PROCEEDINGS OF ECOS 2026 - THE 39TH INTERNATIONAL CONFERENCE ON
EFFICIENCY, **C**OST, **O**PTIMIZATION, **S**IMULATION AND ENVIRONMENTAL IMPACT OF ENERGY SYSTEMS
28 JUNE – 03 JULY, 2026, CONSTANTA, ROMANIA

A Reinforcement Learning Application for the Optimal Management of Multi-Energy System

Joddy Razafitsalama[a], Guido Francesco Frate[a,*], Alessandro Lorenzo Palma[b], Paolo Sdringola[b], Lorenzo Ferrari[a]

[a]University of Pisa, Department of Energy, System, Territory and Construction Engineering (DESTEC), Largo Lucio Lazzarino 1, Pisa, 56122, Italy

[b]Italian National Agency for New Technologies, Energy and Sustainable Economic Development (ENEA), Energy Efficiency Department Research Center Casaccia, Via Anguillarese 301, Rome, 00123, Italy

**Corresponding Author: guido.frate@unipi.it*

Abstract:

The rapid growth of renewable energy sources (RES) introduces significant challenges due to their variability and frequent mismatch with energy demand. Efficient short-term operation of Multi-Energy Systems (MES) is therefore essential to minimize operational costs and ensure reliable RES integration. Advanced control strategies are required to coordinate multiple energy assets under dynamic and uncertain operating conditions. Classical optimization techniques such as Mixed-Integer Linear Programming (MILP) provide optimal solutions but are computationally demanding and depend on accurate forecasts. Conversely, Reinforcement Learning (RL) enables fast real-time control but often struggles with long training times and suboptimal convergence when trained from scratch. This study proposes an RL-based control approach designed to inherit the optimal management strategy obtained from a MILP formulation. The RL agent interacts directly with the MES using current and historical boundary conditions as observations and controls selected components, including a Combined Heat and Power (CHP) unit and a gas boiler. Reward shaping is employed to embed MILP-optimal schedules as guidance, enabling the agent to learn optimal control patterns through deviation-based penalties. The Soft Actor-Critic (SAC) algorithm is adopted for training. The methodology is evaluated on a MES comprising electrical and thermal demands, photovoltaic generation, grid connections, CHP, thermal energy storage, and a backup gas boiler. Results show that incorporating short-term historical information (2 hours) allows the RL agent to closely replicate the MILP scheduling behavior. However, forecast-induced deviations lead to operational cost increases ranging from 10 to 20 % compared to the MILP benchmark. The proposed RL approach successfully inherits MILP-based strategies, achieving near-optimal performance while improving real-time applicability under uncertainty.

Keywords:

Mixed-integer linear programming; Multi-energy system; Optimal energy management; Reinforcement learning; Soft Actor-Critic.

1. Introduction

Renewable energy sources (RES) accelerated expansion requires careful energy resource management in the energy transition context. RES integration and optimal management can be challenging due to their high variability and the fact that their production often does not coincide with energy demand. A mix of storage technologies and sector coupling approaches can be used to cope with RES variability, thus leading to the so-called multi-energy system (MES). The management of MES involves coordinating multiple energy assets and conversion pathways, resulting in a high level of system complexity. Consequently, specialized optimization techniques are required to determine optimal operational strategies, and a wide range of optimization algorithms has been proposed for this purpose.

While such optimization approaches improve system efficiency, reduce operational costs, and enhance RES integration, traditional mathematical formulations typically rely on extensive input data, including accurate forecasts and detailed component models. This reliance leads to significant computational burdens, particularly in large-scale or long-term simulations. Therefore, exploring alternative approaches that can mitigate these limitations is of considerable interest.

This study explores Reinforcement Learning (RL) as an alternative to conventional optimization for MES management. While RL can learn independently, here it is initialized with optimal trajectories from mathematical

optimization. Although optimization is needed initially to generate training data, the trained RL models can accurately approximate optimal strategies in long-term MES simulations, drastically reducing computational time.

RL is particularly suitable for real-time MES dispatch, lowering computational demands and operational costs despite higher upfront training effort. Unlike traditional optimization, RL does not require explicit forecasting, simplifying modeling and reducing operational complexity.

Results show that RL can effectively replicate optimization-based strategies while maintaining cost-efficiency. However, research remains limited, and a standardized framework for systematically replacing optimization with RL in MES applications is still lacking.

1.1. Literature overview

While MES provide a structured framework for integrating multiple energy carriers, as highlighted by Mancarella [1], Mixed-Integer Linear Programming (MILP) has become a key tool for optimizing its energy management. MILP is widely used in optimizing polygeneration and Combined Cooling, Heat, and Power (CCHP) systems because it can reliably find global minima and handle significantly more variables than non-linear techniques. For instance, Bischi et al., [2] developed a MILP model for short-term CCHP planning, incorporating non-linear performance curves through piecewise linear approximations. Similarly, Urbanucci [3] examined the effectiveness of MILP in polygeneration systems, highlighting challenges such as high dimensionality and limited nonlinear modeling capabilities.

Reinforcement Learning (RL) has recently emerged as a promising alternative to traditional optimization approaches. As reviewed by Cao et al., [4], RL enables data-driven control by learning optimal policies through interaction with the environment, thereby avoiding the need for explicit system modeling and forecast-dependent inputs. This characteristic makes RL particularly attractive for complex and uncertain energy systems.

Several studies have demonstrated the effectiveness of RL in energy management applications. For example, Ji et al., [5] formulated microgrid operation as a Markov Decision Process (MDP) and applied Deep Reinforcement Learning (DRL) to derive real-time control policies that minimize operational costs without relying on explicit forecasts. Similarly, Kim, Lim [6] applied RL to smart building energy management, showing that adaptive learning strategies can progressively reduce energy costs under uncertain and dynamic conditions.

More recent contributions further highlight the potential of RL in complex energy systems. Kang et al., [7] applied DRL to the optimal planning of hybrid energy storage systems under renewable energy curtailment, demonstrating that RL can achieve near-optimal performance compared to stochastic optimization while maintaining robustness to uncertainty. In a related context, Ye et al., [8] emphasized the importance of accurate system representation in MES optimization, proposing data-driven and digital twin-based methods to improve model fidelity—an aspect that RL can inherently address through learning-based approaches.

Combining MILP and RL improves MES management by using MILP to generate optimal solutions offline and RL to learn from them, enabling fast, real-time decision-making. However, a fully developed method where RL completely replaces MILP is still missing. This study proposes a framework where RL is trained to mimic MILP's optimal strategies and then used independently in real-time, avoiding heavy computation.

1.2. Innovative contribution

This study proposes a novel approach for near-optimal management of MES, in which traditional optimization algorithms are replaced by trained reinforcement learning models. This substitution significantly reduces computational effort while enabling faster solution times.

A key contribution of the proposed method is its ability to operate without relying on future forecasting data, using only current and historical information. This feature reduces data dependency while maintaining near-optimal performance in terms of operational cost. Moreover is introduced an approach which leverage MILP optimality.

In addition, the study introduces:

- A framework for integrating RL into MES optimization.
- A tailored training strategy in which reward shaping plays a central role in guiding the learning process.
- A comparative analysis between RL-derived operational schedules and optimal solutions obtained via MILP.
- A performance evaluation methodology based on global KPIs, as well as daily and cumulative operational costs and their temporal evolution.

2. Materials and methods

In this section, the methodology applied in this study is presented.

This study aims to train an RL model to manage a MES case study using MILP results as a reference strategy to learn from, inheriting the underlying control patterns. To achieve this goal, a MES residential case study is used and defined (Section 2.1.). A MILP model is defined, and the related optimal schedule results are used as a reference guidance strategy for the RL model to make it learn through the reward (Section 2.2.). Effectively applying RL models to MES requires an integration framework, which is outlined in Section 2.3. Finally, the performance of trained RL models for the case study is compared to that of MILP as highlighted in Section 3.

2.1. Case study

The case study, originally introduced in Razafitsalama et al., [9], is briefly summarized here.

It considers a residential system aggregating the energy demand of 35 households, with requirements limited to electricity, heating, and domestic hot water (DHW). The system includes a Combined Heat and Power unit (CHP), a gas boiler (GB), thermal energy storage (TES), and photovoltaic (PV) generation, and is connected to both the electricity and natural gas grids to meet energy demand. The total annual demand amounts to 183 MWh for electricity, 84 MWh for heating, and 50 MWh for DHW.

The devices selected to meet the demand are listed in Table 1, including their rated design values. As well, the schematic layout of MES is illustrated in Figure 1 here below.

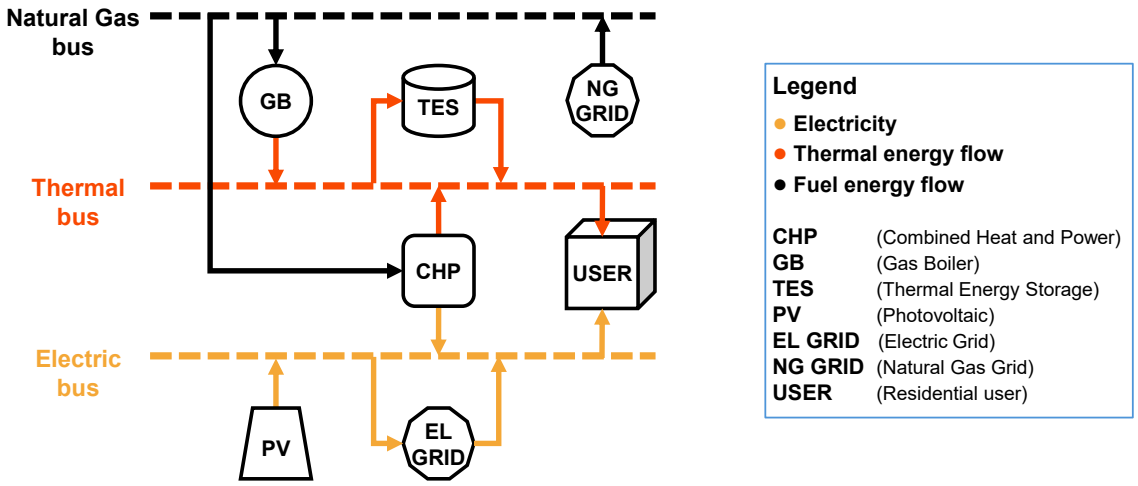

Figure 1. *Schematic layout of the multi-energy system [9].*

PV production is unconstrained and not directly controlled, making it a boundary condition. For TES, the state of charge (SoC) is restricted between 10 % and 100 %, with a charge and discharge period of 2 hours. The minimum operational loads are set at 30 % for CHP and 4 % for GB[1]. The performance curves for CHP and GB follow the definitions in Bischi et al., [2]. The other performance parameters of the remaining device are assumed to be constant.

Table 1. *Rated values of each units defining the multi-energy system [9].*

UNIT	W_{nom}, kW_{el}	Q_{nom}, kW_{th}	F_{nom}, kW_{th}	U_{nom}, kWh_{th}
EL GRID	±32	–	–	–
NG GRID	–	–	335	–
CHP	16	26	50	–
GB	–	273	285	–
TES	–	±80	–	160
PV	25	–	–	–

Energy prices, sourced from Italian Data Stats ARERA [10] and Day-Ahead Market profiled from *Gestore dei Mercati Energetici* (GME) [11], include average values of expenditure on energy. Prices are variable throughout the year, and the energy price ratio, denoted later as α, was adjusted to assess the algorithm's response to different system management strategies under varying price scenarios, as Mitra et al., [12] highlighted in his study.

1 For simplicity, the gas boiler is simplified as one aggregated unit of three with minimum operational loads each of 20 %. The 4 % of minimum loads represents the smaller GB inside the aggregated one

2.1.1. Mixed Integer Linear Programming formulation

The problem formulation is defined by introducing variables that represent the energy inputs and outputs of each device, along with auxiliary binary variables used to encode mutually exclusive operating modes (e.g., on/off status, charging/discharging states).

Energy balance across the system and the satisfaction of demand are enforced through a set of equality and inequality constraints involving these variables. Within the optimization framework, each device is modeled through dedicated constraints that capture its operational characteristics. For the TES, constraints ensure that the SoC remains within prescribed bounds at each time step and that the final SoC matches the initial value, thereby enforcing daily cyclic operation. The performance of the CHP unit and the GB is inherently nonlinear. To retain computational tractability, these nonlinear relationships are approximated using a piecewise-linear formulation.

To guarantee that user energy demand is met, energy balance equations are imposed on virtual nodes, referred to as energy buses, where all relevant energy flows converge and are redistributed. The overall MES configuration, including the interconnection of devices, grids, and users through these energy buses, is already illustrated in Figure 1.

The objective function is defined as the minimization of the total daily operating cost, which includes fuel and electricity purchase costs and revenues from electricity sales. The hourly operational cost is expressed as:

$$c_t = c^{el}_{purc,t} - c^{el}_{sell,t} + c^{fuel}_{purc,t}\,. \tag{1}$$

In the optimization model, time-dependent prices, denoted as p (€/kWh), (hourly for electricity and daily/monthly for natural gas) are used to evaluate the model. A time horizon of $\tau = 24$ h is adopted, consistent with short-term operational scheduling as discussed by Amin, Mourshed [13]. The optimization is performed on a rolling daily basis over the entire year. The daily objective function is thus formulated as:

$$C^{MILP}_d = f_{obj} = \min_{\{P^u\}} \left(\sum_{t=1}^{\tau=24} c^{MILP}_t \right). \tag{2}$$

The solution of the MILP problem yields the optimal schedule $\{P^u\}$, which represents the set of control actions assigned to each device over the daily horizon to minimize the objective function. This optimal scheduling data is subsequently used both as a benchmark and as training data for the learning-based approach.

A closely related formulation is later adopted to define the environment with which the RL agent interacts (see Section 2.3.1.).

2.2. How to leverage the optimal results data to train reinforcement learning models

The study aims to train an RL model to perform optimal energy management, leveraging the MILP optimality choices given the same inputs (except for the forecasts). Having the management performed controlling specific cost-effective device. Therefore, the optimal data from MILP are used as reward information in the training of a RL algorithm. As stated, only a subset of the boundary conditions used in the optimization is employed in model training and utilization. In particular, all data that refer to the future with respect to the moment when the optimization is run (forecasts) are not used. Therefore, the RL algorithm is assumed not to know the future. In other words, when the RL decides how to operate the system at a time step t-th, it will only use information known at that particular time step and past information (e.g., what occurred at the previous time steps).

The rationale behind this choice is that MILP can make optimal decisions because it has perfect knowledge about the future. While the RL algorithm will never be able to make exactly the same choices without comparable knowledge, it may learn to make ”wise” choices based on present and past information by inheriting the MILP behavior, implicitly distilling a ”code of conduct” that approximates MILP optimal choice capabilities. Furthermore, the choice to enter information through the reward allows us not to break the stochastic behavior through the MDP.

The schematic in Figure 2 shows how the main components of RL-architecture interacts. As well, how the optimal results from MILP are used to train the RL algorithms and how these are later used to perform comparisons. The main steps in the procedure are the following:

1. Generate the optimal data results for the MES.
2. Training of the reinforcement learning algorithm through reward’s inheriting optimal data information.
3. Test and employment of the trained RL models.
4. Comparisons of the results between RL and MILP schedules.

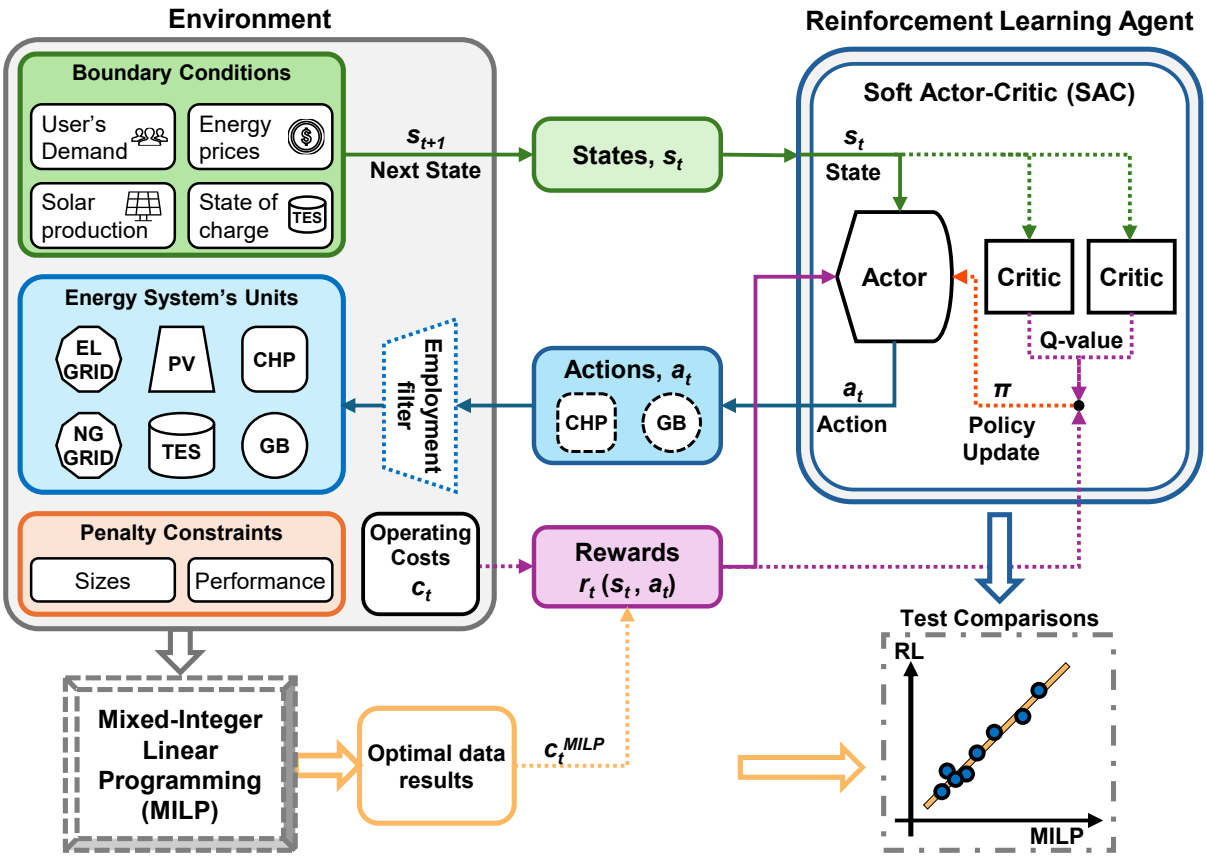

Figure 2. *Representation of the components of the proposed RL-based approach to reach out optimal management of the multi energy system.*

2.3. Reinforcement Learning

Reinforcement learning is a broad area within artificial intelligence concerned with how an *agent* learns to select *actions* based on the current *state* of an environment, in order to maximize a cumulative *reward* signal. Through iterative interaction, the agent observes the system state, takes an action, and receives feedback in the form of a reward, gradually improving its *policy* via trial-and-error as [14] demonstrates.

In this study, the Soft Actor-Critic (SAC) [15] algorithm is adopted to train the agent, motivated by its suitability for MES management. As an off-policy method, SAC achieves high sample efficiency, which is critical when training on data generated from computationally expensive optimization. Additionally, its entropy-regularized objective promotes exploration and robustness under uncertainty, such as fluctuating demand and renewable generation.

Compared to alternatives, SAC offers clear advantages. Proximal Policy Optimization (PPO) is stable but data-intensive due to its on-policy nature, while Deep Deterministic Policy Gradient (DDPG) suffers from instability and hyperparameter sensitivity. SAC overcomes these limitations by combining stable stochastic policies with improved convergence, making it well-suited for continuous control problems like MES operation.

2.3.1. Environment, States and Actions

2.3.1.1. Environment

The environment represents a model-based abstraction of the MES with which the RL agent interacts. It encapsulates the system dynamics and enforces all operational constraints, including energy balances, device performance limits, demand satisfaction, and minimum and maximum operating conditions.

At each time step, the environment receives an action from the agent and returns the corresponding next state, computed by solving the underlying energy balance equations. Device outputs are determined consistently with system constraints and energy flows across the network.

To formalize the energy balance, let e_t^{res} (kWh) denote the residual energy demand at time step t, defined as:

$$e_t^{res} = e_t^{dem} - \sum_u e_t^{in,u} + \sum_u e_t^{out,u}, \tag{3}$$

where e_t^{dem} represents the demand, and $e_t^{in,u}$, $e_t^{out,u}$, denote the input and output energy flows of device u, respectively. The residual energy is used to distinguish between unmet demand and excess production.

Specifically, unmet demand is defined as $e_t^{req} = e_t^{res}$, while energy curtailment due to overproduction is defined as $e_t^{curt} = -e_t^{res}$. Both quantities directly influence the reward signal during training.

The same operating cost formulation introduced in (1), including fuel and electricity purchase costs and revenues from electricity sales, is embedded within the environment and used as the basis for reward computation, as detailed in Section 2.3.2.

2.3.1.2. States

The state variables represent the information observed by the RL agent at each time step and constitute the basis for both policy learning and decision-making. Their definition is critical, as they must capture all relevant system information while avoiding unnecessary complexity.

In this work, the state is defined through a set of boundary conditions (BCs) describing the energy-related inputs of the MES:

- the heating and electricity energy demand, e_t^{dem} (kWh);
- uncontrolled output energy from photovoltaic, e_t^{PV} (kWh);
- the energetic SoC of the TES, SoC_t^{TES} (kWh), that is updated inside the environment every timestep;
- the energy prices $p_{purc,t}^{el}$, $p_{sell,t}^{el}$ on electricity and $p_{purc,t}^{fuel}$ on gas (€/kWh).

These variables are selected as they are readily available in practical applications and sufficient to describe the system dynamics. This choice ensures a compact state representation, reducing the dimensionality of the learning problem without sacrificing essential information. Moreover, temporal patterns are implicitly captured through the evolution of demand and price signals.

Let $\boldsymbol{\mathcal{S}}$ denote the state space of those BCs and $\boldsymbol{s}_t$ the state from the environment at timestep t, defined as:

$$\boldsymbol{s}_t = \left(e_t^{dem}, e_t^{PV}, SoC_t^{TES}, p_{purc,t}^{el}, p_{sell,t}^{el}, p_{purc,t}^{fuel}, \alpha_t\right), \quad \boldsymbol{s}_t \in \boldsymbol{\mathcal{S}}\,. \tag{4}$$

2.3.1.3. Actions

At each time step t, the RL agent selects an action from the action space $\boldsymbol{\mathcal{A}}$ under a specified policy, to meet the system energy demand. The action space is defined as continuous, reflecting the operational flexibility of the energy conversion devices.

In this study, the control actions correspond to the power outputs of the Combined Heat and Power (CHP) unit and the Gas Boiler (GB). This strategy is possible by leveraging energy balances at each timestep through a rule-based strategy.

For a generic unit u, let denote $\boldsymbol{\mathcal{A}}^u$ the continuous action space is bounded between zero output and its nominal capacity P_{nom}^u, i.e. $\boldsymbol{\mathcal{A}}^u \in \{[0, P_{nom}^u]\}$.

Accordingly, the overall action space is defined as a combination of the ones above: $\boldsymbol{\mathcal{A}} = \boldsymbol{\mathcal{A}}^{CHP} \times \boldsymbol{\mathcal{A}}^{GB}$.

Let $\boldsymbol{a}_t \in \boldsymbol{\mathcal{A}}(\boldsymbol{s}_t)$ denote the action proposed by the RL at timestep t under state $\boldsymbol{s}_t$ according to a certain policy π, which defines the mapping from states to actions. Further details on the policy are provided in Section 2.3.2.

2.3.2. Rewards

The reward is the most important key element that affects good learning behavior during the training, but its construction needs a difficult step-by-step fine-tuning. The final purpose is to minimize the daily operational costs, so the reward has to account for at least the hourly costs performed by RL itself.

Let $r_t(\boldsymbol{s}_t, \boldsymbol{a}_t)$ denote the reward function that returns a cost value indicating how much money the MES must pay for the energy used to operate it, when action $\boldsymbol{a}_t$ is taken at state $\boldsymbol{s}_t$.

To account for the impact of the current action on future rewards, the total discounted reward at timestep t under a given policy π, denoted by R_t^π, is defined as the sum of the instant reward at timestep t and discounted rewards from the next timestep, $t+i$, as follows:

$$R_t^\pi = r_t(\boldsymbol{s}_t, \boldsymbol{a}_t) + \sum_{i=1}^{\infty} \gamma^i \cdot r_t(\boldsymbol{s}_{t+i}, \boldsymbol{a}_{t+i})\,, \tag{5}$$

where $\gamma \in [0, 1]$ denotes the discount factor that determines the importance of future rewards from the next timestep, $t+i$. In our approach γ is set 0.9, that implies that the system weighs both current reward and future long-term rewards almost equally.

The objective of the RL is to find an optimal scheduling policy, π, that maximizes the total expected rewards R_t^π when starting in state $\boldsymbol{s}_0$.

The reward r_t is given by the environment as an indicator to guide the update direction of the policy. In the energy systems optimal scheduling problem, the reward function should guide the RL agent to take actions that minimize the operational cost (1) while enforcing the power balance constraints.

Thus, to enhance the training performance of the RL, some numerical penalty must be introduced inside the reward. These penalties have to be introduced as the RL is an iterative process, does not have any future information, so it has to be guided by some penalty or reward, distilling the right behavior of the training process.

The penalties are meant to be related to the violations of some constraints. Let g_t (€) define the penalty at timestep t, the penalty constraints-related are listed here below, where:

- g_t^{req} (€), introduces a weighted penalty proportional to the unsatisfied residual demand e_t^{req} for each thermal and electrical. The penalty weights are energy prices related so they vary along the variable prices; thus, let $\lambda_t^e = p_{purc,t}$ (€/kWh), the penalty denoted as $g_t^{req} = \lambda_t^e \cdot e_t^{req}$;
- g_t^{curt} (€), introduces a weighted penalty proportional to the energy curtailment e_t^{curt} for each thermal and electrical. The penalty weights, as the previous one, are energy prices related, thus, the penalty is denoted as $g_t^{curt} = \lambda_t^e \cdot e_t^{curt}$;
- g_t^{TES} (€), introduces a weighted penalty to the SoC violation in energy terms. The penalty weight set to $\lambda^{TES} = 0.5$ €/kWh is defined;
- g_t^{load} (€), introduces a weighted penalty to the minimum and maximum load for the GB and the CHP. However, as the output of these devices is also an action, the penalty constraints have to be smooth, which means to avoid the agent from completely bypassing the values inside that range, and also because the no-load is effectively a possible outcome of actions.
  Accordingly, a bell curve is defined inside the range of no and minimum load (see Footnote 1). Those curves are illustrated in Figure 3 (a). The maximum penalty value is respectively 5 € and 3 € for GB and CHP, and the thresholds are min-load related.

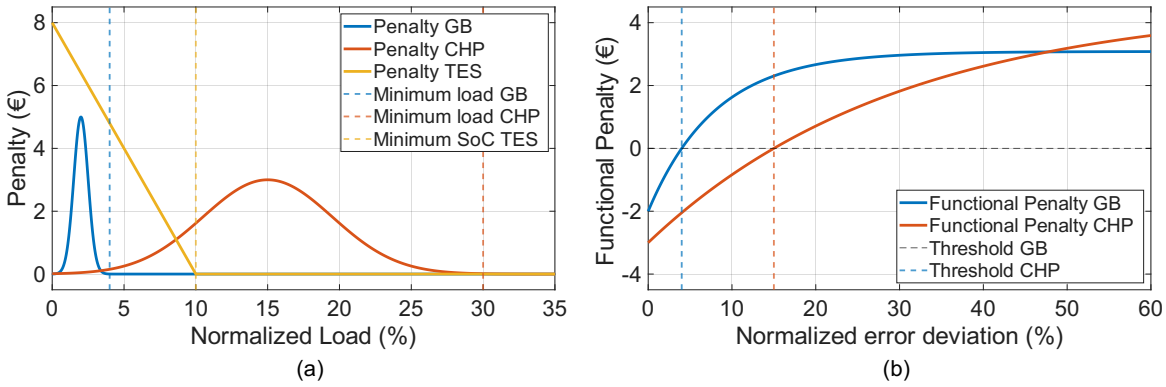

Figure 3. *Penalty curves representation related to the: (a) Normalized load for GB, CHP and TES; (b) Normalized error deviation from optimal schedule for GB and CHP.*

Thus, the operative expenditure from RL that takes into account the above-mentioned penalty and the costs from (1) is:

$$c_t^{RL} = c_t + g_t^{res} + g_t^{curt} + g_t^{TES} + g_t^{load} \quad (€)\,. \tag{6}$$

As a next step, inside the reward is introduced a numerical costs distance between the one performed by RL (6) and the one of MILP (1) as $-(c_t^{RL} - c_t^{MILP})$, making it be able to minimize the numerical distance from MILP.

The chance to insert optimal information inside the reward allows us to define a functional penalty related to the error deviation of an action of device u relative to MILP's schedule. Let g_t^u (€) denote the functional penalty related to representing an exponential function of the relative absolute error defined as: $|a_t^u - e_t^{MILP,u}|$. It is defined to work both as a payoff and as a penalty for distance below or over a threshold, respectively. This will be added later in the reward signal. Those curves are illustrated in Figure 3 (b).

Considering that the main purpose is to perform a daily optimal management, a numerical distance, in terms of daily operative expenditure (Daily OPEX), from the MILP optimal cost for each d-th day, C_d (€), is defined.

Therefore, every τ steps, performed along the training process, the cost from (6) is being summed, defining the daily operational expenditure from RL as C_d^{RL} (€). As these expenditures are expected to be quite relevant, a scaling factor $\sigma = 0.3$ is defined to limit the overcounting reward of the above-mentioned.

2.3.3. Agent

The agent is based on the Soft Actor-Critic (SAC) algorithm, selected for its stability in highly variable environments. SAC employs a dual-critic architecture with target networks and an experience replay buffer, reducing overestimation bias in value function learning.

The architecture comprises one actor and two critic neural networks. Inputs are normalized and match the state space dimensionality. Each network includes two hidden layers with 256 units, providing a standard yet

sufficiently expressive deep neural network configuration.

The actor outputs the parameters of a Gaussian policy (mean and standard deviation) over the continuous action space, while each critic estimates a scalar Q-value, representing the expected discounted return. The dual critics improve value estimation and guide policy updates.

The discount factor is set to $\gamma = 0.9$. Learning rates are 1×10^{-3} for the critics and 1×10^{-4} for the actor, ensuring stable training dynamics. For bounded actions, outputs are appropriately transformed and scaled, while the unbounded Gaussian is retained for entropy computation during training.

All networks use the Gaussian Error Linear Unit (GELU) activation function, which provides smoother nonlinearities than Rectified Linear Unit (ReLU) and improves training stability as [16] demonstrates.

A complete summary of the parameters is provided in Table 2.

Table 2. *Soft Actor-Critic agent*

Parameter	Actor	Critic 1 & 2
Learning rate	1×10^{-4}	1×10^{-3}
Entropy loss	0.02	
Gradient threshold	1	1
Mini batch size	64	
Experience buffer length	6×10^{5}	
Discount factor	0.9	
Layer ×Hidden Nodes	2 × 256	2 × 256
Activation function	GELU	GELU

2.3.4. Training and model evaluation

Following all the explanations, the algorithm must be trained. The dataset is split using a 25 % holdout of randomly selected days (91 test days), separating data into training and test sets for model development and evaluation. Thus, the training set covers patterns from all over the year, counting a 6576 timestep, which are also the training steps. The whole training is achieved by imposing the number of episodes set to 3000, to reach an overall number of steps of 20 million. Finally, the learning rate (defined in Table 2) is being decreased every 200 episodes by a factor of 0.9 to make the agent more exploitative as suggested by [17].

During the training and subsequently for the evaluation, a ”deployment” filter is inserted. It is as crucial as the surrogate model itself. Beyond filtering values based on feasibility, size, performance, and state constraints, it plays a key role in determining the priority of devices in meeting energy demand, defining a rule-based strategy. The need for such a strategy arises because, given the knowledge of demand, the device emulated through actions leverages energy balances at each bus to met demand.

2.3.5. Test Cases

Considering what is described in Section 2.3.2., four main Cases are defined by setting the reward signal in the training process. The test cases are as follows:

- Case 1: $r_t(\boldsymbol{s}_t, \boldsymbol{a}_t) = -\ c_t^{RL}$.

  It represents the baseline case from the RL approach, where the reward is set to minimize the hourly operating cost.
- Case 2: $r_t(\boldsymbol{s}_t, \boldsymbol{a}_t) = -\left(c_t^{RL} - c_t^{MILP}\right)$.

  It represents the case where the reward is set to minimize the hourly cost distance from MILP.
- Case 3: $r_t(\boldsymbol{s}_t, \boldsymbol{a}_t) = -\left(c_t^{RL} - c_t^{MILP}\right) - \sigma \cdot \left(C_d^{RL} - C_d^{MILP}\right)$.

  It represents the case where the reward is set to minimize the hourly cost distance from MILP and the daily operating expenditure with a scaling factor σ.
- Case 4: $r_t(\boldsymbol{s}_t, \boldsymbol{a}_t) = -\left[(c_t^{RL} + g_t^{u}) - c_t^{MILP}\right] - \sigma \cdot \left(C_d^{RL} - C_d^{MILP}\right)$

  It represent the case where, the previous reward (Case 3) includes the functional penalty g_t^u, and moreover, the state input $\boldsymbol{s}_t$ incorporates also two previous timestep, in this case $\boldsymbol{s}_t \in \{\boldsymbol{s}_t, \boldsymbol{s}_{t-1}, \boldsymbol{s}_{t-2}\}$.

To compare results, an analysis of the cumulative rewards during the training is performed to determine the influence of each strategy.

As well, performance metrics such as the normalized root-mean squared error (NRMSE) and the coefficient of variation (CV) are computed for both the objective function and the predicted output. The NRMSE effectively represents device output, as it is normalized to the operating range, while the CV is more relevant for expenditure analysis, as it accounts for variance relative to the mean, which fluctuates based on prices, periods,

seasons, and demand request itself.

Finally, a baseline scenario (Case 0) is introduced to further assess the proposed methodology beyond comparison with MILP. Given that RL is itself an iterative, sequential decision-making approach, it is also relevant to benchmark it against a strategy that does not exploit future information. To this end, Case 0 applies a myopic, hour-by-hour cost-minimization rule. As this approach lacks foresight, its performance is expected to be inferior to the other cases, thereby providing a lower bound against which the effectiveness of the proposed framework can be evaluated.

Simulations and analysis were performed with MATLAB 2025b, and results are presented in Section 3.

3. Results and Discussion

In this section, the methodology described in Section 2., where the aim is to train an RL model to optimally manage a MES leveraging MILP optimal data, is applied to the test cases presented in Section 2.3.5., and the related results are presented.

The first analysis focused on comparing the training performance through the cumulative reward after each case ran for 3000 episodes. The reward trend in Fig. 4 highlights how the different reward signals impact agent training.

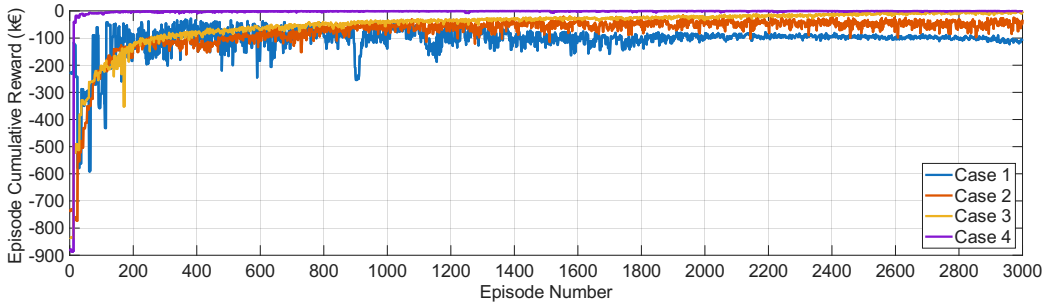

Figure 4. *Comparison of episode's cumulative reward trends during RL's agents training for different reward signals from each case.*

It can be observed that in terms of final cumulative reward, Case 4 has the highest of ~−800 €, whereas the worst one is Case 1 with a final value of −150 000 €. This discrepancy is explained as Case 1 tries to minimize the operating costs, including penalties, in contrast to the other, where they minimize a numerical distance from MILP. This outlines that the worst case hereinafter could be the first. It is also possible to observe that from Cases 2 and 3, the episode rewards saturate approximately after 2000 episodes. While in Case 4, after 200 episodes, the rewards can't increase, so the policy could already be optimal. Thus, this means that with these approaches, the training time could be reduced too.

To illustrate how the trained agents behave to perform actions inside the system, various carpet plots of the output schedules over the test days were generated. The corresponding electrical-side figures are omitted for conciseness, as their patterns remain consistent across cases. Figure 5 compares the thermal-side management strategies applied by different algorithms under variable price scenario. The MILP schedules are displayed at the top, followed by the test cases' results in order.

In this comparative, it can be observed that Case 1 (Fig. 5 (b)), as expected, is the worst compared to MILP, while the others show a related behavior. Moreover, it is observable a poorly performance in the central days (summer representative) where the effective optimal actions are difficult to make due to the high variability of users' demand. The TES-related schedule is from the employment filter that successfully replicates the MILP ones. Case 2 and 4 are much more cost sensitive due to the GB that operate always to fulfill demands, at minimum load with curtailing of the heat flow rate, which has a cost in terms of fuel consumption that increases the total expenditure.

The next analysis focuses on comparing the numeric performance of RL agents against MILP. NRMSE is computed for device outputs, while CV is used for operational costs, Daily and Yearly OPEX, since both serve as normalized error metrics. The bar plots in Fig. 6 (a) summarize and compare these values across all test cases. It can be observed that there is a slight trend toward the more detailed reward signal, from Case 1 to Case 4. The NRMSE for the device is relatively high, denoting a high deviation from the actions decided by RL to the MILP ones, which affects the system management through the employment filter (i.e., EL GRID and NG GRID). TES and the EL GRID, despite being the flexible component, for the thermal and electrical side respectively, can't absorb forecast errors, exhibiting the highest absolute errors. Where the reward carries more information (Case 4), the whole system management performance increases with NRMSE up to 28 % for CHP and 4 % for GB.

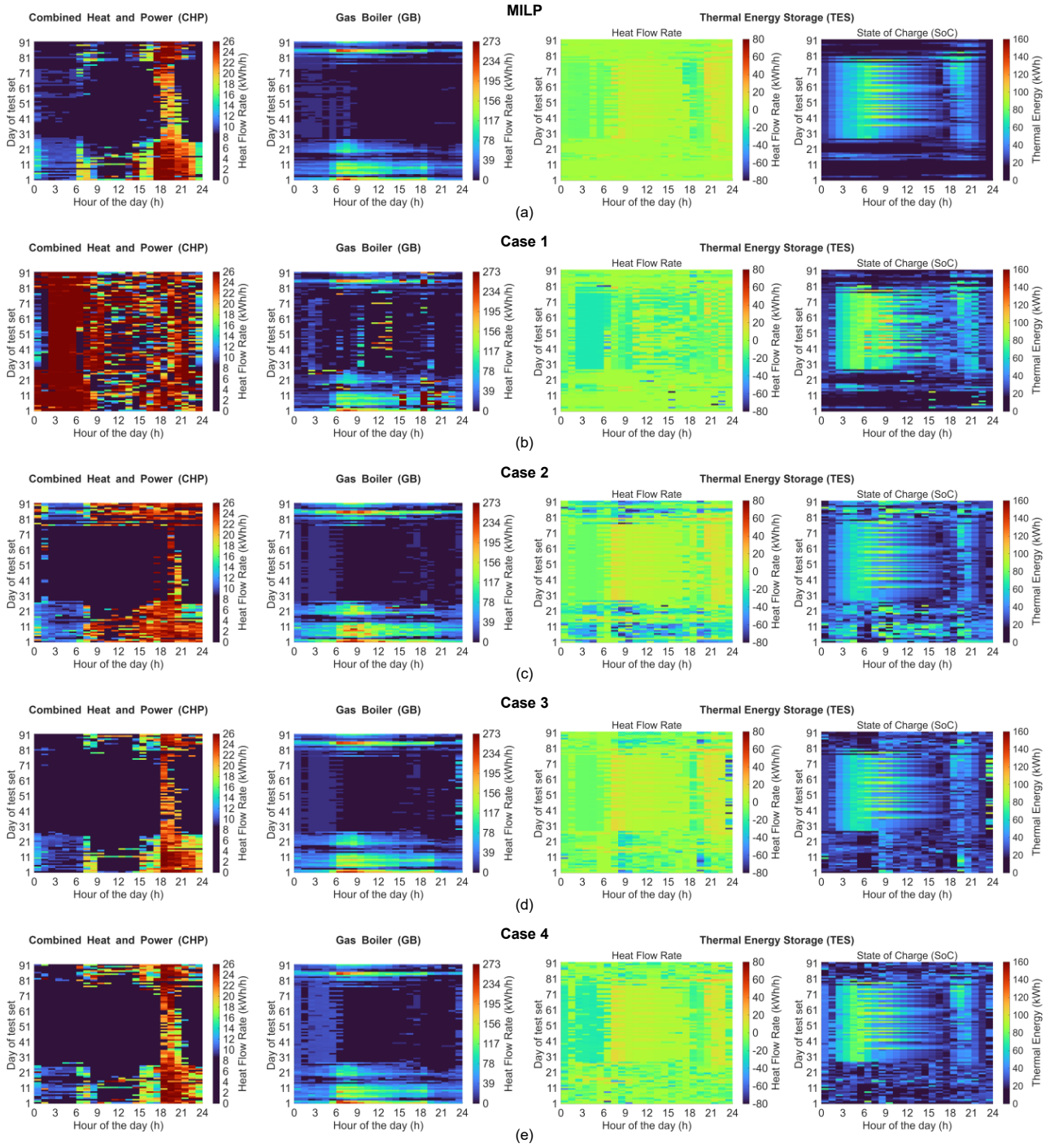

Figure 5. *Carpet plots comparison of the different outputs' schedule management carried out on the thermal side by the trained RL agents. From left to right, for CHP, GB, and TES in terms of the heat flow rate, and in addition, also the state-of-charge of TES. From top to bottom, the schedule is as follows: (a) MILP; (b) Case 1; (c) Case 2; (d) Case 3; (e) Case 4.*

Furthermore, to assess the impact of these errors on total expenditure throughout the entire test days. To this end, the mentioned Daily- and Yearly OPEX are compared against the MILP benchmark for the same test days. Additionally, Test Case 0, which follows a simple rule-based cost-minimization strategy, is included as a reference. The bar graph in Fig. 6 (b) summarizes these comparisons. As shown, Case 1, whose reward includes only the operational cost from RL, yields the highest costs, much higher than those of the rule-based approach. In contrast, Case 2, 3 and 4 achieves the best results, with only a 8–12 % cost surplus compared to MILP. This is also comparable to [7], whose "Internal Prediction" case, in which the distance between MILP and RL in their approach can reach 90 %, could be related to ours.

It appears that the key cost-driving factors are the cost-sensitive devices, CHP and GB. Their strategic decision-making optimally manages the system.

In conclusion, the most effective methodology is to set the reward to minimize cost deviations from the MILP,

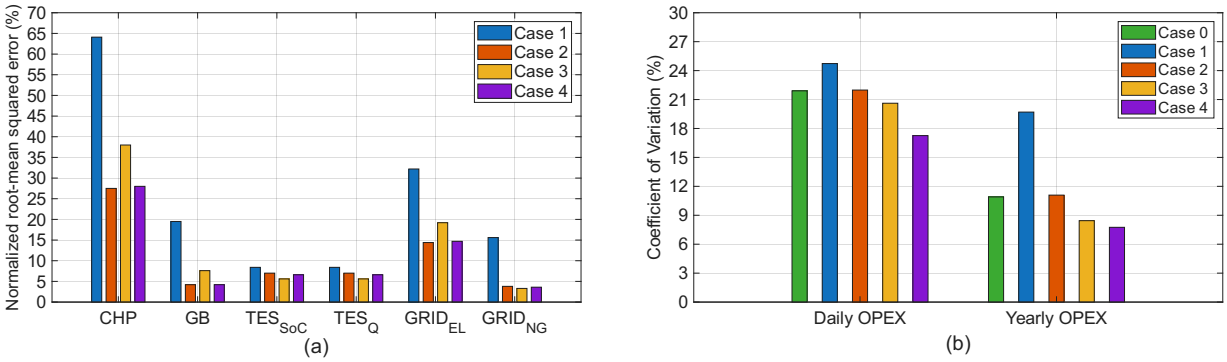

Figure 6. *Bar graphs outlining the performance of the RL approaches for each case in terms of: (a) NRMSE for device schedule relative to MILP's ones; (b) CV relative to MILP benchmark in terms of daily and yearly operating costs, including a reference Case 0 approach.*

using both daily and hourly optimal data, and to include at least the previous two time steps among the states. This heuristic strategy results in large deviations in optimal cost compared to MILP (CV~20%), while still keeping the yearly cost surplus limited to 8–12 %.

4. Conclusion

This paper defines a methodology to obtain optimal management of a MES through RL models. The developed framework includes optimal data generation and schedules for a residential MES, solving an MILP that covers one year of operation using a rolling-horizon approach. Several RL models are trained for different combinations of reward signals using only a subset of the optimization problem as boundary conditions. The trained RL models are then integrated into a heuristic framework that allows controlling the entire MES starting from the nearly optimal schedules of a few devices. Lastly, schedules based on RL models are compared to MILP ones, and a simple rule-based strategy is used as a reference.

The essential key-point results of this study show that:

- A trained RL model can achieve near-optimal MES management by replicating MILP results using only a subset of the information available at the optimization stage. Notably, forecast data on energy demand, RES production, and energy costs are unnecessary, reducing implementation complexity.
- Since the RL model can effectively replace MILP. This approach significantly reduces computational effort in the execution, as the RL model is fast, lightweight, and does not rely on specialized MILP solvers.
- The main disadvantages of this approach are the need to carefully model both the reward and the environment as model-based. This could be time-consuming, along with the training time, which raises the computational effort.
- Inserting MILP optimal data inside the reward boosts the whole performance, making it reach a cost surplus limited to 16–21 %, thus, including optimal information is beneficial to the management.
- The most effective approach is the one in which the reward is set to minimize cost deviations from the MILP, using both daily and hourly optimal data, and including the states at least the previous two timesteps. This heuristic strategy results in large deviations in optimal cost compared to MILP (CV~17 %), but overall has a limited yearly cost surplus of 8 %.

Acknowledgments

This research was funded under the Program Agreement between the Italian National Agency for New Technologies, Energy and Sustainable Economic Development (ENEA) and the Italian Ministry of Environment and Energy Security (MASE) for Electric System Research, within the framework of the 2025–2027 Implementation Plan, Project 1.5 "High-efficiency buildings for the energy transition" (CUP I53C24003330001), Work Package 4 "Promotion of building energy efficiency through increased autonomy, flexibility, and consumption awareness".

Nomenclature

Greek symbols

α	Energy purchase price ratio
γ	Discount factor
λ	Penalty weight, €/kWh
π	Policy
σ	Scaling factor
τ	Optimization horizon, h

Latin Symbols

$\mathcal{A}$	Action spaces
$\boldsymbol{a}$	Actions
C	Daily Cost, €
c	Cost, €
e	Energy, kWh
F	Fuel consumption rate, kWh/h
g	Penalty function, €
p	Prices, €/kWh
Q	Heat flow rate, kWh/h
R	Cumulative reward, €
r	Reward, €
$\mathcal{S}$	State spaces
$\boldsymbol{s}$	States
U	Energy Capacity, kWh
W	Electrical Power, kWh/h

Subscripts and Superscripts

curt	energy curtailment
dem	demand
d	day of the year
el	electric
fuel	fuel
in	energy input
load	load
min	minimum
nom	nominal/rated power
out	energy output
purc	purchased
res	residual energy demand
req	required energy
sell	sell/sold
t	*t*-th hour/timestep
th	thermal
u	*u*-th unit

References

[1] Mancarella P., MES (multi-energy systems): An overview of concepts and evaluation models. Energy 65 (2014), 1–17.

[2] Bischi A. et al., A detailed MILP optimization model for combined cooling, heat and power system operation planning. Energy 74 (2014), 12–26.

[3] Urbanucci L., Limits and potentials of Mixed Integer Linear Programming methods for optimization of polygeneration energy systems. Energy Procedia 148 (2018). ATI 2018 - 73rd Conference of the Italian Thermal Machines Engineering Association, 1199–1205.

[4] Cao D. et al., Reinforcement Learning and Its Applications in Modern Power and Energy Systems: A Review. Journal of Modern Power Systems and Clean Energy 8.6 (2020), 1029–1042.

[5] Ji Y. et al., Real-Time Energy Management of a Microgrid Using Deep Reinforcement Learning. Energies 12.12 (2019).

[6] Kim S., Lim H., Reinforcement Learning Based Energy Management Algorithm for Smart Energy Buildings. Energies 11.8 (2018).

[7] Kang D. et al., Optimal planning of hybrid energy storage systems using curtailed renewable energy through deep reinforcement learning. Energy 284 (2023), 128623.

[8] Ye Z. et al., Optimal Scheduling of Integrated Community Energy Systems Based on Twin Data Considering Equipment Efficiency Correction Models. Energies 16.3 (2023).

[9] Razafitsalama J., Frate G.F., Ferrari L., A Machine Learning Approach to Emulate Mixed-Integer Linear Programming for Optimal Management in Multi-Energy System. Proceeding of the 38th International Conference on Efficiency, Cost, Optimization, Simulation and Environmental Impact of Energy Systems (ECOS 2025). (Paris, France, June 29–July 4, 2025). 2026.

[10] ARERA, Arera: Dati e statistiche. 2025. URL: https://www.arera.it/dati-e-statistiche (visited on 01/25/2026).

[11] Gestore dei Mercati Energetici S.p.A. (GME), Mercato Elettrico - Sito Istituzionale. 2026. URL: https://www.mercatoelettrico.org (visited on 01/25/2026).

[12] Mitra S., Sun L., Grossmann I.E., Optimal scheduling of industrial combined heat and power plants under time-sensitive electricity prices. Energy 54 (2013), 194–211.

[13] Amin A., Mourshed M., Community stochastic domestic electricity forecasting. Applied Energy 355 (2024), 122342.

[14] Sutton R.S., Barto A.G., et al., Reinforcement learning: An introduction. Vol. 1. 1. MIT press Cambridge, 1998.

[15] Haarnoja T. et al., Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. 2018. arXiv: 1801.01290 [cs.LG].

[16] Hendrycks D., Gimpel K., Gaussian Error Linear Units (GELUs). 2023. arXiv: 1606.08415 [cs.LG].

[17] Inc. T.M., *Reinforcement Learning Toolbox version: 25.2 (R2025b)*. Natick, Massachusetts, United States, 2025.