JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 28, 955-969 (2012)

Energy Consumption Scheduler for Demand Response Systems in the Smart Grid*

JUNGHOON LEE, HYE-JIN KIM, GYUNG-LEEN PARK AND MIKYUNG KANG+
Department of Computer Science and Statistics
Jeju National University
Jeju, 690-756 Korea
+*Information Sciences Institute*
University of Southern California
Arlington, VA22203, USA

This paper presents a design and evaluates the performance of a power consumption scheduler in smart grid homes or buildings, aiming at reducing the peak load in them as well as in the system-wide power transmission network. Following the task model consist of actuation time, operation length, deadline, and a consumption profile, the scheduler linearly copies the profile entry or maps a combinatory vector to the allocation table one by one according to the task type, which can be either preemptive or nonpreemptive. The proposed scheme expands the search space recursively to traverse all the feasible allocations for a task set. A pilot implementation of this scheduling method reduces the peak load by up to 23.1 % for the given task set. The execution time, basically approximated by $O(M^{N_{NP}}(3^{\frac{M}{2}})^{N_P})$, where M, N_{NP}, and N_P are the number of time slots, nonpreemptive tasks, and preemptive tasks, respectively, is reduced almost to 2% taking advantage of an efficient constraint processing mechanism which prunes a search branch when the partial peak value already exceeds the current best. In addition, local peak reduction brings global peak reduction by up to 16% for the home-scale scheduling units without any global coordination, avoiding uncontrollable peak resonance.

Keywords: smart grid, demand-side management, energy consumption scheduler, peak load reduction, execution time

1. INTRODUCTION

The modern power system is evolving to a highly interconnected, complex, and interactive network of power distribution facilities, telecommunication, the Internet, and electronic consumer devices [1]. Called the smart grid, this power network is basically built on top of two-way digital communication to fully take advantage of information technologies in delivering electricity from suppliers to consumers. The information technology can support interoperability between diverse and complex objects in the power system using the semantic web, web services, intelligent agents, and real-time communication protocols. The smart grid allows customers to smartly consume electricity both by selecting a preferred supplier and scheduling the operating of each appliance according to the various conditions including the price change and current load. Accordingly, demand-side management is becoming more important to meet the customer requirement as well as to achieve the system goals such as peak load reduction, power cost saving,

Received May 31, 2011; accepted March 31, 2012.
Communicated by Jiman Hong, Junyoung Heo and Tei-Wei Kuo.
* This research was supported by the MKE (The Ministry of Knowledge Economy), Republic of Korea, under IT/SW Creative research program supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2012-(H0502-12-1002)).

energy efficiency, and so on.

The balancing of energy utilization is one of the most critical factors for the efficient operation of an electrical system [2]. Moreover, peak load conditions take place when the simultaneous requests from many users are concentrated on a specific time interval. Moreover, as PHEVs (Plug-in Hybrid Electric Vehicles) are coming to the market, they will double the household load during charging time. This concentration can possibly lead to a serious problem such as the disruption of power provision and temporal power shortage. To cope with this situation, the power system can employ a load-dependent pricing policy. For example, peak load pricing raises prices when the demand for a service is at its highest [3]. In addition, in IBR (Inclining Block Rates) pricing, the marginal price increases by the total quantity consumed [4]. Here, beyond a certain threshold in the total residential load, the electricity price will increase to a higher value. Such a policy creates incentives for end users to distribute their load at different times of the day to avoid higher electricity rates. Above policies expect a consumer-side reaction according to the high price to shift the demand away from peak load hours. On the customers' side, they need more efficient energy consumption scheduling, or interchangeably power scheduling.

Current smart grid technologies can control the power consumption in homes and buildings more intelligently and autonomously, home controllers playing a key role [5]. The operation of each appliance can be started, suspended, resumed, and stopped by a smart meter under the management of a home controller. As a networked appliance, the home controller basically coordinates the interaction among utility companies and home appliances, monitoring devices as shown in Fig. 1. First, the controller exchanges the information on price and demand with the utility company via the global connection such as the Internet or cellular networks. The utility company sends a residential load change to the consumer's home, activating load control, demand response, and price adjustment. Next, the controller interacts with home appliances mainly by a home area network such as Zigbee [6] or power line communication [7], triggering their operation and metering their power consumption. Finally, the controller sends a message either to the IHD (In-Home Display) device or a human user through the mobile phone or web portal to inform him or her of specific events and price changes. Through the global connection, TOC (Total Operation Center) can interact with the respective in-home controllers.

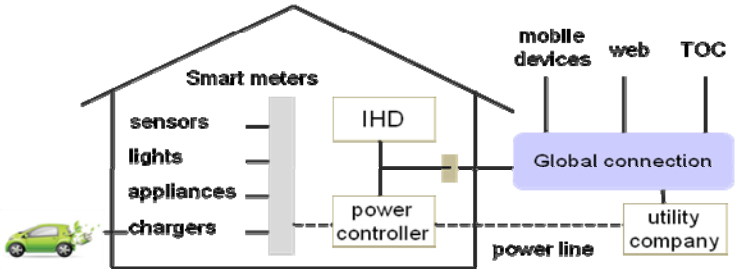

Fig. 1. Power scheduler operation.

Smart scheduling can save money by allowing reduction of energy output during peak demand time. The scheduling function can be performed in any computing devices

such as controllers, mobile devices, high-capacity computing servers, and the like, while their logic is implemented in the controller. The scheduling problem is quite similar to process scheduling in the real-time operating system [8]. Real-time process scheduling allows managing the execution of tasks on processors under the time constraints. Electric devices are also modeled as tasks having the execution time, the start time, and deadlines. The difference lies in that the electric device can run in parallel in energy scheduling as long as the total power does not exceed the current capacity of the transmission cables. The execution time is analogous to the power consumption, but the latter has more complex resource requirement. In addition, the power scheduler focuses on peak load reduction or energy cost saving rather than the deadline meet ratio or schedulablility.

The load concentration is a serious problem as mentioned previously. Moreover, the area-wide power shortage leads to the construction of more power plants. It's not desirable from the environmental perspective. The global coordination among a bunch of homes or buildings doesn't seem to be possible, and the effect of grouping some of them is quite restrictive. Hence, it is important to first reduce the peak load in each individual scheduling unit. To meet such requirement, this paper is to design a power scheduler capable of reducing peak load in homes or buildings. Scheduling is in most cases a very complex time-consuming problem greatly sensitive to the number of tasks. However, with a reasonable heuristic capable of reducing search space size and a practical power load profile, the backtracking-based scheduling scheme can efficiently find an optimal schedule in homes where the number of appliances is generally less than 10.

This paper is organized as follows: After issuing the problem in sections 1, section 2 describes the background of this paper and related work. Section 3 explains the target system configuration and designs a power scheduler for the smart grid homes. After performance measurement results are demonstrated and discussed in section 4, section 5 summarizes and concludes this paper with a brief introduction of future work.

2. RELATED WORK

In the smart grid, the role of information and automation technologies increases extensively to make the power distribution network smarter. An enterprise-level information system includes the broad technical areas in supervisory control & data acquisition, customer services, planning, trading, scheduling, power marketing, billing, accounting, and business management [9]. Most of services handle a large amount of data collected by the underlying sensor network [10] based on distributed intelligence as well as centralized analysis and control. Accordingly, the smart grid information system needs significant computing power, large data storage for GIS (Geographic Information System) and MIB (Management Information Base), along with an efficient data exchange protocol for AMI (Automated Metering Infrastructure) [11]. The home controller needs not a little computing capacity in scheduling even for the operation of just a few home appliances, as the schedule must consider current pricing policy, power requirement from each appliance, and home-generated or accumulated power availability.

Particularly, demand-side resources are managed to meet the available generation and grid's power delivery capabilities at any time [12]. Built on top of a home area network technology, such DSM (Demand-Side Management) is the core of H2G (Home-to-

Grid). Constant monitoring of power usage for each appliance can catch power leak due to stand-by consumption. Programming the control of light, heating, and air conditioning can achieve significant energy savings and meet comfort expectations [2]. Even though DSM cannot reduce the total energy consumption significantly, it can reshape the power load requirement by an efficient scheduling policy, possibly reducing peak load. It can not only avoid temporary employment of expensive dispatchable energy but also the construction of a new power generation facilities. For the integration of DSM, smart devices have a built-in programmable response and control strategy, whereby users are able to program the device performance and set optimal performance levels based on a variety of external parameters [1].

As an example of power consumption scheduling, MAHAS (Multi-Agent Home Automation System) can adapt power consumption to available power resources according to inhabitant comfort and cost criteria [13]. Based on the multi-agent architecture, the power management problem is divided into subproblems involving different agents, each of which tries to solve its own problem independently to find a solution of the whole problem. Particularly, the control algorithm is decomposed into reaction and anticipation mechanisms. While the first protects constraint violations, the second computes the plan for global consumption according to predicted productions and consumptions. The control function coordinates and negotiates the agent operations, sometimes eliminating or adding new agents. This scheme seems to be scalable, as it can reduce down the whole search space for the given optimization problem. However, it cannot guarantee obtaining the optimal solution or avoid the complex interaction between the agents.

[14] discusses a scheduling problem for household tasks to help users save money spent on their energy consumption. Assuming the situation the users at home can freely select or change an electricity supplier, its system model relies on electricity price signals, the availability of locally generated power, and flexible tasks with deadlines. Particularly, this work addresses a descriptive task model where tasks are either preemptive or non-preemptive, that is, tasks can be suspended or not during their operations. A case study shows that cost savings are possible, but fast and efficient solutions are still needed for the scheduler to be deployed in the real-world. This problem stems from the fact that they take relatively fine grained time slots as large as 20 minutes, not integrating any heuristic. So, the complexity of search space reaches $O(2^{MN})$, where N is the number of tasks and M is the number of time slots. According to the performance evaluation conducted in this work, it takes about 35 minutes to generate a schedule on an average performance personal computer, when N is just 7.

A. Mohesenian *et al.* has presented an autonomous and distributed demand-side energy management system among users taking advantage of a two-way digital communication infrastructure, which is commonly provided in the smart grid [4]. Based on the game theory, this scheme formulates an energy consumption scheduling game, where the players are the users and their strategies are the daily schedules of their appliances and loads. This scheme also employs the load profile model having 1 hour time granularity and makes it possible for the utility company to adopt adequate pricing tariffs that differentiate the energy usage in time and level. The global optimal performance in terms of minimizing the energy costs is achieved at the Nash equilibrium of the formulated energy consumption scheduling game. The authors assert that this scheme can reduce the PAR (Peak-Average Ratio), the energy cost, and each user's daily electricity charges. How-

ever, the coordination and consistency management of a group of unit schedulers not only makes the system more complex but also extends the scheduling time.

The SAHPS (Small Autonomous Hybrid Power System) contains renewable and conventional power sources [15]. Conventional generators produce on demand in economic way, and they can provide backup power when the renewable production is not sufficient. Even though the renewable energy source, combined with electric storage, does not emit during their operation, they may produce significant amount of pollutant emissions in their whole life cycle. In its design, economic and environmental criteria are two conflicting objectives. It proposes that the economic objective function should be system's cost of energy, while the environmental one is total CO_2 emissions. Specifically, non-determinant sorting genetic algorithm is combined with a local search procedure to solve the multi-objective optimization problem. In addition, it designs a hybrid PV-wind-diesel-hydrogen-battery installation for the generation of electric energy considering three conflicting objectives, namely, cost, pollution, and unmet load) based on evolutionary algorithm.

3. SCHEDULING SCHEME

The intelligent appliances and chargers can be controlled by the home controller in response to the distribution grid conditions and dynamic prices. The home power management system is able to make local decisions in controlling residential power consumption as well as to manage the demand through price signals [16]. The schedule can be generated and modified according to a task set change any time the customer wants. This section designs an energy consumption scheduler after the definition of our task model.

3.1 Task Model

Each task has its own power consumption characteristics, and this section categorizes them into 3 classes. *Class* 1 tasks must start immediately after the task gets ready and cannot be ceased to the end. For example, the hair drier is started at the moment a user wants. As the power consumption is fixed and there is no other option to adjust their operation, those tasks are not schedulable. Next, *class* 2 tasks don't have to start as soon as they get ready, but their operations are not preemptive. A dish washer or a laundry machine can start any time as long as the task can be completed within a specific deadline. Their operations must keep going to the end without suspension once they have started. Finally, *class* 3 tasks can also start after their activation time as class 2 tasks, but they are preemptive. The electric car charge belongs to this category. The task operation can be suspended and resumed within its deadline. This paper concentrates on schedulable tasks, namely, *class* 2 and *class* 3 tasks, and they will be denoted by nonpreemptive and preemptive tasks, respectively. Other complex tasks can be specified by the combination of the two.

The load power profile is practical for characterizing the power consumption behavior of each appliance. As pointed in [14, 17], the load power profile for a washing machine depends on the set program, its duration, and the water temperature chosen by a

user. Fig. 2 plots a sample power consumption profile for 5 tasks. Task 1 gets ready at time 1 and takes 7 time slots and so on. The length of a time slot can be tuned according to the system requirement on schedule granularity and computing time. In power schedule, the slot length can be tens of minutes, for example, 20 minutes. Tasks 1 through 4 cannot be preempted while Task 5 can. The power consumption for each electric device is aligned to the fixed-size time slot. Actually, each device has its own time scale in its power demand behavior. However, the automatic voltage regulating facility allows us to assume that every appliance has the same time scale in its power demand by selecting the peak or average power as the representative value of the slot. In addition, a smaller slot can generate a more accurate schedule, but there exists a trade-off between the complexity and accuracy. Fig. 2 also plots the per-slot total load and the peak is 10 at slot 7.

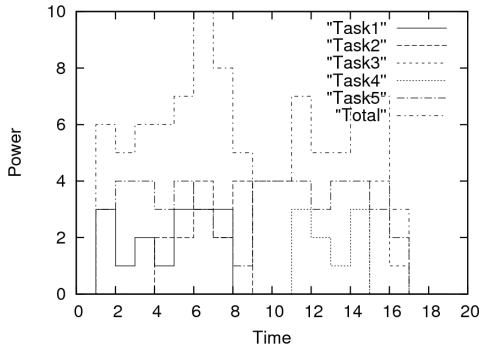

Fig. 2. Task profile and the earliest schedule.

Task T_i can be modeled with the tuple of $<F_i, A_i, D_i, U_i>$. First, F_i indicates whether T_i is preemptive or nonpreemtive. In addition, A_i is the activation time of T_i, D_i is the deadline, and U_i denotes the operation length, which corresponds to the length of the consumption profile entry. A nonpreemptive task can start from its activation time to the latest start time, namely, during the period of A_i through $D_i - U_i$. When a start time is selected, the profile entry is just copied to the allocation table one by one, as the task must not be preempted once it has started. The choice option is bounded by M, the number of time slots, hence the time complexity of search space traversal for a nonpreemptive task is $O(M)$. As contrast, the preemptive task case is quite complex. To meet its time constraint, U_i out of $(D_i - A_i)$ slots must be picked for device activation, and the number of nodes in the search space is equal to ${}_{(D_i-A_i)}C_{U_i}$. Hence, the worst case search space size of a preemptive task is ${}_{M}C_{M/2}$, while its complexity can be approximated as follows:

$$O((M\,|\,(M\,|\,1)|||(M/2+1))/(M/2\,|\,(M/2\,|\,1)|||1))\,|\,O((3/4M\,|\,3/4M||||3/4M)/(1/4M||||1/4M))$$
$$= O(3^{\uparrow}(M/2)) \quad (1)$$

The space bound for a preemptive task depends on the number of combinations, which can grow extensively for large M. Hence, its scheduling time is essentially sensitive to the space size of preemptive tasks [18].

3.2 Backtracking-based Scheduling

Scheduling is known to be a problem of $O(N!)$ complexity, where N is the number of tasks. However, in power scheduling, the number of processors is not a constraint as every appliance can work at the same time. Instead, it's quite similar to the knapsack problem with some specific constraint. Hence, our scheduling policy basically follows the backtracking algorithm which is one of the most convenient techniques for the knapsack problem and other combinational optimization problems. Backtracking incrementally builds a search tree. For the potential search tree that consists of all feasible solutions including invalid ones, the backtracking algorithm traverses recursively, from the root down, in depth-first order. At each intermediate node, which corresponds to a partial solution, it checks whether the node can lead to a valid solution. If it cannot, the remaining subtree is pruned. Otherwise, the recursion proceeds to the next level. Thus, the actual subtree that is traversed is only a part of the potential tree.

It finds all feasible solutions, each of which is an energy consumption schedule represented by an $M \times N$ time table as shown in Fig. 3. The number in a slot corresponds to the power requirement. The power unit is not explicitly specified in this paper, as it has a relative value decided by the smart grid type, for example, homes, buildings, farms, and sensor networks. For each row, a task operation schedule is allocated slot by slot. Hence, the sum of each column is the per-slot power requirement of a schedule. The maximum value is the peak load of the energy consumption schedule. Fig. 3 indicates that the peak load is reduced to 7 at the slots including 7, with appropriate scheduling for the task set given in Fig. 2.

Slot	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Task 1 (N)	0	3	1	2	1	3	3	3	0	0	0	0	0	0	0	0	0	0	0	0
Task 2 (N)	0	0	0	0	0	2	2	4	2	4	0	0	0	0	0	0	0	0	0	0
Task 3 (N)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	1	0	0	0
Task 4 (N)	0	0	0	0	0	0	0	0	0	0	0	3	2	1	3	0	0	0	0	0
Task 5 (P)	0	3	4	4	3	0	0	0	4	3	2	1	4	4	4	3	4	4	3	2
Total	0	6	5	6	4	5	5	7	6	7	2	4	6	5	7	7	5	4	3	2

Fig. 3. Slot allocation for power scheduling.

Fig. 4 illustrates the detailed scheduling algorithm. The allocation table consists of $M \times N$ fields. The allocation procedure fills the allocation table from the first row, each row being associated with a task. Basically, the procedure creates the search space for all feasible allocations. When the allocation procedure reaches a leaf, *EvalAlloc*() is called to check whether this allocation is better than current best in the maximum power consumption. If then, the current best is replaced. If not a leaf, the allocation proceeds. When T_i is nonpreemptive, for all possible start time between A_i to $(D_i - U_i)$, the allocation for $(i + 1)$th row is tried one by one recursively. Here, if the consumption profile for the task is (3, 4, 5, 2), the allocation for the start time 2 will be (0, 0, 3, 4, 5, 2, …) after copying the profile to the table. Oppositely, if T_i is preemptive, the profile is mapped to the table using the corresponding combination vector. For example, if the start time is 2 while the combination vector is (1, 1, 0, 0, 1, 1), the allocation will be (0, 0, 3, 4, 0, 0, 5, 6). Here, T_i will be suspended at time 4, and resumed at 6.

```
procedure AllocTab (i)
input : {T_i | F_i, A_i, D_i, U_i}
    if i equals to N
        EvalAllloc()
    end if
    if T_i is nonpreemtive
        for each start time from A_i to D_i − U_i
            copy the profile to ith row of the allocation table
            if (! CheckConstraint( )) AllocTab (i + 1)
    else
        for each combination of (D_i-A_i)CU_i
            map the profile to ith row of the allocation table
            if (! CheckConstraint())AllocTab (i + 1)
    end if
end procedure
```

Fig. 4. Power scheduling scheme.

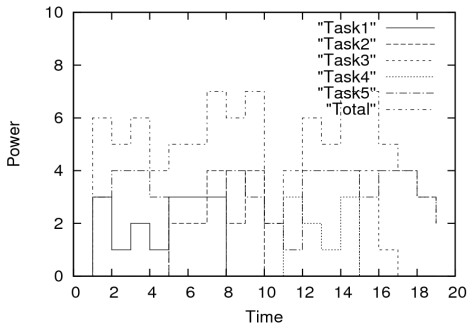

Fig. 5. Power reduction by an optimal schedule.

To map the profile entry of a preemptive task to the row of the allocation table, ${}_{(D_i-A_i)}C_{U_i}$ combination vectors are necessary. For example, when $D_i - A_i$ is 4 and U_i is 2, we have ${}_4C_2$ vectors, namely, (0, 0, 1, 1), (0, 1, 1, 0), (0, 1, 0, 1), (1, 0, 0, 1), (1, 0, 1, 0), and (1, 1, 0, 0). To speed up the allocation procedure, the feasible combinatory vectors are generated and stored in advance of search space expansion. Then, the scheduler maps the combination of each preemptive task to the allocation table one by one, checking the peak power requirement. In addition, *CheckConstraint*() can speed up the search procedure by pruning the unnecessary search tree expansion. If the maximum power requirement of the partial allocation for the tasks from T_0 to T_1 already exceeds the current best, it is needless to proceed to the remaining steps. After all, Fig. 5 shows the power schedule created by our scheme for the task set given in Fig. 2. The preemptive task T_5 has ${}_{19}C_{16}$ choices, and according to the allocation result, it will be started at time 1, suspended at 5, and resumed at 8. The maximum power requirement is 7.

4. EXPERIMENTAL RESULTS

This section implements a prototype of the proposed allocation method using Visual C++ 6.0 to evaluate its performance. Our implementation runs on the platform equipped

with Intel Core2 Duo CPU, 3.0 GB memory, and Windows Vista operating system.

4.1 Peak Load Reduction and Execution Time

The experiment sets the schedule length, namely, *M*, to 20 time units. If one time unit is equal to 20 *min* as in [14], the total schedule length will be 6.6 hours, and it is sufficiently large for the home appliance schedule. For a task, the start time is selected randomly between 0 and *M*, while the operation length distributes randomly, but it will be set to *M* if the finish time, namely, the sum of start time and the operation length, exceeds *M*. All tasks have the common deadline, namely, *M*, considering the situation a customer orders a set of tasks to be done before he or she returns home. In addition, the power level for each time slot has the value of 1 through 5. Here again, the power unit, such as *w* or *kw*, is not specified. In the subsequent experiments, the total power request is roughly proportional to the number of tasks, as each task has the same average power requirement. The *Earliest* scheduling is selected for performance comparison as in [14]. As illustrated in Fig. 2, this scheme initiates the task as soon as the task is ready. As for the comparison of [14] and our scheme, both schemes find an optimal schedule, but our scheme can reduce the scheduling time incomparably.

Fig. 2 indicates that if we do not control or schedule the tasks, the maximum power requirement reaches 10 at time 6. The first experiment measures the peak load reduction according to the number of tasks. For each task set, just one is set to the preemptive task, while the number of tasks ranges from 3 to 8. For each number of tasks, 10 task sets are generated. In the real life scenario, most appliance operations are nonpreemptive except for the special case such as charging stations. The peak power values of respective sets are measured and averaged. Then, the maximum power is compared with the case of *Earliest* scheduling scheme, the comparison result being plotted in Fig. 6. The figure indicates that the proposed scheme achieves the peak load reduction by up to 23.1 % when the number of tasks is 8, and the performance gap largely increases along with the number of tasks. Apparently, the enhancement will get larger according to the individual power requirement.

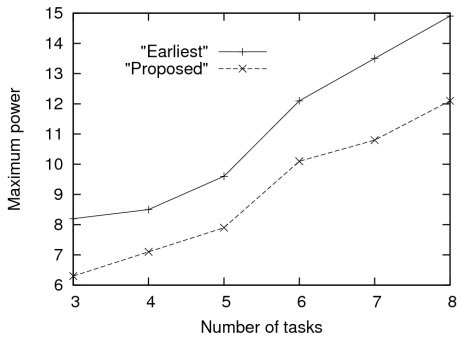

Fig. 6. Maximum power reduction.

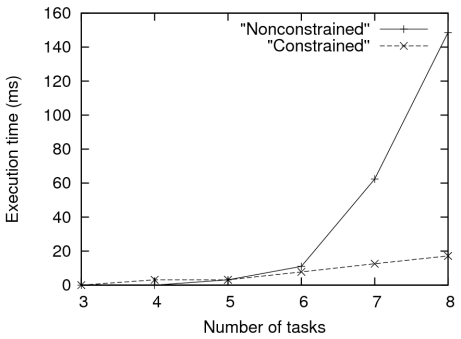

Fig. 7. Execution time comparison.

The next experiment measures the execution time of our scheme, mainly concentrating on the effect of search space pruning. The execution time is measured using Microsoft Windows *GetTickCount* system call which has the 1 *ms* granularity. Figs. 7 and 8

show the measurement results for the cases of 1 preemptive task and 2 preemptive tasks. In each figure, we can see two curves marked *Constrained* and *Nonconstrained*, each of which represents pruning case and nonpruning case, respectively. For the case of more than 2 preemptive tasks, the execution time for the nonconstrained case goes too high, so just the comparison of those two cases is enough to reveal the effect of constraint processing. In Fig. 7, the execution time is at most 140 *ms*, while it reaches 30,000 *ms* in Fig. 8 for the nonconstrained case. As contrast, the constrained search works with much smaller execution time and shows a stable behavior. Actually, the execution time greatly depends on the search space size of the preemptive task. Even if the experiment smoothes this effect by averaging multiple task sets, the task set having a large search space dominates the whole execution time.

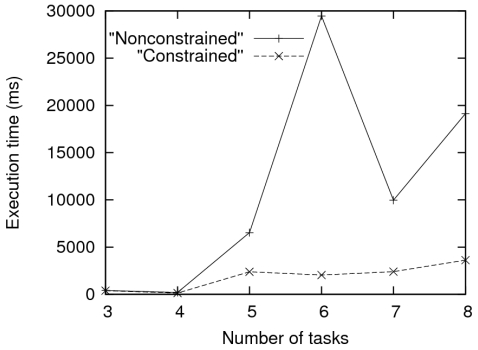

Fig. 8. Execution time comparison.

The constrained search can significantly improve the computation speed by eliminating unnecessary search space traversal, making clear the importance of an efficient heuristic for the practical employment of the scheduling scheme. Anyway, if we further divide N into N_p and N_{np} as the number of preemptive tasks and that of nonpreemptive tasks, respectively, the worst case complexity of search space can be approximated by $O(M^{N_{np}} | (3^{\frac{M}{2}})^{N_p})$. It must be mentioned that, contrary to the task scheduler in the common computer system, the power schedule this paper is targeting at can afford the execution time of a couple of seconds. Moreover, if a customer provides the task profile and triggers the scheduler, the home power controller will automatically generate a schedule and manage their operation, so the customer doesn't need to wait for the schedule to be completed.

4.2 Effect on the Global Peak

When the global coordination of multiple local schedules is not available, respective local peak reduction is important. Moreover, if the scheduler of each home has the same optimization target, tasks for many homes can be scheduled in the same time slot, undesirably making the peak in such time slots. Hence, this section measures the effect of the proposed schedule to the global peak reduction. To calculate the global power consumption, it is necessary to merge a group of schedules. We assume that the slot bound of each

local schedule is aligned for simplicity. Then the power requirement of each slot of respective local schedules is added to generate per-slot power consumption. The global peak can be largest when several local schedules have a peak value on same time slots.

The first experiment measures the average peak value according to the number of scheduling units belonging to a common global schedule group such as a building. The experiment generates 20 task sets and measures the global peak load. Each task set accounts for the change of appliance operations during the specific observation interval. Fig. 9 plots the average peak change according to the number of units, each of which has 5 tasks including 1 preemptive task. When the number of units is equal to or less than 4, the proposed scheme shows a little bit smaller peak value. However, above this point, the earliest allocation scheme shows a smaller peak value. It's because the tasks have common deadline, so the schedule allocates more task operations near the deadline. Even though individual peak values are smaller, peak slots are more likely to concentrate on the same slot. On the contrary, the *Earliest* scheduling scheme makes each operation start at its activation time, randomly distributing the task operation over the schedule interval. By this scheme, local peaks have fewer chances to be on the same slot.

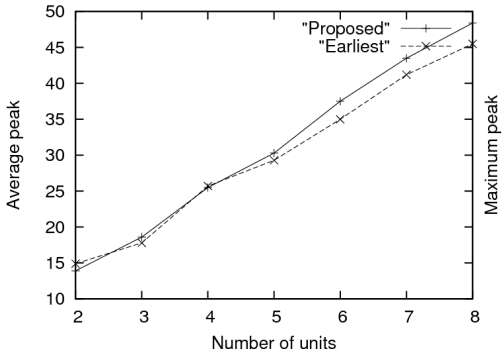

Fig. 9. Global average peak load.

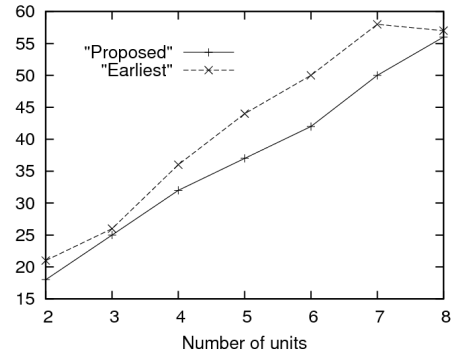

Fig. 10. Global maximum peak load.

Next, Fig. 10 plots the peak for the maximum peak for the generated sets. The maximum peak is the more important performance criteria as it decides the capacity of actual power supply system and power battery. As shown in Fig. 10, the proposed scheme shows smaller peak load even without a global coordination, achieving by up to 16% peak power reduction when the number of units is 6. When the number of units is more than 8, the maximum peaks of both schemes are almost same. The effect of local peak reduction is submerged by the peak concentration to the slots near the common deadline. Anyway, the proposed scheme can prevent the intermittent power load spikes within the scheduling group for the diverse power request pattern changes.

Finally, Fig. 11 plots the effect of number of tasks in the scheduling group having 5 units. In each group, just 1 is a preemptive task. In reality, the instance of preemptive tasks is quite rare. The proposed scheme reduces the peak by up to 20.4% and shows much stable behavior, as it reduces local peak values when the number of tasks is 4. Such predictable pattern can help the power capacity planning. Here again, when the number of tasks is over 8, both schemes show almost same peak values, indicating the impor-

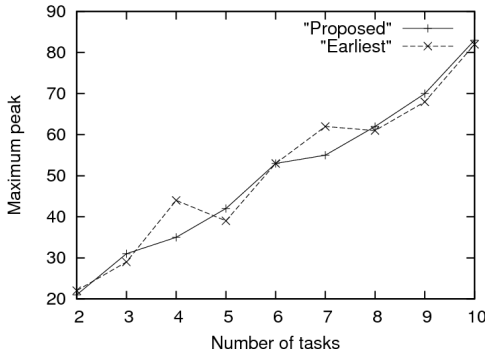

Fig. 11. Global maximum peak load comparison (5 units).

tance of global coordination among scheduling units above that point. At least, for a scheduling unit, not just a single allocation can have the optimal peak, so each scheduling unit can have multiple choices. Appropriately selecting one out of multiple optimal allocations in each scheduling unit can reduce the global peak. However, for the home-scale number of tasks and the reasonable number of units in a scheduling group, the proposed scheme can work well both locally and globally.

5. CONCLUSION

Aiming at reducing the peak load in the individual homes as well as in the system-wide power transmission network, this paper has presented a design, implemented a pilot version, and evaluated the performance of a power consumption scheduler in smart grid homes. Following the task model consist of actuation time, operation length, deadline, and the consumption profile, the scheduler copies the profile for the nonpreemptive task and maps for the preemptive task. The space search mechanism expands recursively to traverse all the possible allocations for a task set. Our implementation shows that this scheme is able to reduce the peak load by up to 23.1%, compared with *Earliest* scheduling. The worst case time complexity is approximated by $O(M^{N_{np}} | (3^{\frac{M}{2}})^{N_p})$, and greatly depends on the search space of a preemptive task. However, constraint processing has reduced the execution time almost to 2% by pruning a search branch when the partial peak already exceeds the current best. Additionally, in terms of global peak reduction, the proposed scheme can reduce the global peak without any global coordination by up to 16% for the home-scale scheduling unit where the number of appliances is generally less than 10.

As future work, we are first planning to develop a sophisticated heuristic and implement a threaded version to cope with the larger number of tasks to extend our scheme to smart buildings. Next, the task model is to integrate the real-time price change and rechargeable battery device such as electric vehicles to the end of efficiently managing the home power consumption with diverse optimization goals, for example, reducing the cost and the like. In addition, along with the power scheduler, we are also pursuing an efficient method to specify the power trade requirement from both sellers and consumers to design a power trader [19].

REFERENCES

1. C. Gellings, *The Smart Grid: Enabling Energy Efficiency and Demand Response*, The Fairmont Press, Liburn, USA, 2009.
2. A. Mady, M. Boubekeur, and G. Provan, "Optimised embedded distributed controller for automated lighting systems," in *Proceedings of the 1st Workshop on Green and Smart Embedded System Technology: Infrastructures*, *Methods*, *and Tools*, 2010.
3. K. Spees and L. Lave, "Demand response and electricity market efficiency," *The Electricity Journal*, Vol. 20, 2007, pp. 69-85.
4. A. Mohsenian-Rad, V. Wong, J. Jatskevich, R. Schober, and A. Leon-Garcia, "Autonomous demand-side management based on game-theoretic energy consumption scheduling for the future smart grid," *IEEE Transactions on Smart Grid*, Vol. 1, 2010, pp. 320-331.
5. Y. Tan, W. Liu, and Q. Qiu, "Adaptive power management using reinforcement learning," *IEEE/ACM International Conference on Computer-Aided Design*, 2009, pp. 461-467.
6. D. Gislason, *ZIGBEE Wireless Networking*, Newnes, Burlington, USA, 2008.
7. M. Yousuf and M. El-Shafei, "Power line communications: An overview – part I," in *Proceedings of International Conference on Innovations and Information Technology*, 2007, pp. 218-222.
8. T. Facchinetti, E. Bibi, and M. Bertogna, "Reducing the peak power through real-time scheduling techniques in cyber-physical energy systems," in *Proceedings of the 1st International Workshop on Energy Aware Design and Analysis of Cyber Physical Systems*, 2010.
9. A. Ipakchi and F. Albuyeh, "Grid of the future," *IEEE Power and Energy Magazine*, 2009, pp. 52-62.
10. J. Lee, H. Song, and A. K. Mok, "Design of a reliable communication system for grid-style traffic control networks," in *Proceedings of the 16th IEEE Real-Time and Embedded Technology and Applications Symposium*, 2010, pp. 133-142.
11. S. Luan, J. Teng, S. Chan, and L. Hwang, "Development of a smart power meter for AMI based on ZigBee communication," *Power Electronics and Drive Systems*, 2009, pp. 661-665.
12. E. Bonneville and A. Rialhe, "Demand side management for residential and commercial end-users," http://www.leonardo-energy.org/Files/DSM-commerce.pdf, 2006.
13. S. Abras, S. Pesty, S. Ploix, and M. Jacomino, "An anticipation mechanism for power management in a smart home using multi-agent systems," in *Proceedings of the 3rd International Conference on From Theory to Applications*, 2008, pp. 1-6.
14. O. Derin and A. Ferrante, "Scheduling energy consumption with local renewable micro-generation and dynamic electricity prices," in *Proceedings of the 1st Workshop on Green and Smart Embedded System Technology: Infrastructures*, *Methods*, *and Tools*, 2010.
15. Y. Katsigiannis, P. Georgilakis, and E. Karapidakis, "Multiobjective genetic algorithm solution to the optimum economic and environmental performance problem of small autonomous hybrid power systems with renewable," *IET Renewable Power Generation*, 2010, pp. 404-419.
16. D. Dollen, "Intelligrid consumer portal telecommunication assessment and specifica-

tion," EPRI Technical Report 1012826, 2005.
17. A. Mohsenian-Rad and A. Leon-Garcia, "Optimal residential load control with price prediction in real-time electricity pricing environments," *IEEE Transactions on Smart Grid*, Vol. 1, 2010, pp. 120-133.
18. J. Lee, G. Park, M. Kang, H. Kwak, and S. Lee, "Design of a power scheduler based on the heuristic for preemptive appliances," in *Proceedings of Asian Conference on Intelligent Information and Database Systems*, 2011, pp. 396-405.
19. K. Matsumoto, T. Maruo, N. Mori, M. Kitayama, and I. Izui, "A communication network model of electric power trading systems using web services," in *Power Tech Conference Proceedings*, 2003, pp. 23-26.

Junghoon Lee received the B.S., M.S., and Ph.D. from Department of Computer Engineering, Seoul National University, Korea. From 1990 to 1992, and also in 1996, he was a senior research engineer at Lab. of optical telecommunication, Daewoo Telecom, Republic of Korea. In 1997, he joined the Department of computer science and statistics at Jeju National University. From 2003 to 2005, he was a visiting scholar at Department of Computer Science, University of Texas at Austin. His research interests include real-time communication and wireless network.

Hye-Jin Kim received M.S. at Department of Computer Science and Statistics from Jeju National University, Jeju, Korea. She was a researcher at ITRC (Information Technology Research Center) of Jeju National University from 2009 to 2011. Currently, she is working several projects at Jeju National University as a Ph.D. course student. Her research interests are telematics and smart grid.

Gyung-Leen Park received B.S. in Department of Computer science from Chung-Ang University. He received M.S. and Ph.D. from Computer Science and Engineering Department at the University of Texas at Arlington, respectively. His research interests include scheduling in parallel and distributed systems, mobile computing, and telematics. In 1997, he was an Assistant Professor at the University of Texas at Arlington. In 1998, he joined the Department of computer science and statistics at Jeju

National University, Korea. Currently, he is the director of the Smart Grid Research Center, Jeju National University.

Mikyung Kang received the B.S., M.S., and Ph.D. degrees at Department of Computer Science and Statistics from Jeju National University, Jeju, Korea. From 2004 to 2006, she was a researcher at ITRC (Information Technology Research Center) of Jeju National University. In 2006, she was selected as a Post-Doctoral Fellow by KRF (Korea Research Foundation). In 2007, she was a visiting scholar at USC/ISI (University of Southern California/Information Sciences Institute). In 2008, she joined the APEX (Adaptive Parallel Execution) Computing Group at USC/ISI. Her research interests are high performance cloud computing, multi-core software, real-time communication, and wireless network.