Heat FlexOffers

a device-independent and scalable representation of electricity-heat flexibility

Lilliu, Fabio; Pedersen, Torben Bach; Siksnys, Laurynas

Published in:
e-Energy 2023 - Proceedings of the 2023 14th ACM International Conference on Future Energy Systems

DOI (link to publication from Publisher):
10.1145/3575813.3597347

Publication date:
2023

Document Version
Publisher's PDF, also known as Version of record

Heat FlexOffers: a device-independent and scalable representation of electricity-heat flexibility.

Fabio Lilliu
Aalborg University
Aalborg, Denmark

Torben Bach Pedersen
Aalborg University
Aalborg, Denmark

Laurynas Šikšnys
Aalborg University
Aalborg, Denmark

ABSTRACT

The increasing relevance of Renewable Energy Sources (RES) makes energy flexibility an extremely important aspect, not only regarding electricity, but also for other energy vectors such as heat. Because of this, there is the need for a flexibility model which can i) provide a common representation of flexibility for different device types, ii) perform aggregation, optimization and disaggregation while scaling for long time horizons and many devices, iii) capture most of the available flexibility, and iv) support energy conversion between different vectors. Properties i)-iii) are addressed by FlexOffer (FO), a device-independent model that describes energy constraints in an approximate yet accurate way. This paper proposes an extension of FOs, **Heat FlexOffers** (HFOs), capable of modeling flexibility for different energy vectors such as heat and handling energy conversion, and therefore addressing iv) as well as i)-iii). HFOs can model the optimal power curve for heat pumps, and can provide constraints for continuous optimization problems while complying to the Smart Grid-Ready (SG-Ready) interface, which operates on discrete states. We show that HFOs are very accurate, being able to retain up to 98.9% of total flexibility before aggregation and 98.1% of it after aggregation. HFOs aggregation is scalable, as $2 \cdot 10^6$ devices can be aggregated for a 24 hours time horizon, vastly outperforming exact models as they fail to aggregate more than 500 devices.

CCS CONCEPTS

• **Theory of computation → Mathematical optimization**.

KEYWORDS

Energy flexibility, Heat-Electricity conversion, Sector coupling

ACM Reference Format:
Fabio Lilliu, Torben Bach Pedersen, and Laurynas Šikšnys. 2023. Heat FlexOffers: a device-independent and scalable representation of electricity-heat flexibility. . In *The 14th ACM International Conference on Future Energy Systems (e-Energy '23), June 20–23, 2023, Orlando, FL, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3575813.3597347

1 INTRODUCTION

Motivation. Today, supporting more renewable energy is of utmost importance. One of the most important challenges regarding renewable sources is to be able to exploit them despite the non-controllable nature of their generation. This can be done by exploiting energy flexibility [22]: that is, the capability of changing the consumption or production of energy in time and/or amount. Usage of flexibility allows to schedule energy consumption in order to maximize usage of renewable energy [33], preventing congestions [12], minimizing energy cost [27, 28] or CO2 emissions [16], and provide ancillary services [36], demand response [6] and peak shaving [5, 13]. Energy flexibility has traditionally been used for electricity only; however, in order to support sector coupling, there is a need for it to cover other energy vectors, like heat. We want our model to have some important properties [25], such as i) represent flexibility from many different device types [18]; ii) optimize flexibility for many different purposes, aggregate a large number of small loads into a few, larger loads [8], and perform the opposite process (which is called *disaggregation*) [29] in a scalable way; iii) be able to represent most/all of the available flexibility; and iv) support multiple energy vectors and conversion between them.

Related work. Many existing models have tried to satisfy one or more of them. Regarding i), [34] and [7] describe flexibility representations in an unified format. Property ii) has been addressed by [21] for flexibility aggregation, and [15] focuses on scalability for both optimization and aggregation. For iii), accuracy is addressed by state-space models, like [19] does for building heating systems and water towers, although those models are in general not scalable. Regarding iv), [3] describes several works on flexibility for energy conversion, in particular electricity-to-heat, describing their optimization objectives. The model called FlexOffer (FO) [32] respects all the properties i) to iii); however, until now, it has been modeling electric energy only. FOs are device-independent [30], allow for aggregation [35] and optimization [14] of a large number of devices, and represent flexibility accurately [23, 24]. As we are interested in energy conversion, we will analyze closely the case for heat pumps, and in particular their compliance to the SG-Ready standards [9], operating modes [4, 10, 11] and constraints [20].

Contributions. The scope of this paper is to extend the FO model in order to handle energy conversion and represent energy vectors different from electricity, in particular heat. We will present a new model, Heat FlexOffers (HFOs), which can capture energy flexibility in terms of heat: it is based on the Dependency FlexOffers(DFOs) [35] model, and thus it can interact with existing FOs for electricity. HFOs consider time as discrete, and divided into intervals called *time slices*; however, HFOs are capable of representing power curves inside each time slice. This allows HFOs to represent flexibility very accurately, capturing up to 98.9% of total flexibility. They are scalable, as it is possible to aggregate, optimize and disaggregate $2 \cdot 10^6$ HFOs for a time horizon of 24 hours, while exact models fail to aggregate more than 500 devices.

e-Energy '23, June 20–23, 2023, Orlando, FL, USA

ACM ISBN 979-8-4007-0032-3/23/06.
https://doi.org/10.1145/3575813.3597347

Parameter	Description
U	Internal energy of the room.
Q	Heat given to the room.
W	Thermodynamic work.
COP	Coefficient of performance of the heat pump.
E	Electrical energy used.
A	Surface area where the heat exchange happens.
c_{ht}	Overall heat coefficient transfer.
c	Heat capacity of the air.
Te	Temperature.
T	Time horizon.

Table 1: Table of symbols used through this paper.

Paper structure. The rest of the paper is structured as follows. Section 2 describes the FO model and the physical model for a heat pump. Section 3 introduces HFOs and describes their generation and aggregation. Section 4 contains the experimental evaluation, and Section 5 concludes the paper and points to future work.

2 PRELIMINARIES

2.1 Heat pump specifications

In this work, we will assume that the heat pumps operate following the Smart Grid Ready (SG-Ready) specifications [9]. SG-Ready is an interface that has been designed for operating heat pumps in order to exploit their flexibility. There are more than 1200 heat pump models adopting this interface. SG-Ready allows four operating modes, which are described in Table 2. [4] [10]

In the *Off* mode, the heat pump operation is switched off. In the *Normal* mode, the heat pump operates within normal set points, in an energy-efficient way. In the *Recommended On* mode, the heat pump operates in an enhanced heating mode, with increased hysteresis. In the *Forced On* mode, the heat pump operates at its maximum power [11]. In this paper, we will assume that heat pumps operate under these four modes.

2.2 Room with stable temperature

In this paper, we will work in the scenario where there are user-defined constraints on temperature: more precisely, a minimum temperature Te_{min} and a maximum temperature Te_{max} for the room. The goal will be to model flexibility and use it for optimization (e.g., cost minimization) while respecting the temperature constraints. This subsection will introduce the physical models for heating that we will consider throughout the paper. However, for simplicity, we will start from the most simple case, where $Te_{min} = Te_{max}$: in other words, we are considering a room where the temperature is kept constant. In this case, the internal energy of the room (denoted by U) remains the same. Eq. 1 describes the first law of thermodynamics. Here and in the rest of the paper, Q is the amount of heat given to the room and W is the amount of thermodynamic work done by the room - in this case, the amount of dispersed heat.

$$\Delta U = Q - W \tag{1}$$

Since U by hypothesis does not change, we have $\Delta U = 0$ and therefore $Q = W$. That is, the amount of heat given to the room must be equal to the amount of dispersed heat.

Operating mode	Description
Off	Heat pump is switched off.
Normal	Heat pump operates in an energy-efficient mode.
Recommended On	Heat pump operates on an enhanced heating mode.
Forced On	Heat pump operates at its maximum power.

Table 2: SG-Ready modes.

The heat is provided to the room by a heat pump. The heat pump converts electrical energy into heat, according to Eq. 2: here, Q is the output heat, E is the amount of electrical energy provided to the heat pump and COP is the coefficient of performance, which determines the ratio at which electrical energy is converted to heat.

$$Q = COP \cdot E \tag{2}$$

Heat is dispersed from the room according to Eq. 3. Here, W is the amount of dispersed energy, with the same notation of Eq. 1. A is the surface area of the room on which heat exchange happens, and c_{ht} is the overall heat transfer coefficient, a value expressed in $\frac{W}{m^2K}$ which determines how much heat is transferred across the walls per surface unit. Finally, Te_{in} and Te_{out} are the temperatures inside and outside the room respectively, and t is the time frame considered.

$$W = A \cdot c_{ht} \cdot (Te_{in} - Te_{out}) \cdot t \tag{3}$$

Substituting Q according to Eq. 2 and W according to Eq. 3, Eq. 1 becomes $Q = A \cdot c_{ht} \cdot (Te_{in} - Te_{out}) \cdot t$ if we are reasoning in terms of heat, or $COP \cdot E = A \cdot c_{ht} \cdot (Te_{in} - Te_{out}) \cdot t$ if we are considering electrical energy.

2.3 Example room

Throughout this paper, we will use a simple running example to illustrate our models. We will consider the case of a room with the following specifics: its section is a rectangle with sides 4 and 5 meters respectively, and it is 3 meters high. The walls have $c_{ht} = 6\frac{W}{m^2K}$ [17], and heat dispersion happens through one of the 3m x 4m side walls. The room temperature is 295K (22°C) and the outside temperature is 275K (2°C). In the room, a *Daikin Altherma*[1] heat pump operates, with maximum operational power $P_{max} = 4.6$ kW and $COP = 3.65$. This model supports the SG-Ready interface.

In this case, the amount of energy dispersed in one hour is

$$W = (3 \cdot 4)m^2 \cdot 6\frac{W}{m^2K} \cdot (295K - 275K) \cdot 3600s = 5184000J = 1.44kWh$$

Which means that an amount of heat equal to $1,44$ kWh has to be provided in order to keep the temperature stable. With a COPof 3.65, the amount of electrical energy needed to provide this amount of heat is $E = \frac{Q}{COP} = \frac{1.44kWh}{3.65} = 0.395kWh$.

[1] https://www.daikin.eu/content/dam/document-library/catalogues/heat/air-to-water-heat-pump-low-temperature/Daikin_Altherma_3/DaikinAltherma3_Product\Catalogue_ECPEN18-786B_English.pdf

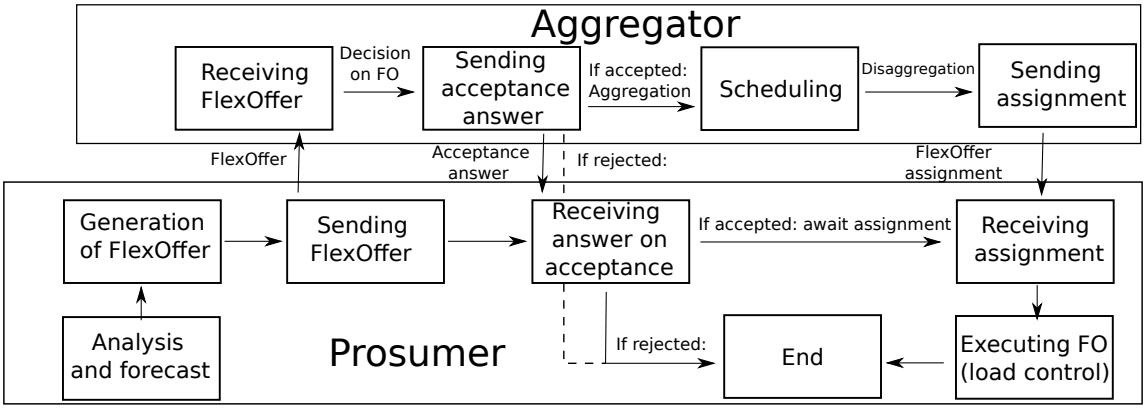

Figure 1: A schematic description of the FO life-cycle[24]

2.4 Room with changing temperature

In this subsection we will study the most general case, where the temperature of the room can change depending on how much energy is provided by the heat pump.

The change of temperature inside the room is correlated with the change of internal energy in it. The formula that encodes this correlation is

$$\Delta Te = \frac{\Delta U}{c \cdot m} \tag{4}$$

where c is the specific heat capacity of the air, and m is the mass of air inside the room. Therefore, if we want to increase the room temperature by a specified amount ΔTe, according to Eq. 1 we have to provide an amount of heat over time t equal to $Q = \Delta U + W$ which, plugging in Eq. 3 and Eq. 4, becomes

$$Q = c \cdot m \cdot \Delta Te + A \cdot c_{ht} \cdot (Te_{in} - Te_{out}) \cdot t. \tag{5}$$

Note that Eq. 3 works under the assumption that temperature does not change, so it can only be used to approximate the problem for small temperature variations. The more generic version of Eq. 3 is $W = A \cdot c_{ht} \cdot \int_{t_0}^{t_1} Te_{in}(t) - Te_{out}(t)dt$, where t_0 is the time at which we start calculation dispersion, t_1 the time at which we stop calculating dispersion, and we assume that Te_{in} and possibly Te_{out} vary in function of time t. In this case, Eq. 5 would become

$$Q = c \cdot m \cdot \Delta Te + A \cdot c_{ht} \cdot \int_{t_0}^{t_1} Te_{in}(t) - Te_{out}(t)dt. \tag{6}$$

This can also be written in differential form

$$P = c \cdot m \cdot \dot{Te}_{in} + A \cdot c_{ht} \cdot (Te_{in} - Te_{out}) \tag{7}$$

where P is power, and $\dot{Te}$ is the derivative of Te over time.

We will show an example of this. We will work under the same settings of the running example, and we suppose that the user-defined constraints for temperature are 293K (20°C) and 297K (24°C) respectively. We show the calculations for bringing the temperature from 293K up to 297K. At 295K the specific heat capacity of the air is $c = 1005\frac{J}{kg \cdot K}$, and density of the air is $1.225\frac{kg}{m^3}$, which means that the mass of air inside the room is $m = 1.225\frac{kg}{m^3} \cdot 60m^3 = 73.5kg$. With the approximation from Eq. 5, we have that in one hour we have to provide an amount of heat equal to

$$Q = 1005\frac{J}{kg \cdot K} \cdot 73.5kg \cdot 4K + 12m^2 \cdot 6\frac{W}{m^2K} \cdot 20K \cdot 3600s = 1.522kWh$$

which means using up an amount of electricity equal to $E = \frac{1.522kWh}{3.65} = 0.417kWh$. The formula from Eq. 6 gives the same result under the assumption that Te_{out} is constant over time and that Te_{in} increases linearly with time.

2.5 FlexOffer life-cycle

The FO model [32] is the starting point of this paper. FOs are representations of flexibility over a sequence of time slices - time intervals of a precisely defined length, e.g., 15 minutes. For each slice, FOs describe a set of constraints on the amount of energy that can be consumed during that, which defines the available flexibility. FOs refer to consumed energy with positive sign, and produced energy with negative sign. The life-cycle of an FO, i.e., the processes an FO goes through from its generation to its end, is described in Figure 1 [24]. In the figure, the interaction between the prosumer and the aggregator [30] is described. An agent operates on the behalf of the prosumer, and all the tasks are carried out automatically. FOs are generated by the prosumer, after a process of analysis and prediction of the available flexibility. The prosumer then sends the FOs to the aggregator, who receives them. Each FO can be either accepted or rejected, depending whether if it is useful for the aggregator or not, and the acceptance answer is sent back to the prosumer. Rejected FOs are discarded, and the cycle ends at this point for them. Accepted FOs are processed (e.g., aggregated to other FOs, optimized) by the aggregator, who then determines schedules for each FO. The aggregator then sends FO schedules to the prosumer, and finally the prosumer agent will operate the devices according to those schedules.

2.6 Description of a FlexOffer

An FO is a representation of flexibility, which describes the amount of energy available for consumption at each time slice. In this subsection, we will refer to the amount of energy that is consumed at the t-th time slice as e_t, and to the total time horizon we are

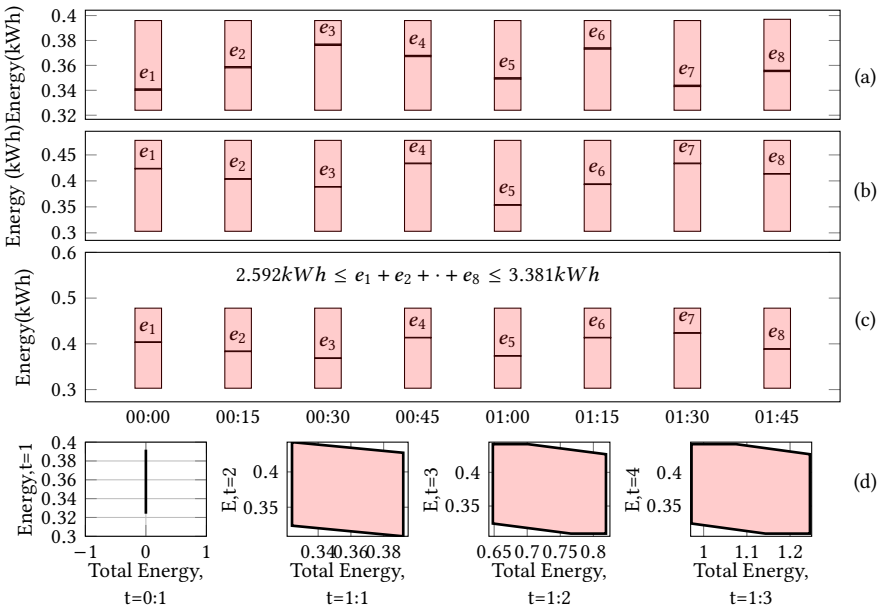

Figure 2: Inner (a), outer (b), TEC (c) SFOs, and DFOs (d)

considering as T - in other words, T is the number of considered time slices. FOs are characterized by the types of contraints used to define them. The simplest constraint type are *slice* (energy) *constraints*. At each time slice t, an energy constraint defines a minimum and a maximum quantity of energy that can be consumed within that slice: in more formal terms, a lower bound e_{min_t} and an upper bound e_{max_t} which define the constraint $e_{min_t} \leq e_t \leq e_{max_t}$. FOs whose constraints are all slice constraints are called *standard FOs* (SFOs). Next, we have *total energy constraints* (TECs). A TEC defines two numbers, TEC_{min} and TEC_{max}, which define the minimum and maximum total amount of energy that can be consumed over the considered T time slices. We define a *total energy constraint standard FO* (TEC-SFO) as an FO whose constraint are all slice constraints and TECs.Finally, we have *dependent energy constraints*. A dependent energy constraint defines, at a certain time t, the minumum and maximum consumable amount of energy; however, this amount depends on how much energy has been consumed before t. We can express this in more formal terms as follows: it is possible to define three real numbers a, b, c such that $a \cdot (e_1 + \ldots + e_{t-1}) + b \cdot e_t \leq c$. FOs with dependency energy contraints are called *dependency FOs* (DFOs). FOs can approximate flexibility in a more conservative or aggressive way, depending on how they are generated. In particular, it may happen that an FO captures less flexibility than the amount which is actually available: this is called an *inner* flexibility approximation. Conversely, if an FO captures more flexibility than what is actually available, we call it an *outer* flexibility approximation. An example of FO constraints can be seen in Figure 2: the figure refers to the case described in Section 2.4. For Figure 2 (a-c) the horizontal lines indicate a possible *schedule*: a schedule is a vector $(e_1, \ldots, e_T)$, where each value e_t indicates the consumption of energy during the $t-$th time slice. Figure 2 (a) and Figure 2 (b) represent SFOs: the SFO in Figure 2 (a) is an inner approximation, the SFO in Figure 2 (b) is an outer approximation. For each time slice, the bar indicates how much energy can be consumed. Going further, Figure 2 (c) shows a TEC FO, in which we can see both the slice constraints and the TEC, which is expressed by the double inequality in the upper part of the figure, which has to be respected in addition to the slice constraints. Finally, Figure 2 (d) represents a DFO. At each time slice t, flexibility is represented by a convex polyhedron, where the x axis is the total amount of energy that has been consumed before time t, and the y axis is the amount of energy that can be consumed at time t. The available values for consumable energy depend on the amount of energy consumed before time t:the idea is that the more the room has been heated up earlier, the less it can be heated up later.

3 HEAT FLEXOFFER MODEL

3.1 State-space model

A state-space model is a model that describes a physical system by a set of state variables, whose interaction is defined by a first order differential equation. From now on, unless specified otherwise, we will assume that Te_{out} is constant. In our case, Eq. 7 describes the interaction between the variables P and Te_{in} by a first order differential equation. Since the relationship between these two variables is linear, we call this a linear time invariant (LTI) system. We want to use, whenever possible, this kind of model to describe the interaction between the variables Te_{in} and Q. In general, an LTI state-space model can represent a discrete time model as

$$\begin{aligned} state(t+1) &= \overline{A} \cdot state(t) + \overline{B} \cdot input(t) + \overline{f} \\ output(t+1) &= \overline{C} \cdot state(t) + \overline{D} \cdot input(t). \end{aligned} \tag{8}$$

In general, *state* indicates the state variables of the physical system that we are considering, *input* the variables in response

to whom the state of the system changes, and *output* any system variable that can be of interest. In our specific case, *state* is the temperature Te_{in}, and we are interested in the energy value Q, which will then be both *input* and *output*. Finally, $\overline{C} = 0$ and $\overline{D} = 1$.

Heat LTI models need to have a state variable, and the most intuitive choice for it is temperature: this choice works well when describing the evolution of a single room. However, when trying to aggregate multiple models, calculations get more complicated. For example, if we have two different rooms with different temperature, the state representing the whole system is not the sum of the temperatures. It is not even their (simple) average, since this would be wrong if the rooms have different volumes (thermal mass). Even if we took a weighted mean of their temperatures, proportional to the volume of each room, this would still be wrong if other parameters (e.g., c_{ht}) were different. However, if we were able to calculate a weighted mean that takes all the parameters into account, what this mean actually represents is the internal energy of each room. Since the total internal energy of the system is simply the sum of the internal energy of each room, we decided to use internal energy as state. Therefore, we will initially generate our models for temperature, since it is more intuitive to do so, and then convert them to describe the internal energy of the system. With the same notation as Section 2, the formula that regulates this conversion can be roughly approximated by

$$U = c \cdot m \cdot Te_{in} \tag{9}$$

.

Finally, it is possible to generate FOs in terms of E, i.e. electrical energy, by just replacing Q with $COP \cdot E$.

3.2 Objectives and power curves

We used Eq. 5 in order to represent the thermodynamic model via an LTI system. However, this representation is inaccurate in general, and can only be a good approximation for small time durations/temperature changes.

Energy is the integral of power over time; however, given that dispersion of heat depends on the current temperature, different functions that describe power over time (which we will refer to as *power curves*) may result in different temperatures obtained by spending the same amount of energy. We want to see which power curves are of interest, and how FOs would be generated from them. In this section we will only consider one single time slice. We denote the time at which the time slice starts and ends as t_0 and t_1, respectively. The notation will be the same as in Section 2.

First, we have to define which objectives are useful. We identified four objectives that may be of interest. *Minimizing energy consumption* is often a primary objective. Other than that, prosumers may want to maximize their *directional flexibility*, i.e., the capability of providing flexibility by increasing (*upwards flexibility*) or decreasing (*downwards flexibility*) their energy consumption, or also being able to provide both types when needed (*bidirectional flexibility*). Bidirectional flexibility often has to be *symmetrical*, i.e., upwards and downwards flexibility have to be as similar as possible. Directional flexibility is important in many energy markets, such as aFRR [31], mFRR [31], FCR [1], FCR-D [1] and FCR-N [1]. In particular, FCR requires bidirectional flexibility, while for example FCR-D requires only downwards flexibility.

In this section we assume that Te_{out} is constant: therefore, Eq. 7: becomes

$$P = c \cdot m \cdot (\dot{Te}_{in}) + A \cdot c_{ht} \cdot (Te_{in} - Te_{out}) \tag{10}$$

where P and Te_{in} are two functions over time t.

3.2.1 Constant power and maximizing bidirectional flexibility. First, we analyze the case where the provided power is constant over time; that is, there is a certain number $P_0 \in \mathbf{R}$ such that $P(t) = P_0$ for every $t \in [t_0, t_1]$. We will refer to this curve as *Constant*. In this case, the power curve formula is trivial, and the amount of energy spent over time will be $Q(t) = P_0 \cdot t$. Regarding temperature, solving Eq. 10 with constant P, we obtain

$$Te(t) = Te_0 + \left(Te_{out} + \frac{P}{A \cdot c_{ht}} - Te_0\right) \cdot \left(1 - e^{-\frac{A \cdot c_{ht}}{c \cdot m} \cdot t}\right). \tag{11}$$

And in particular, $Te(t_1)$ is the value for temperature at the end of the time slice. The function Te is asymptotical to $Te_{out} + \frac{P}{A \cdot c_{ht}}$, and for t_1 approaching infinity, the temperature will tend to this value. When room temperature is kept at the middle point of the available temperature interval, such as in our running example, this curve maximizes bidirectional flexibility. In this case, by choosing $P_0 = A \cdot c_{ht} \cdot (Te_0 - Te_{out})$, the temperature will remain the same for the whole duration of the time interval: the room temperature can then be increased or decreased by an amount $\frac{Te_{max} - Te_{min}}{2}$, which makes directional flexibility symmetrical, as requested. Figure 3(a) shows the power curve (in cyan) and the temperature (in orange) inside the time slice.

3.2.2 Minimizing consumption - maximizing upwards flexibility. In this subsubsection we want to find the power curve that minimizes energy consumption. Intuitively, energy consumption increases with the room temperature, as a higher temperature will increase energy dispersion: the power curve that minimizes consumption will also provide the highest upwards flexibility, as will be proved below. The problem can be formalized as finding the function P that minimizes energy consumption under the following constraints: the room temperature has an initial value Te_0 and a final value Te_1, the room temperature has to always be between a lower bound Te_{min} and an upper bound Te_{max}, and the heat pump must operate with the SG-Ready modes. The power curve that solves this problem will be denoted as *Optimal*. As cost optimization is usually one of the targets for many use cases, if energy price does not change within the considered time slice (and this can be guaranteed by choosing short enough time slices), the minimum energy consumption curve is also the best power curve for cost reduction.

As we will see now, the curve solving this problem is defined by a piecewise-defined function. First, the heat pump will go into *Off* mode until the temperature drops to Te_{min}: therefore, the function assumes the value 0 until a certain critical time t_{crit1}, which is the time needed for the temperature to drop to Te_{min}. Then, the function will assume the value needed to keep the temperature equal to Te_{min}, as the heat pump operates in *Normal* mode. Finally, the heat pump operates in *Forced On* mode, bringing the room temperature from Te_{min} to Te_1: denoting the time needed for the temperature to reach the value Te_1 by t_{crit2}, the value of the function will then be P_{max} for the last t_{crit2} seconds.

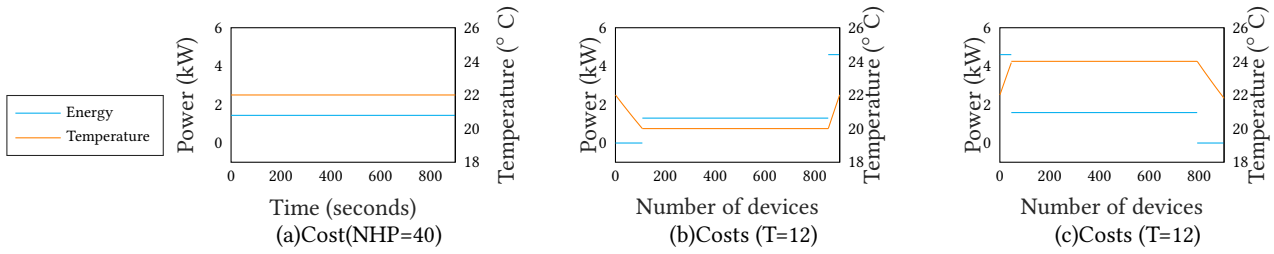

Figure 3: Power curves

Proposition 1. *Let $Te_{in} : [t_0, t_1] \rightarrow \mathbf{R}$ be a function which respects Eq. 10. The function $P(t) : [t_0, t_1] \rightarrow \mathbf{R}$ that minimizes the value $\int_{t_0}^{t_1} P(u)du$ under the constraints*

$$\begin{aligned} &0 \leq P(t) \leq P_{max} \ \forall t \in [t_0, t_1]; \\ &Te_{min} \leq Te_{in}(t) \leq Te_{max} \ \forall t \in [t_0, t_1]; \\ &Te_{in}(t_0) = Te_0, Te_{in}(t_1) = Te_1 \end{aligned} \tag{12}$$

for $Te_0, Te_1 \in [Te_{min}, Te_{max}]$ positive real numbers chosen a priori, is

$$P(t) = \begin{cases} 0 & if\ t \in [t_0, t_{crit1}] \\ A \cdot c_{ht} \cdot (Te_{min} - Te_{out}) & if\ t \in (t_{crit1}, t_1 - t_{crit2}) \\ P_{max} & if\ t \in [t_1 - t_{crit2}, t_1]. \end{cases} \tag{13}$$

where t_{crit1} and t_{crit2} are defined as

$$t_{crit1} = t_0 + \frac{c \cdot m}{A \cdot c_{ht}} \ln \frac{Te_0 - Te_{out}}{Te_{min} - Te_{out}} \tag{14}$$

and

$$t_{crit2} = -\frac{c \cdot m}{A \cdot c_{ht}} \ln \left(1 - \frac{Te_1 - Te_{min}}{Te_{out} - Te_{min} + \frac{P_{max}}{A \cdot c_{ht}}}\right) \tag{15}$$

Proof. If $P = 0$, the amount of time needed for Te to reach the value Te_{min} can be obtained from Eq. 11 by replacing $P = 0$, and is equal to

$$\frac{c \cdot m}{A \cdot c_{ht}} \ln \frac{Te_0 - Te_{out}}{Te_{min} - Te_{out}}.$$

This proves that the function in Eq. 13 minimizes the energy spent in the time interval $[t_0, t_{crit1}]$, and that the temperature constraints are respected: the initial temperature Te_0 is within the constraints, and the temperature lowers until it reaches the value Te_{min} at time t_{crit1}.

Now, for $t \in [t_{crit1}, t_1]$, the amount of energy Q that will be spent can be calculated from Eq. 6. In order to minimize the amount of energy spent, the values ΔTe and $\int Te_{in}(t)$ have to be minimized: this happens if Te_{in} remains equal to Te_{min} for the entire duration of the interval. The corresponding value for heat is

$$Q = A \cdot c_{ht} \cdot (t_1 - t_{crit2} - t_{crit1}) \cdot (Te_{min} - Te_0). \tag{16}$$

The amount of power spent for maintaining the temperature to Te_{min} can be calculated from Eq. 7, and it is equal to $P(t) = A \cdot c_{ht} \cdot (Te_{min} - Te_{out})$. Now, the only thing left is to prove that the function in Eq. 13 minimizes the energy spent in the time interval $[t_1 - t_{crit2}, t_1]$. The reasoning is as follows. First, Eq. 10 is a first order differential equation for Te_{in}, and the solution to this equation is

$$Te(t) = Te_{out} + (Te_0 - Te_{out}) \cdot e^{-\frac{A \cdot c_{ht}}{c \cdot m} \cdot t} + \frac{e^{-\frac{A \cdot c_{ht}}{c \cdot m} \cdot t}}{c \cdot m} \int_{t_0}^{t} P(u) \cdot e^{\frac{A \cdot c_{ht}}{c \cdot m} \cdot u} du. \tag{17}$$

In particular, we see that the higher P is at each moment, the more the temperature increases. So, keeping $Te(t_1) = Te_1$ as a constraint, the strategy that minimizes temperature at each time before t_1 is to keep the temperature at minimum until a certain time, and then use the heat pump at maximum power from that time on, so that the temperature reaches Te_1 at time t_1. We defined t_{crit2} as the time needed for the temperature to reach Te_1: all that remains is to calculate the value of t_{crit2}.

If $P(t) = P_{max}$, $Te_0 = Te_{min}$ and $t_0 = 0$, by definition of t_{crit2} we will have $Te(t_{crit2}) = Te_1$. As the power is constant, we will use Eq. 11. Replacing the values mentioned above, it becomes

$$Te_1 = Te_{min} + \left(Te_{out} + \frac{P_{max}}{A \cdot c_{ht}} - Te_{min}\right) \cdot \left(1 - e^{-\frac{A \cdot c_{ht}}{c \cdot m} \cdot t_{crit2}}\right).$$

Whose solution is as in Eq. 15, and this concludes the proof. □

We can easily see that this power curve respects the temperature constraints of the room. It is compatible with SG-Ready operations: the heat pump will operate in mode *Off* in $[t_0, t_{crit1}]$, *Normal* in $[t_{crit1}, t_1 - t_{crit2}]$, and *Forced On* in $[t_1 - t_{crit2}, t_1]$. Finally, in order to maximize upwards flexibility, the power curve needs to maintain the room temperature to the minimum as long as possible: as we have seen, this is exactly what happens. Figure 3(b) shows how power and temperature change.

3.2.3 Maximizing downwards flexibility. In order to maximize downwards flexibility, the power function P needs to keep the room temperature to the maximum (i.e., equal to Te_{max}) for as long as possible. The function achieving this while respecting the temperature and SG-Ready constraints, behaves in the opposite way of the *Optimal* function: the power consumed is P_{max} (and therefore, the heat pump operates on *Forced On* mode) until the temperature reaches Te_{max}, then its consumption is kept so that the temperature remains constant, and at the very end of the interval the heat pump goes into *Off* mode in order to reach the desired final temperature. We denote by t_{crit1D} and t_{crit2D} respectively the time needed for heating the room up to temperature Te_{max} at the beginning, and the time needed for the room to cool down to temperature Te_1 at

the end. In more formal terms, this is the function that maximizes downwards flexibility:

$$P(t) = \begin{cases} P_{max} & if\ t \in [t_0, t_{crit1D}] \\ A \cdot c_{ht} \cdot (Te_{max} - Te_{out}) & if\ t \in (t_{crit1D}, t_1 - t_{crit2D}) \\ 0 & if\ t \in [t_1 - t_{crit2D}, t_1]. \end{cases} \tag{18}$$

with $t_{crit1D} = -\frac{c \cdot m}{A \cdot c_{ht}} \ln\left(1 - \frac{Te_{max} - Te_0}{Te_{out} - Te_0 + \frac{P_{max}}{A \cdot c_{ht}}}\right)$ and $t_{crit2D} = t_{max} + \frac{c \cdot m}{A \cdot c_{ht}} \ln \frac{Te_{max} - Te_{out}}{Te_1 - Te_{out}}$ This can be proven in the same way as Proposition 1. Figure 3(c) shows how power and temperature change.

3.3 Generation of the flexibility models

3.3.1 Energy constraints. We now want to generate the models for flexibility. As explained in Section 3.1, LTI state-space models are in the form from Eq. 8. Now, from Eq. 11, we can see that Te depends linearly from Te_0 and P (and therefore Q), for the *Constant* curve: because of this, we can write this dependency as in Eq. 8, and we will refer to the model with energy as state variable. The terms $\overline{A}$, $\overline{B}$ and $\overline{f}$ are respectively

$$\overline{A} = e^{-\frac{A \cdot c_{ht} \cdot t_1}{c \cdot m}}; \overline{B} = \frac{1 - e^{-\frac{A \cdot c_{ht} \cdot t_1}{c \cdot m}}}{A \cdot c_{ht} \cdot t_1}; \overline{f} = (1 - e^{-\frac{A \cdot c_{ht} \cdot t_1}{c \cdot m}}) \cdot Te_{out}$$

For this type of power curve, it is possible to generate FOs from the state-space models [35]. However, for the *Optimal* curve, the relationship between temperature and spent amount of energy is not linear anymore. Because of this, we will propose a new algorithm for generating HFOs for those two power curves, which can be seen in Algorithm 1. This algorithm depends on one function, Q_{opt}: given the temperatures Te_0 and Te_1, Q_{opt} calculates the amount of energy needed for the room to go from temperature Te_0 to temperature Te_1 in the duration of one time slice. For simplicity, we will use the notation $Q_{min}(Te) = Q_{opt}(Te_{min}, Te)$ and $Q_{max}(Te) = Q_{opt}(Te_{max}, Te)$ respectively. We will now describe Algorithm 1. Here, T is the time horizon for which we want to issue our HFO, i.e., the number of time slices we want to consider. First, we initialize the HFO (Line 2). We start building the first slice by calculating the maximum (E^1_{max}) and minimum (E^1_{min}) amount of available energy by the functions Q_{min} and Q_{max} (Lines 5-7), and initialize the variables S^i_{min} and S^i_{max} (Lines 7-9): they describe the minimum and maximum amount of energy spent up to the considered time, depending on the amount of energy spent up to the previous time. We need both the E and S variables since we are building polygons like in DFOs, where the x axis represents the total amount of energy used up to that point, and the y axis represent the amount of available energy. Now, for every $t > 1$, we want to build the slice polygon. Before doing so, we calculate the minimum and maximum amounts of energy to spend to reach Te_{min} (Line 11) and Te_{max} (Line 12) respectively, depending on the initial temperature. The polygon will be an hexagon (Line 15, whose vertices represent configurations where minimum and maximum temperature is reached respectively, depending on the amount of energy previously spent. After the polygon is added to the HFO, we calculate the new values for the x-axis coordinates (Lines 16-19).

3.3.2 Heat pump operation. The main difference of this use case with respect to other devices, e.g., batteries, is that SG-Ready heat pumps cannot modulate their amount of energy consumed freely, but have to operate in one of the modes described in Section 2.1. This is simple for HFOs with constant power, as they can always operate in *Normal* mode. However, HFOs using the *Optimal* power curve have to alternate between *Off*, *Normal* and *Forced On* modes. Heat pumps typically run at most 4 cycles per hour [20]. In order to be compliant to the SG-Ready standards, our model needs to i) take into account operation according to the SG-Ready modes, and ii) respect the at most 4 cycles per hour constraint for the number of cycles. In this subsection we will show how we achieve i), while compliance to ii) will be shown in Section 4.1. Demonstrating this with an actual heat pump will be addressed as future work.

Algorithm 1: Generating an HFO

Input: T - time horizon;
(Te_{min}, Te_{max}) - temperature constraints;
Te_0 - starting temperature
Output: *HFO* - An HFO

1 **Function** *generateHFO*(Te_{min}, Te_{max},Te_0)**:**
2 $HFO \leftarrow []$
3 **for** $t \leftarrow 1 : T$ **do**
4 **if** $t = 1$ **then**
5 $E^1_{min} \leftarrow Q_{min}(Te_0)$; $E^1_{max} \leftarrow Q_{max}(Te_0)$
6 $HFO.append([E^1_{min}, E^1_{max}])$
7 $E^2_{min} \leftarrow E^1_{min}$; $E^2_{max} \leftarrow E^1_{max}$
8 $S^1_{min} \leftarrow E^1_{min}$; $S^1_{max} \leftarrow E^1_{max}$
9 $S^2_{min} \leftarrow E^2_{min}$; $S^2_{max} \leftarrow E^2_{max}$
10 **else**
11 $E^1_{min} \leftarrow Q_{min}(Te_{min})$; $E^1_{max} \leftarrow Q_{max}(Te_{min})$
12 $E^2_{min} \leftarrow Q_{min}(Te_{max})$; $E^2_{max} \leftarrow Q_{max}(Te_{max})$
13 $HFO.append(Convex[(S^1_{min}, E^1_{min}),$
14 $(S^1_{min}, E^1_{max}), (S^2_{min}, E^1_{max}), (S^1_{max}, E^2_{max}),$
15 $(S^1_{max}, E^2_{min}), (S^2_{max}, E^2_{min})])$
16 $S^1_{min} \leftarrow \max\{S^1_{min} + E^1_{min}\}$
17 $S^2_{min} \leftarrow \max\{S^1_{max} + E^2_{min}\}$
18 $S^1_{max} \leftarrow \max\{S^1_{max} + E^2_{max}\}$
19 $S^2_{max} \leftarrow \max\{S^1_{min} + E^1_{max}\}$
20 **return** *HFO*

Let us say we issued an HFO for a time horizon T, i.e., for the next T time slices. For each time slice $t \in \{1, \ldots, T\}$, an instruction $I_t = (t_{crit1}, t_{crit2})$ is coded by the two numbers t_{crit1}, t_{crit2}. Within the considered time slice, we will count time starting from 0 and ending at t_1. The heat pump will switch modes as follows: *Off* in the interval $[0, t_{crit1})$, *Normal* in the interval $[t_{crit1}, t_1 - t_{crit2})$, *Forced On* in the interval $[t_1 - t_{crit2}, t_1]$. Algorithm 2 describes how to convert an HFO schedule into instructions for the heat pump. We will use two functions: one is t_{crit2Q}, whose aim is to estimate t_{crit2} not knowing the target temperature, but having information on Q and t_{crit1}. It is expressed by the formula

$$t_{crit2Q}(Q, t_{crit1}) = \frac{Q - A \cdot c_{ht} \cdot (Te_{min} - Te_{out}) \cdot t_1 - t_{crit1}}{P_{max} - A \cdot c_{ht} \cdot (Te_{min} - Te_{out})}.$$

The other is Te_{est}, which aims to estimate the final temperature after the instruction *Forced On*, depending on how much time it will

Algorithm 2: Generating SG-Ready modes from HFOs

Input: T - time horizon;
t_1 - time slice duration;
$(Q_1, \ldots, Q_T)$ - energy schedules;
Te_0 - starting temperature
Output: $I = (I_1, \ldots, I_T)$ - SG-Ready modes encodings

1 **Function** *SG-Ready modes*(Te_{min}, Te_{max},Te_0)**:**
2 $I \leftarrow []$
3 **for** $t \leftarrow 1 : T$ **do**
4 **if** $t = 1$ **then**
5 $t_{crit1} \leftarrow t_{crit1}(Te_0)$
6 **else**
7 $t_{crit1} \leftarrow t_{crit1}(Te_e)$
8 $t_{crit2} \leftarrow t_{crit2Q}(Q_t, t_{crit1})$
9 $I_t \leftarrow [t_{crit1}, t_{crit2}]$
10 $I.append(I_t)$
11 $Te_e \leftarrow Te_{est}(t_{crit2})$
12 **return** I

operate (i.e., depending on t_{crit2}). This function is the same as Eq. 11 calculated for $Te_0 = Te_{min}$; more precisely, $Te_{est}(t_{crit2}) = Te_{min} + \left(Te_{out} + \frac{P_{max}}{A \cdot c_{ht}} - Te_{min}\right) \cdot \left(1 - e^{-\frac{A \cdot c_{ht}}{c \cdot m} \cdot t_{crit2}}\right)$. The algorithm operates as follows. First, we initialize the instruction vector (Line 2). Then, for every time slice t, we calculate t_{crit1} with the usual formula on Te_0 if $t = 1$ (Line 5), or on an estimated value Te_e otherwise (Line 7), and we use this value to calculate t_{crit2} using the function t_{crit2U} (Line 8). After that, we build the t-th instructions vector (Line 9) and we add it to the output vector (Line 10). Finally, we calculate the estimated temperature at the beginning of the $t + 1$-th time slice by the function Te_{est} (Line 11).

3.4 (Dis)aggregation and optimization

As explained in Section 2.5, the aggregator receives the FOs from the prosumer and processes them. The main processes are aggregation, optimization and disaggregation. Given N FOs, aggregating them means combining their flexibitity by generating $M \ll N$ new FOs which can capture the total flexibility of the initial N FOs, although with some losses. Aggregation of DFOs has already been described in [35]. The idea is to aggregate each corresponding slice: for example, the first slice of the aggregated DFO is obtained from the first sliced of each of the N DFOs, and so on. The slices are aggregated by dividing the x-axis interval into many points, summing the minimum and maximum possible energy values of each DFO at each of those points, and then taking the convex hull of the points found this way. The energy constraints modeled by HFOs behave in the same way as DFOs: for each time slice, they specify a minimum and a maximum amount or energy that can be consumed, depending on the amount of energy consumed beforehand. For this reason, HFO can be aggregated in the same way as DFOs. In this paper, we propose another way to aggregate HFOs, which exploits the fact that they have a similar shape. This is described in Algorithm 3. We are assuming that all the correspondng HFO slices have the same shape (which is true seeing how HFOs are generated), and that the vertices of the polygons in those slices are ordered in the same way

Algorithm 3: HFO aggregation

Input: $HFO_1, \ldots, HFO_N$ - HFOs
Output: HFO_A - An aggregated HFO

1 **Function** *aggregateHFO*($HFO_1, \ldots, HFO_N$)**:**
2 $HFO_A \leftarrow []$
3 **for** $t \leftarrow 1 : T$ **do**
4 **if** $t = 1$ **then**
5 $E_{min} \leftarrow 0$; $E_{max} \leftarrow 0$
6 **for** $n \leftarrow 1 : N$ **do**
7 $E_{min} \leftarrow E_{min} + DFO_n.V(1)$
8 $E_{max} \leftarrow E_{max} + DFO_n.V(2)$
9 $HFO_A.append([E_{min}, E_{max}])$
10 **else**
11 $V_1, \ldots, V_6 \leftarrow (0, 0)$
12 **for** $n \leftarrow 1 : N$ **do**
13 **for** $v \leftarrow 1 : 6$ **do**
14 $V_v \leftarrow V_v + HFO_n.V(v)$
15 $HFO_A.append(Convex[V_1, \ldots, V_6])$
16 **return** HFO_A

for all the HFOs. First, we initialize the HFO (Line 2). Then, for $t = 1$, we define the lower and upper constraint for energy as the sum of the lower and upper constraints of each HFO (Lines 5-8), and we add this slice to HFO_A (Line 9). Then, for $t > 1$, we instantiate six vertices $V_1, \ldots, V_6$ (Line 11): all the HFOs will have six vertices (for 2, some are counted twice and there are actually only four, but the construction holds anyway), and we sum the coordinates of the first vertex of each HFO to V_1, the coordinates of the second vertex of each HFO to V_2, and so on (Line 14). Finally, we add the convex hull of the polygon obtained by $V_1, \ldots, V_6$ to HFO_A (Line 15). Repeating the procedure until $t = T$, we build all the slices of HFO_A.

FOs can also be used to solve optimization problems. Solving an optimization problem means finding the minimum of a certain objective function over the energy variables e_t, under a specified set of constraints. In our case, we will use energy cost as the objective function. Using an FO to solve the problem means that the FO defines the constraints over the energy variables: for example, if we optimize a function using the FO defined in Figure 2, the constraints will be $0.324kWh \leq e_t \leq 0.396kWh$ for $t \in \{1, \ldots, 8\}$. Once the optimization problem is solved, a schedule is produced, given by the values $(s_1, \ldots, s_T)$ for the energy variables for which the function reaches the minimum. This usage of FOs allows to solve several optimization problems, such as cost minimization [23], consumption-production balancing [35] and discomfort reduction [14].

When aggregated FOs are optimized and a schedule is produced, the schedule must be divided in many smaller schedules, which will be dispatched to the prosumers (in our case, each heat pump) who generated the aggregated FOs. This process is called *disaggregation*. The DFO disaggregation algorithm is based on the principle of dividing energy amounts between the devices proportionally to the amount of minimum and maximum energy that each of them can provide at each time slice. This is explained in Algorithm 4. We want to disaggregate a schedule s^a for the aggregated HFO called HFO_A into many schedules, $s^1, \ldots, s^N$, for the original HFOs, called

Algorithm 4: HFO disaggregation

Input: $HFO_1, \ldots, HFO_N$ - HFOs;
HFO_A - An aggregated HFO;
$s^a = (s^a_1, \ldots, s^a_T)$, a schedule for HFO_A.
Output: $s^1, \ldots, s^N$ - schedules

1 **Function** *disaggregateHFO*($HFO_1, \ldots, HFO_N, HFO_A, s^a$)**:**
2 $sum \leftarrow 0$
3 **for** $t \leftarrow 1 : T$ **do**
4 $slice \leftarrow HFO_A.slice(t)$
5 $x_t \leftarrow \frac{sum - min(slice.x)}{max(slice.x) - min(slice.x)}$
6 $y_t \leftarrow \frac{s^a_t - f^{min}_{slice}(sum)}{f^{max}_{slice}(sum) - f^{min}_{slice}(sum)}$
7 **for** $n \leftarrow 1 : N$ **do**
8 $slice_n \leftarrow HFO_n.slice(t)$
9 $x \leftarrow x_t \cdot max(slice_n.x) + (1 - x_t) \cdot min(slice_n.x)$
10 $y \leftarrow y_t \cdot f^{max}_{slice_n}(x) + (1 - x) \cdot f^{min}_{slice_n}(x)$
11 $s^n_t \leftarrow y$
12 $sum \leftarrow sum + s^a_t$
13 **return** $s^1, \ldots, s^N$

$HFO_1, \ldots, HFO_N$. We will make use of two functions, $f^{min}_{slice}(x)$ and $f^{max}_{slice}(x)$ which, given an HFO slice, determine the minimum and maximum amount of energy consumable within that slice, provided that the amount of energy consumed up to that point in time is x.

For every $t \in \{1, \ldots, T\}$, we first extract the corresponding HFO_A slice (Line 4) and, denoting by *sum* the partial sum of s^a up to that point, we establish the relative horizontal position of *sum* in the polygon (Line 5). We then use the f^{min} and f^{max} functions to determine the relative vertical position of s^a_t at that specific point of the polygon (Line 6). Now, for every HFO_n, we identify which point of the t-th slice corresponds to the point found for s^a_t (Lines 8-10): this point will become s^n_t, i.e., the t-th element of the n-th schedule. After this, we update the sum of energy used in HFO_A (Line 12), and continue until all the schedules s^n are complete.

3.5 Heat-electricity conversion

HFOs represent electricity-heat flexibility: the input energy vector is electricity and the output energy vector is heat, as HFOs are issued for devices who convert electricity into heat, such as heat pumps. Therefore, it is natural to conclude that HFOs can be *reversed*, in order to represent heat as input and electricity as output. In other words, given an HFO describing the flexibility of a heat pump in terms of heat, it is possible to generate a DFO which represents the equivalent flexibility of the heat pump in terms of consumed electricity, and vice versa. This can be achieved easily, as shown in Algorithm 5. The algorithm works as follows. First, we initialize the DFO (Line 2). At each time t, we extract the corresponding HFO slice (Line 4) and the vertices $V_1, \ldots, V_K$ of the polygon that defines that slice (Line 5). Now, we define a new set of points $V^D_1, \ldots, V^D_K$ in the following way: if the coordinates of V_k are (a, b), the coordinates of V^D_k will be $(\frac{a}{COP}, \frac{b}{COP})$ (Line 7). After this, the convex hull of the points $V^D_1, \ldots, V^D_K$ is the slice of the resulting DFO (Line 8). The procedure is then repeated for all the remaining time slices,

Algorithm 5: HFO-DFO conversion

Input: *HFO* - an HFO;
COP - Coefficient of performance
Output: *DFO* - a DFO

1 **Function** *convertHFO(HFO,COP)*:
2 $DFO \leftarrow []$
3 **for** $t \leftarrow 1 : T$ **do**
4 $slice \leftarrow HFO.slice(t)$
5 $V_1, \ldots, V_K \leftarrow slice.Vertices$
6 **for** $k \leftarrow 1 : K$ **do**
7 $V^D_k \leftarrow \frac{V_k}{COP}$
8 $DFO.append(Convex[V^D_1, \ldots, V^D_K])$
9 **return** *DFO*

Value	Single room scenario	Aggregation scenario
$A(m^2)$	12	12 and 15
$P_{max}(kW)$	4.6	4.6 and 3.2
$Te_{min}(K)$	298	298 and 295
$Te_{max}(K)$	302	302 and 299
$Te_{out}(K)$	280	280 and 284
COP	3.6	3.6 and 3.53
T	12	12 to 20
NHP	1	40 to 100

Table 3: Parameter values used for the experiments

until all the slices of the DFO are generated. It is of course possible to convert an electricity DFO for a heat pump to an equivalent HFO, and the procedure is the reverse as Algorithm 5: in this case the DFO is the input, the HFO is the output, and COP is replaced by COP^{-1}. This way, if an HFO is converted to a DFO and then back to an HFO again, the resulting HFO will be the initial one. Conversion of energy schedules is also easy: given a schedule s for electricity consumption, the equivalent schedule in terms of heat is obtained by multiplying each element of s by COP. In general, energy conversion from one energy vector to another is needed if the primary flexibility comes from another energy vector, e.g., heat or gas, through explicit or implicit storage. For example, a heated room like in our running example corresponds to implicit heat storage. In this case, conversion is needed because flexibility is traded in the electricity market, and therefore if it originates from another energy vector, it has to be converted before it can be sold. We have described the procedure to create and convert HFOs for heat pumps: the procedure for other heat devices, such as HVAC and thermal energy storages, is similar to the one shown in this paper. It is also possible to extend this model to other energy vectors such as gas, but this is beyond the scope of this work.

4 SIMULATIONS AND RESULTS

In order to validate the effectiveness of our algorithms, we have run experiments and measured the amount of flexibility. All the experiments in this section have been coded in *MATLAB* R2020b, and use the *YALMIP* and *mpt3* optimization toolbox. However, they are all software simulations: experiments with a real heat pump will

belong to future work. All the methods, also the existing baselines, have been implemented by us and simulations have been performed: code can be found at https://github.com/FabioLilliu/FlexOffers. Data for spot and imbalance prices have been taken from a sample in NordPool. Throughout this section we will have the objective of maximizing the profit function, which is defined as $PF(e) = -Spr \cdot e$. Here, $e = (e_1, \ldots, e_T)$ is the energy consumed at each time $1, \ldots, T$, and $Spr = (Spr_1, \ldots, Ipr_T)$ are the spot prices at time $1, \ldots, T$. We also denote the imbalance prices by $Ipr = (Ipr_1, \ldots, Ipr_T)$.

4.1 Single heat pump optimization

In the first part of this section we describe profits and imbalance for single devices. However, what actually happens is that the aggregator combines their flexibility models (FOs), as described in Section 2.5). The process of flexibility bidding happens at the aggregator's level, and it is the aggregator who obtains profit from selling flexibility, and is penalized when inaccurate flexibility predictions cause imbalance. However, since the flexibility profile of the aggregator is obtained by combining flexibility profiles of many loads, we will consider the aggregator's profits and imbalance penalties as *split* between the aggregated loads, and the profits/imbalance costs relative to a single load will refer to their specific fraction of the aggregator's. As we are considering the day-ahead spot market, we will consider all the flexibility operations to happen more than 12 hours in advance with respect to the beginning of the day for which flexibility is represented. Finally, spot prices are known in advance, while imbalance prices have to be forecast. It is important to specify that those are the electricity market prices: for this reason, even if we are working with HFOs, from now on we will implicitly assume that right before optimization HFOs are converted into equivalent electricity DFOs, and the resulting schedules back to HFO schedules. We simulate the scenario of a heat pump heating a room: we call this the **Single room scenario**. In this case, the room and heat pump are as described in Section 2.3, except for Te_{min} and Te_{max}; see Table 3 for a complete list of the parameters. We choose a time horizon T, we generate an FO for the next T time units, and we optimize it with the objective to maximize the profit function PF, where the constraints on e are defined by the FO. We then check whether the schedules obtained by the optimization violate the constraints on heat pump power and minimum and maximum temperature; if yes, we calculate the imbalance penalties as the minimum possible cost of the difference between the schedule and a feasible one. This cost is calculated as $Cost(e) = Ipr \cdot e + disc(Te)$, where $disc$ is a discomfort function depending on the temperature Te. We then repeat this procedure for the next T time units again and again, until the simulation covers a total of 365 days. We have run this experiment for the HFOs generated both from the *Constant* (**HFO-Constant**) and the *Optimal* (**HFO-Optimal**) power curves, and for a theoretical optimum approximation for both *Constant* and *Optimal* power curves, which we will denote by **Theo.Opt.-C** and **Theo.Opt-O** respectively. The theoretical optimum approximation enforces the exact constraints on temperature and power, and it is the most possible accurate approximation; however, it is not exact since the initial temperature Te_0 of the room has to be predicted.

Table 4 describes the results for a single heat pump, choosing a time horizon of $T = 12$. The table describes respectively: the **Type** of approximation considered, the total cost before applying imbalance penalties (**Pre-Imb**), the amount of imbalance penalties (**Imb.Pen.**) and the total cost after applying imbalance penalties (**Cost**). We can see that HFO obtained from the *Constant* power curve are able to retain a large amount of flexibility, and their cost is only 1.6% higher compared to **Theo.Opt.-C** Defining retained flexibility as the ratio between **Theo.Opt.** cost and the **HFO-Constant** cost, we have that 98.4% of the total flexibility is retained. HFOs obtained from the *Optimal* approach have a cost only 1.1% higher compared to **Theo.Opt.-C**, so they can retain 98.9% of the total flexibility.

Type	Pre-Imb(€)	Imb.Pen.(€)	Cost(€)
Theo.Opt.-C	131.86	0	131.86
HFO-Constant	130.75	3.21	133.96
Theo.Opt.-O	131.83	0	131.83
HFO-Optimal	130.33	2.99	133.32

Table 4: Results for single heat pump cost optimization.

	0	1	2	3	4+
Hours	8112	333	273	42	0

Table 5: Frequency of number of mode changes per hour

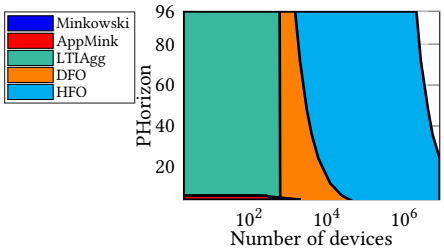

Figure 4: Feasibility and costs for time horizons and devices

During the same experiment, we also tracked the total number of SG-Ready mode changes that the heat pump undergoes each hour. Table 5 describes how many hours has the heat pump spent (out of a total of 8760 in a year) with its mode changing 0, 1, 2, 3 or 4+ times. As we can see, the heat pump never changes mode more than 3 times per hour. In particular, the heat pump does not change its mode within the hour for 8112 hours, 92.6% of the total. There are 333 hours in which the heat pump changes mode once, 273 hours in which it changes mode twice, and only 42 hours (less than 0.5% of the total) in which the heat pump changes mode three times. Thus, our HFO optimization respects the constraints in [20].

4.2 Aggregation and disaggregation

We have also run experiments in order to evaluate the effectiveness of HFO aggregation. We will denote by NHP the number of heat pumps that are aggregated: $\frac{NHP}{2}$ of them are copies of the heat pump described in the running example, in the same scenario. The other $\frac{NHP}{2}$ heat pumps are copies of a *Toshiba RAS-B13* heat pump, with maximum power $P_{max} = 3.2$ kW, which is compatible with the SG-Ready interface, and the considered scenario is a room with

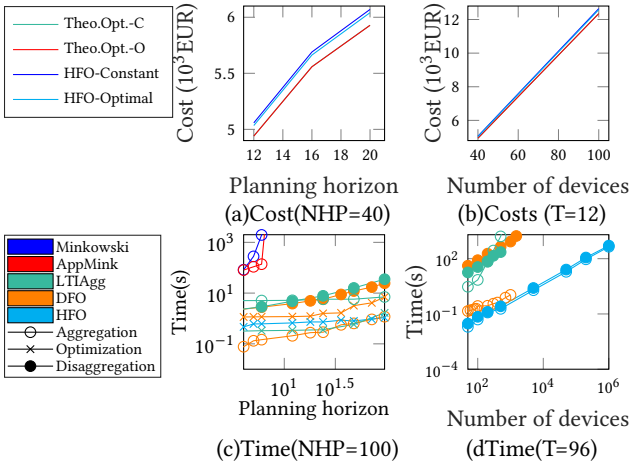

Figure 5: Feasibility and costs for time horizons and devices

section 5m x 5m, and 3m tall, with Te_{min} = 295K, Te_{max} = 299K, and outer temperature Te_{out} = 284K. The baseline approaches against whom we compare it are: **DFO aggregation**, which we introduced in Section 3, **Minkowski**, an approximated Minkowski sum [2] (**AppMink**), and an aggregation approach specific for LTI models, called LTI Aggregation [23] (**LTIAgg**). The **Minkowski** approach performs the Minkowski sum of energy polygons, and the concept for **AppMink** is to approximate the Minkowski sum, obtaining faster aggregation time in exchange for that. Finally, **LTIAgg** creates a *NHP*-dimensional LTI model from the *NHP* 1-dimensional models that are aggregated. Economic revenue is the metric by which we measure the amount of retained flexibility. The experiment has been performed as follows. Given a time horizon T, we generate for each heat pump an LTI model for the *Constant* power curve, an **HFO-Constant** and an **HFO-Optimal**. We then aggregate these flexibility models using the proposed approaches, and optimize them in order to minimize the cost function *PF*. We then calculate imbalance in the same way as the single heat pump experiment, and we assign the schedules to each device through disaggregation. After that, the process is repeated over and over, for the duration of 365 days. We refer to this scenario as the **Aggregation scenario**: Table 3 shows the values used for the parameters in the experiments. In order for an aggregation approach to be considered *feasible*, the operations of aggregation, optimization and disaggregation combined should not take more than 30 minutes. The reason for that is that flexibility bids should be done not more than one hour before the deadline, for the purpose of reducing forecasting [26], and before aggregation flexibility models need also to be instantiated. Figure 4 shows the results for feasibility: the graph shows for how many devices and which time horizons each approach is feasible, with a logarithmic scale on the x axis. As we can see, **HFO aggregation** vastly outperforms all other approaches, being able to aggregate, optimize and disaggregate up to $2 \cdot 10^6$ devices for a time horizon of $T = 96$. In comparison, **DFO** aggregation can only aggregate 1500 devices for $T = 96$, and **LTIAgg** is capable of aggregating only 500 devices, while **Minkowski** and **AppMink** fail already for $T > 6$. The results for profit can be seen in Figures 5(a-b) : they show respectively how the profit increases with respect to T (in Figure 5(a)) and *NHP* (in Figure 5(b)). We are comparing the same approaches we introduced for single heat pump cost optimization. Note that in Figures 5(a-b), the lines for **Theo.Opt.-C** and **Theo.-Opt.-O** are very close to each other and may be hard to distinguish, and the same is true for **HFO-Constant** and **HFO-Optimal**. **Theo.Opt.-C** and **Theo-Opt.-O** are aggregated with the **LTIAgg** approach, while **HFO-Constant** and **HFO-Optimal** are aggregated by the **HFO** approach. Even in this case, we have that aggregation of **HFO-Constant** FOs does not lose much flexibility, as for $T = 12$ the cost is only 2.4% higher compared to **Theo.Opt.-C** and the amount of retained flexibility is 97.7%. This remains true for higher values of *NHP* and T, with minor further losses for high values of T. **HFO-Optimal** FOs retain 98.1% of the flexibility compared to **Theo.Opt.-O**, and this still holds for higher values of *NHP* and T, with minor further losses. Regarding aggregation and disaggregation time, Figure 5 (c) shows the results with respect to T, and Figure 5 (d) with respect to *NHP*. We can see that **Minkowski** and **AppMink** have aggregation times much higher than the other approaches, and become unfeasible for short time horizons. **LTIAgg**, **DFO** and **HFO** grow linearly with respect to T for both aggregation and disaggregation, and **DFO** and **HFO** grow linearly with respect to *NHP* for both aggregation and disaggregation, while **LTIAgg** grows exponentially for aggregation, making it unfeasible for high values of *NHP*.

5 CONCLUSIONS AND FUTURE WORK

The rationale for this paper is to create a flexibility model able to i) represent flexibility from different types of devices; ii) optimize, aggregate and disaggregate flexibility in a scalable way, for many loads and long time horizons; iii) capture almost all of the available flexibility; iv) capture flexibility for different energy vectors, and support conversion among those. We focus in particular on heat, and propose Heat FlexOffers (HFOs), a model based on FlexOffers (FOs) that extends the FO model and enables it to handle energy conversion. This paper describes how HFOs are generated, how they can support different types of continuous power curves despite being a discrete model, how they can be aggregated and disaggregated, and how they comply with the SG-Ready interface. HFOs represent flexibility in a device-independent way, which satisfies i). It has been shown that HFOs can be aggregated, optimized and disaggregated, and this process can be done in at most 30 minutes for $2 \cdot 10^6$ HFOs, while exact baselines fail for more than 500 loads, which accomplishes ii). HFOs are also very accurate, accomplishing iii): they can capture up to 98.9% of total flexibility before aggregation, and 98.1% after aggregation compared to the exact baselines. Finally, iv) is satisfied as HFOs can represent flexibility for heat, and can be converted easily to Dependency FlexOffers (DFOs) for electricity. Future work will regard extension of FOs to different energy vectors, and improvement of the analytic generation algorithms.

ACKNOWLEDGMENTS

This work was supported by the DomOS and FEVER projects, funded under the Horizon 2020 programme, GAs 864537 and 894240 respectively, and Flexible Energy Denmark, funded by Innovation Fund Denmark (Case No. 8090-00069B).

REFERENCES

[1] Poria Astero and Corentin Evens. 2021. Optimum day-ahead bidding profiles of electrical vehicle charging stations in FCR markets. *Electric Power Systems Research.*

[2] Suhail Barot and Josh A. Taylor. 2016. An outer approximation of the Minkowski sum of convex conic sets with application to demand response. *IEEE 55th Conference on Decision and Control.*

[3] Andreas Bloess, Wolf-Peter Schill, and Alexander Zerrahn. 2018. Power-to-heat for renewable energy integration: A review of technologies, modeling approaches, and flexibility potentials. *Applied Energy.*

[4] Jonas Brusokas, Torben Bach Pedersen, Laurynas Siksnys, Dalin Zhang, and Kaixuan Chen. 2021. HeatFlex: Machine learning based data-driven flexibility prediction for individual heat pumps. In *e-Energy.*

[5] Lu Chen, Qingshan Xu, Yongbiao Yang, and Jing Song. 2021. Optimal energy management of smart building for peak shaving considering multi-energy flexibility measures. *Energy and Buildings.*

[6] Yongbao Chen, Peng Xu, Jiefan Gu, Ferdinand Schmidt, and Weilin Li. 2018. Measures to improve energy demand flexibility in buildings for demand response (DR): A review. *Energy and Buildings.*

[7] Edoardo Corsetti, Shariq Riaz, Marco Riello, and Pierluigi Mancarella. 2021. Modelling and deploying multi-energy flexibility: The energy lattice framework. *Advances in Applied Energy.*

[8] Cherrelle Eid, Paul Codani, Yurong Chen, Yannick Perez, and Rudi Hakvoort. 2015. Aggregation of Demand Side flexibility in a Smart Grid: A review for European Market Design. *12th International Conference on the European Energy Market (EEM).*

[9] David Fischer, Marc-Andre Triebel, and Oliver Selinger-Lutz. 2018. A Concept for Controlling Heat Pump Pools Using the Smart Grid Ready Interface. In *IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe).*

[10] David Fischer, Tobias Wolf, and Marc-André Triebel. 2017. Flexibility of heat pump pools: The use of SG-Ready from an aggregator's perspective. In *12th IEA Heat Pump Conference.*

[11] David Fischer, Tobias Wolf, Jeannette Wapler, Raphael Hollinger, and Hatef Madani. 2017. Model-based flexibility assessment of a residential heat pump pool. *Energy.*

[12] R. Fonteijn, T. Van Cuijk, P.H. Nguyen, J. Morren, and J.G. Slootweg. 2018. IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe). In *ISGT-Europe.*

[13] Kyriaki Foteinaki, Rongling Li, Thibault Péan, Carsten Rode, and Jaume Salom. 2020. Evaluation of energy flexibility of low-energy residential buildings connected to district heating. *Energy and Buildings.*

[14] Davide Frazzetto, Bijay Neupane, Torben Pedersen, and Thomas Nielsen. 2018. Adaptive User-Oriented Direct Load-Control of Residential Flexible Devices. *e-Energy.*

[15] Reza Ghaemi, Aditya Kumar, Pierino Bonanni, and Nikita Visnevski. 2020. Scalable Optimal Flexibility Control, modeling and estimation of commercial buildings. In *American Control Conference (ACC).*

[16] Irina Harris, Christine L. Mumford, and Mohamed M. Naim. 2014. A hybrid multi-objective approach to capacitated facility location with flexible store allocation for green logistics modeling. *Transportation Research Part E: Logistics and Transportation Review.*

[17] S.E.G. Jayamaha, N.E. Wijeysundera, and S.K. Chou. 1996. Measurement of the heat transfer coefficient for walls. *Building and Environment.*

[18] Rune Grønborg Junker, Armin Ghasem Azar, Rui Amaral Lopes, Karen Byskov Lindberg, Glenn Reynders, Rishi Relan, and Henrik Madsen. 2018. Characterizing the energy flexibility of buildings and districts. *Applied Energy.*

[19] Rune Grønborg Junker, Carsten Skovmose Kallesøe, Jaume Palmer Real, Bianca Howard, Rui Amaral Lopes, and Henrik Madsen. 2020. Stochastic nonlinear modelling and application of price-based energy flexibility. *Applied Energy.*

[20] Michael Kummert and Michel Bernier. 2008. Sub-hourly simulation of residential ground coupled heat pump systems. *Building Services Engineering Research & Technology.*

[21] Soumya Kundu, Karanjit Kalsi, and Scott Backhaus. 2018. Approximating Flexibility in Distributed Energy Resources: A Geometric Approach. *Power Systems Computation Conference (PSCC).*

[22] Han Li, Zhe Wang, Tianzhen Hong, and Mary Ann Piette. 2021. Energy flexibility of residential buildings: A systematic review of characterization and quantification methods and applications. *Advances in Applied Energy.*

[23] Fabio Lilliu, Torben Bach Pedersen, and Laurynas Siksnys. 2021. Capturing Battery Flexibility in a General and Scalable Way Using the FlexOffer Model. *SmartGridComm.*

[24] Fabio Lilliu, Torben Bach Pedersen, Laurynas Siksnys, and Bijay Neupane. 2023. Uncertain flexoffers, a scalable, uncertainty-aware model for energy flexibility. In *e-Energy.*

[25] Rui Amaral Lopes, Adriana Chambel, João Neves, Daniel Aelenei, and João Martins. 2016. A Literature Review of Methodologies Used to Assess the Energy Flexibility of Buildings. *Energy Procedia.*

[26] Meiqin Mao, Shengliang Zhang, Liuchen Chang, and Nikos D. Hatziargyriou. 2019. Schedulable capacity forecasting for electric vehicles based on big data analysis. *Journal of Modern Power Systems and Clean Energy.*

[27] Ilaria Marotta, Francesco Guarino, Maurizio Cellura, and Sonia Longo. 2021. Investigation of design strategies and quantification of energy flexibility in buildings: A case-study in southern Italy. *Journal of Building Engineering.*

[28] Alice Mugnini, Gianluca Coccia, Fabio Polonara, and Alessia Arteconi. 2021. Energy Flexibility as Additional Energy Source in Multi-Energy Systems with District Cooling. *Energies.*

[29] Fabian L. Müller, Jácint Szabó, Olle Sundström, and John Lygeros. 2019. Aggregation and Disaggregation of Energetic Flexibility From Distributed Energy Resources. *IEEE Transactions on Smart Grid.*

[30] Bijay Neupane, Laurynas Siksnys, and Torben Bach Pedersen. 2017. Generation and Evaluation of Flex-Offers from Flexible Electrical Devices. *e-Energy.*

[31] Ivan Pavić, Tomislav Capuder, and Hrvoje Pandžić. 2022. Analysis of aFRR and mFRR Balancing Capacity & Energy Demands and Bid Curves. In *IEEE 7th International Energy Conference (ENERGYCON).*

[32] Torben Bach Pedersen, Laurynas Siksnys, and Bijay Neupane. 2018. Modeling and Managing Energy Flexibility Using FlexOffers. *SmartGridComm.*

[33] Fabian Scheller, Robert Burkhardt, Robert Schwarzeit, Russell McKenna, and Thomas Bruckner. 2020. Competition between simultaneous demand-side flexibility options: The case of community electricity storage systems. *Applied Energy.*

[34] Paul Schott, Johannes Sedlmeir, Nina Strobel, Thomas Weber, Gilbert Fridgen, and Eberhard Abele. 2019. A Generic Data Model for Describing Flexibility in Power Markets. *Energies.*

[35] Laurynas Siksnys and Torben Bach Pedersen. 2016. Dependency-based FlexOffers: scalable management of flexible loads with dependencies. *e-Energy.*

[36] Hanne Sæle, Andrei Morch, Merkebu Zenebe Degefa, and Irina Oleinikova. 2020. Assessment of flexibility in different ancillary services for the power system. In *17th International Conference on the European Energy Market (EEM).*