Weighted Fair Queuing as a Scheduling Algorithm for Deferrable Loads in Smart Grids

Tuncer Haslak

Abstract Weighted Fair Queuing (WFQ) is implemented for the problem of load scheduling in demand side management. Power demand, wait time and group-to-group fairness are the basis for three variants of the algorithm's implementation. The results are compared to a Greedy strategy with regard to the residual of renewable power supply and the suggested measures of fairness. WFQ proves comparable to Greedy in terms of the primary objective and in addition is capable of equally distributing resources and distress caused by deferral.

Keywords Demand side management · Optimization · Weighted fair queuing

1 Introduction

Renewable energies are a resource that strain the grid through intermittent availability and difficulty in prediction. Demand side management addresses the issue by reversing the paradigm of grid operation and controlling power consumers instead of only power generators.

Within demand side management there is a need for robust algorithms that face the unpredictability aspect of renewable energies, which makes real-time algorithms with no forecast information favorable.

Finding solutions that improve the residual power problem is only the first step. Said problem is the question of how unused renewable supply should be handled. Using the *Greedy* algorithm significant improvements can be accomplished. However, the results indicate that distress would be unevenly spread in the consumer population. If distress is distributed unevenly, compensation would also be distributed unevenly, which means that the market is not design towards fairness.

T. Haslak (✉)
University of Erlangen, Erlangen, Germany
Hof University, Hof, Germany
e-mail: tuncer.haslak@fau.de

This paper introduces *Weighted Fair Queuing (WFQ)* as a scheduling algorithm for deferrable loads in smart grids. Three variant implementations are presented that feature different concepts of fairness: serving more and simultaneously smaller loads, fairly distributing wait times, and treating any number of groups equally.

2 Materials and Methods

2.1 Model

The experimental environment is designed in AnyLogic and uses an interface to control power consuming processes, as presented in [1]. The interface is suitable for any process that is capable of prolonging periods of its activity or inactivity as can be seen in Fig. 1. This requires no intimate knowledge of the process: when an algorithmic decision is made to alter the behavior of an individual load, the request is passed on to the agent representing the load. In accordance with its internal conditions it then accepts or rejects. This is to replicate the sovereignty of loads, as in actuality algorithms are informed of the current states and consider them accordingly.

Every process is divided into 4 states: activity and inactivity, and for each of those one deferrable and one non-deferrable. Processes can alter their power demand from step to step or follow a mathematical function—anything that describes the real behavior. The model captures distinct periods in which switching can be deferred.

Fig. 1 State automaton that governs the state changes for deferrable loads; States can be changed by progression of time (clock) or request (question mark)

Fig. 2 This graph shows the power demand of a generic electric vehicle over approximately 2.5 h with a peak power demand of 22 kW. The simulation contains 300 consumers, this graph exemplifies one specimen

Fig. 3 The automated welding machine requires 150 kW over 1–3 s, repeated 12 times over. In the final step once 230 kW are needed for 5 s. This graph serves to show the internal working of a consumer agent

The deferral and operation times are a matter of survey in the companies (interviews, data sheets and surveillance). The total load shape is based on real loads that are replicated with the simulation. Figures 2 and 3 are two examples of the power demand and the repeated activation of loads.

The analyzed algorithms make no use of forecasting of any kind; they operate based on information of the “now”. The power supply is processed data-point-by-data-point, left to right.

2.2 *Population*

The model population consists of approximately 300 individual loads. These encompass beverage production, metal working, glass finishing, plastics production, textile treatment, sewage treatment, electric vehicles and compressors for both pressure applications and freezing. The switching behavior of a load for the purposes of the simulation is detailed. The realistic accuracy was verified by a test implementation in [2]. The composition of the population, i.e. the proportions of types of loads are elected to reflect a mid-sized city by approximation, based on statistics found in [3, 4] and [5]. The overall peak power demand across the whole population is 25 MW, with single members requiring power between 1 and 350 kW. The average population power demand is 5.8 kW (cf. Fig. 7). State changes are deferrable by 15% to 50% and depend on the current and future state (arrows in Fig. 1) of the individual load. At any given time an average of approximately 20% of loads is deferrable.

2.3 *Objective Function*

Considering the negative impact of renewable energies, optimizing the load population toward a renewable energy supply profile seems adequate. As the volatility of the renewables is the straining property, immediate consumption by the deferrable loads would lead to the minimization of the residual power function and thereby stabilization. Residual power (R) is the amount of power leftover from renewable supply after subtracting the demand. A sample from historic supply data is picked, that consists of wind and solar energy over 24 h with a resolution of one data point per 3 s. This deviates from the resolution of simulated demand, which is continuous. Figure 4 shows the sample. At this resolution the objective function incorporates no useful gradient information.

However, selecting a random supply function is not sufficient: First the data points of the supply function must be scaled according to the demand of the population of deferrable loads. The first simulation experiment is a simple observation of how much power the consumer population requires when there is no optimization at all. This is the so called “natural run” that allows any and all requests to switch on or off. This unimpeded 24 h simulation returns a power demand function ($P(t)_{Demand}$)

Fig. 4 Sample supply function, sum of wind and solar energy in northern Germany

in Watts. The integral thereof is the energy demand (W_{Demand}) in Watt-hours (Eq. 1).

$$\int_0^{24h} P(t)_{Demand} dt = W_{Demand} \quad (1)$$

In Eq. (2) the integrals of the natural run and the supply functions which are W_{Demand} and W_{Supply} are equated. This yields a result for c , the scaling factor. The result of this approach is a function of power supply ($P(t)_{Supply}$) that can satisfy the demand in terms of energy (kWh), but in terms of the progression of power over time (kW) any algorithm must manage transient deficits.

$$\begin{aligned} W_{Demand} &\stackrel{!}{=} c \cdot W_{Supply} \\ P(t)_{Demand} &\neq P(t)_{Supply} \end{aligned} \quad (2)$$

This scenario seems manageable concerning not only algorithms, but is also realistic with regard to resource allocation. The latter meaning that purely renewable supply seems unrealistic and would probably feature a safety factor to account for fluctuations in demand. At the same time, however, in an economic sense a rough balance between supply and demand is sensible, because of investment and operative costs. At the least, equal supply and demand appears to be a good initial metric.

To facilitate the comparison the summation from equation (3) is used, approximating the integral of the residual power for discrete time steps:

$$R = \sum_{t=0}^T |P_{supply} - P_{demand}|(t) \quad (3)$$

This condenses the result of one 24 h simulation into one manageable value.

2.4 Algorithms

2.4.1 Greedy

The *Greedy* strategy (Algorithm 1) is a heuristic search algorithm. It is incomplete, but fast and very simple. As [6] shows it generally yields good results, though it is not capable of systematically finding global optima. I chose this algorithm as a starting point for its popularity, non-specificity and low computational requirements. I have previously outlined the use of the *Greedy* algorithm in [1] and [2]. The algorithm sorts the available candidates according to size and attempts to activate as many as possible, as can be seen in Algorithm 1.

2.4.2 Weighted Fair Queuing

Weighted Fair Queuing (*WFQ*) is a sophisticated algorithm designed by Demers et al. (cf. [7]), building on the work of Nagle [8]. *WFQ* manages congestion in gateway queuing of datagram networks. The problem occurs when participants in a network attempt to send data packets, but some entities always send larger packets than others. *WFQ* replaces the pragmatic FIFO (first in, first out) paradigm by categorically distinguishing datagram sources by packet size and permanently assigning them to queues with predetermined weights. Weight, packet size and a queues history determine which packet gets serviced next.

Identifiable parallels to the scheduling of deferrable loads lead to the application of *WFQ* in a novel way. The deferrable loads also feature a well-defined requirement for resources which is their power demand in Watts. This parameter is equatable to the packet size, as is the network bandwidth to the usable residual power. The significant reason that makes a more sophisticated algorithm desirable is the unfair dissemination of resource with the *Greedy* strategy. By sorting according to size, *Greedy* categorically prefers large loads, which disadvantages smaller loads.

Algorithm 1: Greedy

```

1 while forever do
2   forall Load do
3     if Residual > Load(i) then
4       activate;
5       calculate Residual;
6     else
7       deactivate;
8     end
9   end
10 end
```

WFQ (2) employs a heuristic called *virtual finish time*¹ F (Eq. 4), which is calculated using the packet size, or in this case the wattage of a load l and the weight w of each queue i . Every load is usually permanently assigned to a queue, as are weights to queues.

$$F_i = \frac{l}{w} \quad (4)$$

This yields a numeric value for each load that wants to schedule for activation. For each queue a so-called *session virtual finish time* SF_i is calculated (Eq. 5).

$$SF_i = F_i + \sum_{n=0}^{i-1} F_n \quad (5)$$

Within every queue, all elements are queued on a FIFO basis. The element of the queue i to be scheduled next is F_i . All previously successfully activated elements are added with the summation term F_n . At every point in time the algorithm selects the queue with the lowest SF_i value. The significant difference in comparison to the original *WFQ* implementation is the explicit check (Algorithm 2, ln. 10) whether the current residual power is sufficient for the regarded load. In the network realm decreasing bandwidth will not cause a packet to be rejected service—it will, however, take increasing amounts of times to submit. This “flow” property is not given in the case of deferrable loads, therefore “choking” cannot be applied, and switching decisions are discrete.

Fairness

In datagram networks fairness is the property of allotting a certain resource to any participant according to a predefined key. Such a key maybe simple and assign equal fractions to everyone. In the case of *WFQ*, however the distribution scheme accounts for a priori observations in terms of the typical resource requirements of each participant. With *WFQ* bandwidth cannot be manipulated which means that the amount of sheer data transmitted will at best remain the same (though more packets cause more gaps). In fact the objective is to service a greater number of participants.

Fairness of Wattage

This exact thought is transferred to demand side management applied to a load’s wattage. In contrast to the original network application participants do not forfeit

¹heuristic, not physical time

Algorithm 2: Weighted fair queuing

```

1 begin function calculateSFs
2   forall queues do
3      $F(i) \leftarrow l/w$ ;
4      $SF(i) \leftarrow F(i) + F(n)$ ;
5   end
6 end
7 while  $\neg \text{Abort}$  do
8   calculate Residual;
9   calculateSFs forall Load do
10    if  $\text{Residual} > \text{Load}(i) \wedge \neg \text{Restart}$  then
11      activate;
12       $\text{Restart} \leftarrow \text{true}$ ;
13    else
14       $\text{Abort} \leftarrow \text{true}$ ;
15    end
16  end
17 end

```

after multiple trials in vain. It is implausible to assume that a load be denied access altogether. Loads forcing themselves to power on is unfavorable, but in line with the conventional grid operation paradigm of supplementation in such a case. As the key influenceable property is the length of part of the “on” and “off” episodes of each load, disproportionately large loads compensate with their “on” time. This means that larger loads are operated for shorter periods of time by the algorithm. The objective of employing this queuing scheme is to serve more participants in the same time interval. Here fairness is targeting decongestion.

Fairness of Distress

“Wait time” is a second, additional fairness measure worthy of investigation. Wattage is a strictly objective parameter. If we consider the period that a process can wait to be served, this requires very intimate knowledge of the process to determine the validity or honesty of time parameters as declared by a machine operator. A company might easily misrepresent this information. For the simulation experiment at hand the processes were personally surveyed and documented during live observations. Especially as there was no motivation presented to falsify data, the time parameters can be considered to be sufficiently accurate.

In this regard fairness would disregard the wattage of a load and consider the time it has been waiting to get served. Fairness in this sense is the equal distribution of waiting-distress among the entirety of the load population.

For this variant fixed queue assignments are replaced with dynamic ones. The longer a process waits, the further it moves into queues that make it more likely to be picked next. To this end standardized wait time is favorable, i.e. the fraction of

time waited in proportion to the maximum amount of time the load can wait (Eq. 6). Without this measure loads that provide less flexibility are irrationally rewarded.

$$t_{st} = \frac{t_{waited}}{t_{max}} \quad (6)$$

Fairness Group-by-Group

As for a third measure of fairness the load population is subdivided into regions of equal size, representative of quarters or districts. In this configuration each queue represents one region, and regions compete for the available resources. This option is designed to grant each group the same right to resources. Any attribute can be used to queue in this manner. This generic queuing scheme targets an equal distribution of deferral times and power to independent groups, but not within the groups. As the *virtual finish time* paradigm is still applied, smaller loads are preferred.

Weights and Queues

There is no definitive answer on how weights should be chosen. Existing procedures such as that outlined in [9] are not applicable because the prevailing problem of node congestion in networks is not a relevant phenomenon. Therefore a mode of selecting weights should rely on the available parameters. Each of the fairness measures suggested above requires its own weighing scheme.

2.4.3 Weight by Power (WFQ-Power)

Considering that the resource “power” is engrossed correlating with wattage, a load’s wattage should inversely determine its likelihood of being elected. As a number of queues to which all loads are assigned, 5 was elected as it is typical. The fifth of the population with the highest wattage is assigned to the according quintile, and so forth. The weight for each queue is determined by computing its average wattage. Following equation (4), fairness is established when the *virtual finish times* of all individual loads are equal. This means solving for w and equating the average wattages of all queues. Table 1 shows which weights are assigned to queues, and the respective average wattage within the queue.

For example: At 250kW load # 1 is a member of the queue weighted at 0.09, i.e. the group of highest wattages. Compared to bottom quintile member load # 2 of 20kW and its weight of 1. Load # 1 has a virtual finish time 138 times higher. The mismatch in wattage reflects proportionately in the weights.

Table 1 Averages of power in queues and corresponding weights (*WFQ-Power*)

μ of power in queue	Queue weight
220 kW	0.09
120 kW	0.17
90 kW	0.23
40 kW	0.5
20 kW	1

Table 2 Loads for *WFQ-Time* are dynamically assigned to queues by the percentage of wait time expended from their maximum so far. In order to augment urgency, queue weights are doubled from queue to queue

Expended wait time	Queue weight
80–100%	1
60–80%	0.5
40–60%	0.25
20–40%	0.125
0–20%	0.0625

2.4.4 Weight by Wait Time (WFQ-Time)

Queuing by standardized wait time is carried out dynamically. In contrast to wattage-queuing there is no a priori assignment. With increasing wait time, a load is assigned to queues with higher weights that equate to higher likelihoods of getting picked. Once again there are five queues, each designated for a 20% interval of wait time completed. Table 2 shows how weights are governed for each queue. Decreasing weights augment urgency.

2.4.5 By Region (WFQ-Geo)

In the case of grouping by region equal weights are assigned to all queues. As the fictitious regions comprise equal portions of every category of the overall population, and given the equal weights, this queuing scheme acts comparable to a round-robin. This is true for all types of *WFQ* with equal weights (cf. [8]).

3 Model Limitations

The proposed model requires highly automated processes. The underlying work plans are machine accessible so that the ensuing steps and their duration can be projected. Semi-automation or increased human interaction would hinder this procedure.

Even though they were not encountered during survey, processes that can continuously adapt the level of their power input can only be replicated as a series of

discrete steps. On the contrary this model is a response to the lack of continuously adaptable processes. Yet this is a limitation.

In addition the size of the consumer population cannot be arbitrarily increased. To this end an arbitration mechanism appears feasible. Multiple optimization algorithms would work on a fraction of a larger problem which in turn is consolidated by optimization layers higher up in the hierarchy. This approach could also help to reduce the amount of datagrams that are sent between the loads and the optimizer.

4 Results

In each simulation there is a population of approximately 300 manageable loads that must be scheduled over 24 h following a randomly selected supply signal. There are five experiments:

1. Natural run: All loads operate without any algorithmic interference. This is a reference trial.
2. Greedy
3. WFQ: Power
4. WFQ: Wait time
5. WFQ: Region

Results are compared regarding the fulfillment of the primary optimization objective from equation (3)—the minimization of residual power deviations. The three queuing schemes of the *WFQ* variants have different secondary objectives. The algorithm in experiment 3, with weights defined by power demand, aims to serve more participants, especially smaller ones. Experiment 4 entails queuing by wait time, which is designed to equally distribute the wait time percentages over the entire population. Queuing by region, experiment 5, is designed to allocate the same resources to arbitrarily defined subsets of the population.

In order to judge the fulfillment of these secondary objectives additional parameters must be analyzed. *Cycles* counts all activations which is the sum of every individual power-on that was granted—in the datagram sense this is analogous to serviced packages. The *standardized average wait time* is indicative of how long every activation was deferred in percent of the maximum. *Deferral length* is the sum of all deferrals as time. The *variance* of this value is representative of how evenly deferrals are distributed among the population.

Table 3 summarizes the results of experiments 1–4. As outlined in Sect. 2.3 the significant data is the power consumption function that every algorithm causes. Figure 5 is the consumption profile caused by the *Greedy* strategy. Consumption curves for all algorithms including the natural run (experiment 1) can be found in the appendix (Figs. 7, 8, 9, 10, and 11). The visual differences in the residual curves are **exiguous** due to the algorithmic similarity. The key difference is the residual energy. Systematic differences do not express discernibly in the graphs. Showing which load

Table 3 While *Greedy* performs best in terms of *residual*, it causes a high variance in deferral lengths (inequal waiting). *WFQ-Power* and *WFQ-Time* accomplish their secondary objectives of equal distress distribution (*variance*) and increasing the serviced loads (*cycles*) at the cost of energy optimization (*residual*)

	Natural run	Greedy	WFQ-Power	WFQ-Time	WFQ-Geo
Residual [MWh]	13,455.3	6261.8	6524.7	6749.9	6720.7
Avg. standardized wait [%]	0	34.0	34.7	34.2	35.9
Avg. deferral length [h]	0	3.83	3.69	3.90	3.93
Variance of deferral length	0	0.2027	0.1730	0.0973	0.169
Cycles (switch on & off)	577,783	419,228	489,450	439,278	443,366

Fig. 5 Power consumption for the Greedy algorithm, residual *R* in upper right corner

Table 4 This table shows the distribution of wait times of all queues of the *WFQ-Geo* algorithm. The objective here is to treat all queues equally, which hinders global optimization objectives. Fairness reflects in the similarity of waiting times

Queue	1	2	3	4	5
Avg. standardized wait [%]	35.9	35.6	36.1	35.8	36.0
Avg. deferral length [h]	3.89	3.95	4.02	3.89	3.92
Variance of deferral length	0.172	0.169	0.173	0.168	0.164
Cycles	88,243	87,922	88,101	88,450	87,921

is activated at any given time offers no insights from algorithm to algorithm, as no pattern emerges.

The *Greedy* algorithm performs best by reducing the residual by 54.8%. *WFQ-Time* improves the deferral distribution by a factor of 2.08, while maintaining a very similar deferral length.

Table 4 shows the results of all queues of experiment 5, each representing one region. As this algorithm is specified not to improve the global population, but to serve queues equally, its results cannot be directly compared to the other experiments. The only exception is the residual. Queue deferral length is on average 11% higher compared to *WFQ-Power*. Queue by queue comparison shows that results group close together for all parameters, but worse in general.

Fig. 6 The *WFQ* algorithms can compete with the versatile *Greedy* algorithm in terms of the residual optimization (blue). However *Greedy* focuses 80% of distress on only 69% of the population (green). In addition *WFQ-Power* rejects especially few loads (red)

Figure 6 summarizes the advantages of *WFQ*: it is an improvement on the *Greedy* algorithm which is fast and versatile. At the cost of slightly worse results with regard to the optimization of the power residual, *WFQ* algorithms distribute distress more equally, and serve more loads all else being equal.

5 Discussion

5.1 Improvement

As to be expected the natural run that features no algorithm or adaptation to the objective function causes a high residual. The *Greedy* algorithm delivers the greatest improvement in terms of residual. At the same time this algorithm unevenly distributes the wait time distress across the population which is expressed in the variance of the average deferral time. *WFQ-Time* significantly narrows the variance of deferral, following its secondary objective. The *WFQ-Power* variant is capable of

placing more individual activations which can be seen in its *cycle* count. Keeping in mind that the objective function cannot be fully attained, as indicated in Sect. 2.3, the algorithm still reduces the gap in serviced packages by approximately 60% in comparison to *Greedy*.

The results from the algorithm *WFQ-Geo* in experiment 5 are inferior in general as it causes longer wait times and the variance thereof indicates that wait times are heterogeneously distributed within queues. The objective of equal queue treatment is achieved at the cost of worse results all in all.

5.2 *Distress*

Distress distribution is a desirable trait in demand side management that is not trivially deducible. Wait time is a parameter that cannot be easily subverted, if for example the maximum deferral time is repaid as an incentive. Weighted Fair Queuing is capable of delivering comparable result to a *Greedy* strategy in the primary objective of residual power improvement. In addition it can be adapted to pursue additional objectives. I presented three possible measures of fairness: equality between multiple queues or regions (*WFQ-Geo*), the even distribution of wait times (*WFQ-Time*), and serving more participants (*WFQ-Power*)—the latter being most similar to the original algorithm. This shows that *WFQ* can be used to introduce fairness to the selection of deferrable loads.

5.3 *Methodological Contribution & Policy Advice*

Despite its complexity *Weighted Fair Queuing* is a staple of network congestion management. It is highly functioning and ubiquitous because of its performance and its ability to accomplish fairness. The application to energy distribution problems is novel. At the same time it introduces the problem of fairness into the matter, which was disregarded because of the focus on residual power optimization, monetary compensation or welfare gain.

The optimization problem presented and solved in this paper is significant in its size regarding multiple dimensions. Firstly the time resolution exceeds the commonly quoted 15 min intervals that stem from the tertiary grid balancing realm. Grid stabilization however can only be achieved when the shrinking number of spinning masses can be replaced. The realization that compensatory measure must move below the 30-seconds-threshold is at the core of this research. Secondly, the population is substantial with 300 members and 25 MW peak. Each load is replicated as an agent with internal processes, variables and decision making. Furthermore their states are not estimated or stochastically approximated—decision making and transmission is acute, meaning that statuses are inquired, evaluated, decided on and requests dispatched. Thirdly, no simplifying assumptions are applied

to the target function (power supply) such as smoothing. The complexity of the problem is embraced to the extent that the resolution of the objective function (supply) is the restricting factor. The argument is in favor of preparedness, as methodologically the question answered here is that demand side management features the necessary scalability and agility.

The suggested algorithms are implementations of centralized algorithms. As the technology is still at an early stage centralization is acceptable. Especially as the methodological exploration of this topic—which this research is part of—is still ongoing. For future applications, however, this is not advisable, as a single entity will possess all information on all clients (loads). The controlling entity would be vulnerable to attacks. The presented model, is acting on a request basis which means that a load can reject any suggestion to change its state. Designing intrinsically safe systems increases complexity, but is quintessential policy advice.

5.4 Practical Implications

For companies owning deferrable loads the practical implications are low, especially if processes are highly automated. The test implementation presented in [1] required no manual interaction. While loads were in interference the optimization paused and resumed automatically. The more a deferrable load is interwoven into the process of a company, the more difficulty can arise from automated unexpected deferrals. Deferrable loads are an effort that aim to improve the usage of renewable loads. Some resulting distress is to be expected. Although not insinuated here, market design should compensate for this.

The suggested interface is at the core of the research presented, as it allows for the load to negotiate deferral and reject it. From a load's perspective *WFQ-Time* is the most favorable, as it guarantees that no load is overburdened with deferral. The *Greedy* algorithm is unfavorable, as some loads are asked to defer significantly more often than others (unfairness).

5.5 Algorithm Comparison

There is a multitude of optimization algorithms that can be used. The advantage of the above presented algorithms is their simplicity. They can be categorized as local search algorithms, which means that they use a type of heuristic, like SF_i in Eq. (5). As there are no simplifying assumptions, and every load is designed to be addressed with a direct request, the problem presents itself to be massive and without useful gradient information, which excludes classic optimization algorithms.

A suitable alternative would for example be “*Simulated Annealing*” (cf. [10]). It is an algorithm that begins by amply exploring the solution space with random solutions. A solution in this case is an allowable sequence of activation and pause

for all loads. The algorithm continues by rejecting or accepting results, based on the metaheuristic property (“temperature”). In principle this algorithm converges on the global optimum solution, which local search algorithms are not systematically capable of. The significant disadvantage is considerably higher run time. A preliminary test implementation shows that, after 14 h *Simulated Annealing* delivers results, which are comparable to the *Greedy* strategy which takes approximately 15 min on the same desktop computer (2.5 GHz Intel Core i7 with 16 GB 1600 MHz DDR3 memory).

6 Related Work

Gellings laid out the fundamentals of demand side management in [11].

The objective function and its definition are based on [12] and [13].

Weighted Fair Queuing and the approach of *Generalized Processor Sharing* are based on [7, 8] and [14]. These original sources were used for the adaptation to demand side management.

Most related work is based on [15] where an optimal deferrable load control problem is defined. The focus is placed on market parameters which is why price bounds are employed. [16] focuses on the problem laid out in this publication. To solve this problem a decentralized algorithm is employed, which communicates the power residual to all participants. This is in contrast to centralized solution efforts, but capable of scheduling multiple instances of the same load. [17] utilizes a similar approach but only for singular placement of loads in 24 h.

[18] uses a stochastic model which defers one activation of each load and no reactivation. The underlying idea is very strongly connected to the paradigm of grid operation. This means that the time scales of balancing power in the European grid are adopted and all activity takes places in time steps of 15 min, and deferral times are at least 4 h. The significant difference is the lack of understanding and anticipating load behavior. The publication focuses on the simulation of wind power only and the cost of demand side management.

[19] has a similar approach in that only one activation instance of every participant is considered. The multitude of loads is approached as a combinatorial problem. Here the smallest time step is one hour. In neither of these loads can extend their operative time.

Formulated as a constraint problem regarding the availability of the demand side, [20] manages a day-ahead scenario by interacting with energy markets. By the introduction of welfare they move away from monetary quantification. They outline the difference in time scale between markets (larger steps) and the load behavior (smaller steps).

Appendix

Fig. 7 Power consumption for a non-optimized reference run; 1 h equals 100 simulation units

Fig. 8 Power consumption for the Greedy algorithm

Fig. 9 Power consumption for the WFQ-Power algorithm

Fig. 10 Power consumption for the *WFQ-Time* algorithm

Fig. 11 Power consumption for the *WFQ-Geo* algorithm

References

1. T. Haslak, *SACI 2016 – 11th IEEE International Symposium on Applied Computational Intelligence and Informatics, Proceedings*, 2016, pp. 381–384. <https://doi.org/10.1109/SACI.2016.7507406>
2. C. Brabec, M. Neswal, R. German, Final Project Report: Smart Grid Solar. Tech. rep. (2019)
3. A. Kollmann, C. Amann, C. Elbe, V. Heinisch, Lastverschiebung in Haushalten, Industrie, Gewerbe und kommunaler Infrastruktur Potenzialanalyse fuer Smart Grids. Tech. Rep. 2 (2013)
4. M. Buddeke, C. Krüger, F. Merten, Modellbeschreibung: Einsatzmodell für Flexibilitätsoptionen im europäischen Stromsystem Projektbericht. Tech. rep. (2016)
5. L. von Bremen, M. Buddeke, D. Heinemann, A. Kies, D. Kleinhans, C. Krüger, F. Merten, M. Preute, S. Samadi, T. Vogt, W. Lukas, Ergebnisse und Empfehlungen des BMBF-Forschungsprojektes Regenerative Stromversorgung & Speicherbedarf in 2050. Tech. Rep. September (2016)
6. T. Weise, *Global Optimization Algorithms – Theory and Application* (2008). <https://doi.org/10.1109/HIS.2007.11>
7. A. Demers, S. Keshav, S. Shenker, Analysis and simulation of a fair queueing algorithm (1989). <https://doi.org/10.1145/75247.75248>. <http://portal.acm.org/citation.cfm?doid=75247.75248>
8. J. Nagle, SIGCOMM Comput. Commun. Rev. **14**(4), 61 (1984). <http://doi.acm.org/10.1145/1024908.1024910>
9. E. Magaña, D. Morató, P. Varaiya, in *10th International Conference on Telecommunications, ICT 2003*, vol. 2, 2003, pp. 917–922. <https://doi.org/10.1109/ICTEL.2003.1191562>
10. S. Kirkpatrick, C.D.J. Gelatt, M.P. Vecchi, *Science* **220**pp(4598), 671 (1983)

11. C. Gellings, Proc. IEEE **73**(10), 1468 (1985). <https://doi.org/10.1109/PROC.1985.13318>.
<https://ieeexplore.ieee.org/document/1457586/>
12. G. Strbac, Energy Policy **36**(12), 4419 (2008). <https://doi.org/10.1016/j.enpol.2008.09.030>
13. W.P. Schill, Energy Policy **73**, 65 (2014). <https://doi.org/10.1016/j.enpol.2014.05.032>
14. A.K. Parekh, R.G. Gallager, IEEE/ACM Trans. Netw. **1**(3), 344 (1993). <https://doi.org/10.1109/90.234856>
15. A.J. Conejo, J.M. Morales, L. Baringo, IEEE Trans. Smart Grid **1**(3), 236 (2010). <https://doi.org/10.1109/TSG.2010.2078843>
16. L. Gan, A. Wierman, U. Topcu, N. Chen, S.H. Low, *Proceedings of the Fourth International Conference on Future Energy Systems (e-Energy '13)*, pp. 113–124 (2013). <https://doi.org/10.1145/2567529.2567553>. <http://dl.acm.org/citation.cfm?id=2487179>
17. G. Graditi, M.L. Di Silvestre, R. Gallea, E.R. Sanseverino, IEEE Trans. Ind. Inf. **11**(1), 271 (2015). <https://doi.org/10.1109/TII.2014.2331000>
18. M. Klobasa, *Dynamische Simulation eines Lastmanagements und Integration von Windenergie in ein Elektrizitätsnetz*. Ph.D. thesis (2007). <https://doi.org/10.3929/ethz-a-005484330>
19. T.P.I. Ahamed, S.D. Maqbool, E.a. Al-Ammar, N. Malik, *2011 2nd IEEE PES International Conference and Exhibition on Innovative Smart Grid Technologies*, 2011, pp. 1–4. <https://doi.org/10.1109/ISGTEurope.2011.6162637>
20. L. Jiang, S. Low, *2011 49th Annual Allerton Conference on Communication, Control, and Computing*, Allerton, 2011, pp. 1334–1341. <https://doi.org/10.1109/Allerton.2011.6120322>

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.