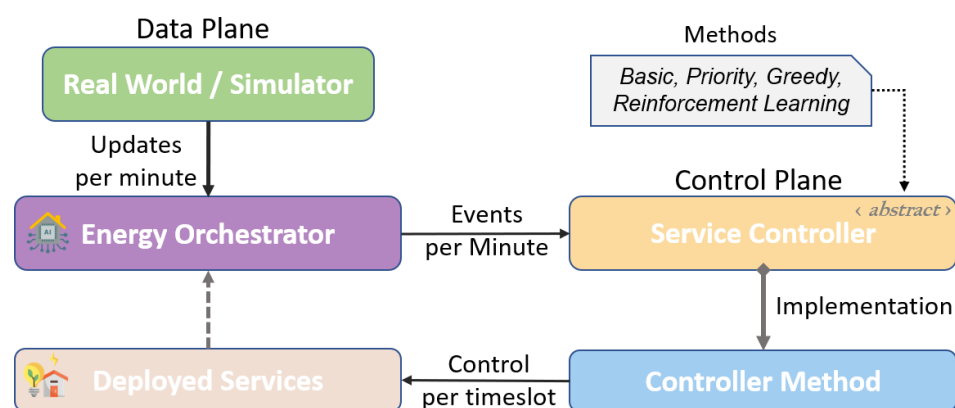


Figure 3. Orchestrator architecture.

##### 4.1. Basic Approach

The most basic service orchestration algorithm we propose is the Basic method, which is limited to keeping the services always on according to their execution rules. Therefore, in each timeslot, the controller will iterate through the list of deployed services and for each of them check its execution rule to determine if it can be switched on or not.

This method is neither practical nor efficient, but it provides an upper bound on the bad performance of the methods that we use for comparison with other solutions in the results section. Furthermore, this method can be seen as a comparison with an ordinary scenario where there is no service orchestrator and the services execution rules are controlled by standard plug-in time switches.

#### 4.2. Priority-Based Approach

As a first approach to the service orchestrator, we developed a simple algorithm that, when a critical energy level is identified in a timeslot, sorts the services by priority and turns off the lower priority services, and when the energy production increases, it turns them back on in order of priority.

Algorithm 1 shows the complete functionality of the proposed method. It starts initially by defining at 0, a variable (*position*) that will be used to determine the position of the last service with the lowest priority turned off (*starting from the end of the list of services sorted from highest to lowest priority*). Then, in each timeslot, a reduction factor (*bR*) of the current battery level will be calculated between 0 and *mBL*, depending on the current position with respect to the defined number of services.

After that, the method determines whether the energy state is critical or normal by checking if the average battery level is lower or higher than their respective thresholds minus the battery reduction factor. If the energy level is critical, the variable *position* is increased by one (*up to the number of services*) to turn off the next lower-priority service; otherwise, the variable is reduced by one (*minimum 0*) to turn on the last lower-priority service turned off. This algorithm has a linear computational cost of order  $O(n)$ .

---

#### Algorithm 1: Priority-based Algorithm

---

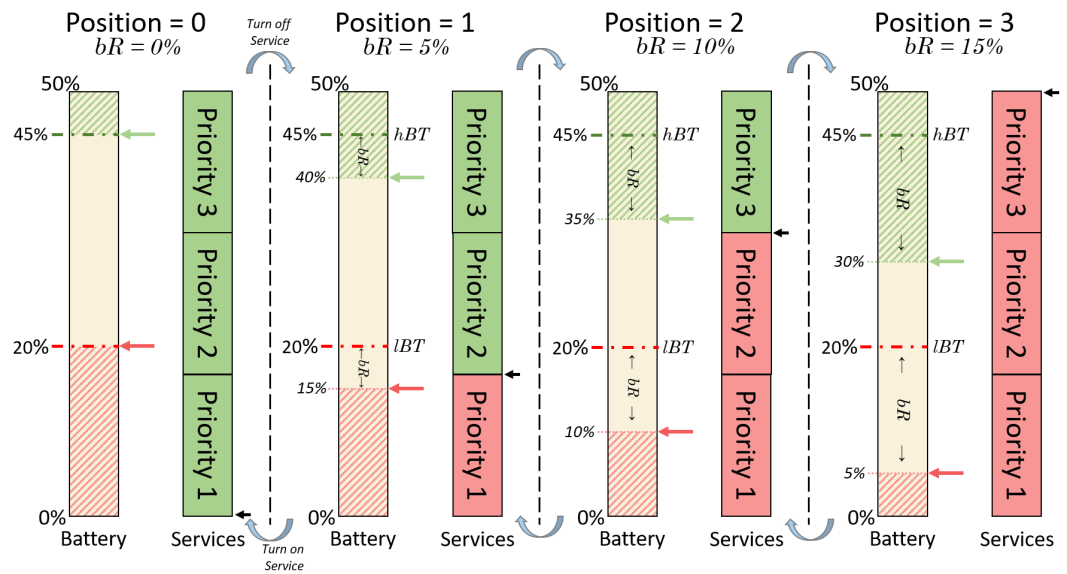
**Parameters:** list of services sorted by priority *svcs*, average battery level percentage *aBL*, low battery threshold *lBT*, high battery threshold *hBT*, battery level modifier *mBL*

```

1 begin
2   position ← 0
3   for each timeslot t do
4     // Battery reduce factor
5     bR ← mBL * (position / |svcs|)
6     // Critical level of energy
7     if aBL < lBT - bR then
8       | position ← max(position + 1, |svcs|)
9     // Normal level of energy
10    else if aBL > hBT - bR then
11      | position ← min(position - 1, 0)
12    end
13    // Services sorted from high to low priority
14    for i ← 1 to |svcs| do
15      | if i ≤ |svcs| - position then
16        | Turn on the service svcsi
17      else
18        | Turn off the service svcsi
19      end
20    end
21    Wait until the end of the time slot
22  end
23 end
```

---

Figure 4 shows an example of the possible states that the algorithm can have for certain parameters.

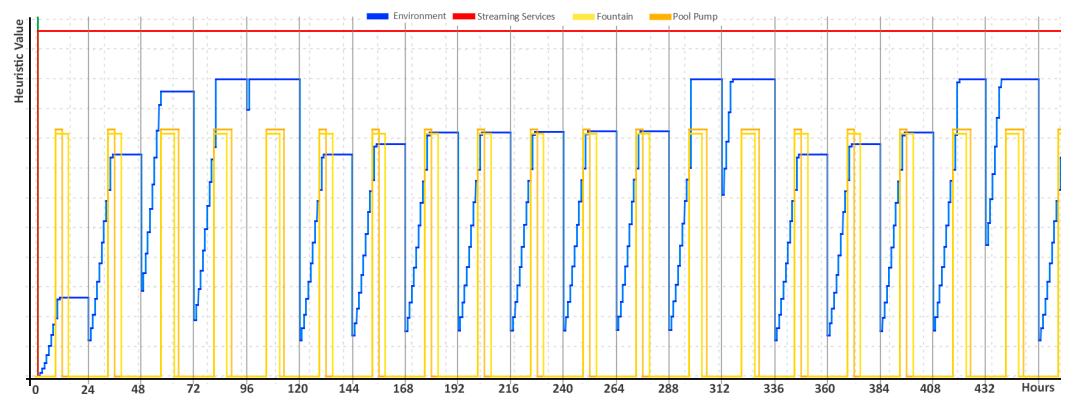


**Figure 4.** Example of priority-based algorithm states (parameters  $|svcs| = 3$ ,  $hBT = 45\%$ ,  $lBT = 20\%$  and  $mBL = 15\%$ ), when turning off or turning on lower priority services (upper and lower arrows).

### 4.3. Greedy Approach

A common technique for solving computational problems is Greedy algorithms, which are based on the use of heuristics to directly determine sub-optimal solutions. The major advantage of these methods is that they require little computational resources and execution time to achieve local solutions that are often very close to the optimal solution.

As an alternative and more dynamic approach, we also designed an algorithm that activates or deactivates services by comparing their heuristic value with the heuristic value of the environment. Consequently, in each timeslot, the algorithm will determine the heuristic value of the environment and the heuristic value of each service to compare them and turn off all services that have a value below the environment value. Figure 5 shows the value of the heuristics of some services together with the value of the environment from the result of an example simulation.



**Figure 5.** Example of heuristic values in the simulator.

The heuristic value of the environment is defined as a threshold given by the square of the inverse battery level. On the other hand, the value of each service is defined as the product of a constant, the square root of its priority, and the power of eight of its consumption relative to the battery. The complete control process for each service is detailed in Algorithm 2, and the computational complexity of the method is  $O(1)$  and is executed by each of the services individually.

**Algorithm 2:** Greedy Algorithm per Service

**Parameters:** daily minimum battery level percentage  $mB$ , trade-off constant  $C$ , priority of the service  $p$ , power consumption of the service  $pC$ , and the total battery storage capacity  $bC$

```

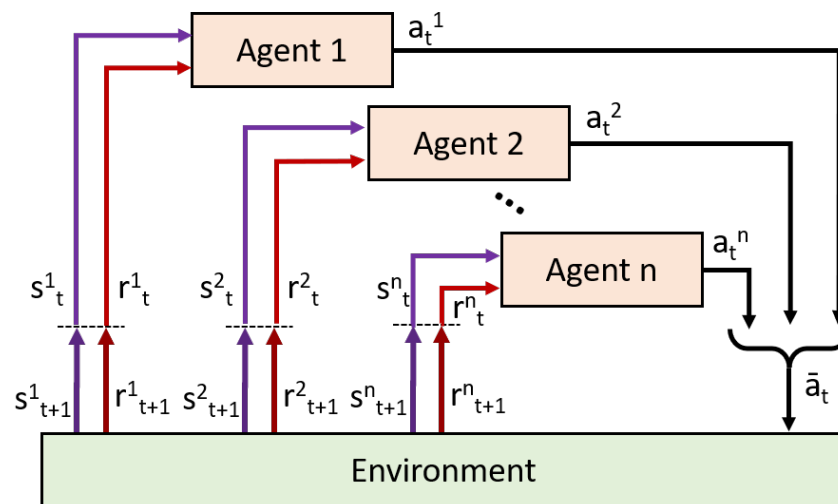
1 begin
2   for each timeslot  $t$  do
3      $threshold \leftarrow (100 - mB)^2$ 
4      $heuristic \leftarrow C * \sqrt{p} * (1 - (pC/bC))^8$ 
5     if  $heuristic < threshold$  then
6       Turn off the service
7     else
8       Turn on the service
9     end
10    Wait until the end of the time slot
11  end
12 end

```

**4.4. Reinforcement Learning Approach**

As seen in the literature [73,74], one of the most promising artificial intelligence methods for dynamic control of autonomous systems is reinforcement learning, which has been shown to be much more effective than other AI and heuristics approaches [75,76] in real-time control system.

Therefore, we propose using multiagent reinforcement learning to design the CPS service orchestrator. Thus, each service will run an RL agent that will make control decisions locally to optimise a global reward [77]. Figure 6 shows the behaviour of a multiagent RL system, in which each agent performs local actions according to their local states and rewards and is then jointly applied to modify the global environment.



**Figure 6.** Multiagent reinforcement learning.

RL agents are implemented as a Q-Learning model-free algorithm, so each of them will store a Q-table that will be updated based on local rewards, status, and actions. At each control time step of an episode, the agent will decide a control action ( $a_t \in A = \{0, 1\}$ ), whether to turn off (*action 0*) or turn on (*action 1*) the service for that time step. The decision will depend on the environment, which is based on the battery percentage, the minimum daily battery percentage, and the real-time energy available, calculated as the difference between energy production and energy consumption in Wh. Figure 7 depicts an overview of the proposed Q-Learning agent.

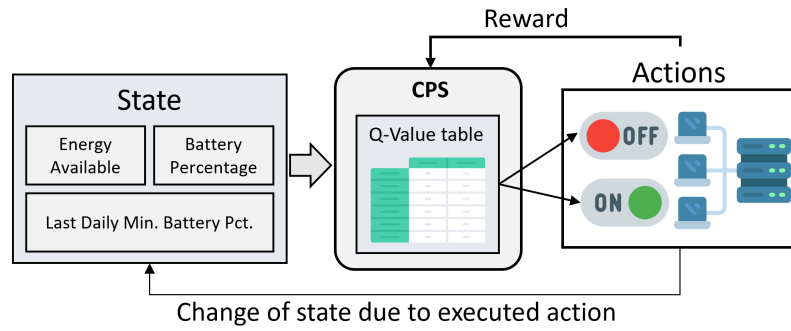


Figure 7. CPS Q-Learning agent.

Classical Q-Learning algorithms require a finite state space, so it is necessary to discretize continuous values of the environment before using them. The discretisation process of the state parameters is summarised in Figure 8; the scheme followed is a direct discretisation that splits the range of values into sections of equal width [78].

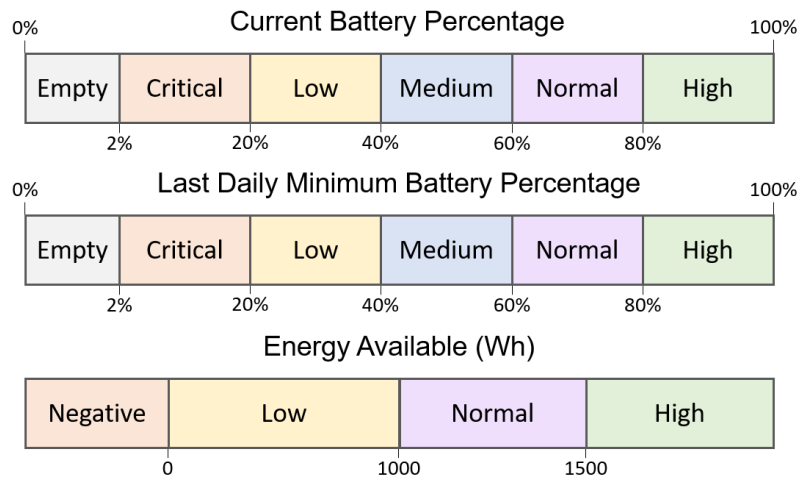


Figure 8. Discretisation of state parameters.

The learning process uses a Q-Value table to store and query the value of the Q-function for each state and action. When an action is performed, the new Q-Value in the table is updated according to the following one-step Q update formula:

$$Q(s_t, a_t) = (1 - \alpha) Q(s_t, a_t) + \alpha (C_t(S) + \gamma \min_a Q(s_{t+1}, a)) \tag{1}$$

The reward obtained after the execution of an action is a piecewise function of two elements that depends on the action taken, the parameters of the service, and the new state of the environment. Therefore, the reward of service “S” at time “t” is formulated as the following weighted sum between the service priority ( $S_p$ ), the current operating time ( $S_{rt}$ ), the waiting time ( $S_{wt}$ ), the inverse battery level ( $B_t$ ), the minimum daily inverse battery level ( $B_t^m$ ), and the generator usage ( $G_t$ ):

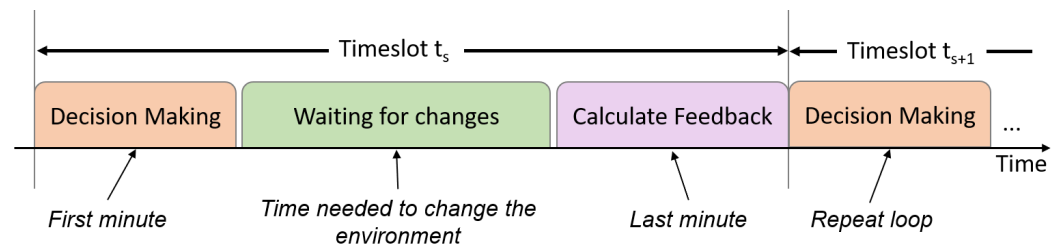
$$C_t(S) = \begin{cases} \beta_2 S_{wt} - \beta_4 B_t^m & a = 0 \\ \beta_0 S_p + \beta_1 S_{rt} + \beta_2 S_{wt} + \beta_3 B_t + \beta_4 B_t^m + \beta_5 G_t & a = 1 \end{cases} \tag{2}$$

where each  $\beta_x$  is the trade-off constant for each parameter. In order to change the order of magnitude of some of the parameters, several modifications have been made to the raw parameters. The  $B_t$ ,  $B_t^m$ , and  $G_t$  parameters are reduced by the exponential consumption over battery factor, defined as the power of two of the division of the service consumption by the total battery. Also, the raw value of the inverse battery level (100 minus the battery level) is squared to exponentially increase the penalties when the battery level is low.



In addition, the control policy follows a decay  $\epsilon$ -greedy approach; thus, the best action, the one with the higher Q-Value, will only be selected if a random number  $e$  is greater than or equal to  $\epsilon$ . Otherwise, the action is chosen randomly from the set of feasible actions. Since the policy is *decay*, the value of  $\epsilon$  will be reduced by the simulation time (*to one-tenth per week*) and the priority of the service (*higher priority, higher reduction*).

Regarding the workflow of the method, the control process divides the operating time into slots of fixed duration. In the first minute of each timeslot, the decision-making algorithm is executed for the current state of the environment; after that, the algorithm waits until the last minute of the timeslot to determine the reward for the action taken based on the average state of the waiting time. This control loop is repeated by each agent continuously until the end of the simulation. Figure 9 illustrates the described behaviour.



**Figure 9.** Timeslot structure of the RL control process.

Therefore, each agent will run the RL algorithm independently and synchronised with the global timeslots to update at the end of each one the local reward of its actions. The proposed multiagent solution based on decay  $\epsilon$ -greedy Q-Learning is shown in Algorithm 3, which has computational complexity  $O(|S||A|)$ .

---

### Algorithm 3: Decay $\epsilon$ -greedy Q-Learning Algorithm

---

**Parameters:** discount factor  $\gamma$ , learning rate  $\alpha$ , exploration rate  $\epsilon$ , and weighting parameters  $\beta$

```

1 begin
2   for each timeslot  $t$  do
3     Observe actual state  $s_t$ 
4     Determine feasible action set  $A'$  from  $A$ 
5      $e \leftarrow$  random number from  $[0,1]$ 
6      $\epsilon' \leftarrow \epsilon \div S_p(1 + \frac{\text{simulationMinute}}{1120})$ 
7     if  $e < \epsilon'$  then
8        $a_t \leftarrow$  randomly select an action from  $A'$ 
9     else
10       $a_t \leftarrow \arg \min_{a \in A'} Q(s_t, a)$ 
11    end
12    Execute energy management action  $a_t$ 
13    Wait for next feedback step
14    Observe new state  $s_{t+1}$ 
15    Calculate reward  $C_t$  by (2)
16    Update  $Q(s_t, a_t)$  according to (1)
17  end
18 end

```

---

## 5. Simulations Results and Discussion

In this section, the performance of the proposed solution is analysed using the results of a smart house simulator [79]. The detailed simulation results are available in our GitHub repository [80] and the source code of the Java simulator implemented with the proposed methods is also available in our GitHub repository [81] so that anyone can easily test the methods and verify the results. We used the simulator to simplify testing and ensure repeatability, but the original experiments to verify if the simulator was realistic were

performed in a real home with a Node-Red control system; therefore, our proposal is easily applicable to real environments directly by integrating the algorithms into the control plane of the smart home system.

### 5.1. Simulation Setup

To evaluate our proposal, we configured the aforementioned simulator to emulate a house with different energy sources and a set of CPS services. The characteristics of the energy system of the house are summarised in Table 1, and the parameters of the reinforcement learning algorithm are shown in Table 2.

**Table 1.** Characteristics of the simulated house.

Parameter	Value
Simulated Time	25 days
Procedural data generation	Disabled
Battery capacity	7000 Wh
Battery discharge rate	4000 W
Solar peak production	2200 W
Cloudy days	First 4 days
Cloudy solar production	80% of a sunny day
Unexpected load days	4th, 13th, 17th, and 22nd
Unexpected load (kW)	1.2, 1.2, 1.8, and 1.8
Unexpected load time	Between 10:30 a.m. and 12:00 p.m.

**Table 2.** Reinforcement learning algorithm parameters.

Parameter/RL Algorithm	Single
Timeslot	60 m
Learning rate $\alpha$	0.2
Discount factor $\gamma$	0.8
Initial $\epsilon$	0.2
Initial Q-Value (Off, On)	0 and 50
Value of $\beta_x$	50, 0.02, $-0.01$ , $-40$ , $-200$ , $-2000$

To ensure a fair comparison between methods, the procedural option to randomly modify the solar generation in the simulator was deactivated.

In addition, two unexpected events were implemented to verify the performance of the methods in critical situations. The first one consists of simulating cloudy weather (as it typically has a significant impact on solar production [82] and is a relatively common phenomenon) for the first four days so the solar output drops to 80%. The second one simulates the occasional switch-on from 10:30 to 12:00 of a high-consumption device (1.2 kW and 1.8 kW).

Finally, to demonstrate the behaviour of the energy orchestrator, a set of CPS services, both manageable and unmanageable, are deployed. Table 3 lists the deployed services showing the consumption and run rules for each of them.

**Table 3.** CPS services deployed (configuration 1).

Service	Smart	Priority	Load	Rule
Fence lights	No	-	15 W	8 p.m.–8 a.m.
Facade lights	No	-	10 W	8 p.m.–8 a.m.
Fridge	No	-	120 W	All time
CCTV DVR	Yes	10	20 W	All time
Internet	Yes	8	40 W	All time
Pool Pump	Yes	4	600 W	9 a.m.–5 p.m. (max 3 h)
Streaming Svcs.	Yes	2	30 W	All time
Fountain	Yes	1	35 W	9 a.m.–3 p.m.

### 5.2. Tested Methods

Using the simulator, we obtained a complete set of data on the performance of the orchestrator and the energy consumption and production for 25 days. We tested four control algorithms for the orchestrator. The first one, called Basic (*BSC*), does not perform any intelligent control of the services; it only turns on and off the services according to its running rules. The second method, Priority-based (*PB*), sorts the services according to their priority, and if it detects that there is a critical energy level, it turns off the lower-priority services until the energy production increases.

The third one is a Greedy (*GDY*) method that is based on comparing the value of a dynamic heuristic that depends on the state of the environment with the heuristic value of each service. In case the service has a lower value, it is turned off.

Lastly, the fourth one is the multiagent reinforcement learning (*RL*) algorithm explained before, which dynamically decides whether to turn services off or on at each timeslot (subject to execution rules). Due to the nature of *RL* algorithms, the agent needs to gradually learn the best actions to take based on the rewards. For this reason, the algorithm requires repeating the simulation several times (saving the *Q*-tables) to progressively converge to the best policy. In our scenario, the simulation had to be repeated five times in order to obtain the best solution (*RL5*).

### 5.3. Tested Configurations

To verify the correct performance of the methods, we designed three different configurations of the simulation scenario. The first one (*configuration 1*) is the main scenario of the discussion and the focus of our analysis, and its deployed services and simulation parameters are mentioned above.

The second one (*configuration 2*) is the same scenario but the fountain and pool pump services are changed to high priority to check if it still optimises energy usage properly. The Table 4 lists the service parameters.

Finally, the last configuration (*configuration 3*) is totally different with respect to the services that are deployed; in this case, they are all services with similar consumptions (lists in Table 5) in order to test a scenario with other kinds of services.

**Table 4.** CPS services deployed (configuration 2).

Service	Smart	Priority	Load	Rule
Fence lights	No	-	15 W	8 p.m.–8 a.m.
Facade lights	No	-	10 W	8 p.m.–8 a.m.
Fridge	No	-	120 W	All time
Pool Pump	Yes	8	400 W	9 a.m.–5 p.m. (max 3h)
Fountain	Yes	6	35 W	9 a.m.–3 p.m.
Internet	Yes	3	50 W	All time
CCTV DVR	Yes	2	40 W	All time
Streaming Svcs.	Yes	1	30 W	All time

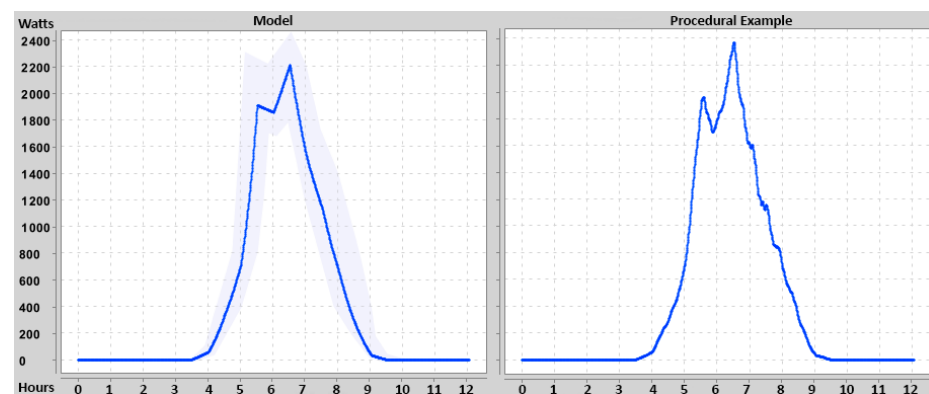
**Table 5.** Smart services deployed (configuration 3).

Smart Service	Priority	Load	Rule
Kubernetes Cluster	10	150 W	All time
Internet	9	25 W	All time
VPN Router	8	15 W	All time
Dashboard Services	4	50 W	All time
Game Servers	2	140 W	All time
Dashboard Displays	1	180 W	10 a.m.–6 p.m.

#### 5.4. Data Models

All the information used by the simulator to determine the solar production and passive consumption of the system is based on statistical models modelled using data from a real solar infrastructure in Murcia (Spain) for two months, identical to the one described in Section 2. There are two models for solar generation, one based on actual measurements of solar output and the other based on solar irradiance measured by a meteorological station [83].

Figure 10 shows graphically the solar generation model used for the simulations, where, for each hour, the mean value of solar radiation and the upper and lower bound of the 95% confidence interval are indicated using a blue shadow (left graph). The confidence interval is used to procedurally generate each day differently but with correlated values by using Perlin noise [84], as can be seen in the right graph of the aforementioned figure, thus enabling the dynamic and realistic behaviour of the simulations.



**Figure 10.** Solar energy production model with confidence interval (left) and procedural energy generation example (right).

As previously mentioned, the procedural generation option is disabled in all our simulations to ensure a fair comparison by providing the same input data for all simulations; thus, the simulated solar production in our tests is the same as shown in the second graph in the figure above for all days in all simulations.

### 5.5. Experimental Results and Analysis

The most important metrics to consider when evaluating each method are the average daily percentage of execution of each service (relative to its maximum time) and the final energy consumption of the generator, since using the generator is considered exceptional and should be avoided as much as possible. Therefore, the goal for the comparison of each method is that the generator usage is as low as possible, while the average run time of each service is as close to 100% as possible. It is important to note that the primary goal is to keep services running as much as possible, so in some scenarios, very low generator usage is not ideal, as it can significantly reduce the average execution time of each service. Table 6 shows the performance of each method in the simulator according to configuration 1 and the simulator parameters mentioned above.

**Table 6.** Performance of methods (configuration 1).

	BSC	PB	GDY	RL	RL5
CCTV DVR	100%	93.7%	100%	99.8%	100%
Internet	100%	89.3%	100%	100%	100%
Pool Pump	100%	90.7%	72.0%	88%	70.7%
Streaming Services	100%	74%	100%	100%	99.2%
Fountain	100%	11.3%	49.3%	98.7%	99.3%
Generator (Watts)	20,180	5934	7502	14,680	7335

As shown in the table, the Basic method represents the worst possible performance as it keeps all services on; therefore, the generator consumption result of this method serves as the upper bound of the methods to be compared.

The second proposal, the Priority method, provides the best result with respect to generator usage at the cost of significantly reducing the running time of lower-priority services. This method could be considered useful in simple scenarios where no dynamism is required and only priority-based management is relevant; however, in scenarios such as the one we propose, more flexibility and real-time adaptability of the algorithms are needed to maximise the uptime of all services while reducing the generator usage.

On the other hand, the Greedy method offers superior final performance concerning the consumption of the generator and by keeping the services on as much as possible. This is due to the use of dynamic heuristics that change according to the energy state, shutting down all services that have a lower value. Thus, when solar production is low or the battery reaches critical levels, the lowest priority services with the highest consumption are preferentially switched off (e.g., pool pump). The disadvantage of this method is the low dynamism, as the trade-off between priority, consumption, execution time, and energy status is manually defined and does not easily adapt to changes in the system.

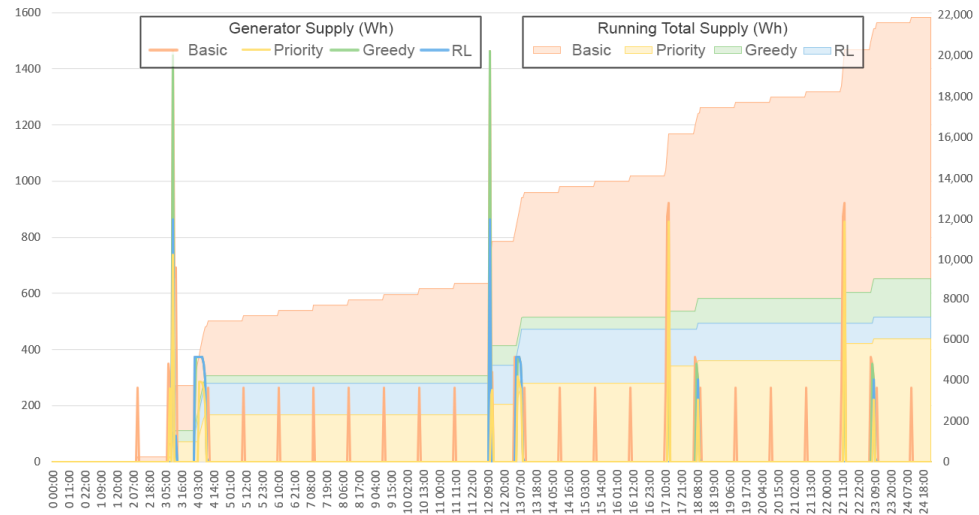
In contrast, the method based on reinforcement learning is completely flexible, as it continuously adapts to changes and does not need to define any heuristics beforehand. The drawback is the need to repeat its execution several times (episodes) until the algorithm converges (learns) to the optimal solution. For that reason, the table compares the result of the simulation with the initial RL method (without knowledge) and the result after running the simulation five times, keeping the Q-tables of each agent.

As can be seen, the method without initial knowledge hardly turns off the services, as it does not yet know the most appropriate actions to take, which leads to a high generator consumption. On the contrary, the trained method achieves the best solution as its generator



usage is the lowest while keeping the services on as much as possible. The proposed method not only adapts in real time to unexpected events but is also able to shift the execution of scheduled services to the hours of highest solar production or to the hours it estimates to be the most suitable.

To graphically summarise the comparison between the methods according to their generator usage, Figure 11 shows the punctual and the aggregate usage of the generator in watts during the simulation execution.



**Figure 11.** Comparison of generator usage of each method.

The Basic method uses the generator continuously every night as the batteries run out of energy. However, Priority and Greedy methods reduce the use of the generator only to occasions where unexpected consumption occurs. Similarly, the RL method only uses the generator when the energy production is not sufficient due to unexpected consumption, but it also manages to minimise the use of the generator by shifting or limiting the execution time of the energy-intensive services.

Figure 12 shows a detailed comparison of each method in a three-day segment of the simulation, where the behaviour of each algorithm in different situations can be more precisely appreciated. Each graph in the figure shows the consumption of the active services in each hour (timeslot) combined with the battery level (per mille) and the energy consumption (W).

The first graph shows the default behaviour of the system, in which the services are turned on according to their execution rules, causing the system to run out of energy during peak hours on both the second and third days. Also, on the third day, an unexpected consumption of 1.2 kW occurs.

The second graph illustrates the previously mentioned behaviour of the priority-based method; when critical energy situations occur, the lowest priority services in each timeslot are progressively shut down in order until energy production improves and then gradually switched back on.

Likewise, the third graph shows how the Greedy method detects the low battery status of the second day in order not to turn on any services with low priority. In addition, during the unexpected consumption on the third day, it also turns off the low-priority services to save energy.

In contrast, the last graph shows how the RL algorithm is able to dynamically adapt the execution of the services to reduce the use of the generator while maximising the running time. One of the most important results of the proposed method is the shift of the pool pump service to the period of the highest battery level and solar production. Also, the partial turn-on of the pump service in case of detecting a low energy production level or total turn-off in case of unexpected high consumption proves the proper adaptive behaviour of the algorithm.

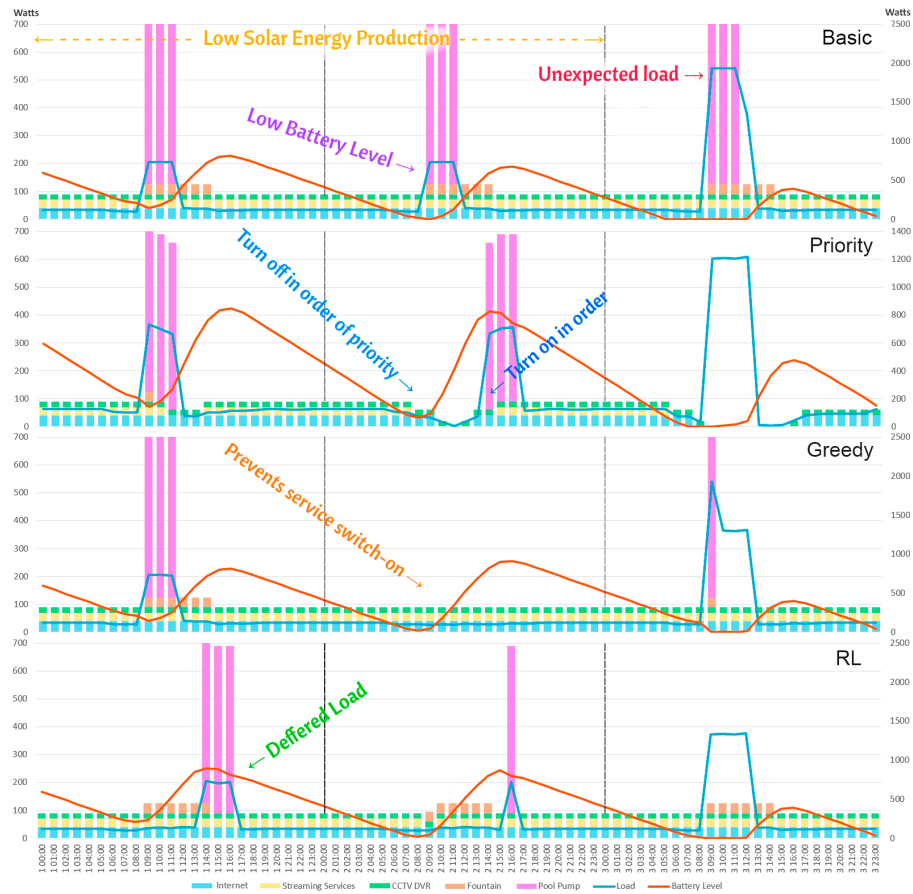


Figure 12. Three days simulation of each method.

Finally, the complete simulation result of 25 days is shown in Figure 13 in the same format as the previous figure. Both the Greedy and RL methods avoid the continuous discharge of the batteries due to the unexpected load, but the RL approach constantly demonstrates its adaptive behaviour by shifting the energy consumption of some services or performing partial or intermittent power cycles when the energy conditions are critical.

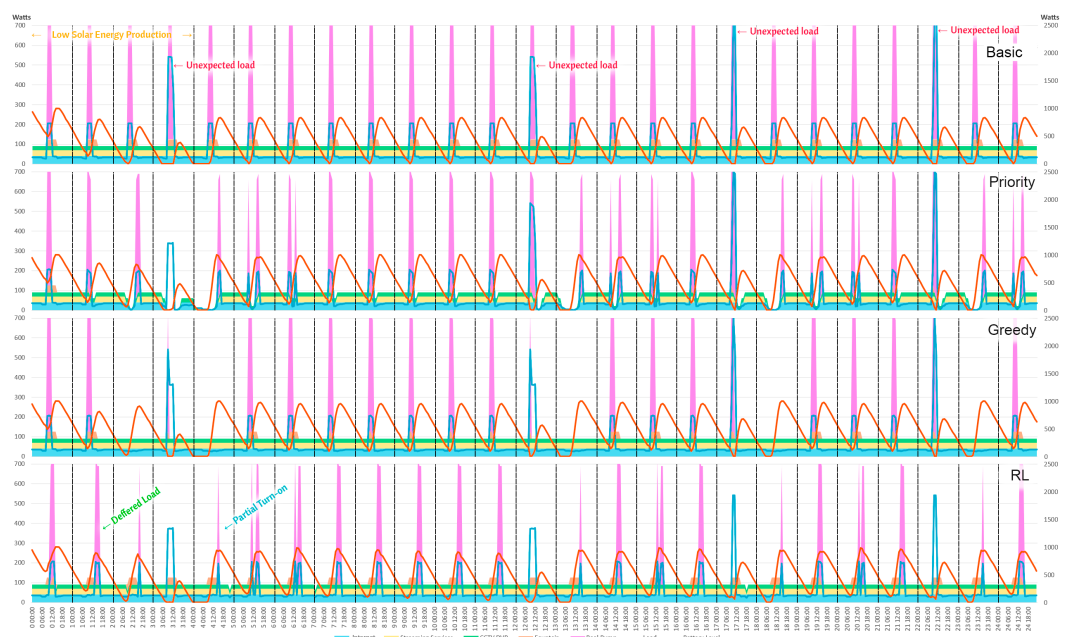
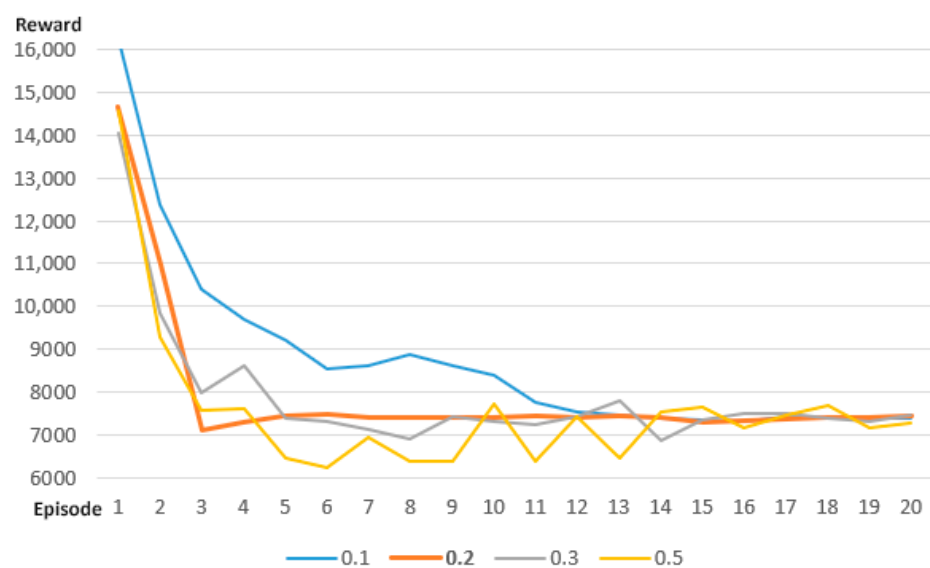


Figure 13. Twenty-five-day simulation of each method.

### 5.6. Parameters Tuning

In the previous section, we presented the results of our proposal taking into account the parameters of the RL agent in Table 2. The value of these parameters was established by testing each of them in different simulations to determine which was the most appropriate value for each parameter. In this sub-section, a brief overview of the results of each variable of the simulator is presented in order to justify those that have been finally chosen.

One of the most important parameters of reinforcement learning is the learning rate ( $\alpha$ ): the lower the rate, the lower the value of new rewards. Figure 14 shows how the generator usage evolves after each simulation episode for different values of alpha. When the value is 0.1, it can be observed that the system takes a long time to stabilise the generator usage at the best value (around 7400), as its learning rate is too slow. On the other hand, when alpha is 0.5, the best use of the generator is quickly achieved due to the high impact of rewards on new Q-Values; however, the system does not stabilise easily over several simulations.



**Figure 14.** Evolution of reward per episode for different values of alpha.

As a result, intermediate values, such as a learning ratio of 0.2, are considered that keep the best of both, converge quickly, and are reasonably stable in the following episodes.

Furthermore, the value of the reward is defined by several parameters as a weighted sum based on each  $\beta_x$ , which has an impact on the final performance of the system. In the following tables, we summarize the results of the simulation (running time of services and use of generator) in the fifth episode for different values of the three most relevant reward components (service priority, battery level, min battery level) so as to determine the behaviour of each of them.

The first table (Table 7) depicts the impact on the simulation results of different weighting values of the service priority. The lower the value, the less important the service priority is for the RL agent's decisions; on the contrary, high values denote the high importance of the priority, so the agent will try not to shut down those services. For this reason, the table shows that increasing the importance in the final reward of the priority of the services increases the use of the generator, as the agent tries to keep the pool pump (medium priority and high consumption) turned on for a longer time. For our simulations, we use 50, as we consider it to be the best fit in terms of giving importance to the priority of service but not so much as to drastically increase the generator usage.

**Table 7.** Performance in the 5th episode for different service priority ( $S_p$ ) weights ( $\beta_0$ ).

	10	50	100	200
CCTV DVR	99.8%	100%	99.8%	100%
Internet	99.7%	99.8%	99.5%	99.8%
Pool Pump	2.71%	70.7%	88.3%	90.7%
Streaming Services	99.5%	98.5%	99.8%	99.8%
Fountain	98.7%	99.3%	98.0%	98.7%
Generator (Watts)	3784	7320	12,990	15,410

In addition, another important parameter of the reward is the current battery level, as it balances the importance that the agent gives to the remaining stored energy. We compare different values of  $\beta_3$  in Table 8 with the value used in the simulations section (40). As opposed to the previous one, more importance to the battery level means less use of the generator as the running time of the service decreases. Having a low value would not be particularly useful since the agent would not take into account the battery energy level; on the other hand, a very high value would make the agent not give enough attention to the other parameters and just optimise the battery usage. For this reason, we use an average value that provides a reasonable trade-off between the battery level and the other parameters.

**Table 8.** Performance in the 5th episode for different battery level ( $B_t$ ) weights ( $\beta_3$ ).

	10	20	40	80	160
CCTV DVR	100%	99.8%	100%	100%	99.7%
Internet	100%	99.7%	99.8%	99.5%	100%
Pool Pump	85.3%	76.0%	70.7%	68.4%	56.2%
Streaming Svcs	99.8%	99.5%	98.5%	99.3%	99.5%
Fountain	97.3%	97.3%	99.3%	99.3%	100%
Generator (W)	12,910	9035	7320	6632	4521

Likewise, the impact on the simulation results of parameter  $\beta_4$  was tested with various values to determine the most suitable. In this case, the parameter controls the weight that has the lowest battery level of the last day ( $B_t^m$ ) so that the worst stored energy state of the system on the previous day is modelled as a negative reward to avoid it on the current day; in other words, it is used to penalise past actions that produced a low level of stored energy.

Table 9 presents the results of the simulations for five different values, and as can be seen, the central value is chosen for the same reason as the previous parameter.

**Table 9.** Performance in the 5th episode for different min-battery ( $B_t^m$ ) weights ( $\beta_4$ ).

	50	100	200	300	400
CCTV DVR	99.7%	100%	100%	100%	100%
Internet	99.7%	99.8%	99.8%	99.8%	99.8%
Pool Pump	85.3%	82.7%	70.7%	50.7%	22.7%
Streaming Svcs	99.7%	98.8%	98.5%	99.2%	99.2%
Fountain	100%	99.3%	99.3%	98.7%	99.3%
Generator (W)	12,970	11,655	7320	5261	3908

### 5.7. Performance of Other Configurations

After analysing the results of the first configuration in the previous section, it can be concluded that the RL algorithm works properly. However, it is relevant to determine whether the algorithm is able to perform similarly in scenarios with other characteristics.

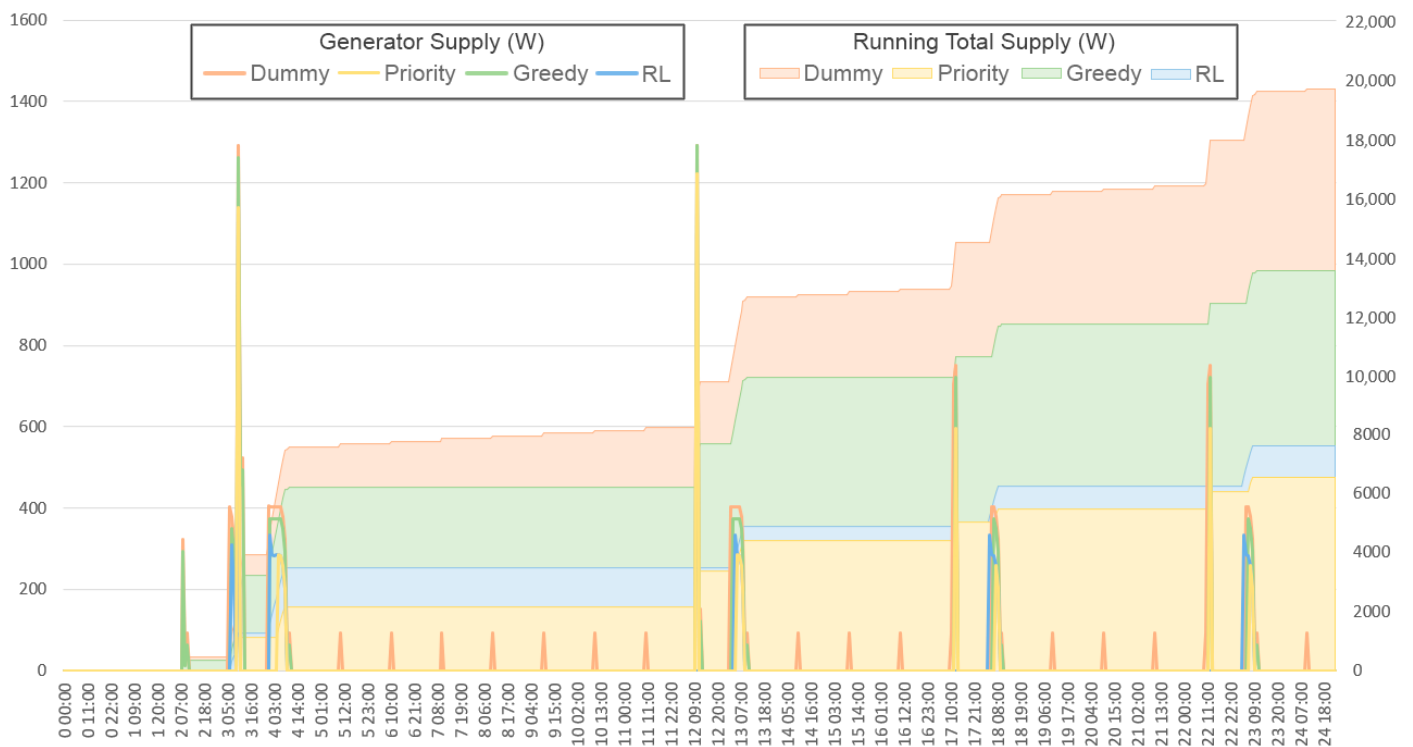
Configuration 2 is designed to be similar to configuration 1 but changes the priority of the high-demand devices; in particular, the pool pump and fountain are given maximum priority to force the algorithm to optimise power consumption considering the low-demand services much more. In the same way as the first configuration, the performance of each method for this configuration is summarised in Table 10.

**Table 10.** Performance of methods (configuration 2).

	BSC	PB	GDY	RL	RL5
Pool Pump	100%	98.7%	100%	98.7%	92.1%
Fountain	100%	51.3%	100%	100%	99.3%
Internet	100%	75.5%	100%	93.7%	93.7%
CCTV DVR	100%	67.8%	100%	81.2%	80.2%
Streaming Services	100%	60.2%	43.3%	61.3%	99.2%
Generator (Watts)	23,227	4828	13,659	10,036	7325

As shown in the table, the Basic method is still an upper limit of bad performance, which, in this case, is easily improved by the Greedy method just by reducing the Streaming Services runtime. Also, the Priority solution achieves a low generator usage but, at the same time, a low execution time of the services, especially those with low priority.

On the other hand, the method based on reinforcement learning provides the best performance in both the initial and the trained versions by balancing the execution time of all services according to their rewards. Figure 15 details the single and aggregate generator usage of each method during the simulation.



**Figure 15.** Comparison of generator usage of each method (configuration 2).



For a detailed view of the behaviour in this scenario, a three-day overview is provided in Figure 16. Similar to the previous scenario, the RL algorithm tends to defer the execution of the high-consumption services, but it does not turn them off, as they now have maximum priority. On the contrary, the other services (lower priority) have to reduce their execution time in order to limit energy consumption.

Lastly, Figure 17 summarises the full 25-day simulation run for this configuration, detailing for each day and hour the behaviour of the three proposed methods.

Likewise, we also propose configuration 3, which follows a different approach to verify the performance of the RL algorithm when managing services of similar consumption, instead of easily detecting critical services due to their high consumption. In this scenario, the trade-off between services runtime is crucial to optimise energy consumption, in particular the final generator usage. The results of the 25-day simulations are summarised in Table 11, comparing the percentage of execution of the services in each method and showing the final usage of the generator.

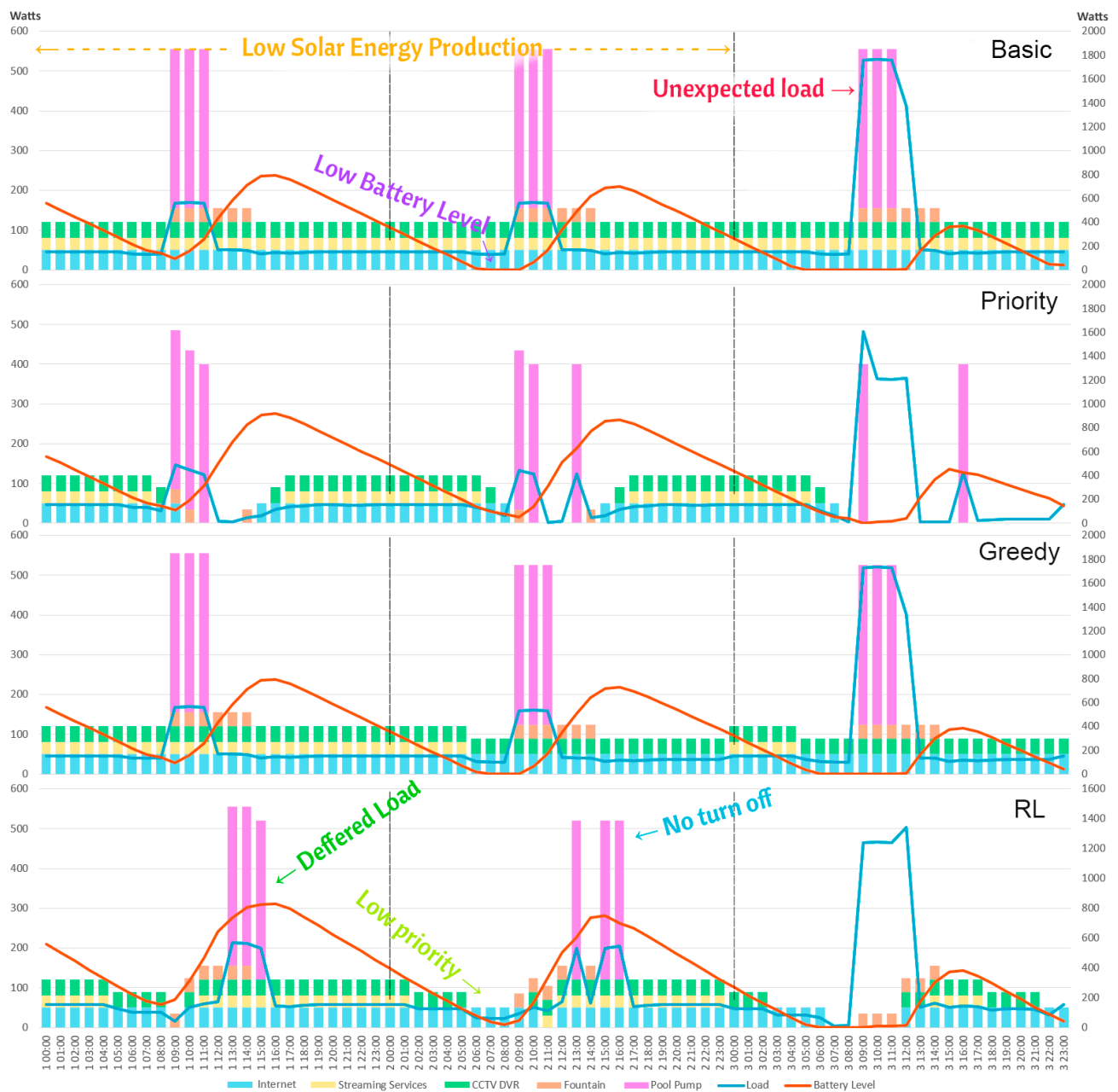


Figure 16. Three-day simulation of each method (configuration 2).

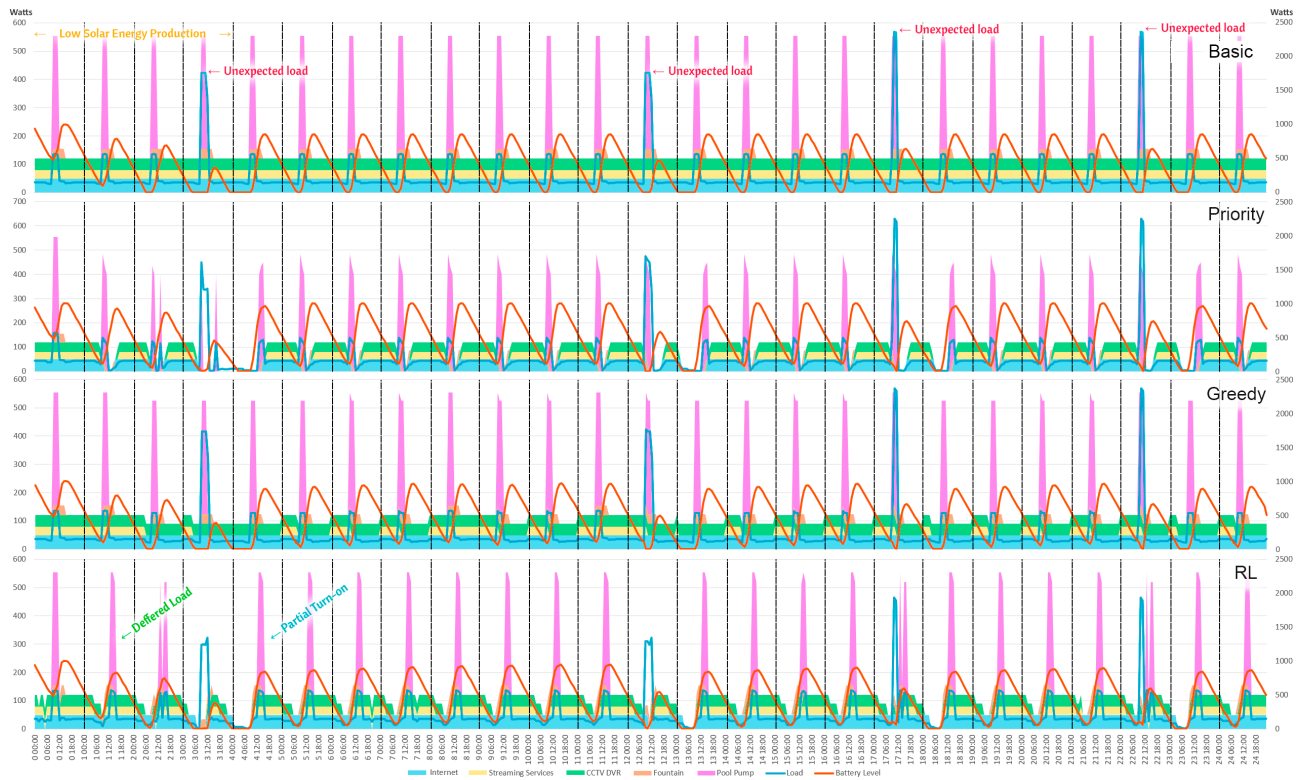


Figure 17. Twenty-five-day simulation of each method (configuration 2).

Table 11. Performance of methods (configuration 3).

	BSC	PB	GDY	RL	RL5
Kubernetes-cluster	100%	98.8%	100%	98.8%	99.8%
Internet	100%	97.5%	100%	100%	99.7%
VPN-router	100%	95.5%	100%	100%	100%
Dashboard-services	100%	87.2%	100%	99.8%	99.8%
Game-servers	100%	78.8%	100%	86.7%	85.7%
Dashboard-displays	100%	40.0%	44%	81.5%	71.0%
Generator (Watts)	16,204	115.2	6455	5932	2532

As can be seen in the table, the Basic method defines the worst possible performance if no dynamic control algorithm is used. The greedy method significantly improves performance but only by turning off the lowest priority service for most of the days, and the priority method shows the same behaviour as the previous configuration, with very low generator usage due to the reduced execution time of the services in order of priority. In fact, in this case, the reduction is so drastic that the generator is only used at a specific moment (approx. 100 W) in the middle of the fourth day when unexpected consumption occurs. However, this low consumption is reached by significantly reducing the uptime of the services, which suggests that as a basic strategy for non-critical services, it is very useful, but in more complex scenarios where we want to optimise both uptime and consumption, it is not the most appropriate method.

Besides, RL methods achieve the best performance as we expect, the non-knowledge version narrowly outperforms the Greedy method but manages to better balance the execution time of each service, improving the overall quality of service. The trained version of the RL method stands out from the other methods as it achieves the lowest generator usage and an acceptable service execution ratio, even while shifting the execution of services by being able to shift the execution of services to periods with better energy production or in case it is needed to make partial executions of services in order to save energy.

The evolution of the generator usage (with cumulative) during the simulation of this scenario is shown in Figure 18; furthermore, Figure 19 entirely details the performance of the proposed methods during the full simulation.

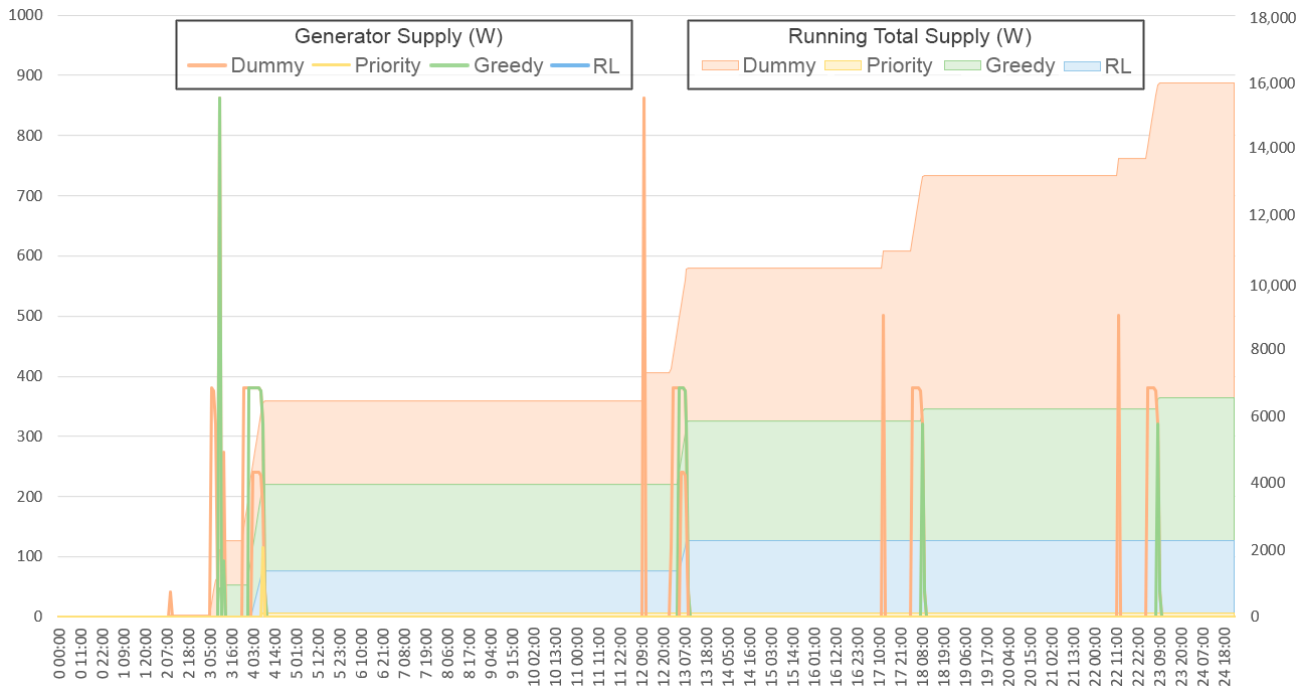


Figure 18. Comparison of generator usage of each method (configuration 3).

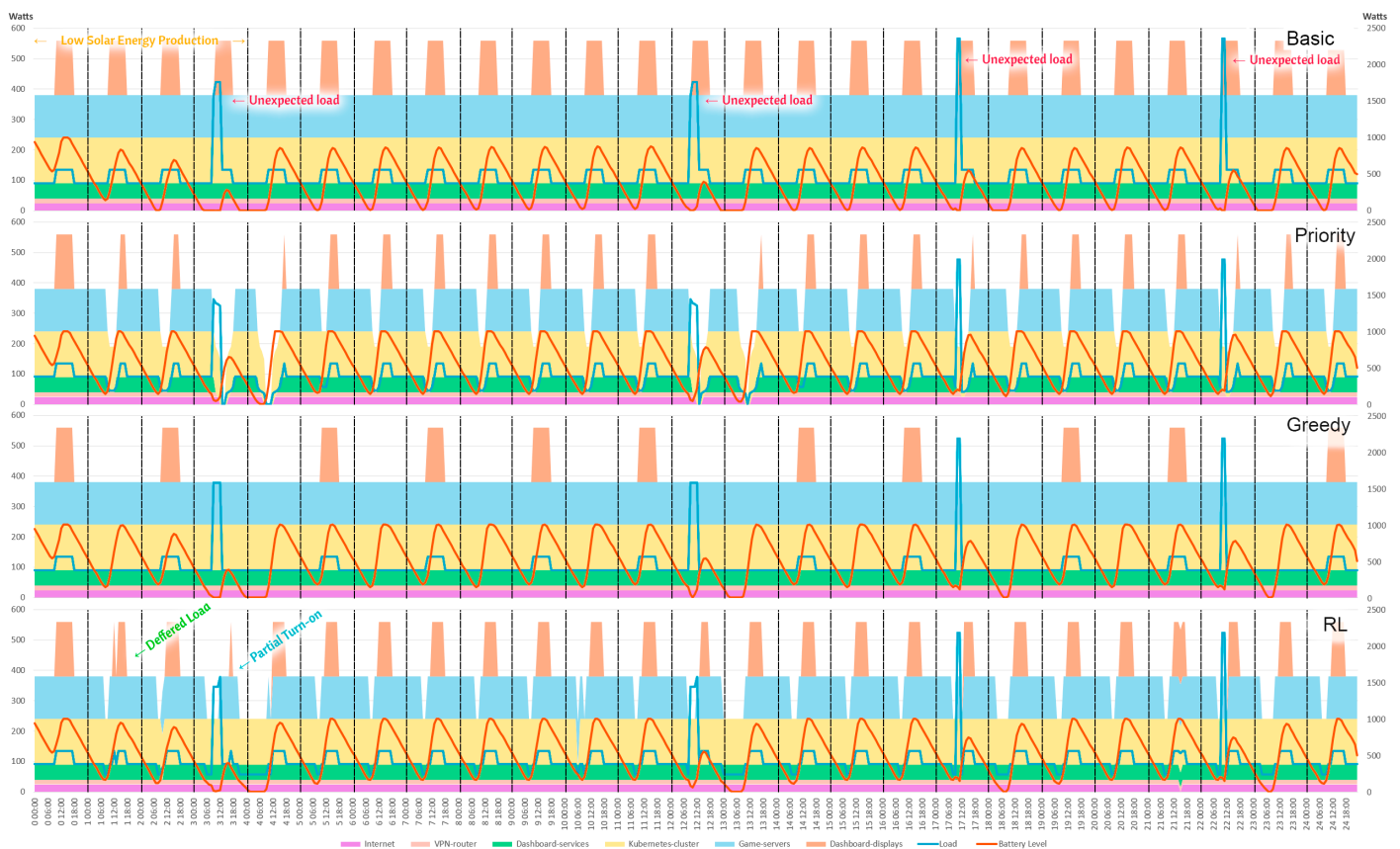


Figure 19. Twenty-five-day simulation of each method (configuration 3).

## 6. Conclusions and Future Work

This paper proposes the design of a smart house service orchestrator intending to reduce the use of non-renewable energy. The energy orchestrator manages services modelled as a cyber-physical system, and we propose using a multiagent reinforcement algorithm to intelligently and dynamically control the energy usage of the services. Therefore, we formulate the environment and the management of each CPS service as independent Q-Learning agents that will perform actions and receive feedback on each time slot.

To verify the performance of the proposed method, an open-source simulator is also developed in which three scenarios are deployed to compare the proposed solution with a priority-based, heuristic-based, and basic-based one. The experimental results show that our proposed method is superior to the priority-based and heuristic-based ones, and it also achieves the intended autonomous and dynamic behaviour without any guidance. In addition, the source code of the simulator and the results of the simulations are available in our GitHub repository for verification and reproducibility of the results.

However, we consider that the performance of our proposal can be improved using neural networks (DQL) to overcome the Q-Learning limitations, such as a low number of states and discrete variables. Furthermore, the architecture of the orchestrator could be enhanced to provide a passive knowledge transfer model following the multilayer RL proposal of our previous work on tasks offloading in edge computing [85].

**Author Contributions:** Conceptualization: A.R.-E., R.R.-E. and A.F.S.G.; formal analysis: A.R.-E. and R.R.-E.; funding acquisition: A.F.S.G.; investigation: A.R.-E. and R.R.-E.; methodology: A.R.-E.; software: A.R.-E.; supervision and validation: R.R.-E. and A.F.S.G.; writing: A.R.-E. and R.R.-E. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the FPI Grant 21463/FPI/20 of the Seneca Foundation in Region of Murcia (Spain) and partially funded by the ONOFRE project (Grant No. PID2020-112675RB-C44) and the MASTERPIECE project (Grant No. 101096836).

**Data Availability Statement:** The most relevant data, such as simulation results, codes, and input data are available on our GitHub, mentioned in Section 6 of the paper. More detailed results and the complete software will be made available on request.

**Conflicts of Interest:** We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

## Abbreviations

The following abbreviations are used in this manuscript:

RL	Reinforcement Learning
CPS	Cyber-Physical System
MPC	Model Predictive Control
SVR	Support Vector Regression
BN	Bayesian Network
DAG	Directed Acyclic Graph
LP	Linear Programming
MILP	Mixed-Integer Linear Programming
DP	Dynamic Programming
GA	Genetic Algorithm
AI	Artificial Intelligence
ANN	Artificial Neural Network
RNN	Recurrent Neural Network
CNN	Convolutional Neural Network
FNN	Feedforward Neural Network
XGBoost	eXtreme Gradient Boosting
LightGBM	Light Gradient Boosting Machine

HVAC	Heating, Ventilating and Air Conditioning
PV	Photovoltaics
EV	Electric Vehicle
HEV	Hybrid Electric Vehicle
PHEV	Plug-in Hybrid Electric Vehicle
W	Watt
Wh	Watts per hour
BSC	Basic control algorithm
PB	Priority-based control algorithm
GDY	Greedy control algorithm

#### Nomenclature of the Priority Method:

aBL	Average battery level percentage
lBT	Low battery threshold
hBT	High battery threshold
mBL	Battery level modifier
svcs	List of services sorted by priority

#### Nomenclature of the Greedy Method:

mB	Daily minimum battery level percentage
C	Trade-off constant
p	priority of the service
pC	Power consumption of the service
bC	Total battery storage capacity

#### Nomenclature of the RL Method:

$\alpha$	Learning rate
$\gamma$	Discount factor
$\epsilon$	Exploration rate
$C_t(S)$	Reward of service "S" at time "t"
$\beta_x$	Trade-off constant for each reward parameter
$S_p$	Service priority
$S_{rt}$	Service current operating time
$S_{wt}$	Service waiting time
$B_t$	Inverse battery level
$B_t^m$	Minimum daily inverse battery level
$G_t$	Current daily generator usage
$Q(s, a)$	State-Action value function

## References

- Gill, S.S.; Tuli, S.; Xu, M.; Singh, I.; Singh, K.V.; Lindsay, D.; Tuli, S.; Smirnova, D.; Singh, M.; Jain, U.; et al. Transformative effects of IoT, Blockchain and Artificial Intelligence on cloud computing: Evolution, vision, trends and open challenges. *Internet Things* **2019**, *8*, 100118. [\[CrossRef\]](#)
- Radanliev, P.; De Roure, D.; Van Kleek, M.; Santos, O.; Ani, U.P.D. Artificial intelligence in cyber physical systems. *AI Soc.* **2021**, *36*, 783–796. [\[CrossRef\]](#) [\[PubMed\]](#)
- Schranz, M.; Di Caro, G.A.; Schmickl, T.; Elmenreich, W.; Arvin, F.; Şekercioğlu, A.; Sende, M. Swarm Intelligence and cyber-physical systems: Concepts, challenges and future trends. *Swarm Evol. Comput.* **2021**, *60*, 100762. [\[CrossRef\]](#)
- Mosterman, P.; Zander, J. Industry 4.0 as a Cyber-Physical System study. *Softw. Syst. Model.* **2016**, *15*, 17–29. [\[CrossRef\]](#)
- Serpanos, D. The Cyber-Physical Systems Revolution. *Computer* **2018**, *51*, 70–73. [\[CrossRef\]](#)
- Cao, K.; Hu, S.; Shi, Y.; Colombo, A.W.; Karnouskos, S.; Li, X. A Survey on Edge and Edge-Cloud Computing Assisted Cyber-Physical Systems. *IEEE Trans. Ind. Inform.* **2021**, *17*, 7806–7819. [\[CrossRef\]](#)
- Khujamatov, K.; Reypanzarov, E.; Khasanov, D.; Akhmedov, N. Networking and Computing in Internet of Things and Cyber-Physical Systems. In Proceedings of the 2020 IEEE 14th International Conference on Application of Information and Communication Technologies (AICT), Tashkent, Uzbekistan, 7–9 October 2020; pp. 1–6. [\[CrossRef\]](#)
- Mazumder, S.K.; Kulkarni, A.; Sahoo, E.A. A Review of Current Research Trends in Power-Electronic Innovations in Cyber-Physical Systems. *IEEE J. Emerg. Sel. Top. Power Electron.* **2021**, *9*, 5146–5163. [\[CrossRef\]](#)
- Belussi, L.; Barozzi, B.; Bellazzi, A.; Danza, L.; Devitofrancesco, A.; Fanciulli, C.; Ghellere, M.; Guazzi, G.; Meroni, I.; Salamone, F.; et al. A review of performance of zero energy buildings and energy efficiency solutions. *J. Build. Eng.* **2019**, *25*, 100772. [\[CrossRef\]](#)



10. Petrolo, R.; Loscri, V.; Mitton, N. Cyber-Physical Objects as Key Elements for a Smart Cyber-City. In *Management of Cyber Physical Objects in the Future Internet of Things: Methods, Architectures and Applications*; Guerrieri, A., Loscri, V., Rovella, A., Fortino, G., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 31–49. [[CrossRef](#)]
11. Gielen, D.; Boshell, F.; Saygin, D.; Bazilian, M.D.; Wagner, N.; Gorini, R. The role of renewable energy in the global energy transformation. *Energy Strategy Rev.* **2019**, *24*, 38–50. [[CrossRef](#)]
12. Celasun, O.; Mineshima, A.; Arregui, N.; Mylonas, V.; Ari, A.; Teodoru, I.R.; Black, S.; Zhunussova, K.; Iakova, D.M.; Parry, I.W.H. Surging Energy Prices in Europe in the Aftermath of the War: How to Support the Vulnerable and Speed up the Transition Away from Fossil Fuels. *IMF Work. Pap.* **2022**. [[CrossRef](#)]
13. Grubb, M. Navigating the crises in European energy: Price Inflation, Marginal Cost Pricing, and Principles for Electricity Market Redesign in an Era of Low-Carbon Transition. Institute for New Economic Thinking Working Papers Series No. 191. 2022. Available online: <https://ssrn.com/abstract=4210683> (accessed on 27 October 2024).
14. Mahmood, Y.; Kama, N.; Azmi, A.; Yaacob, S. An IoT based Home Automation Integrated Approach: Impact on Society in Sustainable Development Perspective. *Int. J. Adv. Comput. Sci. Appl.* **2020**, *11*, 240–250. [[CrossRef](#)]
15. Farzaneh, H.; Malehmirchegini, L.; Bejan, A.; Afolabi, T.; Mulumba, A.; Daka, P.P. Artificial Intelligence Evolution in Smart Buildings for Energy Efficiency. *Appl. Sci.* **2021**, *11*, 763. [[CrossRef](#)]
16. Kumar, A.; Sharma, S.; Goyal, N.; Singh, A.; Cheng, X.; Singh, P. Secure and energy-efficient smart building architecture with emerging technology IoT. *Comput. Commun.* **2021**, *176*, 207–217. [[CrossRef](#)]
17. Kylili, A.; Fokaides, P.A. European smart cities: The role of zero energy buildings. *Sustain. Cities Soc.* **2015**, *15*, 86–95. [[CrossRef](#)]
18. Mason, K.; Grijalva, S. A review of reinforcement learning for autonomous building energy management. *Comput. Electr. Eng.* **2019**, *78*, 300–312. [[CrossRef](#)]
19. Lu, R.; Hong, S.H.; Yu, M. Demand Response for Home Energy Management Using Reinforcement Learning and Artificial Neural Network. *IEEE Trans. Smart Grid* **2019**, *10*, 6629–6639. [[CrossRef](#)]
20. Xu, X.; Jia, Y.; Xu, Y.; Xu, Z.; Chai, S.; Lai, C.S. A Multi-Agent Reinforcement Learning-Based Data-Driven Method for Home Energy Management. *IEEE Trans. Smart Grid* **2020**, *11*, 3201–3211. [[CrossRef](#)]
21. Liu, Y.; Zhang, D.; Gooi, H.B. Optimization strategy based on deep reinforcement learning for home energy management. *CSEE J. Power Energy Syst.* **2020**, *6*, 572–582. [[CrossRef](#)]
22. Mbuwir, B.V.; Ruelens, F.; Spiessens, F.; Deconinck, G. Battery Energy Management in a Microgrid Using Batch Reinforcement Learning. *Energies* **2017**, *10*, 1846. [[CrossRef](#)]
23. Kell, A.J.M.; McGough, A.S.; Forshaw, M. Optimizing a domestic battery and solar photovoltaic system with deep reinforcement learning. *arXiv* **2021**, arXiv:2109.05024.
24. Abedi, S.; Yoon, S.W.; Kwon, S. Battery energy storage control using a reinforcement learning approach with cyclic time-dependent Markov process. *Int. J. Electr. Power Energy Syst.* **2022**, *134*, 107368. [[CrossRef](#)]
25. Li, Y.; Wang, R.; Yang, Z. Optimal Scheduling of Isolated Microgrids Using Automated Reinforcement Learning-Based Multi-Period Forecasting. *IEEE Trans. Sustain. Energy* **2022**, *13*, 159–169. [[CrossRef](#)]
26. Alfaverh, F.; Denai, M.; Sun, Y. Demand Response Strategy Based on Reinforcement Learning and Fuzzy Reasoning for Home Energy Management. *IEEE Access* **2020**, *8*, 39310–39321. [[CrossRef](#)]
27. Zhou, S.; Hu, Z.; Gu, W.; Jiang, M.; Zhang, X.P. Artificial intelligence based smart energy community management: A reinforcement learning approach. *CSEE J. Power Energy Syst.* **2019**, *5*, 1–10. [[CrossRef](#)]
28. Chen, S.J.; Chiu, W.Y.; Liu, W.J. User Preference-Based Demand Response for Smart Home Energy Management Using Multiobjective Reinforcement Learning. *IEEE Access* **2021**, *9*, 161627–161637. [[CrossRef](#)]
29. Nepal, B.; Yamaha, M.; Yokoe, A.; Yamaji, T. Electricity load forecasting using clustering and ARIMA model for energy management in buildings. *Jpn. Archit. Rev.* **2020**, *3*, 62–76. [[CrossRef](#)]
30. Panapongpakorn, T.; Banjerdpongchai, D. Short-Term Load Forecast for Energy Management Systems Using Time Series Analysis and Neural Network Method with Average True Range. In Proceedings of the 2019 First International Symposium on Instrumentation, Control, Artificial Intelligence, and Robotics (ICA-SYMP), Bangkok, Thailand, 16–18 January 2019; pp. 86–89. [[CrossRef](#)]
31. Di Silvestre, M.L.; Riva Sanseverino, E. Modelling energy storage systems using Fourier analysis: An application for smart grids optimal management. *Appl. Soft Comput.* **2014**, *14*, 469–481. [[CrossRef](#)]
32. Serale, G.; Fiorentini, M.; Capozzoli, A.; Bernardini, D.; Bemporad, A. Model Predictive Control (MPC) for Enhancing Building and HVAC System Energy Efficiency: Problem Formulation, Applications and Opportunities. *Energies* **2018**, *11*, 631. [[CrossRef](#)]
33. Bruni, G.; Cordiner, S.; Mulone, V.; Rocco, V.; Spagnolo, F. A study on the energy management in domestic micro-grids based on Model Predictive Control strategies. *Energy Convers. Manag.* **2015**, *102*, 50–58. [[CrossRef](#)]
34. Zhou, F.; Li, Y.; Wang, W.; Pan, C. Integrated energy management of a smart community with electric vehicle charging using scenario based stochastic model predictive control. *Energy Build.* **2022**, *260*, 111916. [[CrossRef](#)]
35. Zamhuri Fuadi, A.; Haq, I.N.; Leksono, E. Support Vector Machine to Predict Electricity Consumption in the Energy Management Laboratory. *J. RESTI (Rekayasa Sist. Dan Teknol. Inf.)* **2021**, *5*, 466–473. [[CrossRef](#)]
36. Ma, Z.; Ye, C.; Ma, W. Support vector regression for predicting building energy consumption in southern China. *Energy Procedia* **2019**, *158*, 3433–3438. [[CrossRef](#)]

37. Zhong, H.; Wang, J.; Jia, H.; Mu, Y.; Lv, S. Vector field-based support vector regression for building energy consumption prediction. *Appl. Energy* **2019**, *242*, 403–414. [[CrossRef](#)]
38. Tian, Z.; Si, B.; Shi, X.; Fang, Z. An application of Bayesian Network approach for selecting energy efficient HVAC systems. *J. Build. Eng.* **2019**, *25*, 100796. [[CrossRef](#)]
39. Shoji, T.; Hirohashi, W.; Fujimoto, Y.; Amano, Y.; Tanabe, S.i.; Hayashi, Y. Personalized Energy Management Systems for Home Appliances Based on Bayesian Networks. *J. Int. Counc. Electr. Eng.* **2015**, *5*, 64–69. [[CrossRef](#)]
40. Amayri, M.; Ploix, S.; Kazmi, H.; Ngo, Q.D.; El Safadi, A. Estimating Occupancy from Measurements and Knowledge Using the Bayesian Network for Energy Management. *J. Sens.* **2019**, *2019*, 1–12. [[CrossRef](#)]
41. Ayenew, M.; Lei, H.; Li, X.; Kekeba, K.; Assefa, M.; Tegene, A.T.; Muhammed, S.B.; Leka, H.L. Data Analytics and Machine Learning for Reliable Energy Management: A Case Study. In Proceedings of the 2022 19th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP), Chengdu, China, 16–18 December 2022; pp. 1–6. [[CrossRef](#)]
42. Ayub, M.A.; Khan, H.; Peng, J.; Liu, Y. Consumer-Driven Demand-Side Management Using K-Mean Clustering and Integer Programming in Standalone Renewable Grid. *Energies* **2022**, *15*, 1006. [[CrossRef](#)]
43. Umetani, S.; Fukushima, Y.; Morita, H. A linear programming based heuristic algorithm for charge and discharge scheduling of electric vehicles in a building energy management system. *Omega* **2017**, *67*, 115–122. [[CrossRef](#)]
44. Bio Gassi, K.; Baysal, M. Analysis of a linear programming-based decision-making model for microgrid energy management systems with renewable sources. *Int. J. Energy Res.* **2022**, *46*, 7495–7518. [[CrossRef](#)]
45. Fedjaev, J.; Amamra, S.A.; Francois, B. Linear programming based optimization tool for day ahead energy management of a lithium-ion battery for an industrial microgrid. In Proceedings of the 2016 IEEE International Power Electronics and Motion Control Conference (PEMC), Varna, Bulgaria, 25–28 September 2016; pp. 406–411. [[CrossRef](#)]
46. Luna, A.C.; Diaz, N.L.; Graells, M.; Vasquez, J.C.; Guerrero, J.M. Mixed-Integer-Linear-Programming-Based Energy Management System for Hybrid PV-Wind-Battery Microgrids: Modeling, Design, and Experimental Verification. *IEEE Trans. Power Electron.* **2017**, *32*, 2769–2783. [[CrossRef](#)]
47. Javadi, M.S.; Gough, M.; Lotfi, M.; Esmaeel Nezhad, A.; Santos, S.F.; Catalão, J.P. Optimal self-scheduling of home energy management system in the presence of photovoltaic power generation and batteries. *Energy* **2020**, *210*, 118568. [[CrossRef](#)]
48. Tischer, H.; Verbic, G. Towards a smart home energy management system—A dynamic programming approach. In Proceedings of the 2011 IEEE PES Innovative Smart Grid Technologies, Perth, WA, Australia, 13–16 November 2011; pp. 1–7. [[CrossRef](#)]
49. Wei, Q.; Liu, D.; Shi, G.; Liu, Y. Multibattery Optimal Coordination Control for Home Energy Management Systems via Distributed Iterative Adaptive Dynamic Programming. *IEEE Trans. Ind. Electron.* **2015**, *62*, 4203–4214. [[CrossRef](#)]
50. Liu, C.; Wang, Y.; Wang, L.; Chen, Z. Load-adaptive real-time energy management strategy for battery/ultracapacitor hybrid energy storage system using dynamic programming optimization. *J. Power Sources* **2019**, *438*, 227024. [[CrossRef](#)]
51. Song, Z.; Guan, X.; Cheng, M. Multi-objective optimization strategy for home energy management system including PV and battery energy storage. *Energy Rep.* **2022**, *8*, 5396–5411. [[CrossRef](#)]
52. Rajasekhar, B.; Pindoriya, N.; Tushar, W.; Yuen, C. Collaborative Energy Management for a Residential Community: A Non-Cooperative and Evolutionary Approach. *IEEE Trans. Emerg. Top. Comput. Intell.* **2019**, *3*, 177–192. [[CrossRef](#)]
53. Arabali, A.; Ghofrani, M.; Etezadi-Amoli, M.; Fadali, M.S.; Baghzouz, Y. Genetic-Algorithm-Based Optimization Approach for Energy Management. *IEEE Trans. Power Deliv.* **2013**, *28*, 162–170. [[CrossRef](#)]
54. Ferrández-Pastor, F.J.; Gómez-Trillo, S.; Nieto-Hidalgo, M.; García-Chamizo, J.M.; Valdivieso-Sarabia, R. Intelligent Power Management System Using Hybrid Renewable Energy Resources and Decision Tree Approach. *Proceedings* **2018**, *2*, 1239. [[CrossRef](#)]
55. Shcherbakov, M.; Kamaev, V.; Shcherbakova, N. Automated Electric Energy Consumption Forecasting System Based On Decision Tree Approach. *IFAC Proc. Vol.* **2013**, *46*, 1027–1032. [[CrossRef](#)]
56. Bassi, A.; Shenoy, A.; Sharma, A.; Sigurdson, H.; Glossop, C.; Chan, J.H. Building Energy Consumption Forecasting: A Comparison of Gradient Boosting Models. In Proceedings of the 12th International Conference on Advances in Information Technology (IAIT 2021), Bangkok, Thailand, 29 June–1 July 2021; [[CrossRef](#)]
57. Lu, H.; Cheng, F.; Ma, X.; Hu, G. Short-term prediction of building energy consumption employing an improved extreme gradient boosting model: A case study of an intake tower. *Energy* **2020**, *203*, 117756. [[CrossRef](#)]
58. Aksoy, N.; Genc, I. Predictive models development using gradient boosting based methods for solar power plants. *J. Comput. Sci.* **2023**, *67*, 101958. [[CrossRef](#)]
59. Abiodun, O.I.; Jantan, A.; Omolara, A.E.; Dada, K.V.; Mohamed, N.A.; Arshad, H. State-of-the-art in artificial neural network applications: A survey. *Heliyon* **2018**, *4*, e00938. [[CrossRef](#)]
60. Xie, S.; Hu, X.; Qi, S.; Lang, K. An artificial neural network-enhanced energy management strategy for plug-in hybrid electric vehicles. *Energy* **2018**, *163*, 837–848. [[CrossRef](#)]
61. Fan, C.; Wang, J.; Gang, W.; Li, S. Assessment of deep recurrent neural network-based strategies for short-term building energy predictions. *Appl. Energy* **2019**, *236*, 700–710. [[CrossRef](#)]
62. Khan, Z.A.; Ullah, A.; Ul Haq, I.; Hamdy, M.; Maria Mauro, G.; Muhammad, K.; Hijji, M.; Baik, S.W. Efficient Short-Term Electricity Load Forecasting for Effective Energy Management. *Sustain. Energy Technol. Assess.* **2022**, *53*, 102337. [[CrossRef](#)]

63. Ganesh, A.H.; Xu, B. A review of reinforcement learning based energy management systems for electrified powertrains: Progress, challenge, and potential solution. *Renew. Sustain. Energy Rev.* **2022**, *154*, 111833. [[CrossRef](#)]
64. Fu, Q.; Han, Z.; Chen, J.; Lu, Y.; Wu, H.; Wang, Y. Applications of reinforcement learning for building energy efficiency control: A review. *J. Build. Eng.* **2022**, *50*, 104165. [[CrossRef](#)]
65. Yu, L.; Qin, S.; Zhang, M.; Shen, C.; Jiang, T.; Guan, X. A Review of Deep Reinforcement Learning for Smart Building Energy Management. *IEEE Internet Things J.* **2021**, *8*, 12046–12063. [[CrossRef](#)]
66. Ladosz, P.; Weng, L.; Kim, M.; Oh, H. Exploration in deep reinforcement learning: A survey. *Inf. Fusion* **2022**, *85*, 1–22. [[CrossRef](#)]
67. Mazyavkina, N.; Sviridov, S.; Ivanov, S.; Burnaev, E. Reinforcement learning for combinatorial optimization: A survey. *Comput. Oper. Res.* **2021**, *134*, 105400. [[CrossRef](#)]
68. Ji, Y.; Wang, J.; Xu, J.; Fang, X.; Zhang, H. Real-Time Energy Management of a Microgrid Using Deep Reinforcement Learning. *Energies* **2019**, *12*, 2291. [[CrossRef](#)]
69. Li, W.; Cui, H.; Nemeth, T.; Jansen, J.; Ünlübayir, C.; Wei, Z.; Zhang, L.; Wang, Z.; Ruan, J.; Dai, H.; et al. Deep reinforcement learning-based energy management of hybrid battery systems in electric vehicles. *J. Energy Storage* **2021**, *36*, 102355. [[CrossRef](#)]
70. Fan, L.; Su, H.; Wang, W.; Zio, E.; Zhang, L.; Yang, Z.; Peng, S.; Yu, W.; Zuo, L.; Zhang, J. A systematic method for the optimization of gas supply reliability in natural gas pipeline network based on Bayesian networks and deep reinforcement learning. *Reliab. Eng. Syst. Saf.* **2022**, *225*, 108613. [[CrossRef](#)]
71. Bisen, D.; Lilhore, U.K.; Manoharan, P.; Dahan, F.; Mzoughi, O.; Hajjej, F.; Saurabh, P.; Raahemifar, K. A Hybrid Deep Learning Model Using CNN and K-Mean Clustering for Energy Efficient Modelling in Mobile EdgeIoT. *Electronics* **2023**, *12*, 1384. [[CrossRef](#)]
72. Kuppusamy, R.; Teekaraman, Y.; Kumar, K. Fuzzy-Based Energy Management System with Decision Tree Algorithm for Power Security System. *Int. J. Comput. Intell. Syst.* **2019**, *12*, 1173. [[CrossRef](#)]
73. Recht, B. A Tour of Reinforcement Learning: The View from Continuous Control. *arXiv* **2019**, arXiv:1806.09460. [[CrossRef](#)]
74. Schreiber, T.; Netsch, C.; Baranski, M.; Müller, D. Monitoring data-driven Reinforcement Learning controller training: A comparative study of different training strategies for a real-world energy system. *Energy Build.* **2021**, *239*, 110856. [[CrossRef](#)]
75. Nazib, R.A.; Moh, S. Reinforcement Learning-Based Routing Protocols for Vehicular Ad Hoc Networks: A Comparative Survey. *IEEE Access* **2021**, *9*, 27552–27587. [[CrossRef](#)]
76. Lee, H.; Song, C.; Kim, N.; Cha, S.W. Comparative Analysis of Energy Management Strategies for HEV: Dynamic Programming and Reinforcement Learning. *IEEE Access* **2020**, *8*, 67112–67123. [[CrossRef](#)]
77. Buşoniu, L.; Babuška, R.; De Schutter, B. Multi-agent Reinforcement Learning: An Overview. In *Innovations in Multi-Agent Systems and Applications—1*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 183–221. [[CrossRef](#)]
78. Liu, H.; Hussain, F.; Tan, C.L.; Dash, M. Discretization: An Enabling Technique. *Data Min. Knowl. Discov.* **2002**, *6*, 393–423. [[CrossRef](#)]
79. Robles-Enciso, A.; Robles-Enciso, R.; Skarmeta, A.F. Sim-PowerCS: An extensible and simplified open-source energy simulator. *SoftwareX* **2023**, *23*, 101467. [[CrossRef](#)]
80. Robles-Enciso, A. MA-RL CPS Simulations Results. 2022. Available online: <https://github.com/alb1183/MARL-CPS-results/tree/main/Journal%20Paper> (accessed on 27 October 2024).
81. Robles-Enciso, A. Sim-PowerCS Simulator, 2022. Available online: <https://github.com/alb1183/Sim-PowerCS/tree/Journal-Paper> (accessed on 27 October 2024).
82. Adam Samorzewski, A.K. Energy models for wireless networks with mobile access nodes. *Przeгляд Telekomunikacyjny* **2022**, *4*, 376–379. [[CrossRef](#)]
83. Robles-Enciso, R. Personal Weather Station—Casa Ruinas—IALGUA2, 2022. Available online: <https://www.wunderground.com/dashboard/pws/IALGUA2> (accessed on 27 October 2024).
84. Inoue, N.; Yamagata, E.; Kataoka, H. Initialization Using Perlin Noise for Training Networks with a Limited Amount of Data. In Proceedings of the 2020 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 10–15 January 2021; pp. 1023–1028. [[CrossRef](#)]
85. Robles-Enciso, A.; Skarmeta, A.F. A multi-layer guided reinforcement learning-based tasks offloading in edge computing. *Comput. Netw.* **2023**, *220*, 109476. [[CrossRef](#)]

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.