Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence

Randomized Load Control: A Simple Distributed Approach for Scheduling Smart Appliances

Menkes van den Briel, Paul Scott, Sylvie Thiébaux
NICTA and Australian National University
Australia
{menkes, paul.scott, sylvie.thiebaux}@nicta.com.au

Abstract

A significant portion of the electricity network capacity is built to run only a few days a year when demand peaks. As a result, expensive power generation plants and equipment costing millions of dollars are sitting idle most of the time, which increases costs for everyone. We present *randomized load control*, a simple distributed approach for scheduling smart appliances. Randomized load control schedules the start time of appliances that are programmed to run within a specified time window, so that the aggregate load achieves a given ideal load. Our results show that we do achieve the given ideal load to a great extent. This is remarkable as the approach is completely distributed and preserves customer privacy as the scheduling happens within each house or building separately.

1 Introduction

Smart grid initiatives are about modernizing the electrical grid and integrating large amounts of renewable energy. The aim is to make the grid more reliable, lower system losses, improve utilization of power generation capacity and make it easier to manage outages. Load management plays a significant role in realizing these goals as it aims to reduce peak demand, which causes a strain on the grid network and leads to soaring costs for the utility.

Smart appliances are the next generation of appliances that have the ability to communicate with a smart meter or an energy management system. They are intended to reduce energy use and make our lives easier at the same time. For example, a smart washing machine can automatically adjust to the low-energy wash cycle during high-rate periods or inform you when it is done. Smart appliances may also be programmed to run within a specified time window. This allows users to select a period of time for when the appliance may run.

Consider, for example, a customer who decides to wash his clothes at 8:00am in the morning just before leaving to work. He wants his wash to be done by 5:30pm, which is when he expects to get back home. The customer programs the washing machine to run the long hot wash cycle to be completed by 5:30pm. If the long hot wash cycle takes 2 hours and 15 minutes then the washing machine may start as early as 8:00am or as late as 3:15pm in order to finish by 5:30pm. The customer is indifferent about when the washing machine starts running as long as it is finished by 5:30pm.

Imagine thousands or even millions of customers programming their washing machine, dryer, dishwasher, electric vehicle, or other appliance in this way. Some customers may want to run their appliances overnight, while others prefer to run theirs during the day. Some customers are flexible and allow for wide time windows, whereas others are stricter and allow for only narrow time windows.

The utility can take advantage of this, because by scheduling the start time of many appliances it can perform a higher degree of load management. On the other hand, when many appliances are programmed in this way it becomes a challenge to schedule them. With potentially millions of customers, the scheduling should be done carefully, because if too many appliances are scheduled to start at the same time it can cause peaks in energy use. While a centralized scheduler has an overview of what has been scheduled so far, the scale and dynamic nature of the problem (requests to run an appliance within a certain time window arrive constantly) can make such an approach impractical. Also, privacy can be a major issue in such an approach. So instead, we focus on a distributed approach where the scheduling is done at the house or building level. The advantage of such an approach is that it is efficient even for very large networks.

We refer to the energy that the utility has to supply as the *load*. The load is the sum of the *shiftable load* and *non-shiftable load*. The shiftable load is an aggregate of all individual loads that can be programmed to run within a certain time window and the non-shiftable load is an aggregate of all individual loads that are required on demand including, for example, lights and home entertainment. An example load forecast and its breakdown into shiftable and non-shiftable load are shown in Figure 1 (top).

We propose *randomized load control* as a simple distributed approach to schedule smart appliances. The main idea is to schedule smart appliances based on the *ideal shiftable load*. The ideal shiftable load can be derived from the *ideal load* and a forecast of the non-shiftable load. The ideal load is the load that the utility wishes to supply, for example, in order to minimize operating cost. An example ideal load, non-shiftable load forecast, and the resulting ideal shiftable load are shown in Figure 1 (bottom).

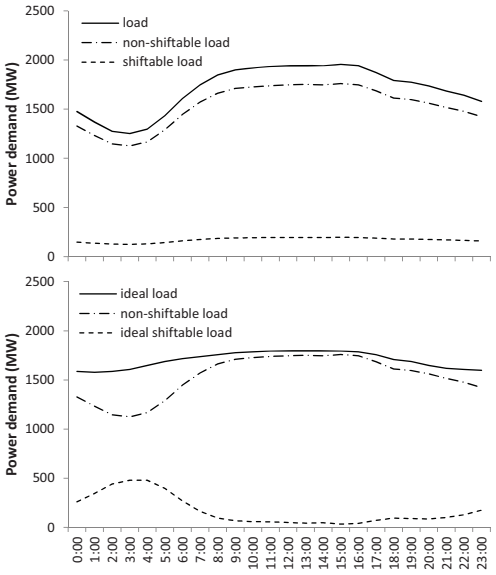

Figure 1: The load is the sum of shiftable and non-shiftable load (top). The ideal shiftable load is the ideal load minus the non-shiftable load (bottom).

In randomized load control, the utility transmits the ideal shiftable load to each smart meter in its network. When a customer programs a smart appliance to run within a specified time window, a request is send to his smart meter. The smart meter then calculates a start time using the time window and the energy consumption profile of the request. The start time is chosen at random, following a probability distribution and then communicated back to the smart appliance. When many appliances in a smart grid are scheduled in this way, the aggregate load will try to achieve the ideal shiftable load. It is important to note that while there is a two-way communication between smart meters and smart appliances, only a one-way communication is needed from the utility to the smart meters in its network. As a result, randomized load control ensures customer privacy.

Randomized load control shows some similarities with the concept of *direct load control*. In direct load control the utility directly controls the load of participating appliances that are selected by the customer. This is done by sending a signal that forces these appliances to consume only a fraction of their normal power consumption. Customer participation is voluntary, but participation in load control events generally is not. For example, a customer may allow the utility to have direct control over the pool pump, water heater, or the air conditioning unit during periods of peak demand.

In randomized load control the utility indirectly controls the aggregated load of all appliances that are programmed to run within a certain time window. Customer participation is voluntary, because the customer can always choose to run the appliance without providing a time window. The key difference with direct load control is that in randomized load control the utility does not have, nor need, direct control over individual appliances. The utility just needs to transmit the ideal shiftable load to all smart meters in its network. The smart meters who are distributed over the grid network then do the scheduling for potentially millions of appliances by randomly assigning start times, following a probability distribution. Hence, the utility indirectly controls the aggregated load of all these appliances. Specifically, the load control is achieved by dictating the shape of this probability distribution, which is derived from the ideal shiftable load.

In terms of participation incentives, wider time windows are preferred by the utility as they allow for a better scheduling of the shiftable load. As a result, the utility should encourage customers to be flexible by providing discounted rates for wider time windows. These discounted rates may be based on real time prices, time-of-use prices, or any other pricing scheme that the utility has in place. Essentially, randomized load control can be implemented on top of, or in parallel to existing load management initiatives.

The remainder of this document is organized as follows. Section 2 discusses related work and Section 3 describes the randomized load control approach. Section 4 describes a way to calculate an optimal schedule using perfect knowledge and experimental results are presented in Section 5. Finally, conclusions are described in Section 6.

2 Related Work

The problem of scheduling smart appliances is often studied in the context of *demand side management* (DSM). DSM can be seen as a set of interconnected activities that enables customers to change the shape and magnitude of their electricity load profile [Gellings and Chamberlin, 1993]. In general, the goal of DSM is to encourage customers to use less energy during periods of peak demand, or to move the use of energy to off-peak hours. While DSM may not decrease total energy use, it is expected to reduce the need for grid expansion investments, which are very capital intensive.

The demand for plug-in hybrid electric vehicles (PHEVs) is expected to increase, which makes the scheduling of smart appliances even more important. PHEVs are high-power appliances that during charging can almost double the load of an average residential customer [Ipakchi and Albuyeh, 2009].

Various approaches for scheduling smart appliances have been proposed. One of them is *dynamic pricing*, also referred to as *smart pricing*, which encourages customers to manage their own loads. There are several kinds of dynamic pricing programs, including: real time pricing (RTP), critical peak pricing (CPP), and time-of-use (TOU) pricing.

In RTP, the utility provides a price for electricity that reflects the cost of generating it, for example, a day or an hour ahead of time. Hence, with RTP the price of electricity may vary significantly at different times of the day and even at different times of the year. RTP allows price sensitive customers to schedule their appliances outside periods of peak demand when the prices are high. In CPP, prices may reflect the cost of generating and/or purchasing electricity at the wholesale level based on real-time market conditions during certain periods of peak demand. At other periods, TOU pricing is in effect. TOU pricing is not widely considered to be dynamic because prices are not based on market conditions. Instead,

prices are fixed, but do reflect higher costs during periods of peak demand and lower cost during off-peak hours.

While various studies, such as [Pyrko, 2006; Sanghvi, 1989], have shown the value of dynamic pricing in reducing peak demand, there are also concerns that programs like these can be difficult to understand by customers [Ann-Piette *et al.*, 2009]. An experimental study on user acceptance of smart home technologies concludes that in order to allow acceptance by a broad customer base, convenient and customer friendly solutions are recommended [Paetz *et al.*, 2011]. Another issue with dynamic pricing is load synchronization. In particular, with RTP and TOU pricing a large portion of the load may be shifted from peak hours to off-peak hours potentially creating new peaks at these times [Mohsenian-Rad and Leon-Garcia, 2010; Ramchurn *et al.*, 2011]. There are studies, however, that use penalties for deviation from past behavior [Voice *et al.*, 2011], or implement a decision-theoretic agent [Reddy and Veloso, 2012] to mitigate the impact of this effect.

Most DSM approaches have focused on individual interactions between the utility and the customers. For example, in RTP, each customer is expected to respond to real time prices individually. Rather than focusing on each customer individually, the objective of DSM is to ensure that the aggregate load satisfies some desired properties [Mohsenian-Rad *et al.*, 2010]. Ultimately, the utility is only concerned about the total load that it has to supply. This is what separates randomized load control from most of the DSM approaches that have been studied and/or deployed so far. In randomized load control, the utility transmits the ideal shiftable load to each customer and based on this load each smart appliance that is programmed to run within a specified time window is scheduled.

The work closest to randomized load control is a water-filling based scheduling algorithm [Shinwari *et al.*, 2012]. The water filling algorithm distinguishes hard (non-shiftable) loads from soft (shiftable) loads and defines the water level to be greater than or equal to the maximum value of the hard load. Compared to randomized load control, the water level could be interpreted as having an ideal load that is constant. In randomized load control the ideal load can take on any shape, consequently it provides a more flexible approach to scheduling smart appliances.

3 Randomized Load Control

Let T be the number of time periods. For the utility, we respectively define s_t, n_t, and d_t to be the shiftable, non-shiftable, and load forecast at time period t. We also respectively define s_t^* and d_t^* to be the ideal shiftable load and the ideal load at time period t. Hence, as shown in Figure 1, for each time period t the load is the sum of the shiftable and non-shiftable load $d_t = s_t + n_t$ and the ideal shiftable load is the ideal load minus the non-shiftable load $s_t^* = d_t^* - n_t$.

For each smart meter, let N be a set of requests from the smart appliances it interacts with. Each request i defines the operating properties of the appliance that needs to be scheduled. The duration or length of the request, in number of time periods, is δ_i and the time window is characterized by the earliest start time period e_i and the latest start time period l_i. The energy consumption profile of request i is defined by the vector Γ_i:

$$\Gamma_i = [\gamma_{i1}, ..., \gamma_{i\delta_i}]$$

where the scalar $\gamma_{i\tau}$, such that $1 \leq \tau \leq \delta_i$, is the energy consumption of request i in the τ-th time period from starting the appliance.

We can schedule each request independently by generating a probability distribution function that is based on the ideal shiftable load. This probability distribution function is then used to randomly select a start time for the request. For each request $i \in N$ and each time period t, the probability that request i is scheduled to start in time period t is defined as:

$$P_{it} = \begin{cases} \dfrac{count(i,t)}{sumcount(i)} & \text{if } e_i \leq t \leq l_i \\ 0 & \text{otherwise} \end{cases}$$

The $count(i,t)$ function counts how many hypothetical requests with the same profile as request i can be scheduled to start at time period t in order to achieve the ideal shiftable load s_t^*. The $sumcount(i)$ counts how many such requests could be scheduled over the given time window. The $sumcount(i)$ is simply the sum of $count(i,t)$ over all time periods $e_i \leq t \leq l_i$.

$$sumcount(i) = \sum_{e_i \leq t \leq l_i} count(i,t)$$

While there are different ways to define $count(i,t)$, we describe and compare three approaches. Note that, the time windows provide hard constraints. Each request must be scheduled within its time window unless it is canceled by the customer. As a result, the probability distribution P_{it} is defined over the allowable time window only, outside this time window the probabilities are zero.

The first approach is similar to the water-filling based scheduling method [Shinwari *et al.*, 2012], but the fill-to level at time period t is determined by the ideal shiftable load $s_t^* (= d_t^* - n_t)$ rather than the difference between the maximum non-shiftable load and the non-shiftable load $(max_{1 \leq t \leq T}\{n_t\} - n_t)$.

$$count(i,t) = s_t^* \tag{1}$$

Since this approach ignores the energy consumption profile of request i we refer to it as *unit-count*. This may sound somewhat counter intuitive, but one could argue that this approach implicitly assumes that each request has a unit load profile, that is, $\Gamma_i = [1]$ for each $i \in N$.

The second approach modifies the first approach by taking the load profile of the request into consideration. In words, it identifies the most restrictive count on how many hypothetical requests with the same profile as request i can be scheduled to start at time period t.

$$count(i,t) = min_{1 \leq \tau \leq \delta_i} \left\{ \frac{s^*_{t+\tau-1}}{\gamma_{i\tau}} \right\} \tag{2}$$

We refer to this approach as *profile-count*. Note that, if request i has a unit load profile, then the profile-count approach is identical to the unit-count approach.

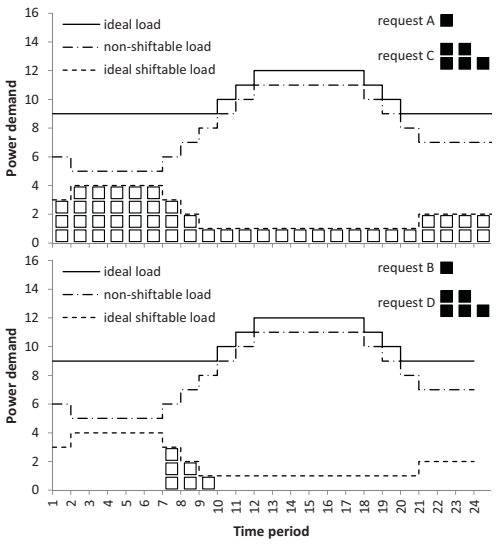

Figure 2: Requests A and C have an earliest start time of 0 and a latest start time of 23 (top). Requests B and D have an earliest start time of 6 and a latest start time of 8 (bottom).

Example 1 (Unit Load). *We assume that each time period takes one hour. Consider request A, see Figure 2 (top), which has an earliest start time* $e_A = 1$ *and a latest start time* $l_A = 24$*. Request A has a duration of one hour* $\delta_A = 1$ *and a unit load profile* $\Gamma_A = [1]$*. In order to calculate the probability that request A is scheduled to start at time period 0 we must calculate* $count(A, 1)$ *and* $sumcount(A)$*. For unit-count we have* $count(A, 1) = s^*_1 = 3$ *and* $sumcount(A) = 48$*. Hence,* $P_{A,1} = 3/48 = 0.0625$ *and in a similar way we can calculate* $P_{A,2} = 4/48 = 0.0833$ *and* $P_{A,3} = 4/48 = 0.0833$*, and so on. For profile-count we have* $count(A, 1) = min_{1 \le \tau \le 1}\{s^*_{1+\tau-1}/\gamma_{A\tau}\} = min\{3/1\} = 3$ *and* $sumcount(A) = 48$*. Hence,* $P_{A,1} = 3/48 = 0.0625$*, and similarly* $P_{A,2} = 4/48 = 0.0833$*,* $P_{A,3} = 4/48 = 0.0833$ *and so on.*

Consider request B, see Figure 2 (bottom), with $e_B = 7$*,* $l_B = 9$*,* $\delta_B = 1$ *and* $\Gamma_B = [1]$*. For unit-count and profile-count we have* $P_{B,7} = 3/6 = 0.5$*,* $P_{B,8} = 2/6 = 0.3333$*,* $P_{B,9} = 1/6 = 0.1667$*, and for all other time periods we have a probability of zero.*

Example 2 (Load Profile). *Again, we assume that each time period takes one hour. Consider request C, see Figure 2 (top), with* $e_C = 1$*,* $l_C = 24$ *and* $\delta_C = 3$*. The load profile for request C is* $\Gamma_C = [2, 2, 1]$*. For unit-count we have* $count(C, 1) = s^* = 3$ *and* $sumcount(C) = 48$*. Hence,* $P_{C,1} = 3/48 = 0.0625$ *and in a similar way we can calculate* $P_{C,2} = 4/48 = 0.0833$*,* $P_{C,3} = 4/48 = 0.0833$*, and so on. For profile-count we have* $count(C, 1) = min_{1 \le \tau \le 3}\{s^*_{1+\tau-1}/\gamma_{C\tau}\} = min\{3/2, 4/2, 4/1\} = min\{1.5, 2, 4\} = 1.5$ *and* $sumcount(C) = 22.5$*. Hence,* $P_{C,1} = 1.5/22.5 = 0.0667$ *and similarly* $P_{C,2} = 2/22.5 = 0.0889$ *and* $P_{C,3} = 2/22.5 = 0.0889$*.*

Consider request D, see Figure 2 (bottom), with $e_D = 7$*,* $l_D = 9$*,* $\delta_D = 3$ *and* $\Gamma_D = [2, 2, 1]$*. For unit-count we have* $P_{D,7} = 3/6 = 0.5$*,* $P_{D,8} = 2/6 = 0.3333$*,* $P_{D,9} = 1/6 = 0.1667$*, and for all other time periods we have a probability of zero. For profile-count we have* $P_{D,7} = min\{3/2, 2/2, 1/1\}/2 = 0.5$*,* $P_{D,8} = 0.5/2 = 0.25$*,* $P_{D,9} = 0.5/2 = 0.25$*, and for all other time periods we have a probability of zero.*

The third approach involves solving an optimization problem. The main idea here is to curve-fit the entire ideal shiftable load by using only load profiles of request i. The number of profiles needed would then correspond to the number of hypothetical requests that can be scheduled to start in time period t in order to achieve the ideal shiftable load, which is $count(i, t)$.

To curve-fit the ideal shiftable load with load profiles of request i we minimize the squared deviation between the ideal shiftable load and the hypothetical load that is achieved from aggregating these load profiles. In particular, we define the following variables:

- $y_t \ge 0$, the number of load profiles (of request i) to start at time period t.
- $z_t \ge 0$, the aggregated load achieved at time period t.

This allows us to formulate the following optimization problem:

$$\text{Min} \sum_{1 \le t \le T} (s^*_t - z_t)^2$$

$$\sum_{1 \le \tau \le \delta_i} \gamma_{i\tau} y_{t-\tau+1} = z_t \quad 1 \le t \le T \tag{3}$$

$$y_t \ge 0 \quad 1 \le t \le T \tag{4}$$

$$z_t \ge 0 \quad 1 \le t \le T \tag{5}$$

Constraints (3) ensure that the z_t variables represent the aggregated load achieved at time period t. Constraints (4) and (5) are the non-negativity constraints on the variables.

The value of the y_t variables gives the number of hypothetical requests that can be scheduled to start in time period t in order to achieve the ideal shiftable load. Hence, we have:

$$count(i, t) = y_t \tag{6}$$

We refer to this approach as *sqdev-count* as it minimizes the squared deviation. Note that, if request i has a unit load profile, then the sqdev-count approach is identical to the unit-count approach. With unit load profiles we have $\gamma_t = 1$, so one can set y_t equal to s^*_t for each $1 \le t \le T$.

We tried various other alternatives to define $count(i, t)$ as well, but since the results were quite similar we decided to present only these three as their characteristics are the most distinct.

4 Optimal scheduling with perfect knowledge

In our experiments we want to compare the three approaches described in Section 3. Since our objective is to try and achieve the ideal shiftable load, one may think that the optimal schedule always results in the ideal shiftable load. This, however, is not true because of the time windows. Each request has a certain time window during which the appliance

must run. If the distribution of these time windows is disproportionate with respect to the shape of the ideal shiftable load, then the optimal schedule will not achieve the ideal shiftable load.

In order to calculate the optimal schedule we must determine a start time for each request such that the aggregate load achieves the ideal shiftable load as much as possible. For this, we assume a central scheduler who has perfect knowledge of all (current and future) requests. The problem is that even with perfect knowledge, finding the optimal start time for thousands or even millions of requests remains a computational nightmare. In our experiments, however, we use k_i instantiations of each request $i \in N$ (see, for example, column four in Table 1). By using k_i instantiations of a request i, we effectively transform a request as defined in Section 3 into a request type, which allows us to scale up significantly. Since the number of request types is typically much smaller than the number of request instantiations it allows for a more simplified model.

Again, the main idea is to curve-fit the ideal shiftable load, but this time with the load profiles of all requests types $i \in N$. The objective is to minimize the squared deviation between the ideal shiftable load and the load achieved by scheduling the requests, such that all k_i requests of type i are scheduled to start within their respective time windows. In particular, we define the following variables:

- $x_{it} \in \mathbb{N}$, the number of requests of type i to start at time period t.

In addition, we have the z_t variables as defined previously in Section 3. Now, we formulate the following optimization problem:

$$\text{Min} \sum_{1 \le t \le T} (s^*_t - z_t)^2$$

$$\sum_{\substack{i \in N, 1 \le \tau \le \delta_i: \\ e_i \le t-\tau+1 \le l_i}} \gamma_{i\tau} x_{i,t-\tau+1} = z_t \quad 1 \le t \le T \tag{7}$$

$$\sum_{e_i \le t \le l_i} x_{it} = k_i \quad i \in N \tag{8}$$

$$x_{it} \in \mathbb{N} \quad i \in N, 1 \le t \le T \tag{9}$$

$$z_t \ge 0 \quad 1 \le t \le T \tag{10}$$

Constraints (7) ensure that the z_t variables represent the aggregated load achieved at time period t, and constraints (8) ensure that all instantiations of request type i are scheduled within their respective time window. Constraints (9) and (10) are the non-negativity (and integer) constraints on the variables.

In our experiments we solve the linear programming relaxation of this formulation as the integrality gap is small.

5 Experimental Results

We present two experiments to compare the three approaches: *unit-count*, *profile-count*, and *sqdev-count*. The first set of experiments is to analyze the characteristics of each approach and the second experiment is to apply the approaches using representative data. While we performed several more experiments than presented here, we believe that these two summarize our main results. Also, we looked considerably into comparing randomized load control with other approaches, but we were unable to determine a like-for-like comparison with prior works primarily because to required inputs are different. Oftentimes a real time price is used instead of the ideal shiftable load.

Request	Load Profile	Time Window	Quantity
Test 1	(60, 1.0).	00:00-23:59	1,000
Test 2	(120, 1.0).	00:00-23:59	1,000
Test 3	(60, 1.0).	00:00-07:59	1,000
		08:00-15:59	1,000
		16:00-23:59	1,000
		00:00-23:59	1,000
Test 4	(120, 1.0).	00:00-07:59	1,000
		08:00-15:59	1,000
		16:00-23:59	1,000
		00:00-23:59	1,000

Table 1: Requests used in first set of experiments.

Request	Load Profile	Time Window	Quantity
ShortWash	(5, 1.8), (50, 0.6), (5, 1.8).	09:00-14:59	170,000
		15:00-17:59	170,000
		17:00-07:59	300,000
LongWash	(5, 2.4), (50, 0.6), (5, 1.2), (45, 0.6), (5, 2.4).	08:00-18:59	90,000
		09:00-13:59	170,000
		14:00-20:59	170,000
		16:00-08:59	300,000
ShortDry	(50, 1.8).	13:00-16:59	30,000
		14:00-20:59	90,000
		18:00-09:59	240,000
LongDry	(90, 1.8).	10:00-15:59	30,000
		14:00-23:59	90,000
		09:00-18:59	100,000
		18:00-08:59	240,000
ShortDish	(50, 0.6), (10, 1.8).	07:00-14:59	36,000
		12:00-14:59	100,000
		21:00-08:59	100,000
LongDish	(80, 0.6), (20, 1.8).	19:00-08:59	100,000
		20:00-09:59	100,000
SmallPHEV	(240, 3.0).	08:00-14:59	800
		19:00-06:59	4,000
		17:00-07:59	4,300
LargePHEV	(300, 3.6).	18:00-08:59	4,300
		19:00-06:59	4,300

Table 2: Requests used in second experiment.

Table 1 and 2 show the input data that we used. For example, in Table 2, the power demand of the ShortWash is 1.8kW during the first 5 minutes, 0.6kW during the next 50 minutes and 1.8kW during the last 5 minutes. Hence, the total energy consumption of the ShortWash is 0.8kWh (= (5*1.8 + 50*0.6 + 5*1.8)/60). We have a total of 170,000 ShortWash requests with a time window of 09:00-14:59, another 170,000 ShortWash requests with a time window of 15:00-17:59, and so forth. Each time window indicates the earliest start time and the latest start time. The latest start time is always rounded down based on the granularity of a time period. So, if we use time periods of 1 hour (5 minutes) then 12:59 means the latest start time period is 12:00 (12:55) and not 13:00.

For our first set of experiments we ran four tests using the requests, with respective names, given in Table 1. The results are given in Figure 3, in which the four tests are presented from left to right. The x-axis shows time and the y-axis shows

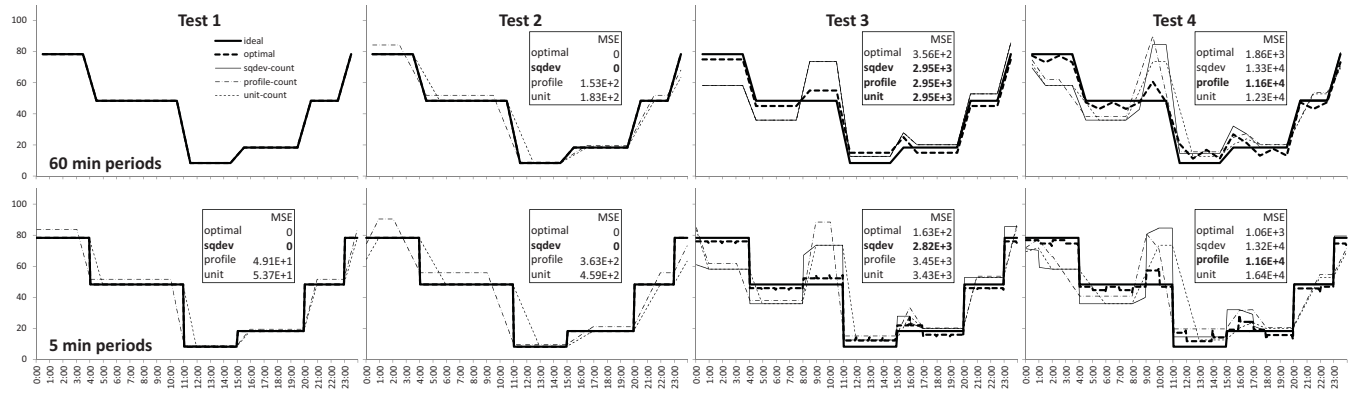

Figure 3: Results from experiment 1. Test 1 through Test 4 are presented from left to right. Time periods of 60 minutes (5 minutes) are used in the top (bottom) graphs. The legend in the top left graph applies to all graphs in this figure.

power demand in kW. The top graphs use time periods of 1 hour and the bottom graphs use time periods of 5 minutes (shorter time periods tend to exacerbate the phenomena observed). The curve labeled *ideal* represents the ideal shiftable load. The other curves present the expected average of each approach rather than the result of sampling. Recall that requests are scheduled according to a probability distribution. If we were to sample, then the curves would zigzag around the expected average that is shown.

Apart from the top left graph which includes the legend that applies to all graphs in this figure, each graph displays a table with the mean squared error (MSE) indicating a measure of "fitness" in curve-fitting the ideal shiftable load. The approach with the lowest MSE, not including optimal, is highlighted in bold.

A couple of observations. In Test 1 and 3 using time periods of 1 hour, the three approaches provide the same results. The reason for this is that the requests in these tests have a unit load profile. While a unit load can be defined to be any quantity, in this particular example one unit load is exactly 1kWh as both Test 1 and 3 draw 1 kW for 60 minutes. When we consider only unit load profiles the three approaches all produce the same probability distribution and so the expected averages are identical.

It is interesting to note the bump between 08:00 and 10:00 in Test 3 and 4 when using both time periods of 1 hour and of 5 minutes. First, it is important to note that in these tests the optimal schedule does not achieve the ideal shiftable load, and even shows a bit of a bump at these times. The reason for these bumps is that the total load that needs to be scheduled between 08:00 and 15:59 is significantly more than the total ideal shiftable load during that time. Because higher probabilities are given to time periods between 08:00 and 10:00 (due to the shape of the ideal shiftable load) these periods end up with most of the load that is scheduled for 08:00-15:59.

In general, but especially looking at Test 1 and 2, we observe that the profile-count approach is reactive towards upwards changes and anticipatory towards downward changes in the ideal shiftable load. The unit-count approach is reactive towards any change. The sqdev-count approach, on the other hand, mostly imitates the shape of the ideal shiftable load. This is in line with what one would expect given the way these approaches calculate $count(i, t)$.

In our second experiment we use more representative data. In fact, the load curve shown in Figure 1 has the same shape as the average daily load (not including weekends) for the state of New South Wales, Australia over the month December, 2010[1]. The ideal load in Figure 1 is an estimate and the non-shiftable load is set to 90% of the total load. Hence, the shiftable load takes up 10% of the total load, which until PHEVs penetration increases is likely higher than actual. As a comparison, Ramchurn et al. [2011] mention that the shiftable load takes up around 20% of the total energy consumption of a house. Total load, however, typically breaks down into residential, commercial and industrial.

The set of requests in Table 2 is derived from the number of households[2], the average usage, and the energy use per cycle[3]. The load profiles and time windows were selected to ensure coverage and diversity at the same time.

Figure 4 shows the results of using three different ideal shiftable load curves with the same area under the curve. The x-axis shows time and the y-axis shows power demand in MW. The curve labeled *uniform* represents the load achieved from scheduling the requests uniformly over their respective time window. The ideal shiftable load in the first (left) graph is identical to the one in Figure 1. The ideal shiftable load in the second graph is constant, and the one in the third graph mirrors the ideal shiftable load in the first graph. While the ideal shiftable load in the second and third graph likely won't have any practical relevance, we use these three very different curves to show the effectiveness of randomized load control.

Each graph in Figure 4 uses the same input from Table 2, but as can be seen, the different ideal shiftable load curves will lead to a very different scheduling of the requests. The three graphs show that utility can achieve a significant level

[1]http://www.aemo.com.au/Electricity/Data/Price-and-Demand/Aggregated-Price-and-Demand-Data-Files

[2]http://www.abs.gov.au/AUSSTATS/abs@.nsf/DetailsPage/1338.1Dec%202010?OpenDocument

[3]http://www.carbonfootprint.com/energyconsumption.html

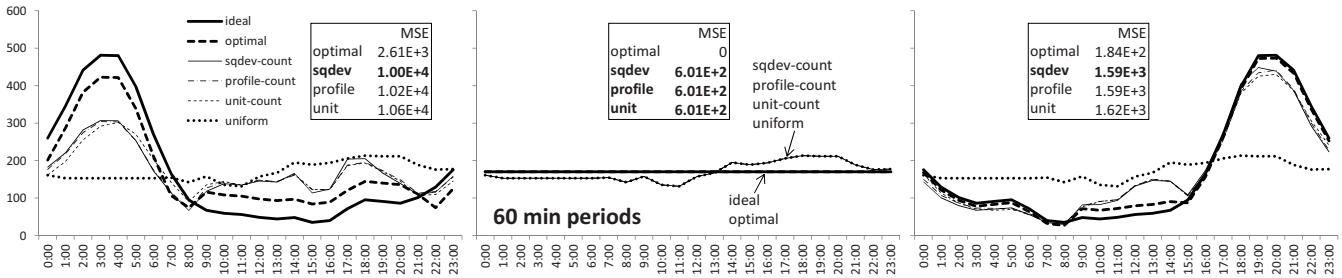

Figure 4: Results from experiment 2. The legend in the left graph applies to all graphs in this figure.

of control over the aggregated load of all requests in Table 2. We also looked at the results for time periods of 5 minutes and 1 minute and they are very similar.

The quality of curve-fitting the ideal shiftable load really depends on the shape of the ideal shiftable load and the distribution of the time windows of the requests. For example, randomized load control gives a much better fit in the third graph than in the first graph of Figure 4. The reason is that there are more requests with time windows covering the evening hours than those covering the early morning hours. This was also observed using other ideal shiftable load curves.

Overall, having run various tests, we believe that randomized load control provides a very simple way to influence the shiftable load. Given its simplicity, scalability, and the fact that no sensitive data from the consumer is shared with the utility, we believe that the approach can be very practical.

6 Conclusions

Randomized load control is a simple distributed approach for scheduling smart appliances. The main idea is to schedule smart appliances based on a probability distribution function that is derived from the ideal shiftable load. Unlike existing approaches where customers are expected to respond to real time prices, in randomized load control the utility simply needs to motivate its customers to program their appliances to run within a specified time window. The utility could achieve this by, for example, providing discounted rates for wider time windows.

Apart from residential appliances, the approach applies to any appliance (including commercial and industrial) that can be scheduled to run within a specified time window. For example, charging electric golf carts used by security, gardeners, and golf clubs.

Our results show that randomized load control can achieve the ideal shiftable load to a great extent. This lowers the operating cost for the utility, which ultimately lowers the utility bill for the customers. Given that the approach is relatively simple, respects privacy, and gives customers adequate flexibility, we believe it can be very practical.

There are, however, several directions that require future work. Even though our results are promising, further validation is needed. We are looking into using comprehensive power systems simulation software such as GridLAB-D[4], and analyzing the robustness of randomized load control. For example, what if the ideal load and the non-shiftable load are over- or under-forecasted, or similarly what if the customer participation in randomized load control is higher or lower than expected?

Other directions for future work include: (1) analyzing customer incentives that are needed for randomized load control to be successful, and (2) considering physical and operational constraints of the underlying power distribution system. Such constraints may depend on network capacities, voltage regulation, or the fluctuations of renewable energy generation. This would allow us to tailor an ideal shiftable load curve for individual substations and study the impact on the overall network.

[4]http://www.gridlabd.org/

Acknowledgements

This work is supported by NICTA's Optimization Research Group as part of the Future Energy Systems project. We thank our project members and reviewers for useful discussions and helpful suggestions. NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

References

[Ann-Piette *et al.*, 2009] M. Ann-Piette, G. Ghatikar, S. Kiliccote, D. Watson, E. Koch, and D. Hennage. Design and operation of an open, interoperable automated demand response infrastructure for commercial buildings. In *Journal of Computing and Information Science in Engineering*, volume 9, pages 1–99, 2009.

[Gellings and Chamberlin, 1993] C.W. Gellings and J.H. Chamberlin. *Demand Side Management: Concepts and Methods*. PennWell Books, 1993.

[Ipakchi and Albuyeh, 2009] A. Ipakchi and F. Albuyeh. Grid of the future. *IEEE Power Energy Magazine*, 7(2):52–62, 2009.

[Mohsenian-Rad and Leon-Garcia, 2010] A.H. Mohsenian-Rad and A. Leon-Garcia. Optimal residential load control with price prediction in real-time electricity pricing environments. *IEEE Transactions on Smart Grid*, 1(2):120–133, 2010.

[Mohsenian-Rad *et al.*, 2010] A.H. Mohsenian-Rad, V.W.S. 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*, 1(3):320–331, 2010.

[Paetz *et al.*, 2011] A-G. Paetz, B. Becker, W. Fichtner, and H. Schmeck. Shifting electricity demand with smart home technologies - an experimental study on user acceptance. In *30th USAEE/IAEE North American Conference Online Proceedings*, 2011.

[Pyrko, 2006] J. Pyrko. Load demand pricing - case studies in residential buildings. In *Proceedings of the International Energy Efficiency in Domestic Appliances and Lighting Conference*, 2006.

[Ramchurn *et al.*, 2011] S.D. Ramchurn, P. Vytelingum, A. Rogers, and N. Jennings. Agent-based control for decentralised demand side management in the smart grid. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems - Innovative Applications Track (AAMAS)*, pages 5–12, 2011.

[Reddy and Veloso, 2012] P.P. Reddy and M.M. Veloso. Factored models for multiscale decision-making in smart grid customers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2012.

[Sanghvi, 1989] A. Sanghvi. Flexible strategies for load/demand management using dynamic pricing. *IEEE Transactions on Power Systems*, 4(1):83–93, 1989.

[Shinwari *et al.*, 2012] M. Shinwari, A. Youssef, and W. Hamouda. A water-filling based scheduling algorithm for the smart grid. *IEEE Transactions on Smart Grid*, 3(2):710–719, 2012.

[Voice *et al.*, 2011] T. Voice, P. Vytelingum, S.D. Ramchurn, and A. Rogers. Decentralised control of micro-storage in the smart grid. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2011.