Social Networking Reduces Peak Power Consumption in Smart Grid

Qiuyuan Huang, Xin Li, Jing Zhao, Dapeng Wu, Xiang-Yang Li

***Abstract*—Minimizing the peak power consumption of electrical appliances under delay requirements is shown to be NP hard. To address this, we propose a "family plan" approach that partitions users into groups and schedules users' appliances to minimize the peak power consumption of each group. Our scheme leverages the social network topology and statistical energy usage patterns of users. To partition users into groups with the potential of reducing peak power consumption, our distributed clustering scheme seeks such a partition of users into groups that the total power consumption in each group of users achieves minimum variance. Then, given a set of jobs of users' appliances to be scheduled in the next scheduling period, we use a distributed scheduling algorithm to minimize the peak power consumption of each group of users. Our simulation results demonstrate that our scheme achieves a significant reduction in user payments, peak power consumption, and fuel cost.**

***Index Terms*—Power grid, social network, family plan, distributed clustering, trace-driven simulator.**

I. Introduction

Energy consumption in buildings represents approximately 74% of the nation's electricity consumption [1]. However, electricity consumption is not efficient in most households, which results in waste of billions of dollars. To reduce the cost of power consumption by households and power generation by utility companies, techniques for scheduling jobs of electrical appliances were proposed [2], [3]; to make it work, each electrical appliance needs to be equipped with a Demand Response Switch (DRS) device[1] and can be switched on/off by the DRS device [3] and all the DRS devices follow the scheduling commands from the scheduler. These techniques seek to reduce the peak power consumption of users, since doing so can help reduce the fluctuation of power consumption, thereby improving the reliability of smart power grids, and reducing the power generation cost incurred by start-up/termination of power generators [4].

Another benefit of reducing the peak power consumption of users is that it may also reduce the payment of users if the utility company charges users based on the peak power consumption as well as the total energy consumption, which is a widely adopted utility cost paradigm in smart grid [3]. Hence, reducing the peak power consumption of users is a win-win situation for both utility companies and the electricity users.

Q. Huang, X. Li, and D. Wu are with Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611. J. Zhao and X.-Y. Li are with Department of Computer Science, Illinois Institute of Technology, Chicago, IL 60616.

[1]DRS devices are already available in the market. DRS devices can communicate with controllers/schedulers over secure tunnels in the Internet or a household network.

However, minimizing the peak power consumption of schedulable jobs of electrical appliances of users under delay requirements is shown to be NP hard [3]. Peak power consumption can be reduced by scheduling jobs of each individually controllable family (or unit), or scheduling jobs of all units served by a utility company. Clearly, scheduling jobs of all units served by a utility company will lead to a lower peak demand, but it will incur a larger communication overhead and managerial cost. To address this, we take a divide-and-conquer approach, *i.e.*, divide users into small groups and then schedule/shift jobs to minimize the peak power consumption of each group, which will attain a good balance between overhead and reduction in the Peak power to Average power Ratio (PAR).

To optimally partition users into groups, we propose a novel distributed clustering scheme, which leverages the social network topology of users and statistical energy usage patterns of users. The reason why we promote a social networking approach is that users need to be socially connected (*i.e.*, being friends of each other) to be willing to join the same group and cooperatively schedule jobs of their electrical appliances to reduce peak power consumption and save cost. This is similar to multiple friends' joining a family plan of a mobile phone company to save the cost of mobile phone bills [5]. In our approach, each user/node only shares its statistical energy usage patterns with its trusted friends/nodes who are also users of the same utility company; we call these trusted nodes 'socially connected' nodes of this user. This trusted friendship forms a social network/graph. Online social networks such as Facebook and Twitter can help create such a social graph.

Given a social graph, our clustering scheme aims at finding such groups of users that the total power consumption in each group of users (which are socially connected) achieves minimum variance, *i.e.*, we prefer to group users having negative covariance of power consumption patterns. Such clustering will lead to a natural leveling of the power consumption of users within a group as negative covariance of users implies that their periods of high power consumption are less likely to overlap (or synchronize). Since it is difficult to manage a large group and schedule jobs of users in a large group, our clustering scheme also considers the constraint on the maximum number of users allowed in a group. Different from the spectral clustering algorithms for graph partitioning [6], which are centralized, our clustering algorithm is distributed. Similar to distributed consensus protocols [7], our distributed clustering algorithm only uses local information exchange with the aim of achieving global optimality, *i.e.*, finding clusters with minimum variance.

Then, given users' jobs to be scheduled in the next scheduling period, we use an approximation algorithm, called Earliest Deadline First (EDF), to minimize the peak power consumption of each group of users. Our EDF scheduling algorithm can be implemented in a distributed manner.

Since the available data-sets do not contain electricity usage of a large number of households, we resort to simulated data. We develop a trace-driven simulator, which is based on an Auto-Regressive Moving Average (ARMA) model and an exponential distribution, to generate household energy consumption data and job data. Simulation results demonstrate that our proposed scheme achieves significant saving in user payments and fuel cost and a large reduction on peak power consumption. For example, under our scheme (i.e., the EDF scheduling for groups of users, obtained by the minimum-variance clustering), the user payment is reduced by 44.7% and the peak power is reduced by 47.7%, compared to the EDF algorithm for a single user; under our scheme, the user payment is reduced by 17.8% and the peak power is reduced by 18.9%, compared to the EDF algorithm for groups of users, obtained by random grouping.

In summary, the main contributions of this paper are 1) a novel family plan approach for reducing peak power consumption in smart grid, 2) a distributed clustering scheme, with an unprecedented scalability for a large network due to its linear complexity, 3) a scheduler, with an unprecedented capability of dealing with jobs having constrained starting times, and 4) a trace-driven simulator, with an unprecedented capability of simulating electricity usage data and job data of an unlimited number of households. Our trace-driven simulator allows us to conduct large-scale simulations for performance evaluation of techniques used in smart grid.

The rest of the paper is organized as follows. Section II describes the system under study and our family plan approach for reducing peak power consumption in smart grid. Section III presents our clustering algorithm that partitions users into groups. Section IV and Section V present the design of our simulator and simulation results, respectively. We review the related work in Section VI and conclude the paper in Section VII.

II. System Description

In this section, we describe the system and our family plan approach for reducing peak power consumption in smart grid.

A. Problem Formulation

For simplicity, we only consider one utility company and its customers. Assume the utility company has a set $\mathcal{U} = \{U_1, U_2, \cdots, U_{N_u}\}$ of N_u users. The jobs of these users are classified into two types: non-deferrable and deferrable. Non-deferrable jobs are jobs that must be run at some specific time. In contrast, deferrable jobs are schedulable within a desirable period. Examples of deferrable jobs include jobs of dish washers and plug-in electrical vehicles. We consider scheduling of jobs which will happen in next 24 hours (our method can be extended to any time interval). We partition 24 hours into T_X time periods, each of which has a length of T_L minutes. E.g., $T_L = 10$ minutes and then $T_X = 24 \times 60/10 = 144$. Let $X_{i,t}$ denote the sample average[2] of (random) total power consumption of all jobs by User i in time slot t. Then the statistical power consumption pattern of User i is defined by $\mathbf{X}_i \triangleq [X_{i,1}, X_{i,2}, \cdots, X_{i,T_X}]$. Our goal of clustering is to cluster users into groups such that the variance of power consumption of a group is minimized. Such a minimum-variance grouping can achieve more peak reduction than random grouping of users, in that our minimum-variance grouping tends to place users of different electricity usage patterns into the same group; thus the periods of high power consumption of different users are less likely to overlap, making the peak-minimizing scheduler easy to schedule jobs to achieve low peak demand. See Section V-B5 for more insights.

Let $\mathcal{G}_j$ denote the set of users in Group j. Problem 1 defines the minimum variance clustering problem.

Problem 1 (Minimum Variance Clustering Problem)**.** *Consider a set of users $\mathcal{U}$, each user i with a power consumption pattern $\mathbf{X}_i$. Then the minimum variance clustering problem is to partition users into a collection of groups $\{\mathcal{G}_j\}$, such that the maximum variance of groups is minimized.*

Problem: Minimum Variance Clustering
Objective: Minimize $\max_{j\in\{1,2,\cdots,g\}} var(\mathcal{G}_j)$
subject to:

$$\begin{cases} 1)\ \{\mathcal{G}_1, \mathcal{G}_2, \cdots, \mathcal{G}_K\} \text{ is a partition of } \mathcal{U}, \text{i.e.,} \bigcup_{j=1}^{K} \mathcal{G}_j = \mathcal{U} \\ 2)\ var(\mathcal{G}_j) = \sum_{i\in\mathcal{G}_j} var(P_i) + \sum_{i\neq m\in\mathcal{G}_j} 2cov(P_i, P_m), \forall j \end{cases}$$

$var(P_i)$ is the variance of the power consumption of User i, and can be estimated by sample variance, i.e.,

$$var(P_i) = \frac{1}{T_X}\sum_{t=1}^{T_X}(X_{i,t})^2 - (\frac{1}{T_X}\sum_{t=1}^{T_X} X_{i,t})^2. \quad (1)$$

$cov(P_i, P_m)$ is the covariance between the power consumptions of User i and User m, and can be estimated by sample covariance, i.e.,

$$cov(P_i, P_m) = \frac{1}{T_X}\sum_{t=1}^{T_X} X_{i,t}X_{m,t} - (\frac{1}{T_X}\sum_{t=1}^{T_X} X_{i,t})(\frac{1}{T_X}\sum_{t=1}^{T_X} X_{m,t}). \quad (2)$$

Our family plan approach for job scheduling is to partition socially-connected users into small groups and distributively schedule the jobs of each group. The reason why we call it family plan approach is that grouping users is like each group of users join the same family plan to cooperatively schedule/shift jobs of their electrical appliances with an aim at reducing peak power consumption and saving cost.

Next, we briefly describe the three components developed in this paper: clustering algorithm, scheduler, and simulator.

B. Clustering Algorithm

Given a social graph, our clustering algorithm aims at finding such groups of users that the total power consumption in each group of users (which are socially connected) achieves

[2]A sample average in time slot t is obtained by averaging the samples of the same slot t in different days.

minimum variance. We prove that grouping with minimum total co-variance for all pairs of users in the group is equivalent to grouping with minimum variance of the aggregated power consumption pattern of all users in the group. Observe that the latter criterion implies that the PAR of the group is small. For easy management of a family-plan group and easy scheduling of jobs of users within the group, our scheme limits the maximum number of users allowed in a group. Our distributed clustering algorithm only uses local information exchange with an aim of achieving global optimality, *i.e.*, finding clusters with minimum variance. Whenever a cluster reaches the maximum size allowed, this cluster of users stops growing and exits, *i.e.*, it will not participate in future rounds of finding clusters with minimum variance. The clustering algorithm terminates when all constrained minimum-variance clusters are found. When the clustering algorithm terminates, each user is assigned to only one cluster/group. Section III will describe this clustering algorithm in detail.

C. Scheduler

A scheduler is intended to move some jobs from peak hours to off-peak hours so that the peak power consumption in each group can be further reduced. Without loss of generality, we consider a discrete time system and a time period $[0, T]$ during which N_J jobs of electrical appliances, $\mathbf{J} = \{J_1, J_2, \cdots, J_{N_J}\}$, need to be scheduled. Here T is the length of the scheduling period; typically T is 24 hours. The i-th job J_i has a demand profile D_i parameterized by $(d_i, \tau_i, t_i^e, t_i^l)$, where d_i denotes J_i's average power consumption[3], τ_i denotes its duration, t_i^e and t_i^l denote the earliest and latest starting time of the job, respectively. The starting time s_i of the job must satisfy $t_i^e \le s_i \le t_i^l$. 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 any time slot in interval $[a, b]$ and 0 at any time slot in $[0, a)$ or $(b, T]$. We assume that the job cannot be interrupted once it starts[4]. Under a given schedule, the total load at time t is

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

Our scheduler is intended to minimize the peak demand. The problem is formulated as below.

Problem 2 (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 $T > 0$, is formulated as:*

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

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

[3]If we consider instantaneous power consumption, d_i will be a function of time. This case will be considered in our future work.

[4]We will consider interruptible jobs in our future work.

For the special case that $t_i^e = 0$ and $t_i^l = T - \tau_i$, $\forall J_i \in \mathbf{J}$, Problem 2 has been proven to be NP-hard [3]. Hence, Problem 2 in the general case is also NP-hard.

Given users' jobs to be scheduled in the next scheduling period, we use an approximation algorithm, called Earliest Deadline First (EDF), shown in Algorithm 1, to minimize the peak power consumption of each group of users. Note that Algorithm 1 can be applied to 1) jobs of a single user, and 2) jobs of all the users in one family-plan group. In Algorithm 1, jobs are partitioned into two subsets: long job set $\mathbf{J_l}$ and short job set $\mathbf{J_s}$; if the job duration τ_i is less than $\frac{T}{a}$ (a is a user-specified parameter and $1 < a \le 2$), then job J_i belongs to the short job set $\mathbf{J_s}$; otherwise, it belongs to the long job set $\mathbf{J_l}$. For each long job J_i in $\mathbf{J_l}$, it is scheduled to start at its earliest starting time, i.e., $s_i = t_i^e$. Note that a long job cannot be sequentially combined with another long job in the time period $[0, T]$, i.e., once a long job J_i ends, there is not enough remaining time to run another long job J_k because $\tau_i + \tau_k > T$ for any long job J_i and any long job J_k ($i \ne k$). In contrast, it is possible to schedule multiple short jobs in a sequential manner if the sum of their durations $\sum_i \tau_i$ is not more than T. For short jobs $J_i \in \mathbf{J_s}$, these jobs are sorted in the increasing order of their latest starting time t_i^l (i.e., we sort short jobs from the most urgent job to the least urgent job); if multiple jobs have the same t_i^l, then further sort these jobs in the decreasing order of their power demand d_i. Then schedule the sorted short jobs, using a greedy algorithm, i.e., schedule the more urgent job first; if multiple jobs have the same t_i^l, schedule a job with a larger power demand first. In Step 13 of Algorithm 1, $e_{k_{i-1}}$ denotes the end time of job $J_{k_{i-1}}$.

Algorithm 1 Earliest Deadline First (Centralized)

Input: Job profile $D_i = (d_i, \tau_i, t_i^e, t_i^l)$, $\forall i \in \{1, \cdots, N_J\}$; parameter a and T.
Output: Job starting time s_i, $\forall i \in \{1, \cdots, N_J\}$.

1: **for** each job J_i ($i = 1, \cdots, N_J$) **do**
2: **if** $\tau_i < \frac{T}{a}$ **then**
3: Put job J_i into the short job set $\mathbf{J_s}$;
4: **else**
5: Put job J_i into the long job set $\mathbf{J_l}$;
6: **for** each job $J_i \in \mathbf{J_l}$ **do**
7: $s_i = t_i^e$;
8: Sort short jobs $J_i \in \mathbf{J_s}$ in the increasing order of their latest starting time t_i^l with a tie break rule that a job of larger power demand d_i comes first; denote the sorted jobs by $J_{k_1}, \cdots, J_{k_m}$, where $m = |\mathbf{J_s}|$ and $|\mathbf{J_s}|$ is the cardinality of set $\mathbf{J_s}$;
9: **for** each job $J_{k_i} \in \mathbf{J_s}$ ($i = 1, \cdots, m$) **do**
10: **if** $i == 1$ **then**
11: $s_{k_i} = t_{k_i}^e$;
12: **else**
13: $e_{k_{i-1}} = s_{k_{i-1}} + \tau_{k_{i-1}}$;
14: **if** $t_{k_i}^l \ge e_{k_{i-1}}$ **then**
15: $s_{k_i} = \max\{e_{k_{i-1}}, t_{k_i}^e\}$;
16: **else**
17: $s_{k_i} = t_{k_i}^e$;

Algorithm 1 can deal with both deferrable jobs and non-deferrable jobs. Both deferrable jobs and non-deferrable jobs can be characterized by a demand profile $(d_i, \tau_i, t_i^e, t_i^l)$ where i is the job index. For a deferrable job i, $t_i^e < t_i^l$, i.e., there is a gap between the earliest starting time and the latest starting

time so that there is flexibility to schedule the job during $[t_i^e, t_i^l]$. For a non-deferrable job i, $t_i^e = t_i^l$, i.e., the earliest starting time equals the latest starting time; hence, the job must start at time t_i^e.

We would like to make a remark regarding the sub-optimality of Algorithm 1. In fact, Algorithm MP in [3] is a special case of Algorithm 1 in this paper; that is, for the special case that each job J_i has the same earliest starting time $t_i^e = 0$ and the same latest job completion time $t_i^l + \tau_i = T$, Algorithm 1 in this paper reduces to Algorithm MP in [3]. For this special case, it can be proven [3] that Algorithm 1 has an approximation ratio of at most 7, i.e., in the worse case, the peak power produced by Algorithm 1 is not larger than 7 times the optimal peak power. For a general case, it is difficult to find a tight approximation ratio.

Different from the centralized Algorithm MP in [3], in this paper, we can implement the EDF algorithm in a distributed manner as follows: each user can exchange information about its jobs (to be scheduled) with other users in the same group, and each user can use Algorithm 1 (applied to all the jobs of the users in the same group) to find a sub-optimal schedule of its own jobs; each user, actually, the household control center of each user, applies the resulting schedule of the jobs (via the DRS devices) to its electrical appliances.

D. Simulator

Since the available data-sets do not contain electricity usage of a large number of households, we resort to simulated data. We develop a trace-driven simulator, which is based on an Auto-Regressive Moving Average (ARMA) model and an exponential distribution to generate household energy consumption data and job data. Section IV will describe our simulator in detail.

E. Practicality of Our Scheme

We would like to emphasize that our family-plan approach is practical. Here is our suggested implementation. 1) Each customer/user (of the same utility company) who is willing to participate in the family plans can create an account on Facebook. 2) Each of these users can download a matching software of the utility company from the Facebook App store; this matching App will run Algorithm 3. Note that a user only exchanges its state vector with its neighbors (e.g., its trusted friends on Facebook), in Step 7 of Algorithm 3. The matching App will partition all the users into groups, and notify each user of its group membership. If all users agree with the assigned group membership, we move on to the next phase (scheduling); otherwise, we remove those groups who already agreed to the assigned membership and re-run the matching App for the remaining users until all users are assigned with group memberships. 3) In the scheduling phase, each user can download a scheduling App from the Facebook App store; each user will manually enter those jobs that are automatically switched on/off by its DRS devices; then the scheduling App of each user exchanges information about its jobs (to be scheduled) with the scheduling App of other users in the same group; then the scheduling App of each user runs Algorithm 1 (applied to all the jobs of the users in the same group) and obtains a sub-optimal schedule of its own jobs; then the scheduling App of each user applies the resulting schedule of the jobs (via the DRS devices) to its electrical appliances.

From the above discussion, it can be seen that our family-plan approach consists of two control strategies, which run in two different time scales.

1) The first control strategy is to partition users into groups so that the scheduling problem for each cluster can be solved with low complexity (otherwise, the complexity of the scheduling problem is NP hard) and with a potential of achieving small fluctuation of power consumption in each cluster. This control runs in a long time scale, e.g., two years of contract for a family plan like that for mobile cell phones. To optimally partition users into groups (in the sense of minimum variance), we propose a distributed clustering scheme, which leverages the social network topology of users and statistical energy usage patterns of users in a long time scale.
2) The second control strategy is to schedule the jobs of electric appliances in a short time scale (e.g., every minute) so as to actually minimize the fluctuation of power consumption in each cluster.

There is a potential problem for the above approach. If a user does not disclose its true urgency of its jobs, e.g., always declaring higher urgency for all its jobs than necessary, then the user will gain an advantage of having its jobs complete with less delay. To address this problem, we propose the following mechanism. After all the users agree with the assigned group membership, they need to sign a contract with the utility company for a duration, say, two years. The two-year contract stipulates that if the PAR of a group is less than a given threshold, each user in the group receives a low electricity rate; otherwise, each user in the group receives a high electricity rate. Under this mechanism, each user in the same group has to disclose its truthful degree of urgency of jobs; otherwise, it will increase the PAR of the group, resulting in a penalty for each user in the cluster. Such a contract produces a win-win situation for both the utility company and the electricity consumers; i.e., the electricity consumers will enjoy a low electricity rate while the utility company will have smoother power demand, resulting in reduced ramp-up/ramp-down cost of power generators. This reduced ramp-up/ramp-down cost of power generators allows the utility company to compensate the low electricity rate offered to its customers who signed the two-year contract.

III. Clustering Schemes

Let $P_i(t)$ denote the instantaneous power consumption of User i in time slot t. Let $var(X)$ be the variance of random variable X and $cov(X,Y)$ be the covariance between X and Y. Then we have

Fact 1.

$$\begin{cases} var(X+Y) = var(X) + var(Y) + 2 \times cov(X,Y) \\ var(X + \sum_i Y_i) = var(X) + var(\sum_i Y_i) + \sum_i 2cov(X, Y_i) \end{cases} \quad (3)$$

From Fact 1, we know that a new user i can be added to a group $\mathcal{G}_g$ of users to further reduce the variance of the power consumption of all users in the group if and only if

$$var(P_i) < -\sum_{j\in\mathcal{G}_g} 2cov(P_i, P_j)$$

Our goal is to find a clustering so that the variance of summed power consumption is as small as possible.

Let G_u denote the social graph formed by N_u users; the graph G_u has a node set V_u and an edge set E_u. Denote $|V_u|$ the cardinality of set V_u. Assume that N_u users are partitioned into multiple groups, and the members of Group j form a set denoted by $\mathcal{G}_j$. Notice that $var(\sum_i P_i) = \sum_i var(P_i) + \sum_{i\neq j} 2cov(P_i, P_j)$. Then minimal variance clustering problem studied here can be reduced to a graph clustering problem: each (user) node $i \in V_u$ has a weight $var(P_i)$, and each edge (i, j) has a weight $2cov(P_i, P_j)$. Then we need to find a clustering such that the maximum cluster weight (*i.e.*, sum of weights of nodes and edges within the cluster) is minimized. This problem is obviously NP-hard as many NP-hard problems are special cases of this problem (*e.g.*, subset sum problem).

For ease of understanding, we first present a centralized version of our clustering algorithm in Algorithm 2; then present a distributed version in Algorithm 3.

Algorithm 2 outputs groups of users $\{\mathcal{G}_j\}$ and the number of groups is not an input parameter. Algorithm 2 is a clustering algorithm that partitions users into K groups where K is unknown to the algorithm. This is different from K-means clustering algorithm and spectral clustering algorithms.

In Algorithm 2, each user/node i is assigned with a state vector $[g_i, \sigma^2_{g_i}, \mathcal{G}_{g_i}]$, where g_i is the index of the group $\mathcal{G}_{g_i}$, to which User i is assigned, and $\sigma^2_{g_i}$ is the variance of $\sum_{j\in\mathcal{G}_{g_i}} P_j$. Let $\mathcal{N}(i)$ denote the set of nodes that are neighbors of Node i in graph G_u. $var(P_i)$ can be estimated by Eq. (1) and $cov(P_i, P_m)$ can be estimated by Eq. (2).

Algorithm 2 is a greedy algorithm. It begins from an initial setting, i.e., each node forms a group. Then each iteration (from Step 5 to Step 18) improves over the previous iteration, i.e., each iteration is guaranteed to achieve a smaller variance than the previous iteration; if there is no change between two iterations, the algorithm stops.

The following proposition shows that Algorithm 2 has linear complexity per iteration.

Proposition 1. *The computational complexity per iteration of Algorithm 2 is* $O(|E_u|)$.

Proof. In Algorithm 2, initializing every node with its group index requires $O(N_u)$ time. Each iteration of Algorithm 2 takes linear time in the number of edges ($O(|E_u|)$). At each node i, we first group the neighbors according to their group index, with complexity $O(d_i)$, where d_i is the node degree of Node i. We then pick the group of minimum variance and assign Node i to this group, requiring a worst-case time of $O(d_i)$. This process is repeated at all nodes and hence an overall time is $O(|E_u|)$ for each iteration. □

In Algorithm 2, as the number of iterations increases, the number of nodes that finalize their minimum-variance-group membership increases. From our experiments, we found that irrespective of N_u, 95% of the nodes or more finalize their group membership by the end of the fifth iteration. Since the number of iterations is independent of $|E_u|$ (as verified in our experiments), the overall complexity of Algorithm 2 is $O(|E_u|)$.

Algorithm 2 Minimum-Variance Clustering (Centralized)

Input: Group size constraint N_c, social graph G_u, $\mathbf{X}_i$ ($\forall i \in V_u$).
Output: $\{\mathcal{G}_j\}$.

1: **for** i from 1 to $|V_u|$ **do**
2: (Initialization) $g_i = i$;
3: (Initialization) $\sigma^2_{g_i} = var(P_i)$;
4: (Initialization) $\mathcal{G}_i = \{i\}$;
5: **while** $|V_u| \neq 0$ **do**
6: **for all** $i \in V_u$ **do**
7: For each distinct g_k ($k \in \mathcal{N}(i)$) and $g_k \neq g_i$, calculate $\gamma_{g_k} = var(P_i) + \sigma^2_{g_k} + \sum_{m\in\mathcal{G}_{g_k}} 2cov(P_i, P_m)$;
8: Let $\gamma_i^{min} = \min_{k\in\mathcal{N}(i)\&g_k\neq g_i} \gamma_{g_k}$;
9: $k^* = \arg\min_{k\in\mathcal{N}(i)\&g_k\neq g_i} \gamma_{g_k}$;
10: **if** $\gamma_i^{min} < \sigma^2_{g_i}$ **then**
11: $\sigma^2_{g_i} = \sigma^2_{g_i} - var(P_i) - \sum_{m\in\mathcal{G}_{g_i}\setminus\{i\}} 2cov(P_i, P_m)$;
12: Move User i from $\mathcal{G}_{g_i}$ to $\mathcal{G}_{g_{k^*}}$;
13: **if** $|\mathcal{G}_{g_i}| == 0$ **then**
14: Remove $\mathcal{G}_{g_i}$;
15: $g_i = g_{k^*}$;
16: $\sigma^2_{g_i} = \gamma_i^{min}$;
17: **if** $|\mathcal{G}_{g_{k^*}}| == N_c$ **then**
18: Output $\mathcal{G}_{g_{k^*}}$ and remove all nodes in $\mathcal{G}_{g_{k^*}}$ and associated links from graph G_u;
19: **if** no change in any group **then**
20: Output all groups and remove all nodes and links from graph G_u

Algorithm 3 can run in an asynchronous manner to avoid inconsistencies in updating the group membership.. In an asynchronous manner, in each slot, only one node is allowed to update its group membership based on information collected in the previous slot. There are many ways of choosing a node for update. One method is to use a token. A node needs to have a token to update its group membership. First randomly select a node and assign a token to this node. After finishing update, this node randomly relays the token to one of its neighbors. This relay process continues till Algorithm 3 terminates. Since grouping does not have stringent delay requirements, the delay caused by asynchronous implementation of Algorithm 3 is tolerable. It is possible to speed up Algorithm 3 by a synchronous manner. We leave this for future study.

Since each user may not want to share its statistical energy usage pattern with others, we suggest the following method to address this privacy issue. Each user can encrypt the data before sending it to another party. A receiving node can use the encrypted data to compute without knowing the actual value of the data [8].

Remark 1. *The centralized clustering algorithm (i.e., Algorithm 2) requires a central controller, i.e., each user has to*

Algorithm 3 Minimum-Variance Clustering (Distributed)

Input: Group size constraint N_c, social graph G_u, $\mathbf{X}_i$ ($\forall i \in V_u$).
Output: $\{\mathcal{G}_j\}$.

1: **for** each user i ($i = 1, \cdots, |V_u|$) **do**
2: (Initialization) $g_i = i$;
3: (Initialization) $\sigma^2_{g_i} = var(P_i)$;
4: (Initialization) $\mathcal{G}_i = \{i\}$;
5: **for** each user i ($i = 1, \cdots, |V_u|$) **do**
6: **if** User i has not finalized its membership with a group **then**
7: User i exchanges its state vector $[g_i, \sigma^2_{g_i}, \mathcal{G}_{g_i}]$ with each of its neighbors User k ($k \in \mathcal{N}(i)$);
8: User i requests for $\mathbf{X}_m$ from User m if $m \in \mathcal{G}_{g_k}$ ($k \in \mathcal{N}(i)$) and $\mathbf{X}_m$ is not stored in the memory of User i;
9: For each distinct g_k ($k \in \mathcal{N}(i)$) and $g_k \neq g_i$, User i calculates $\gamma_{g_k} = var(P_i) + \sigma^2_{g_k} + \sum_{m \in \mathcal{G}_{g_k}} 2cov(P_i, P_m)$;
10: Let $\gamma_i^{min} = \min_{k \in \mathcal{N}(i) \& g_k \neq g_i} \gamma_{g_k}$;
11: $k^* = \arg\min_{k \in \mathcal{N}(i) \& g_k \neq g_i} \gamma_{g_k}$;
12: **if** $\gamma_i^{min} < \sigma^2_{g_i}$ **then**
13: $\sigma^2_{g_i} = \sigma^2_{g_i} - var(P_i) - \sum_{m \in \mathcal{G}_{g_i} \setminus \{i\}} 2cov(P_i, P_m)$;
14: User i notifies each of the users in $\mathcal{G}_{g_i}$ about its leaving the group, and sends the new value of $\sigma^2_{g_i}$ to these users;
15: Each of the users in $\mathcal{G}_{g_i}$ removes User i from $\mathcal{G}_{g_i}$, and update the value of $\sigma^2_{g_i}$;
16: $g_i = g_{k^*}$;
17: $\sigma^2_{g_{k^*}} = \gamma_i^{min}$;
18: User i notifies each of the users in $\mathcal{G}_{g_{k^*}}$ about its joining the group, and sends the new value of $\sigma^2_{g_{k^*}}$ to these users;
19: Each of the users in $\mathcal{G}_{g_{k^*}}$ add User i to $\mathcal{G}_{g_{k^*}}$, and update the value of $\sigma^2_{g_{k^*}}$;
20: **if** $|\mathcal{G}_{g_{k^*}}| == N_c$ **then**
21: Each of the users in $\mathcal{G}_{g_{k^*}}$ finalizes its membership with $\mathcal{G}_{g_{k^*}}$;

report its jobs (to be scheduled) to the central controller. An issue is how to select a central controller. Will every user agree with such a selection of central controller? Who wants to be the central controller? A disadvantage of a central controller is that it becomes a single point of failure, which is not robust; in other words, if the central controller breaks down, we cannot run Algorithm 2. The distributed clustering algorithm (i.e., Algorithm 3) does not have any of these issues that we just mentioned.

Besides system robustness, there is another important reason for using the distributed clustering algorithm (Algorithm 3). In our suggested practical implementation of our proposed family-plan approach in Section II-E, the distributed clustering algorithm (Algorithm 3) is required since there is no centralized controller in the system.

IV. Design of Trace-Driven Simulator

In this section, we present our design of trace-driven simulator for large-scale simulations in smart grid. In Section IV-A, we describe our household electricity consumption simulator. Section IV-B presents our job demand profile simulator.

A. Household Electricity Consumption Simulator

To evaluate the performance of our proposed clustering scheme, we need electricity consumption data for a large number of households. Specifically, for each household i, we need $\{X_{i,t}\}$ the statistical-average electricity consumption as a function of time t ($t \in \{1, 2, \cdots, T_X\}$). However, such data for a large number of households is not available in the public domain. We only obtained such data for six households [9]. In this section, we will describe our design of a trace-driven simulator, which is capable of generating simulated $\{X_{i,t}\}$ for an unlimited number of households. The simulated $\{X_{i,t}\}$ is statistically similar to the real-world $\{X_{i,t}\}$, i.e., they have similar probability distribution functions and auto-correlation functions.

Assume that the real-world $\{X_{i,t}\}$ is a stationary stochastic process and can be modelled by an autoregressive moving-average (ARMA) process, which is defined as below [10].

Definition 1. *(ARMA(p,q) Process) The process $\{X_t, t = 0, \pm 1, \pm 2, ...\}$ is said to be an ARMA(p,q) process if $\{X_t\}$ is stationary and if for every t,*

$$X_t - \sum_{i=1}^{p} \phi_i X_{t-i} = Z_t + \sum_{j=1}^{q} \theta_j Z_{t-j} \tag{4}$$

where $\{Z_t\}$ are independent, identically distributed Gaussian random variables with zero mean and variance σ_n^2.

To determine an ARMA model from samples, we need to determine the orders p, q, parameters $\{\phi_i\}$, $\{\theta_j\}$ and the white noise variance σ_n^2. The problem of determining orders p and q is a model selection problem. A typical criterion for model selection is Akaike information criterion (AIC). However, for autoregressive models, the AIC has a tendency to overestimate p. The Bayesian information criterion (BIC) is another criterion, which attempts to correct the overfitting nature of the AIC. For a zero-mean ARMA(p,q) process, BIC is given by

$$BIC = (N_s - p - q)\ln[\frac{N_s \widehat{\sigma}_n^2}{N_s - p - q}] + N_s(1 + \ln\sqrt{2\pi}) + (p+q)\ln[\frac{\sum_{t=1}^{N_s} X_t^2 - N_s \widehat{\sigma}_n^2}{p+q}] \tag{5}$$

where $\widehat{\sigma}_n^2$ is the maximum likelihood estimate of the white noise variance σ_n^2, and N_s is the number of samples. The BIC provides a consistent order selection procedure in the sense that if the samples $\{X_1, ..., X_{N_s}\}$ are truly measurements of an ARMA(p,q) process, and if $\widehat{p}$ and $\widehat{q}$ are the estimated orders found by minimizing the BIC, then $\widehat{p} \to p$ and $\widehat{q} \to q$ with probability one as $N_s \to \infty$. Thus, we determine the order p and q by minimizing BIC: $\{\widehat{p}, \widehat{q}\} = \arg\min_{p,q} BIC$.

Algorithm 4 shows our procedure for generating electricity consumption samples of a simulated household. The first part of Algorithm 4 is to use model selection and parameter estimation to determine the best-fit model and optimal parameter values. The techniques used in Steps 2 and 9 can be found in Ref. [10]. The second part is to use the resulting model to generate simulated electricity consumption process $\{\widehat{X}_{i,t}\}$.

B. Job Demand Profile Simulator

To evaluate the performance of our scheduler, we need job data for a large number of households. Specifically, for each household, we need the demand profile of each job.

Algorithm 4 Household Electricity Consumption Simulator

Input: p_{max}, q_{max}, and electricity consumption measurements of a real-world household $\{X_{i,t}\}$.
Output: Electricity consumption samples of a simulated household $\{\widehat{X}_{i,t}\}$.

$p_{min} = 1$;
$q_{min} = 1$;
Calculate autocorrelation function (**ACF**) $R(\tau)$ and partial autocorrelation function (**PACF**) $\alpha(\tau)$, where τ is a time lag;
Determine the model type according to the features of ACF and PACF, i.e., determine whether it is an AR(p), or MA(q), or ARMA(p, q) process;
if it is an AR(p) process **then**
    $q_{min} = 0$;
    $q_{max} = 0$;
if it is an MA(q) process **then**
    $p_{min} = 0$;
    $p_{max} = 0$;
for $p = p_{min}$ **to** p_{max} **do**
    for $q = q_{min}$ **to** q_{max} **do**
        Estimate parameters $\{\phi_i, \theta_j : i = 1, ..., p; j = 1, ..., .q\}$;
Find $p \in \{p_{min}, \cdots, p_{max}\}$ and $q \in \{q_{min}, \cdots, q_{max}\}$ that minimize BIC in Eq. (5);
Use the resulting model to generate samples $\{\widehat{X}_{i,t}\}$;

Examples of jobs are listed in Table I. For the k-th job $J_{i,k}$ of User i, it has a demand profile $D_{i,k}$ parameterized by $(d_{i,k}, \tau_{i,k}, t^e_{i,k}, t^l_{i,k})$, where $d_{i,k}$ denotes $J_{i,k}$'s average power consumption, $\tau_{i,k}$ denotes its duration, $t^e_{i,k}$ and $t^l_{i,k}$ denote the earliest and latest starting time of the job, respectively. However, such job data for a large number of households is not available in the public domain. We only obtained such job data for six households [9]. In this section, we will describe our design of a job profile simulator, which is capable of generating $D_{i,k}$ for an unlimited number of households. The statistical averages of simulated $\{D_{i,k}\}$ need to match $\{\widehat{X}_{i,t}\}$ generated by Algorithm 4, i.e., the expectation $E[\sum_{k\in\{k:t^e_{i,k}\in[a,b-\tau_{i,k}]\&t^l_{i,k}\in[a,b-\tau_{i,k}]\}} d_{i,k}\times\tau_{i,k}]$ should equal $\sum_{t=a}^{b}\widehat{X}_{i,t}$ for all a, b ($0 \le a < b \le T$).

For simplicity, we assume all jobs have constant instantaneous power consumption[5] and constant duration[6]. For the same household i used in Algorithm 4, we also obtained the real-world power measurements of each job of Household i. The average power, duration, and frequency of each job for one of the six households are listed in Table I. Note that each of the six real-world households has such a table of job profile; so we have six different tables of job profile. To save space, we only list one household's job profile in Table I.

To generate the earliest starting time of each job of the j-th type for User i, we use an exponential distribution to generate the interval between two consecutive jobs of the same type (j-th type), denoted by β_j. That is, the probability density function of β_j is

$$p(\beta_j) = \lambda_j e^{-\lambda_j \beta_j} \quad (6)$$

[5]For simplicity, we use average power to replace instantaneous power.

[6]If the power consumption and duration of a job are assumed to be random, we can use Gamma distribution to model them since power consumption and duration are positive-real-valued and Gamma distribution has a support of $(0, \infty)$ and is versatile in modeling various shape of distribution. We will leave this for future work.

TABLE I
PROFILE OF JOBS IN A HOUSEHOLD

Job name	Avg power (W)	Avg duration (sec)	Avg frequency (times/day)
Oven	1840	330	2.44
Dishwasher	450	7020	0.23
Refrigerator	191	360	19.33
Kitchen Outlet	16	86400	2.33
Lighting	81	210	123.32
Washer Dryer	267	1590	0.68
Microwave	1580	30	8.6
Bathroom	1618	210	0.82
Stove	593	210	0.36
Disposal	28	60	0.36
Furnace	610	570	6.88
Sub-panel	239	3510	0.63

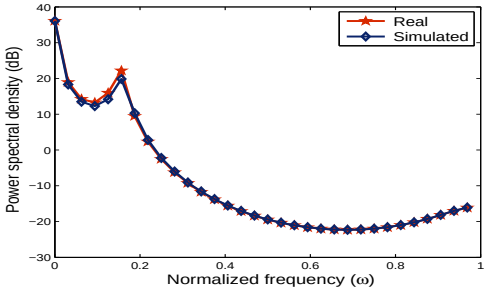

Fig. 1. Power spectral density of real-world data and simulated data

where λ_j is the average frequency of the j-th type of job, i.e., a value in the fourth column of Table I. Since λ_j is in unit of times/day, β_j needs to be in unit of one day (24 hours). We generate the first $\beta_{j,1}$ using the distribution in (6). If $\beta_{j,1} \ge 1$ day, no job is generated for the j-th type of job for this user and move on to generate another type of job; else then let $t^e_{i,1} = \beta_{j,1}$ and then generate the second $\beta_{j,2}$. If $t^e_{i,1} + \beta_{j,2} \ge 1$ day, then $t^l_{i,1} = 1 - \tau_{i,j}$ (where $\tau_{i,j}$ is the duration of the j-th type of job of User i and is in the third column of Table I) and move on to generate another type of job; else then let $t^e_{i,2} = t^e_{i,1} + \beta_{j,2}$ and $t^l_{i,1} = t^e_{i,2} - \tau_{i,j}$. Repeat this process for each type of job for this user until $E[\sum_{k\in\{k:t^e_{i,k}\in[a,b-\tau_{i,k}]\&t^l_{i,k}\in[a,b-\tau_{i,k}]\}} d_{i,k}\times\tau_{i,k}] = \sum_{t=a}^{b}\widehat{X}_{i,t}$ for all a, b ($0 \le a < b \le T$). We usually choose $b - a = 10$ minutes.

V. Simulation Results

A. Simulation Setting

1) Network Topology Simulator: To evaluate the performance of our clustering scheme, we adopt widely-used LFR random network generator [11], [12] to generate large-scale realistic social network topologies, which have the following properties: 1) the node degree follows a power law distribution, and 2) the distribution of cluster/community sizes also follows a power law distribution. We assume that there are 1000 households in the system. So we use the LFR network generator to generate a small-world scale-free network of 1000 nodes, representing a social network.

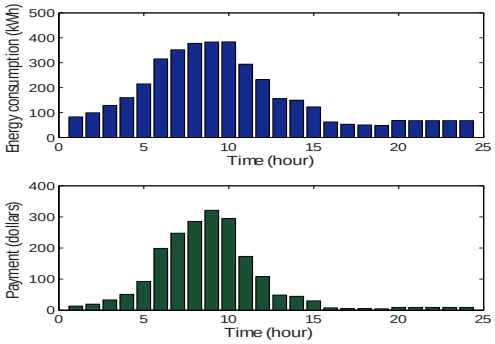

Fig. 2. Energy consumption and payment without family plan

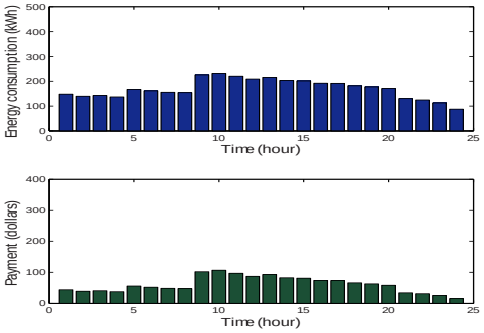

Fig. 3. Energy consumption and payment under family plan ($N_c = 5$)

2) Measurement Data: Our trace-driven simulator needs real-world electricity consumption measurements as input. From [9], [13], we obtained a set of several weeks of power data for 6 different homes, which includes two types of power data: 1) the whole household power consumption measured at a rate of 1 Hz, and 2) power consumption of up to 24 individual appliances in a home, measured at a rate of 0.5 Hz. Using the power measurements of each appliance in a household, we obtain Table I, which is used as job profiles.

3) Generation of Household Electricity Consumption Data: Algorithm 4 takes the power measurements of each of the six household as input to generate 200 electricity consumption processes $\{\widehat{X}_{i,t}\}$ ($i = 1, \cdots, 200$; $t = 1, \cdots, T_X$). Since we have power measurements of six households, we generate a total of 1200 electricity consumption processes. Since there are 1000 households in the system, we randomly pick 1000 out of the 1200 processes and assign each selected process to one household in the social network.

4) Generation of Job Demand Profile Data: As described in Section IV-B, given $\{\widehat{X}_{i,t}\}$ and Table I, the job demand profile simulator generates job demand profiles for each of the 1000 households in the social network.

5) Energy Pricing Model: To evaluate how much cost is saved under our proposed scheme, we need an energy pricing model to convert user power consumption into user payment. In our simulations, we use a quadratic pricing model [2] as below

$$C_t = r \times (L_t)^2, \tag{7}$$

where C_t is the price/user-cost in time slot t, L_t is the total load/energy-consumption in slot t, and r is a rate and we set $r = 0.02$. Since (7) penalizes more on large power consumption than on small power consumption, a load schedule that minimizes the peak power consumption would obviously reduce the user cost at the same time. This pricing model has been used by utility companies as incentives for users to comply with peak-minimization load schedules.

B. Performance of Proposed Schemes

In this section, we evaluate the performance of our proposed scheme by simulation.

1) Accuracy of Our Trace Driven Simulator: Fig. 1 shows the power spectral density (PSD) of the real-world data and the simulated data generated by Algorithm 4. It can be observed that the PSD of the simulated data agrees very well with that of the real-world data, indicating high accuracy of Algorithm 4.

2) PAR and User Payment under Family Plan Scheme: In this section, we compare our family plan scheme with the case of not using family plan. Fig. 2 shows energy consumption and user payment as a function of time for the case of not using family plan. Fig. 3 shows energy consumption and user payment as a function of time for our family plan scheme. The user payment/cost is obtained by (7). As shown in Fig. 2, when the family plan scheme is not used, the PAR for 1000 households is 2.69 and the total user payment is \$2842.3. In contrast, when the family plan scheme is used (under $N_c = 5$), the PAR reduces to 1.45 (46.1% less) and the total user payment reduces to \$1693.4 (40.4% less). In addition, when the family plan scheme is used, the load becomes much smoother, compared to no family plan.

3) Fuel Cost under Family Plan Scheme: To show the gain of reducing peak power consumption from the utility company perspective, we use the values in Tables II and III [4] to compute the fuel cost for one million households. The reason why we consider one million households is because we need one million households to meet the output power of a power generator; the demand of 1000 households is not sufficient to meet the minimum power generated by a power generator in Table II.

TABLE II
CHARACTERISTICS OF GENERATORS

Unit	Minimum power (MW)	Maximum power (MW)
G_1	50	250
G_2	50	200

TABLE III
FUEL DATA

Unit	a (MBtu)	b (MBtu/MWh)	c ($_{(MBtu/MM^2h)}$)	Start-up fuel (MBtu)	Fuel price ($/MBtu)
G_1	0.0024	12.33	28	1500	1
G_2	0.0044	13.29	39	1500	1

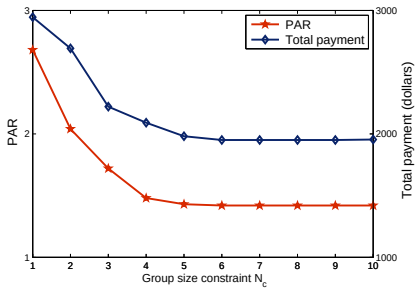

Fig. 5. PAR and user payment vs. group size constraint N_c

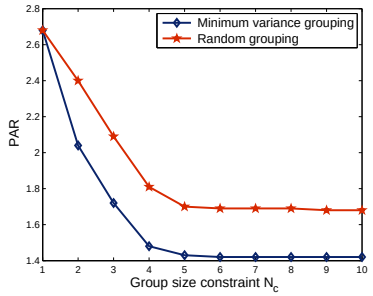

Fig. 6. PAR: minimum variance grouping vs. random grouping

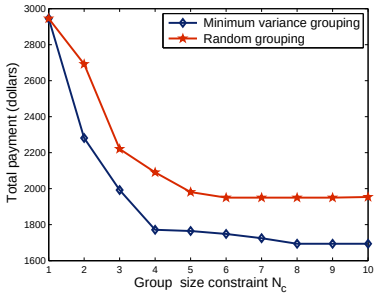

Fig. 7. Payment: minimum variance grouping vs. random grouping

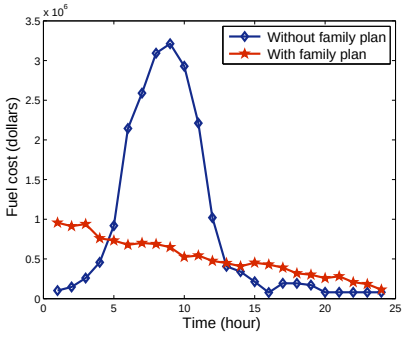

Fig. 4. Fuel cost in each hour.

We assume two thermal generators provide power for one million households. We also assume that the energy consumption of users scales with the number of users, which is reasonable; hence, the energy consumption of these one million households is simply 1000 times the energy consumption of the thousand households obtained in Section V-B2. From Table III, Generator G_1 costs less than Generator G_2 if both produce the same amount of power. So, as long as G_1 can satisfy the demand, we would always use G_1 instead of G_2. Let L_t denote the power demand/load in time slot t. If $L_t \leq 250$ MW, only G_1 is on. The amount of fuel needed [4] is $a+b\times L_t+c\times L_t^2 = 0.0024+12.33L_t+28L_t^2$ (MBtu) and it costs $0.0024+12.33L_t+28L_t^2$ dollars since 1 MBtu costs 1 dollar according to Table III. When L_t becomes larger than 250 MW, G_2 also needs to be activated and the cost of start-up fuel 1500 (MBtu) needs to be added. If $250 < L_t \leq 450$, both G_1 and G_2 need to be on. Since G_1 has higher efficiency, G_1 should produce 250 MW and G_2 should produce $(L_t - 250)$ MW, and the cost of fuel is $0.0024+12.33\times 250+28\times 250^2+0.0044+13.29(L_t-250)+39(L_t-250)^2$. If $L_t > 450$, some load has to be shed.

Using the energy consumption obtained in Section V-B2 (magnified by 1000 times), we calculate the fuel cost in each hour, shown in Fig. 4. In 24 hours, the total fuel cost for no family plan is \$21,343,000; and the total fuel cost under family plan is \$11,265,000. Hence, the family plan approach saves 47.22% in fuel cost.

4) Effect of N_c: In this section, we study the effect of group size constraint N_c on the performance of our family plan scheme. Fig. 5 shows PAR and user payment as a function of N_c for our family plan scheme. The case for $N_c = 1$ corresponds to no family plan. It can be observed that the larger N_c, the lower PAR and the lower user payment. The reason is obvious: the larger group size, the more jobs to be scheduled, resulting in lower PAR under our peak-minimizing scheduler. But the catch is that the strategy for grouping needs to be carefully selected; random grouping + peak-minimizing scheduling is not optimal; it can be improved by minimum-variance grouping + peak-minimizing scheduling. We demonstrate this in the next section.

5) Minimum-Variance Grouping vs. Random Grouping: In this section, we compare performance of random grouping + peak-minimizing scheduling with minimum-variance grouping + peak-minimizing scheduling. Fig. 6 shows PAR as a function of N_c for the minimum-variance grouping and the random grouping. Fig. 7 shows user payment as a function of N_c for the minimum-variance grouping and the random grouping. It can be observed that the minimum-variance grouping achieves much better performance than the random grouping. For $N_c = 3$, our minimum-variance grouping helps reduce PAR and user payment by 17.0% and 11.6%, respectively, compared to random grouping. The reason of this performance gain is that different from random grouping, our minimum-variance grouping tends to place users of different electricity usage patterns into the same group; thus the periods of high power consumption of different users are less likely to overlap, making the peak-minimizing scheduler easy to schedule jobs to achieve low peak demand. E.g., under our minimum-variance grouping, a household that usually has high power usage between 3pm and 7pm may be grouped with another household that usually has high power usage between 7pm and 11pm; then a peak-minimizing scheduler can easily achieve a big reduction in peak power consumption. In contrast, under random grouping, two households with the same peak usage hour may be placed in the same group; a peak-minimizing scheduler cannot achieve a low peak demand since many jobs of the two households need to be scheduled within the same hour – there is little room to shift jobs within this peak hour even if some other time periods have no jobs. However, this is unlikely to happen under our minimum-variance grouping since grouping two households with the same peak period will result in very high variance.

6) EDF Scheduling vs. No Scheduling: In this section, we evaluate how much gain the EDF scheduling (i.e., Algorithm 1) can achieve, compared to no scheduling. Here, no scheduling means that the users can run the jobs at will (without any coordination), i.e., each job J_i is randomly assigned with a starting time s_i, which is uniformly distributed in $[t_i^e, t_i^l]$. We study three cases.

- Case 1: $t_i^e = 0$ and $t_i^l = T - \tau_i$ ($\forall i$). We choose $T = 24$ hours. Case 1 represents the most flexible constraint, that is, each job has the maximum flexibility/range, i.e., $[0, T - \tau_i]$, for scheduling.
- Case 2: $t_i^l - t_i^e \leq 2$ hours ($\forall i$). Case 2 represents a more stringent constraint, i.e., each job must be scheduled within an interval of two hours.
- Case 3: $0 < t_i^l - t_i^e \leq 2$ hours (for deferrable job J_i) and $t_j^l - t_j^e = 0$ (for non-deferrable job J_j). Case 3 represents a mixture of deferrable jobs and non-deferrable jobs; e.g., a refrigerator is a non-deferrable appliance. Based on the measurement data in six houses [9], [13], on average, 89.76% power is consumed by deferrable jobs and the remaining power is consumed by non-deferrable jobs. Hence, our simulator generates deferrable jobs, which consume 89.76% of the total power on average, and non-deferrable jobs, which consume 10.24% of the total power on average.

For Case 1, we observe that the PAR values for the EDF scheduling and no scheduling are 1.34 and 1.99, respectively; hence the EDF scheduling reduces PAR by 32.67%, compared to no scheduling. For Case 2, we observe that the PAR values for the EDF scheduling and no scheduling are 1.43 and 1.59, respectively; hence the EDF scheduling reduces PAR by 10.06%, compared to no scheduling. For Case 3, we observe that the PAR values for the EDF scheduling and no scheduling are 1.45 and 1.60, respectively; hence the EDF scheduling reduces PAR by 9.37%, compared to no scheduling. It can be observed that the smaller interval of $[t_i^e, t_i^l]$, the less PAR reduction. This is because the smaller interval of $[t_i^e, t_i^l]$, the less room to shift jobs around under the EDF scheduling. For a non-deferrable job i, since $t_i^e = t_i^l$, our EDF scheduling algorithm is not allowed to shift the job and hence there will be no gain for non-deferrable jobs. The lower percentage of non-deferrable jobs in the total power consumption, the higher gain achieved by our EDF scheduling algorithm.

In summary, the simulation results demonstrate that our family plan approach achieves significant saving in user payments and fuel cost in power generation and a large reduction on peak power consumption.

VI. Related Work

Peak demand reduction by model predictive control (MPC) with real-time electricity pricing was studied in [14]. Some recent papers (*e.g.*, [15]) focus on designing online algorithms for using UPS units for cost reduction via 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 [16] or improving performance and availability [17].

The component of partitioning users into groups in our scheme is closely related to the graph clustering problem [18], for which various novel techniques (see [19] for a survey) have been proposed, including spectral clustering techniques [6]. These techniques often have high complexity, usually $O(|V_u|^3)$ where $|V_u|$ is the number of nodes in a graph. Our graph clustering scheme is distributed and has linear complexity $O(|E_u|)$. For a social network, which is a scale-free network, the number of edges $|E_u|$ is in the same order of the number of nodes $|V_u|$; hence our clustering scheme has a complexity of $O(|V_u|)$ for a social network. Therefore, it is scalable for a large social network or a large scale-free network. The reason why our clustering scheme has a linear complexity instead of $O(|V_u|^3)$ as in spectral clustering, is because our scheme does not use eigen-decomposition of the Laplacian matrix of a global topology/graph (which requires $O(|V_u|^3)$); instead, our scheme only uses local information and hence the overall complexity linearly depends on the number of edges.

Within each cluster, we schedule jobs to further reduce the PAR. Scheduling was extensively studied, including link scheduling in wireless networks [20], and parallel job scheduling [21]. Smart grid job scheduling has also been studied previously for various objectives: incentive [2], stochastic reliability [22], reducing single-unit PAR [3], and micro-grid [23]. Some problems closely related to job scheduling in smart grid studied here include rectangle packing [24] and Orthogonal Rectangular Strip Packing Problem (SPP) [25]. In these traditional packing problems, jobs are "rigid", thus, many holes exist in the packing solution. As power consumption is simple sum of the power consumed by all appliances, traditional techniques cannot be directly applied.

VII. Conclusions

In this paper, we proposed a family-plan approach for collaboratively reducing the peak power consumption. Our scheme leverages the social network structure among users and heterogeneous energy consumption patterns of households, to schedule the load among consumers in the service area of the same utility company. We designed a linear-time-complexity clustering algorithm for fully exploiting the heterogeneity of users' energy consumption patterns, and a sub-optimal load scheduling algorithm in each group to further reduce the PAR. Using synthetic data generated by our trace-driven simulator, our evaluations show that our proposed scheme significantly reduces the peak power as well as user payment and fuel cost of power generation. Last but not the least, our family-plan approach can be applied to an arbitrary topology although we only showed simulation results for a scale-free network of 1000 nodes in Section V. Our future direction is to consider scheduling of interruptible jobs.

References

[1] L. Perez-Lombard, J. Ortiz, and C. Pout, "A review on buildings energy consumption information," *Energy and buildings*, vol. 40, no. 3, pp. 394–398, 2008.

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

[3] S. Tang, Q. Huang, X.-y. Li, and D. Wu, "Smoothing the energy consumption: Peak demand reduction in smart grid," in *Proceedings of IEEE INFOCOM*, 2013, pp. 1133–1141.

[4] R. Jiang, J. Wang, and Y. Guan, "Robust unit commitment with wind power and pumped storage hydro," *IEEE Transactions on Power Systems*, vol. 27, no. 2, pp. 800–810, 2012.

[5] www.att.com/shop/wireless/plans/familyplans.html.

[6] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.

[7] L. Schenato and G. Gamba, "A distributed consensus protocol for clock synchronization in wireless sensor network," in *46th IEEE Conference on Decision and Control*, 2007, pp. 2289–2294.

[8] C. Gentry, "Computing arbitrary functions of encrypted data," *Communications of the ACM*, vol. 53, no. 3, pp. 97–105, 2010.

[9] http://redd.csail.mit.edu/.

[10] P. J. Brockwell and R. A. Davis, *Time series: theory and methods*. Springer, 1991.

[11] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Physical Review E*, vol. 78, no. 4, p. 046110, 2008.

[12] https://sites.google.com/site/andrealancichinetti/files.

[13] J. Z. Kolter and M. J. Johnson, "Redd: A public data set for energy disaggregation research," in *Proceedings of the SustKDD Workshop on Data Mining Applications in Sustainability*, 2011, pp. 1–6.

[14] S. Caron and G. Kesidis, "Incentive-based energy consumption scheduling algorithms for the smart grid," in *1st IEEE International Conference on Smart Grid Communications (SmartGridComm)*, 2010, pp. 391–396.

[15] Y. Guo, Z. Ding, Y. Fang, and D. Wu, "Cutting down electricity cost in internet data centers by using energy storage," in *IEEE Global Telecommunications Conference (GLOBECOM)*, 2011.

[16] Q. Zhu, F. M. David, C. F. Devaraj, Z. Li, Y. Zhou, and P. Cao, "Reducing energy consumption of disk storage using power-aware cache management," in *Proceedings of IEEE Software*, 2004, pp. 118–118.

[17] J. S. Chase, D. C. Anderson, P. N. Thakar, A. M. Vahdat, and R. P. Doyle, "Managing energy and server resources in hosting centers," in *ACM SIGOPS Operating Systems Review*, vol. 35, no. 5, 2001, pp. 103–116.

[18] S. E. Schaeffer, "Graph clustering," *Computer Science Review*, vol. 1, no. 1, pp. 27–64, 2007.

[19] R. Xu, D. Wunsch *et al.*, "Survey of clustering algorithms," *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 645–678, 2005.

[20] A. Nusairat and X. Li, "Wimax/ofdma burst scheduling algorithm to maximize scheduled data," *IEEE Transactions on Mobile Computing*, 2011.

[21] D. Ye and G. Zhang, "On-line scheduling of parallel jobs," in *Structural Information and Communication Complexity*. Springer, 2004, pp. 279–290.

[22] M. He, S. Murugesan, and J. Zhang, "Multiple timescale dispatch and scheduling for stochastic reliability in smart grids with wind generation integration," in *Proceedings of IEEE INFOCOM*, 2011, pp. 461–465.

[23] Y. Huang, S. Mao, and R. Nelms, "Adaptive electricity scheduling in microgrids," *Proceedings of IEEE INFOCOM*, 2013.

[24] K. Jansen and G. Zhang, "On rectangle packing: maximizing benefits," in *Proceedings of the 15th annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2004, pp. 204–213.

[25] B. S. Baker and J. S. Schwarz, "Shelf algorithms for two-dimensional packing problems," *SIAM Journal on Computing*, vol. 12, no. 3, pp. 508–525, 1983.