Competitive algorithms for demand response management in smart grid

Vincent Chau, Shengzhong Feng, Nguyen Kim Thang

Competitive algorithms for demand response management in a smart grid

Vincent Chau[1] · **Shengzhong Feng**[2,3] · **Nguyễn Kim Thắng**[4]

Abstract
We consider a scheduling problem that abstracts a model of demand response management in a smart grid. We investigate the problem with a set of unrelated machines, and each job j (representing a client demand) is characterized by its release date, and its power request function expressing its request demand at specific times. Each machine has an energy power function, and the energy cost incurred at a time depends on the load of the machine at that time. The goal is to find a non-migrative schedule that minimizes the total energy. We give a competitive algorithm for the problem in the online setting where the competitive ratio depends (only) on the power functions of machines. In the setting with typical energy function $P(z) = z^\nu$, the algorithm is $\Theta(\nu^\nu)$-competitive, which is *optimal* up to a constant factor. Our algorithm is *robust* in the sense that the guarantee holds for arbitrary request demands of clients. This enables flexibility on the choices of clients in shaping their demands—a desired property in a smart grid. We also consider a particular case in the offline setting in which jobs have unit processing time, constant power request, and identical machines with energy function $P(z) = z^\nu$. We present a 2^ν-approximation algorithm for this case.

Keywords Smart grid · Scheduling · Approximation algorithm

1 Introduction

Electrical smart grid is one of the significant challenges in the twenty-first-century (US Department of Energy 2009). It is a network of the electricity distribution that promotes the traffic information between producers and consumers to adjust the electricity flow in real time, i.e., it aims to improve the journey of electricity through information and communication technologies in contrast to the traditional power system. It has been raised (Chen et al. 2013) that in the US power grid, 10% of all generation assets and 25% of distribution infrastructure are required for less than 400 hours per year, which represents roughly 5% of the time (US Department of Energy 2009). A smart grid is a power grid system that optimizes the efficiency of the power generation, distribution, and consumption, and eventually the storage, of the energy to coordinate the electric network, from the production to the consumer. It can be noticed that the power grid may not be efficient during the peak demand hour if the management of the smart grid is not well handled. Indeed, the cost of electricity production can be high if there is a high demand punctually, and the electricity suppliers may charge the consumer according to the generation cost. Therefore, the cost

Vincent Chau is supported by the national key research and development program of China under grant No. 2019YFB2102200, by the Fundamental Research Funds for the Central Universities, and by the CAS President's International Fellowship Initiative n° 2020FYT0002, 2018PT0004. Nguyen Kim Thang is supported by the ANR project OATA n° ANR-15-CE40-0015-01, Hadamard PGMO and DIM RFSI.

✉ Vincent Chau
vincentchau@seu.edu.cn

Shengzhong Feng
fengsz@nsccsz.cn

Nguyễn Kim Thắng
thang@ibisc.univ-evry.fr

[1] School of Computer Science and Engineering, Southeast University, Nanjing, China

[2] Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China

[3] National Supercomputing Centre in Shenzhen, Shenzhen, China

[4] IBISC, University Paris Saclay, Evry, France

of electricity can be different over time; intuitively, we have a lower price during off-peak hours and a higher price during peak hours. That is why Hamilton and Gulhar (2010) investigated the *demand response management* to overcome this problem. The goal of each user is to minimize his own cost by requesting the electricity during off-peak hours and reduce the peak load while satisfying his demand. Thus, *demand response management* is mainly beneficial to consumers.

It can be seen as a scheduling problem. Each user is a job with the same release date and deadline in which the job cannot be scheduled before the release date, nor after the deadline. Furthermore, a user is defined by an electricity demand over time, which can also be represented in the scheduling problem. Finally, we have a cost that will be charged to users depending on the load at each moment. The goal is to minimize the total cost while satisfying all demands. A more formal definition is given in the next section.

In this paper, we consider a general online scheduling problem which models the demand response management, and we design algorithms toward the following main purposes of a smart grid:

- Optimizing energy consumption.
- Enabling customer choice and letting them react rationally.

1.1 Model definition

We consider the following scheduling problem. We are given m unrelated machines and a set of n jobs. Here, machines represent different resources in a smart grid or different electrical sub-networks. Each job represents the demand of a client, and the client customizes the demand. Specifically, each job j is characterized by its release date r_j and an *arbitrary* power request function $h_{i,j,k} : \mathbb{N} \to \mathbb{R}^+$, meaning that if a job j starts at time k in machine i, then its request demand at time t is $h_{i,j,k}(t)$. We denote the *execution* of job j as $s_{i,j,k}$ if job j is processed in machine i with the starting time k. In the problem, *migration* of jobs between machines is not allowed. In other words, a job must be executed in exactly one machine since, in the case of migration, the communications and storage/reloading data of jobs are costly. Given a schedule (executions of all jobs), the total *load* at time t in machine i is $\sum_{j,k} h_{i,j,k}(t)$ where the sum is taken over all job executions in machine i. The total *energy* is defined as $\sum_i \sum_t P_i\left(\sum_{j,k} h_{i,j,k}(t)\right)$ where P_i is an energy power function of machine i. Typically, $P_i(z) = z^{\nu_i}$ for some constant parameter $\nu_i \geq 1$. In the problem, we consider P_i as *arbitrary* non-decreasing functions, and possibly non-convex. The goal is to find a feasible schedule that minimizes the total energy consumption over all times.

In the paper, we consider both offline and online settings. In the offline setting, the scheduler has full knowledge of all parameters. In contrast, in the online setting, jobs arrive over time, and the scheduler is aware of jobs (and their parameters) only at their arrival time. The online setting is appropriate for the dynamic nature of demand response management. The presented model encompasses the previous ones (Koutsopoulos and Tassiulas 2011) in the literature. In the latter, jobs have release date r_j, deadline d_j, processing time $p_{i,j}$ and jobs have to be processed non-preemptively. The power request of a job j is some constant h_j during its non-preemptive execution. The model captures it by defining

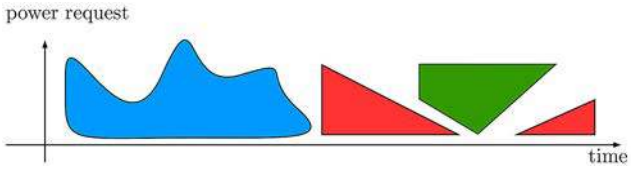

Fig. 1 Example of a schedule with arbitrary power requests during execution of jobs. Each color corresponds to a different job. Here, the red job has two disjoint parts. Note that more than one job can be scheduled at the same time since we are only interested in the total power request over time.

$$h_{i,j,k}(t) = \begin{cases} h_j & \text{if } r_j \leq k \leq d_j - p_{ij} \text{ and } t \in [k, k + p_{ij}], \\ 0 & \text{if } r_j \leq k \leq d_j - p_{ij} \text{ and } t \notin [k, k + p_{ij}], \\ \infty & \text{otherwise.} \end{cases} \tag{1}$$

Geometrically, in the model shown in Eq. (1), each job corresponds to a rectangle, and the problem essentially consists of packing rectangles to minimize the total energy. In this case, the power request is constant from the beginning to the end of the request. For short, we call the model defined by Eq. (1) *rectangle scheduling*. We investigate this model in Sect. 3 in the offline context.

We also consider a generalized version in which there is no condition on the demand (i.e., energy request) of jobs, and clients can specifically customize the demands. Geometrically, each job in our model has an *arbitrary* form, which represents different power requests during its execution (see Fig. 1). Note that each job is not necessarily continuous, for instance, a task may need to be interrupted and wait a fixed amount of time before resuming. Additionally, we consider that jobs are scheduled non-preemptively, i.e., once a job starts, its power request follows the function $h_{i,j,k}$. Hence, the model offers flexible choices to clients along the line of smart grid's purposes. This model will be studied in Sect. 2 in the online context.

Table 1 Summary of competitive ratios (resp. approximation ratio) in the rectangle scheduling model with power function z^ν for a *single* machine

	Processing time p_j	Power request h_j	Prior best-known results	Our result
Online	Unit	uniform (i.e., $h_j = h_{j'}$)	$\min\{(4\nu)^\nu/2 + 1, 2^\nu(8e^\nu + 1)\}$ Liu et al. (2020)	
		Arbitrary	$2^\nu(8e^\nu + 1)$ Liu et al. (2020)	
	Arbitrary	Arbitrary	$O\left(\log^\nu\left(\frac{p_{\max}}{p_{\min}}\right)\right)$ Liu et al. (2020)	$\Theta(\nu^\nu)$
			$\Omega(\nu^\nu)$ Liu et al. (2020)	
Offline	Unit	Unit	Optimal polynomial time Burcea et al. (2016)	
		Arbitrary	$2^{\nu+1}$ Liu et al. (2020)	2^ν

Our result holds for *unrelated machines*

1.2 Related works

In this section, we summarize related works in the model of *rectangle scheduling*, which, to the best of our knowledge, is the only one that has been studied so far.

Koutsopoulos and Tassiulas (2011) formulated the rectangle scheduling model where the cost function is piecewise linear. They show that the problem is NP-hard, and Burcea et al. (2016) showed the NP-hardness of the general problem where the cost function is convex. In the offline setting, Burcea et al. (2016) gave a polynomial-time algorithm for the case of unit height (i.e., unit power request) and unit width (i.e., duration of request). In the same work, they showed several structural lemmas and proposed an online algorithm that achieves a constant competitive ratio for some particular cases. Furthermore, in the full version (see Liu et al. 2016), Liu et al. showed that the offline case, where jobs have unit processing time but with arbitrary power request, admits a $2^{\nu+1}$-approximation algorithm which is based on the results of the dynamic speed scaling problem (Albers 2010; Bell and Wong 2015; Yao et al. 1995).

In the online setting, Feng et al. (2015) proposed a simple greedy algorithm which is 2-competitive for the unit case and the power function is z^2. However, Liu et al. (2020) showed that the greedy algorithm is in fact at least 3-competitive by providing a counterexample. Liu et al. (2019) showed that the greedy algorithm is optimal but left the exact competitive ratio as open. Liu et al. (2020) considered the single machine setting in which they presented an online $\Theta\left(\log^\nu\left(\frac{p_{\max}}{p_{\min}}\right)\right)$-competitive algorithm where $p_{\min} = \min_j\{p_j : p_j > 0\}$ and $p_{\max} = \max_j p_j$. This is the best-known algorithm (even in offline setting) where jobs have arbitrary width and arbitrary height and the power energy function is z^ν. Furthermore, for special cases of jobs with unit processing time, Liu et al. (2020) also gave competitive algorithms. A summary of the results can be found in Table 1.

Besides, Salinas et al. (2013) considered a multi-objective problem to minimize energy consumption cost and maximize some utility that can be the profit for the operator as well as for the clients. On the other hand, a related problem is to manage the load by considering different prices of electricity over time (Fang et al. 2016; Maharjan et al. 2013). Recent surveys of the area can be found in (Hamilton and Gulhar 2010; Lui et al. 2010; Alford et al. 2012).

1.3 Our contribution and approaches

In this paper, we investigate the online and offline aspects of the problem.

Online setting. The main result of the paper is a competitive algorithm for the problem in the online setting, where the competitive ratio is characterized by a notion called *smoothness* (Roughgarden 2015; Thang 2020) of the machine energy power functions. Informally, the algorithm assigns and executes each job that arrives on a machine in such a way that minimizes the marginal increase in the total cost.

In designing a competitive algorithm for the problem, we consider a primal–dual approach. The main difficulty in proving the performance of the algorithm is that all known LPs have an unbounded integrality gap, even for the particular case of rectangle scheduling. Intuitively, the drawback of all known LPs is that in the optimal fractional solution, jobs are fractionally assigned to machines. In contrast, in the optimal integer solution, migration of jobs is not allowed. To bypass this obstacle, we consider the primal–dual framework based on configuration linear programs in (Thang 2020). The framework is presented to reduce the integrality gap and also to study problems with nonlinear, non-convex objective functions. The approach is particularly useful since the energy power functions are nonlinear. Employing the techniques from Thang (2020), we derive a greedy algorithm with competitive ratio characterized by the notion of *smoothness*, which is defined as follows.

Definition 1 A function $f : \mathbb{R}^+ \to \mathbb{R}^+$ is (λ, μ)-*smooth* if for any sets of nonnegative numbers $A = \{a_1, \ldots, a_n\}$ and

$B = \{b_1, \ldots, b_n\}$, the following inequality holds:

$$\sum_{i=1}^{n}\left[f\left(a_i + \sum_{j=1}^{i} b_j\right) - f\left(\sum_{j=1}^{i} b_j\right)\right] \leq \lambda \cdot f\left(\sum_{i=1}^{n} a_i\right) + \mu \cdot f\left(\sum_{i=1}^{n} b_i\right)$$

A set of cost functions $\{f_e : e \in \mathcal{E}\}$ is (λ, μ)-*smooth* if every function f_e is (λ, μ)-*smooth*.

Specifically, in the problem, assuming that all energy power functions are (λ, μ)-*smooth* for some $\lambda > 0$ and $0 < \mu < 1$, our algorithm is $\lambda/(1-\mu)$-competitive. For energy power functions of forms $P_i(z) = z^{\nu_i}$, they are $\left(O(\nu^{\nu-1}), \frac{\nu-1}{\nu}\right)$-smooth where $\nu = \max_i \nu_i$. That leads to the competitive ratio $O(\nu^{\nu})$ which improves upon the best-known $\Theta\left(\log^{\nu}\left(\frac{p_{\max}}{p_{\min}}\right)\right)$-competitive algorithm where $p_{\min} = \min_j\{p_j : p_j > 0\}$ and $p_{\max} = \max_j p_j$. Our competitive ratio is not only independent on the jobs' parameters, but it is indeed *optimal* up to a constant factor. The matching lower bound is given by (Liu et al. 2020, Theorem 9) for a single machine in the rectangle scheduling model. In particular, Liu et al. (2020) gave a lower bound which is $\frac{1}{3}\left(\log \frac{p_{\max}}{p_{\min}}\right)^{\nu}$ where $\log \frac{p_{\max}}{p_{\min}} = \nu$.

Our greedy algorithm has several interesting features for a smart grid. First, the algorithm is simple and easy to implement, which makes it practically appealing. Note that despite the simplicity of our algorithm, no bounded competitive ratio has been known even for the rectangle scheduling model. Secondly, the algorithm performance guarantee holds for jobs with arbitrary varying power requests (arbitrary forms). Apart from answering open questions raised by Liu et al. (2020), it is particularly useful for demand response management. Once the algorithm is publicly given, and clients are charged accordingly to the marginal increase in the total energy cost, clients can *arbitrarily* customize their demand to minimize their payment. This property is desirable since it enables the clients to react rationally. On the side of the smart grid management, no modification in the algorithm is needed while always maintaining the competitiveness (optimality in case of typical energy functions).

Offline setting. In the offline setting, we consider the rectangle scheduling model when jobs have unit processing time. We give an 2^{ν}-approximation algorithm, which improves upon the $2^{\nu+1}$-approximation algorithm by Liu et al. (2016) in two aspects. First, it slightly improves the competitive ratio. Secondly, our result holds for multiple (identical) machine environment. Our algorithm makes use of the approximation algorithm for scheduling problems with convex norm objective functions given by Azar and Epstein (2005). The latter is designed by solving a convex relaxation and round to an integer solution using the Lenstra–Shmoys–Tardos scheme (Lenstra et al. 1990).

2 A competitive online algorithm

Formulation. In the model, the execution of a job is specified by two parameters: (1) a machine in which it is executed and (2) a starting time. Note that these parameters fully represent the demand of a job, including the power request at any time t during its execution. Formally, we denote the *execution* of job j as $s_{i,j,k}$ if job j is processed in machine i with the starting time k. Recall that if the execution of a job j is $s_{i,j,k}$, then the request demand of the job at time t is $h_{i,j,k}(t)$. Let $\mathcal{S}_j$ be a set of feasible executions of job j. For example, in the rectangle scheduling model, $\mathcal{S}_j$ consists of $s_{i,j,k}$ for all machines i and starting time k such that $r_j \leq k \leq d_j - p_{ij}$. As the set of machines and times[1] are finite, so is the set $\mathcal{S}_j$ for every job j. Let $x_{i,j,k}$ be a variable indicating whether the execution of job j is $s_{i,j,k} \in \mathcal{S}_j$. We say that A is a *scheduling configuration* (configuration in short) in machine i if A is a feasible schedule of a subset of jobs, i.e., the cost of a feasible schedule is a finite value. Specifically, A consists of tuples (i, j, k) meaning that the execution of job j is $s_{i,j,k}$. Note that, given a feasible scheduling configuration A, for each job j there exists exactly one tuple of form $(\cdot, j, \cdot)$. For a scheduling configuration A and machine i, let $z_{i,A}$ be a variable such that $z_{i,A} = 1$ if and only if for every tuple $(i, j, k) \in A$, we have $x_{i,j,k} = 1$. In other words, given a scheduling configuration A, $z_{i,A} = 1$ if and only if for every job j such that $(i, j, k) \in A$ for some i and k, the execution of job j is $s_{i,j,k}$. (Additionally, $z_{i,A} = 0$ if there exists a tuple $(i, j, k) \in A$, but the execution of job j is not $s_{i,j,k}$). Given a scheduling configuration A, let $A(t)$ be the load (height) of the corresponding schedule at time t. We denote the energy cost of a configuration A of machine i as $c_i(A) := \sum_t P_i(A(t))$. We consider the following formulation and the dual of its relaxation.

$$\begin{aligned}
\text{Primal}: \min \sum_{i,A} c_i(A) z_{i,A} & \\
\sum_{i,k:s_{i,j,k}\in\mathcal{S}_j} x_{i,j,k} = 1 & \qquad \forall j \\
\sum_{A:(i,j,k)\in A} z_{i,A} = x_{i,j,k} & \qquad \forall i, j, k \\
\sum_{A} z_{i,A} = 1 & \qquad \forall i \\
x_{i,j,k}, z_{i,A} \in \{0, 1\} & \qquad \forall i, j, k, A
\end{aligned}$$

[1] For convenience, we consider schedules up to a time T, which can be arbitrarily large but finite

$$\text{Dual}: \max \sum_j \alpha_j + \sum_i \gamma_i$$
$$\alpha_j \le \beta_{i,j,k} \qquad \forall i, j, k$$
$$\gamma_i + \sum_{(i,j,k)\in A} \beta_{i,j,k} \le c_i(A) \qquad \forall i, A$$

In the primal, the first constraint guarantees that a job j has to be processed by some feasible execution (in some machine). The second constraint ensures that if job j follows the execution $s_{i,j,k}$, then in the solution, the scheduling configuration of machine i must contain the tuple (i, j, k) corresponding to execution $s_{i,j,k}$. The third constraint says that in the solution, there is always a scheduling configuration (possibly empty set) associated with a machine i.

Algorithm. We first interpret the dual variables and dual constraints, and we derive useful observations for a competitive algorithm. Variable α_j represents the increase in energy to the arrival of a job j. Variable $\beta_{i,j,k}$ stands for the marginal energy if job j follows execution $s_{i,j,k}$. By this interpretation, the first dual constraint indicates the greedy behavior of an algorithm. That is, if a new job j is released, select an execution $s_{i,j,k} \in \mathcal{S}_j$ that minimizes the marginal increase in the total energy. Formally, let A_i^* be the set of current schedule of machine i, and initially, $A_i^* \leftarrow \emptyset$ for every machine i. At the arrival of job j, select an execution $s_{i^*,j,k^*} \in \mathcal{S}_j$ such that

$$s_{i^*,j,k^*} \in \arg\min_{s_{i,j,k}\in\mathcal{S}_j} \left[c_i(A_i^* \cup s_{i,j,k}) - c_i(A_i^*)\right]$$

or equivalently,

$$s_{i^*,j,k^*} \in \arg\min_{s_{i,j,k}\in\mathcal{S}_j} \sum_t \left[P_i\left(A_i^*(t) + h_{i,j,k}(t)\right) - P_i\left(A_i^*(t)\right)\right]$$

where $(A_i^* \cup s_{i,j,k})$ is the current schedule with additional execution $s_{i,j,k}$ of job j. Note that in configuration $(A_i^* \cup s_{i,j,k})$, the load at time t in machine i is $P_i\left(A_i^*(t)+h_{i,j,k}(t)\right)$. Then, assign job j to machine i^* and process it according to the corresponding execution of s_{i^*,j,k^*}.

Dual variables. Assume that all energy power functions P_i are (λ, μ)-smooth for some fixed parameters $\lambda > 0$ and $\mu < 1$, then we construct a dual feasible solution in the following way. Let $A_{i,\prec j}^*$ be the scheduling configuration of machine i (due to the algorithm) prior to the arrival of job j. Define α_j as $1/\lambda$ times the increase in the total cost due to the arrival of job j. In other words, if the algorithm selects the execution s_{i^*,j,k^*} for job j, then

$$\alpha_j = \frac{1}{\lambda}\left[c_{i^*}(A_{i^*,\prec j}^* \cup s_{i^*,j,k^*}) - c_{i^*}(A_{i^*,\prec j}^*)\right]$$
$$= \frac{1}{\lambda}\sum_t \left[P_{i^*}\left(A_{i^*,\prec j}^*(t) + h_{i^*,j,k^*}(t)\right) - P_{i^*}\left(A_{i^*,\prec j}^*(t)\right)\right]$$

For each machine i and job j, we set

$$\beta_{i,j,k} = \frac{1}{\lambda}\left[c_i(A_{i,\prec j}^* \cup s_{i,j,k}) - c_i(A_{i,\prec j}^*)\right]$$
$$= \frac{1}{\lambda}\sum_t \left[P_i\left(A_{i,\prec j}^*(t) + h_{i,j,k}(t)\right) - P_i\left(A_{i,\prec j}^*(t)\right)\right].$$

Finally, for every machine i, we define the dual variable

$$\gamma_i = -\frac{\mu}{\lambda} c_i(A_i^*)$$

where A_i^* is the schedule on machine i (at the end of the instance).

Lemma 1 *The dual variables defined above are feasible.*

Proof By the definition of dual variables, the first constraint reads

$$\frac{1}{\lambda}\left[c_{i^*}(A_{i^*,\prec j}^* \cup s_{i^*,j,k^*}) - c_{i^*}(A_{i^*,\prec j}^*)\right]$$
$$\le \frac{1}{\lambda}\left[c_i(A_{i,\prec j}^* \cup s_{i,j,k}) - c_i(A_{i,\prec j}^*)\right]$$

This inequality follows immediately from the choice of the algorithm.

We now show that the second constraint holds. Fix a machine i and an arbitrary configuration A on machine i. The corresponding constraint reads

$$-\frac{\mu}{\lambda}c_i(A_i^*) + \frac{1}{\lambda}\sum_{(i,j,k)\in A}\left[c_i(A_{i,\prec j}^* \cup s_{i,j,k}) - c_i(A_{i,\prec j}^*)\right]$$
$$\le c_i(A)$$
$$\Leftrightarrow \sum_{(i,j,k)\in A}\left[c_i(A_{i,\prec j}^* \cup s_{i,j,k}) - c_i(A_{i,\prec j}^*)\right]$$
$$\le \lambda c_i(A) + \mu c_i(A_i^*)$$
$$\Leftrightarrow \sum_{(i,j,k)\in A}\sum_t \left[P_i(A_{i,\prec j}^*(t) + h_{i,j,k}(t)) - P_i(A_{i,\prec j}^*(t))\right]$$
$$\le \lambda \sum_t P_i(A(t)) + \mu \sum_t P_i(A_i^*(t)) \qquad (2)$$

where $A_{i,\prec j}^*(t)$ is the load (height) of machine i (due to the algorithm) at time t before the arrival of job j.

Observe that $A^*_{i,\prec j}(t)$ is the sum of power requests (according to the algorithm) at time t of jobs assigned to machine i prior to job j. As the power function P_i is (λ, μ)-smooth, for any time t we have

$$\sum_{(i,j,k)\in A}\left[P_i\left(A^*_{i,\prec j}(t)+h_{i,j,k}(t)\right)-P_i\left(A^*_{i,\prec j}(t)\right)\right] \leq \lambda P_i\left(\sum_{(i,j,k)\in A} h_{i,j,k}(t)\right)+\mu P_i\left(A^*_i(t)\right)$$

Summing over all times t, Inequality (2) holds. Therefore, the lemma follows. □

We are now ready to prove the main theorem.

Theorem 1 *If all energy power functions are (λ, μ)-smooth, then the algorithm is $\lambda/(1-\mu)$-competitive. In particular, if $P_i(z) = z^{\nu_i}$ for $\nu_i \geq 1$, then the algorithm is $O(\nu^\nu)$-competitive where $\nu = \max_i \nu_i$.*

Proof By the definitions of dual variables, the dual objective is

$$\sum_j \alpha_j + \sum_i \gamma_i = \sum_i \frac{1}{\lambda} c_i(A^*_i) - \sum_i \frac{\mu}{\lambda} c_i(A^*_i) = \frac{1-\mu}{\lambda}\sum_i c_i(A^*_i)$$

Besides, the cost of the solution due to the algorithm is $\sum_i c_i(A^*_i)$. Hence, the competitive ratio is at most $\lambda/(1-\mu)$.

Particularly, energy power functions of forms $P_i(z) = z^{\nu_i}$ for $\nu_i \geq 1$ are $\left(O(\nu^{\nu-1}), \frac{\nu-1}{\nu}\right)$-smooth for $\nu = \max_i \nu_i$. In fact, the smoothness follows (smooth) inequalities in (Cohen et al. (2012)), which states: for $\nu > 1$ and for any sets of nonnegative numbers $A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_n\}$, it always holds that

$$\sum_{i=1}^{n}\left[\left(a_i+\sum_{j=1}^{i} b_j\right)^\nu-\left(\sum_{j=1}^{i} b_j\right)^\nu\right] \leq O(\nu^{\nu-1})\cdot\left(\sum_{i=1}^{n} a_i\right)^\nu+\frac{\nu-1}{\nu}\cdot\left(\sum_{i=1}^{n} b_i\right)^\nu$$

That implies the competitive ratio ν^ν of the algorithm for power functions $P_i(z) = z^{\nu_i}$. □

3 An approximation algorithm for unit processing time jobs

In this section, we investigate the offline case with a set of identical machines where jobs have unit processing time but different power requests on different machines. Note that this corresponds to the restricted model of *rectangle scheduling*. We consider typical energy power function $P(z) = z^\nu$ for every machine, and we assume that jobs need to be assigned to time slot. Burcea et al. (2016) showed that this problem is NP-hard by a reduction to the 3-partition problem even for the case where jobs have common release time and common deadline.

Let $\Theta = \cup_{j=1}^n \{r_j + a \mid a = -n, \ldots, n\} \cup_{j=1}^n \{d_j + a \mid a = -n, \ldots, n\}$ to be a set of time slots. We show that it is sufficient to consider only schedules in which jobs are processed within these time slots. In particular, this set contains $O(n^2)$ time slots and will help to design a polynomial-time approximation algorithm.

Lemma 2 *The schedules in which jobs start at a date in Θ are* dominant. *In other words, any schedule can be transformed into one in which jobs start at a date in Θ without increasing the cost.*

Proof It is sufficient to consider a single machine and show how to transform the schedule of the machine to the new one such that each job starts at a date in Θ without increasing the cost.

Let t be the first moment where jobs that are assigned to this time slot do not belong to Θ. We consider the maximal continuous interval from time slot t in which every time slot has at least one job that is assigned. If the considered interval is $[t, u)$, then the time slot $u + 1$ is idle.

First, we observe that the length of this interval is lower or equal to n. Indeed, in the worst case, each job is assigned to different time slots. We shift this interval, as well as the jobs, by one unit time to the right, i.e., after the shift, the interval will be $[t + 1, u + 1)$. Three cases may occur (see Fig. 2):

- we reach another job. We then consider the new maximal continuous interval and continue to shift it.
- we reach a deadline. The starting time of the interval must be in Θ since the length of the interval is at most n.
- none of the above cases, we continue to shift the interval to the right.

By this operation, we observe that the cost of the schedule remains the same because the costs of time slots are independent. By doing such modification, jobs are executed in the same way, with the same order, and with the same cost, the only difference is the time slots in which the jobs are executed. □

The main idea is to reduce the smart grid problem to the following *L_ν-norm problem*. In the latter, we are given a set $\mathcal{J}$ of n jobs and a set $\mathcal{M}$ of m unrelated machine. Each job $j \in \mathcal{J}$ has a processing time $p_{i,j}$ if it is assigned to machine i. We define the decision variable $y_{i,j} = 1$ if the job j is

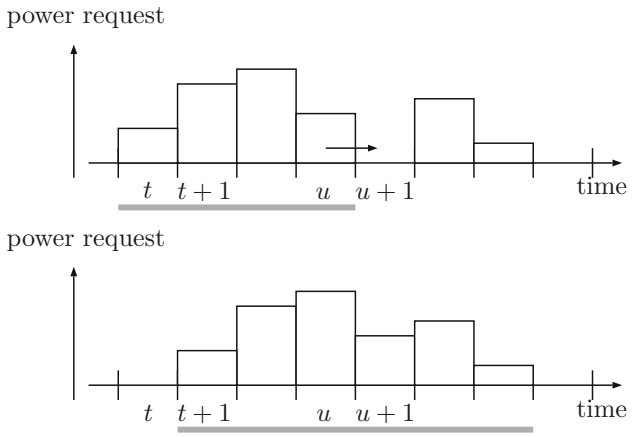

Fig. 2 Illustration of a shift of an interval. After the shift, the former interval meets another job. We then need to consider the continuous interval from time slot $t+1$. It corresponds to the first case in the proof of Lemma 2

assigned to machine i, and $y_{i,j} = 0$ otherwise. The goal is to minimize the following function:

$$\sqrt[\nu]{\sum_{i\in\mathcal{M}}\left(\sum_{j=1}^{n} y_{i,j}\, p_{i,j}\right)^{\nu}} \tag{3}$$

Lemma 3 *The problem of a smart grid with unit processing time jobs and identical machines can be polynomially reduced to the L_ν-norm minimization problem on unrelated machines.*

Proof By Lemma 2, there is a polynomial number of time-slots to which jobs can be assigned. We create a corresponding machine (i,t) for each time slot $t \in \Theta$ and each machine i. Similarly, we create a new job $j' \in \mathcal{J}$ (in L_ν-norm problem) which corresponds to job $j \in J$ (in the smart grid problem) in the following way:

$$p_{(i,t),j'} = \begin{cases} h_j & \text{if } t \in [r_j, d_j) \\ +\infty & \text{otherwise} \end{cases} \tag{4}$$

Given a schedule for the L_ν-norm problem with cost C, we show how to build a feasible schedule for the smart grid problem with cost C^ν.

For each job $j \in \mathcal{J}$ that is assigned to machine $(i,t) \in \mathcal{M}$ in the L_ν-norm problem, we schedule this job at the time slot t on machine i in the initial problem. By doing that, the load at any time slot t on machine i in the initial problem equals the load of the machine (i,t) in the L_ν-norm problem. Therefore, the constructed schedule for the initial problem has cost C^ν, where C is the cost of the schedule in the L_ν-norm problem. □

By Lemma 3, solving the smart grid problem with unit processing time jobs and identical machines is essentially solving the L_ν-norm problem. Hence, in our algorithm (Algorithm 1), we invoke the Azar–Epstein algorithm (Azar and Epstein 2005) to get an approximation algorithm for the latter. Roughly speaking, the Azar–Epstein algorithm consists of solving a relaxed convex program and rounding fractional solutions to integral ones using the standard scheme of Lenstra et al. (1990). Given a solution for the L_ν-norm problem, we reconstruct a feasible solution for the smart grid problem with an approximation ratio of 2^ν.

Algorithm 1 Approximation algorithm for the smart grid scheduling problem with unit processing time jobs and identical machines

1: $\Theta = \cup_{j=1}^{n}\{r_j + a \mid a = -n, \ldots, n\} \cup_{j=1}^{n} \{d_j + a \mid a = -n, \ldots, n\}$
2: Let $\Pi = \emptyset$ be the set of machines and $\mathcal{J} = \emptyset$ be the set of jobs
3: **for** each $t \in \Theta$ and each machine i **do**
4: Create a machine (i,t) and $\Pi \leftarrow \Pi \cup \{(i,t)\}$
5: **end for**
6: **for** each job j **do**
7: Create a new job j' with $p_{(i,t),j'} = h_j$ if $t \in [r_j, d_j)$, otherwise we have $p_{(i,t),j'} = +\infty$
8: $\mathcal{J} \leftarrow \mathcal{J} \cup \{j'\}$
9: **end for**
10: Apply the Azar-Epstein algorithm (Azar and Epstein 2005) on instance $(\Pi, \mathcal{J})$.
11: Build the schedule for the smart grid problem as in Lemma 3.

Theorem 2 *Algorithm* 1 *achieves an approximation ratio of* 2^ν.

Proof By Lemma 3, we know that given an assignment of jobs for the L_ν-norm problem on unrelated machines of cost C, we can construct a schedule for the smart grid problem with a cost of C^ν in polynomial time. Thus, we have $(OPT_L)^\nu = OPT_{SG}$ where OPT_L is the optimal cost of the L_ν-norm problem and OPT_{SG} is the optimal cost of the smart grid problem.

Besides, Azar–Epstein algorithm (Azar and Epstein 2005) is 2-approximation for the L_ν-norm problem. Therefore, we have $OPT_L \le C \le 2OPT_L$. Finally, by raising each term of the inequality by a power of ν, we have $(OPT_L)^\nu \le C^\nu \le 2^\nu (OPT_L)^\nu$, so $OPT_{SG} \le C^\nu \le 2^\nu OPT_{SG}$. The theorem follows. □

4 Concluding remarks

In the paper, we have considered a general model of demand response management in a smart grid. We have given a competitive algorithm that is optimal (up to a constant factor) in typical settings. Our algorithm is robust to arbitrary demands

and so enables the flexibility of the choices of clients in shaping their needs. The paper gives rise to several directions for future investigations. First, in the scheduling aspect, it would be interesting to consider problems in the general model with additional requirements such as precedence constraints. Secondly, in the game theory aspect, designing pricing schemes that allow clients to react reasonably while maintaining the efficiency in the energy consumption has received particular interests from both theoretical and practical studies in smart grid. Through the primal–dual viewpoint, dual variables can be interpreted as the payments of clients. An interesting direction is to design a pricing scheme based on primal–dual approaches.

Acknowledgements We thank Prudence W. H. Wong for insightful discussion and anonymous reviewers for useful comments that help to improve the presentation.

References

Albers, S. (2010). Energy-efficient algorithms. *Communications of the ACM, 53*(5), 86–96.

Alford, R., Dean, M., Hoontrakul, P., & Smith, P. (2012). *Power systems of the future: The case for energy sotrage, distributed generation, and microgrids*. Tech Rep: Zpryme Research & Consulting.

Azar, Y., & Epstein, A. (2005). Convex programming for scheduling unrelated parallel machines. In: Proceedings of 37th Annual ACM Symposium on Theory of Computing, pp 331–337

Bell, P. C., & Wong, P. W. H. (2015). Multiprocessor speed scaling for jobs with arbitrary sizes and deadlines. *Journal of Combinatorial Optimization, 29*(4), 739–749.

Burcea, M., Hon, W., Liu, H. H., Wong, P. W. H., & Yau, D. K. Y. (2016). Scheduling for electricity cost in a smart grid. *Journal of Scheduling, 19*(6), 687–699.

Chen, C., Nagananda, K., Xiong, G., Kishore, S., & Snyder, L. V. (2013). A communication-based appliance scheduling scheme for consumer-premise energy management systems. *IEEE Transactions on smart Grid, 4*(1), 56–65.

Cohen, J., Dürr, C., Thang, N. K. (2012). Smooth inequalities and equilibrium inefficiency in scheduling games. In: International Workshop on Internet and Network Economics, pp 350–363

Fang, K., Uhan, N. A., Zhao, F., & Sutherland, J. W. (2016). Scheduling on a single machine under time-of-use electricity tariffs. *Annals OR, 238*(1–2), 199–227.

Feng, X., Xu, Y., & Zheng, F. (2015). Online scheduling for electricity cost in smart grid. *COCOA, Springer, Lecture Notes in Computer Science, 9486,* 783–793.

Hamilton, K., & Gulhar, N. (2010). Taking demand response to the next level. *IEEE Power and Energy Magazine, 8*(3), 60–65.

Koutsopoulos, I., & Tassiulas, L. (2011). Control and optimization meet the smart power grid: Scheduling of power demands for optimal energy management. In: e-Energy, ACM, pp 41–50

Lenstra, J. K., Shmoys, D. B., & Tardos, É. (1990). Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming, 46*(1), 259–271.

Liu, F., Liu, H. H., & Wong, P. W. H. (2016). Optimal nonpreemptive scheduling in a smart grid model. CoRR abs/1602.06659. http://arxiv.org/abs/1602.06659

Liu, F., Liu, H., & Wong, P. W. H. (2020). Non-preemptive scheduling in a smart grid model and its implications on machine minimization. *Algorithmica, 82*(12), 3415–3457. https://doi.org/10.1007/s00453-020-00733-3.

Liu, F. H., Liu, H. H., & Wong, P. W. (2019). Greedy is optimal for online restricted assignment and smart grid scheduling for unit size jobs. In: International Workshop on Approximation and Online Algorithms. Springer, pp 217–231

Lui, T. J., Stirling, W., & Marcy, H. O. (2010). Get smart. *IEEE Power and Energy Magazine, 8*(3), 66–78.

Maharjan, S., Zhu, Q., Zhang, Y., Gjessing, S., & Basar, T. (2013). Dependable demand response management in the smart grid: A stackelberg game approach. *IEEE Trans Smart Grid, 4*(1), 120–132.

Roughgarden, T. (2015). Intrinsic robustness of the price of anarchy. *Journal of the ACM, 62*(5), 32.

Salinas, S., Li, M., & Li, P. (2013). Multi-objective optimal energy consumption scheduling in smart grids. *IEEE Transactions on Smart Grid, 4*(1), 341–348.

Thang, N. K. (2020). Online primal-dual algorithms with configuration linear programs. In: 31st International Symposium on Algorithms and Computation, ISAAC 2020, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, LIPIcs, vol 181, pp 45:1–45:16. https://doi.org/10.4230/LIPIcs.ISAAC.2020.45

US Department of Energy (2009) The smart grid: An introduction. https://energy.gov/oe/downloads/smart-grid-introduction-0

Yao, F. F., Demers, A. J., & Shenker, S. (1995). A scheduling model for reduced CPU energy. In: FOCS, IEEE Computer Society, pp 374–382