Smoothing the Energy Consumption: Peak Demand Reduction in Smart Grid

Shaojie Tang†, Qiuyuan Huang‡, Xiang-Yang Li$^{\dagger *}$, Dapeng Wu‡
†Department of Computer Science, Illinois Institute of Technology, USA
*School of Software, TNLIST, Tsinghua University
‡Department of Electrical & Computer Engineering, University of Florida

***Abstract*—Assume that a set of Demand Response Switch (DRS) devices are deployed in smart meters for autonomous demand side management within one house. The DRS devices are able to sense and control the activity of each appliance. We propose a set of appliance scheduling algorithms to 1) minimize the peak power consumption under a fixed delay requirement, and 2) minimize the delay under a fixed peak demand constraint. For both problems, we first prove that they are NP-Hard. Then we propose a set of approximation algorithms with constant approximation ratios. We conduct extensive simulations using both real-life appliance energy consumption data trace and synthetic data to evaluate the performance of our algorithms. Extensive evaluations verify that the schedules obtained by our methods significantly reduce the peak demand or delay compared with naive greedy algorithm or randomized algorithm.**

I. Introduction

It was reported by U.S. Department of Energy in 2008 [25] that energy consumptions from buildings account for approximately 74% of the nation's total electricity consumption. Unfortunately, due to inefficient energy consumption pattern in most buildings, billions of dollars were wasted. To improve energy usage efficiency, two different approaches could be used: *reducing consumption* and *shifting consumption* [18]. Notice that reducing consumption could be achieved through either more careful consumption behaviors or constructing more energy-efficient appliances. For example, we could encourage people to use less energy consuming appliances. In this work, we are particularly interested at the second approach. As a complimentary approach to the first one, we stress the importance of reducing peak demand through shifting heavy-load appliances to off-peak hours, which makes it easy for power generation to match the demand.

Energy consumption or load scheduling have been well studied since thirty years ago [5], [9] [28]. However, recent advancements in smart metering and smart grid infrastructure allow us to adopt fine-grained energy consumption scheduling. In this work, we aim at optimally scheduling the household or industry energy consumption in each appliance in order to reduce the peak energy consumption or demand. We assume that several appliances within one building access to one energy resource. Each appliance is equipped with a Demand Response Switch (DRS) device. The DRS devices are connected to the power line and are able to communicate with each other. The DRS devices will follow the optimal energy consumption schedule to coordinate each appliance.

In this work, we study two energy consumption schedule problems. The first problem is *Peak Demand Minimization* problem. For this problem, we assume that all appliances or jobs must be scheduled within a given finite time duration. Our goal is to find an appliance scheduling strategy that will minimize the *peak demand* while not violating the delay constraint. Here the peak demand is defined as the largest power demand at any time instant. The other problem studied in this work is called *Delay Minimization* problem, which is a dual problem to the first one. Essentially, we assume that there is a predefined maximum peak demand constraint, and the objective is to design an appliance scheduling strategy that will minimize the *delay* while not violating the peak demand constraint. We first prove that both problems are NP Hard. We further propose several approximation algorithms (MP algorithm for minimizing the peak demand, and MD1 and MD2 algorithms for minimizing the delay) which can achieve constant approximation ratios on the performances. We then briefly discuss the online setting of energy consumption scheduling problem. Under online setting, a simple yet efficient greedy algorithm is proposed. This algorithm has a constant competitive ratio for online delay minimization problem. We conduct extensive evaluations to study the performances of our methods using both real-life appliance energy consumption data trace and synthetic data. Extensive evaluations verify that the schedules obtained by our methods significantly reduce the peak demand or delay compared with naive greedy algorithm or randomized algorithm. For example, MP algorithm achieves an average of more than 30% savings on peak power consumption compared to the Greedy algorithm, both in small-scale and large-scale evaluations. MD1 and MD2 algorithms outperform the Greedy algorithm by more than 50% and 40% savings on execution time respectively.

The paper is organized as follows. Section II and Section III introduce the system model, motivation and problem formulation. Section IV and Section V describe our scheduling schemes for both peak demand minimization problem and delay minimization problem. We investigate online setting of both problems in Section VI. Extensive evaluation results based on real data trace are reported in Section VII. We review related work in Section VIII and conclude the paper with some future work directions in Section IX.

II. System Model

We consider a discrete time system and a time period $[0, T]$ during which n electrical jobs or appliance, $\mathbf{J} = \{J_1, J_2, \cdots, J_n\}$, should be scheduled. Here T typically is 24 hours. We adopt similar notations used in [6] in the rest of this paper: the i-th job J_i has a demand profile D_i parameterized by (d_i, τ_i), where d_i denotes J_i's instantaneous power consumption and τ_n represents its duration. When the job J_i is scheduled to start at time s_i, it specifies a power consumption function:

$$D_i(t) = d_i \cdot \mathbf{I}_{[s_i, s_i+\tau_i]}(t),$$

where $\mathbf{I}_{[a,b]}(t)$ is a step function that has value 1 at the interval $[a, b]$ and 0 elsewhere. We assume that the job cannot be interrupted once it starts. In general, the starting time s_i should satisfy $t_i^s \leq s_i \leq t_i^d$, where t_i^s and t_i^d represent the earliest and latest starting time of the job. In this work, we aim at assigning each job J_i a starting time $0 \leq s_i \leq T - \tau_i$ under various constraints to reduce the peak demand. Consequently, under a given schedule, the total load on the network at time instance t is

$$D(t) \triangleq \sum_{i=1}^{n} D_i(t) = \sum_{i=1}^{n} d_i \cdot \mathbf{I}_{[s_i, s_i+\tau_i]}(t).$$

We next reveal the motivations of this study by introducing two widely adopted utility cost model. For each of those models, in order to reduce the utility cost, it is essential to find an energy consumption scheduling with small peak demand.

Utility Cost Model I: Let C represent the utility cost from a group of appliances. In a demand based tariff for commercial energy customers [21], the utility cost is calculated by

$$C = p_u \cdot D_{Tot} + p_d \cdot D_{Peak}$$

where $D_{Peak} = \max_{t \in [0,T]} D(t)$ is the *peak aggregate demand* and $D_{Tot} = \sum_{t \in [0,T]} D(t)$ represents the total energy consumption, p_u the *usage price*, and p_d the *demand price*. Usually, p_d is significantly higher than p_u, *e.g.*, in Pennsylvania, p_d is 240 times p_u [21]. The penalty for high peak power consumption is particularly significant, which motivates research on reduction of peak power consumption. Clearly, given a fixed amount of total demands, the utility bill is minimized when the peak power consumption is minimized under this utility model.

Utility Cost Model II: The second utility cost model is defined based on smooth differentiable quadratic function [18]. For each hour of the time horizon $h \in \{1, \ldots, 24\}$, the cost of the hour C_h depends on the total load of the hour L_h and the hourly rate r. The quadratic cost function is defined as $C_h(L_h) = r \times (L_h)^2$. This cost function is increasing and strictly convex. Under this strictly convex function, it is reasonable to minimize the peak demand in order to reduce the overall cost.

III. Problem Formulation

In this paper, we first study the **Peak Demand Minimization Problem** for peak demand reduction. We assume that time horizon is finite with duration $\mathbf{T}$ since users do not want to delay their jobs forever.

Problem 1 (Peak Demand Minimization Problem)**.** *Compute starting time s_i for each job J_i to minimize the peak demand D_{Peak}. Then the optimization problem for minimizing peak demand during a finite horizon $\mathbf{T} > 0$, is formulated as:*

Problem: Peak Demand Minimization Scheduling
Objective: Minimize D_{Peak}
subject to:

$$\begin{cases} (1)\ D_i(t) = d_i \mathbf{I}_{[s_i, s_i+\tau_i]}(t) \\ (2)\ D(t) \triangleq \sum_i D_i(t) = \sum_{i=1}^{n} d_i \mathbf{I}_{[s_i, s_i+\tau_i]}(t) \\ (3)\ 0 \leq s_i \leq \mathbf{T} - \tau_i \\ (3)\ D_{Peak} = \max_{t \in [0,T]} D(t) \end{cases}$$

We first show that even when all durations τ_n are equal, finding an optimal schedule is still an NP-hard problem (*i.e.*, can be reduced to the *Subset-Sum* problem).

Lemma 1. *The Peak Demand Minimization Problem is NP-Hard.*

Proof: We will reduce our problem from the *Subset-Sum* Problem: Given a set of integers $A = \{I_1, \cdots, I_n\}$, determine whether there exists a subset A^c of numbers from A such that the sum of those numbers

$$\sum_{I_i \in A^c} I_i = \sum_{I_i \in A} I_i/2$$

Then given an input as listed above, we construct a peak demand minimization problem as follows: In the constructed case, there are n jobs $J_1, \cdots, J_n$ each of which has identical duration $\mathbf{T}/2$. We assume the demand of each job $d_i = I_i$. We want to decide if there is a schedule with peak demand $\sum_{I_i \in A} I_i/2$, which clearly is the minimum.

Clearly if there exists a subset of numbers from A such that the sum of those numbers $\sum_{I_i \in A^c} I_i = \sum_{I_i \in A} I_i/2$, we are able to find a scheduling with minimum height $\sum_{I_i \in A} I_i/2$ by scheduling all jobs in A^c at time 0, and scheduling all remained jobs at time $\mathbf{T}/2$. On the other hand, if there exists a scheduling with height $\sum_{I_i \in A} I_i/2$, we can pick those jobs that start from time 0 and the union of their demands must be a feasible A^c. This finishes the proof. ■

In this work, we also study the **Delay Minimization Problem** under a given peak demand constraint. Delay Minimization problem is dual to Peak Demand Minimization problem. In this problem, we aim at finding an energy consumption schedule so as to minimize the finish time for executing all the jobs, while keeping the peak power consumption under a constraint $\mathbf{D}$. The finish time for executing all the jobs is given by

$$T = \max_{i=\{1,\cdots,n\}} (s_i + \tau_i)$$

where s_i is produced by the scheduling algorithm. $\max_i(s_i + \tau_i)$ is the time that the last job is finished. The intuition here is as aforementioned: in order to cut down cost on energy consumption, users have the incentive to restrict the maximum peak consumption to a pre-defined value $\mathbf{D}$, since penalty for peaks in power consumption is significantly high.

Problem 2 (Delay Minimization Problem)**.** *Compute a starting time s_i for each job J_i, so as to minimize the delay or finish time $\max_i s_i + \tau_i$. Then the optimization problem for minimizing finish time given user-defined maximum peak power consumption $\boldsymbol{D} > 0$, is formulated as:*

Problem: Delay Minimization Scheduling
Objective: Minimize T
subject to:

$$\begin{cases} (1)\ \ D_i(t) = d_i \mathbf{I}_{[s_i, d_i]}(t) \\ (2)\ \ D(t) \triangleq \sum_{i=1}^{n} D_i(t) = \sum_i d_i \cdot \mathbf{I}_{[s_i, s_i + d_i]}(t) \\ (3)\ \ D_{Peak} = \max_{t \in [0,T]} D(t) \\ (4)\ \ D_{Peak} \leq \boldsymbol{D} \\ (5)\ \ T = \max_{i \in [1,n]}(s_i + \tau_i) \end{cases}$$

Similar to the previous problem, finding an optimal delay minimization schedule is still an NP-hard problem even when all durations τ_n are equal.

Lemma 2. *The Delay Minimization Problem is NP-Hard.*

Proof: We will reduce the Delay Minimization Problem from classical *Bin-Packing* Problem: Given a bin size V and a list of sizes of the items $\{a_1, a_2, \cdots, a_n\}$ to pack, find an integer B and a B-partition $\{S_1, S_2, S_3, \cdots, S_B\}$ of $\{1, \cdots, n\}$ such that $\sum_{i \in S_k} a_i \leq V$ for all $k = 1, \cdots, B$. Here $\bigcup_i S_i = \{1, 2 \cdots, n\}$, and $S_i \cap S_j = \phi$. A solution is optimal if it has minimal B.

Then given an input as listed above, we construct a delay minimization problem as follows. In the constructed case, there are n jobs $J_1, \cdots, J_n$ each of which has identical duration 1. We assume the demand of each job $d_i = a_i$ and the peak demand constraint is $\mathbf{D} = V$.

Thus if there exists a B-partition of numbers $\{a_1, a_2, \cdots, a_n\}$, we are able to find a scheduling with maximum height no larger than $\mathbf{D}$ by scheduling all jobs whose demand belong to the same partition at the same starting time. On the other hand, if there exists a scheduling with height at most $\mathbf{D}$, we can put the demands of those jobs that share the same starting time in the same partition. This finishes the proof. ■

IV. Peak Demand Minimization

Because the peak demand minimization problem is NP-hard, we next seek approximation algorithms to tackle this problem.

A. Greedy Algorithm with Constant Approximation

In this section, we propose a simple yet efficient method (Algorithm MP) which is proved to have a constant approximation ratio. In the rest of this paper, for ease of presentation, we will use "demand" and "height", "duration" and "width" interchangeably for each job. It is not difficult to observe that the appliance scheduling problem (especially the peak demand minimization problem) bears a lot of similarity to the rectangle packing problem [2], [13], [19], [20], [26] that has been extensively studied in the literature, by treating each job as a rectangle (where the demand is the height of the rectangle and the duration is the width of the rectangle). However, the methods for packing rectangles cannot be directly applied to solve the peak demand minimization problem here due to the following differences: the rectangles are rigid while the energy consumption is simply addition.

The basic idea of our method is as follows. According to each job's duration, we partition all jobs into two subsets: wide jobs $\mathbf{J}_1$ and narrow jobs $\mathbf{J}_2$. Here $\mathbf{J}_1$ essentially contains all "*wide*" jobs *e.g.*, $\tau_i \geq \frac{\mathbf{T}}{a}$ for each job $J_i \in \mathbf{J}_1$. And $\mathbf{J}_2$ contains all "*narrow*" jobs: for each job $J_j \in \mathbf{J}_2$, we have $\tau_j < \frac{\mathbf{T}}{a}$. Here a is some constant which can be optimized later. Intuitively, for these wide jobs, the summation of their total electrical demands is at the same order of the minimum peak value for scheduling these wide jobs alone. Thus, we already have a good approximation ratio for these wide jobs.

For the narrow jobs, again we partition them into different groups again based on the demand: the jobs at each group here will have similar demand. Thus packing the jobs in each of the group using greedy approach should result in a constant approximation for these jobs. Then the challenging part is to show that the combined solution of all possible groups are indeed a good approximation to the global optimum.

The procedures of our method are presented in Algorithm 1.

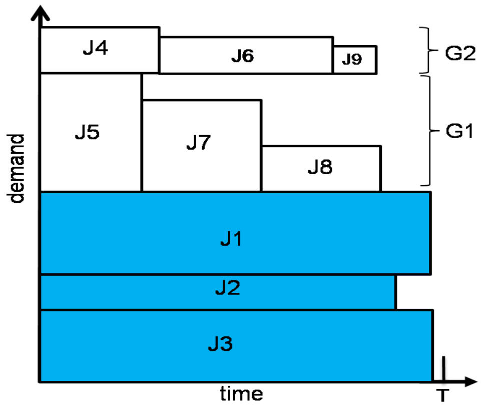


Fig. 1. Schedule by MP.

Example 1 Figure. 1 illustrates one schedule returned by Algorithm MP. In this example, jobs $J_1 \sim J_3$ belong to wide jobs and $J_4 \sim J_{10}$ belong to narrow jobs. Further, J_5, J_7, J_8 belong to $\mathbf{G}_1$, and J_4, J_6 and J_9 belong to $\mathbf{G}_2$. According to Algorithm MP, jobs J_1, J_2 and J_3 are scheduled at time 0. All jobs in $\mathbf{G}_1$ are scheduled at the first strip and all jobs in $\mathbf{G}_2$ are scheduled within the second strip.

Lemma 3. *The height of all wide jobs $\mathbf{J}_1$ returned by Algorithm MP is at most a times the minimum height.*

Algorithm 1 MP: Greedy Scheduling for Reducing the Peak

```
Input: The job profile for each job.
Output: The job scheduling.
1: Partition jobs into wide jobs $\mathbf{J}_1$ and narrow jobs $\mathbf{J}_2$. For those wide jobs in $\mathbf{J}_1$, we simply schedule them using a simple greedy: sort the jobs based on their demand, and schedule the $k$th job after $(k-1)$-th job if possible, or schedule the $k$-th job at time 0 otherwise.
2: For narrow jobs in $\mathbf{J}_2$, we sort them in non-increasing order of demands. Assume there are $m$ narrow jobs and $\mathbf{J}_2 = \{J_1, J_2, \cdots, J_m\}$ is sorted list where $d_1 \geq d_2 \geq \cdots \geq d_m$.
    1) We further partition $\mathbf{J}_2$ to several groups, $\mathbf{G}_1, \mathbf{G}_2, ...$, such that the demand of each job within one group differs by at most 2, e.g., $\frac{d_j}{2} \leq d_i \leq 2d_j$ for any two job $J_i$ and $J_j$ within the same group.
    2) We schedule each group one by one as follows: Starting from the first job $J_1$ in the first group $\mathbf{G}_1$, $J_1$ is scheduled to start at time 0; $J_2$ is scheduled just after job $J_1$ (i.e., at time $\tau_1$); $J_3$ is scheduled just after job $J_2$ until any one of the following conditions is satisfied:
        a) All jobs in $\mathbf{G}_1$ are scheduled at the first strip, e.g., $\sum_{J_i \in \mathbf{G}_1} \tau_i \leq T$, where the first strip is defined as the rectangle with height $d_1$ and width $\mathbf{T}$;
        b) Some jobs in $\mathbf{G}_1$ can not be scheduled at the first strip, e.g., $\sum_{J_i \in \mathbf{G}_1} \tau_i > T$;
    If case a) happens, this strip is called "almost-empty", we will then schedule the next group of narrow jobs $\mathbf{G}_2$ using a new strip with starting time 0. If case b) happens, we call this strip "almost-full-filled", we continue to schedule the remaining jobs in $\mathbf{G}_1$ using a new strip with starting time 0.
```

Proof: The duration of each job in $\mathbf{J}_1$ is at least $\frac{\mathbf{T}}{a}$. We next prove this lemma through simple area argument. Assume that the height of all wide jobs $\mathbf{J}_1$ returned by Algorithm MP is D, thus the total area of those jobs in $\mathbf{J}_1$ is at least $\frac{\mathbf{T}}{a} \cdot D$. Clearly, no scheduling method can achieve height smaller than $\frac{\mathbf{T}}{a} D / \mathbf{T} = \frac{D}{a}$, thus our scheduling achieves the height at most a times the optimal solution. ■

Lemma 4. *For all the jobs contained in an* almost-full-filled *strip, the height of those jobs resulted from Algorithm MP is at most $\frac{2a}{a-1}$ times the minimum height.*

Proof: We prove this lemma using area argument. Given a full-filled strip, assume the height of the first job in that strip is d. We first notice that the total area of those jobs that are contained in one 'almost-full-filled' strip is at least $(\mathbf{T} - \frac{\mathbf{T}}{a})\frac{d}{2}$. Then the height that is contributed from those jobs in the optimal scheduling is at least $(\mathbf{T} - \frac{\mathbf{T}}{a})\frac{d}{2}/\mathbf{T} = (1 - \frac{1}{a})\frac{d}{2}$. Notice that the height resulted from our scheduling is d. Therefore the height based on our scheduling is at most $\frac{2a}{a-1}$ times the minimum height. ■

Lemma 5. *The overall height of all 'almost-empty' strips is at most $2d_1$.*

Proof: According to the procedures of Algorithm MP, the height of the first almost empty strip is at most d_1 where d_1 represents the demand of the first job in $\mathbf{G}_1$. Recall that the demand of each job within one group differs by at most 2, therefore the demand of each job in $\mathbf{G}_2$ is less than $\frac{d_1}{2}$. It follows that the second almost-empty strip has height at most $\frac{d_1}{2}$. Through simple induction, we have the overall height of all almost empty strips is at most $d_1 + \frac{d_1}{2} + \frac{d_1}{4} + \cdots \leq 2d_1$. This finishes the proof. ■

Theorem 1. *Algorithm MP achieves an approximation ratio at most 7 by setting $a = 2$.*

Proof: First of all, based on Lemma 3, we have the overall height of scheduled jobs in $\mathbf{J}_1$ is at most a times the minimum height. Lemma 4 and Lemma 5 together imply that the overall height of scheduled jobs in $\mathbf{J}_2$ is at most $2 + \frac{2a}{a-1}$ times the optimal solution. Therefore the overall height resulted from our scheduling is less than $a + 2 + \frac{2a}{a-1}$ times the optimal solution. By setting $a = 2$, Algorithm MP achieves 7 approximation. ■

V. Delay Minimization

In this section, we study the delay minimization scheduling problem. We propose two efficient greedy algorithms based on similar approaches that are developed in [34]. The differences between our approach and the one in [34] are as follows. First, in [34], they study online scheduling of parallel jobs, where each job requires one or multiple machines simultaneously. As a problem input in [34], the job arrival list is unknown. However, as in our scenario, we can adjust the arriving time of each job which is actually the output of our problem. Second, we develop a more elegant proof to derive the constant approximation ratio for the second algorithm.

Both two algorithms follow simple greedy rule: Schedule each job one by one according to certain ordering, and every job is scheduled at its earliest starting time without violating peak demand constraint. In the first algorithm, we sort all jobs in non-increasing order of their durations. In the second algorithm, all jobs are ordered in non-increasing order of their demands. For each of those algorithms, we are able to derive constant approximation bounds.

A. Non-increasing Duration based Ordering

We sort all the jobs in $\mathbf{J}$ in non-increasing order of their durations. Suppose that the ordered list is $\mathbf{J} = \{J_1, J_2, \cdots, J_n\}$ where $\tau_i \geq \tau_{i+1}$ for all i.

A job J_i is called *tall* if its demand d_i is larger than $\mathbf{D}/2$, otherwise it is called *short*. Recall that $\mathbf{D}$ denotes the maximum peak power consumption that is predefined by customers. The OCCUPANCY RATIO of a schedule at any time t is defined as the amount of occupied power resource

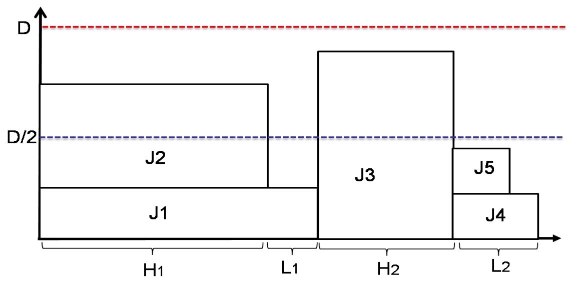


Fig. 2. Schedule by method MD1.

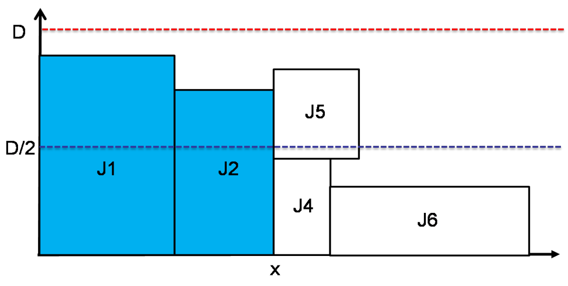


Fig. 3. Schedule by method MD2.

Algorithm 2 Non-increasing Duration based Ordering (MD1)

Input: The job profile $J_i = (d_i, \tau_i)$ for each job.
Output: The energy consumption scheduling.

```
1: Sort all jobs in the non-increasing order of durations $J_i \in \mathbf{J}$ $(1 \leq i \leq n)$;
2: for $i$ from 1 to $n$ do
3:     Schedule $J_i$ at the earliest starting time without violating the peak demand constraint;
```

Algorithm 3 Non-increasing Demand based Ordering (MD2)

Input: The demand profile $D_i = (d_i, \tau_i)$ for each job.
Output: The energy consumption scheduling.

```
1: Sort all jobs in the non-increasing order of demands $J_i \in \mathbf{J}$ $(1 \leq i \leq n)$;
2: for $i$ from 1 to $n$ do
3:     Schedule job $J_i$ at the earliest starting time without violating the peak demand constraint;
```

at time t divided by $\mathbf{D}$, which represents the efficiency of resource utilization. A simple yet efficient greedy algorithm based on the ordering of non-increasing duration is described in Algorithm MD1. In the rest of this paper, let T^* represent the minimum delay resulted from optimal scheduling under peak demand constraint $\mathbf{D}$, and let T_1 denote the resulted delay from Algorithm MD1.

Example 2 Figure. 2 illustrates a schedule returned from Algorithm MD1. In this example, the occupancy ratio during interval H_1 (and H_2) is larger than $1/2$, and the occupancy ration during interval L_1 (and L_2) is smaller than $1/2$.

Theorem 2. *The approximation ratio of Algorithm MD1 is at most* 2.

The proof follows similar proof in [34] [1], thus omitted here to avoid redundant presentation.

B. Non-increasing Demand based Ordering

Under this method, we sort all the jobs in $\mathbf{J}$ in non-increasing order of their demands. Suppose that the ordered list is $\mathbf{J} = \{J_1, J_2, \cdots, J_n\}$ where $d_i \geq d_{i+1}$ for all i.

Example 3 Figure. 3 illustrates a schedule generated by Algorithm MD2. In this example, both tall jobs J_1 and J_2 are scheduled consecutively. The rest of the short jobs are scheduled at its earliest starting time.

Same as introduced in previous discussion, a job J_i is called *tall* if its demand d_i is larger than $\mathbf{D}/2$, otherwise it is called *short*. We next describe our second greedy algorithm according to non-increasing demand based ordering in Algorithm MD2.

[1]For the online setting, the jobs arrive in non-increasing order of their processing times.

Theorem 3. *The approximation ratio of Algorithm MD2 is at most* 4.

Proof: As in Algorithm MD2, all tall jobs are scheduled consecutively from time 0. Please refer to Example 3 as an illustration where both jobs J_1 and J_2 are scheduled consecutively. Let x denote the length of the schedule during which all tall jobs are processed. We have $T^* \geq x$ because no two tall jobs can be scheduled at the same time. Denote by T_2 the finish time of the last job J_n whose process time determines the makespan. Avoiding trivial cases, we assume J_n is a short job. Otherwise, if J_n is a tall job, we immediately obtain $T_2 \leq T^*$. Notice that $T^* \geq \tau_n$. Let s_n denote the starting time of job J_n. Again, to avoid trivial cases, we assume $s_n > x$, because, otherwise, we immediately obtain $T_2 \leq 2T^*$ together with fact that $T^* \geq \tau_n$ and $T^* \geq x$. Therefore in the following proof, we have $s_n > x$. Notice that all jobs scheduled during interval $(x, T_2]$ must be short jobs.

We next prove that the occupancy ratio at any time in $(x, T_2]$ is no less than $1/2$. The proof is conducted through simple contradiction argument. Assume that at some point t in $(x, T_2]$, the occupancy ratio is less than $1/2$. Because all jobs scheduled after t must be short jobs whose demand is no larger than $\mathbf{D}/2$, we can always move a job scheduled right after t forward a bit to let it start at time t. This contradicts to Algorithm MD2 in which all jobs are scheduled at its earliest starting time. By combining all previous discussions, we have $T_2 \leq T^* + 2T^* + T^* = 4T^*$. This completes the proof. ■

VI. Online Scheduling of Appliances

In the previous problem setting, we assume all jobs and their profiles are received at the initial stage. However, this setting may not be true in practical. Therefore, we next briefly discuss the online version of scheduling problem in smart grid. We assume that each job arrives one by one sequentially. Each job is known at its release date, no information on this job known before its arrival time. In this setting jobs arrive one by one, upon each job's arrival, the algorithm has to decide immediately when to assign each job.

We propose a set of online greedy algorithms to tackle both peak demand minimization problem and delay minimization problem. Online greedy algorithms work as follows, (1) For the demand minimization problem, we always schedule currently released job at some time which can minimize increased peak demand; (2) For the delay minimization problem, we schedule current job at its earliest starting time without violating peak demand constraint.

To evaluate the performance of online greedy algorithms, we provide a lower bound on the competitive ratio of any non-preemptive online algorithm for delay minimization problem. Here competitive ratio of a online algorithm is defined as the ratio between the solution returned from that online algorithm and the one returned by best offline algorithm.

Lemma 6. *[8] For general online delay minimization problem, no deterministic non-preemptive online algorithm can achieve competitive ratio better than 1.347.*

In fact, the competitive ratio of online greedy algorithm is bounded by 2 based on [34] [11].

Theorem 4. *For delay minimization problem, online greedy algorithm has competitive ratio* 2.

VII. Performance Evaluation

In this section, we present the simulation results and assess the performance of our proposed algorithms. We implemented the MP, MD1 and MD2 scheduling algorithms and compared the proposed approaches to Greedy algorithm in different problem settings and scale settings.

In our experiments, we employ the energy cost model used in [18] for evaluation. In this model, a local power distribution network is considered with one energy source and several load subscribers. To evaluate the performance of proposed algorithms in terms of peak power consumption and execution delay, we randomly select jobs from the residential appliance catalog as reported in [18] for scheduling. The demand profile (power demand and daily usage) of each appliance is determined according to the residential power consumption data released by Office of Energy Efficiency of Canada in 2005 [1]. Sample residential appliances include dishwasher (daily usage: 1.44 kWh), clothes washer (daily usage: 1.94 kWh), clothes dryer (daily usage: 2.5 kWh), and PHEV (daily usage: 9.9 kWh), etc. We report peak demand, execution delay as well as the effect of different constraints. Our experimental result on utility charge verifies that the proposed algorithms are able to improve power efficiency as well as reduce energy cost by achieving more *even load* in the power system.

A. Performance on Peak Power Consumption

In this set of experiments, we investigate the performance on peak power consumption during a finite time horizon. We implemented Greedy scheduling algorithm and our proposed MP algorithm. The finite time horizon T is set to be 100 hours. For small-scale evaluation, we randomly picked 50 jobs from the residential appliance catalog and schedule them using the algorithms under consideration. For large-scale evaluation, we randomly picked 500 jobs for scheduling. The peak power consumption values with respect to schedules produced by the two algorithms are given in Figure 4 respectively.

As shown in the figure, MP scheduling algorithm achieves an average of more than 30% savings on peak power consumption compared to the Greedy algorithm, both in small-scale and large-scale evaluations. This implies a significant cut of expenses on energy consumption as peak power prices are 200–400 times that of the nominal rate. More importantly, it shows that MP algorithm can achieve significant peak demand reduction while being efficient and scalable by coordinating the power-consuming jobs more meticulously.

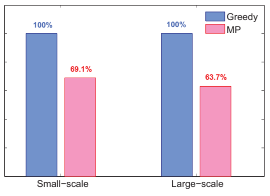


Fig. 4. Peak power consumption percentage using Greedy algorithm as baseline.

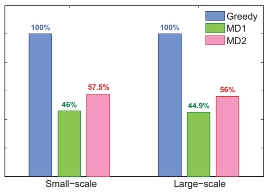


Fig. 5. Execution delay percentage using Greedy algorithm as baseline.

B. Performance on Execution Delay

In this set of experiments, we study the performance of scheduling algorithms in terms of execution delay under a pre-defined peak constraint. We implemented MD1, MD2 algorithms with comparison to the Greedy algorithm. We used the same sets of small-scale jobs and large-scale jobs aforementioned for performance evaluation. The peak constraint $\mathbf{D}$ is set to be 5 times the highest-demand job in the evaluation set. The execution delays with respect to schedules produced by the three algorithms are shown in Figure 5.

As illustrated in the figure, MD1 and MD2 algorithms outperform the Greedy algorithm by more than 50% and 40% savings on execution time respectively while both manage to keep peak power consumption in the desired range. The reason is that while Greedy algorithm randomly pick a job for scheduling each time, MD1 and MD2 algorithms explore more efficient combinations of jobs according to their profiles and assign jobs as "tight" as possible for parallel execution. Thus the "power space" is more fulfilled with respect to the peak constraint and the execution time is minimized.

C. Effect of Peak Constraints

In this set of experiments, we test the performance of MD1 and MD2 algorithms under various pre-defined peak constraints. We use 10 large-scale sets of jobs randomly selected from the evaluation set to get a comprehensive comparison among MD1, MD2 and the Greedy algorithm. The execution delays of the schedules produced by the three algorithms are shown in Figure 6. The x axis represents the times relationship of peak constraint $\mathbf{D}$ to the power demand d_i of the biggest job J_i in the evaluation set, *i.e.* $i = \text{argmax}_j\, d_j$. We use the execution delay produced by Greedy algorithm when $\mathbf{D} = d_i$ as baseline for comparison. The reported results are averaged over 10 large-scaled sets of jobs aforementioned.

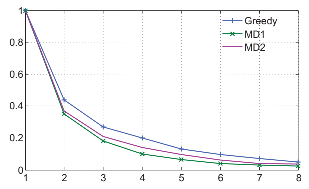


Fig. 6. Execution delay percentage using Greedy algorithm as baseline.

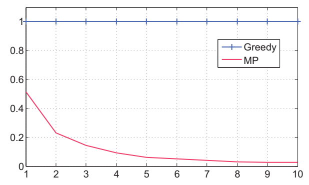


Fig. 7. Peak demand percentage using Greedy algorithm as baseline.

As shown in the figure, again, MD1 and MD2 outperforms Greedy algorithm in terms of execution delay under various peak constraints.

D. *Effect of Time Constraint*

In this set of experiments, we study the performance of MP algorithm under various pre-defined time constraints. We use 10 large-scale sets of jobs (the same as used in previous subsection) to compare the performance of MP algorithm with Greedy algorithm in terms of peak energy demand. The experimental results on peak under various time constraints are shown in Figure 7.

As shown in the figure, MP algorithm outperforms Greedy algorithm in terms of peak demand under various time constraints. The reason is that Greedy algorithm schedules all the jobs to execute as early as possible without respect to the varying time constraints. As a result, all the jobs are scheduled by Greedy algorithm to execute simultaneously from the very beginning (time slot 0) which leads to significant peak demands throughout the cases. In contrast, schedules produced by MP algorithm leads to continuous decreasing in peak demand as time constraint increases. This is consistent with the nature of MP algorithm for deriving *even* loads.

E. *Performance on Utility Charge*

In this set of experiments, we present the performance of the proposed scheduling algorithms in terms of energy consumption distribution and utility charge in total and for each individual power subscriber respectively within a 24 hour period. For the simplicity of illustration, we adopt the same energy cost model as previous where a local power distribution network is considered with one energy source and several load subscribers. We considered two scenarios in our experiment with different constraint settings for performance evaluation.

1) Time Constraint Setting: The first scenario we studied is the case where the time horizon T for scheduling all the jobs is restricted. Recall that MP algorithm is designed to determine the energy consumption scheduling to reduce the peak demand D_{peak}. We consider 10 load subscribers in the local area network. Our results can be simply extended to more load subscribers as our optimization model used in MP algorithm considers the job assignments only based on job profiles (energy demand and duration) no matter *who* released the job. For each subscriber we randomly select 15 appliances from the residential appliance catalog. In our experiment, we set the time constraint T to be 24 hours.

In this experiment, we adopt Utility Model II which is introduced in Section II based a smooth differentiable quadratic function as the pricing model.

The experimental results on scheduled power consumptions and hourly cost in the system using Random algorithm, Greedy algorithm, and MP algorithm are shown in Figure 8, Figure 9, and Figure 10 respectively.

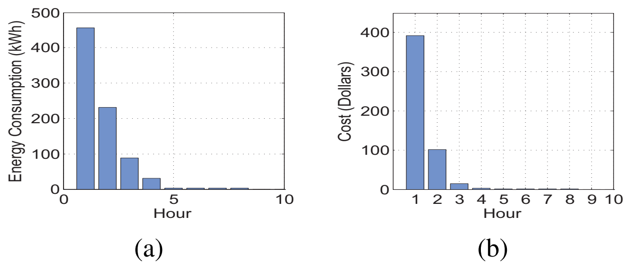


Fig. 8. Scheduled energy consumption (a) and corresponding cost (b) when Greedy algorithm is used. In this case, total cost is \$509.09.

As illustrated in Figure 8, when Greedy algorithm is used, the peak demand reaches as high as 457 kWh, the peak-to-average-ratio (PAR) is 11.4 and the total energy cost is \$509.09.

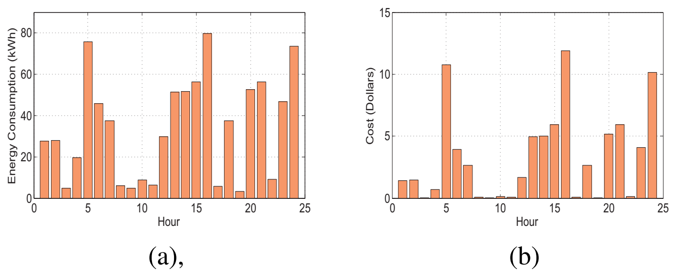
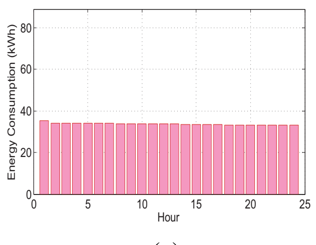


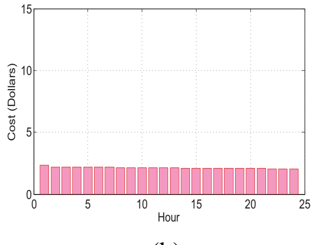
Fig. 9. Scheduled energy consumption (a) and corresponding cost (b) when Random algorithm is used. In this case, total cost is \$86.24.

As shown in Figure 9, when Random algorithm is used, the peak demand reduces to 79.72 kWh, the PAR reduces to 1.99 and the total energy cost reduces to \$86.24.

As shown in Figure 10, when MP algorithm is used, the peak demand reduces to as low as 35.2 kWh (*i.e.*, 55.9% less than Greedy algorithm), the PAR reduces to as low as 1.05 (*i.e.*, 47.2% less than Greedy algorithm) and the total energy cost reduces to as low as \$51.10 (*i.e.*, 40.7% less than Greedy algorithm). In face, we have more *even* load in this case. Note that each subscriber consumes the same amount of energy in the three cases, but it simply schedules its consumption more efficiently in the case that MP algorithm is used. In this case, all subscribers will even pay less to the utility company as shown in Figure 11. Therefore, the subscribers would be



(a)



(b)

Fig. 10. Scheduled energy consumption (a) and corresponding cost (b) when Greedy algorithm is used. In this case, total cost is \$51.10.

willing to participate in the proposed automatic demand side management system.

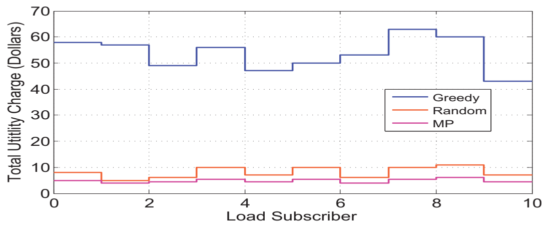


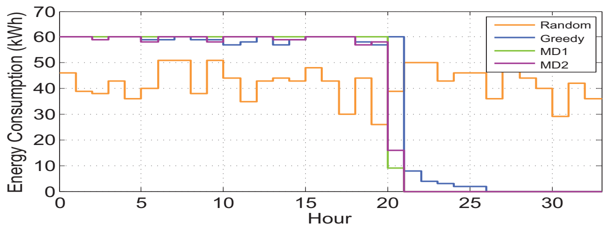
Fig. 11. Total charges for each subscriber using Greedy algorithm, Random algorithm and MP algorithms.

2) Peak Constraint Setting: The second scenario we studied is the case where the peak D_{peak} is constrained and pre-defined by the load subscribers. Recall that MD1 and MD2 algorithms are designed to determine the energy consumption scheduling to minimize the execution time T given the peak constraint. We consider the same power network settings used in the first scenario. 10 load subscribers are considered in the local power distribution network, with 15 appliances randomly selected for each subscriber in the network. The demand profile of each appliance is pre-defined and the peak demand constraint is set to be $D_{peak} = 60$ kW/h. The quadratic cost function $C_h(L_h) = r \times (L_h)^2$ is used to calculate the total energy cost as well as the utility charge for individual load subscribers. The results are shown in Figure 12.

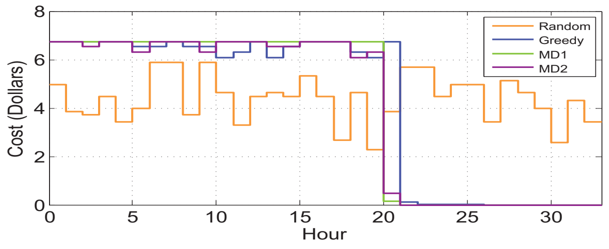
As shown in Figure 12, Random algorithm produces a random workload during the execution as it assigns appliances to random time slots while keeps peak demand under the constrained value. We observe from Figure 12(a) that MD1 and MD2 algorithm achieves slightly better performance than Greedy algorithm in terms of finish time. The reason is that all three algorithms assign appliances as early as possible as long as the peak constraint is satisfied. We can calculate the total cost from Figure 12(b) produced by each scheduling algorithm is \$140.76 (Random), \$137.96 (Greedy), \$135.15 (MD1) and \$132.83 (MD2) respectively.

VIII. Related Works

Peak demand reduction by model predictive control (MPC) with real-time electricity pricing was studied in [6] [21] [23]. One recent set of papers (*e.g.*, [3], [4]) focus on designing online algorithms for using UPS units for cost reduction via



(a)



(b)

Fig. 12. Scheduled energy consumption (a) and corresponding cost (b) when Greedy, Random, MD1 and MD2 algorithms are used respectively.

shaving workload "peaks" that correspond to higher energy prices. It provides a worst-case competitive ratio analysis. The other body of works study workload shifting for power cost reduction [37] or other reasons such as performance and availability [7]. Some research has considered consumers with access to multiple utility providers, each with a different carbon profile, power price and availability and looked at optimizing cost subject to performance and/or carbon emissions constraints [15].

Some problems closely related to job scheduling in smart grid include link scheduling in wireless networks [22], [32], [36], parallel job scheduling [34], [11], [27], [10], [12] rectangle packing [13], [19], [20], [26] and Orthogonal Rectangular Strip Packing Problem (SPP) [2], [14], [17], [24], sometimes also called orthogonal packing in two dimensions. Essentially, the goal is to find the best orthogonal placement of a set of rectangles on a strip of rectangular stock sheet with a fixed width and infinite height. In contrast to the problem considered here, power resource assigned to a job need to be "contiguous" in a solution to the strip packing problem. To the best of our knowledge, the performance ratio of the best algorithm for online orthogonal strip packing is 6.99, which is due to Baker and Schwarz [2].

IX. Conclusion

In this work, we studied two energy consumption scheduling problems in smart grid. The goal of this study is to reduce the peak demand under bounded delay constraint or reduce the delay under a given peak demand constraint. For each of those problems, we designed several scheduling methods and we theoretically proved that all these methods have constant approximation ratios. We also investigated online version of these problems and proposed several algorithms with proven performance guarantees. Through extensive evaluations using real data trace, we show that our methods are indeed effective in reducing peak demand or delay.

There are several interesting questions that are left for future investigation. First, we would like to extend the results studied in this work to incorporate more complex and general problem settings, *e.g.*, (1) all appliances have heterogenous release times and deadlines, (2) some appliances have to be scheduled periodically [33] such as air conditioner; (3) the energy demand and duration of operation by appliances may depend on the environment, *e.g.*, air conditioner, (4) it is also interesting to deal with users with selfish behavior [31] [29] [30] [16]. We are also interested at seeking more concrete solutions to online appliance scheduling problems that will minimize the electricity bill [35].

X. Acknowledgement

The research of authors is partially supported by NSFC under Grant No. 61170216, No. 61228202, No. 61272426, NSF under Grant CNS-0832120, CNS-1035894, ECCS-1247944, China 973 Program under Grant No.2011CB302705, and China Postdoctoral Science Foundation funded project under grant No. 2012M510029.

References

[1] Energy consumption of major household appliances shipped in canada. In *Natural Resources Canada, 2005* (Dec. 2005), Office of Energy Efficiency.

[2] Baker, B., and Schwarz, J. Shelf algorithms for two-dimensional packing problems. *SIAM Journal on Computing 12* (1983), 508.

[3] Bar-Noy, A., Feng, Y., Johnson, M., and Liu, O. When to reap and when to sow–lowering peak usage with realistic batteries. *Experimental Algorithms* (2008), 194–207.

[4] Bar-Noy, A., Johnson, M., and Liu, O. Peak shaving through resource buffering. *Approximation and Online Algorithms* (2009), 147–159.

[5] Braun, J. Reducing energy costs and peak electrical demand through optimal control of building thermal storage. *ASHRAE transactions 96*, 2 (1990), 876–888.

[6] Caron, S., and Kesidis, G. Incentive-based energy consumption scheduling algorithms for the smart grid. In *First IEEE International Conference on Smart Grid Communications (SmartGridComm),* (2010), pp. 391–396.

[7] Chase, J., Anderson, D., Thakar, P., Vahdat, A., and Doyle, R. Managing energy and server resources in hosting centers. In *ACM SIGOPS Operating Systems Review* (2001), vol. 35, pp. 103–116.

[8] Chen, B., and Vestjens, A. Scheduling on identical machines: How good is lpt in an on-line setting? *Operations Research Letters 21*, 4 (1997), 165–169.

[9] Fahrioglu, M., and Alvarado, F. Designing incentive compatible contracts for effective demand management. *IEEE Transactions on Power Systems, 15*, 4 (2000), 1255–1260.

[10] Feitelson, D., Rudolph, L., Schwiegelshohn, U., Sevcik, K., and Wong, P. Theory and practice in parallel job scheduling. In *Job Scheduling Strategies for Parallel Processing* (1997), Springer, pp. 1–34.

[11] Hurink, J., and Paulus, J. Online algorithm for parallel job scheduling and strip packing. *Approximation and Online Algorithms* (2008), 67–74.

[12] Islam, M., Balaji, P., Sadayappan, P., and Panda, D. Qops: A qos based scheme for parallel job scheduling. In *Job Scheduling Strategies for Parallel Processing* (2003), Springer, pp. 252–268.

[13] Jansen, K., and Zhang, G. On rectangle packing: maximizing benefits. In *Proceedings of ACM-SIAM symposium on Discrete algorithms* (2004), pp. 204–213.

[14] Kenyon, C., and Remila, E. Approximate strip packing. In *IEEE Symposium on Foundations of Computer Science,* (1996), pp. 31–36.

[15] Le, K., Bianchini, R., Martonosi, M., and Nguyen, T. Cost- and energy-aware load distribution across data centers. *Proceedings of HotPower* (2009).

[16] Li, X., Sun, Z., Wang, W., Chu, X., Tang, S., and Xu, P. Mechanism design for set cover games with selfish element agents. *Theoretical Computer Science 411*, 1 (2010), 174–187.

[17] Martello, S., Monaci, M., and Vigo, D. An exact approach to the strip-packing problem. *INFORMS Journal on Computing 15*, 3 (2003), 310–319.

[18] Mohsenian-Rad, A., Wong, V., Jatskevich, J., and Schober, R. Optimal and autonomous incentive-based energy consumption scheduling algorithm for smart grid. In *Innovative Smart Grid Technologies (ISGT),* (2010), pp. 1–6.

[19] Murata, H., Fujiyoshi, K., Nakatake, S., and Kajitani, Y. Rectangle-packing-based module placement. In *IEEE/ACM International Conference on Computer-Aided Design* (1995), pp. 472–479.

[20] Murata, H., Fujiyoshi, K., Nakatake, S., and Kajitani, Y. Vlsi module placement based on rectangle-packing by the sequence-pair. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 15*, 12 (1996), 1518–1524.

[21] Nghiem, T., Behl, M., Mangharam, R., and Pappas, G. Green scheduling of control systems for peak demand reduction. In *IEEE Conference on Decision and Control and European Control Conference (CDC-ECC),* (2011), pp. 5131–5136.

[22] Nusairat, A., and Li, X. WiMax/OFDM burst scheduling algorithm to maximize scheduled data. IEEE Transaction on Mobile Computing, 11(11): 1692-1705 (2012)

[23] Oldewurtel, F., Ulbig, A., Parisio, A., Andersson, G., and Morari, M. Reducing peak electricity demand in building climate control using real-time pricing and model predictive control. In *IEEE Conference on Decision and Control (CDC),* (2010), pp. 1927–1932.

[24] Ortmann, F., Ntene, N., and Van Vuuren, J. New and improved level heuristics for the rectangular strip packing and variable-sized bin packing problems. *European Journal of Operational Research 203*, 2 (2010), 306–315.

[25] Perez-Lombard, L., Ortiz, J., and Pout, C. A review on buildings energy consumption information. *Energy and buildings 40*, 3 (2008), 394–398.

[26] Schiermeyer, I. Reverse-fit: A 2-optimal algorithm for packing rectangles. *European Symposium on Algorithm* (1994), 290–299.

[27] Seiden, S., Sgall, J., and Woeginger, G. Semi-online scheduling with decreasing job sizes. *Operations Research Letters 27*, 5 (2000), 215–221.

[28] Tang, S., Li, X., Shen, X., Zhang, J., Dai, G., and Das, S. Cool: On coverage with solar-powered sensors. In *IEEE ICDCS /* (2011), pp. 488–496.

[29] Xu, P., and Li, X. TOFU: Semi-truthful online frequency allocation mechanism for wireless networks. *Networking, IEEE/ACM Transactions on 19*, 2 (2011), 433–446.

[30] Xu, P., Li, X., and Tang, S. Efficient and strategyproof spectrum allocations in multichannel wireless networks. *IEEE Transactions on Computers, 60*, 4 (2011), 580–593.

[31] Xu, P., Wang, S., and Li, X. SALSA: Strategyproof online spectrum admissions for wireless networks. *IEEE Transactions on Computers, 59*, 12 (2010), 1691–1702.

[32] Xu, X., Li, X.-Y., and Song, M., Efficient Aggregation Scheduling in Multihop Wireless Sensor Networks with SINR Constraints *IEEE Transactions on Mobile Computing*, 2012.

[33] Xu, X., Li, X., Wan, P., and Tang, S. Efficient scheduling for periodic aggregation queries in multihop sensor networks. *IEEE/ACM Transactions on Networking (TON) 20*, 3 (2012), 690–698.

[34] Ye, D., and Zhang, G. On-line scheduling of parallel jobs. *Structural Information and Communication Complexity* (2004), 279–290.

[35] Zhao J., Li., X.-Y., Payment Minimizing Job Scheduling in Smart Grid With Differential Pricing Models manuscript, 2013

[36] Zhou, Yaqin, Li, X.-Y., Liu, M., Li, Z.,, Tang, S., Mao, X., Huang, Q., Distributed link scheduling for throughput maximization under physical interference model. *IEEE INFOCOM*, 2012, pp 2691-2695

[37] Zhu, Q., David, F., Devaraj, C., Li, Z., Zhou, Y., and Cao, P. Reducing energy consumption of disk storage using power-aware cache management. In *IEE Proceedings-Software,* (2004), pp. 118–118.