Energy 203 (2020) 117769

Contents lists available at ScienceDirect

Energy

journal homepage: www.elsevier.com/locate/energy

Genetic-programming-based multi-objective optimization of strategies for home energy-management systems

Jernej Zupančič [a, b, *], Bogdan Filipič [a, b], Matjaž Gams [a, b]

[a] Jožef Stefan Institute, Jamova cesta 39, 1000, Ljubljana, Slovenia
[b] Jožef Stefan International Postgraduate School, Jamova cesta 39, 1000, Ljubljana, Slovenia

ARTICLE INFO

Article history:
Received 11 July 2019
Received in revised form
27 March 2020
Accepted 1 May 2020
Available online 7 May 2020

Keywords:
Home energy-management system (HEMS)
Genetic programming
Multi-objective optimization
Tree-based strategy
Timetable-based strategy
Multi-objective reinforcement learning (MORL)

ABSTRACT

Home energy-management systems can optimize performance either by computing the next step dynamically – online, or rely on a precomputed strategy used to introduce the next decision – offline. Further, such systems can optimize based on only one or several objectives. In this paper, the multi-objective optimization of offline strategies for home energy-management systems is addressed. Two approaches are compared: the common timetable-based versus our approach based on decision trees. The timetable-based strategy is optimized using a multi-objective genetic algorithm, while the tree-based strategy is optimized using multi-objective genetic programming. As a result, a set of rules that comprise the trees for efficient management of an energy system is generated automatically. First, the approaches are addressed theoretically, with the finding that the tree-based approach is more powerful than the timetable-based approach. Second, the performance of the tree-based approach is compared with the performance of the timetable-based approach and manually defined strategies in an experiment involving real-world data. A performance increase of up to 17% in terms of the cost objective was confirmed for the tree-based approach. This is achieved without changing the user habits, i.e., there is no need of having to adapt the appliance usage to the energy-management system.

1. Introduction

As users become more concerned about the environment, regulating authorities are increasingly restricting the consumption of non-renewable energy, while the deployment of smart grids [1] continues to increase. In addition, methods and systems for smart electrical energy management in homes, industrial facilities and office buildings are becoming ever more important.

This work is motivated by the lack of an energy-management system that:

1. Can be automatically personalized to a particular home energy system deployment.
2. Can outperform standard timetable-based energy-management systems.
3. Can take into account several conflicting objectives.
4. Is not computationally expensive and can be deployed on low-cost hardware.
5. Does not require the user to change the user habits.

Current smart-home systems, especially the ones that are commercially available, use relatively simple and predefined control mechanisms for home energy management. Even the solutions that can be personalized by learning user habits and adjusting the performance of the smart home accordingly, usually perform the optimization with respect to a single objective only, e.g., decreasing the costs.

Energy-management systems [2] for smart homes typically model the problem of energy management as a scheduling problem. First, predictive models for solar irradiation [3], wind speed [4], consumption [5] and/or prices [6], are computed and then used in the optimization of an energy-management schedule for the next time horizon. One day is the usual time horizon used. Since the re-computation is required for every time horizon, such strategies are classified as rolling time horizon strategies and sometimes referred to as “classical” strategies [7].

The result of some energy-management system optimization

* Corresponding author. Jožef Stefan Institute, Jamova cesta 39, 1000, Ljubljana, Slovenia.
E-mail addresses: jernej.zupancic@ijs.si (J. Zupančič), bogdan.filipic@ijs.si (B. Filipič), matjaz.gams@ijs.si (M. Gams).

https://doi.org/10.1016/j.energy.2020.117769

Nomenclature

$I_{standard}$	standard irradiation
$T_{standard}$	standard temperature
$Charge_i$	energy input into the battery in the i-th time interval
$Discharge_i$	energy output out of the battery in the i-th time interval
SoC_i	battery's state of charge at the end of the i-th time interval
k	correction coefficient
k_d	discharge coefficient for battery self-discharging rate
$Balance_i$	energy balance in the i-th time interval
Cost	total operation cost for running the energy-management system strategy
$Cost_i^{buy}$	buy price for the i-th time interval
$Cost_i^{sell}$	sell price for the i-th time interval
Green	total green factor for running the energy-management system strategy
$Grid_i^{in}$	amount of electrical energy sold to the grid in the i-th time interval
$Grid_i^{out}$	amount of electrical energy bought from the grid in the i-th time interval
$Loss_i$	the amount of energy lost in the i-th time interval
$PV_{max}^{receive}$	maximum electrical power that can be sent the grid
PV_i	photovoltaic module average power production in the i-th time interval
$PV_{declared}$	declared power of the photovoltaic module
TM_i	average photovoltaic module temperature in the i-th time interval
f_{max}	coefficient for limiting the total power that can be sent to the grid
I_i	total irradiation in the i-th time interval
L_i	electrical load in the i-th time interval
T_i	average temperature in the i-th time interval
W_i	average wind speed in the i-th time interval
HEMS	home energy-management system
MORL	multi-objective reinforcement learning
NSGA-II	non-dominated sorting genetic algorithm II

methods are the schedules for a set of managed appliances [8]. This impacts the user comfort, since the user has to adapt to the time of allowed appliance usage in order to meet the optimization objective. To address this problem, some approaches [9] take into account certain aspects of the user comfort, such as indoor temperature, illumination of the occupied room, electric vehicle range and preferred time window for appliance operation.

Recently, deep neural networks have been used to model the energy consumption and weather, indicating the possibility for accurate predictions when large amounts of diverse data are available. Deep neural networks with multilayer perceptron were used for short-term power probability density forecasting in Ref. [10], while recurrent neural networks were assessed for short-term building energy prediction in Ref. [11]. In order for these techniques to work well, the predictions have to be rather or even extremely accurate, which is hard to achieve in real life due to the uncertainty associated with external factors and parameters, mainly the local weather, but also the energy consumption that results from users' activities at any particular time. Furthermore, predictive models and optimization strategies are tightly coupled in such systems [12]. However, this is rarely addressed, e.g., the deficiency of predictive models is not taken into account during the optimization phase. Since the weather and the users' activities directly influence the production and consumption of energy in the home, predefined scheduling might not be an appropriate technique for optimum energy-flow management.

Some studies present energy-management systems that do not use scheduling. For example, in Ref. [13] Markov decision processes are used, and in Ref. [14] fuzzy-logic expert systems are deployed, with the claim of near-optimally managed energy flows. Fuzzy logic-based energy-management system is proposed in Ref. [15], where a rule set for energy management is generated by means of a hierarchical genetic algorithm with the aim of profit optimization. All three papers indicate the deficiencies of using schedules for energy-consumption optimization, such as increased computational costs due to frequent optimization runs, an inability to adapt to new, unexpected situations, increased computational costs due to complex prediction models such as deep neural networks, and an inability to run a high-quality energy-management program on-site using cost-efficient equipment.

While the majority of approaches address only one optimization criterion, others, such as [16], acknowledge the need to optimize the system according to multiple contradictory criteria. The reason is that often-used and practically relevant criteria, such as energy consumption, carbon emission, self-consumption, and costs, are usually conflicting in the sense that improving one objective can worsen the other. For instance, if selling the energy is economically beneficial, then increasing the energy sales to the grid lowers the operational costs and thereby enhances the profit, but at the same time reduces the self-sufficiency rate. The common techniques transform the multiple criteria into a single objective, usually applying a weighted-sum approach, and then perform single-objective optimization.

Energy storage and management system design optimization for a photovoltaic-battery energy storage system using both weighted sum approach and Pareto-based multi-objective optimization is addressed in Ref. [17]. Further, a complex rule-based energy-management strategy is proposed, indicating that designing such strategy by hand is a laborious process without guarantees on optimal performance.

Another way of transforming multiple objectives into a single objective is to define optimum points in the objective space that the strategies try to achieve. This is called steering. Steering approaches to Pareto-optimal multi-objective reinforcement learning of strategies for the control of local battery storage for a residential solar-power system are presented in Ref. [18].

The parameter-based optimization of energy-management systems according to multiple objectives is described in Ref. [19], where the daily optimization of the operational schedule using optimized timetables is performed, and in Ref. [20], where the reference points for lower-level controllers are dynamically optimized.

Some approaches [21] construct a thermal and an electrical model based on existing data and other inputs, such as energy rules, to derive an overall energy model that is then used to predict electrical and thermal demand and production. The optimization that takes into account the overall energy model is then performed in order to generate suggestions for additional energy rules that can be applied to the energy-management system by the building managers.

However, to the best of our knowledge, no system presented in the related work can perform a robust, well-performing offline

strategy optimization for an energy-management system and provide multiple trade-off solutions in configurations with conflicting objectives, which is the case in the presented approach. The contributions of this work are as follows.

1. The proposed tree-based solutions are computationally less expensive than some of the online approaches, which require recomputation for each next time period. The computationally intensive step for the proposed tree-based strategies can be performed only a few times in a year and can be executed off-site, preferably in a cloud. Only the tree-based solutions can then be transferred on-site. The solutions can then be easily implemented on a home energy-management system hardware, since they comprise only simple arithmetic operations and if-then rules.
2. The proposed tree-based strategy outperforms other often-used strategies, those based on timetables and manually defined strategies, by up to 17% in terms of the cost objective while keeping the green objective fixed, as evident from the experimental results.
3. The superiority of the tree-based strategies over the timetable based strategies with respect to the expressive power is proven theoretically.
4. The proposed approach is based on the true Pareto-based multi-objective optimization, where the user can pick the solution with the preferred trade-off after the trade-offs are clearly presented to him or her. This is an advantage over the weight-based multi-objective approaches where the weights are usually chosen beforehand, when the trade-offs are not yet evident.
5. The user of the proposed approach does not have to change his or her habits and can use any appliance at any time. This is in contrast with some of the methods that prescribe time intervals for appliance usage, which have to be considered by the user in order to achieve the optimal criteria values.

The advantage of the presented approach, based on a tree-based strategy (an example is provided in Fig. 1), over a timetable-based strategy (an example is provided in Fig. 2) is presented in the following example. Assume that the price of electrical energy changes dynamically throughout the day. This is known as real-time pricing and is already available in certain parts of the world [22]. Assume that, generally, the price for electrical energy starts increasing in the morning. Since this is a typical behavior, the cost-effective, robust, timetable-based strategy would learn to sell the surplus energy in the morning instead of storing it in the battery for later use. Now assume that on a particular morning it is very sunny and windy. In this case, the solar and wind farms produce a large amount of electrical energy, which becomes available on the energy market, thereby lowering the price of electrical energy. A robust, timetable-based strategy would continue to sell the energy at a low price, since it only takes into account the time of the day, and selling the energy is beneficial on a typical day. A tree-based strategy, however, also takes into account the low (or even negative) price. At times of an exceptionally low price, the selling of energy is not beneficial and an optimized, tree-based strategy, which would learn this, would deffer the selling of energy to a later time. The advantage of the tree-based strategy is three-fold. First, the energy stored in the battery could be used by a smart home, thereby increasing its independence from the grid. Second, the cost would be lower, since the profit made from selling the energy at a low price is surpassed by the cost of buying more expensive energy at a later time. Third and overall, the tree-based strategy is more flexible and enables adaptation to the current situation based on the previous construction of potential decisions needed in most relevant situations.

The rest of the paper is structured as follows. In Section 2, the energy-management system optimization problem is introduced, and in Section 3, energy-management strategies are discussed. In Section 4, the presented framework is described. In Section 5, the experiments and results are presented, while Section 6 discusses the case-study results and Section 7 concludes the paper.

2. Problem formulation

The problem of managing electrical energy in a smart home that has one or more sources of electrical energy, an electrical energy storage option, a smart grid, and an electrical energy consumption or load is addressed. The proposed Home Energy-Management System (HEMS) does not manage the devices by turning them on or off or by setting different modes of operation.

The HEMS problem is, therefore, to find one or a set of the best (according to one or multiple objectives) strategies that decide on how much energy to buy from or sell to the grid or how much energy to store in the battery based on the past, present and predicted-future states (regarding the price of electrical energy, the production and consumption of electrical energy, and the state of charge of the battery).

The overall schematic that illustrates the problem of managing the electrical energy in a smart home is presented in Fig. 3. The historical data is first retrieved and used as an input for the optimization procedure. The result of optimization is a set of near-optimal strategies with respect to multiple criteria, e.g., *green* and *cost* criteria. The user or the HEMS operator then chooses a solution strategy that is preferred by the user, which is then uploaded to the HEMS central unit and utilized to manage the energy flows within the system. Additional data is recorded and can be reused in the event of another optimization run.

2.1. Model

The home energy system (Fig. 4) comprises components that are only energy sources (photovoltaic modules and wind generators), only energy consumers (load), or both (grid, battery). The HEMS is responsible for managing the energy flows between the home energy system's components. For the purpose of this paper, all the hardware details are abstracted away, and only the logic of the control of the energy flow is addressed.

Photovoltaics. A regression model [23] is used to determine the energy generation for the given weather conditions. Given the declared power of the photovoltaic modules ($PV_{declared}$), the total irradiation (I_i), where subscript i denotes the i-th time interval, the average wind speed (W_i) and the average temperature (T_i), the power output (PV_i) can be approximated as follows:

$$TM_i = 0.943 \cdot T_i + 0.0195 \cdot I_i - 1.528 \cdot W_i + 0.3529 \quad (1)$$

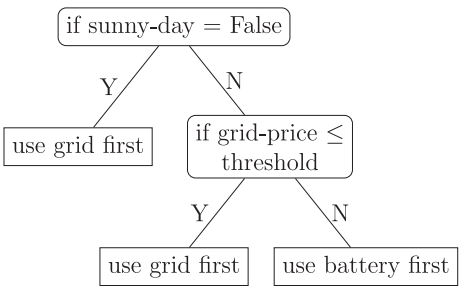

Fig. 1. An example tree-based strategy.

00:00–08:45	use grid first
08:45–11:00	use battery first
11:00–12:30	use grid first
12:30–15:00	use grid first
15:00–22:00	use battery first
22:00–24:00	use grid first

Fig. 2. An example timetable-based strategy.

$$\mathrm{PV}_i = \mathrm{PV}_{declared} \cdot \frac{I_i}{I_{\mathrm{standard}}} \cdot (1 + k \cdot (\mathrm{TM}_i - T_{\mathrm{standard}})), \tag{2}$$

where TM_i is the average photovoltaic module temperature, $I_{standard} = 1000\ \mathrm{W/m^2}$ is the standard irradiance, $T_{\mathrm{standard}} = 25\,^{\circ}\mathrm{C}$ is the standard temperature, and $k = -0.004$ is the correction coefficient.

Battery. The battery's state of charge at the end of the i-th time interval (SoC_i) is calculated from the previous state of charge (SoC_{i-1}), the battery charge (Charge_i), the discharge ($\mathrm{Discharge}_i$), and the self-discharging rate k_d:

$$\mathrm{SoC}_i = k_d \cdot \mathrm{SoC}_{i-1} + \mathrm{Charge}_i - \mathrm{Discharge}_i. \tag{3}$$

During each time interval, the battery can either charge or discharge:

$$\forall i; \mathrm{Charge}_i \cdot \mathrm{Discharge}_i = 0. \tag{4}$$

Further, that energy cannot be transferred between the grid and the battery.

The smart grid sends the price signals to the HEMS. Buy and sell prices are denoted with $\mathrm{Cost}_i^{\mathrm{buy}}$ and $\mathrm{Cost}_i^{\mathrm{sell}}$, respectively. During each time interval, the HEMS sends $\mathrm{Grid}_i^{\mathrm{in}}$ to or receives $\mathrm{Grid}_i^{\mathrm{out}}$ electrical energy from the grid:

$$\forall i; \mathrm{Grid}_i^{\mathrm{in}} \cdot \mathrm{Grid}_i^{\mathrm{out}} = 0. \tag{5}$$

Sending the energy to the grid can be limited by coefficient $f_{\max} \in [0, 1]$, determined by the national law:

$$\forall i; \mathrm{Grid}_i^{\mathrm{out}} \le \mathrm{PV}_{\max}^{\mathrm{receive}} = f_{\max} \cdot \mathrm{PV}_{\mathrm{declared}}. \tag{6}$$

Electrical load. The energy consumption of all the electrical devices in a residential home during the i-th time interval is aggregated into the electrical load (L_i). The load management [24] is not covered in this paper.

2.2. Objectives

Often-used objectives in HEMS include running costs, CO_2 emission, self-consumption rate, maximum peak load, total energy consumption and battery life expectancy. The objectives considered in this paper are the running costs and the green factor.

The **running costs** (Cost) comprise the cost of buying and the profit of selling the electrical energy during each time interval:

$$\mathrm{Cost} = \sum_{i=0}^{n} \left(\mathrm{Grid}_i^{\mathrm{out}} \cdot \mathrm{Cost}_i^{\mathrm{buy}} - \mathrm{Grid}_i^{\mathrm{in}} \cdot \mathrm{Cost}_i^{\mathrm{sell}} \right), \tag{7}$$

where $\mathrm{Grid}_i^{\mathrm{out}}$ and $\mathrm{Grid}_i^{\mathrm{in}}$ denote the amount of electric energy bought from or sold to the system, respectively. Further, $\mathrm{Cost}_i^{\mathrm{buy}}$ and $\mathrm{Cost}_i^{\mathrm{sell}}$ are the prices for buying and selling the electric energy to or from the system.

The **green factor** (Green) represents the level of independence of a home with the HEMS from the grid:

$$\mathrm{Green} = \frac{\sum_{i=0}^{n} \left(\mathrm{PV}_i - \mathrm{Loss}_i - \mathrm{Grid}_i^{\mathrm{in}} \right)}{\sum_{i=0}^{n} L_i}, \tag{8}$$

where subscript i indicates i-th time interval, PV_i is the amount of energy produced, Loss_i is the amount of energy that is not sold or used in the HEMS – effectively being lost, $\mathrm{Grid}_i^{\mathrm{in}}$is the amount of energy sold to the grid, and L_i is the electrical energy load.

2.3. Simulator

The simulator models the energy flows in the home energy

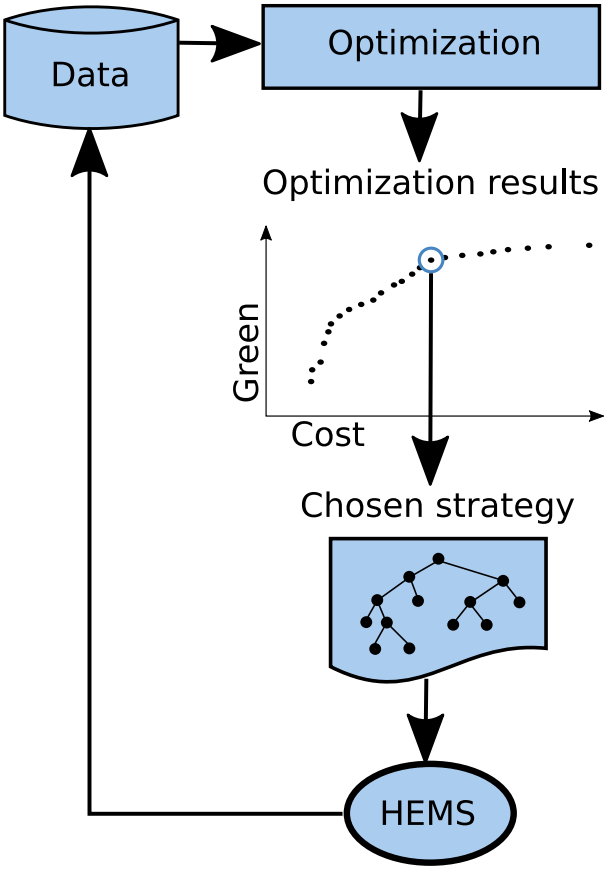

Fig. 3. The flow for managing the electrical energy in a smart home.

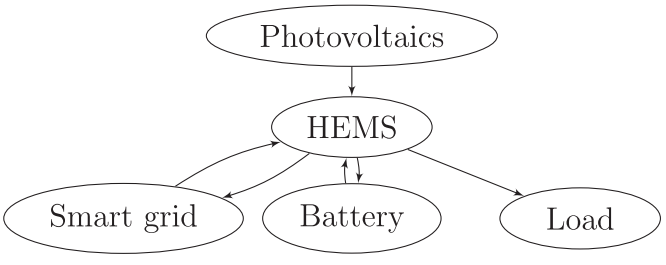

Fig. 4. Home energy-system model.

system in discrete time intervals with respect to the given energy-management strategy and is used to evaluate the performance of the strategy according to the specified objectives. The following simulator components are considered: photovoltaic module, smart grid, battery, load, and HEMS. For each time interval, the simulator receives the following input data:

1. Grid energy prices.
2. Energy production.
3. Load.

Additionally, the home energy system's configuration consists of:

1. Battery: maximum charge rate, maximum discharge rate, minimum state of charge, maximum state of charge, initial state of charge, self-discharge rate, charge efficiency, discharge efficiency.
2. Photovoltaics: peak power (the maximum power of a photovoltaic module in a standardized test).
3. Grid: efficiency of selling energy to the grid, peak power sell coefficient (specifying the maximum power allowed for transmitting the electrical energy from the home energy system to the grid).

The remaining energy balance (Balance_i) in each time interval is denoted with:

$$\text{Balance}_i = \text{PV}_i - L_i, \tag{9}$$

where PV_i is the electrical energy production and L_i is the electrical energy load. There are two options during each time interval: either the HEMS first uses the battery in order to store/obtain energy or it uses the grid in order to sell/buy the energy. If the balance is positive, there is energy excess, and in the case of a negative balance, there is a lack of energy.

In each time interval the HEMS executes a control action and receives the values of the objectives (cost and green factor).

3. Strategies for home energy-management systems

Two types of controllers or strategies were developed and tested for the purpose of this study: the timetable-based strategy and the tree-based strategy. The HEMS strategy maps the current HEMS state into the control action. In this paper, both strategies use the same control actions:

1. α_0 − use battery first:
   (a) If the balance < 0: Try to use the energy from the battery first. If this is not sufficient, supply energy from the grid.
   (b) If the balance ≥ 0: Try to put the extra energy into the battery first. If the production is greater than the maximum charge rate, sell the energy to the grid up to the grid sell limit and discard the rest.
2. α_1 − use grid first:
   (a) If the balance < 0: Get all the required energy from the grid.
   (b) If the balance ≥ 0: Try to sell the energy to the grid up to the sell limit. If there is energy left, charge the battery. If there is still some energy left, discard it.

Although the control actions are the same, the HEMS strategies differ in how the automatic decisions to choose one particular control action are taken, i.e., they use different mappings from the HEMS state space into the control action state space.

Definition 1. (HEMS strategy). *The HEMS strategy φ is a mapping from the HEMS state space Σ to the action space A.*

Definition 2. (Optimum HEMS strategy). *The optimum HEMS strategy is defined as:*

$$\varphi^* = \text{argmin}_{\varphi \in \Phi} f_D(\varphi), \tag{10}$$

where f is an objective function that depends on the parameters D, and $\Phi = \{\varphi; \Sigma \rightarrow {}^{\varphi}A\}$ is a set of all the mappings from Σ to A.

For the purpose of this study, f is a simulator that computes the running costs and the green factor as two conflicting objectives.

3.1. Timetable strategy

In the timetable strategy, the action is specified for each time interval. Using only the time-based decision making can be efficient in domains where the working conditions are periodic, e.g., a user goes to work at a specific hour on a workday, which is partially the case in the energy-management domain, and not much other data is available. Because of its simplicity it is often used in control-strategy problems.

Often, the timetable strategy is used in combination with some prediction mechanism trained on a training dataset. Next, for each time interval, where the HEMS states are partially predicted using the previously built predictive models, the problem of finding the optimum timetable(s) is solved. This kind of strategy requires continuously solving the optimization problem. Moreover, the quality of the strategy depends on the accuracy of the predictive models.

In this paper, the focus is on finding robust timetable strategies that do not require resource-intensive continuous recomputation for each new time horizon (i.e., a set of consecutive time intervals). This requires finding timetable strategies that perform well on the training data, given some longer time horizon, i.e., the optimization problem is solved once and the solution is then used by the HEMS for new data.

Definition 3. (The timetable strategy). *Timetable strategy τ_s is a mapping from time space T to action space A.*

Note that the HEMS state space $\Sigma = T \times F$, where the time space T is only one dimension, while F includes all the other feature dimensions (price, energy consumption, energy production and others).

For the purpose of this study, the timetable strategy's mapping is discretized in order to correspond to the discretized dataset D, i.e., for each possible value of the time of the day an action is specified.

3.2. Tree-based strategy

In the tree-based strategy, the decision about which action to take is based on a logic flow as defined by a binary decision tree. An example of such a decision tree is shown in Fig. 5. The inner nodes represent tests of the form: is the value of the feature X (e.g., the current balance) greater than or equal to the value v (e.g., 0). Based on the result, the control logic proceeds along either of the two branches. The terminal nodes, i.e., the leaves, represent actions to be taken (e.g., use the battery first). This decision-making flow is repeated each time a decision needs to be made.

In this case, the following features are used: the current minute of the day, the average previous day's buy price, the average previous day's sell price, the current load, the current production, the current buy price, and the current sell price.

The test value v for the feature X is calculated dynamically, based

on the possible values that can be achieved, given the decision flow. Each inner node is represented by the feature X and the relative test value $v_{\text{relative}} \in (0, 1)$. If the relative test value does not yet exist (because the test node has not yet been visited in the simulation run) it is randomly generated and from this value the absolute test value (v_{absolute}) is calculated as

$$v_{\text{absolute}} = v^X_{\text{min}} + v_{\text{relative}} \cdot \left(v^X_{\text{max}} - v^X_{\text{min}} \right), \tag{11}$$

where v^X_{min} and v^X_{max} are the current minimum and maximum values that can be attained in the current node for the feature X. When the feature X is first used in a test, v^X_{min} and v^X_{max} are the minimum and maximum values, respectively, for the feature X in the data. Assume that v^X_i is the value for the feature X in the i-th time interval. At the root node the following values are set:

$$v^X_{\text{min}} = \min \left\{ v^X_i \right\}_{i=0}^{n}, \tag{12}$$

$$v^X_{\text{max}} = \max \left\{ v^X_i \right\}_{i=0}^{n}. \tag{13}$$

However, when X has already been used in a test in some predecessor node, then v^X_{min} and $v^X_{\text{max}}$are adjusted as follows: if $v^X_i \leq v_{absolute}$, then $v^X_{\text{max}} = v_{absolute}$, otherwise $v^X_{\text{min}} = v_{absolute}$.

Definition 4. (Tree-based strategy). *A tree-based strategy ψ is a mapping from Σ to A, where ψ is in the form of a binary decision tree.*

3.3. Comparison of strategies for timetable-based and tree-based expressive powers

Definition 5. *The expressive power is the size of the strategy mapping $\varphi(\Sigma)$.*

Theorem 1. *The expressive power of the tree-based strategies Ψ is greater than the expressive power of the timetable-based strategies T: $T \subsetneq \Psi$.*

Proof. *First, let us prove that each timetable strategy can be represented by some decision-tree strategy. Without loss of generality, assume a timetable strategy τ_s with daily periodicity represented by the following pairs*

$$\tau_s = [(t_0, a_0), (t_1, a_1), \ldots, (t_{n-1}, a_{n-1}), (t_n, a_n)]. \tag{14}$$

Here, each pair (t_j, a_j) means that in the time interval $(t_{j-1}, t_j]$ action a_j is used, where t_{-1} is defined as $t_{-1} = t_n =$ midnight. *For the timetable strategy τ_s a corresponding tree-based strategy can be constructed that behaves exactly as shown in Fig. 6. This proves that a set of timetable strategies T is included within the set of tree-based strategies Ψ, therefore,*

$$\forall \tau \in T \; \exists \psi \in Psi : \tau \sim \psi \Rightarrow T \subseteq \Psi. \tag{15}$$

Second, let us prove that a tree-based strategy exists that does not have a timetable-based strategy representation. First, a feature that is not exactly repeated after a certain time is needed, which effectively means any feature with the exception of the time-of-day feature. An example tree-based strategy as in Fig. 5 for instance uses the current balance feature and the difference between the current sell price and the average previous day's sell price. The example tree-based strategy cannot be represented using any timetable-based strategy, since neither of the features used by the tree-based strategy are periodic. Therefore,

$$\exists \psi \in \Psi \wedge \nexists \tau \in T : \psi \sim \tau \Rightarrow \Psi \not\subseteq T. \tag{16}$$

Combining the two results proves the theorem:

$$T \subseteq \Psi \wedge \Psi \not\subseteq T \Rightarrow T \subsetneq \Psi. \tag{17}$$

Theorem 1 can also be demonstrated experimentally, as seen in Fig. 7. To demonstrate the different expressive powers, 500,000 random timetable-based and tree-based strategies were generated and their performance with respect to the two objectives (cost and green factor) was compared on two different datasets. The mapping of the timetable-based strategies is observed to be included in the mapping of the tree-based strategies.

4. Optimization framework

In order to solve the optimization problem from Section 2, two alternative methods are utilized: one for the timetable strategy optimization and one for the tree-based strategy optimization. For solving this problem the methods from the field of evolutionary computation are used. The methods originate from the idea of evolution, i.e., solution candidates are put into an artificial environment, where they can interact with each other and produce offspring. The survival of the solution candidates depends on their fitness, derived from the objective function. Over several generations (i.e., iterations) the fittest solution candidates emerge as the solutions to the problem. When there are multiple conflicting objectives to be considered for optimization, a single best solution does not exist. Instead, a set of incomparable solutions (i.e., a Pareto set) are optimum in such cases, i.e., considering two solutions from the Pareto set, one is always better than the other for at least one objective. Population-based evolutionary algorithms, such as the

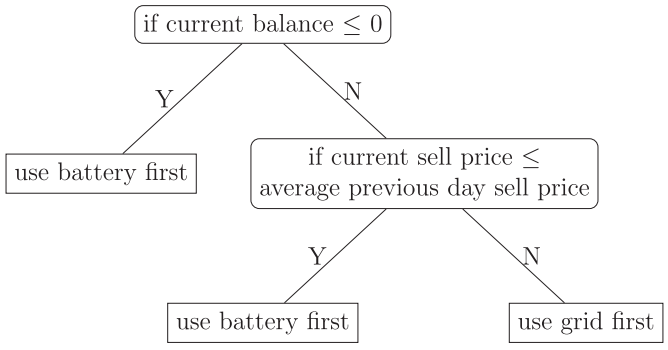

Fig. 5. An example tree-based strategy defined by a binary decision tree.

```
if current_time_of_day ≤ t0:
    use_action(a0)
else:
    if current_time_of_day ≤ t1:
        use_action(a1)
    else:
        ...
            if current_time_of_day ≤ tn−1:
                use_action(an−1)
            else:
                if current_time_of_day ≤ tn:
                    use_action(an)
```

Fig. 6. *A decision tree corresponding to the timetable τ_s*

well-known algorithm NSGA-II [25], are especially suitable for these problems, since they evolve a set of solution candidates in a single run.

Here, NSGA-II handles the optimization of the timetable strategy. Individuals are represented as a hash map, with mappings from the time-of-day to a set of actions. For the purpose of this study, a day was divided into half-hour intervals and the actions set comprised two actions: α_0, where the battery is used first, and α_1, where the grid is used first. This results in a timetable representation in the form of a list of 48 binary values, where for each half-hour interval either action α_0 or α_1 is taken.

The multi-objective genetic programming algorithm [26] is applied to the optimization of the tree-based strategy. Individuals are represented as binary decision trees, where each inner node (decision node) tests whether a feature satisfies a condition or not. Terminal or leaf nodes denote the actions in the simulator. The following genetic programming operators are applied to evolve the solutions:

Initialization: ramped half-and-half method (half of the initial decision trees are full trees, the other half are of varying depth).

Selection: a non-dominated sorting-based selection as in NSGA-II [25] is used for the purpose of selecting individuals based on the values of multiple criteria.

Mutation: uniform mutation, which randomly selects a sub-tree in the existing tree and replaces it by a randomly generated sub-tree.

Crossover: one point crossover. It randomly selects sub-trees in each of two parent individuals and exchanges them.

Reproduction: this operator returns a copy of the parent individual.

Bloat control: common problem of increasingly larger programs in later generations of the evolutionary algorithm run is addressed by the static limit as proposed by Ref. [26]. If a node is positioned at the depth specified by the static limit and is at the same time the root of a sub-tree, it is replaced by a random leaf node from its sub-tree.

The overview of the algorithm is presented in Fig. 8.

The values of the parameters of the inner nodes are generated randomly when a tree is initialized.

5. Case study

In this section, the data used for the simulation and the strategy optimization are described. Additionally, the setup and the experimental results are presented.

5.1. Data

Real-life data is used in the experimental evaluation of the strategy optimization. Weather data was obtained from the Slovenian Environment Agency weather portal [27] for the location of "Bilje pri Novi Gorici, Slovenia" for the period between January 1, 2007, and October 23, 2016. Energy consumption data is used from the available Electricity Load Diagrams 2011–2014 dataset [28]. Load data with the id "MT_003" is used for these experiments. The electrical energy pricing data was obtained from the historical intra-day continuous pricing data for the German market at the Epexspot portal [29]. The weighted average price is used for both – buy and sell prices.

All the data was resampled to 30-min intervals and the missing values were filled using the forward-fill method, i.e., each missing value is set to a value of the first existing, previously valid value. Sub-sample of the data is visualized in Fig. 9.

In order to properly test the strategies, the data was divided into two datasets – the training dataset that included the data from April 1, 2014, to June 30, 2014, and the test dataset that included the data from July 1, 2014, to September 30, 2014. Both datasets cover a period of 3 months of summer and transition periods. The training dataset is used during the process of optimization. In order to prevent overfitting (the over-adaptation of strategies to the already seen data), the strategies are also tested on a previously unseen test dataset, which gives a sense of the strategy generalization.

5.2. Alternative strategies

In order to obtain a sense of the strategy optimization performance, the results are compared to three strategy groups: manually defined strategies, robust timetable strategies, and near-optimum strategies.

The first group includes a set of manually defined strategies,

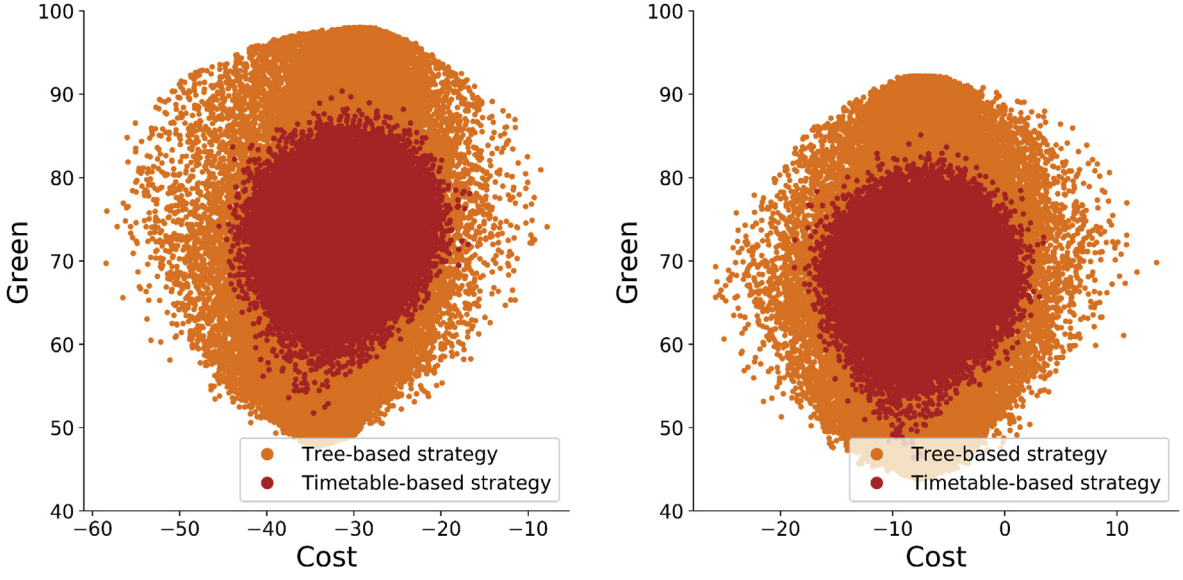

Fig. 7. *Comparing the expressive powers of the timetable-based and tree-based strategies on two different datasets.*

```
population = initialize_population(N)
evaluate(population)
operators = [mutation, crossover, reproduction]
for g in range(ngen):
    offspring = []
    for individual in population:
        operator = pick_operator(
                        operators, cxpb, mutpb
                    )
        new_individual = operator(individual)
        offspring.append(new_individual)
    evaluate(offspring)
    population = select(population + offspring, N)
```

Fig. 8. Multi-objective genetic programming algorithm for the optimization of the HEMS strategy.

normally used by the HEMS manufacturers and found in the literature [13]:

1. Battery-first strategy: in each time interval use the battery-first action $-\alpha_0$,
2. Grid-first strategy: in each time interval use the grid-first action $-\alpha_1$,
3. Opportunist sell strategy: if the current sell price is higher than the average previous day's sell price and there is an excess of energy, sell the energy to the grid (i.e., use the grid-first action α_1), otherwise utilize the battery (i.e., use the battery-first action α_0).

The second group includes the robust timetable strategies described in Section 3.1 and obtained using the NSGA-II algorithm, as described in Section 4.

The third group includes the near-optimum strategies used for the purpose of assessing the performance of the strategy optimization method. In order to obtain the near-optimum strategy, a multi-objective optimization using NSGA-II is performed in order to find a binary vector of length 4368 and 4416, which defines an action for each possible time interval in the training and test datasets, respectively. This requires solving two optimization problems, one for the training and one for the test dataset.

5.3. Optimization runs

The optimization was performed using a workstation with an Intel Xeon CPU E5-1620 v4 @3.5 GHz with 32 GB of RAM. Python [30] was used as the programming language and the DEAP library [31] was used as the optimization framework. The EMS simulator [32] was implemented using the NumPy [33] and Pandas [34] libraries. The hypervolume indicator [35], which measures the area covered by non-dominated solutions, was used to monitor the evolution of the strategies and compare the optimization runs.

Due to the resource-intensive experiment, the parameter settings were chosen based on a smaller set of preliminary runs. As a result, the parameter settings for the timetable-based and tree-based strategy optimizations specified in Tables 1 and 2, respectively, were set.

For the simulator, the parameter settings reported in Table 3 were used.

Each optimization was executed 40 times in order to obtain representative results, with the exception of the near-optimum strategy optimization, which was run only once. The parameter settings used in the near-optimum strategy optimization were the same as in the timetable-based strategy, with the exception of the number of generations, which was increased to 40,000. The increase was needed for the hypervolume value to stabilize.

5.4. Results

The hypervolume progress plots for the optimization of the timetable-based, tree-based and near-optimum strategies on the training data are presented in Fig. 10. The progress of each optimization approach is shown, i.e., 40 runs of the timetable-based strategy and tree-based strategy optimizations, and one run of the near-optimum strategy optimization. The reference point used for the hypervolume calculation was determined from the data. First, the worst values for each of the objectives were calculated and then multiplied by 1.1, if the value was positive, or 0.9, if the value was negative. The same reference point was used for all the hypervolume calculations, so that the hypervolume values are comparable. The number of generations in the timetable-based strategy and tree-based strategy optimizations was set to 1000, while for the near-optimum strategy it was increased to 40,000, keeping the number of individuals the same. Therefore, the normalized generation or evolution progress (i.e., the current iteration divided by the total number of iterations) is used for the horizontal axis.

The hypervolume indicator stabilizes in late generations; therefore, increasing the number of generations is not necessary.

The percentile values of the last generation for the optimization of the timetable-based, tree-based and near-optimum strategies on training data are presented in Table 4. The percentile values for near-optimum strategies remain the same, because only one run was executed.

The performance of the non-dominated strategies in the final populations of all the optimization runs together with the performance of the predefined strategies is shown in Fig. 11. Each found solution is represented by a dot with the values of the Cost objective on the horizontal axis and 100 $\times$ Green objective value on the vertical axis. In order to observe the generalizability of the solutions, the performance of each strategy was also evaluated on the previously unseen test data (the right-hand plot).

Observe that the negative Cost value actually means profit and solutions to the far left-hand side are preferred. At the same time, the solutions with the highest Green value at the top are preferred.

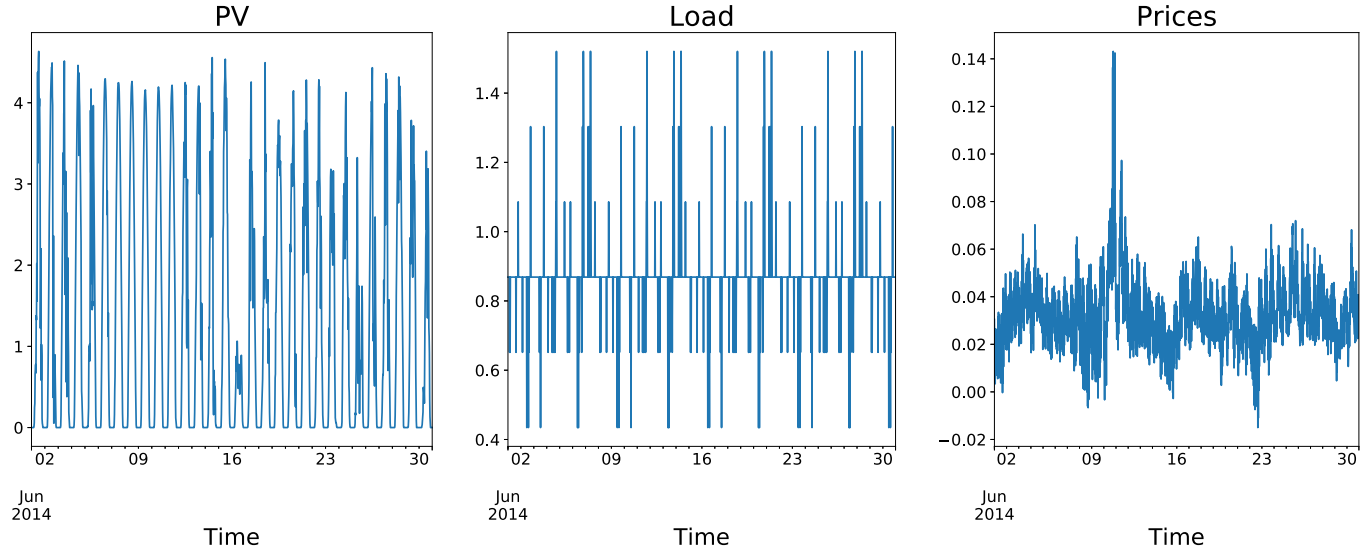

Fig. 9. Sub-sample of the data from April 2014: electrical energy production in kWh (left), electrical energy consumption in kWh (center), and electrical energy prices in EUR/kWh (right).

Table 1
Parameter settings for the timetable-based strategy optimization.

Parameter name	Parameter value
Number of parents	500
Number of offspring	500
Number of generations	1000
Gene mutation probability	1/24
Crossover probability	0.9

Table 2
Parameter settings for the tree-based strategy optimization.

Parameter name	Parameter value
Number of parents	500
Number of offspring	500
Number of generations	1000
Individual mutation probability	0.1
Crossover probability	0.9
Initial minimum tree depth	1
Initial maximum tree depth	10
Mutation minimum tree depth	0
Mutation maximum tree depth	3
Static limit for the maximum tree depth	10

Table 3
Parameter settings for the simulator.

Parameter name	Parameter value
$PV_{declared}$	10.0
$Charge_{max}$	12.5
$Discharge_{max}$	12.5
SoC_{min}	1.0
SoC_{max}	50.0
SoC_0	1.0
Battery self-discharge rate	0.0
Battery charge efficiency	1.0
Battery discharge efficiency	1.0
f_{max}	0.7
Selling energy to grid efficiency	1.0

The ideal solution would, therefore, be present in the upper-left corner. As a general rule in such cases, solutions that are to the upper left-hand side are considered the best. Furthermore, observe that nearly every two points on the front that belongs to the same strategy type are incomparable. When a decision maker chooses one over the other, the performance of the HEMS improves according to one objective and worsens according to the other. This enables the decision maker to select a trade-off between the objectives according to his or her preferences. The trade-off is presented clearly in the case of the true multi-objective optimization,

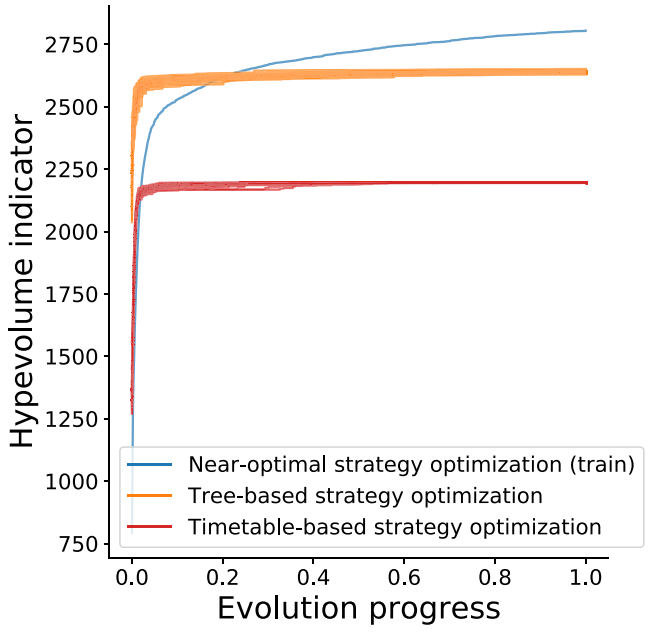

Fig. 10. Evolution of the hypervolume metric using NSGA-II for the multi-objective timetable-based strategy optimization, the proposed algorithm for the multi-objective tree-based strategy optimization, and the near-optimum strategy optimization.

which is not always possible when other methods are used.

Examples of generated tree-based strategies are presented in Figs. 12–14, where the trees correspond to strategies with the best cost criterion, median cost criterion and best green criterion, respectively, for one of the optimization runs. Similarly, examples of generated timetable-based strategies are presented in Figs. 15–17, where the timetables correspond to strategies with the best cost criterion, median cost criterion and best green criterion, respectively, for one of the optimization runs.

The performance of the chosen examples is further compared in Table 5. In addition, the improvement of the tree-based strategy over timetable-based strategy is presented in the *Improvement* column of Table 5.

6. Discussion

In this section, the differences between the timetable-based, tree-based and manually defined approaches are discussed. Every tree-based strategy found using the proposed method either dominates or is not comparable with any robust timetable-based strategy (Fig. 11). When observed as a whole, the found tree-based strategies dominate the found timetable-based strategies and the predefined strategies. This domination is especially evident for the lower cost (i.e., higher profit) solutions (Fig. 11). In our experiments, a 17.1% increase in profit can be observed (Table 5), when comparing two particular solutions of two random optimization runs that obtain the best cost objective values in their respective runs. The green objective of the tree-based approach is lower in that case, however, this is due to the fact that much better solution regarding the cost objective was found. If a solution with a better green objective is preferred, the decision maker may choose another one. The non-dominated greenest solutions are comparable regarding the green objective, however, the cost objective of the tree-based strategies is better. In the case of the presented examples, a 10.0% improvement can be observed (Table 5). For trade-off solutions, i.e., those that do not obtain maximum or minimum values of the objectives, the tree-based strategies that perform similarly regarding the green objective outperform the timetable-based strategies in terms of the cost objective, and the tree-based strategies that perform similarly regarding the cost objective outperform the timetable-based strategies in terms of the green objective (Fig. 11). This is the consequence of tree-based solutions dominating the timetable-based solutions. In the case of the presented examples, where the solutions with the median cost objective values in their respective runs were inspected, the cost objective values are similar, while a 5.8% improvement can be observed in the green objective.

A similar conclusion can be drawn for the test data set, where the relations between the strategy types remain the same, although the scale (i.e., the range of objective values achieved) changes. This indicates good generalizability of the proposed method and the robust timetable-based strategy optimization.

The timetable-based and tree-based strategies dominate the predefined strategies used in a commercially available HEMS, with the exception of the “use-battery-first” strategy, which achieves a very good value according to the green-factor criterion.

According to the hypervolume indicator, the timetable-based strategy optimization and tree-based strategy optimization converge at approximately the same rate, while the proposed approach finds significantly better-performing strategies (Fig. 10 and Table 4). The median hypervolume indicator value of the tree-based strategy optimization runs was about 20% higher than the median hypervolume indicator value of the timetable-based strategy optimization runs, without the overlap between the maximum and minimum values. The indicator values of the near-optimum strategy are the highest, as expected (Table 4). It is important to take into account that these values are not achievable in real-life. Since theoretical Pareto front is not available, it is approximated using optimization, where decision for each time interval is sought for the whole operation period with data availability. To achieve the Pareto front approximation, the optimization budget was greatly increased for the near-optimal strategy optimization, 40,000 vs. 1000 generations, while keeping all the other population parameters the same.

On the other hand, there is still room for improvement of the tree-based strategy optimization, as is evident from the difference in the performance of the found near-optimum strategies and tree-based strategies. This is probably because of too few features being used in the tree-based strategies. Since for the described approach the features have to be designed by hand, some important features that could increase the performance are probably missing. An automatic feature-generation method could increase the method’s performance; however, at the cost of strategy explainability.

Some rolling time horizon strategies may perform better, however, at the cost of increased online computational complexity. Note that in the presented approaches, the strategies are pre-computed based on simulation and historical data. In order for the HEMS to utilize a one particular chosen strategy, that strategy has to be uploaded onto the HEMS controller, while the strategies could be computed only once per several months. Afterward, only simple arithmetic and *if-then-else* rules with a low number of total operations are required for the controller execution of the chosen strategy. This is in contrast with the rolling time horizon strategies that usually perform expensive optimization every day or even every few hours.

True multi-objective optimization enables the user to choose the trade-off solution strategy following post-hoc analysis of all the provided trade-offs. The schematic of the decision making is presented in Fig. 3, while the actual trade-offs can be observed in Fig. 11. Some optimization approaches require the user to choose weights in advance, even before the optimization is executed. Choosing the weights, however, can be non-intuitive for a user, since the weights mix different types of objectives, e.g., green and cost objectives in our case. Performing a true multi-objective optimization enables displaying a front of the non-dominated, i.e., the best, solutions on a graphical user interface. In the case of

Table 4
Hypervolume metric percentile values for different optimization approaches.

Percentile	Near-optimal strategy optimization (train)	Timetable-based strategy optimization	Tree-based strategy optimization
0th	2804.688949	2195.143914	2629.173918
5th	2804.688949	2195.143922	2631.142458
25th	2804.688949	2195.145606	2635.754203
50th	2804.688949	2195.147142	2638.006121
75th	2804.688949	2195.147142	2640.276218
95th	2804.688949	2195.147142	2648.242975
100th	2804.688949	2195.147142	2651.027347

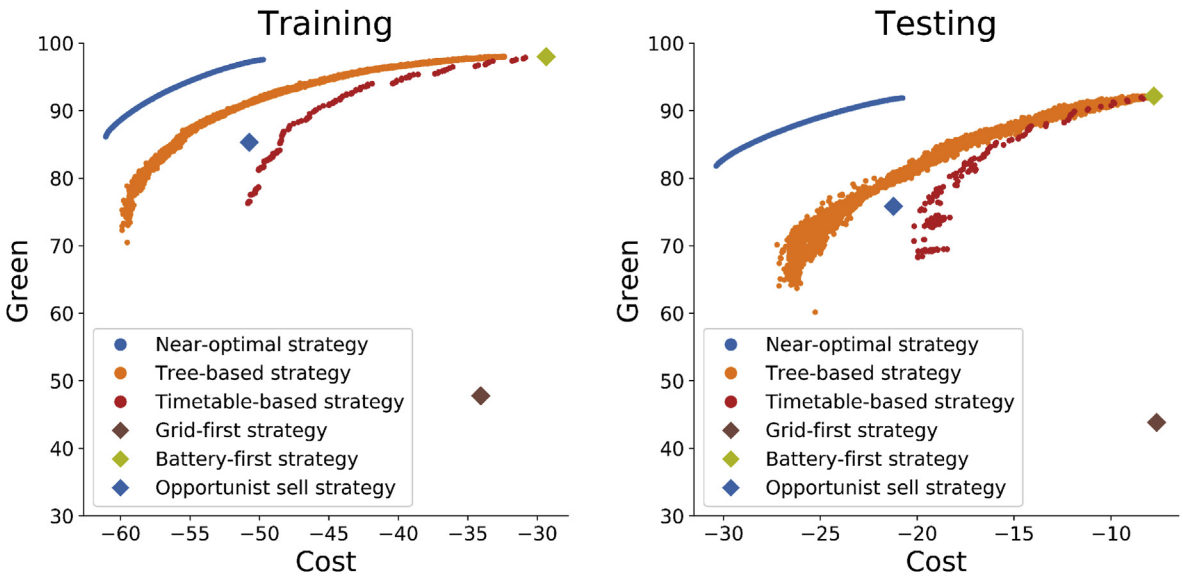

Fig. 11. Union of the final non-dominated solutions (left: training data, right: test data) of all the runs using NSGA-II for the multi-objective timetable-based strategy optimization, the proposed algorithm for the multi-objective tree-based strategy optimization, the near-optimum strategy optimization, and other predefined strategies.

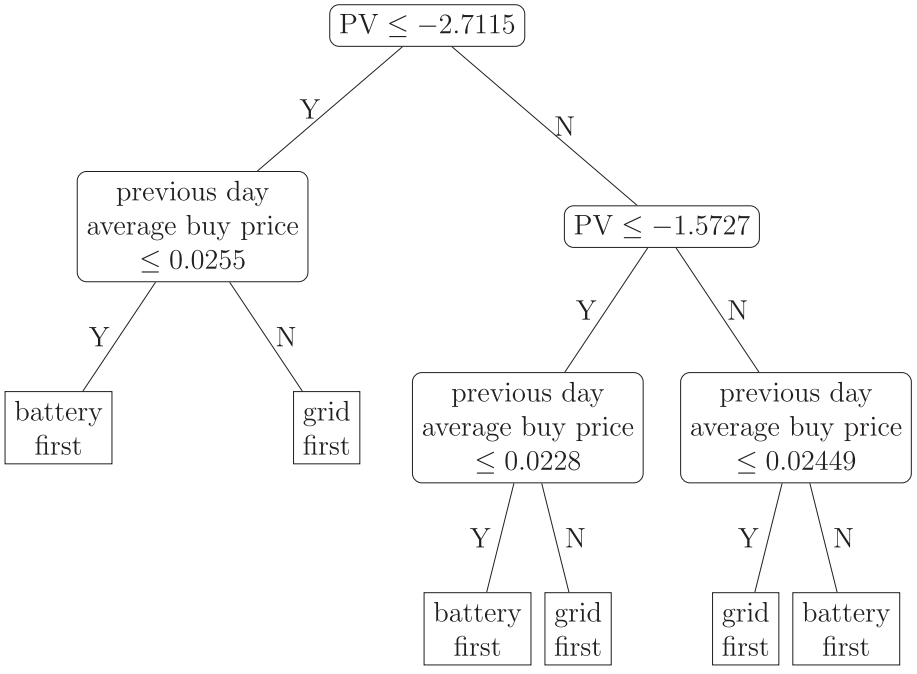

Fig. 12. Tree-based strategy with best cost criterion performance in the last generation of one random run.

two objectives, the user is presented with a two-dimensional plot of strategy performance. Inspecting different solutions the user can easily observe the gains in one objective and losses in the other. Note that the gain in one objective necessarily means a loss in the other objective, since all the presented non-dominated solutions are incomparable according to the Pareto-dominance relation.

Since the presented approach only manages the energy flows and no appliance is directly controlled, it means that there is no need for the user to change his or her behavior regarding the usage of home appliances. This is in contrast with some other approaches in the related work, where some appliances can be operational only inside the specified time window, in order to achieve the optimal performance.

7. Conclusions

The optimization of HEMS is an important issue in sustainable smart home energy management. Our approach describes the optimization procedure that enables the users to choose the preferred trade-off between multiple criteria: costs, ecology, comfort etc. The optimization solutions can be precomputed in advance and are represented in the form of decision trees, integrating higher-level strategic with lower-level operational decisions, e.g., *sunny-afternoon* and *rainy-evening* with an adaptation to the actual situation. This results in well performing strategies that require low computational resources when deployed in HEMS.

Instead of schedules, our system automatically designs decision

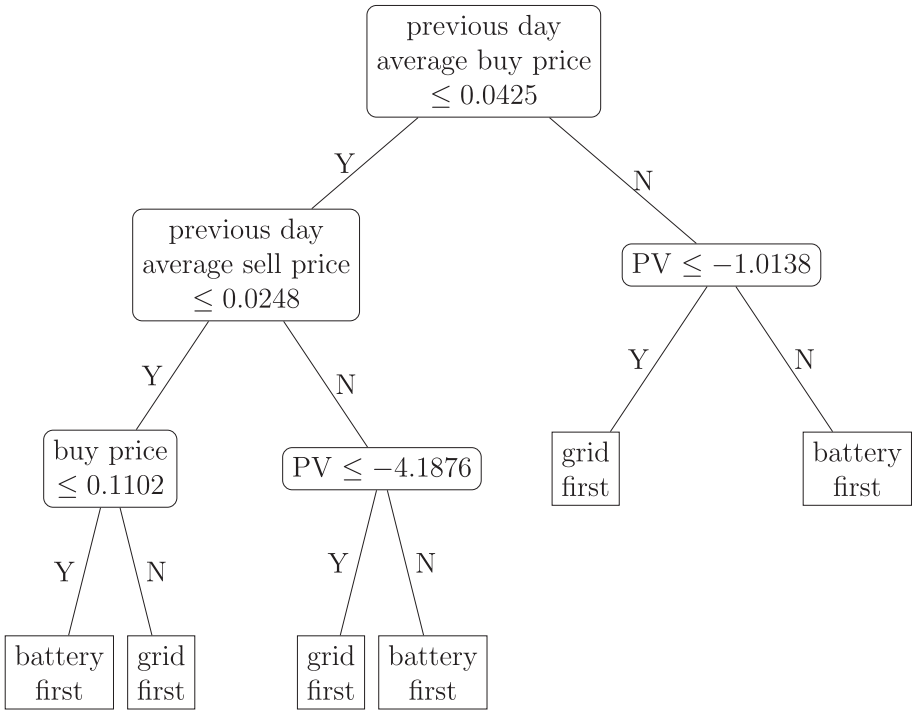

Fig. 13. Tree-based strategy with median cost criterion performance in the last generation of one random run.

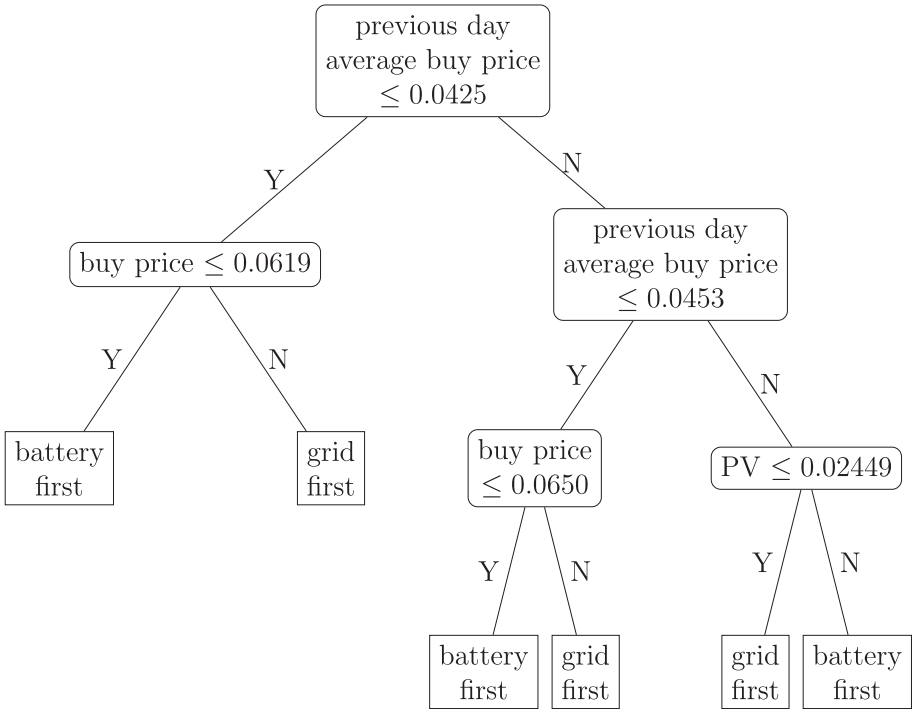

Fig. 14. Tree-based strategy with best green criterion performance in the last generation of one random run.

trees based on history data, and optimizes them. Computationally, the decision trees are as demanding as schedules, but offer greater expression power and advanced optimization possibilities. To demonstrate the improvements, the tree-based strategy optimization is compared to the timetable-based strategy optimization and predefined strategies, those found in the literature and those used by currently available energy-management systems. Additionally, the near-optimum strategy optimization with NSGA-II was used as a reference to measure how the performance of different approaches can be approximated to the optimal one. The tree-based strategies consistently outperformed the timetable-based strategies on the training and test data. The optimization of tree-based strategies yielded solutions with the hypervolume indicator value increase of 20% over the hypervolume indicator value

00:00–00:30	use battery first	10:00–10:30	use grid first
00:30–01:00	use grid first	10:30–11:00	use battery first
01:00–01:30	use battery first	11:00–13:30	use grid first
01:30–02:00	use grid first	13:30–14:30	use battery first
02:00–02:30	use battery first	14:30–15:00	use grid first
02:30–03:00	use grid first	15:00–15:30	use battery first
03:00–04:00	use battery first	15:30–16:00	use grid first
04:00–04:30	use grid first	16:00–16:30	use battery first
04:30–05:00	use battery first	16:30–17:00	use grid first
05:00–05:30	use grid first	17:00–17:30	use battery first
05:30–07:00	use battery first	17:30–18:30	use grid first
07:00–08:30	use grid first	18:30–19:00	use battery first
08:30–09:00	use battery first	19:00–19:30	use grid first
09:00–09:30	use grid first	19:30–23:30	use battery first
09:30–10:00	use battery first	23:30–24:00	use grid first

Fig. 15. Timetable-based strategy with best cost criterion performance in the last generation of one random run.

00:00–00:30	use battery first	10:00–10:30	use grid first
00:30–01:00	use grid first	10:30–11:00	use battery first
01:00–01:30	use battery first	11:00–13:30	use grid first
01:30–02:00	use grid first	13:30–14:30	use battery first
02:00–02:30	use battery first	14:30–15:00	use grid first
02:30–03:00	use grid first	15:00–15:30	use battery first
03:00–04:00	use battery first	15:30–16:00	use grid first
04:00–04:30	use grid first	16:00–16:30	use battery first
04:30–05:00	use battery first	16:30–17:00	use grid first
05:00–05:30	use grid first	17:00–17:30	use battery first
05:30–07:00	use battery first	17:30–18:30	use grid first
07:00–08:30	use grid first	18:30–19:00	use battery first
08:30–09:00	use battery first	19:00–19:30	use grid first
09:00–09:30	use grid first	19:30–23:30	use battery first
09:30–10:00	use battery first	23:30–24:00	use grid first

Fig. 16. Timetable-based strategy with median cost criterion performance in the last generation of one random run.

00:00–24:00	use battery first

Fig. 17. Timetable-based strategy with best green criterion performance in the last generation of one random run.

Table 5
Objective values of selected solutions and improvement.

Solution	Timetable-based strategy optimization		Tree-based strategy optimization		Improvement	
	Cost	Green	Cost	Green	Cost	Green
Best cost	− 50.847	76.278	− 59.551	73.254	+ 17.1%	− 3.9%
Median cost	− 47.551	88.098	− 47.762	93.203	+ 0.4%	+ 5.8%
Best green	− 29.363	98.011	− 32.302	98.011	+ 10.0%	0.0%

obtained from the timetable-based strategies. Three representative solutions of a random optimization run for tree- and timetable-based approaches were also compared. The tree-based solution with the best cost objective resulted in a 17% improvement in the cost objective, when compared to the timetable-based solution. The median cost solutions of both approaches performed similarly. In the case of the best green solutions, the tree-based approach yielded a 10% improvement in the cost objective, while keeping the green objective the same, when compared to the timetable-based approach. The higher expression power of the tree-based strategies compared with the timetable-based strategies was also proven theoretically.

The main advantages of the presented approach over the existing ones are the following:

1. The proposed tree-based solutions are computationally much less expensive than online approaches that require optimal strategy recomputation for each time period.
2. Compared to other precomputed strategy approaches demanding approximately the same online computing capabilities, the proposed tree-based strategies outperform the timetable-based and manually defined strategies by up to 17% in terms of the cost objective while keeping the green objective fixed.
3. The proposed approach is based on the true Pareto-based multi-objective optimization, where the user can pick the solution with the preferred trade-off after the trade-offs are clearly presented to him or her. This improves the user experience with respect to the approaches that use the weighted sum of the objectives, since the weights are hard to define in advance.
4. The proposed approach manages the energy flows to and from the battery and the grid, which means that there is no need for the user to adjust his or her habits of using the appliances as is the case in some of the energy-management systems.
5. The source code for running the energy-management systems simulation and optimization is provided.

The limitations of the presented approach are:

1. To fully adapt the strategies to a particular location, historical data on the load, electric energy prices and solar irradiation is required with a 30-min resolution. This data is often not available in traditional energy systems.
2. To offload the computationally intensive optimization step to an off-site location, an internet connection is required. While this is usually not a problem, since smart home energy-management systems are often already connected, it could present an obstacle in some cases.

Several research directions based on the presented work are possible. A similar approach can be tested for other strategies, such as neural networks; however, the decision trees can still be preferred when an explainable model is required.

Additionally, the feature generation could be automated. Currently, there is a need to specify the features by hand, which requires some domain knowledge. Feature transformation of the time-series data could be applied in order to increase the performance; however, at the cost of explainability.

The proposed tree-based strategy optimization method could be improved using cooperative bi-level optimization methods. The two identified optimization levels are the following: a decision-tree structure optimization on the upper level and a threshold-value optimization that corresponds to the given decision tree on the lower level. Both levels are cooperative – they both strive to optimize the same objectives. Currently, a random search using only one instance generation is used at the lower level, i.e., the inner nodes' test values for each new tree-based strategy are generated randomly and are not subject to any optimization.

Furthermore, the method of multi-objective strategy optimization could be applied to other domains besides energy management. Optimizing the energy-management system as presented in this paper can be categorized as a Multi-Objective Reinforcement Learning (MORL) problem with complex rewards (the rewards are not additive and are delayed). This class of problems has only recently been addressed [36]. The proposed method can be classified as a multi-policy approach for MORL problems [36] and is therefore applicable to several other domains. Since the implementation is available [32], the described problem can also be included in MORL benchmark problems. The simulator provides a complex, real-life MORL problem implementation, of which only a few are reported in the literature.

Declaration of competing interest

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

Acknowledgment

This work was supported by the Slovenian Research Agency (Research core funding No. P2-0209).

References

[1] Kovacic Z, Giampietro M. Empty promises or promising futures? The case of smart grids. Energy 2015;93:67–74.
[2] Leitão J, Gil P, Ribeiro B, Cardoso A. A survey on home energy management. IEEE Access 2020;8:5699–722. https://doi.org/10.1109/ACCESS.2019.2963502.
[3] Wen L, Zhou K, Yang S, Lu X. Optimal load dispatch of community microgrid with deep learning based solar power and load forecasting. Energy 2019;171:1053–65.
[4] Deshmukh M, Deshmukh S. Modeling of hybrid renewable energy systems. Renew Sustain Energy Rev 2008;12(1):235–49.
[5] Foucquier A, Robert S, Suard F, Stéphan L, Jay A. State of the art in building modelling and energy performances prediction: a review. Renew Sustain Energy Rev 2013;23:272–88.
[6] Catalão JPS, Mariano SJPS, Mendes VMF, Ferreira LAFM. Short-term electricity prices forecasting in a competitive market: a neural network approach. Elec Power Syst Res 2007;77(10):1297–304.
[7] Zia MF, Elbouchikhi E, Benbouzid M. Microgrids energy management systems: a critical review on methods, solutions, and prospects. Appl Energy 2018;222:1033–55. https://doi.org/10.1016/j.apenergy.2018.04.103. URL, http://www.sciencedirect.com/science/article/pii/S0306261918306676.
[8] Essiet IO, Sun Y, Wang Z. Optimized energy consumption model for smart home using improved differential evolution algorithm. Energy 2019;172:354–65. https://doi.org/10.1016/j.energy.2019.01.137. URL, http://www.sciencedirect.com/science/article/pii/S0360544219301537.

[9] Lu Q, Lü S, Leng Y, Zhang Z. Optimal household energy management based on smart residential energy hub considering uncertain behaviors. Energy 2020;195:117052. https://doi.org/10.1016/j.energy.2020.117052. URL, http://www.sciencedirect.com/science/article/pii/S0360544220301596.
[10] Guo Z, Zhou K, Zhang X, Yang S. A deep learning model for short-term power load and probability density forecasting. Energy 2018;160:1186–200.
[11] 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;236:700–10.
[12] Roy K, Mandal KK, Mandal AC. Ant-lion optimizer algorithm and recurrent neural network for energy management of micro grid connected system. Energy 2019;167:402–16. https://doi.org/10.1016/j.energy.2018.10.153. URL, http://www.sciencedirect.com/science/article/pii/S0360544218321509.
[13] Berlink H, Costa AH. Batch reinforcement learning for smart home energy management. In: Proceedings of the 24th international joint conference on artificial intelligence. AAAI; 2015. p. 2561–7.
[14] Chaouachi A, Kamel RM, Andoulsi R, Nagasaka K. Multiobjective intelligent energy management for a microgrid. IEEE Trans Ind Electron 2013;60(4):1688–99.
[15] Leonori S, Paschero M, Mascioli FMF, Rizzi A. Optimization strategies for microgrid energy management systems by genetic algorithms. Appl Soft Comput 2020;86:105903. https://doi.org/10.1016/j.asoc.2019.105903. URL, http://www.sciencedirect.com/science/article/pii/S1568494619306842.
[16] Zhang Y, Zeng P, Li S, Zang C, Li H. A novel multiobjective optimization algorithm for home energy management system in smart grid. Math Probl Eng 2015;2015:19.
[17] Liu J, Chen X, Yang H, Li Y. Energy storage and management system design optimization for a photovoltaic integrated low-energy building. Energy 2020;190:116424. https://doi.org/10.1016/j.energy.2019.116424. URL, http://www.sciencedirect.com/science/article/pii/S036054421932119X.
[18] Vamplew P, Issabekov R, Dazeley R, Foale C, Berry A, Moore T, Creighton D. Steering approaches to Pareto-optimal multiobjective reinforcement learning. Neurocomputing 2017;263:26–38.
[19] Kitamura S, Mori K, Shindo S, Izui Y, Ozaki Y. Multiobjective energy management system using modified MOPSO. IEEE international conference on systems, man and cybernetics, vol. 4. IEEE; 2005. p. 3497–503. 2005.
[20] Yang R, Wang L. Multi-objective optimization for decision-making of energy and comfort management in building automation and control. Sustainable Cities and Society 2012;2(1):1–7.
[21] R. Carty, J. T. Wenzinger, Systems and methods for optimizing energy and resource management for building systems, US Patent App. 14/971,236 (Jun. 30 2016).
[22] Public act 094-0977. June 2019, http://www.ilga.gov/legislation/publicacts/fulltext.asp?Name=094-0977; 2006.
[23] Muzathik A. Photovoltaic modules operating temperature estimation using a simple correlation. Int J Energy Eng 2014;4(4):151.
[24] Zhu J, Lin Y, Lei W, Liu Y, Tao M. Optimal household appliances scheduling of multiple smart homes using an improved cooperative algorithm. Energy 2019;171:944–55.
[25] Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 2002;6(2):182–97.
[26] Koza JR. Genetic programming as a means for programming computers by natural selection. Stat Comput 1994;4(2):87–112.
[27] ARSO, archive Weather. http://meteo.arso.gov.si/met/en/app/webmet. [Accessed December 2018].
[28] Trindade A. Electricityloaddiagrams20112014 data set. https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014. [Accessed December 2018].
[29] EPEXSPOT. Intraday continuous. https://www.epexspot.com/en/market-data/intradaycontinuous/intraday-table/-/DE_HIST. [Accessed December 2018].
[30] Python. https://www.python.org/. [Accessed May 2019].
[31] Fortin F-A, De Rainville F-M, Gardner M-A, Parizeau M, Gagné C. DEAP: evolutionary algorithms made easy. J Mach Learn Res 2012;13:2171–5.
[32] Zupančič J. HEMS simulator and multiobjective strategy optimization. April 2019, https://gitlab.com/Jernej88/phd-2/; 2019.
[33] van der Walt S, Colbert SC, Varoquaux G. The NumPy array: a structure for efficient numerical computation. Comput Sci Eng 2011;13(2):22.
[34] McKinney W. Data structures for statistical computing in Python. In: Proceedings of the 9th Python in science conference; 2010. p. 51–6. Austin, TX.
[35] Zitzler E, Thiele L. Multiobjective optimization using evolutionary algorithms – a comparative case study. In: International conference on parallel problem solving from nature. Springer; 1998. p. 292–301.
[36] Roijers DM, Vamplew P, Whiteson S, Dazeley R. A survey of multi-objective sequential decision-making. J Artif Intell Res 2013;48:67–113.