Smart Home Energy Management Systems Based on Non-Intrusive Load Monitoring

Wenjin (Jason) Li, Xiaoqi Tan, Danny H.K. Tsang
Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong.
Email: {wliab@connect.ust.hk, xtanaa@ust.hk, eetsang@ece.ust.hk}

***Abstract*—The concept of smart home has attracted considerable attention in recent years, and energy management is one of its key component. This attributes to the growing concern in environmental protection and energy conservation, as well as the demands for big data collection from utility companies and policy makers. Current solutions often approach this problem by either the centralized non-intrusive load monitoring (NILM) or the decentralized smart controls but seldom both, rendering them impractical to some extent. Therefore, in this paper, we propose a novel framework of smart home energy management systems incorporating both approaches, so that accurate power consumption monitoring and intuitive interaction with the home appliances are simultaneously achieved. The smart components directly control the appliances, while the central controller coordinates the data collection and communication. The key feature is the capability of automatically mapping the appliances to their corresponding sockets, reducing the necessity for manual initial setup. Numerical simulations prove the accuracy and efficiency of the framework. We believe that our systems, if widely deployed, can benefit not only individual households by saving energy bills and simplifying life but also society by the big data generated.**

I. Introduction

Increasing emphasis on environmental protection and smarter energy uses nowadays calls for better energy management systems. For households this typically involves managing the power consumption of all appliances and controlling their operation states. Such smart home systems provide users with not only a more concrete illustration of their energy consumption behaviors but also the possibility to remotely or even automatically control their home appliances. These functions will eliminate the energy wastes and the potential risks introduced by the appliances running overtime. Moreover, utility companies can exploit the big data collected by the systems for better load forecasting and policy making, which has been a key motivation to push forward the deployment of smart meters [1].

Currently both centralized and decentralized approaches are applied in the implementations of such systems in the research prototypes or the commercial products. Specifically, the *centralized* approach utilizes non-intrusive load monitoring (NILM) to obtain individual energy consumptions, which are inferred from an aggregated power signal collected by a single smart meter at the main cable. This signal is simply the summation of all individual appliances' power consumptions, and it is usually stable except for occasional noises. Changes only happen along with a power event, such as switching an appliance on or off or changing its operation state. As a result, they are typically sparse in time, say, once per several minutes, and the power signatures that are unique to different types of appliances (e.g. transient noises, state transitions) can be retained, which are then used for load pattern recognition. Hart [2] first introduces this idea and uses the edges in real and reactive power signals as signatures. A number of studies in the next two decades extend this model by using other quantities such as jump in current [3], voltage distortion [4] and transient noise [5]. A more complete survey of the state-of-the-art NILM techniques can be accessed in [1]. Recently a startup company named Neurio [6] has released a commercial product to identify the usage pattern of five major home appliances and provide intelligent load shifting recommendations accordingly. NILM approaches are cost-effective by reducing the measurement and estimation to a single point. However, the accuracy is thus largely limited. Furthermore, purely centralized approaches do not have dedicated communication channels between the central controller and the individual appliances. As a result, even with extremely accurate load disaggregation, the control actions such as turning the appliances on or off still need to be manually carried out. This is no exception for the current commercial products like Neurio. In fact, providing recommendations already seems to be the best that it can do by itself, making it imperfect for practical use.

The *decentralized* approach involves two major categories of components. The first are the smart components, such as smart sockets and plugs, each dedicatedly connected to an appliance. The other is the central controller (e.g. a smart router) communicating with the smart components with short-range wireless communication techniques, such as Wi-Fi, ZigBee or Bluetooth Low Energy. The smart components usually have very simple structures to reduce costs, since an ordinary family would need dozens of them to connect to all appliances. This means that they seldom do exact measurement on current or power; instead they collect simpler information, such as the on-off states of the appliances connected. Besides the state sensor, each smart component also includes a relay switch to control its state, as well as a communication module, though different types of components may have their own specific elements depending on their functionalities. (E.g. a rolling blind control would have a motor to set the blind position.) The capability of remote control can be achieved through a Web or smartphone app interface as long as the central controller is connected to the Internet. It is worthwhile to point out that a smart socket essentially converts an existing *dumb* appliance to a *smart* one that is Internet-enabled, which saves cost compared to purchasing new smart devices especially for homes or organizations with limited budgets. One research prototype of this approach is nPlug [7]. It performs voltage and frequency measurement at plug level to infer peak periods and schedules load shifting accordingly. But it differs from its commercial counterparts in that each nPlug performs accurate measurement, which cannot scale easily even within a single household due to its heavy capital cost. On the other hand, SmartThings [8] and Lutron home control systems [9] are the representatives of the commercial products. They provide users with intuitive ways to interact with their daily home appliances such as lights, dimmers, washers, heaters, TV sets, computers and electronic door locks with integrated smart

978-1-4673-8289-2/15/$31.00 ©2015 IEEE

yet simple components. Facilitated by the two-way feedback control, real-time status of all connected sockets and switches can be accessed reliably. However, the shortcoming of them is mainly the lack of rich monitoring compared to the ones using NILM such as Neurio, reducing the incentive for many families to adopt them. Furthermore, they usually require a tedious initial setup process. Specifically, it is highly time-consuming to manually map all sockets to the appliances; otherwise the user interface only shows the socket IDs, which are unintuitive to control as users normally do not want to memorize the mappings. What makes things worse is that they are not intelligent enough to *adapt* to changing settings once the setup is done (e.g. an appliance being plugged into another socket), making the stored information easily outdated.

Almost all currently available research prototypes or commercial products in smart home energy management utilize either one of the two aforementioned approaches, with its own disadvantages that hinder the further development and wider deployment. We believe that incorporating both approaches into the design and implementation of such systems can avoid their respective shortcomings and provide users with a better experience in automating their home. Therefore, in this paper we propose a new framework of smart home energy management systems based on both centralized NILM and distributed controls. In particular, we are motivated to tackle the lack of control channels in the former approach, and to enhance the richness of monitoring and the intelligence of self-configuration and adaption in the latter. We solve these problems by introducing the capability of automatically mapping the appliances and the connected sockets, without the necessity for the appliances to report their identities or for the sockets to perform sophisticated measurements.[1]

This paper is organized as follows. Section II explains our system design, which is further divided into three parts – the smart components, the central controller and the user interface. The automatic mapping algorithm is described in detail in this section. The setup of the experiment to evaluate the feasibility and performance of our prototype is in Section III, and the results of the numerical simulation and the discussion are in Section IV. Finally we conclude our paper and propose some future research possibilities in Section V.

II. System Design

Fig. 1 illustrates the overall structure of our system design. The solid arrows in the diagram indicate the directions of the current flow. The components attached with antennas are connected to the domestic WLAN, while those linked through dashed lines are connected to the Internet. We will elaborate on each of the three main parts below.

A. Smart Components

These are at the bottom level of our system, connected to the individual appliances. There are different types of them, including but not limited to sockets, switches and rolling blind controls, so as to serve different uses. Nevertheless, all of them share the following three common modules described below, from bottom to top as illustrated in Fig. 1.

[1]Actually the idea of automatic mapping is not limited to a domestic home setting. It may also be applied to other scenarios, such as deferrable load shifting in smart grid and electric vehicle (EV) charging. See Section II-B3 for more detail.

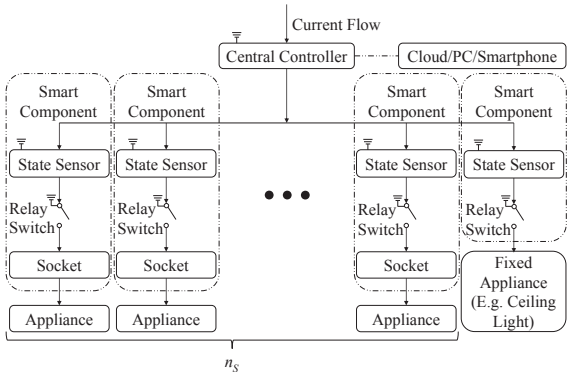

Fig. 1: Smart Home Energy Management System Diagram

1) Relay Switch: This changes the on-off state of the smart component based on the control signals received. Meanwhile the physical switch may still remain for backup purpose in the case of a failure of the networked control.

2) State Sensor: This senses whether the appliance connected to it is operating or not. Although in principle we can monitor the state of the relay switch for simplicity, this does not necessarily indicate that of the appliance. For instance, a user may forget to turn off the relay switch after manually shutting the appliance down, or the appliance may have designed its own ways of control (e.g. an air-conditioner has continuous temperature, speed and timer settings) so the user would be better off leaving the relay switch always on to avoid being limited to on-off only. Moreover, considerable appliances have standby states that consume much lower power than the normal operating ones, which are not regarded as being on by many NILM algorithms. As the appliance's true operation state is critical to the automatic system configuration described in the later sections, we need to dedicate a sensor to it. Since the exact amount of current or power is not to be measured, a simple threshold checking device would be adequate for the decision making. By doing so, the total cost for the entire system would not increase much.

3) Communication Module: This communicates with the central controller via short-range wireless communication. The states of the relay switch and the state sensor are reported to the central controller. It also listens to the control signals sent from there for execution.

B. Central Controller

This is the coordinator of the whole system. It lies in the entry point of the entire home circuit, and coordinates all data collection and processing, part of which is carried out by itself. It has the following five major functions in chronological order.

1) Collecting state information from all smart components: As mentioned previously, it listens to the reported state information from all smart components installed.

2) Carrying out NILM: At the main cable of the house's fuse box, a power sensor is deployed to collect the aggregated power signal. The power signatures that are unique to different types of appliances are then extracted for disaggregation of a NILM algorithm. The majority of the numerous state-of-the-art algorithms can generate a global signature database that applies to most households, thus saving the effort of manual initial training. Specifically, the disaggregation usually follows a similar format of taking in the aggregated power signal and outputting the power consumption of each recognizable

appliance over time, so efforts have been made recently [10] [11] to fit various algorithms to a unified interface of an evaluation framework. As a result, it is worth pointing out that our proposed system does not rely on a particular NILM algorithm. Instead, it can switch flexibly among a variety of them to find the one with the best performance.

3) Pairing up the appliances and the sockets: Considering that most existing home appliances do not have a unified web control interface, it is more practical to remotely switch their operation states via the smart sockets connected. Therefore knowing the mapping between the appliances and the sockets enables us to control them more intuitively. (Note that fixed appliances such as ceiling lights and fans are directly controllable through the IDs of their fixed control components, so they do not have to undergo this process.) Unfortunately, as have been previously mentioned, the current smart home solutions do not produce the mapping automatically due to the fact that the smart sockets alone cannot identify the appliance type. However, as it will become clear later on, in our proposed system it is possible with the help of NILM.

The flow chart in Fig. 2 illustrates the following automatic mapping process. After successfully completing the NILM process, a list of all recognizable appliances in the household and their detailed power consumption profiles should be produced. For the purpose of automatic mapping, we just need to know when each appliance is in its normal operating (on) state, excluding the standby mode, if any. As discussed previously, the state information can be acquired by a simple threshold checking. Note that the threshold may not be identical for different appliances. Assume that there are totally n_A recognized appliances (excluding fixed ones like ceiling lights) over N sampling points. For the i-th appliance where $i \in \{1, 2, ..., n_A\}$, we denote its binary on-off state at the t-th sample ($t \in \{1, 2, ..., N\}$) as

$$A_i(t) = \begin{cases} 1 & \text{if the appliance is on,} \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

At the same time, we have also got the socket-level operation state information. Assume that we have n_S sockets ($n_S \geq n_A$ as there are usually some spare sockets and some connected to unrecognized appliances), and that each socket is connected to only one appliance. (Even if a power bar is to be used, we can use a "smart" power bar and ignore the socket which its cable is plugged into. In other words, each socket on the power bar is simply regarded as an individual smart socket with its own monitoring devices.) For the j-th socket, similar to the case of the appliances, we denote its state at the t-th sample as

$$S_j(t) = \begin{cases} 1 & \text{if the appliance connected is on,} \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Again, the on state is exclusive of the standby mode, if any. It is worth noting that for both groups of data, the conversion to the binary format is just for simplicity, and the mapping capability indeed is not limited to binary-state appliances.

Now we want to find out the optimal mapping between the appliances and the smart sockets. One way to define the optimum is as follows. Denote $\mathcal{M}$ as the set of all possible mappings from the 1st, 2nd, ..., n_A-th appliances to the m_1-th, m_2-th, ..., m_{n_A}-th sockets, that is to say,

$$\begin{aligned} \mathcal{M} = \{\mathbf{m} = (m_1, m_2, ..., m_{n_A}) | \\ m_k \in \{1, 2, ..., n_S\}\ \forall k \in \{1, 2, ..., n_A\}, \\ m_p \neq m_q\ \forall p, q \in \{1, 2, ..., n_A\} \text{ and } p \neq q\}. \end{aligned} \tag{3}$$

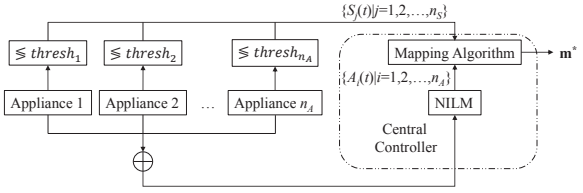

Fig. 2: Mapping Flow Chart

In this way, each $\mathbf{m}$ essentially represents a possible permutation of picking n_A unique integers from $\{1, 2, ..., n_S\}$. Define the total squared error between the *on* time of each appliance and that of its corresponding socket as

$$E(\mathbf{m}, N) = \sum_{t=1}^{N} \sum_{i=1}^{n_A} (A_i(t) - S_{m_i}(t))^2, \tag{4}$$

so $E(\cdot)$ is a summation of binary values that measure whether $A_i(t)$ deviates from $S_j(t)$ at each sample t. Then the best mapping $\mathbf{m}^* = (m_1^*, m_2^*, ..., m_{n_A}^*)$ minimizes this $E(\cdot)$:

$$\mathbf{m}^* = \arg \min_{\mathbf{m} \in \mathcal{M}} E(\mathbf{m}, N). \tag{5}$$

As this approach minimizes the total error, we may call it the "global" algorithm.

Since there are $\frac{n_S!}{(n_S - n_A)!}$ possible $\mathbf{m}$'s in $\mathcal{M}$, we may immediately come up with a naive algorithm to exhaustively iterate through the entire set and find the optimal mapping that gives the minimum $E(\cdot)$. This process is of complexity $\mathcal{O}[n_S! N]$. As a result, the long computation time would make it of no practical use. In fact, after a close inspection of the formulation, we may write the total squared error between each pair of appliance and socket in the matrix form as

$$E = [e_{ij}]_{n_A \times n_S} = \left[\sum_{t=1}^{N} (A_i(t) - S_j(t))^2 \right]_{n_A \times n_S}. \tag{6}$$

We also write each possible $\mathbf{m}$ in $\mathcal{M}$ as

$$M = [m_{ij}]_{n_A \times n_S} \tag{7}$$

where

$$m_{ij} = \begin{cases} 1 & \text{if } m_i = j, \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

Apparently, there must be exactly one 1 for each row of M, and exactly n_A columns have one 1. All of the rest entries are 0. Each $\mathbf{m}$ produces a different matrix M, so the problem may still be solved exhaustively. However, it can also be expressed as a standard linear optimization problem:

$$\min_{M} \quad \sum_{i=1}^{n_A} \sum_{j=1}^{n_S} e_{ij} m_{ij} \tag{9}$$

$$\text{s.t.} \quad \sum_{j=1}^{n_S} m_{ij} = 1\ \ \forall i = 1, 2, ..., n_A, \tag{10}$$

$$\sum_{i=1}^{n_A} m_{ij} \leq 1\ \ \forall j = 1, 2, ..., n_S, \tag{11}$$

$$m_{ij} \geq 0\ \ \forall i = 1, 2, ..., n_A \text{ and } j = 1, 2, ..., n_S. \tag{12}$$

This indeed resembles a typical bipartite matching or optimal assignment problem [12], where E is the cost matrix and M is the assignment matrix. Since each m_{ij} can only take 0-or-1 values, M is totally unimodular. So there is always an optimal integer solution, which can be obtained efficiently in polynomial time by, for instance, the Hungarian algorithm [13].

However, a global mapping algorithm may not be the best. Consider the state sequences of both the appliances and the sockets as points in an N-dimensional hyperspace. The global mapping essentially assigns one socket to each appliance to minimize the total Euclidean distance. In reality, one socket can simultaneously be the nearest neighbors to multiple appliances in this space. If it is taken up by one appliance, the other appliances may have to seek their subsequent nearest neighbors, which may lie much farther away. In order to minimize the total distance over all appliances, the first appliance may be better off with a suboptimal (which may be a little farther than its true optimal), so that the remaining appliances happily get their true optimal. Currently any NILM algorithm has its own strengths and weaknesses in disaggregating certain types of appliances, so this problem widely exists, which necessitates an alternative way to look at the task.

We may consider to change the objective to minimizing the Euclidean distance for *each* appliance and its corresponding socket whenever possible. Specifically, the algorithm flow is explained below in Algorithm 1. In this context, we call a mapping for an appliance as *unique* iff. the appliance is mapped to a socket that is not mapped to any other appliance. It is trivial that the computational complexity is $\mathcal{O}[n_A n_S N]$, which is linear to the data size. This can be called the "greedy" algorithm as each appliance greedily pick the socket nearest itself without considering the total error over the system. This can solve the problem of the global one.

Algorithm 1 Greedy Mapping Algorithm

Input: $\{A_i(t)|i = 1, 2, ..., n_A\}, \{S_j(t)|j = 1, 2, ..., n_S\}$ $(t = 1, 2, ..., N)$
Output: $\mathbf{m}^* \in \mathcal{M}$

```
find the nearest neighboring socket for each appliance
fix the mappings that are already unique
while at least one socket is assigned to multiple appliances do
    find the next nearest neighboring socket for each remaining appliances among the unassigned sockets
    fix the mappings that are already unique
end while
```

There are also a few practical concerns when applying the automatic mapping algorithms. Note that N will be very large due to the frequent sampling, which also significantly affects the computational complexity. Meanwhile, the mapping is prone to change over time. In reality, the mobile appliances are usually the chargers for laptops, smartphones, tablet PCs and so on. The users normally take them away to use outside in the day and bring back home to charge at night. When charging, they tend to simply find any empty sockets to plug them in. The samples taken more than 24 hours ago thus would not help much in deciding the current mapping, and N only has to include the samples taken within one day. Alternatively, the contributions of the historical samples can be scaled down with some weights in error calculation. A second way to further simplify the process is to exploit the results from the previous executions in assisting the current decision making, in the sense that the past mapping does not need updating if there is no significant increase in error using it for the current sample.

Therefore, with the aforementioned mapping algorithms, we have solved the problem of lacking the capability of self-configuration and adaptive intelligence in many of today's smart home energy management solutions. The idea of automatic mapping is not only useful in the smart home setting. It can be applied similarly to some other scenarios as well. For example, in the charging scheduling of multiple EVs within a microgrid, different models may have their own charging constraints, so when they are plugged into the sockets, the scheduling decisions have to be sent to each individual socket. Considering the fact that most of the charging sockets are wall-mounted, it is thus important to know which socket corresponds to which EV so as to perform a specifically objective-oriented charging scheduling (e.g. peak shaving, providing ancillary services, etc.). With our proposed automatic mapping concept, the system operator learns the exact mapping automatically, and thus extra communication between the individual EVs and the system operator is eliminated.

4) Receiving control commands and forwarding them to the smart components for execution: The users can send commands to the central controller through the user interface. In addition, automatic control decisions may also be generated based on the usage statistics. These are forwarded to the smart components in concern for them to perform the desired actions.

5) Advanced applications: Depending on the users' needs, more advanced applications can be built on top of the proposed framework. For example, the central controller can learn the usage pattern of different appliances and provide load shifting suggestions according to the peak periods and electricity tariff fluctuation. And once it detects some abnormal behaviors such as any appliance does not stop after finishing its job, it can send a warning message to the user's smartphone and trigger some control actions such as automatically switching off the smart socket to which the appliance is connected. Certainly as an embedded system, the central controller may not be powerful enough at all times, so it can also send the data to the cloud for more complicated processing. The cloud, after gathering big data from the widely deployed systems in millions of homes, may perform something more meaningful than merely serving for individuals as well, such as load prediction for the grid or assisting in distribution policy making.

C. User Interface

This is what the users directly interact with. It can be a web or smartphone app interface for use on different end-devices. All controllable appliances are displayed, as well as anything else that the system wants the user to know such as the power consumption breakdown diagram and the emergency warnings for the abnormal appliance behaviors. The users can control the appliances with one click and observe instant feedback. Any automatic control action should also produce readable results that pop up on the users' devices. Since the hardware and software implementation is still ongoing, we thus leave the detailed introduction of the UI for future work.

III. Numerical Simulation Design

Our framework can freely pick any well-known NILM algorithms, which have been well evaluated in their respective literatures. As the main contribution of this paper is the proposed automatic mapping algorithms, we are going to evaluate this part only. And once this is done, the feasibility of the system as a whole is trivial.

A. Simulation Setup

We implement the algorithms on MATLAB and evaluate the framework of our smart home energy management systems by conducting numerical simulation on the NILM-Eval framework [10] against the ECO (Electricity Consumption and Occupancy) data set [14]. The NILM-Eval framework is designed in the way that we can plug in any NILM algorithm and any smart home data set, as long as they fit in the provided interface, so as to compare the performance of different NILM algorithms. It is also perfect for our purpose since we expect the NILM outputs to be in simple readable formats, which is not always achieved by every NILM algorithms available but guaranteed by NILM-Eval.

The NILM algorithm we use is the Weiss' algorithm [15]. It uses the nearest neighbor classifier for the power signatures. In its literature, the parameter r determines the required maximum distance between the detected power event and its assigned pattern. The smaller r is, the stricter the maximum variation requirement is to assign a detected power event to an appliance pattern. Therefore altering r produces different levels of disaggregation accuracy and enables us to see the mapping performance with regard to that of NILM.

The ECO data set is a comprehensive data set for NILM research, containing 1Hz aggregate consumption and plug-level data from 6 Swiss households over 8 months. Among them, Household 2 has the most complete socket-level information, which is useful for setting the ground truth, so we use this household to set up our experiment data. In order to smooth out the fluctuation due to random errors and occasionally missing data while maintaining a reasonable data size, we use the data from the first 90 days instead of from a few days only. This home has 12 appliances according to the description of the ECO data set, 9 of which are successfully recognized by Weiss' algorithm in the original literature. In order to normalize the data, the appliances that always operate simultaneously with some others[2] or are not connected to any sockets[3] are eliminated, such as the TV and the air exhaust. On the other hand, those appliances that do not contribute much to the overall power consumption are omitted from the evaluation results.[4] Eventually, the fridge, the freezer and the dishwasher remain.

B. Simulation Steps

1) Prepare Evaluation Data: We use only the aggregated power signal to conduct one of the default experiments defined in NILM-Eval, which is to run the Weiss' Algorithm on the first 90 days of Household 2 of the ECO data set with different settings of r. The disaggregation results are stored in several files on the disk, which contain the time stamps when the power events for each appliance occur. They are converted to binary on-off states for all sampling points. When there is a rising edge in power at a certain time for an appliance, that appliance will stay in the on state from that time on until a falling edge is detected. For the socket-level (or plug-level as in the original ECO literature) data, we pass them through a threshold checker to obtain the binary states. The thresholds are as defined in the data set literature and are different across appliance types. Both sides of the data in the original data set have a small portion of time missing. We simply consider these time periods as remaining in the previous states until the data become available again. As the first 15 days are used as the training days as defined by the setup file, mapping is to be performed on the remaining 75 days only.

2) Perform Automatic Mapping: The first 15 days of training data provides the required signature database to the Weiss' Algorithm. Based on our assumption that data older than one day are no longer useful for mapping, the database does not convey any information of the mapping. For the rest 75 days, we hide the names of the sockets in the socket-level monitoring data and provide them to the mapper along with the NILM results. The mapper then tries to find each appliance's corresponding socket according to the algorithm described in Section II-B3. The objective is to find the appliance to which each socket is mapped, and to see whether the appliance's name is identical to the socket's name in the ECO data set. In this way the real mapping is simulated. The performance is evaluated by the accuracy of each appliance, defined as the empirical probability of it being mapped correctly to its socket.

IV. Results and Discussion

A. Global and Greedy Algorithms

In Section II-B3 we proposed two mapping algorithms. Here we execute them under two different r settings and compare their accuracies in order to find the more appropriate one. To reduce the data size, the original data are down-sampled to $\frac{1}{900}$Hz. This is in consideration of the fact that the occurrences for the appliances to move and for the smart components to report their states are sparse in time (to save power). Furthermore, the socket for the tablet PC, the undisaggregated appliance by Weiss', is ignored at this time. Looking at the results of the three appliances shown in Table I, the greedy algorithm outperforms the global one with reduced complexity, which is the expected behavior for real data. Therefore, in the remaining simulations we will use the greedy algorithm only.

B. Accuracies vs. r

The greedy algorithm is used and the other settings are identical to the previous subsection. The accuracies are shown in Table II. They become relatively stable when $r \geq 0.5$, which is also when the NILM performance stabilizes as shown in the results of [10].

TABLE I: Mapping Accuracies w.r.t. Selected Algorithm

r	0.1		1	
Algorithm	Global	Greedy	Global	Greedy
Fridge	70.67%	100.00%	100.00%	100.00%
Freezer	41.33%	100.00%	73.33%	100.00%
Dishwasher	54.67%	88.00%	82.67%	90.67%

[2]In practice, we can always do batch controls for them.

[3]Although their power consumptions can be disaggregated, they have dedicated connections to the home circuit which is directly controllable, so mapping is not necessary.

[4]Most NILM algorithms are not yet capable of disaggregating these power consumptions, posing unnecessary difficulty on the mapping stage since knowing their correspondences does not provide much additional gain to energy saving. Nevertheless, their data still exist in the evaluation to simulate real situations, though their results are omitted.

TABLE II: Mapping Accuracies *vs.* r

r	0.05	0.1	0.2	0.3	0.4	0.5
Fridge	97.33%	100.00%	100.00%	100.00%	100.00%	100.00%
Freezer	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
Dishwasher	62.67%	88.00%	92.00%	92.00%	90.67%	90.67%
r	0.6	0.7	0.8	0.9	1.0	2.0
Fridge	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
Freezer	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
Dishwasher	90.67%	90.67%	90.67%	90.67%	90.67%	90.67%

TABLE III: Mapping Accuracies *vs.* Sampling Interval

Sampling Interval (s)	1	15	60	120	300	900
Fridge	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
Freezer	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
Dishwasher	92.00%	92.00%	92.00%	92.00%	92.00%	90.67%

TABLE IV: Mapping Accuracies With Extra Socket

r	0.1		1	
With Extra Socket?	No	Yes	No	Yes
Fridge	100.00%	100.00%	100.00%	100.00%
Freezer	100.00%	100.00%	100.00%	100.00%
Dishwasher	88.00%	17.33%	90.67%	20.00%

C. Accuracies vs. Sampling Interval

To see whether downsampling has any effect on mapping accuracies, we fix $r = 1$ and still do not consider the socket for the tablet PC. The accuracies of all appliances under different sampling intervals are displayed in Table III, which shows that most of them do not change much after downsampling. As a result, we can freely use the method to reduce data size and accelerate computation. The practical meaning is that our framework does not deteriorate under low reporting rates of the smart components.

D. Accuracies With Existence of Extra Sockets

In reality, there are usually some spare sockets on the wall that are not occupied by any appliance as well as some connected to the unrecognized appliances. In our data set, the socket for the tablet PC essentially falls into this category, so we examined the accuracies before and after including this extra socket in mapping, with the results shown in Table IV. The fridge and the freezer remain having similar mapping accuracies for both of the chosen r values, but the dishwasher experiences significant drop. This indicates that taking extra sockets into consideration tends to increase confusion between the appliances that are worse disaggregated. It is thus highly important to recognize the extra sockets and remove them before passing the data to the mapper. Alternatively, some mechanism can be set up to verify the correctness of mapping so as to avoid any consequence introduced by mismatching. For instance, the central controller may ping all sockets and check whether their reported states match the real-time NILM result, or the system may allow for manual calibration.

V. Conclusion

We have presented a novel framework of smart home energy management systems utilizing both centralized NILM and distributed controls to exploit the advantages of both approaches while avoiding their disadvantages. The key contribution is its capability of automatically mapping the appliances to sockets, reducing the burden of initial system setup. The framework proves to be accurate for the major home appliances by the numerical simulation on a smart home data set, ECO, though with space for improvement with regard to low-power appliances. As a result, deploying our systems widely will save energy and simplify interaction with the appliances for individual households, and benefit society by the economy of scale and the big data collected. The idea of automatic mapping can also find its use in other areas like deferrable load shifting in EV charging. Future research directions could include improving accuracy and robustness of the mapping algorithm by filtering the raw power signals, incorporating more sensor data (e.g. impedance) in NILM, implementing a real hardware prototype of our proposed framework on some industry-standard embedded system platform such as Raspberry Pi, etc.

References

[1] A. Zoha *et al.*, "Non-intrusive load monitoring approaches for disaggregated energy sensing: A survey," *Sensors,* vol. 12, no. 12, pp. 16838–16866, Dec. 2012.

[2] G. W. Hart, "Nonintrusive appliance load monitoring," *Proc. IEEE,* vol. 80, no. 12, pp. 1870–1891, Dec. 1992.

[3] F. Sultanem, "Using appliance signatures for monitoring residential loads at meter panel level," *IEEE Trans. Power Del.,* vol. 6, no. 4, pp. 1380–1385, Oct. 1991.

[4] J. Roos *et al.*, "Using neural networks for non-intrusive monitoring of industrial electrical loads," in *Instrumentation and Measurement Technology Conf.,* 1994, pp. 1115–1118.

[5] S. N. Patel *et al.*, "At the flick of a switch: Detecting and classifying unique electrical events on the residential power line," in *Proc. 9th Int. Conf. Ubiquitous Computing (Ubicomp '07), ACM,* 2007, pp. 271–288.

[6] Neurio Technology, Inc. (2015). *Neurio Home Intelligence™* [Online]. Available: https://www.neur.io/

[7] T. Ganu *et al.*, "nPlug: A smart plug for alleviating peak loads," in *Proc. 3rd Int. Conf. Future Energy Syst.: Where Energy, Computing and Communication Meet, ACM,* 2012, pp. 1–10.

[8] SmartThings, Inc. (2015). *SmartThings — Life Like Never Before* [Online]. Available: http://www.smartthings.com/

[9] Lutron Electronics Co., Inc. (2015). *Lutron Electronics, Inc. - Dimmers And Lighting Controls* [Online]. Available: http://www.lutron.com/en-US/Pages/default.aspx

[10] R. Cicchetti, "NILM-Eval: Disaggregation of real-world electricity consumption data," Master's thesis, ETH Zurich, 2014.

[11] N. Batra *et al.*, "NILMTK: An open source toolkit for non-intrusive load monitoring," in *Proc. 5th Int. Conf. Future Energy Syst., ACM,* 2014, pp. 265–276.

[12] G. Caron *et al.*, "The assignment problem with seniority and job priority constraints," *Operations Research,* vol. 47, no. 3, pp. 449–454, Jun. 1999.

[13] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics Quart.,* vol. 2, no. 1–2, pp. 83–97, Mar. 1955.

[14] C. Beckel *et al.*, "The ECO data set and the performance of non-intrusive load monitoring algorithms," in *Proc. 1st ACM Int. Conf. Embedded Sys. Energy-Efficient Buildings (BuildSys 2014),* Memphis, TN, USA, 2014, pp. 80–89.

[15] M. Weiss *et al.*, "Leveraging smart meter data to recognize home appliances," in *2012 IEEE Int. Conf. Pervasive Computing and Communications (PerCom),* 2012, pp. 190–197.