Distributed Demand-Side Management in Smart Grid: how Imitation improves Power Scheduling

Antimo Barbato[†], Antonio Capone [†], Lin Chen [‡], Fabio Martignon [‡¶] and Stefano Paris [§‖]
[†] DEIB
Politecnico di Milano
{antimo.barbato,antonio.capone}@polimi.it
[‡] LRI
Paris-Sud University
{lin.chen,fabio.martignon}@lri.fr
[¶] Institut Universitaire de France (IUF)
[§] Mathematical and Algorithmic Sciences Lab
[‖] France Research Center - Huawei Technologies Co. Ltd.
LIPADE - Paris-Descartes University
stefano.paris@{huawei.com,parisdescartes.fr}

***Abstract*—Demand-Side Management (DSM) systems represent an efficient method to improve the performance of Smart Grid infrastructures by controlling users' power loads. In this paper, we focus our analysis on *fully distributed* DSM systems especially designed to reduce the peak demand of groups of residential users. In our proposed scheme, each appliance decides autonomously its scheduling using only limited information on the energy price fixed by the retailer, thus greatly reducing the system complexity as well as the need of information exchanges. We develop two schedule-selection policies based on the Proportional Imitation Rule, where at each iteration all appliances switch to a new schedule with a probability proportional to the cost difference between the actual and cheapest schedules of the previous iteration. We analyze the proposed learning methods based on realistic instances in several use-case scenarios, and show their effectiveness in terms of cost reductions (both local and system-wide) as well as convergence speed to stable and efficient system equilibria.**

I. Introduction

The design of power grids has been historically driven by the need to meet the peak demand of users, which is largely uncontrollable. As a consequence, grid resources are under-utilized for most of the time. In order to address this issue, Demand-Side Management (DSM) methods can be used, since they are designed to properly control and schedule users' loads [1]. Specifically, DSM solutions can be applied to shift users' demand from peak to off-peak periods, thus reducing the need for generation, transmission and distribution capacity, as well as power grids investments. Similarly, DSM systems can also be employed to mitigate other issues related to electric grids such as the integration of Renewable Energy Sources (RESs), which are intermittent and still largely uncontrollable, hence creating problems in the demand-supply balancing process. To this end, DSM systems can be used to properly scheduling users' loads based on the availability of renewable energy [2].

In the field of DSM systems, *distributed* methods have gained increased momentum. Relevant improvements of the grid efficiency and performance can indeed be obtained only by coherently managing the energy resources of groups of users whose differences and randomness, in terms of electricity consumption needs, can be exploited to adapt the overall load demand to the grid requirements. To this end, several centralized frameworks have been proposed in the literature, aiming at controlling the electric loads of groups of collaborative customers [3]. However, these solutions require a centralized controller to gather users' information and optimize their energy plans. As a consequence, a large volume of data must be collected and transmitted through the Smart Grid network, thus introducing scalability constraints, as well as novel threats to customers' security and privacy [4]. For these reasons, distributed DSM methods have been proposed in which decisions are taken *locally* by users. In such context, Game Theory represents the ideal framework to design distributed DSM solutions, since it permits to model and study the interactions among the independent rational players of the power grid [5]. In this case, the users' load scheduling problem is formulated as a game, where *players* are the consumers and their *strategies* are the schedules of their electric appliances. The goal of the game is to reduce the peak of the total demand, the overall energy costs, or users' electricity bills [6].

Game theoretic DSM methods are designed to drive the system to equilibria that improve the performance of the power grid from a system-wide perspective. However, converging to the game equilibria is a non-trivial challenge and *learning algorithms* are required to enable players to reach the desired game outcome [7]. Learning methods are usually iterative processes in which players, in turn, estimate the utility associated with their strategies based on their knowledge of the game state, and decide which strategy to play in the current iteration depending on the decision logic of the algorithm. Several learning algorithms have been proposed in the literature which differ in the learning style and in the assumptions on the interaction among players. In the *Regret Matching* methods [8], for example, players attempt to minimize their regret from using a certain strategy. These methods rely on the assumptions that each player is able to estimate both its own utility and the one he would have obtained by playing all other actions. On the other hand, in *Reinforcement Learning* methods [7], players attempt to maximize their utility rather than considering the regret associated with their actions. Specifically, at each iteration of the algorithm, actions leading to higher utilities are associated with higher probabilities to be chosen in the next stage. *Regret Matching* and *Reinforcement Learning* methods, as well as numerous other algorithms, have been extensively studied in several research fields, including robotics and telecommunications [9]. For this reason, some of the solutions proposed in these fields may be applied to game theoretic DSM frameworks. However, security and privacy

concerns may raise when applying these methods to real implementations of demand management solutions. Learning algorithms proposed in [6] and [10], for example, require each player to broadcast his appliances schedule to either the energy service provider or to other users, thus introducing serious privacy issues [4].

In this paper, we propose a distributed learning algorithm which enables consumers to autonomously converge to the equilibria of DSM load scheduling games. Our algorithm is particularly tailored for demand-side management frameworks used to schedule the electric devices of residential consumers on a daily basis in a distributed fashion, with the goal of minimizing their bills, thus improving their Quality of Experience. In this context, a dynamic pricing is used, where energy tariffs are defined as a function of the *overall power demand* of users.

In the learning algorithm proposed in this paper, which is based on a *reinforcement learning* approach, the scheduling decision problem of each player is modeled as a Markov chain: each feasible appliance schedule is associated with a state of the chain, and state transition probabilities are updated at every iteration, depending on the bills of players. In particular, we develop two imitation-based schedule-selection policies based on the Proportional Imitation Rule [11], where at each iteration, all appliances switch to a new schedule with a probability proportional to the *cost difference* between the actual and cheapest schedules of the previous iteration.

We analyze the performance of the proposed learning methods based on realistic instances of the DSM load scheduling game, and numerically show its effectiveness in converging, in few iterations, to the system equilibria in several use-cases.

The remainder of the paper is structured as follows: Section II provides an overview of the game theoretic demand-side management framework that we have adopted in this work. Section III describes the learning algorithms that we have designed to converge to the Nash Equilibrium of the game. Performance assessment is provided in Section IV and, finally, conclusions are drawn in Section V.

II. Distributed Demand Management Game

In this work, we consider a demand-side management framework, based on a non-cooperative game theoretical approach [12], which is used to efficiently schedule electric devices in a distributed fashion. Table I summarizes the notation used in this paper. Specifically, each user h of a group of residential consumers, $\mathcal{H}$, has a set of appliances, $\mathcal{A}$, that have to be scheduled over a 24-hour time period divided into a set, $\mathcal{T}$, of time slots. Each appliance a of user h must be executed only once during the day within a time window delimited by a minimum starting-time slot, ST_{ah}, and a maximum ending-time slot, ET_{ah}. Moreover, each device is characterized by a load profile, l_{ahf}, having a duration of F_{ah} time slots, with l_{ahf} representing the power consumption of a in the fth time slot of its load profile and $f \in \mathcal{F}_{ah} = \{1, 2.., F_{ah}\}$. The objective of each user is to minimize his daily bill. A dynamic pricing is used to define the price of electricity at time $t \in \mathcal{T}$,

TABLE I
Summary of the notation used in this paper.

$\mathcal{A}$	Set of appliances (a: an appliance)
$\mathcal{F}_{ah}$	Set of phases of appliance a of user h (f_{ah}: a phase)
$\mathcal{H}$	Set of residential consumers (h: a consumer)
$\mathcal{I}$	Set of strategies of all players (i.e., $\mathcal{I} \triangleq \{\mathcal{I}_n\}_{n\in\mathcal{N}}$)
$\mathcal{I}_n$	Set of strategies of player n (i.e., $\mathcal{I}_n \triangleq \{x_{nt}\}_{n\in\mathcal{N}}$)
$\mathcal{K}$	Set of iterations of the learning process (k: an iteration)
$\mathcal{N} = \mathcal{A} \times \mathcal{H}$	Set of players (n: a player, i.e. an appliance of a user)
$\mathcal{S}_n$	Set of states of player n (s: a state)
$\mathcal{T}$	Set of time slots (t: a time slot)
$\mathcal{U}$	Set of utilities of all players (i.e., $\mathcal{U} \triangleq \{U_n\}_{n\in\mathcal{N}}$)
c_t	Cost function
c^{Anc}	Cost of ancillary services
c^{En}	Slope of the cost function
ET_{ah}	Maximum ending slot for appliance a of user h
F_{ah}	Duration of the load profile of appliance a of user h
l_{aht}	Power consumption of appliance a of user h in slot t
ST_{ah}	Minimum starting slot for appliance a of user h
U_n	Utility of player n
x_{nt}	Binary variable (set to 1 if appliance n starts in slot t)
y_t	Total power demand of all users at time slot t
y_{nt}	Power demand of player n at time slot t
π^{SL}	Supply limit defined by the retailer

c_t, which is modelled as an increasing function of the total power demand, y_t, of the group of users $\mathcal{H}$ at time t:

$$c_t(y_t) = c^{Anc} + c^{En} \cdot y_t \qquad \forall t \in \mathcal{T} \tag{1}$$

where c^{Anc} is the cost of ancillary services (e.g., electricity transport, distribution and dispatching, frequency regulation, power balance) and c^{En} is the slope of the cost function.

In [12], we show that if each appliance decides *autonomously* its scheduling in a fully distributed fashion (*Single-Appliance DSM*) with the goal of minimizing its bill, only a negligible increase of the users' bill is found with respect to the case in which each user schedules the whole set of his home appliances (*Multiple-Appliance DSM*). For this reason, in this paper, we will use the *Single-Appliance DSM* model since it requires a less complex architecture. In this case, in fact, there is no need for a home server to collect all appliances information and play on behalf of the householder, thus greatly simplifying the architecture design and system configuration.

A. Single-Appliance Game Formulation

The appliance scheduling problem is modelled as a game $G = \{\mathcal{N}, \mathcal{I}, \mathcal{U}\}$: $\mathcal{N} = \mathcal{A} \times \mathcal{H}$ is the players set (player $n = (a, h)$ is the appliance a of consumer h), $\mathcal{I} \triangleq \{\mathcal{I}_n\}_{n\in\mathcal{N}}$ is the set of strategies which correspond to the appliances scheduling and $\mathcal{U} \triangleq \{U_n\}_{n\in\mathcal{N}}$ is the set of utility functions that coincide with the devices electricity bills. Specifically, the strategy of player n is $\mathcal{I}_n \triangleq \{x_{nt}\}_{n\in\mathcal{N}}$, where x_{nt} are binary variables defined for each activity $n \in \mathcal{N}$ and for each time slot $t \in \mathcal{T}$. These variables are equal to 1 if appliance n starts in time slot t, 0 otherwise. Each appliance (player) n chooses its strategy $\mathcal{I}_n$ to minimize its cost U_n. The utility function of each player, U_n, is defined as a function of $\mathcal{I}$:

$$U_n(\mathcal{I}) = \sum_{t\in\mathcal{T}} y_{nt} \cdot c_t(y_t) \tag{2}$$

where y_{nt}, which represents the amount of electricity demand of appliance n at time t, is a function of x_{nt} and c_t (the electric energy price at time t) is a function of y_t, which corresponds to the total power demand of players at time t.

Mathematically, the power scheduling game is formalized as follows:

$$G: \min_{\mathcal{I}_n} U_n(\mathcal{I}_n, \mathcal{I}_{-n}) = \sum_{t \in \mathcal{T}} y_{nt} \cdot c_t(y_t), \ \forall n \in \mathcal{N}. \tag{3}$$

The solution of the power scheduling game is characterized by a Nash Equilibrium (NE), a strategy profile $\mathcal{I}^* = (\mathcal{I}_n^*, \mathcal{I}_{-n}^*)$ from which no player has an incentive to deviate unilaterally. The feasible power scheduling alternatives that form the strategy space $\mathcal{I}_n$ of each player $n = (a, h)$ (i.e., each appliance a of consumer h) must satisfy the following set of constraints:

$$\mathcal{I}_n = \Bigg\{ \overrightarrow{x}_n = [x_{n1}...x_{nt}...x_{n|\mathcal{T}|}] \in \{0,1\}^{|\mathcal{T}|} :$$

$$\sum_{t=ST_n}^{ET_n - F_n + 1} x_{nt} = 1 \tag{4}$$

$$y_{nt} = \sum_{f \in \mathcal{F}: f \leq t} l_{nf} x_{n(t-f+1)} \qquad \forall t \in \mathcal{T} \tag{5}$$

$$\sum_{a \in \mathcal{A}} \sum_{f \in \mathcal{F}: f \leq t} l_{ahf} x_{ah(t-f+1)} \leq \pi^{SL} \qquad \forall t \in \mathcal{T} \Bigg\}. \tag{6}$$

Constraints (4) guarantee that appliance n starts in exactly one time slot within the interval (ST_n, ET_n). Constraints (5) determine the daily consumption profile of the appliance in each time slot, which depends on its scheduling. Finally, constraints (6) limit the overall power consumption of consumer h, since in every time slot $t \in \mathcal{T}$ the electricity bought from the grid cannot exceed the Supply Limit (SL) defined by the retailer and denoted by π^{SL}.

Let $\mathbf{U}$ be the total price paid by all players to the electricity retailer. One can prove that G is a potential game if $c_t(y_t)$ is convex with respect to y_t, with $\mathbf{U}(\mathcal{I})$ being the potential function. Potential games have several nice properties, such as the existence of at least one pure Nash equilibrium. Furthermore, such games have the Finite Improvement Property: any sequence of asynchronous improvement steps is finite and converges to a pure equilibrium.

III. Distributed Learning Algorithms

The DSM load scheduling game presented in Section II can improve the efficiency of the power grid by means of shifting users' demand from peak to off-peak periods. However, a proper method is required to converge to the game equilibria. For this reason, we have designed a learning algorithm that enables players to reach the desired game outcome in a distributed fashion, without any communication among users.

The learning algorithm here proposed, which is based on the Proportional Imitation Rule [11], is defined as an iterative process. Specifically, in each day k, all players (i.e., electric appliances) $n \in \mathcal{N} = \mathcal{A} \times \mathcal{H}$ simultaneously perform the following steps.

1) Each player evaluates the game state by estimating both his own daily bill paid at iteration $k-1$ (i.e., $U_n^{k-1} = \sum_{t \in \mathcal{T}} y_{nt}^{k-1} \cdot c_t^{k-1}$) and the bill he would have paid if he had executed any other schedule; in this way, players are able to estimate the quality (in terms of costs) of the schedule selected in the previous iteration.
2) According to the game state and the decision logic of the algorithm described below, each player selects the schedule for the current iteration k, $\overrightarrow{x_n^k}$, defined as follows:
$\overrightarrow{x_n^k} = \left[x_{n1}^k ... x_{nt}^k ... x_{n|\mathcal{T}|}^k\right]$.

We say that players *learn* to play an equilibrium if, after a given number of iterations, the schedules profile $\overrightarrow{X} = \{\overrightarrow{x_n} = \left[x_n^1 ... x_n^k ... x_n^{|\mathcal{K}|}\right], \forall n \in \mathcal{N}\}$ converges to an equilibrium strategy. Note that this algorithm relies on the assumption that after every iteration (i.e., day) k, every player n is able to estimate both his own daily bill and the bill he would have paid if he had executed any other schedule. To this end, we suppose that at the end of each iteration, the retailer broadcasts to players the electricity tariff applied that day, c_t^k, defined based on the aggregated power demand of users on day k.

The iterative learning process of each appliance $n \in \mathcal{N}$ can be therefore modeled based on a Markov chain, shown in Figure 1. At each iteration k of the algorithm, the process is in some state s which gives to player n a certain utility (i.e., bill) U_n^k. At the next iteration $k+1$, the process randomly moves to a new state s' based on the transition probabilities, and receives a new bill U_n^{k+1}.

In our algorithm, each state of the Markov chain is associated with a feasible schedule of the appliance n. To this end, we define a subset of time slots $\mathcal{S}^n \subseteq \mathcal{T}$ which satisfy constraints (4):

$$\mathcal{S}^n = \{t \in \mathcal{T} : t \in [ST_n; ET_n - F_n + 1]\} \tag{7}$$

thus the states of the Markov chain of the appliance n correspond to the elements of the set of feasible schedules of n.

States transition probabilities $P(s_i, s_j)$, with s_i, $s_j \in \mathcal{S}^n$, are updated at every iteration of the algorithm depending on the game state. In this work, we propose two policies to update the transition probabilities: *Two-State* (TS) policy and *Multi-State* (MS) policy. Let $\hat{s}$ be the schedule selected for player n at iteration $k-1$. The two policies update the

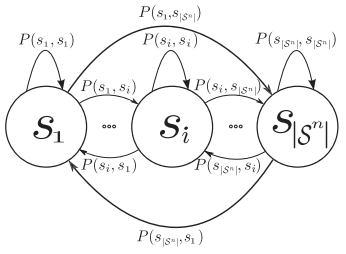


Fig. 1. Markov decision chain of player n used in the DSM load scheduling learning process.

transition probabilities from $\hat{s}$ to any state $s \in \mathcal{S}^n$ at iteration k as detailed hereafter.

A. Two-State Policy

The TS policy considers only two possible alternatives: keeping to use the old schedule or switching to the cheapest schedule among the feasible ones. The transition probabilities are updated according to the difference between the corresponding schedules electricity prices to foster the change whenever the alternative schedule is more convenient. More specifically, the TS policy performs the following steps:

1) Evaluate the daily bill that player n would have paid if he had executed any schedule $s \in \mathcal{S}^n$ at iteration $k-1$, $U_n^{s,k-1}$:

$$U_n^{s,k-1} = \sum_{t \in \mathcal{T}} y_{nt}^{k-1}(s) \cdot c_t^{k-1} \tag{8}$$

where $y_{nt}^{k-1}(s)$ is the power demand of player n at iteration $k-1$, and is defined according to constraints (5) (note that $x_{nt} = 1$ if $t = s$, 0 otherwise). In estimating these utilities, only the schedules s that satisfy constraints (6) are considered.

2) Identify the schedule $\bar{s}$ that minimizes the bill:

$$\bar{s} = \min_{s \in \mathcal{S}^n} U_n^{s,k-1} \tag{9}$$

3) Update the transition probabilities, $P(s_i, s_j)$, for all s_i, $s_j \in \mathcal{S}^n$:

$$P(s_i, s_j) = \begin{cases} 1 - \dfrac{U_n^{\bar{s},k-1}}{U_n^{\hat{s},k-1}} & \text{if } s_i = \hat{s},\ s_j = \bar{s},\ s_i \neq s_j \\ \dfrac{U_n^{\bar{s},k-1}}{U_n^{\hat{s},k-1}} & \text{if } s_i = s_j = \hat{s} \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

Based on this policy, at each iteration k, the initial Markov decision chain of Figure 1 is reduced to a two-state chain, represented in Figure 2. As a consequence, only two schedules are considered in each iteration: $\hat{s}$, which is the one chosen on the previous day, and $\bar{s}$, which is the one that, based on estimates, would have given the lowest bill in the previous iteration. The transition probabilities from $\hat{s}$ are computed based on the ratio between the bills associated with these two states: the larger their difference (hence, the lower the $\frac{U_n^{\bar{s},k-1}}{U_n^{\hat{s},k-1}}$ ratio), the higher the probability to change and implement such more convenient schedule.

B. Multi-State Policy

Differently from the TS approach, the MS policy considers all cheaper schedules among all the feasible alternatives besides the old schedule. More specifically, the MS policy performs the following steps:

1) Evaluate the daily bill that player n would have paid if he had executed any schedule $s \in \mathcal{S}^n$ at iteration $k-1$, $U_n^{s,k-1}$:

$$U_n^{s,k-1} = \sum_{t \in \mathcal{T}} y_{nt}^{k-1}(s) \cdot c_t^{k-1} \tag{11}$$

where $y_{nt}^{k-1}(s)$ is defined according to constraints (5). Also in this case, only the schedules s that satisfy constraints (6) are considered in estimating the player's bill.

2) Identify the set of schedules, $\bar{\mathcal{S}}^n$, whose bills are lower than that paid by selecting the schedule $\hat{s}$:

$$\bar{\mathcal{S}}^n = \{s \in \mathcal{S}^n : U_n^{s,k-1} < U_n^{\hat{s},k-1}\}. \tag{12}$$

3) Update the transition probabilities $P(s_i, s_j)$ for all s_i, $s_j \in \mathcal{S}^n$:

$$P(s_i, s_j) = \begin{cases} \dfrac{1}{|\bar{\mathcal{S}}^n|}\left(1 - \dfrac{U_n^{s_j,k-1}}{U_n^{\hat{s},k-1}}\right) & \text{if } s_i = \hat{s}, s_j \in \bar{\mathcal{S}}^n \\ 1 - \displaystyle\sum_{\bar{s}_l \in \bar{\mathcal{S}}^n} P(\hat{s}, \bar{s}_l) & \text{if } s_i = s_j = \hat{s} \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

Based on this policy, at each iteration k, the Markov decision chain of Figure 1 is reduced to one composed of $|\bar{\mathcal{S}}^n| + 1$ states, as represented in Figure 3. As a consequence, $|\bar{\mathcal{S}}^n| + 1$ schedules are considered in each iteration: $\hat{s}$, which is the one chosen on the previous day, and schedules $\bar{\mathcal{S}}^n$, which are those that, based on estimates, would have allowed the player to pay a bill lower than that actually paid by selecting the state $\hat{s}$. Similarly to the TS policy, the transition probabilities from $\hat{s}$ to other states are computed based on the ratio between the bills associated with the corresponding schedules. The lower the $\frac{U_n^{\bar{s},k-1}}{U_n^{\hat{s},k-1}}$ ratio between the old and cheaper schedule, the higher is the probability to change and use such more convenient schedule. As one can see, the *Multi-State* policy has a more intensive exploration stage with respect to the *Two-State* policy.

Note that at iteration 1 of the algorithm, each player selects a schedule s according to a uniform distribution over $\mathcal{S}^n$, since no information on the game state is available.

IV. Numerical Results

In order to evaluate the performance of the proposed learning algorithms, we have tested them on realistic instances of the DSM scheduling problem [13]. In this section, we first describe the methodology used in our tests, then we present and discuss the numerical results obtained by our algorithms.

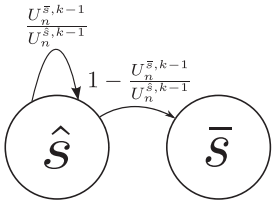


Fig. 2. Markov decision chain of player n, at iteration k, with a *Two-State* policy.

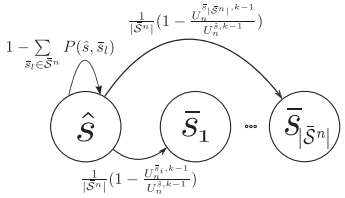


Fig. 3. Markov decision chain of player n, at iteration k, with a *Multi-State* policy.

A. Tests Methodology

The proposed learning policies are evaluated over a period of 100 days (i.e., iterations), each one represented by a set $\mathcal{T}$ of 24 time slots of 1 hour each. We consider groups, $\mathcal{H}$, of 10, 50 and 100 users, each one connected to the grid with a power limit, π^{SL}, of 3 kW. Every consumer has 4 shiftable devices out of 11 realistically-modeled appliances[1]. As for the load scheduling flexibility (i.e., size of the $[ST_n, ET_n]$ time-window), three cases are considered: *No Flexibility*, in which the appliances scheduling is fixed and cannot be optimized, *Low Flexibility* and *High Flexibility* in which, respectively, 3 and 8 different possible schedules (equivalent to 3 and 8 consecutive starting time-slots) are randomly set for each device. For each of these cases, the starting-time slot of the appliances, ST_n, is randomly selected for each user to represent a population of heterogeneous consumers. On the other hand, the ending-time slot, ET_n, is defined based on the value chosen for ST_n in order to guarantee the number of different possible schedules associated with the corresponding flexibility level.

The electricity tariff, which is defined based on the dynamic pricing tariff currently used in Italy, is obtained by adding to the day-ahead market clearing prices the costs of ancillary services (e.g., electricity transport, distribution and dispatching). More specifically, we fix the cost of ancillary services $c^{Anc} = 50 \times 10^{-6}$ € and the slope of the pricing function $c^{En} = (0,11 \times 10^{-6})/|\mathcal{H}|$€/kWh.
In order to evaluate the performance of the proposed schemes, we compare the Nash Equilibrium of the DSM game with the outcome obtained by applying the distributed learning algorithms, in terms of:

- *Total utility*: defined as the electricity bill of the group of consumers, $\mathcal{H}$.
- *Peak demand*: is the peak of the aggregated power demand of the group $\mathcal{H}$, and is defined as $\max_t y_t$.
- *Fairness*: represents the fairness of the DSM solution in terms of sharing of the energy bill among users, and is measured using the *Jain's Fairness Index*.

Note that for each scenario considered in our tests, 5 different instances are generated. In Subsection IV-B, we only report the average results obtained for each test case.

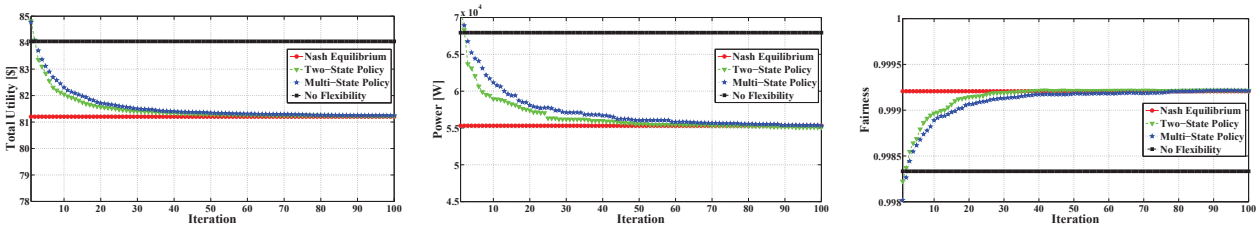
[1]Namely, *shiftable* devices: washing machine, dishwasher, boiler, vacuum cleaner; *fixed* devices: refrigerator, purifier, lights, microwave oven, oven, TV, iron

B. Performance Evaluation

Numerical results are reported in Figures 4, and 5. Specifically, in these figures, we analyze the dynamics of our learning policies by showing the aggregated users' bill, the peak demand, and the fairness of the appliances scheduling solution obtained with the proposed learning algorithms (i.e., using both the *Two-State* and *Multi-State* policies) in the case of 100 users. These results are compared with those associated with the Nash equilibrium of the DSM game in order to verify the convergence of the learning method. Moreover, we also report the performance of the appliance scheduling game with no scheduling flexibility, i.e., when the usage of electric devices is not modified by the DSM system. Numerical results show that both proposed learning policies converge rapidly to the Nash equilibrium. Specifically, the *Two-State* policy converges slightly more quickly than the *Multi-State* one, which has more alternative schedules to select at each iteration. The main reason for this result is that by limiting the exploration of the appliances schedules to only two states, as in the TS policy, at each iteration k only a few customers actually change their strategies with respect to the previous iteration, thus reducing the game dynamics. As a consequence, in this scenario, the players' estimation of the game state, which is obtained using the information on the past iteration of the algorithm (i.e., c_t^{k-1}), is more accurate.

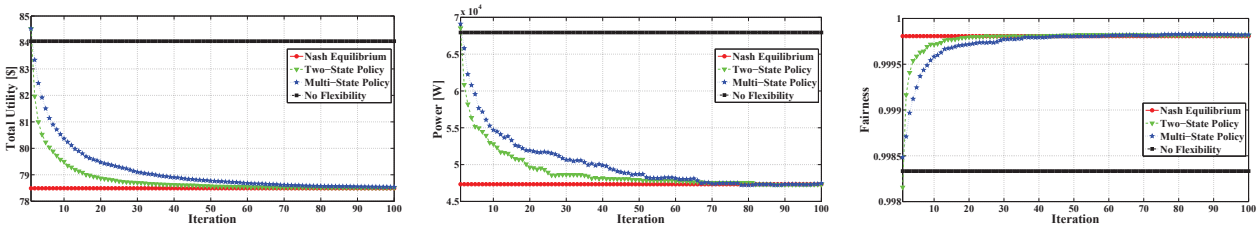
We further observe that the convergence time of the algorithms strongly depends on the flexibility of the system in scheduling the electric appliances. Specifically, the algorithms converge more quickly to the Nash equilibrium with a short flexibility level: in this case, each player has a lower number of strategies to try, thus reducing the exploration process of the learning method. However, shorter convergence times come at the cost of worse performance of the DSM system: smaller execution intervals for shiftable appliances always result in a higher bill and peak demand, as well as lower fairness, since the system contains fewer alternative schedules and the distributed DSM mechanism spreads the execution of the appliances over smaller time intervals. Nevertheless, it is worth noting that even with low flexibility, the learning algorithms always allow players to achieve better results than those obtained with fixed scheduling choices (i.e., without the DSM system), even before converging to the equilibrium of the game.

We emphasize that the convergence speed for the proposed distributed learning algorithms increases with the number of shiftable appliances which participate in the scheduling game. As illustrated in Table II, the scenario with 400 shiftable appliances is always characterized by a lower convergence time. Furthermore, it can also be observed that both learning policies achieve solutions within 5% of the electricity price obtained at the Nash equilibrium in very few iterations. Such results confirm the applicability of our distributed learning algorithms to real use-case scenarios, where thousands of users would participate in the DSM game with the expectation of quickly obtaining a reduction of their bills.



(a) Aggregated Utility (b) Peak Power (c) Fairness

Fig. 4. Aggregated utility, peak power demand, and fairness with 100 users and high flexibility.



(a) Aggregated Utility (b) Peak Power (c) Fairness

Fig. 5. Aggregated utility, peak power demand, and fairness with 100 users and high flexibility.

TABLE II
CONVERGENCE TIME (MEASURED AS NUMBER OF ITERATIONS)

	10 Users (40 Appliances)		**50 Users (200 Appliances)**		**100 Users (400 Appliances)**	
NE Gap	TS	MS	TS	MS	TS	MS
0.1%	150	136	60	123	51	81
1%	18	38	12	27	11	26
5%	3	6	2	4	2	4
10%	1	2	1	1	1	1

V. CONCLUSIONS

In this paper, we proposed two distributed reinforcement learning algorithms based on the proportional imitation rule for DSM systems. Specifically, each appliance of the system decides autonomously its best schedule for the next iteration by modulating the schedule-selection probability according to the difference between the paid electricity price and the cheapest schedule. The scheduling decision problem is modeled as a Markov chain, where each feasible appliance schedule is associated with a state of the chain, and states transition probabilities are updated using the bills paid by players.

We evaluated the performance of our proposed learning methods based on realistic instances of the DSM load scheduling game, demonstrating numerically that they quickly converge to stable Nash equilibria which lead to cheaper bills than those obtained without demand-side management. For this reason, our proposed algorithms represent promising and very efficient solutions to implement DSM systems in future Smart Grid infrastructures.

ACKNOWLEDGMENTS

This work has been partially funded by Italian MIUR project SHELL “Ecosistemi domestici condivisi e interoperabili per ambienti di vita sostenibili, confortevoli e sicuri”, Regione Lombardia project SCUOLA “Smart Campus as Urban Open LAbs”, and by French ANR in the framework of the Green-Dyspan project.

REFERENCES

[1] P. Palensky and D. Dietrich, “Demand side management: Demand response, intelligent energy systems, and smart loads,” *IEEE Trans. on Industrial Informatics*, vol. 7, no. 3, pp. 381–388, 2011.

[2] G. Strbac, “Demand side management: Benefits and challenges,” *Energy Policy*, vol. 36, no. 12, pp. 4419–4426, 2008.

[3] A. Barbato and A. Capone, “Optimization models and methods for demand-side management of residential users: A survey,” *Energies*, vol. 7, no. 9, pp. 5787–5824, 2014.

[4] Z. Wang and G. Zheng, “Residential appliances identification and monitoring by a nonintrusive method,” *Smart Grid, IEEE Transactions on*, vol. 3, no. 1, pp. 80–92, March 2012.

[5] W. Saad, Z. Han, H. V. Poor, and T. Basar, “Game-theoretic methods for the smart grid: an overview of microgrid systems, demand-side management, and smart grid communications,” *Signal Processing Magazine, IEEE*, vol. 29, no. 5, pp. 86–105, 2012.

[6] A.-H. 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 Trans. on Smart Grid*, vol. 1, no. 3, pp. 320–331, 2010.

[7] R. S. Sutton and A. G. Barto, *Introduction to reinforcement learning*. MIT Press, 1998.

[8] M. H. Bowling, “Convergence and no-regret in multiagent learning,” *Neural Information Processing Systems (NIPS)*, 2004.

[9] L. Rose, S. Lasaulce, S. M. Perlaza, and M. Debbah, “Learning equilibria with partial information in decentralized wireless networks,” *IEEE Communications Magazine*, vol. 49, no. 8, pp. 136–142, 2011.

[10] C. Ibars, M. Navarro, and L. Giupponi, “Distributed demand management in smart grid with a congestion game,” 2010, pp. 495–500.

[11] S. Iellamo, L. Chen, and M. Coupechoux, “Proportional and double imitation rules for spectrum access in cognitive radio networks,” *Computer Networks*, vol. 57, no. 8, pp. 1863–1879, 2013.

[12] A. Barbato, A. Capone, L. Chen, F. Martignon, and S. Paris, “A distributed demand-side management framework for the smart grid,” *Computer Communications*, vol. 57, pp. 13–24, 2015.

[13] ECORET Project, Official web site (ITA), http://www.rse-web.it/progetti.page?RSE_originalURI=/progetti/progetto/documento/178/312827&objId=178&typeDesc=Rapporto&RSE_manipulatePath=yes&docIdType=1&country=ita, apr 2014.