Scheduling Delays and Curtailment for Household Appliances with Deterministic Load Profiles using MPC

Yingzhao Lian, Yuning Jiang*, Colin N. Jones, and Daniel F. Opila

Abstract—Smart home appliances can time-shift and curtail their power demand to assist demand side management or allow operation with limited power, as in an off-grid application. This paper proposes a scheduling process to start appliances with time-varying deterministic load profiles. Self-triggered model predictive control is used to limit the household net power demand below a given threshold. Meanwhile, deterministic load profiles are more difficult to schedule compared to variable charging or thermal loads because system failure will occur once power demand is not satisfied. The proposed scheme formulates the decision of the load shifting time as a continuous optimization problem, and an inhomogeneous time grid system is introduced to handle the optimization of different appliances and their consensus at this resolution. The efficacy of the proposed scheme is studied by numerical comparison with a mixed-integer MPC controller and by a case study of three home appliances and an interruptible washing machine.

Index Terms—Load scheduling, home automation, interruptible load, smart grid, self-triggered model predictive control

I. INTRODUCTION

The residential sector is a large energy consumer, accounting for 28% of the total energy consumption in Europe, and scheduling this load plays a key role in demand side management (DSM) [1]. DSM is an arrangement of actions to modify the power demand on the user side, and load scheduling for home appliances has experimentally shown a peak load reduction up to 40% [2]. Thus, this method can improve conditions in areas suffering from power shortages, such as Pakistan [3] and Lebanon [4]. Load scheduling of intelligent home appliances is popular even aside from power considerations, based on an online survey conducted in Europe [5]. In particular, 50% of the participants preferred to use the scheduling function, 29% of them mentioned a reduced electricity tariff, and 7% got a stability improvement in home power management.

Motivated by these benefits, the load scheduling problem has been widely studied, and model predictive control (MPC) is an attractive solution because it can enforce explicit constraint satisfaction by synthesizing the system behavior forecast recursively [6]. The control performance is varied by the design of the objective, which can reflect the occupants' comfort and operating time preference [7], [8].

In this work, we consider loads whose demand follows a specific time-varying profile once started, termed a "de-terministic" power demand profile. Unlike the continually variable loads often considered in existing work, such as vehicle charging [9] or an averaged on-off thermal loads [10], the loads considered here can be delayed by a time shift, curtailed, or interrupted at specific break points, but their demand follows the profile when running. Among these actions, time shifts pose the major challenge, which makes the time of different loads not aligned and therefore nonlinear. To address this problem, most previous results formulate the load profile of an appliance as a discrete-time sequence, where the time shift is an integer-valued decision variable [11]–[14]. Moreover, due to the computational complexity of this mixed-integer problem, most previous work does not consider constraints and load curtailment except for [11].

This paper proposes a new continuous formulation, which is inspired by self-triggered MPC. Self-triggered MPC can systematically optimize the active/triggered time of a device proactively with explicit constraint enforcement [15], and has been widely used in sensor networks and Internet-of-Things [16]. As opposed to the mixed-integer formulation [17], [18], the triggering time can be a continuous decision variable [19]–[21] in this scheme. This paper interprets the time shift as a trigger time viewed from the current time step, and transcribes the load scheduling problem into a continuous optimization problem such that we can take full advantage of continuous numerical solvers.

The contributions are summarized as follows: (a) A continuous formulation of the load scheduling problem with time shifts. This framework is compatible with interruptible loads with explicit break points, and its computational complexity caused by the shifting decision is constant; (b) An inhomogeneous time grid system is introduced to enable control of the computational complexity of each load and constraint; (c) The system demand respects a time-varying power limit, as would occur off-grid or in microgrids, or if the utility controls demand via a power limit signal rather than a price.

The paper is organized as follows: the load scheduling problem and its corresponding optimization problem are introduced in Section II. The numerical implementation into a continuous tractable formulation is introduced in Section III, alongside a comparison with previous mixed integer formulations. The numerical results and a comparison with the mixed-integer formulation are in Section IV, followed by conclusions in Section V.

Notations: $\mathbb{Z}_a^b$ denotes the set of integers ranging from a to b with $\mathbb{Z}$ the set of non-negative integers. $\{x_i\}_{i \in T}$ denotes a set indexed by set T, and the index is dropped when there is no confusion as $\{x_i\}$. $\mathcal{A} \backslash \mathcal{B} := \{x | x \in \mathcal{A},\ x \notin \mathcal{B}\}$.

YL, YJ and CJ are with Automatic Control Laboratory, EPFL. (email: yingzhao.lian, yuning.jiang,colin.jones@epfl.ch)

DO is with United States Naval Academy, Electrical and Computer Engineering Department. (email: opila@usna.edu)

II. PROBLEM STATEMENT

Here we consider the coordination of multiple deterministic loads, appliances for example. This is relevant for varying electricity prices, but is a particular problem when both power and energy are limited as in off-grid operation, a low-power connection, or when responding to a demand limit signal from a grid operator. We consider the case of a dwelling with a battery and a limited external power source like solar, fuel cell, etc. which can reduce its output if needed. Additionally, the primary goal is to ensure feasibility and to minimize delays, in contrast to a typical objective of minimizing energy costs with no limit on power.

Once an appliance is requested to start, it should launch as soon as possible considering the feasible power profile, the total power limit, and total available energy. In the rest of this paper, we use absolute time in all notation.

A. The Load Profile Models

Two different load types are considered: uninterruptible loads, whose execution cannot be paused once started, and interruptible loads with explicit break points. An example for the latter case is a washing machine with a drying function, which can pause after rinsing with a reasonable delay. If we consider an uninterruptible load, indexed by i, launched at time t_{s_i}, its actual power consumption $\overline{u}_i(t)$ is expected to follow a predefined nominal power profile defined by

$$\begin{cases} \overline{u}_i(t), & t \in [t_{s_i}, t_{s_i} + T_{e_i}], \\ 0, & \text{otherwise}, \end{cases} \quad (1)$$

where T_{e_i} is the time span of the execution of i-th load.

Similarly, an interruptible load with explicit break points can be modelled as a sequence of uninterruptible subtasks. In particular, we define $\mathcal{I}_s$, the set that includes all the uninterruptible loads and the first subtask of the interruptible loads. This means that if an index j represents a subtask of an interruptible load but not the first, then we have $j \notin \mathcal{I}_s$, and accordingly we can define a mapping $\mathfrak{p}(j)$ that returns the finishing time of its precedent subtask for all $j \notin \mathcal{I}_s$. Taking a washing machine with a drying function as an example, we can model it as two uninterruptible subtasks indexed by i and j, then we have $i \in \mathcal{I}_s$ and $j \notin \mathcal{I}_s$ such that $\mathfrak{p}(j) = t_{s_i} + T_{e_i}$. Since each subtask must start after the earlier one finishes, $t_{s,j} \geq \mathfrak{p}(j)$ always holds for all $j \notin \mathcal{I}_s$.

B. The Scheduling Strategies

The external power source nominally provides two functions, power to support the instantaneous loads and energy over time to prevent battery discharge. Distributed generation sources are variable and difficult to predict, and we can consider $P_{\min}(t)$ as the minimum power that can be reliably expected at a given time, and P_{avg} as the expected average power over an interval $[t_0, t_0 + T]$. While it is often true that $P_{\text{avg}} > P_{\min}(t)$, the uncertainty limits reliable operation, hence we only consider $P_{\min}(t)$ in this study. The excess power from the external source will be stored in the battery and allow for more power use later without resorting to the grid.

Multiple appliances can run concurrently, and, due to the discussion about $P_{\min}(t)$ and the battery above, the total power consumption from these active appliances are required to be upper bounded by $u_{\max}(t)$ as

$$\sum_{i \in \mathcal{I}(t)} u_i(t) \leq u_{\max}(t) := P_{\text{batt}}(t) + P_{\min}(t), \quad (2)$$

where $\mathcal{I}(t)$ is the set of all active appliances at time t and P_{batt} denotes the power supplied by the battery, whose charge level is modelled by

$$\frac{dE_{\text{batt}}(t)}{dt} = -P_{\text{batt}}(t).$$

The battery capacity is bounded within $[0, E_{\max}]$. Its charging/discharging rate is limited such that $P_{\text{batt}}(t) \in [\underline{P}_{\text{batt}}, \overline{P}_{\text{batt}}]$ with $\overline{P}_{\text{batt}}/\underline{P}_{\text{batt}}$ the maximal dis-/charge rate. It is possible to consider the charging/discharging efficiency based on the method given in [12], but for the sake of clarity we consider an ideal battery. Additionally, the capacity and allowable dis-/charge rate of the battery can affect the performance of the system [22].

To start the appliances as early as possible after their requests without violating this constraint, shifting and curtailment are considered:

1) Load shifting: The execution of multiple loads can be shifted after their request to avoid overlap of the peak power consumption. If an appliance is requested at t_{r_i}, the delay between the request and the actual starting time of the appliance is required to be bounded by Δ_i^d,

$$0 \leq t_{s_i} - t_{r_i} \leq \Delta_i^d.$$

The preference of a responsive execution is modelled by a monotonically increasing loss function $J_i^d(t_{s_i} - t_{r_i})$.

2) Load curtailment: The load applied $u_i(t)$ is allowed to mismatch the nominal load profile $\overline{u}_i(t)$. The resulting curtailment is required to be bounded as

$$\forall\, t \in [t_{s_i}, t_{s_i} + T_{e_i}],\ 0 \leq \overline{u}_i(t) - u_i(t) \leq c_i(t),$$

where $c_i(t)$ is the maximal curtailment at time t. Accordingly, the preference of matching the nominal profile is modelled by objective $J_i^c(\cdot)$, the corresponding accumulated curtailment cost is given by

$$L_i^c(u_i(\cdot), t_{s_i}) := \int_{t_{s_i}}^{t_{s_i} + T_{e_i}} J_i^c(\overline{u}_i(t) - u_i(t)) \mathrm{d}t. \quad (3)$$

Based on the aforementioned scheduling strategies, at time instance t, the home appliances are characterized by two sets, the set of appliances that have been requested to launch, dubbed $\mathcal{I}_r(t)$ and the set of appliances that are executing currently at time t, dubbed $\mathcal{I}(t)$. In particular, $\mathcal{I}_r(t)$ includes all the loads that have been requested to launch and are not yet finished i.e., $(t \geq t_{r_i})$, as well as its subsequent loads if it is part of an interruptible load. Note that a requested load can be delayed, thus we also have $\mathcal{I}(t) \subset \mathcal{I}_r(t)$.

Remark 1 *Note that the proposed scheme can also handle curtailment cost J_i^c as a function of time. This scenario appears when portions of the profile are more critical. Meanwhile, the priority of a specific load can be reflected in the penalty in its corresponding J_i^c and J_i^d.*

C. Load Scheduling MPC

At t_0, given the list of requested loads $\mathcal{I}_r(t_0)$, we can summarize the load scheduling MPC problem as:

$$\underset{t_{s_i}, P_{\text{batt}}(\cdot), u_i(\cdot)}{\text{minimize}} \sum_{i \in \mathcal{I}_r(t_0)} J_i^d(t_{s_i} - t_{r_i}) + L_i^c(u_i(\cdot), t_{s_i}) \quad (4a)$$

subject to

$$\sum_{i \in \mathcal{I}_r(t_0)} u_i(t) \leq P_{\min}(t) + P_{\text{batt}}(t), \ t \in [t_0, t_e], \quad (4b)$$

$$\frac{d\hat{E}(t)}{dt} = -P_{\text{batt}}(t), \ t \in [t_0, t_e], \ \hat{E}(t_0) = E(t_0), \quad (4c)$$

$$\hat{E}(t) \in [0, E_{\max}], \ P_{\text{batt}}(t) \in [\underline{P}_{\text{batt}}, \overline{P}_{\text{batt}}], \quad (4d)$$

$$\begin{cases} 0 \leq \overline{u}_i(t) - u_i(t) \leq c_i(t), & t \in [t_{s_i}, t_{s_i} + T_{e_i}] \\ u_i(t) = 0, & \text{otherwise} \end{cases} \quad (4e)$$

$$0 \leq t_{s_i} - t_{r_i} \leq \Delta_i^d, \ i \in \mathcal{I}_r(t_0). \quad (4f)$$

$$t_{r_i} = \mathfrak{p}(i), \ i \in \mathcal{I}_r(t_0) \backslash \mathcal{I}_s. \quad (4g)$$

The objective penalizes both the delay and the curtailment, and $\hat{E}(\cdot)$ denotes the virtual counterpart of the actual battery $E(\cdot)$ optimized over the prediction horizon $[t_0, t_e]$. Additionally, the total power consumption constraint (4b) is imposed on $[t_0, t_e]$. Therefore, the end of the horizon t_e is required to be later than the finish time of the all the requests. Note that the start times t_{s_i} are decision variables, we therefore choose $t_e \geq \max_{i \in \mathcal{I}_r(t_0)}\{t_{r_i} + T_{d,i} + T_{e_i}\}$. The constraint (4e) ensures that $u_i(t) = 0$ when it is not operating. Finally, for those loads outside $\mathcal{I}_s$ constraint (4g) synthetically enforces that a component of an interruptible load can execute only after its preceding load is finished.

III. NUMERICAL IMPLEMENTATION

Problem (4) is numerically intractable due to the infinite dimensional decision variable $u_i(\cdot)$. This section discusses the transcription of this infinite dimensional problem to a finite dimensional form.

A. Inhomogeneous Time Grids

The source of the computational intractability of (4) is the $u_i(\cdot)$ in continuous time. Therefore, we adopt the idea of direct optimal control [23] that relaxes the constraints (4b)-(4e) to the evaluation on finite time instances. To this end, we introduce the following inhomogeneous time grids:

1) Global time grid: we denote it by $\mathcal{T}_g := \{t_{g,j}\}_{j \in \mathbb{Z}_0^{N_g}}$ with $t_0 = t_{g,0} < t_{g,1} \cdots < t_{g,N_g-1} < t_{g,N_g} = t_e$. This time grid is introduced to evaluate the constraints (4b), (4c) and (4d) point by point. $\mathcal{T}_g$ shifts with respect to the current time instance t_0, and its spacing is user-defined and fixed.

2) Local time grid: we denote it by $\mathcal{T}_i := \{t_{i,j}\}_{j \in \mathbb{Z}_0^{N_{l,i}}}$ with $t_{s_i} = t_{i,0} < t_{i,1} \cdots < t_{i,N_{l,i}-1} < t_{i,N_{l,i}} = t_{s_i} + T_{e_i}$ for the i-th load. This local grid is used to evaluate the actual input in (4e), and it is defined relative to the starting time t_{s_i} and only the evaluation on these points $\{u_i(t)\}_{t \in \mathcal{T}_i}$ for all $i \in \mathcal{I}_r(t_0)$ are optimized.

The direct benefit from this inhomogeneous time grid is on the implementation side. If we fix the resolution of the local time grid, for example, as $\{j \cdot \delta t_i\}_{j \in \mathbb{Z}_0^{N_{l,i}}}$ with δt_i

the resolution of this time grid, the local time grid is then uniquely defined by $\{t_{s_i} + j \cdot \delta t_i\}_{j \in \mathbb{Z}_0^{N_{l,i}}}$ with a single scalar decision variable t_{s_i}. Meanwhile, as the time grids are introduced independently to each constraint in (4b) and (4e), the resolution of each time grid can be chosen individually. This can help limit the computational complexity, by reducing the resolution of the time grid on less relevant loads. Finally, an example that visualizes these grids for a numerical example is shown in Figure 2 in Section IV-B.

B. Input Interpolation

Note that the local time grid of a running load is fixed in terms of absolute time, while the global time grid is always receding with respect to the current time step. Thus, the local time grids do not need to overlap the global time grid. In order to evaluate each load on the global time grid, an approximation $\{u_i(t)\}_{t \in \mathcal{T}_g}$ of the actual input profile $u_i(\cdot)$ is required. This approximation can be numerically given by an interpolation of $\{u_i(t)\}_{t \in \mathcal{T}_i}$. One classical approach is to use a Lagrange polynomial with order $N_{l,i}$. However, high order polynomials cause high sensitivity with respect to t_{s_i}, and thus are not desirable for numerical optimization [24].

In our experiments, we noticed that this high sensitivity issue can lead to convergence failures even if a third order piece-wise polynomial is used. Based on our implementations, there are two interpolation methods that are reliable for the considered problem, piece-wise linear interpolation and kernel interpolation, which is a generalized interpolation scheme that includes polynomial interpolation as a special case [25, Chapter 4]. The regularity of a kernel regression can be controlled by the choice of its kernel function [25, Chapter 2], and one such example of good regularity is the radial basis function (RBF) kernel $k(x, y) = \exp(-\frac{\|x-y\|^2}{2\sigma^2})$.

Remark 2 *Only continuous interpolation splines are desirable in solving the load scheduling problem. More specifically, a discontinuous point will zero out the gradient between the global time grid and t_{s_i}, leading to a frozen t_{s_i} if a gradient-based solver is used. As a result, piecewise constant interpolation is not used in this problem. This is not restrictive as continuous functions can approximate any bounded-variation discontinuous function up to arbitrary accuracy.*

C. Finite Dimensional Problem

We interpret the $P_{\text{batt}}(t)$ by a piece-wise constant input on $\mathcal{T}_g$ such that the solution of (4c) can be worked out analytically, and we define it by $\hat{E}^*(t)$ in the following. Then, we state our finite order approximated problem by

$$\underset{\substack{t_{s_i}, \{P_{\text{batt}}(t)\}_{t \in \mathcal{T}_g} \\ \{u_i(t)\}_{i \in \mathcal{I}_r(t_0), t \in \mathcal{T}_i}}}{\text{minimize}} \sum_{i \in \mathcal{I}_r(t)} \left(J_i^d(t_{s_i} - t_{r_i}) + \tilde{L}_i^c(u_i(\cdot), t_{s_i}) \right)$$

subject to: $\sum_{i \in \mathcal{I}_r(t_0)} u_i(t) \leq P_{\min}(t) + P_{\text{batt}}(t), \ t \in \mathcal{T}_g$

$$\hat{E}^*(t) \in [0, E_{\max}], P_{\text{batt}}(t_{g,i}) \in [\underline{P}_{\text{batt}}, \overline{P}_{\text{batt}}], \ t \in \mathcal{T}_g, \quad (5a)$$

$$u_i(\cdot) = \mathcal{F}(u_i(t), t \in \mathcal{T}_i), \quad (5b)$$

$$0 \leq \overline{u}_i(t) - u_i(t) \leq c_i, \ i \in \mathcal{I}_r(t_0), \ t \in \mathcal{T}_i, \quad (5c)$$

$$0 \leq t_{s_i} - t_{r_i} \leq \Delta_i^d, \ i \in \mathcal{I}_r(t_0), \ t \in \mathcal{T}_i, \quad (5d)$$

$$t_{r_i} = \mathfrak{p}(i), \ i \in \mathcal{I}_r(t_0) \backslash \mathcal{I}_s, \tag{5e}$$

where $\tilde{L}_i^c$ represents the numerical integration of (3) which is implemented by the collocation method in the following numerical results [26, Chapter 3]. The non-convexity enters in constraint (5b), where $\mathcal{F}(\cdot)$ denotes the interpolation of the actual input (Section III-B). The utilization of the absolute time in the numerical implementation is important, as the t_{r_i}, t_{s_i} and t_0 uniquely defines all the time grids used in this problem. The online operation follows the receding horizon MPC scheme, details can be found in [6, Chapter 1.3], [19].

Remark 3 *Based on our experiments, it is important to remove redundant decision variables to avoid introducing unnecessary sensitivity to the numerical solver. For example, if a load i is started at $\tilde{t}$ and has not yet finished, one should avoid adding an extra equality constraint $t_{s_i} = \tilde{t}$. Instead, the code should take t_{s_i} as a constant $\tilde{t}$, and discard its related constraints such as (5d). Based on this idea, the starting time and the past input of a running load are treated as constant, and the code only keeps its future input in the list of decision variables.*

D. Discussions

To better illustrate the benefit of the continuous formulation, we first review its standard discrete-time mixed integer formulation as follows:

$$\underset{\substack{t_{s_i}, \{P_{\text{batt}}(t_{m_j})\} \\ \{u_i(t_{m_j})\}_{i \in \mathcal{I}_r(t_0)}}}{\text{minimize}} \quad \sum_{i \in \mathcal{I}_r(t)} \Big(J_i^d(t_{s_i} - t_{r_i}) \tag{6a}$$
$$+ \sum_{j=0}^{N_m} T_s \cdot J_i^c(\overline{u}_i(t_{m_j}) - u_i(t_{m_j})) \Big)$$

subject to: $t_{m_j} = t_0 + jT_s, \ j \in \mathbb{Z}_0^{N_m}$,

$$\sum_{i \in \mathcal{I}_r(t_0)} u_i(t_{m_j}) \leq P_{\min}(t_{m_j}) + P_{\text{batt}}(t_{m_j}), \tag{6b}$$

$$\hat{E}(t_{m_{j+1}}) = \hat{E}(t_{m_j}) - T_s \cdot P_{\text{batt}}(t_{m_j}), \ \hat{E}(t_{m_0}) = E(t_0),$$

$$\hat{E}(t_{m_j}) \in [0, E_{\max}], P_{\text{batt}}(t_{m_j}) \in [\underline{P}_{\text{batt}}, \overline{P}_{\text{batt}}], \tag{6c}$$

$$\forall i \in \mathcal{I}_r(t_0), \ t_{s_i} \in \{t_0 + jT_s\}_{j \in \mathbb{Z}}, \tag{6d}$$

$$t_{r_i} = \mathfrak{p}(i), \ i \in \mathcal{I}_r(t_0) \backslash \mathcal{I}_s \tag{6e}$$

$$0 \leq t_{s_i} - t_{r_i} \leq \Delta_i^d, \tag{6f}$$

$$\begin{cases} 0 \leq \overline{u}_i(t_{m_j}) - u_i(t_{m_j}) \leq c_i, \ t \in [t_{s_i}, t_{s_i} + T_{e_i}] \\ u_i(t_{m_j}) = 0, \text{otherwise} \end{cases} \tag{6g}$$

where N_m denotes the prediction horizon and T_s is the sampling time, and the objective (6a) approximates the integral of curtailment loss by the Euler method. The decision of the shifting t_{s_i} is determined by an integer variable in (6d). This mixed integer problem (6) was used in [11], [12] without considering the interruptible load, and the major difference between the mixed-integer formulation and the proposed continuous problem (5) lies in the decision of the shifting time t_{s_i} and the complexity of the time grids. In particular, the discrete time formulation only has one grid of the same resolution, i.e., the sampling time T_s, among all the loads and constraints. Therefore, the complexity of this shared time grid depends on the highest resolution that is required for a proper decision of the load profile or the

shifting. Accordingly, a finer T_s will significantly increase the numerical complexity of the problem because the number of the integers feasible for constraints (6f) increases. In comparison, the proposed scheme in (5) gets rid of this limitation in two aspects:

- The feasible set of shifting t_{s_i} is a continuum and the complexity with respect to the decision of t_{s_i} is fixed regardless of the resolution of any time grids.
- Each load has its local time grid, thus any change of its local grid will not affect other time grids.

Overall, the continuous formulation is preferable from a computational aspect. It can easily adapt to a problem with nonlinear dynamics, while the mixed integer formulation suffers from the numerical intractability issue as a non-convex mixed integer problem. Furthermore, the choices of solvers with the continuous formulation is more flexible. On one hand, it can use a local algorithm that is easier to deploy in embedded systems with limited computational power [27]. On the other hand, similar to the mixed integer solvers, it can choose a global solver [28] to ensure global optimality without polynomial time convergence. Last but not least, the continuous formulation is more scalable in that it can use distributed optimization algorithms because the appliances are only coupled via the affine constraint (4b).

IV. NUMERICAL RESULTS

We first consider uninterruptible loads to compare the proposed continuous formulation (5) with the mixed integer version (6). Then an example with interruptible loads based on real data validates the proposed scheme. We consider the worst case, so the actual generated power is only $P_{\min}$.

A. Benchmark Comparison

This example considers uninterruptible loads, and the proposed scheme is compared against the benchmark mixed-integer formulation (6), which is implemented with `Yalmip` interfacing the `Gurobi` MIQP solver [29], one of the fastest commercial solvers, and BNB, an open source mixed integer solver. Moreover, the continuous problem (5) is implemented with `Casadi` [30] interfacing the `Ipopt` solver [31]. All code is run on a laptop with Intel i7-11800H and 32 GB of memory. We optimize the implementation of the mixed integer formulation by an auxiliary system [11][1], without which solution time is reported to be significantly longer [11].

In this comparison, two loads are considered, whose load profiles are plotted as blue solid lines in Figure 1. The shifting time upper bound and the curtailment upper bound are, $\Delta_i^d = 1$ [h] and $c_i = 0.5$[kW]. The shifting loss and the curtailment loss are $J_i^d = 10 \max\{t_{s_i} - t_{r_i} - 0.1, 0\}^2$, $J_i^c = (\overline{u}_i(t) - u_i(t))^2$, where the shifting loss is zero if the shifting time is less than 0.1 [h]. Accordingly, to ensure that the discrete version (6) can discover this property, its sampling time T_s is set below 0.1 [h]. To make good use of the inhomogeneous time grid system, the local time grid in the

---

[1]This pure delayed system outputs the load profile once started, which converts an integer delay variable to a sequence of binary decision variables.
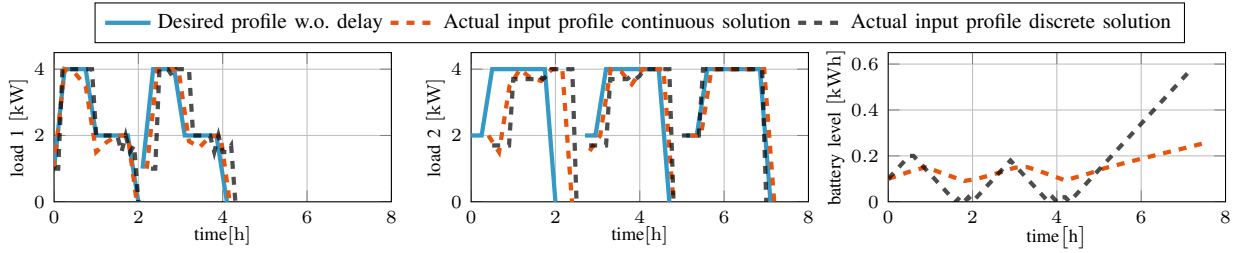
Fig. 1: Comparison of closed loop behaviour between continuous-time (5) and mixed-integer formulations (6) with $T_s = 0.1[h]$

continuous formulation (5) has a resolution of $0.2\,[\mathrm{h}]$, with a global time grid of resolution $0.1\,[\mathrm{h}]$. The load of each appliance is interpolated by kernel regression with the RBF kernel. The total power consumption is upper bounded by $P_{\min}(\cdot) = 5.5[\mathrm{kW}]$. Meanwhile, the battery is bounded by $\underline{P}_{\mathrm{batt}} = -0.2[\mathrm{kW}]$, $\overline{P}_{\mathrm{batt}} = 0.2[\mathrm{kW}]$, $E_{\max} = 1[\mathrm{kWh}]$.

Regarding the aforementioned $P_{\min}(\cdot)$ and the battery dynamics, if these two loads are requested to start at the same time, the only feasible solution is to shift the second load and curtail at least one of them. A sequence of requests are sent to the system, and the closed-loop simulation results are plotted Figure 1. It is observed that both schemes successfully shift the second load and curtail one of them to ensure feasibility. However, the continuous scheme tends to introduce more curtailment which results in less delay. The statistics of the proposed continuous formulation and the mixed integer formulation with different sampling times are summarized in Table I. In particular, both the continuous and the discrete version are initialized by a random feasible shifting, and the solution time is calculated by the average of 50 independent runs. As the mixed-integer formulation solves an MIQP and can therefore find a global optimum, we can see that the continuous formulation results in a comparable performance, while using significantly less solution time. Meanwhile, we can see that the solution time of the mixed-integer formulation is sensitive to the sampling time and solver, even though smaller sampling time can improve its optimality.

	Solution Time (Gurobi/BNB)	Delay Loss	Curtailment Loss	Total Loss
Continuous	1.1011 [s]	1.2181	0.2525	1.4706
MIQP ($T_s = 0.1[h]$)	9.1340 [s] 49.2573 [s]	1.300	0.1440	1.4440
MIQP ($T_s = 0.05[h]$)	16.6564 [s] 194.34 [s]	1.275	0.1450	1.4200
MIQP ($T_s = 0.025[h]$)	36.9109 [s] N.A	1.2875	0.1215	1.4090

TABLE I: Statistics of two formulations: loss is the accumulated actual loss evaluated on all the closed-loop input profiles. The solution time for BNB with $T_s = 0.025[h]$ is dropped as each run takes more than 2 hours.

B. Case Study with Interruptible Loads

We now consider the scheduling of three home appliances: an oven, washing machine and dishwasher, within which the washing machine is interruptible between the washing and the drying program. The load profiles based on real-world data are plotted as thick blue lines Figure 2 (a)-(c) [32], where the washing program of the washing machine takes 1.5 hours and the drying program takes another 2 hours. To reduce the model complexity, we filter out the fluctuations and use a piece-wise constant function to model the desired
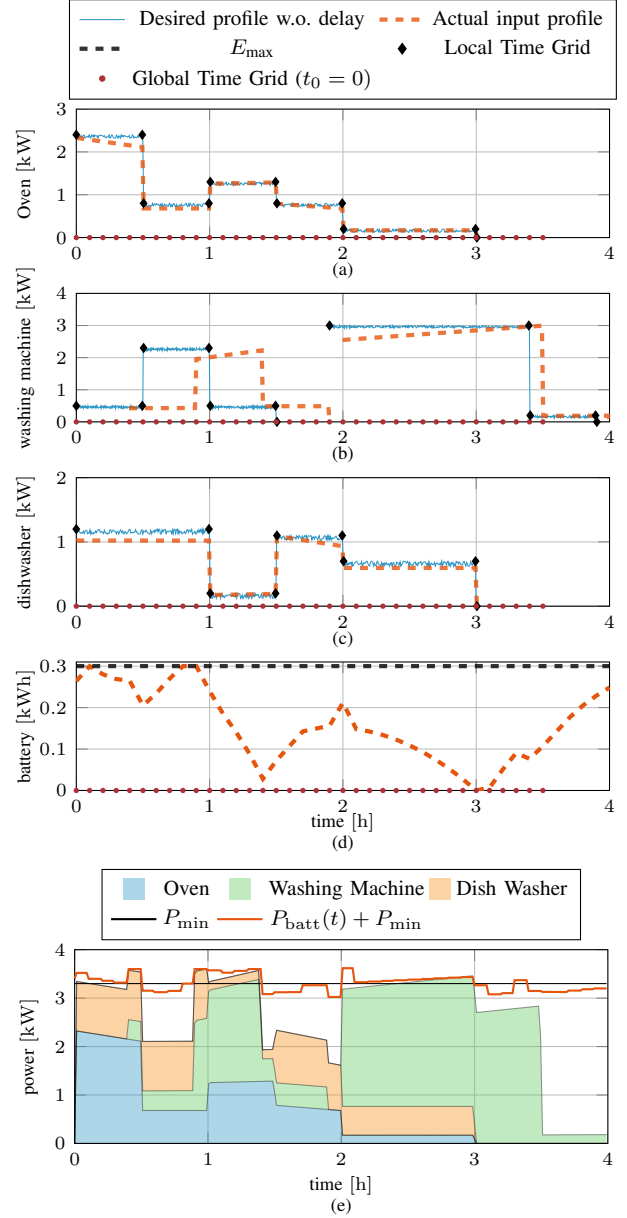

Fig. 2: Case study of home appliance scheduling. (a)-(c): Individual appliance profiles with local and global time grid markers. (d) Battery Energy. The global time grid is plotted on the horizontal axis in (a)-(d) to show the correspondence between the local and global grids and the load profile. The global grid is shown as used at $t_0 = 0$; the grid is shifts with respect to t_0. (e): Stacked chart of the power consumption of all the appliances. The curve of $P_{\mathrm{batt}}(t) + P_{\min}$ shows the change of the power supply profile and how the power limit constraint (4b) is satisfied. The input trajectories are smooth as they are the planned input profiles.

load profile. These load profiles have significantly different patterns so we use the proposed inhomogeneous time grid system to minimize the number of decision variables. The global grid is equidistant with a resolution of 0.1 [h], while each load has a non-equidistant local time grid chosen to capture the major patterns in the desired profile, shown with diamonds in Figure 2. The time grids are:

Oven: $[0, 0.495, 0.5, 0.995, 1.0, 1.495, 1.5, 1.995, 2, 2.99, 3]$

$$\text{Washing Machine} : \begin{cases} [0, 0.495, 0.5, 0.995, 1.0, 1.495, 1.5] \\ [0, 1.495, 1.5, 1.995, 2] \end{cases}$$

Dishwasher: $[0, 0.995, 1, 1.495, 1.5, 1.995, 2, 2.995, 3]$.

A piece-wise linear interpolation is used. The $P_{\min}$ is 3.2[kW] with a battery dynamics confined by $\underline{P}_{\text{batt}} = -0.2$[kW], $\overline{P}_{\text{batt}} = 0.2$[kW], $E_{\max} = 0.3$[kWh]. Meanwhile, the delay penalty and the curtailment penalty are defined by $J_{d_i} = 10(t_{s_i} - t_{r_i})^2$, $J_{c_i} = 0.1\,|\overline{u}_i(t) - u_i(t)|$.

The result of this scheduling problem is shown in Figure 2, from which it is observed that the oven is launched without any delay, and the washing machine is shifted to ensure the satisfaction of the power limit. It is also observed the battery is pre-charged at around 0.5 [h] and 1.4 [h] and then discharged at around 0.9 [h] and 2 [h] to compensate the peak load in the washing and the drying program. It is noteworthy to point out the curtailment adapts to the available power. This can be observed from the curtailment of the drying program, which is only introduced at around 2 [h] but not 3.5 [h]. This is because the dishwasher finished at 3 [h] and the power supply can sustain the drying program.

V. Conclusions

This work proposes a continuous-time formulation of a load scheduling problem within a household. An inhomogeneous time grid system is introduced to decouple the model complexity of each load. The advantage over the standard mixed integer version is validated by a numerical example with better solution time and comparable closed-loop performance. Finally, the proposed scheme is validated in a scheduling problem consisting of an oven, an interruptible washing machine and a dishwasher.

References

[1] S. Vazquez, S. M. Lukic, E. Galvan, L. G. Franquelo, and J. M. Carrasco, "Energy storage systems for transport and grid applications," *IEEE Tran. Ind. Electron.*, vol. 57, no. 12, pp. 3881–3895, 2010.

[2] M. Erol-Kantarci and H. T. Mouftah, "Wireless sensor networks for cost-efficient residential energy management in the smart grid," *IEEE Tran. Smart Grid*, vol. 2, no. 2, pp. 314–325, 2011.

[3] M. H. Baloch, G. S. Kaloi, and Z. A. Memon, "Current scenario of the wind energy in pakistan challenges and future perspectives: A case study," *Energy Reports*, vol. 2, pp. 201–210, 2016.

[4] F. Fardoun, O. Ibrahim, R. Younes, and H. Louahlia-Gualous, "Electricity of lebanon: problems and recommendations," *Energy Procedia*, vol. 19, pp. 310–320, 2012.

[5] R. Stamminger and A. Schmitz, "Load profiles and flexibility in operation of washing machines and dishwashers in europe," *Int. J. Consumer Studies*, vol. 41, no. 2, pp. 178–187, 2017.

[6] S. V. Raković and W. S. Levine, *Handbook of model predictive control*. Springer, 2018.

[7] R. Godina, E. M. Rodrigues, E. Pouresmaeil, J. C. Matias, and J. P. Catalão, "Model predictive control home energy management and optimization strategy with demand response," *Applied Sciences*, vol. 8, no. 3, p. 408, 2018.

[8] D. Mariano-Hernández, L. Hernández-Callejo, A. Zorita-Lamadrid, O. Duque-Pérez, and F. S. García, "A review of strategies for building energy management system: Model predictive control, demand side management, optimization, and fault detect & diagnosis," *Journal of Building Engineering*, vol. 33, p. 101692, 2021.

[9] X. Hou, J. Wang, T. Huang, T. Wang, and P. Wang, "Smart home energy management optimization method considering energy storage and electric vehicle," *IEEE Access*, vol. 7, pp. 144010–144020, 2019.

[10] Y. Lian, J. Shi, M. P. Koch, and C. N. Jones, "Adaptive robust data-driven building control via bi-level reformulation: an experimental result," *arXiv preprint arXiv:2106.05740*, 2021.

[11] G. L. Sinsley and D. F. Opila, "Model predictive control for dynamic load scheduling in small power systems," in *Proc. 2020 Amer. Control Conf. (ACC)*, pp. 4140–4146, 2020.

[12] A. Di Giorgio, L. Pimpinella, and F. Liberati, "A model predictive control approach to the load shifting problem in a household equipped with an energy storage unit," in *Proc. 20th Medi. Conf. Control & Autom. (MED)*, pp. 1491–1498, 2012.

[13] Z. Yang, K. Long, P. You, and M.-Y. Chow, "Joint scheduling of large-scale appliances and batteries via distributed mixed optimization," *IEEE Tran. Power Sys.*, vol. 30, no. 4, pp. 2031–2040, 2014.

[14] F. Qayyum, M. Naeem, A. S. Khwaja, A. Anpalagan, L. Guan, and B. Venkatesh, "Appliance scheduling optimization in smart home networks," *IEEE Access*, vol. 3, pp. 2176–2190, 2015.

[15] W. P. Heemels, K. H. Johansson, and P. Tabuada, "An introduction to event-triggered and self-triggered control," in *Proc. 51st IEEE Conf. Decis. Control (CDC)*, pp. 3270–3285, 2012.

[16] N. Stricker, Y. Lian, Y. Jiang, C. Jones, and L. Thiele, "Joint energy management for distributed energy harvesting systems," in *Proc. 19th ACM Conf. Embedded Netw. Sensor Sys. (SenSys'21)*, ACM, 2021.

[17] J. B. Berglind, T. Gommans, and W. Heemels, "Self-triggered mpc for constrained linear systems and quadratic costs," *IFAC Proc. Vol.*, vol. 45, no. 17, pp. 342–348, 2012.

[18] D. Bernardini and A. Bemporad, "Energy-aware robust model predictive control based on noisy wireless sensors," *Automatica*, vol. 48, no. 1, pp. 36–44, 2012.

[19] Y. Lian, S. Wildhagen, Y. Jiang, B. Houska, F. Allgöwer, and C. N. Jones, "Resource-aware asynchronous multi-agent coordination via self-triggered mpc," in *Proc. 59th IEEE Conf. Decis.Control (CDC)*, pp. 685–690, 2020.

[20] Y. Lian, Y. Jiang, N. Stricker, L. Thiele, and C. N. Jones, "Resource-aware stochastic self-triggered model predictive control," *IEEE Contr. Syst. Lett.*, vol. 6, pp. 1262–1267, 2022.

[21] Y. Lian, Y. Jiang, N. Stricker, L. Thiele, and C. N. Jones, "Robust resource-aware self-triggered model predictive control," *IEEE Contr. Syst. Lett.*, vol. 6, pp. 1724–1729, 2022.

[22] G. L. Sinsley, D. F. Opila, E. S. Oh, and J. D. Stevens, "Upper bound performance of shipboard power and energy systems for early-stage design," *IEEE Access*, vol. 8, pp. 178600–178613, 2020.

[23] H. G. Bock and K.-J. Plitt, "A multiple shooting algorithm for direct solution of optimal control problems," *IFAC-PapersOnLine*, vol. 17, no. 2, pp. 1603–1608, 1984.

[24] R. L. Burden, J. D. Faires, and A. M. Burden, *Numerical analysis*. Cengage learning, 2015.

[25] V. I. Paulsen and M. Raghupathi, *An introduction to the theory of reproducing kernel Hilbert spaces*, vol. 152. Cambridge university press, 2016.

[26] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral methods: fundamentals in single domains*. Springer Science & Business Media, 2007.

[27] J. Nocedal and S. Wright, *Numerical optimization*. Springer Science & Business Media, 2006.

[28] C. A. Floudas, *Deterministic global optimization: theory, methods and applications*, vol. 37. Springer Science & Business Media, 2013.

[29] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual (accessed: 2022-05-10)." https://www.gurobi.com.

[30] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "Casadi: a software framework for nonlinear optimization and optimal control," *Math. Program. Comput.*, vol. 11, no. 1, pp. 1–36, 2019.

[31] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Math. Program.*, vol. 106, no. 1, pp. 25–57, 2006.

[32] V. T. A. R. Institute, "Appliance data[online]." https://ari.vt.edu/research-data.html, 2013.