Negotiation-Based Task Scheduling to Minimize User's Electricity Bills under Dynamic Energy Prices

Ji Li, Yanzhi Wang, Tiansong Cui, Shahin Nazarian, and Massoud Pedram
Department of Electrical Engineering
University of Southern California, Los Angeles, CA, USA
{jli724, yanzhiwa, tcui, snazaria, pedram}@usc.edu

***Abstract*—Dynamic energy pricing is a promising technique in the Smart Grid that incentivizes energy consumers to consume electricity more prudently in order to minimize their electric bills meanwhile satisfying their energy requirements. This has become a particularly interesting problem with the introduction of residential photovoltaic (PV) power generation facilities. This paper addresses the problem of task scheduling of (a collection of) energy consumers with PV power generation facilities, in order to minimize the electricity bill. A general type of dynamic pricing scenario is assumed where the energy price is both time-of-use and total power consumption-dependent. A negotiation-based iterative approach has been proposed that is inspired by the state-of-the-art Field-Programmable Gate Array (FPGA) routing algorithms. More specifically, the negotiation-based algorithm is used to rip-up and re-schedule all tasks in each iteration, and the concept of congestion is effectively introduced to dynamically adjust the schedule of each task based on the historical scheduling results as well as the (historical) total power consumption in each time slot. Experimental results demonstrate that the proposed algorithm achieves up to 51.8% improvement in electric bill reduction compared with baseline methods.**

Keywords—Smart Grid; Dynamic Pricing; Algorithm; Optimization.

I. INTRODUCTION

With the advent of digital economy and information era, electric power industries are confronted with new challenges, including sustainable development, environmental protection, climate change, and the emergence of users' demands for better supply quality and higher reliability [1]. The intention of transferring to a low-carbon economy has driven a rapid growth of the weight of global electricity supplied by renewable energy in the 21st century [2]. The photovoltaic (PV) system, which converts solar radiation into electricity, is considered to be one of the most promising types of renewable energy technologies and has garnered global attention in recent years due to the growing energy demand and concerns over climate change [3]. The global operating capacity of PV energy generation is 40GW, 71GW and 100GW in 2010, 2011 and 2012, respectively [2]. Driven by falling prices, environmental concerns and government policies, industrial, commercial and residential consumers are increasingly becoming consumers of PV energy in a growing number of countries [2]. This necessitates the consideration of the optimal deployment and control of PV energy generations.

Besides the attempt to use renewable energy such as PV energy generation, the *smart grid* is another promising technology that has received global attention in both industry and academia to maintain a robust and resilient electricity system [4]. A smart grid is an intelligent power grid delivering electricity from suppliers to industrial, commercial and residential consumers with various sensing and data processing facilities as well as embedded control and decision-making protocols [5]. In a smart grid, electricity is generated by (renewable or non-renewable) power generation facilities, delivered through transmission lines and distributed by the local distribution infrastructures to end users. A major challenge is that the load profile of consumers varies dramatically as a function of time, season and other factors [6][7]. Meanwhile, the amount of generation, transmission and distribution capacities that utility companies need to provision depends on the peak power demand rather than the average, and the huge difference between energy consumption levels at peak usage hours and off-peak hours has resulted in not only cost inefficiencies and potential power delivery failures (brown-outs and black-outs), but also environmental pollution due to the over-provisioning of the Power Grid and the resulted energy waste [4][6]. Without a smart grid technique, the conventional power grid will require the U.S. government alone to invest hundreds of billions of dollars in new power plants over the next two decades to meet the worst-case electricity demand, resulting in a huge cost, energy waste and potential environmental problems [4].

Dynamic pricing, which is supported by the increasing deployment of new technologies of the smart meters on the consumer side [4], is an effective approach to improve the load management [4][6][8]. The main idea behind dynamic pricing is to make the electricity price dependent on users' real-time electricity usage, which enables users to shape their power demands to lower their monthly electric bills and at the same time make a contribution to the grid performance, reliability and economics [4]. The total gain achieved by applying dynamic energy pricing can be as high as 10% for the entire energy economy [8]. Effective implementation of dynamic energy pricing faces many challenges. The most difficult step is to predict energy users' reactions to various dynamic energy pricing policies, and several existing works have focused on this. The authors in [9] map this problem to the multiple knapsack problem which enables cheap and efficient solutions. An autonomous application scheduling scheme is proposed in [10] which predicts the load utilization based on history patterns. In [8], a task scheduling problem is formulated with an objective function to minimize electricity bill of cooperative

users under dynamic energy prices, assuming a certain task cannot be scheduled outside a desired time period. The authors in [5] introduce an inconvenience cost to those tasks scheduled outside the desired time slots to make the problem more realistic and treat the instantaneous power consumption limit as a hard constraint. This mandatory upper bound of power consumption in [5] is essential to shave the peak loads under the grid's limit and avoid blackouts or brownouts. However, it is not realistic for utility companies to apply a hard maximum power constraint to the users, and the branch and bound algorithm presented in [5] may not have a solution mainly because each iteration only rips up a certain part of the tasks and the remaining unchanged tasks are blocking a lot of possible solutions. Moreover, neither of the previous works [5][8][9][10] considers the special characteristics of off-grid PV energy, which is an important part of today's energy sources.

In this paper, our approach to minimize users' electricity bills under dynamic prices takes off-grid residential PV energy generation into consideration and uses an iterative approach that is inspired by the Triptych Field-Programmable Gate Array (FPGA) routing software developed in [11][12]. The negotiation-based task scheduling algorithm in this work differs from the previous works in several aspects including the formulation of rip-up and retry as well as the introduction of off-grid PV energy generation. The concept of congestion is also introduced in the proposed algorithm which dynamically adjusts the congestion degree based on the historical scheduling results as well as the total power consumption in each time slot. Effective PV prediction algorithms can be applied to predict the PV power generation profile ahead of time, however, to focus on the task scheduling instead of predicting the PV power, we assume part of the user's energy is supplied by the off-grid PV system and we use PV power generation profiles measured at (i) Duffield, VA, measured in the year 2007, and (ii) Los Angeles, CA, measured in the year 2012 [15]. To the best of our knowledge, this is the first work that introduces off-grid PV energy and congestion cost into the demand-side task scheduling problem under dynamic pricing.

The remaining of the paper is organized as follows. Section II formally presents the definition of the terminologies, system model and problem statement. Section III shows our negotiation-based task scheduling algorithm. Section IV reports the experimental results, and the paper is concluded in Section V.

II. System Model and Cost Functions

In this paper, we consider a single *user* who pays a unified electricity bill that covers the total energy consumed by a group of tenants in research laboratories, households, office spaces, factories, warehouses, etc. A negotiation-based task scheduling algorithm is developed to minimize the user's electricity bills determined by a cost function comprised of a time-dependent price, a power-dependent electricity price and an inconvenience cost. Without loss of generality, we focus on the task scheduling problem in one day in this paper and our approach can be easily extended to a longer period such as week, month, season and year.

We adopt a *slotted time* model, i.e., all system cost parameters and constraints as well as scheduling decisions are provided for discrete time intervals of constant length. In this model, the entire task scheduling period is divided into a fixed number of equal-sized time slots and the time resolution τ is defined as the largest size of time slot that all timing properties of each task can slot into. For example, if every task starts at the beginning of an hour, lasts integer multiples of half hours with deadline at the end of an hour, and the price function updates every 20 minutes, the time resolution τ should be 10 minutes. Each day is thus divided into T time slots and we use $T= 24$ and $\tau = 60$ minutes in our experiments.

For the user of interest, we assume that there are a number of tasks that should be executed daily. These tasks are identified by index i. The set of task indices is $\mathcal{T} = \{1, \dots, N\}$ where N is the total number of tasks for the energy user. Each task i has an earliest start time S_i, a deadline E_i and a fixed operation length D_i to complete the task. We denote the power consumption of task i in time slot t by $p_i(t)$ as t ranges from 1 to T and i ranges from 1 to N. We assume the tasks are non-interruptible, i.e., each task has to operate in a number of consecutive time slots. After the scheduling, task i actually starts at time λ_i and completes at $\lambda_i + D_i$. It is preferable to start task i no earlier than S_i and complete it no later than E_i. In the previous papers such as [8], the earliest start time and deadline are hard constraints of a task. However in our model, we make the scheduling more realistic by allowing task i to be scheduled outside the preferable time window $[S_i, E_i]$ but with an incurred *inconvenience cost* I_i which is determined by the user. The inconvenience cost represents the penalty to schedule task i outside the preferable time window $[S_i, E_i]$. Besides, each task consumes electricity according to a known power dissipation profile, i.e., the power consumption of task i will follow a known profile regardless of the start time λ_i. The timing specifications, inconvenience cost and power profile for all tasks are assumed to be provided by the user before the beginning of the day and keeps unchanged during the day.

Figure 1 gives an example of household task scheduling solution. Each bar represents a task which occupies a number of time slots. Any task that crosses the midnight can be divided into two equivalent parts, e.g. the two periods of air conditioner represents the entire air conditioner period crossing the midnight from 3pm to 6am.

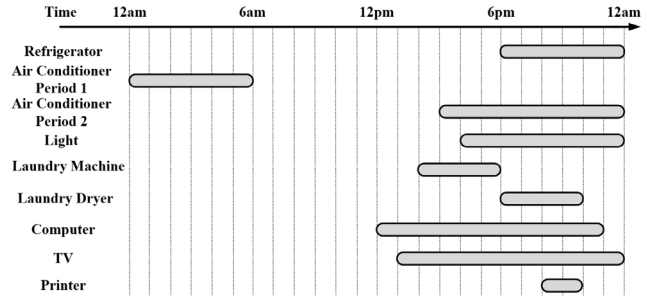

Fig. 1. Example of a household task scheduling problem.

We use a combined price model $\xi(t,\ \omega(t))$ comprised of a time-of-use (TOU) price that is determined by the time slot t, and a power-dependent price that depends on $\omega(t)$ which represents instantaneous power consumption drawn from the grid in time slot t, i.e., the *grid power*. The price function

$\xi(t, \omega(t))$ is assumed to be monotonically increasing with respect to $\omega(t)$, which means the price will rise if more power is consumed in a certain time slot t. Therefore, this combined price model incentivizes the user to shift loads from the peak hours to off-peak periods. However, going too far is as bad as not going far enough. If most users shift their tasks in the same way, the power plant might fail to match the load during some time periods which are originally considered as off-peak time. Considering this, we assume the utility company also sets a maximum power constraint in each time slot for the user so as to avoid potential outage. This maximum power constraint is denoted by $\beta(t)$ for time slot t and it is considered as a *soft* upper bound in the proposed algorithm because it is not practical for the utility company to set a hard maximum power constraint in reality. The detailed meaning of the soft bound is that the energy price will rise dramatically when the total power consumption in a time slot t exceeds $\beta(t)$, and the user can violate this bound but with an additional cost. By applying this soft bound, the chance to see an outage is much lower.

As stated in the introduction, a part of the energy is generated by the off-grid residential PV system. The PV power generation available to the user in time slot t is denoted by $pv(t)$. We assume that the user is not equipped with energy storage equipment, and therefore, the unused PV energy cannot be stored and is wasted. The investment and maintenance of the PV system is not in the scope of this work and we assume this part of cost is covered by the user, and thus, the PV power generation can be considered to be free to the user. A wise strategy is to utilize generated PV power as much as possible. After introducing the off-grid PV power, the grid power $\omega(t)$ consumed by the user in time slot t is calculated by:

$$\omega(t) = \begin{cases} 0, if\ pv(t) \geq \sum_{i=1}^{N} p_i(t) \\ \sum_{i=1}^{N} p_i(t) - pv(t), if\ pv(t) < \sum_{i=1}^{N} p_i(t) \end{cases} \quad (1)$$

Using the above definitions, the energy user's cost minimization problem can be modeled as follows:

Cost Minimization Problem for an Energy User.

Find the optimal start time λ_i for $1 \leq i \leq N$.

Minimize:

$$Total\ Cost = \sum_{t=1}^{T} \xi\big(t, \omega(t)\big) \cdot \omega(t) + \sum_{i=1}^{N} I_i(\lambda_i) \quad (2)$$

where the inconvenience cost for task i is given as

$$I_i(\lambda_i) = \begin{cases} 0, & S_i \leq \lambda_i \leq E_i - D_i \\ I_i, & otherwise \end{cases} \quad (3)$$

Subject to (Soft Bound[1]):

$$\omega(t) \leq \beta(t) \quad (4)$$

The available PV energy $pv(t)$ in time slot t is efficiently utilized when the following condition is satisfied:

$$\sum_{i=1}^{N} p_i(t) \geq pv(t) \quad (5)$$

[1]The scheduler is encouraged to satisfy the soft bound in all time slots. However, if there is no such solution, the scheduler can violate this soft bound and the penalty in energy cost is already captured in the objective function (2).

III. Negotiation-Based Energy Cost Minimization

In this section, a negotiation-based approach inspired by an adaptive routing algorithm in commercial FPGAs [11][12] is developed to find a suitable solution to the cost minimization problem (task scheduling problem) described in Section II. This section is organized as follows: part A provides a general description of the congestion-based routing approach in FPGAs; part B explains the motivation to introduce the concept of negotiation-based routing method to the task scheduling problem; part C presents the proposed negotiation-based task scheduling algorithm.

A. Negotiation-Based Routing Algorithm in FPGAs

Routing is an important step of the *Computer Aided Design* (CAD) process that is necessary to implement a circuit in FPGA [13]. The goal of a routing algorithm is to find a feasible solution which connects input signals of digital logic elements to output signals according to the gate-level logic descriptions using the limited routing resources such as physical wires and switches. On top of that, an effective routing algorithm tries to reduce the length of the longest path (i.e., the critical path) that determines the performance of the circuit. As explained in [11], the main constraint in the routing problem is that different signals cannot share the same routing resources, e.g. two signals cannot share the same physical wire in an FPGA.

In reference work [11], *congestion* in a routing problem indicates sharing of routing resources. As stated above, a routing solution must resolve congestion in every routing resource node. In [11], a *signal router* that only considers performance chooses the shortest path for each signal, which generates the fastest circuit but leads to a significant amount of congestions. Thus, a *congestion cost function* is introduced to each routing resource node and a *global router* guides the signal router to avoid those resources with high congestion cost. A solution is found when there is no congestion in the circuit.

Besides the introduction of the concept of congestion, the algorithm in [11] is implemented in an iterative manner. In each iteration, every signal is ripped-up and re-routed based on the latest congestion cost that is updated at the end of the previous iteration. To avoid the situation where a resource node is always shared, a history term is introduced to the congestion cost formula of every resource node. The effect of the history term is to permanently increase the cost of using congested nodes so that routes through other nodes are attempted [11]. In some sense, the rationality behind this rip-up and re-route scheme is to guide the signal router to look at a larger scope of possible scheduling paths because a fixed signal may block a large number of potential routing solutions. The negotiation-based routing algorithm provides the best solution known so far for the FPGA routing problem.

B. Motivation to Introduce Negotiation-Based Routing Method to Task Scheduling Problem

The cost minimization problem described in Section II, also known as the task scheduling problem, is an integer nonlinear programming problem subject to nonlinear constraints. In this kind of problems, no theoretically optimal results can be derived in polynomial time, nor are they likely to ever be available [14]. Using nonlinear optimizers to find the optimal solution in a task scheduling problem with a large set of

unknowns is impractical because of the unacceptable run time [5]. Therefore, instead of finding the optimal task scheduling solution, we seek for a good enough solution so that the implementation of the proposed algorithm is feasible with respect to computational complexity and the result is much better than the baseline approach.

Let us first analyze and conclude the similarities between the FPGA routing problem and the task scheduling problem:

- Finding the optimal solution incurs an extremely long run time, thus in both problems it is reasonable to find a suitable solution instead of the optimal one.
- Certain times of rip-up and retry are required to find a suitable solution, and hence, an iterative approach should be applied in both problems.
- There are limited resources in both problems with a cost in each resource node. The cost is delay and congestion in the FPGA routing problem, while the cost is price (increase) in the task scheduling problem.

Despite the similarities stated above, the task scheduling problem is quite different from the FPGA routing problem. Different tasks in the task scheduling problem can share a time slot even if the power consumption drawn from the grid (i.e., the grid power) exceeds the power limit in the time slot, whereas in the routing problem resources (physical wires and switches) cannot be shared by different signals. Besides the difference in resource sharing, these two problems have different objective functions. In the task scheduling problem, the goal is to schedule all tasks in $\mathcal{T} = \{1, \dots, N\}$ so that the total energy cost function (2) is minimized, whereas the objective function in the routing problem is to minimize the critical path delay represented by the sum of delay values of all the routing resources in that path. Moreover, the price function $\xi(t,\ \omega(t))$ in the cost minimization problem is a function of time slot t and instantaneous grid power consumption $\omega(t)$, and hence the price function of a time slot is more complex than the delay function of a routing resource. Based on the above differences, we cannot directly map the FPGA routing problem into to the cost minimization problem. However, we apply the concept of congestion used in the routing problem to help solve the cost minimization problem.

C. Negotiation-Based Task Scheduling Algorithm

A *Negotiation-Based Task Scheduling* (NBTS) algorithm, which is inspired by the congestion-based FPGA routing algorithm in [11], is proposed to find a suitable solution with a low computational complexity.

At the beginning of the day, the user provides the properties of each task i, namely, S_i, E_i, D_i, I_i and its power profile, and the utility company provides the price function and power constraint in each time slot t. The measured PV power generation $pv(t)$ is also given for each time slot t [15]. These are the inputs to the problem. The set of scheduled start times of all tasks in $\mathcal{T}$ is denoted by $\Lambda = \{\lambda_1, \dots, \lambda_N\}$. This is the output of the proposed algorithm. In the context of our problem formulation, a suitable solution is described as a schedule Λ in which most time slots utilize the PV energy efficiently and no slots or only a few slots exceed the power constraint set by the utility company. As mentioned in Section II, an iterative approach is applied in the proposed algorithm. We set K as the maximum iteration times.

At the beginning of the i-th iteration, all the tasks are ripped-up, and then we schedule task 1 to task N one by one. To introduce the iterative method, let $\omega_i^j(t)$ denote the total grid power consumed in time slot t after the j-th task has been scheduled in the i-th iteration where $t \in [1, T]$, $j \in [1, N]$ and $i \in [1, K]$. In the remaining subsection, we will introduce the concept of congestion during the discussion of each step in the proposed algorithm.

First, we analyze the 1st iteration and introduce the concept of *intra-iteration congestion* (to be precisely defined later). Before we schedule the 1st task, $\omega_1^1(t)$ is zero in all slots since no task is scheduled. We find the best possible start time λ_1 to schedule task 1 so that the overall energy cost $\sum_{t=1}^{T} \xi\left(t, \omega_1^1(t)\right) \cdot \omega_1^1(t) + I_1(\lambda_1)$ is minimized. Then the final grid power profile $\omega_1^1(t)$ for $t \in [1, T]$ is calculated based on the power profile of task 1.

The time slots occupied by task 1 are more likely to see a higher energy price since the price function in each slot t is monotonically increasing with respect to $\omega(t)$. Therefore, it is reasonable to lower the priority of these slots in the following steps in this iteration. We define a *congested time slot* as the time slot that is occupied by at least one task within the iteration. To quantify the congestion degree of a time slot, we define the *intra-iteration congestion term* $R(t)$ as the number of tasks that have been scheduled to occupy time slot t in one iteration. Before task 1 is scheduled, $R(t)$ is 0 in all slots and after task 1 has been scheduled, $R(t)$ becomes 1 in those slots chosen by task 1.

Before we schedule the j-th task in the 1st iteration, the total grid power in each slot t is $\omega_1^{j-1}(t)$, the price in each slot is $\xi(t, \omega_1^{j-1}(t))$, and the value of $R(t)$ is in the range of 0 to $j-1$. We will find the best start time of the j-th task to minimize the *congestion cost* of the j-th task. The congestion cost is defined as follows:

$$CC = \Sigma_{t=1}^{T} \left(\xi\left(t, \omega_1^j(t)\right) \cdot \omega_1^j(t) - \xi\left(t, \omega_1^{j-1}(t)\right) \cdot \omega_1^{j-1}(t) \right) \cdot (A_0 R(t) + 1) + I_j(\lambda_j) \quad (6)$$

where $\Sigma_{t=1}^{T} \left(\xi\left(t, \omega_1^j(t)\right) \cdot \omega_1^j(t) - \xi\left(t, \omega_1^{j-1}(t)\right) \cdot \omega_1^{j-1}(t) \right) + I_j(\lambda_j)$ is the *cost increase* after scheduling task j in the 1st iteration, and the term $A_0 R(t)$ (A_0 is a positive weight coefficient) is added to guide the scheduler to avoid congested slots. The λ_j value that minimizes (6) is chosen as the start time for task j, and then the final value of $\omega_1^j(t)$ for each time slot t is updated based on the chosen λ_j.

At the end of the 1st iteration, we obtain a schedule Λ that consists of λ_j for $j \in [1, N]$. When the power constraint is tight, it is possible that the total energy cost is not minimized and some time slots fail to use PV energy efficiently even though the term $R(t)$ has been introduced to avoid congestion within the 1st iteration. Thus, we need more iterations to achieve a lower total energy cost. In the subsequent iterations, it is reasonable to lower the priority of those congested slots that

result in high energy cost in the 1st iteration. Meanwhile, it is also reasonable to increase the priority of the time slots in which PV energy generation has not been utilized efficiently in the 1st iteration.

To guide the scheduler to achieve these two goals, we introduce two *inter-iteration congestion terms*. The 1st term is denoted by $H_1(t)$, which represents the total times in the previous iterations when the power constraint has been exceeded in time slot t. The 2nd term is denoted by $H_2(t)$ that is the total times in the previous iterations when PV energy generation is not fully utilized in time slot t. The inter-iteration congestion terms are updated at the end of each iteration (whereas the intra-iteration term is updated within one iteration). After introducing the two inter-iteration congestion terms, we integrate the three congestion terms into

$$C_{slot}(t) = A_0 R(t) + A_1 H_1(t) - A_2 H_2(t) + 1 \quad (7)$$

where A_1 and A_2 are positive weights of $H_1(t)$ and $H_2(t)$, respectively[2]. Thus, when the scheduler is scheduling the j-th task to start execution at time λ_j in the i-th iteration, the congestion cost CC' of the j-th task is then redefined by

$$CC' = \Sigma_{t=1}^{T} \left(\xi\left(t, \omega_i^j(t)\right) \cdot \omega_i^j(t) - \xi\left(t, \omega_i^{j-1}(t)\right) \cdot \omega_i^{j-1}(t) \right) \cdot C_{slot}(t) + I_j(\lambda_j) \quad (8)$$

In (7) and (8), the term $A_1 H_1(t)$ guides the scheduler to lower the priority of the time slots which violate the power constraint in the previous iterations. Meanwhile, the term $A_2 H_2(t)$ slightly lowers the cost of the time slots in which PV energy generation are not used efficiently in previous iterations. The λ_j value that minimizes the objective function (8) is chosen as the start time for task j. After that, we update the final value of $\omega_i^j(t)$ for $t \in [1, T]$ based on the chosen λ_j before scheduling the next task.

To solve the problem that a number of suitable solutions are blocked when a task always occupies the same time slots, we introduce the 3rd inter-iteration congestion term $h(j,t)$ which indicates the number of times in the previous iterations when task j occupies time slot t. Initially, $h(j,t)$ is 0 for all tasks in all time slots. This term permanently increases if this task j occupies the time slot t in one iteration, and after several iterations, the task may give up those time slots due to the increasing congestion cost. Of note, the difference between $h(j,t)$ and the above-mentioned three congestion terms is that $h(j,t)$ is introduced to each task and each time slot while the other three terms are integrated to each time slot.

When the scheduler is scheduling the task j in the i-th iteration and it tries to set λ_j as the start time, the final congestion cost CC'' is recalculated by

$$CC'' = \Sigma_{t=1}^{T} \left(\xi\left(t, \omega_i^j(t)\right) \cdot \omega_i^j(t) - \xi\left(t, \omega_i^{j-1}(t)\right) \cdot \omega_i^{j-1}(t) \right) \cdot (ah(j,t) + 1) \cdot C_{slot}(t) + I_j(\lambda_j) \quad (9)$$

where a is the weight of $h(j,t)$. The λ_j value that minimizes (9) is chosen, and we update the final value of $\omega_i^j(t)$ for $t \in [1, T]$ based on the chosen start time before scheduling the next task.

[2]The proposed algorithm can be applied to a user without off-grid PV energy supply by setting $A_2 = 0$.

The proposed algorithm terminates when the total energy cost calculated by (2) is not decreased in L consecutive iterations or the K-th iteration is completed. The computational complexity of the proposed algorithm is $O(KNT^2)$. Figure 2 shows the pseudo code for the proposed algorithm.

Algorithm *Negociation-Based Task Scheduling* ()
// Schedule tasks to minimize users' eletricity bills
Initialize tasks, PV profile, price functions, inconvenience cost functions, constraints and congestion terms
Set iteration counter $i = 0$
Do
  $i = i + 1$
  Rip up all tasks
  Loop over all tasks j
    Loop over all time slots $t \in [1, T - D_j + 1]$
      Set the start time $\lambda_j = t$
    Choose the λ_j that minimizes (9)
    Update the intra-iteration congestion term $R(t)$
  Loop over all time slots t
    Update $H_1(t)$ and $H_2(t)$
    Loop over all tasks j
      Update $h(j,t)$
  Calculate the overall energy cost by (2)
Until total energy cost is not decreased for L iterations or $i > K$
Return Λ

Fig. 2. The proposed negociation-based task scheduling algorithm.

IV. Simulation Results

In this section, to demonstrate the effectiveness of the proposed NBTS algorithm, cases corresponding to the abovementioned pricing models are examined. As mentioned above, the duration of a time slot is set to one hour, and we examine the task scheduling problem in one day. For this reason, the duration of a task is integer multiples of one hour and it cannot exceed 24 hours. Moreover, power consumptions of the tasks are determined with a granularity of one hour.

We assume the input data from the user are given at the beginning of the day and remain unchanged during the day. The preferable start time, end time, duration and inconvenience cost are generated arbitrarily for each task. The power profile of each task is randomly generated based on power consumption data about real appliances. Besides, PV profile of that day is predicted using effective PV power generation prediction algorithms. We examine a household task scheduling case. The user owns a 6,000-watt (6 kW) PV system that can provide around 6 kilowatts of electricity per hour under optimum conditions. According to Phillips, the Florida Solar Energy Center has determined there are 1,489 optimum solar generating hours per year, accompanied with hours of less than optimum generation [16]. Therefore, the 6 kW PV system has around 4 hours per day on average under optimum generation conditions.

We also assume the input data from the utility company are available at the start of the day and keep unchanged through the day. The utility company will set the TOU-dependent price and the power limit for each time slot. The price function presented in the task scheduling problem is assumed to be monotonically increasing. The part of the power consumption that exceeds the power limit will incur twice higher energy prices in our experiments.

The baseline is implemented using the greedy algorithm that schedules each task to start at the best possible time slot based on the TOU-dependent price component of the energy pricing model and the total power consumption of all previously scheduled tasks. The baseline algorithm will check whether the power constraint is satisfied in each time slot and it will reschedule from the 1st task that has no possible start time to satisfy the power constraints in all time slots. For example, in a task scheduling problem with 5 tasks, if tasks 1, 2 and 3 have been scheduled and task 4 cannot be scheduled to make sure that the power constraint is satisfied in every time slot, in the next iteration, task 4 will be scheduled with the highest priority. Of note, our baseline is not aware of the part of PV energy, and in order to make the comparison fair, the peak PV energy cannot cover more than 6 tasks in a time slot.

Table I shows the simulation results of the proposed NBTS algorithm compared to the baseline greedy optimization method when the power limit in each slot is set so that greedily scheduling each task to its best start time will violate the power constraint. The total task number N is set as a parameter of the program and we arbitrarily generate 10,000 cases for task number ranging from 5 to 50 and report the average performance in Table I. In order to test the effectiveness of the rip-up policy in both algorithms, we consider a task scheduling problem that has 25 tasks and arbitrarily generate 10,000 cases for different degrees of power constraint, i.e., medium, tight and very tight, and define a figure-of-merit called *error rate* that is the number of cases whose schedule violate power constraint in at least one time slot over the total number of the simulated cases. The results are shown in the Table II.

TABLE I. PERFORMANCE AND TIME COMPLEXITY OF THE NEGOCIATION-BASED TASK SCHEDULING ALGORITHM AND THE BASELINE

Task Number	***Average total price of baseline (cents)***	***Average price of NBTS (cents)***	***Average performance increase factor***	***Average time complexity of NBTS (s)***
5	103.94	68.46	1.518	0.004
10	227.43	154.69	1.470	0.010
15	372.71	260.44	1.430	0.018
20	530.19	391.91	1.352	0.029
25	726.54	560.11	1.297	0.041
30	954.16	746.20	1.278	0.060
35	1161.51	955.03	1.216	0.086
40	1403.74	1171.31	1.198	0.115
45	1657.40	1417.57	1.169	0.108
50	1931.87	1649.42	1.171	0.153

TABLE II. PERFORMANCE AND AVERAGE ERROR RATE OF THE NEGOCIATION-BASED TASK SCHEDULING ALGORITHM AND THE BASELINE

Constraint Intensity	***Average performance increase factor***	***Average Error Rate of Baseline***	***Average Error Rate of NBTS***
Medium	1.297	11.40%	0.76%
Tight	1.389	27.29%	1.59%
Very Tight	1.593	61.04%	5.77%

It can be seen from Table I that the proposed algorithm can achieve 21.6%-51.8% improvement when N is below 40 and more than 15% improvement when N is from 40 to 50. Meanwhile, the simulation time of the NBTS is not large. Moreover, the proposed algorithm is effective when the power constraint in each time slot is tighter, and it achieves less than 6% error rate under the very tight constraint condition while the greedy baseline has 61.04% error rate.

V. CONCLUSION

The overarching goal of this paper was to develop an effective algorithm to minimize the user's electric bills under dynamic energy prices. The concept of congestion and off-grid PV power generation was introduced to the proposed negotiation-based task scheduling algorithm, and the proposed algorithm was implemented and tested for different test schemes. The results demonstrated the ability of the proposed algorithm for up to 51.8% energy cost saving compared to a greedy baseline method.

ACKNOWLEDGEMENT

This work is sponsored in part by a grant from the National Science Foundation.

REFERENCES

[1] Q. Li and M. Zhou, "The future-oriented grid-smart grid," J. Comput., vol. 6, no. 1, pp. 98–105, 2011.

[2] Renewable 2013 Global Status Report (GSR), http://www.ren21.net/ren21activities/globalstatusreport.aspx

[3] A. Otaibi and S. Jandal, "Solar Photovoltaic Power in the State of Kuwait," IEEE Photovoltaic Specialists Conference (PVSC), 2011.

[4] U.S. Department of Energy, Grid Energy Storage, Dec. 2013.

[5] H. Goudarzi, S. Hatami, and M. Pedram, "Demand-side load scheduling incentivized by dynamic energy prices," Proc. 2nd Int'l Conf. on Smart Grid Communications, Oct. 2011.

[6] T. Cui, Y. Wang, H. Goudarzi, S. Nazarian, and M. Pedram, "Profit Maximization for Utility Companies in an Oligopolistic Energy Market with Dynamic Prices," IEEE Online Conference on Green Communications, 2012.

[7] L. Yao, W. C. Chang, and R. L. Yen, "An iterative deepening geneticalgorithm for scheduling of direct load control,"IEEE Trans. Power Syst., vol. 20, no. 3, pp. 1414–1421, Aug. 2005.

[8] S. Hatami and M. Pedram, "Minimizing the Electricity Bill of Cooperative Users under a Quasi-Dynamic energy pricing Model," Proc. Smart Grid Communications Conf., 2010.

[9] K. Nallaperumal, S. Subramanian, and S. Sapatnekar, "Residential task scheduling under dynamic pricing using the multiple knapsack method," in Proc. of ISGT, 2012.

[10] C. Adika and L. Wang, "Autonomous appliance scheduling for household energy management", IEEE Trans. on Smart Grid, vol. 5, no. 2, pp. 673-682, Mar. 2014.

[11] C. Ebeling, L. McMurchie, S. Hauck, and S. Burns, "Placement and routing tools for the triptych FPGA," IEEE Trans. on VLSI Systems, vol. 3, no. 4, pp. 473-482, Dec. 1995.

[12] G. Borriello, C. Ebeling, S. Hauck and S. Burns, "The Triptych FPGA Architecture," IEEE Trans. on VLSI Systems, vol. 3, no. 4, pp. 491-501, Dec. 1995.

[13] D. Gomez-Prado and M. Ciesielski, "A Tutorial on FPGA Routing," http://www-unix.ecs.umass.edu/~dgomezpr/Research/Routing.pdf

[14] Hemmecke, R., Köppe, M., Lee, J., Weismantel, R. : Nonlinear integer programming. In: Jünger, M.,Liebling, T., Naddef, D., Nemhauser, G., Pulleyblank, W., Reinelt, G., Rinaldi, G., Wolsey, L.A.(eds.) 50 Years of Integer Programming 1958–2008. The Early Years and State-of-the-Art Surveys, Springer, Berlin (2009).

[15] Y. Wang, X. Lin, and M. Pedram, "Adaptive Control for Energy Storage Systems in Households With Photovoltaic Modules," IEEE Trans. on Smart Grid, vol. 5, no. 2, pp. 992-1001, Mar. 2014.

[16] Common questions about photovoltaic systems answered, https://www.clayelectric.com/photovoltaicinfo.aspx