Decentralized elastic electricity demand scheduling

Bruno Mendivez Vasquez
Monash University
bruno.mendivezvasquez@monash.edu

Lachlan L. H. Andrew
University of Melbourne
lachlan.andrew@unimelb.edu.au

Julian Garcia
Monash University
julian.garcia@monash.edu

***Abstract*—The adoption of intermittent renewables in the electricity grid will increasingly require loads to track available generation. This paper proposes several algorithms to schedule *flexible-power loads* that arrive unpredictably to a grid-connected microgrid. These are based on standard schedulers such as Shortest Remaining Processing Time (SRPT), adapted to the fact that appliances have maximum power ratings. A simple decentralized scheduler is applied to ensure that the aggregate load does not exceed the generation, and a gain control mechanism is proposed to stabilize the system. The proposed scheduler has two sources of sub-optimality: determining the values of the control signals, and their overall structures. In order to separate these two effects, we consider a full communication version as a benchmark in order to assess performance against the SRPT-based algorithms and fair sharing. This full communication "decentralized" approach approximates the performance of SRPT-based policies, suggesting that a true decentralized controller may be feasible in the future.**

***Index Terms*—micro-grid, on-line scheduling**

I. Introduction

An energy grid must, at all times, balance power injected into the network and power extracted from it. In the past, variations in demand have been compensated by adjusting generation. However, in the context of grid-connected microgrids, renewable supplies have limited scope to increase generation to meet demand.

Energy storage systems are a potential solution to this problem, but these remain costly to employ at scale [1]. Another option is to use on-demand backup generators that are able to cope with demand changes, but these solutions typically use fossil fuels, increasing both costs and emissions [2].

A valuable component in the mix of solutions to this problem is demand side management (DSM), which shifts some control from the generation side to customers and their devices. The focus of this approach is on loads that can adjust their demands to match the availability of electricity using smart-grid technologies (*e.g.*, remote control of devices) [3]. Different devices will have different degrees of freedom for control, and therefore be suited to different DSM strategies. These degrees of freedom include power and energy requirements, load predictability and other constraints such as deadlines. Our work is therefore complementary to most of the existing work on load scheduling, since different schedulers can be used at the same time on different sets of loads.

We are interested in loads that need a fixed amount of energy, but can adjust their power. Applications fitting this profile are plentiful, and include for example, heating water to a set temperature or fully charging a battery. We are also interested in the typical case in which loads cannot be predicted and scheduled in advance, in contrast to the common assumption that loads are known a day ahead and their start times can be controlled [4], [5]. Instead, power allocation is dynamic, as the schedule is adjusted on-line when devices enter and exit the system. We also consider demands that must eventually be met, rather than having deadlines as in [6].

Considering demands with fixed energy, variable power, unpredictable arriving times and no deadlines, our goal is to minimize the average response time (*i.e.*, the time taken to fulfill each device's demand). However, the response time of a device depends on its power draw limit, also called maximum power rating or rated capacity.

If the rated capacity for each device exceeds the maximum generation capacity then the optimal scheduling policy [7] would be Shortest Remaining Processing Time first (SRPT). Under SRPT, all generated power is consumed by the device that can be completed soonest, or has the fewest Joules remaining of its demand. In this setting, the two are equivalent.

In contrast, when rated capacities are less than the maximum generation capacity, it is optimal for multiple demands to consume power simultaneously, sharing the generation. In this case, SRPT is not suitable. This problem can be approached by considering a scheduler with multiple servers. For k identical servers, SRPT is asymptotically optimal under heavy Poisson load, but in the worst case it has an unbounded competitive ratio; that is, its cost can be an unbounded factor higher than that of a scheduler that knows future loads [8].

Other schedulers might be suitable when there is no information on the amount of energy required [9], or when trying to minimize slowdown [10], which is the ratio between response time and expected duration of use when power draw occurs at a maximum rate. However, these schedulers cannot be used in our setting because they are centralized, requiring some information to be transmitted explicitly from households to the central controller. To overcome this issue, we propose and evaluate by simulation an approach reminiscent of network utility maximization, long studied in data networks [11], [12]. Here, each limited resource (*i.e.*, the generation) computes a congestion signal that is communicated to each load, from which the device computes its allowable consumption rate.

We first provide a formal description of the problem at hand, in Section II; where a joint optimization problem is defined based on a global signal that measures the amount of electricity availability. A congestion signal, reflecting the grid's supply and demand profiles, is calculated by the scheduler based on the total load of the grid, and then transmitted to all devices. Locally, every device calculates its own instantaneous power demand based on both the global signal and also its progress through its task. In practice, a second broadcast signal is required, but communication remains minimal.

We then proceed to describe assumptions about the workload model (Section III), including service and arrival. Section IV discusses and empirically evaluate different scheduling algorithms including centralized ones, and various levels of decentralization. Section V concludes the paper.

II. Problem Formulation

The demand response scheme of this paper applies to the following type of workload. Devices that do not fit this description may exist in the system, but are not controlled by this scheme.

For each i in an index set $\mathcal{I}$, a householder activates a device at time a_i, which will require an amount of energy $F_i > 0$ to complete its task. The device has a rated capacity $B_i \in (0, \infty]$, and can draw power no faster than that. The size F_i, capacity B_i and arrival time a_i of the request are unknown until its arrival time. Demands are *elastic*, in that they can draw power at a curtailed rate $x_i(t) \in [0, B_i]$, for $t \in \mathbb{R}$. The task is considered completed at the first time e_i such that $\int_{a_i}^{e_i} x_i(t)\,dt \geq F_i$, and jobs have no hard deadlines. Elements of $\mathcal{I}$ are referred to as "jobs".

We define the available generation $G(t)$ as the maximum generation obtained from all connected sources such as wind, solar and the main grid. The net generation $G_N(t) = G(t) - x_0(t)$ excludes the power used by devices not under our control, $x_0(t)$. Henceforth we will deal only with the net generation. Let $\mathcal{I}_t = \{i \in \mathcal{I} : a_i \geq t \wedge \int_{a_i}^{t} x_i(\tau)\,d\tau < F_i\}$ be the set of jobs currently in the system. The scheduling task is to solve the maximization problem

$$\min_{x,e} \quad \sum_{i \in \mathcal{I}} e_i - a_i \tag{1}$$

$$s.t. \quad \int_{a_i}^{e_i} x_i(t)\,dt \geq F_i \qquad \forall i \in \mathcal{I}_t \tag{2}$$

$$0 \leq x_i(t) \leq B_i \qquad \forall i \in \mathcal{I}_t \tag{3}$$

$$\sum_{i \in \mathcal{I}_t} x_i(t) \leq G_N(t) \qquad \forall t. \tag{4}$$

The decision variables for problem (1)–(4) are the demand scheduling function $x = (x_i(t), \forall i \in \mathcal{I})$ and the finishing time vector $e = (e_i, \forall i \in \mathcal{I}_t)$. Constraint (2) ensures that i finishes by its end time e_i, (3) enforces device power ratings, and (4) ensures that the aggregate demand is satisfied by the available generation.

Without the upper bound in (3), problem (1)–(4) would be solved by shortest remaining processing time (SRPT) policy [7]. We are also interested in minimizing slowdown, defined as the ratio between actual response time $e_i - a_i$ and the minimum response time F_i/B_i. If the system is equivalent to a multi-server system (all B_i are equal and G_N is constant and divisible by B_i) then this would be solved by the minimum Residual times Size (RS) [13].

III. Workload model

A. Job model

A job consists of a service requirement (amount of energy to receive) F_i, and a maximum power (rated capacity) B_i. However, these are correlated: an electric vehicle has high F_i and B_i, and both are low for a phone. However, human patience is nearly constant. We therefore model the minimum completion time, F_i/B_i, as being independent of the job size F_i. The marginal distributions of F_i and F_i/B_i were obtained in two steps. The first is fitting a bounded Pareto distribution, with $F(x) = (1 - (l/x)^\alpha)/(1 - 1/h)^\alpha$, to the REDD dataset [14]. This yielded $\alpha = 0.41$, $l = 0.02\,\text{h}$ and $h = 9.22\,\text{h}$. This can yield implausibly large B_i values. The second step was accept/reject sampling to accept only jobs with $B_i < 2.4\,\text{kW}$.

B. Generation (service) model

Arriving jobs are served depending on the available generation power $G(t)$, shared among all households in a microgrid. Solar power is the main source of $G(t)$, although when it is insufficient, households draw power from the main grid to maintain a minimum available power G_b,

$$G(t) = \begin{cases} G_b & \text{if } G_{PV}(t) \leq G_b \\ G_{PV}(t) & \text{otherwise.} \end{cases} \tag{5}$$

Individual solar generation profiles were obtained from the Ausgrid dataset [15] in order to construct G_{PV}.

C. Arrival model

Jobs arrive according to a time varying Poisson process, with intensity proportional to the measured energy consumption in [15], and scaled to obtain different loads.

The mean system's load is then $\rho = \lambda/(\mu g)$, where ρ is the mean load, λ is the mean arrival rate, $1/\mu$ is the mean job size and g is the mean generation.

A load of, say, $\rho = 0.6$ leads to the system being in overload during much of the day, and underload overnight, and so we expect high mean delays even for quite low average loads.

IV. Scheduling algorithms

We consider two classes of schedulers. The first, classical schedulers adapted to capacity limitation, B_i, use knowledge of all jobs that have arrived. We will investigate the impact of B_i on these schedulers. The second class are new distributed schedulers, using minimal information. These are simple proof-of-concept schedulers, aiming to determine whether or not such distributed scheduling is feasible.

A. Centralized schedulers

The first schedulers are based on the following standard single-server schedulers, which apply when $B_i \geq \max_t G(t)$. These all know how many jobs there currently are and how much service each has received; the first two also know F_i. Here, $x_i(t)$ is the rate of service given to job i at time t.

- SRPT (Shortest Remaining Processing Time): minimizes response time [7]. All service $G(t)$ goes to the job that has the smallest non-zero remaining service requirement, $r_i(t) = F_i - \int_{a_i}^t x_i(\tau)\,d\tau$.

- RS (Residual times Size; also SPTP, smallest processing time product): minimizes slowdown for homogeneous Poisson arrivals [16]. All service goes to the job that minimizes $F_i r_i(t)$.
- LAS (Least Attained Service) [17]: minimizes response time for decreasing hazard rate (DHR) jobs, among policies that do not know job sizes. All service goes to the job that has received the least service, $R_i = \int_{a_i}^t x_i(\tau)\,d\tau$, with ties sharing equally.
- PS (Processor Sharing): benchmark that allocates equal resources to all jobs.

We now introduce versions of these schedulers that are applicable when individual jobs are rate limited.

If $B_i < G(t)$, the system no longer behaves as a single server queue. If $G(t)/B_i$ is a constant integer, independent of time t and job i, then the system is a multi-server system with $G(t)/B_i$ servers. We refer to these schedulers as "bounded" variants, denoted B-SRPT, B-RS, B-LAS and B-PS.

Each of the above schedulers, except PS, gave service to the job that minimizes some $\theta(t)$. SRPT uses $\theta_i(t) = r_i(t)$, RS uses $\theta_i(t) = F_i r_i(t)$ and LAS uses $\theta_i(t) = F_i - r_i(t)$. When individual jobs have rate constraints B_i, service is shared as follows. At each time, the jobs are indexed by s in increasing order of $\theta(t)$. That is, $\theta_{s_i}(t) \le \theta_{s_j}(t)$ for $i < j$. Let $\Theta = \max\{j : \sum_{i=1}^{j} B_{s_i} \le G(t)\}$ be the number of low-θ jobs that can be served at their rated capacities. Then job s_j gets rate

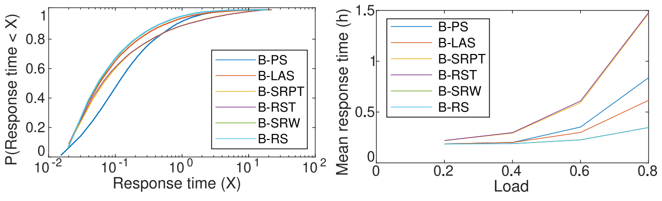
$$x_{s_j}(t) = \begin{cases} B_{s_j} & \text{if } j \le \Theta \\ G(t) - \sum_{j=1}^{\Theta} B_{s_j} & \text{if } j = \Theta + 1 \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

Schedulers such as SRPT treat (remaining) service time as synonymous with remaining work. When jobs have different limits B_i, these are no longer equivalent. In particular, SRPT chooses to serve the job that can be eliminated from the system most quickly. This corresponds to serving the job with minimum r_i/B_i. Should B-SRPT sort by $\theta_i = r_i/B_i$ or by $\theta_i = r_i$? We evaluate both, under the names B-SRPT and B-SRW (bounded shortest remaining work). B-RS can also use r_i/B_i in place of r_i, reflecting the name "processing *time* product". We call this B-RST.

B. Performance evaluation: centralized schedulers

These schedulers were evaluated in the context of a small microgrid consisting of five houses is powered by a shared solar array, which can be used by all incoming jobs in the microgid. Jab arrivals are a 10 day non-homogeneous Poisson process. Unless specified, the load of the system is $\rho = 0.6$ and the base generation $G_b = 100\,W$ per household.

Figure 1(a) shows the probability of response time exceeding a certain threshold. Our benchmark B-PS is the worst for jobs with short response times, but very long response times are more likely with B-SRPT and B-RST. In contrast to SRPT, B-SRPT has a poor mean response time, twice that of B-PS.



(a) Response time CDF, $\rho = 0.6$. (b) Response time vs load.

Fig. 1. Response time for bounded, centralized schedulers.

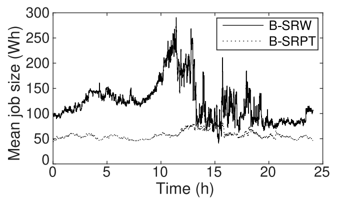


Fig. 2. Mean job size for a 24 hour profile comparing B-SRPT with B-SRW.

In contrast, B-SRW ($\theta_i = r_i$ not r_i/B_i) performs well. This may be because, although B-SRPT minimizes the time until a job leaves the system, it consumes excessive power doing so, which slows other jobs. Quantitatively, approximate the rate at which jobs are being served by $\sum_i x_i/r_i$ where x_i is the service rate given to job i. It is reasonable to seek to maximize this, subject to $x_i \le B_i$ and $\sum_i x_i \le G$. This is what B-SRW does. When $B_i > G$, it reduces to SRPT. In contrast, B-SRPT only reduces to SRPT if all B_i are equal.

One possible reason that scaling by B_i is harmful is the correlation between job size F_i and capacity B_i; it prioritizes large jobs, so many small jobs stay in the system (Fig. 2).

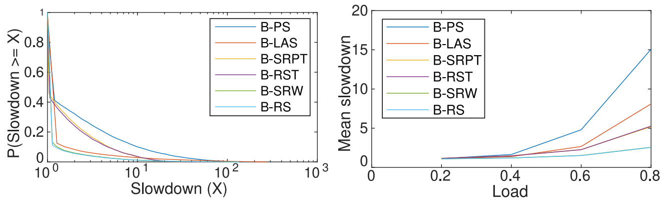
B-RS performs marginally better than B-SRPT since SRPT is suboptimal for multi-server response time [8]. However, it is not established that RS is better, although RS is optimal for slowdown with multiple servers. It is unclear whether the difference seen here is a simulation artifact, a property unique to our system, or also true of parallel server systems.

B-LAS outperforms B-PS, as bounded Pareto is nearly DHR. More impressively, it is also close to B-SRW, despite not knowing remaining service times.

Slowdown is the ratio between the time a job spends in the system and F_i/B_i, the minimum time it can stay in the system. Figure 3(a) shows the complementary cumulative distribution function, cCDF, for $\rho = 0.6$. This time, B-PS performs worst, instead of the time-based schedulers B-SRPT and B-RST.

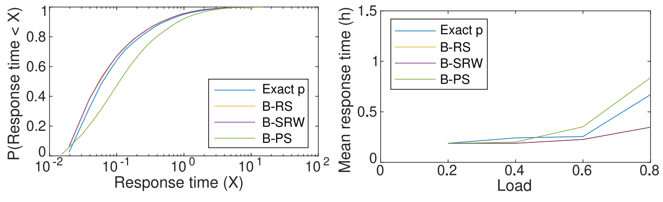
These slowdowns seem alarmingly high for a load of only 0.6. However, recall that there are long periods of overload, which dominate the slowdown figures. Hence, a scheme like this cannot by itself match demand to a solar generation profile. However, it can form a useful component of a system, combined with storage, diversified renewables and scheduling of loads whose start times are flexible [18].

This shows that substantial gains come from applying scheduling theory to devices in a microgrid, rather than attempting to supply all devices with their rated power, B_i,



(a) Slowdown CDF, $\rho = 0.6$. (b) Slowdown vs load.

Fig. 3. Slowdown for bounded, centralized schedulers.



(a) Response time CDF, $\rho = 0.6$. (b) Response time vs load.

Fig. 4. Response time for bounded, centralized schedulers, and the distributed control with exact knowledge of the Lagrange multiplier, p.

or sharing power equally, as done by B-PS. It also shows that scheduling should not be based on remaining service *time*, as in the names of SRPT and SPTP, but rather on remaining *work*.

C. Decentralized schedulers

We have shown that the best schedulers depend on knowing the remaining work (energy, r_i) of each task and rated capacity B_i of each device at a central point. A big disadvantage of centralized schedulers is the need for a two-way communication of both B_i and r_i and also the rate allocation.

We now seek an alternative with minimal communication. It instead uses one or two signals that can be broadcast using power line communication [19], and implicit measurement of the jobs' loads. The main signal is $p(t)$, a congestion measure. One variant also signals $g(t)$, how much capacity is unused. These determine how much power each device should draw.

1) Exact congestion signals: We first investigate whether $p(t)$ is sufficient, *if* $p(t)$ is chosen so that devices consume exactly the available power, $G(t)$. In our example, device i consumes

$$x_i(t) = D_i(p(t)) = B_i \exp(-p(t) r_i(t)). \tag{7}$$

The "demand function", $D_i(\cdot)$ determines the allocation. This provides soft prioritization of jobs with small residual times, r, rather than the hard prioritization of the centralized schedulers. Rather than serving the load with the shortest remaining energy exclusively, this shares the generation among all loads and simply provides a weighting towards loads with short remaining energy.

Making these weights very disparate increases prioritization, which should improve performance, but also causes the congestion signal $p(t)$ to become very large when a flow is nearly finished, and drop suddenly when the demand is finally met. This is problematic for algorithms that use an estimate, $\hat{p}(t)$. With less prioritization, the fluctuations are more tractable, but less priority is given to the nearly finished loads. This appears to be an unavoidable trade off.

The response time for this ideal case is shown in Fig. 4. These show that the performance of the distributed scheduler with an accurate value for $p(t)$ is comparable to that of the best centralized schedulers.

The challenge is how estimating $p(t)$ in a decentralized manner. The rest of this paper explores increasingly sophisticated attempts at doing that.

2) Naive distributed algorithm: The first distributed approach for finding p is taken from network utility maximization in Internet congestion control [12]. It notes that $x(p)$ is the solution to

$$\max \sum\nolimits_{i \in \mathcal{I}_t} \int_0^x D_i^{-1}(\xi)\, d\xi \tag{8}$$

$$s.t. \sum\nolimits_{i \in \mathcal{I}_t} x_i(t) \le G(t) \tag{9}$$

where D_i is the demand function in (7), and $p(t)$ is the Lagrange multiplier of the constraint. Thus, searching for the optimal p is a equivalent to a dual optimization algorithm for this problem.

The scheduler starts with an initial estimat $\hat{p}(0)$. Being at the generator, it knows the available generation, and measures the total power draw of all devices. It then updates $\hat{p}$ using gradient descent and broadcasts it to all devices, which apply (7). The estimate $\hat{p}$ is then updated and the process repeated.

Under this naive approach, the system will repeatedly become overloaded as r_i decreases. The simplest remedy is to replacing $G(t)$ in (9) by $\beta G(t)$ with $\beta < 1$. In Internet terminology, this is using a "virtual queue" [20].

This model was evaluated given a solar generation profile $G(t)$, with $G_b = 500\,\mathrm{W}$ and load $\rho = 1$. Figures 5(a) and (b) show the power allocation when $\beta = 0.9$, and the amount of overload for $\beta \in [0.9, 0.99]$. The latter shows that even targeting a utilization as low as 90% results in unacceptable overload. This is because the ratio of standard deviation to mean of the demand in under (8)–(9) is high. Next, we address that directly.

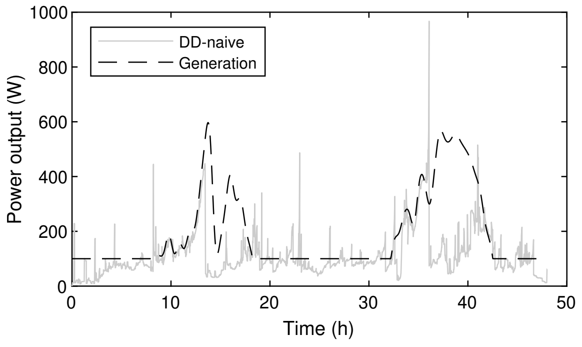
3) Capping new flows: The peaks of demand occur when a new demand arrives. If there were N demands, then the new demand is around $1 + 1/N$ times higher. The new rate then exceeds generation unless $N > \beta/(1-\beta)$. To avoid this, a new demand should initially receive no more than the gap between the generation and the sum of the existing demands.

We thus broadcast an additional signal: a direct measure of the supply and demand mismatch (or *gap*), defined as

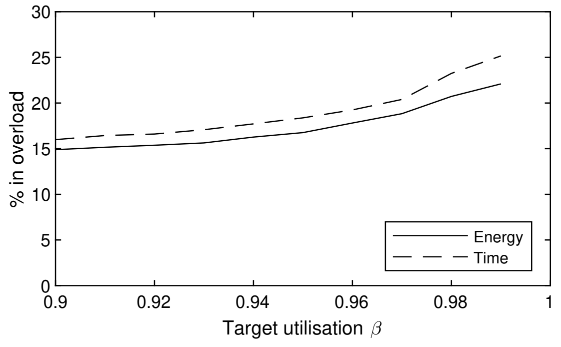
$$g(t+\delta) = G(t) - \sum\nolimits_{i \in \mathcal{I}} x_i(t) - x_0(t). \tag{10}$$

Every load i maintains a record of the accumulated history of gaps from arrival time a_i until t, by $c_i(t) = c_i(t-\delta) + g(t)$.

Each load receive $\hat{p}(t)$ and $g(t)$ and calculates a target rate $\hat{x}_i(t)$ by (8). The power consumption is then capped by $x_i(t) = \min(\hat{x}_i(t), c_i(t))$.

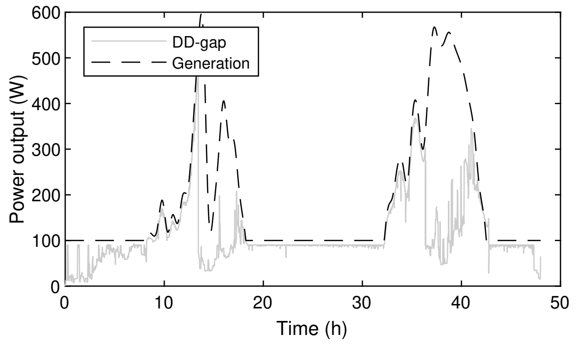


(a) Power output vs time.

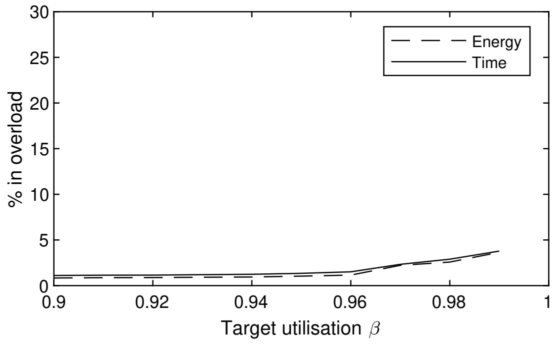


(b) Energy/Time overload vs target utilization.

Fig. 5. Power output and overload CDF for DD-naive.



(a) Power output vs time, $\rho = 0.6$, $\beta = 0.9$.



(b) Energy/Time overload vs target utilization, $\rho = 0.6$.

Fig. 6. Power output and overload CDF for DD-gap

A load's initial cap is $c_i(a_i) = g(a_i)$ aiming to prevented overshoot before the existing loads have responded to increased p. As $c_i(t)$ increases, loads will stop being capped, making the allocation $\hat{x}_i(t)$ predominant. This capping reduces both spikes in x as soon as a load arrives and spikes in $\hat{p}$ due to rates adapting, which aids the gradient descent.

However, the capping results in oscillations. Let $\mathcal{J} \subseteq \mathcal{I}_t$ be the set of loads such that $x_i(t) = c_i(t)$. Oscillations occur when the gap $g(t)$ is allocated to multiple capped demands, that is, $|\mathcal{J}| > 1$. Each capped load increases its power draw by $g(t)$, causing a negative gap $g(t+\delta)$. The congestion signal $\hat{p}$ increases to cancel the overshoot. Requests governed by $\hat{p}$ will lower their consumption and the system will return to a state similar to that before the overshoot, and the cycle repeats.

Steady state resumes when $|\mathcal{J}| \leq 1$, i.e., requests' $c_j(t)$ values have grown to exceed $\hat{x}_j(t)$.

4) Gain-controlled feedback: To avoid oscillations, the gap $g(t)$ must be shared among all requests $j \in \mathcal{J}$. To support this, the controller maintains an *estimate* of $|\mathcal{J}|$. It is initialized to $h(0) = 1$ and increased by one if the most up to date gap reading is negative, $g(t) < 0$. Otherwise, it will decrease it slowly, clipped below at 1. The estimate is

$$h(t) = \begin{cases} \max(1, 0.99h(t-\delta)) & \text{if } g(t) \geq 0 \\ h(t-\delta) + 1 & \text{if } g(t) < 0. \end{cases} \quad (11)$$

The controller now broadcasts a scaled gap: $q(t) = g(t)/h(t)$.

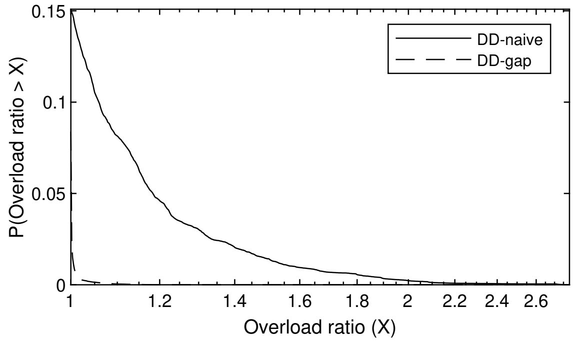
Loads calculate their new caps as $c_i(t) = c_i(t-\delta) + q(t)$ and are initialized by $c_i(a_i) = q(a_i)$.

This controller, which we call DD-gap, was tested with the same parameters as previous attempts. As shown in Fig. 6, it significantly reduces overshoot. At the bottom of the figure, all decentralized schemes are compared in terms of energy and time overload (keeping the workload constant while changing β); showing that the feedback control strategy further reduces both the time the system is in overload, and also the magnitude.

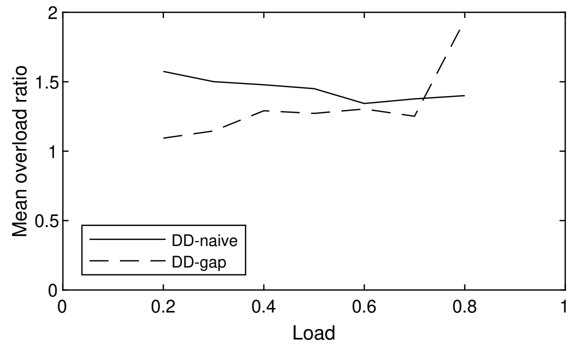
Figure 7(a) shows the probability of overload events exceeding a given power threshold. The probability of exceeding the generation is quite low for DD-gap, as the opposite occurs for the naive version. In 7(b), DD-gap shows a clear advantage over DD-naive in terms of the ratio between power output (consumption) and input (generation). The overload magnitude is usually much higher for the naive version. However, at higher loads, DD-gap suffers from instability issues, creating overshoots due to oscillations in the congestion signal.

D. Future directions

The two main challenges are accommodating new flows that arrive, and the rapid increase in aggressiveness as r_i tends to zero, which is the feature that allows the scheme to surpass B-PS, followed by a sudden drop off in demand. A promising approach to the latter is for the scheduler to attempt to track the value of r_i (and B_i) of the most aggressive flow. This would allow the scheduler to set p based on the estimated



(a) Overload ratio CDF, $\rho = 0.6$, $\beta = 0.9$.



(b) Mean overload ratio vs load, $\beta = 0.9$.

Fig. 7. Overload ratio CDF and mean ratio vs load for DD-naive and DD-gap.

value of r_i at the next time step, instead of the previous time step. This would be a higher value of p, preventing overload in the lead up to a job finishing. It also allows the scheduler to know that a job has just finished, and so to decrease p rapidly, avoiding wasted capacity.

V. Conclusions

This paper has investigated the feasibility of "stretching" electric loads that require a constant energy, but can vary their power, such as heating water. This would be used in conjunction with scheduling that shifts fixed-power loads. This is similar to a single or multi-server queue, but it differs in that individual devices have different constraints on the maximum rate at which they can draw power, and so new scheduling disciplines have been developed and assessed.

It was shown that schedulers reliant on the notion of (remaining) "service time" should be interpreted in terms of (remaining) work instead. The resulting generalization of RS minimized slowdown among disciplines tested, and tied with the generalization of SRPT for the best response time. These outperformed the generalization of PS in both metrics.

A more ambitious goal is to perform efficient scheduling without requiring the residual work and power rating of each job to be communicated to the scheduler. The latter part can be effectively solved by broadcasting a congestion signal. Devices then draw an amount of power that decreases with the level of congestion, but increases as they near completion. However, the former part remains an open problem.

This study has been mainly empirical, but has opened up interesting new theoretical avenues, by proposing a generalization of the notion of a k-server system. Understanding such systems may yield better schedulers than those considered here, or may shed light on the difficulties of distributed implementations.

Acknowledgment

This work was funded in part by Australian Research Council grant DP190102134.

References

[1] A. Majzoobi and A. Khodaei, "Application of microgrids in providing ancillary services to the utility grid," *Energy*, vol. 123, pp. 555–563, 2017.

[2] W. Su, J. Wang, and J. Roh, "Stochastic energy scheduling in microgrids with intermittent renewable energy resources," *IEEE Trans. Smart Grid*, vol. 5, no. 4, pp. 1876–1883, 2014.

[3] L. Gelazanskas and K. A. Gamage, "Demand side management in smart grid: A review and proposals for future direction," *Sustainable Cities and Society*, vol. 11, pp. 22–30, 2014.

[4] N. Li, L. Chen, and S. H. Low, "Optimal demand response based on utility maximization in power networks," in *IEEE Power and Energy Society General Meeting*, 2011, pp. 1–8.

[5] Y. Wang, S. Mao, and R. M. Nelms, "Centralized online algorithm for optimal energy distribution in connected microgrid," in *Online Algorithms for Optimal Energy Distribution in Microgrids*. Springer, 2015, pp. 31–61.

[6] L. Gan, U. Topcu, and S. H. Low, "Optimal decentralized protocol for electric vehicle charging," *IEEE Trans. Power Systems*, vol. 28, no. 2, pp. 940–951, 2013.

[7] L. Schrage, "A proof of the optimality of the shortest remaining processing time discipline," *Op. Res.*, vol. 16, no. 3, pp. 687–690, 1968.

[8] I. Grosof, Z. Scully, and M. Harchol-Balter, "SRPT for multiserver systems," *Performance Evaluation*, vol. 127, pp. 154–175, 2018.

[9] M. Harchol-Balter, *Performance modeling and design of computer systems: queueing theory in action*. Cambridge University Press, 2013.

[10] S. Yang and G. De Veciana, "Size-based adaptive bandwidth allocation: Optimizing the average QoS for elastic flows," in *Proc. INFOCOM*. IEEE, 2002, pp. 657–666.

[11] F. Kelly, "Charging and rate control for elastic traffic," *Trans. Emerging Telecommunications Technologies*, vol. 8, no. 1, pp. 33–37, 1997.

[12] S. H. Low and D. E. Lapsley, "Optimization flow control I: Basic algorithm and convergence," *IEEE/ACM Trans. Networking*, vol. 7, no. 6, pp. 861–874, 1999.

[13] E. Hyytiä, S. Aalto, and A. Penttinen, "Minimizing slowdown in heterogeneous size-aware dispatching systems," *SIGMETRICS Perform. Eval. Rev.*, vol. 40, no. 1, p. 2940, Jun. 2012.

[14] J. Z. Kolter and M. J. Johnson, "REDD: A public data set for energy disaggregation research," in *SIGKDD Workshop on data mining applications in sustainability*, 2011, pp. 59–62.

[15] E. L. Ratnam, S. R. Weller, C. M. Kellett, and A. T. Murray, "Residential load and rooftop PV generation: an australian distribution network dataset," *Int. J. Sustainable Energy*, vol. 36, no. 8, pp. 787–806, 2017.

[16] E. Hyytiä, S. Aalto, and A. Penttinen, "Minimizing slowdown in heterogeneous size-aware dispatching systems," *ACM SIGMETRICS Performance Evaluation Review*, vol. 40, no. 1, pp. 29–40, 2012.

[17] R. Righter, J. G. Shanthikumar, and G. Yamazaki, "On extremal service disciplines in single-stage queueing systems," *J. Applied Probability*, vol. 27, no. 2, pp. 409–416, 1990.

[18] S. He, M. Wallace, G. Gange, A. Liebman, and C. Wilson, "A fast and scalable algorithm for scheduling large numbers of devices under real-time pricing," in *Int. Conf. Principles Practice Constraint Programming*. Springer, 2018, pp. 649–666.

[19] L. T. Berger, A. Schwager, and J. J. Escudero-Garzás, "Power line communications for smart grid applications," *Journal of Electrical and Computer Engineering*, vol. 2013, 2013.

[20] S. S. Kunniyur and R. Srikant, "An adaptive virtual queue (avq) algorithm for active queue management," *IEEE/ACM Transactions on networking*, vol. 12, no. 2, pp. 286–299, 2004.