An efficient technique for solving the scheduling of appliances in smart-homes

Julien M. Hendrickx[1], Raphaël M. Jungers[1], Guillaume Vankeerberghen[1] and Laurence A. Wolsey[2]

***Abstract*— In the development of the electricity grid, Demand Response is a current subject of research for reasons of robustness, efficiency and renewable energy integration. Among the aspects of Demand Response, dynamic pricing is critical, and real-time pricing is a favored way of implementing it. The efficiency of such methods would however rely crucially on the ability and willingness of the end-user to react to varying prices.**

For this reason, we look at the problem of automatically scheduling the appliances that a user needs to run when the energy price is sent by the provider. We propose a discrete-time formulation for this problem, show how it can be solved by a minimum cut algorithm and compare it with previously proposed techniques. Compared to an exact method requiring a mixed integer programming solver, our approach produces solutions very close to optimal in shorter running times and does not require proprietary software.

I. INTRODUCTION

The growing penetration of intermittent sources of renewable energy in the grid and the benefits in efficiency and robustness of flattening peak loads have spurred tremendous interest in Demand Response. Demand Response consists in finding ways of influencing the electricity consumption of end-users in (near) real-time [1]. One means of influencing demand is by dynamic pricing, where prices can change on short notice, e.g, a few hours or a day ahead. Among dynamic pricing options, real-time pricing is a favored approach [2]. Real-time pricing consists in setting a price for a specific future period, e.g., adjusting the price every hour or setting hourly-based prices for the following day. Such pricing policies have become realizable thanks to the advent of cheap smart-meters and means of communication, and are a current subject of research [3]. Hence, dynamic pricing is realizable and could be very advantageous, but it would be useful only if the end-users are willing and able to react to these incentives. It is this part of the problem with which we are concerned.

To this end, different solutions have been proposed to efficiently and automatically control energy-consuming tasks in homes or facilities. For instance, [4] develops a system to analyze and optimize user consumption given hourly-based day-ahead prices. In [5] and [6] the authors treat the problem of scheduling home appliances when price and demand are stochastic. Another example comes from the Automatic Control Lab at KTH in the context of the Royal Seaport Project (www.stockholmroyalseaport.com) [7]. There, the authors propose a method to optimally schedule home appliances when the prices are given one day ahead.

In the present paper we consider the problem of scheduling different energy-consuming tasks when the energy price profile is given and fixed for a prescribed period. We show that this problem can be formulated in discrete-time as a certain job scheduling problem which can be solved via a minimum cut algorithm [8]. We give a bound on the loss of optimality that the use of discretized starting-times could cause. We provide a specialized implementation of a min-cut algorithm and compare its performance with the continuous-time method in [7].

We show that the discretization introduced by our formulation has a very limited effect on the cost of the solution found and comes with an improved computation time. Computation time is important in this real-time setting, especially if one further aims at joint optimizing for a large number of users. Moreover, our algorithm is easily implementable and does not require proprietary software as opposed to the specialized Mixed Integer Linear Programming solvers used in other approaches [4], [7], [9]. The latter issue might be of importance if a large deployment is envisioned.

II. SETTING

We use the same setting as in [7] as it is well documented and stems from collaboration with different actors on the Royal Seaport Project (www.stockholmroyalseaport.com). We are concerned with the problem of scheduling a list of user-specified home-appliances in a certain period of time $[0,T]$ when the electricity price is varying and known over that period. That is, we suppose that the user has chosen a list of appliances, which we index by $i = 1,\dots,N_{app}$, that need to be run during the concerned period. We suppose that there is some freedom in the time to process them, T_i^{lb} is the earliest starting-time and T_i^{ub} is the latest ending-time. The user can also specify precedence relations between certain appliances. For instance, appliance 3 may not start until appliance 5 has finished. The price per kWh is received from the energy provider, $c : [0,T] \to \mathbb{R}^+$, and the optimization is run to propose (and implement) an optimal schedule.

Running these appliances requires some energy, but the power load is not necessarily constant over the time of processing. This is modeled by defining n_i ordered phases for

This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office. The research is supported by the Concerted Research Action (ARC) of the French Community of Belgium. The scientific responsibility rests with its authors. R. M. J. is a F.R.S.-FNRS Research Associate, G. V. is a F.R.I.A. Fellow.

[1]ICTEAM Institute, Université catholique de Louvain, Belgium;

[2]Center of Operations Research and Econometrics (CORE), Université catholique de Louvain, Belgium;

{julien.hendrickx, raphael.jungers, laurence.wolsey} @uclouvain.be

each appliance during which the power load is assumed constant. These *energy phases* (sometimes abbreviated e.p.'s), indexed by $j = 1, \dots, n_i$, have certain times of processing, T_{ij} in hours, and a certain energy requirement Er_{ij} in Wh. The power load of an energy phase is thus considered equal to Er_{ij}/T_{ij}. The time separating the energy phases might be flexible, this is modeled by a minimum delay $\underline{D}_{ij}$ and a maximum delay $\overline{D}_{ij}$ $j = 1, \dots, n_i - 1$ between the end and the beginning of two consecutive energy phases of an appliance i. For instance, one cycle of a washing machine might be broken down in a pre-washing phase followed by a heating phase, a maintenance phase and then different rinsing phases that could be delayed within some time interval [7]. The notation is summarized in Table I.

TABLE I
CONTINUOUS-TIME PARAMETERS.

Specifications of the appliances	
$i \in \{1, \dots, N_{app}\}$	Index for the N_{app} appliances
$j \in \{1, \dots, n_i\}$	Index for the n_i energy phases (e.p.) of appliance i
Er_{ij}	Total energy required by e.p. j of appliance i, in Wh
T_{ij}	Time of processing of e.p. j of appliance i, in h
$\underline{D}_{ij}$	Minimum delay between e.p. j and $j+1$, in h
$\overline{D}_{ij}$	Maximum delay between e.p. j and $j+1$, in h
User preferences	
$[i_1, i_2]$	Precedence; appliance i_2 may not start before the end of i_1
T_i^{lb}	Earliest starting time for appliance i, in h
T_i^{ub}	Latest ending time for appliance i, in h
Cost	
$c : [0, T] \to \mathbb{R}^+$	Price of energy during the day, in SEK/Wh ($SEK =$ Swedish krona)

We provide an algorithm for finding a minimum cost schedule of such energy phases. Note that in [7], the authors also consider that the energy provider sends a profile of the CO_2 production per kWh over the period. This situation is justified by the case of Sweden where generally, during the day a large part of the energy is provided by renewable sources whereas during the night it is mainly bought from neighboring countries relying more on fossil fuels. The authors then compute the Pareto frontier between the economical and the low-CO_2 objectives and provide the user with a choice. We do not explicitly treat the CO_2 objective here but the computation of a Pareto frontier can be carried out within our framework by optimizing with costs that correspond to different convex combinations of the economical and CO_2 cost or by fixing a price per g of CO_2.

III. DISCRETE-TIME FORMULATION AND EQUIVALENT MIN-CUT PROBLEM

We express the problem of finding an optimal schedule by discretizing the set of starting times and obtain a Binary Linear Program. This program can be efficiently solved, even though it is combinatorial. We then link this problem with one already treated in the literature that can be solved by finding a minimum $s-u$-cut in a suitably defined digraph. Finally, we explain how to construct this digraph in our case.

A. Discrete-time formulation

The decision variables over which we optimize are the starting-times of the energy phases. We discretize the problem by restraining these starting-times to multiples of a chosen Δ_T with $N_t \Delta_T = T$ and we index them by $t = 1, \dots, N_t$, corresponding to the (continuous) times $0, \Delta t, 2\Delta t, \dots (N_t - 1)\Delta t$. We then define a binary variable for each starting-time of each energy phase,

$$x_{ij,t} = \begin{cases} 1 & \text{if e.p. } j \text{ of appliance } i \text{ starts at } t \\ 0 & \text{otherwise} \end{cases} .$$

In continuous-time, $x_{ij,t} = 1$ means that energy phase ij starts at time $(t-1)\Delta_T$.

In fact, the earliest starting-times T_i^{lb} and latest ending-times T_i^{ub} of each appliance, together with the precedence relations, the times of processing and the minimum delays induce earliest and latest starting-times for each energy phase. These can be computed by updating them through a breadth-first search in the directed forest of precedence relations and its reverse. From now on, we will consider that we have these and denote them by e_{ij} for the earliest and l_{ij} for the latest integral starting-times. Therefore, we eliminate all $x_{ij,t}$ for $t < e_{ij}$ and $t > l_{ij}$ and this implicitly implements the constraints on the time frame for running each appliance.

We associate a cost $w_{ij,t}$ to each feasible starting-time of each energy phase. This cost is equal to the cost of providing the energy through a constant power load of $P_{ij} = Er_{ij}/T_{ij}$ over its continuous time of processing,

$$w_{ij,t} = \int_{\Delta_T(t-1)}^{\Delta_T(t-1)+T_{ij}} \frac{Er_{ij}}{T_{ij}} c(s) ds \quad t = e_{ij}, \dots, l_{ij}.$$

So, the cost of a schedule can be expressed linearly in the $x_{ij,t}$,

$$\sum_{i=1}^{N_{app}} \sum_{j=1}^{n_i} \sum_{t=e_{ij}}^{l_{ij}} w_{ij,t} x_{ij,t}. \tag{1}$$

It remains to constrain the precedences between energy phases and between appliances. To express all these in one set of constraints, let us define a set of time-lags. These time-lags are such that if $t_{i_1 j_1}$ and $t_{i_2 j_2}$ are two feasible integral starting-times for two energy phases with time-lag $d_{i_1 j_1, i_2 j_2}$ they must satisfy $t_{i_1 j_1} + d_{i_1 j_1, i_2 j_2} \leq t_{i_2 j_2}$. We define the following time-lags,

- for each appliance i and for its e.p.'s $j = 1, \dots, n_i - 1$, we define $d_{ij,i(j+1)} = \left\lceil \frac{T_{ij} + \underline{D}_{ij}}{\Delta_T} \right\rceil$ to enforce the minimum delay between two subsequent e.p.'s,
- for each appliance i and for its e.p.'s $j = 2, \dots, n_i$, we define $d_{ij,i(j-1)} = -\left\lfloor \frac{T_{i(j-1)} + \overline{D}_{i(j-1)}}{\Delta_T} \right\rfloor$ to enforce less than the maximum delay between two subsequent e.p.'s,
- for each precedence relation between appliances, $[i_1, i_2]$, we define $d_{i_1 n_{i_1}, i_2 1} = \left\lceil \frac{T_{i_1 n_{i_1}}}{\Delta_T} \right\rceil$ to enforce the timely order between these appliances.

The constraints on the precedence and min and max delays can now be expressed by

$$\sum_{s=e_{i_1 j_1}}^{t} x_{i_1 j_1,s} \geq \sum_{s=e_{i_2 j_2}}^{t+d_{i_1 j_1,i_2 j_2}} x_{i_2 j_2,s} \tag{2}$$

for all couples $i_1 j_1$, $i_2 j_2$ with a time-lag and for t going from $\max\{e_{i_1 j_1}, e_{i_2 j_2} - d_{i_1 j_1,i_2 j_2}\}$ to $\min\{l_{i_1 j_1}, l_{i_2 j_2} - d_{i_1 j_1,i_2 j_2}\}$. Indeed, the left-hand side is one if $i_1 j_1$ has started at or before time t and the right-hand side is one if $i_2 j_2$ has started at or before $t + d_{i_1 j_1,i_2 j_2}$. Hence, these inequalities encode that if $i_1 j_1$ has not been started at or before t then $i_2 j_2$ cannot have been started at or before $t + d_{i_1 j_1,i_2 j_2}$, or conversely, if $i_2 j_2$ has been started at or before $t + d_{i_1 j_1,i_2 j_2}$ then $i_1 j_1$ must have been started at or before t. The last constraints to be added are that all e.p.'s have to be started once during the period,

$$\sum_{s=e_{ij}}^{l_{ij}} x_{ij,s} = 1, \tag{3}$$

and the implicit constraint that all $x_{ij,t}$ are binary.

The program is

$$\begin{array}{ll} \min_{x_{ij,t} \in \{0,1\}} & (1) \\ \text{s.t.} & (2), (3) . \end{array} \tag{4}$$

One should note that this program is feasible if there are no cycles in the precedence relations and if one has chosen a sufficiently small Δ_T. Such a Δ_T always exists if there is an interval of feasible continuous starting-times for each energy phase. From now on we assume that this is the case, if it were not the case there would be no room for optimizing.

Program (4) is a Binary Linear Program and as such would not be easy to efficiently solve. However, its particular structure allows us to design efficient algorithms for solving it. As a first hint on this, if one applies the change of variables $z_{ij,t} = \sum_{s=e_{ij}}^{t} x_{ij,s}$ one could verify that the constraints $A\mathbf{z} \geq b$ of the LP relaxation are such that A is Totally Unimodular, meaning that its determinant is either 1 or -1, and b is integral, so that the LP relaxation has an integral optimal solution [10]. That said, we do not prove it here but rather use the fact that our formulation highlights the correspondence with the problem in [8, Sec. 2.] for which a minimum $s-u$-cut equivalent problem is given.

B. Minimum cut

Our discretized problem is equivalent to the Project Scheduling with Start-Time Dependent Costs described in [8, Sec. 2.]. Indeed, this problem consists of scheduling a set of jobs $J = \{1, \ldots, n\}$ with integral processing times p_k, $k \in J$. Each job causes a cost w_{kt} if it is started at integral time $t \in \{0, \ldots, T\}$ and there are prescribed maximum and minimum time-lags between some jobs. The set of jobs with time-lags is denoted by $L \subseteq J \times J$, i.e., a set of ordered job pairs. For each pair (k,l) in L there is an integral time-lag d_{kl} that can be either positive or negative. So that if t_k, t_l denote feasible integral starting-times for jobs k, l with $(k,l) \in L$ then one must have $t_l \geq t_k + d_{kl}$. With the notation introduced in the last Section one can see that our problem is equivalent by replacing the jobs by our energy phases.

It is proven in the same paper that an optimal schedule for this problem can be retrieved from a minimum cut in a suitably defined digraph. For the sake of completeness, we provide the definition of this digraph in our case.

We define the node set $V = \{v_{ij,t} \,|\, i = 1, \ldots, N_{app}; j = 1, \ldots, n_i; t = e_{ij}, \ldots, l_{ij} + 1\} \cup \{s\} \cup \{u\}$, i.e., for each energy phase we have a node for each feasible starting-time plus one, and we have a source s and a sink u. The set of arcs is best understood when separated in three classes. We give some intuition on the rationale behind their definition, more details can be found in [8].

- **Assignment arcs:** $(v_{ij,t}, v_{ij,(t+1)})$ for each energy phase ij and for $t = e_{ij}, \ldots, l_{ij}$, with cost equal to $w_{ij,t}$. Each of these represents the feasible starting-time t for energy phase ij, if it is in the minimum cut then the optimal starting-time for ij is t. These arcs will be the only one to be cut as the others have infinite cost.
- **Temporal arcs:** $(v_{i_1 j_1,t}, v_{i_2 j_2,(t+d_{i_1 j_1,i_2 j_2})})$ for all time-lags $d_{i_1 j_1,i_2 j_2}$ and t from $\max\{e_{i_1 j_1} + 1, e_{i_2 j_2} + 1 - d_{i_1 j_1,i_2 j_2}\}$ to $\min\{l_{i_1 j_1}, l_{i_2 j_2} - d_{i_1 j_1,i_2 j_2}\}$, with infinite cost. These arcs impose the time-lags. When there is a $(v_{i_1 j_1,t}, v_{i_2 j_2,(t+d_{i_1 j_1,i_2 j_2})})$ arc with infinite cost, then if a certain assignment arc $(v_{i_2 j_2,s}, v_{i_2 j_2,(s+1)})$ is in the cut with $s < t + d_{i_1 j_1,i_2 j_2}$ (implying that $i_2 j_2$ starts in s), then an assignment arc $(v_{i_1 j_1,r}, v_{i_1 j_1,(r+1)})$ must be in the cut for some $r < t$ (implying that $i_1 j_1$ starts before t) for the cut to have a finite cost.
- **Auxiliary arcs:** $(s, v_{ij,e_{ij}})$ and $(v_{ij,(l_{ij}+1)}, u)$ for all energy phase ij, with infinite cost. These link the source and sink to the earliest and latest nodes of each energy phase. Because s and u have to be separated by the $s-u$-cut and because these arcs have infinite cost they force to cut an assignment arc for each energy-phase.

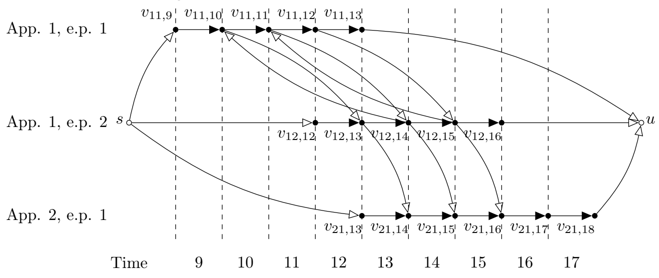
Example 1. *To illustrate this definition consider a simple case with two appliances as described in Table II and* $T = N_t = 24$ *(*$\Delta_T = 1h$*). We set user-preferences to the precedence* $[1, 2]$ *between the appliances and times of use* T_1^{lb} *=9:00,* T_1^{ub} *=16:00,* T_2^{lb} *=9:00 and* T_2^{ub} *=19:00. We can easily compute the earliest and latest starting-times;* $e = [9, 12, 13]$ *and* $l = [12, 15, 17]$*. The digraph for this case is pictured on Fig. 1.*

TABLE II
SPECIFICATIONS FOR EXAMPLE 1.

Appliance 1:	e.p. 1	e.p. 2
T_{1j} [h]	1	1
$\underline{D}_{1j}$ [h]	2	
$\overline{D}_{1j}$ [h]	3	
Appliance 2:	e.p. 1	
T_{2j} [h]	2	

Remark. The network construction allows more general constraints on the time of use of appliances. Indeed, with

Fig. 1. Digraph for Example 1. The arcs with white arrowheads are temporal and auxiliary arcs with infinite capacity, the ones with black arrowheads are assignment arcs.



some care, one could implement a constraint such as "appliance i has to run in $[T_i^{lb}, T_i^{ub}]$ *or* in $[T_i^{lb'}, T_i^{ub'}]$" for non-overlapping intervals by adding assignment arcs with infinite cost.

IV. Cost of Discretization and Algorithm

A. Bound on the cost of discretization

We now provide a bound on the additional cost that might be endured when forcing a schedule to use starting-times that are multiples of Δ_T instead of any continuous time.

Proposition 1: If c^*_{ct} and c^*_{dt} are the cost of an optimal continuous-time and an optimal discrete-time schedule respectively then

$$c^*_{dt} - c^*_{ct} \leq \sum_{i=1}^{N_{app}} \sum_{j=1}^{n_i} h_{ij} \Delta_T \frac{Er_{ij}}{T_{ij}} \left(\max_{s \in [(e_{ij}-1)\Delta_T, (l_{ij}-1)\Delta_T + T_{ij}]} c(s) - \min_{s \in [e^c_{ij}, l^c_{ij}]} c(s) \right), \tag{5}$$

where h_{ij} is the maximum of the maximum number of ordered energy phases that have to run before energy phase ij and after energy phase ij (counting itself as one), e_{ij}, l_{ij} and e^c_{ij}, l^c_{ij} are respectively the discrete and continuous earliest and latest starting-times of energy phase ij.
Note that all these parameters can be computed by Breadth First Search in the forest of precedence relations and its reverse.

Proof: We show how to construct a discrete-time feasible schedule with starting-times that are displaced by at most $h_{ij}\Delta_T$ from the optimal continuous starting-times. If an energy phase has no precedence with regard to any other one its continuous-time can be rounded to the next or previous multiple of Δ_T so that it is displaced by less than Δ_T. Recall that we assumed in Section III-A that the time frame is large enough and Δ_T small enough to make one of these feasible. Let us then consider a sequence of k chronologically ordered energy phases with continuous starting-times $S_1, \dots, S_k$. Suppose that S_1 is the earliest feasible continuous starting-time, we can place 1 to the next multiple of Δ_T which displaces it by less than Δ_T. If the subsequent energy phase followed at the earliest it has to be delayed by the same amount and then this time has to be rounded to the next multiple of Δ_T. Which displaces it by less than $2\Delta_T$. If the second energy phase did not follow at the earliest it can also be set within $2\Delta_T$. One can carry this procedure on along the sequence. The same propagation of displacements could happen in the other direction, if the k optimal continuous starting-times were the latest possible. This is the reason why h_{ij} has to be the maximum between the maximum number of sequential energy phases that have to run before ij and the maximum number of sequential energy phases that have to run after ij (counting itself as one).

Since this discrete-time schedule is feasible, its additional cost is an upper bound for the one of the optimal discrete-time schedule. The difference between the cost of the constructed discrete-time schedule and the continuous-time one can be bounded by the added cost of running all the displaced loads at the most expensive time instead of the cheapest one,

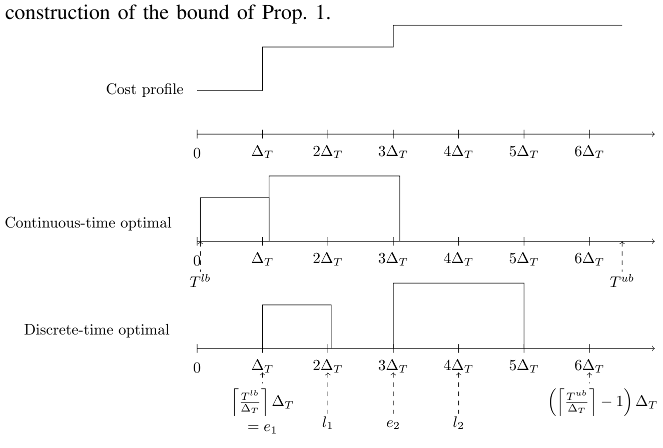
$$\sum_{i=1}^{N_{app}} \sum_{j=1}^{n_i} h_{ij} \Delta_T \frac{Er_{ij}}{T_{ij}} \left(\max_s c(s) - \min_s c(s) \right). \tag{6}$$

We can further refine this expression, if we have the discrete and continuous earliest and latest starting-times of each energy phase, by taking the maximum and minimum between them so as to obtain Eq. (5). ■

We will see in the simulations that this bound is usually very conservative. Nevertheless, the following example shows that it can be reached in some pathological cases.

Example 2. *The scheduling problem with two energy phases represented on Fig. 2 shows that the bound of Prop. 1 can be tight.*

Fig. 2. Sketch of a schedule with two energy phases which is pathological for the cost of discretizing. The lower bound T^{lb} on the first starting time is just above 0, and the first e.p. lasts for just above Δt. In continuous time, the first e.p. can thus start just after 0 and the second one a bit after Δt. However, when discretizing with a period Δt, the constraint on the starting time of the first e.p. forces it to start at Δt, and to finish a bit after $2\Delta t$. As a result, the second e.p. starts at $3\Delta t$, similarly to what happens in the construction of the bound of Prop. 1.



B. Algorithm and complexity

There exist many polynomial-time algorithms for solving the maximum $s-u$-flow/minimum $s-u$-cut problem. Our problem has the specificity that the capacities/costs of the arcs are real numbers, and some have to be very high (infinite). These features make augmenting-path algorithms unsuitable [11]. For our problem, one can compute that the number of nodes is $n = \mathscr{O}(N \cdot N_t)$ and the number of arcs is $m = \mathscr{O}(N \cdot N_t + N + n_{pred}N_t)$ where N is the number of energy phases, N_t is the number of time intervals in the discretization, and n_{pred} is the number of precedence relations between appliances. Hence, based on the comparison of complexities in [11, Fig 7.19], we chose to implement the highest-label variant of the preflow-push algorithm (a.k.a. push-relabel). It has a worst case complexity of $\mathscr{O}(n^2\sqrt{m})$ for n the number of nodes and m the number of arcs. Our implementation is based on the survey [12] and uses the global-relabeling and gap-relabeling heuristics. Indeed, we could observe their substantial benefits on the computation time of the algorithm on our type of networks. Even more advanced algorithms with better theoretical bounds exist and could be implemented with the use of advanced data structures [13], [14].

As a comparison, [7] uses a formulation involving type 2 special ordered set (SOS2) constraints, which impose that at most two of the variables in a vector are nonzero, and that these variables must be consecutive. Such formluations require a specialized branching technique. The number of variables on which one has to branch is proportional to the number of energy phases and the number of break-points in the cost. We will observe in the simulations that its computation time can explode when the number of energy phases grows.

V. SIMULATIONS AND COMPARISON

We now test and compare our method. We use `cplexmilp` (with SOS2 constraints) through a mex-file in Matlab for solving the continuous-time formulation from [7]. We implemented our algorithm in C++ to also use it in Matlab through a mex-file. The graph class we implemented is based on [15] and parts of the implementation are inspired from the `push_relabel_max_flow` method of the Boost Graph Library [16]. Our code is available at perso.uclouvain.be/raphael.jungers/contents/App_Sched.zip. All computations were performed on a 64bits Windows system with an Intel Core i7-3770, 3.40GHz and 16.0 Gb of RAM.

A. Computation time and optimality versus discretization

1) Setup: We use an instance introduced and solved in [7] to compare our optimal cost with the one obtained there in continuous-time and compare the solving times, both as a function of our level of discretization.

We look at the schedule of three appliances; a dishwasher with 6 energy phases, a washing machine with 8 energy phases and a dryer with a single energy phase. The specifications of these appliances are detailed in Table III. The user preferences are that the dryer cannot start before the washing machine has finished, both must run between 00:00 and 23:00, and the dishwasher should run between 19:00 and 24:00. The cost of the energy is hourly based and corresponds to the day-ahead price in Sweden on January 5, 2010. It is pictured on Fig. 3[1].

We solve this instance with the continuous-time formulation and our formulation for values of N_t ranging from $24 \cdot 30$ to $24 \cdot 120$ by steps of 5.

TABLE III

SPECIFICATIONS OF THE APPLIANCES.

Dishwasher:	e.p. 1	e.p. 2	e.p. 3	e.p. 4	e.p. 5	e.p. 6		
Er_{1j} [Wh]	16	751.2	17.3	1.6	572.6	1.7		
T_{1j} [h]	0.248	0.535	0.1683	0.072	0.305	0.873		
$\underline{D}_{1j}$ [h]	0	0	0	0	0			
$\overline{D}_{1j}$ [h]	0.083	0.083	0.083	0.083	0.083			
Washing machine:	e.p. 1	e.p. 2	e.p. 3	e.p. 4	e.p. 5	e.p. 6	e.p. 7	e.p. 8
Er_{2j} [Wh]	118	5.5	2054.9	36.6	18	18	17	78
T_{2j} [h]	0.433	0.11	0.995	0.332	0.167	0.173	0.172	0.33
$\underline{D}_{2j}$ [h]	0	0	0	0	0	0	0	
$\overline{D}_{2j}$ [h]	0.167	0.167	0.167	0.167	0.167	0.167	0.167	
Dryer:	e.p. 1							
Er_{3j} [Wh]	2426.3							
T_{2j} [h]	2.01							

Fig. 3. Day-ahead electricity price in Sweden on January 5th, 2010. Data from Nordpool Spot (www.nordpoolspot.com).

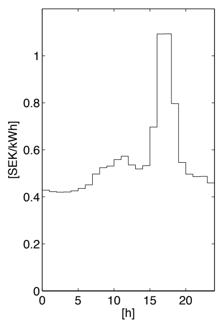
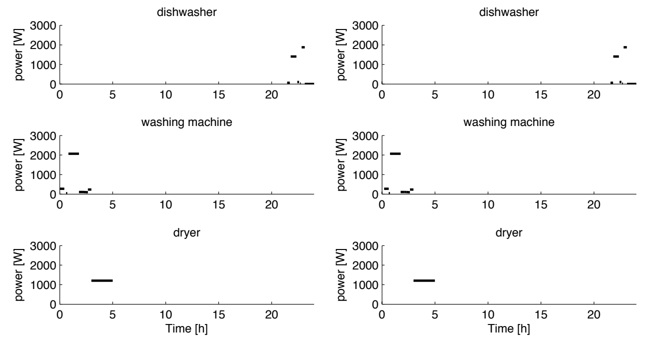


Fig. 4. Left: optimal schedule from [7]. Right: optimal schedule from discretized formulation with $\Delta_T = 1/50$h.



2) Results and observations: The optimal continuous-time schedule derived in [7] is pictured on the left of Fig. 4, it has a cost of 2.6755 SEK, the optimal schedule obtained from our discretized formulation with $\Delta_T = 1/50$ is pictured on the right of Fig. 4, it has a cost of 2.6773SEK, i.e., 100.0703 percent of the optimal continuous-time cost.

Fig. 5 shows the comparison of the computation time (left) and the optimal value (right). We observe that the min-cut

[1]Given the data, we use Swedish Kronor (SEK). On January 5, 2010 1SEK was worth 0.14USD.

Fig. 5. Comparisons as a function of the discretization, circles correspond to the min-cut formulation and lines to the continuous-time values (constant). Left: comparison of the total computation time (from plain data to solution), right: optimal values.

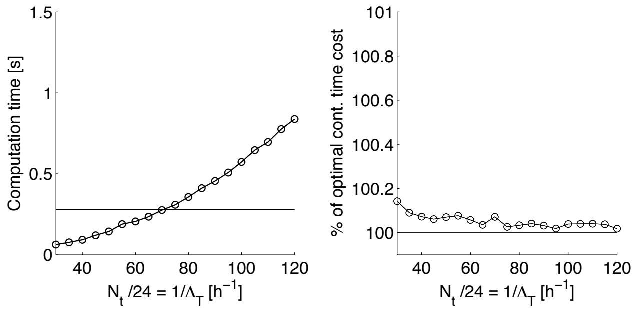
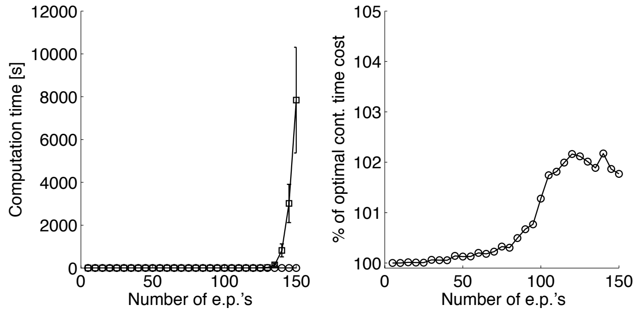


Fig. 6. Comparison as a function of the number of ordered jobs, circles correspond to the min-cut formulation, squares to the continuous-time formulation. Left: comparison of the computation time (average and standard deviation on 10 trials), right: max suboptimality of the discrete-time optimal among trials.



formulation is faster than the continuous-time formulation until $N_t \approx 24 \cdot 70$, which corresponds to a discretization period of less than one minute. As the complexity analysis predicted, the evolution is slow with respect to N_t and the computation-time of our formulation stays low (below 1sec) even for a finer discretization. Regarding the optimal value, we see that on this instance of the problem the cost of discretizing is below 0.1% of the optimal cost for $N_t = 24 \cdot 40$ ($\Delta_T = 1.5$min) and after. This shows that very little is lost from discretizing the starting-times. We computed the bound (5) for this case and found it to be $22.5021\Delta_T = 22.5021\frac{24}{N_t}$. Hence, one can compute that to ensure from this bound that the discretized solution is no more than $0.1SEK$ larger than the optimal continuous-time solution, we should take $N_t \geq 226 \cdot 24 = 5424$. For $N_t = 24 \cdot 50$ the bound is 0.45SEK but the real additional cost from discretizing is only 0.0024SEK. So, on this instance, our method performs much better than the worst case scenario provided by Prop. 1.

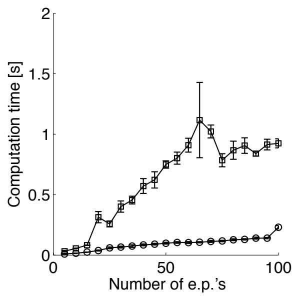
B. Computation time and optimality versus number of energy phases

1) Setup: We compare the solving time and optimality of the solution when the number of ordered energy phase grows. That is, we start with one appliance with 5 energy phases and solve for a growing number of energy phases in one appliance.

We add energy phases by five. More precisely, at each step we add five more energy phases, with a duration equal to $1, 2, 3, 4$ and 5 minutes, minimum delays of 1 minute and maximum delays are 60 and 30 minutes alternatively. The total energy requirements are uniformly distributed on $[0, 500]$ Wh. The cost is the one from Fig. 3. We performed 10 such trials and plotted the average time of computation and the standard deviation on the left of Fig. 6. For each number of jobs, the right of Fig. 6 shows the maximum, among the trials, of the optimal cost obtained as a percentage of the continuous-time optimal cost. We discretized with $N_t = 40 \cdot 24$ ($\Delta_T = 1.5$ min) for the min-cut formulation.

2) Results and observations: Regarding the computation time, one can observe that the continuous time formulation becomes intractable from circa 130 energy phases and more. That is due to the branch and bound algorithm of the SOS2 constraints. Having seen that explosion, let us zoom in to compare the computation times before it. This is pictured on Fig. 7. It shows that from the beginning the min-cut formulation is faster.

Fig. 7. Zoom in Fig. 6, left.



Regarding the optimal value with discrete starting-times as compared to the optimal continuous-time cost, we observe that the error grows with the number of energy phases. This is not surprising as a long sequence of ordered energy phases might propagate the cost of the discretization as shown in Prop. 1. However, the error stays below 3% of the cost even with 150 energy phases.

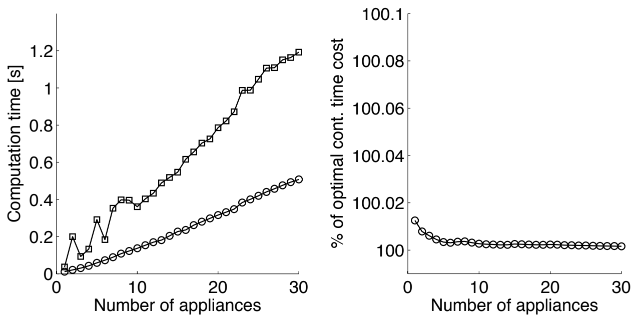
C. Computation time and optimality versus number of appliances

1) Setup: We compare the computation time and optimality when the number of appliances to schedule grows but, without precedence relations between them. This is different from Section V-B as for a same total number of energy phases the maximum size of an ordered sequence stays constant and small here. Moreover, one can see that a problem with k unrelated appliances is equivalent to k problems with one appliance each.

We grow the number of appliances by adding one appliance with five energy phases at each step. The energy phases added have a random duration distributed uniformly on $[0, 20]$ minutes. Their minimum delays are $[2, 2, 2, 1]$ minutes, maximum delays are 60 and 30 minutes alternatively. Their total energy requirement is uniformly distributed on $[0, 500]$

Wh. The appliances can be started and ended anytime during the day. The cost is the one from Fig. 3. We performed 10 such trials and plotted the average time of computation on the left of Fig. 8. For each number of appliances, the right of Fig. 8 shows the maximum, among the trials, of the optimal cost obtained as a percentage of the continuous-time optimal cost. We discretized with $N_t = 40 \cdot 24$ ($\Delta_T = 1.5$ min) for the min-cut formulation.

Fig. 8. Comparison as a function of the number of appliances, circles correspond to the min-cut formulation, squares to the continuous-time formulation. Left: comparison of the time to treat the data and total time (average on 10 trials), right: max suboptimality of the min-cut optimal among trials.



2) Results and observations: We observe that both computation times grow linearly in the number of unrelated appliances. This is consistent with the fact that solving the problem for k unrelated appliances is equivalent to solving k problems with one appliance each. We also see that the computation time of the min-cut formulation increases less steeply than that of the continuous time formulation.

Regarding the error, it is surprising that the maximum error among trials decreases with the number of appliances. Anyway, we see that the error always stays below 0.015% of the optimal continuous time cost for these numbers of appliances. This is much lower than in the case of long sequences of ordered energy phases, which is not surprising given our analysis of the discretization cost and its propagation.

VI. CONCLUSIONS

In this work, we studied how to facilitate Demand Response from the end-user side when dynamic pricing is implemented. Our context relies on real-time pricing, in the sense that the price profile is fixed for a certain future period, such as a day, and communicated to the users. In this context, we have presented, analyzed and tested a new way of expressing and solving the optimal scheduling of home-appliances.

The discretization of the problem that we formulate can be efficiently solved by finding a minimum-cut in a suitably defined digraph. The simulations show that the optimal cost of our discrete-time optimum is near-optimal and that they can be found with a reduced time of computation.

We believe that this kind of approach could be adapted to other scheduling problems in the smart-grid, and can for example open the way towards joint scheduling of appliances at a large scale.

ACKNOWLEDGMENT

We thank Kin Cheong Sou and the Automatic Control lab at KTH for introducing us to the problem and its challenges, and for having kindly shared their data and codes.

REFERENCES

[1] V. S. K. M. Balijepalli, V. Padhan, S. A. Khaparde, and R. M. Shereef, "Review of demand response under smart grid paradigm," in *Innovative Smart Grid Technologies - India (ISGT India), 2011 IEEE PES*, 2011, pp. 236–243.

[2] S. Borenstein, M. Jaske, and A. Rosenfeld, "Dynamic pricing, advanced metering and demand response in electricity markets," UC Berkeley: Center for Study of Energy Markets, Tech. Rep., 2002.

[3] M. Roozbehani, M. A. Dahleh, and S. K. Mitter, "Volatility of Power Grids under Real-Time Pricing," *Power Systems, IEEE Transactions on*, vol. 27, no. 4, pp. 1926–1940, 2012.

[4] T. Bapat, N. Sengupta, S. K. Ghai, V. Arya, Y. B. Shrinivasan, and D. Seetharam, "User-sensitive Scheduling of Home Appliances," in *Proceedings of the 2nd ACM SIGCOMM workshop on Green networking*, ser. GreenNets '11. New York, NY, USA: ACM, 2011, pp. 43–48.

[5] T. Kim and H. Poor, "Scheduling Power Consumption With Price Uncertainty," *Smart Grid, IEEE Transactions on*, vol. 2, no. 3, pp. 519–527, 2011.

[6] D. O'Neill, M. Levorato, A. Goldsmith, and U. Mitra, "Residential Demand Response Using Reinforcement Learning," in *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*, 2010, pp. 409–414.

[7] K. C. Sou, M. Kördel, J. Wu, H. Sandberg, and K. H. Johansson, "Energy and CO_2 Efficient Scheduling of Smart Home Appliances," in *2013 European Control Conference*, Zürich, Switzerland, July 2013.

[8] R. H. Möhring, A. S. Schulz, F. Stork, and M. Uetz, "Solving project scheduling problems by minimum cut computations," *Management Science*, pp. 330–350, 2003.

[9] K. C. Sou, J. Weimer, H. Sandberg, and K. H. Johansson, "Scheduling Smart Home Appliances Using Mixed Integer Linear Programming ," in *50th IEEE CDC-ECC*, Orlando, USA, December 2011.

[10] L. A. Wolsey, *Integer Programming*, ser. Discrete Mathematics and Optimization. Wiley-Interscience, 1998.

[11] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Networks Flows: Theory, Algorithms, and Applications*. Prenctice Hall, 1993.

[12] B. V. Cherkassky and A. V. Goldberg, "On Implementing the Push-Relabel Method for the Maximum Flow Problem," *Algorithmica*, vol. 19, pp. 390–410, 1994.

[13] A. V. Goldberg and R. E. Tarjan, "A New Approach to the Maximum Flow Problem," *Journal of the ACM*, vol. 35, pp. 921–940, 1988.

[14] J. B. Orlin, "Max flows in O(nm) time or better," in *Proceedings of the 2013 Symposium on the Theory of Computing*, 2013, pp. 765–774.

[15] R. Sedgewick, *Algorithms in C++, Part 5: Graph Algorithms, Third Edition*. Addison-Wesley Professional, 2001.

[16] J. G. Siek, L.-Q. Lee, and A. Lumsdaine, *Boost Graph Library, The User Guide and Reference Manual*. Addison-Wesley Professional, 2001.