Minimal-Variance Distributed Deadline Scheduling in a Stationary Environment

Yorie Nakahira
Caltech
ynakahir@caltech.edu

Andres Ferragut [*]
Universidad ORT Uruguay
ferragut@ort.edu.uy

Adam Wierman
Caltech
adamw@caltech.edu

ABSTRACT

Many modern schedulers can dynamically adjust their service capacity to match the incoming workload. At the same time, however, variability in service capacity often incurs operational and infrastructure costs. In this paper, we propose distributed algorithms that minimize service capacity variability when scheduling jobs with deadlines. Specifically, we show that *Exact Scheduling* minimizes service capacity variance subject to strict demand and deadline requirements under stationary Poisson arrivals. We also characterize the optimal distributed policies for more general settings with soft demand requirements, soft deadline requirements, or both. Additionally, we show how close the performance of the optimal distributed policy is to that of the optimal centralized policy by deriving a competitive-ratio-like bound.

Keywords

Deadline scheduling, Service capacity control, Exact Scheduling, Online algorithms

1. INTRODUCTION

Traditionally, the scheduling literature has assumed a static or fixed service capacity. However, it is increasingly common for modern applications to have the ability to dynamically adjust their service capacity in order to match the current demand. For example, power distribution networks match the energy supply demand as it changes over time. When using cloud computing services, one can modify the total computing capacity by changing the number of computing instances and their speeds.

The ability to adapt service capacity dynamically gives rise to challenging new design questions. In particular, how to reduce the *variability* of service capacity is of great importance in such applications since peaks and fluctuations often come with significant costs [7, 24, 31].

For example, the emerging load from electric vehicle charging stations leads to challenges in power distribution networks. Charging stations require stability in power consumption because fluctuations and large peaks in power use may strain the grid infrastructure and result in a high peak charge for the station operators. The stations also prefer predictable power consumption because purchasing power in real time is typically more expensive than purchasing in advance. Cloud content providers also prefer stable and predictable service capacity because on-demand contracts for compute instances (e.g., Amazon EC2 and Microsoft Azure) are typically more expensive than long-term contracts. Additionally, significant fluctuations in service capacity induce unnecessary power consumption and infrastructure strain for computing equipment.

Thus, in situations where service capacity can be dynamically adjusted, an important design goal is to reduce the costs associated with variability in the service capacity while maintaining a high quality of service. In this work, we study this problem by minimizing the *variance* of the service capacity in systems where jobs arrive with demand and deadline requests. Our focus on variance is motivated by power distribution networks, where the size of jobs and (active) service capacity are small compared to the total energy resources available and where contracts often explicitly depend on service capacity variability, e.g., if a charging station participates in the regulation market, then costs/payments rely explicitly on the variance of the total capacity [3, 30].

Because of the scale of the service systems considered, the goal of this work is to design *distributed* scheduling algorithms that minimize the variance of service capacity subject to service quality constraints, e.g., meeting job deadlines and satisfying job demands. Our focus is on distributed scheduling algorithms that use only local information about each job to decide the service rate, since implementing centralized algorithms is likely to be prohibitively slow and costly in large-scale service systems. Power distribution networks and cloud computing are unlikely to be able to access global information about all jobs and servers in the system in real time when deciding the service rate of individual jobs.

Related work. Although the literature on deadline scheduling is large and varied, optimal algorithms are only known for certain niche cases. Examples of classic scheduling algorithms include Earliest Deadline First [17, 25] and Least Laxity First [17], among others. Beyond these classic algorithms, more modern algorithms simultaneously perform admission control and service rate control in order to exploit the flexibility arising from soft demand or deadline requirements, e.g., [8, 23, 29].

The trade-offs between service quality and costs associated with variability have become a focus only recently, but already many exciting results have appeared, contrasting the performance of classical algorithms, e.g., [6, 11, 12]. These issues have also been studied extensively in the areas

[*] A. Ferragut was partially supported by ANII-Uruguay under grants FSE_1_2016_1_131605 and FCE_1_2017_1_136748

IFIP WG 7.3 Performance 2018 Toulouse, France

of cloud computing and power distribution systems. Algorithms for cloud computing have been proposed to control the variability of power usage in data centers while achieving high service quality for deferrable jobs (see [10,14,22,33] and the references therein). Algorithms for power distribution systems have been designed to reduce the strain on the power grid while supplying energy demand to deferrable loads (see [5,9,13,26,27,32] and the references therein).

Interesting optimality results have been obtained in some limited settings, such as deterministic worst-case settings [2], single server systems [4, 28], and/or heavy traffic settings [16, 20]. For example, in heavy traffic settings, the dynamic behavior of discrete queueing systems can be approximated by a continuous-state process involving Brownian motion, for which there exist established tools to optimize [16, 20]. On the other hand, optimizing queueing systems without continuous-state approximations remains to be a hard problem. In particular, the problem of designing *optimal* algorithms that minimize service capacity variability while achieving high service quality has remained open. Solving this problem is a challenging task due to the heterogeneity of jobs (diversity in service requests) and the size of the state and decision space (numbers of possible configurations on existing job profiles and the set of feasible control policies).

Contributions of this paper. In this paper, we adapt tools from optimization and control theory to characterize the optimal distributed policies in a broad range of settings without any approximations. Further, we provide a novel, competitive-ratio-like bound that describes the gap between the performance of an optimal distributed policy and the performance of an optimal centralized policy.

2. SYSTEM MODEL

The goal of this paper is to characterize the minimal-variance online scheduling policies for systems with the ability to dynamically adjust their service capacity. We use a continuous time model where $t \in \mathbb{R}$ denotes a point in time. Each job, indexed by $k \in \mathcal{V} = \{1, 2, \dots\}$, is characterized by a random arrival time a_k, a random service demand σ_k, and a random sojourn time τ_k. The deadline of job k is then defined as $a_k + \tau_k$. In order to formulate the scheduler design problem, we introduce the arrival profiles, the service profiles, the system dynamics, and the design objectives below.

Arrival profiles. We represent the set of arriving jobs as a marked point process $\{(a_k; \sigma_k, \tau_k)\}_{k\in\mathcal{V}}$ in $\mathbb{R} \times S$, where the arrival times $a_k \in \mathbb{R}$ are the set of points, and the service requirements $(\sigma_k, \tau_k) \in S$ are the set of marks. We assume that the marked point process is a stationary independently marked Poisson Point Process, which is defined by rate Λ and a mark density measure $f(\sigma, \tau)$ on S [1].[1] This also implies that $\{(a_k; \sigma_k, \tau_k)\}_{k\in\mathcal{V}}$ is a Poisson point process on $\mathbb{R} \times S$ with an intensity function $\Lambda f(\sigma, \tau)$. Intuitively, $\Lambda \int_A f(\sigma, \tau) d\sigma d\tau$ is the rate at which jobs with service requirement $(\sigma, \tau) \in A \subset S$ arrive. We additionally assume that S is bounded, $S \subset \{(\sigma, \tau) : \tau \geq \sigma \geq 0\}$[2], and the service demand σ and sojourn time τ has finite first and second moments.

Service profiles. The service system works on each job $k \in \mathcal{V}$ with a *service rate* $r_k(t) \geq 0$. To meet the service demand of job k, its service rate must satisfy

$$\int_{a_k}^{\infty} r_k(t) dt = \sigma_k, \qquad k \in \mathcal{V}, \tag{1}$$

where $r_k(t)$ is assumed to be an integrable function of t. Moreover, the service rate can take non-zero values only when the job sojourns in the system, *i.e.*, $r_k(t) = 0$ for any $t \notin [a_k, a_k + \tau_k)$. Without loss of generality, $r_k(t) = 1$ is assumed to be the maximum rate: that is, $r_k(t)$ can take any values in $[0, 1]$, and $r_k < 1$ corresponds to throttling down service speed at the expense of prolonging job completion times. The above sojourn time and maximum rate constraints can be jointly written as

$$0 \leq r_k(t) \leq \mathbf{1}_{\{t\in[a_k, a_k+\tau_k)\}}, \tag{2}$$

where $\mathbf{1}_A$ denotes the indicator function for an event A. The service capacity is defined by

$$P(t) = \sum_{k\in\mathcal{V}} r_k(t),$$

which is associated with the instantaneous resource consumption of the service system. Note that we assume that $P(t)$ has no upper bound. This means that there is always enough capacity to serve the jobs and, thus, there is no queueing in the system.

System dynamics. At each time $t \in \mathbb{R}$, job k has a remaining demand $x_k(t) = \sigma_k - \int_{a_k}^{t} r_k(h) dh$ and a remaining time $y_k(t) = a_k + \tau_k - t$. The set of remaining jobs in the system can be considered as a point process $\{(x_k(t), y_k(t))\}_k$ in $\mathbb{R}^2$, where the first coordinate (x) represents the remaining demand and the second coordinate (y) represents the remaining time. At time t, each point (job) has velocity $-r_k(t)$ in the direction of x-coordinate and velocity -1 in the direction of y-coordinate.

Scheduling algorithms. An online scheduling algorithm decides the service rates in real-time without using the future job arrival information. For scalability, we additionally restrict our attention to the following form of *distributed* algorithms which decide the service rate of a job only using its own information:

$$r_k(t) = u(x_k(t), y_k(t)) \quad \in [0, 1]. \tag{3}$$

Here, $u : \mathbb{R}_+ \times \mathbb{R} \to \mathbb{R}_+$ is an integrable deterministic function of the remaining demand $x_k(t)$ and the remaining time $y_k(t)$ of each job k at time t. The policy u also uniquely determines the vector field in the space of the point process $\{(x_k(t), y_k(t))\}_k$, which in turn defines the velocity $(-u(x, y), -1)$ of points (jobs) at (x, y).

We study policies of the form (3) assuming a situation where there is enough capacity available to satisfy the demand ($P(t)$ may be scaled up arbitrarily), and so the focus is on determining the optimal service rate for the jobs in a distributed manner.

Under any policy of the form (3), the set of jobs remaining in the system converges to a stationary distribution. This stationary distribution is a spatial Poisson point process with an intensity function $\lambda(x, y)$ satisfying

$$0 = \frac{\partial}{\partial x}(\lambda(x, y)u(x, y)) + \frac{\partial}{\partial y}\lambda(x, y) + \Lambda f(x, y), \tag{4}$$

[1] Here, we use $(a; \sigma, \tau)$ to denote random variables and $(a_k; \sigma_k, \tau_k)$ to denote one realization of them in job k.

[2] The condition $\tau_k \geq \sigma_k$ requires each job $k \in \mathcal{V}$ to have a service demand σ_k that is no more than the maximum service that can be provided within its sojourn time τ_k.

where x is the remaining demand and y is the remaining time. Equation (4) is derived from the *continuity equation*, which describes the transport of a 'mass' [12,34].[3] Because the remaining job distribution converges to a stationary distribution, $P(t)$ also converges to a stationary distribution.[4]

Design objectives. We consider minimizing service capacity variability for the settings with hard service constraints, soft demand constraints, soft deadline constraints, and soft demand and deadline constraints. In the case of *strict demand constraints*, we consider the following optimization problem:

$$\underset{u:(1)(2)(3)(4)}{\text{minimize}} \qquad \mathrm{Var}(P), \tag{5}$$

where $\mathrm{Var}(P)$ is a functional of u and $\lambda(\sigma,\tau)$ that satisfies (4). The optimization problem (5) has demand constraints as in (1) and deadline constraints as in (2). The constraint (3) restricts the optimization variable u to be distributed.

In the case of *soft demand constraints*, we relax the demand requirements (1) and penalize the amount of unsatisfied demands with a unit cost δ. In this setting, we consider balancing the service capacity variance and the expected cost for unsatisfied demands:

$$\underset{u:(2)(3)(4)}{\text{minimize}} \qquad \mathrm{Var}(P) + \mathbb{E}[\delta U], \tag{6}$$

where $U(t) = \sum_{k\in\mathcal{V}:a_k+\tau_k=t} x_k(t)$ is the total amount of remaining demands for jobs departing at time t.

In the case of *soft deadline constraints*, we relax the deadline requirements (2) and penalize deadline extensions with a unit cost ϵ. Let $\hat{\tau}_k$ be the actual sojourn time of job $k \in \mathcal{V}$:

$$0 \le r_k(t) \le \mathbf{1}_{\{t\in[a_k,a_k+\hat{\tau}_k)\}}. \tag{7}$$

So $\hat{\tau}_k - \tau_k$ is the duration of deadline extension, and let $W(t) = \sum_{k\in\mathcal{V}:a_k+\hat{\tau}_k=t} \hat{\tau}_k - \tau_k$ be the total duration of deadline extensions for jobs departing at time t. We consider balancing the service capacity variance and the expected cost for deadline extensions:

$$\underset{u:(1)(3)(4)(7)}{\text{minimize}} \qquad \mathrm{Var}(P) + \mathbb{E}[\epsilon W]. \tag{8}$$

In the case of *soft demand and deadline constraints*, we relax both the demand requirements (1) and the deadline requirements (2). The system needs to pay a cost of δ for each unit of unsatisfied demands and a cost of ϵ for each unit of deadline extensions. In this setting, we consider balancing the service capacity variance, the expected cost for unsatisfied demands, and the expected cost for deadline extensions:

$$\underset{u:(3)(4)(7)}{\text{minimize}} \qquad \mathrm{Var}(P) + \mathbb{E}[\delta U] + \mathbb{E}[\epsilon W]. \tag{9}$$

Generalizing above cases, we consider the case where the unit costs for unsatisfied demands and deadline extensions are heterogeneous among jobs. Let δ_k be the unit cost for the unsatisfied demand of job $k \in \mathcal{V}$, and ϵ_k be the unit cost for its deadline extension. The set of jobs is assumed to be a independently marked Poisson point process $\{(a_k;\sigma_k,\tau_k,\delta_k,\epsilon_k)\}_{k\in\mathcal{V}}$, where the unit costs $(\delta_k,\epsilon_k) \in \mathbb{R}_+^2$ are the additional marks of jobs. We assume that (δ_k,ϵ_k) are identically distributed random variables with a joint density measure $f(\delta)f(\epsilon)$ (hence independent from each other as well) and are also statistically independent from $(a_k;\sigma_k,\tau_k)$. To account for the heterogeneous costs, we consider scheduling policies of the form

$$r_k(t) = \bar{u}(x_k(t), y_k(t), \delta_k, \epsilon_k) \ge 0 \tag{10}$$

and the optimization problem

$$\min_{\bar{u}:(7)(10)} \mathrm{Var}(P(t)) + \mathbb{E}\left[\sum_{\substack{k\in\mathcal{V}:\\ a_k+\hat{\tau}_k=t}} \delta_k x_k(t)\right] + \mathbb{E}\left[\sum_{\substack{k\in\mathcal{V}:\\ a_k+\hat{\tau}_k=t}} \epsilon_k(\hat{\tau}_k - \tau_k)\right]. \tag{11}$$

Motivating examples. The general model we have defined is meant to give insight into the design trade-offs that happen in applications with dynamic capacity, e.g., electric vehicle charging or cloud content providers while exploring design trade-offs using a simple, general model.

However, to highlight the connection to our motivating examples, consider first the case of electric vehicle charging [18]. In this case, each job $k \in \mathcal{V}$ corresponds to an electric vehicle with an arrival time a_k, an energy demand σ_k, and a sojourn time τ_k. At each time t, the charging station provides vehicle k with a charging rate of $r_k(t)$ by drawing $P(t) = \sum_{k\in\mathcal{V}} r_k(t)$ amount of power from the grid. When doing so, a stable resource usage is highly desirable because fluctuations and large peaks in $P(t)$ can lead to a high peak charge or strain the grid. Moreover, a predictable resource use is also important when purchasing energy from the day-ahead market, whose price is lower and less volatile than that of the real-time market. Note that our model assumes $P(t)$ is unbounded and, thus, corresponds to a setting where there are more charging stations than arriving cars. It is important to relax this assumption and consider the case where $P(t)$ is limited in future work.

In the case of cloud content providers, each job $k \in \mathcal{V}$ corresponds to a task (requested to the cloud or data centers) with an arrival time a_k, a work requirement σ_k, and an allowable waiting time τ_k. The service system works on job k with speed $r_k(t)$ using $P(t) = \sum_{k\in\mathcal{V}} r_k(t)$ number of computers (or amount of power). Thus, given a good estimate of the future resource use, a cloud content provider can reserve resources through a long-term contract, whose price is lower and less volatile than that of a short term contract. This motivates its scheduling algorithm to achieve a predictable resource use. Note that our model considers the case where $P(t)$ is unbounded and, thus, the data center has enough capacity to avoid congestion, i.e., is in low utilization. Such periods are common, since data centers often operate at utilizations as low as 10% [15], but managing congestion and queueing is crucial in data center operations and this is not captured in our model. The importance of managing congested periods highlights the need for future work to consider how an upper bound on $P(t)$ impacts the results presented here.

[3] In our setting, the 'mass' is the probability density. When the system is stationary, the mass (probability density) is conserved over time, as in (4).

[4] In this paper, we use the following notation: $\mathbb{E}[P]$ and $\mathrm{Var}(P)$ represent the stationary mean and variance of a stochastic process $\{P(t)\}_{t\in\mathbb{R}}$, while $\mathbb{E}[P(t)]$ and $\mathrm{Var}(P(t))$ represent the instantaneous mean and variance of $P(t)$ at time t.

3. OPTIMAL DISTRIBUTED ALGORITHMS

In this section, we characterize the optimal distributed scheduling policies in a wide range of settings, starting with the simplest and moving toward the most complex. To begin, we focus on strict service requirements and show that Exact Scheduling minimizes the stationary variance of the service capacity (Section 3.1). Relaxing the demand requirements, we show that a variation of Exact Scheduling minimizes the weighted sum of both the stationary variance of the service capacity and the penalty for unsatisfied demand (Section 3.2). Relaxing the deadline requirements, we show that a different variation of Exact Scheduling minimizes the weighted sum of both the stationary variance of the service capacity and the penalty for demand extension (Section 3.2). Finally, we consider the case when both the demand and deadline requirements are relaxed (Section 3.4) and show that the optimal policy becomes significantly more complex in this case. However, note that all the optimal algorithms we identify are in closed-form, and thus provide clear interpretations and insights regarding the optimal trade-offs between reducing service capacity variability, satisfying the demands, and meeting deadlines. Moreover, it is interesting that the minimum service capacity variance is achieved by these simple algorithms, all of which are extremely scalable and easy to implement.

3.1 Strict demand and deadline requirements

We first consider the case of strict service requirements and show a closed-form formula of the algorithm that minimizes the stationary variance $\mathrm{Var}(P)$. To do so, it is worth noting that peaks in service rate amplifies the uncertainties in the future arrivals, which in turn produces large variance in $P(t) = \sum_k r_k(t) = \sum_k u(x_k(t), y_k(t))$. In order to minimize peaks subject to strict service requirements, one can consider using a flat service rate, which is achieved by the scheduling policy

$$u(x,y) = \begin{cases} \frac{x}{y}, & \text{if } y > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

This policy is known as Exact Scheduling and works by finishing all jobs *exactly* at their deadlines using constant service rates. It is also highly scalable because it is distributed and asynchronous, and it does not require much computation or memory use. Although existing literature has analyzed its performance in various settings [7,12,19,21], optimality guarantees have been difficult to obtain. In this section, we show that Exact Scheduling minimizes the variance of service capacity under time-homogeneous job arrivals and strict demand constraints.

Theorem 1. *Exact Scheduling* (12) *is the optimal solution of* (5) *and achieves the optimal value*

$$\mathrm{Var}(P) = \Lambda \mathbb{E}\left[\frac{\sigma^2}{\tau}\right].$$

Theorem 1 shows the achievable performance improvement by performing distributed service capacity control. If no control is applied, then $r_k(t) = \mathbf{1}_{[a_k, a_k+\sigma_k]}(t)$, and the stationary mean and variance of $P(t)$ is $\mathbb{E}(P) = \mathrm{Var}(P) = \Lambda\mathbb{E}[\sigma]$. By performing a distributed service capacity control, the stationary variance can be reduced by

$$\Lambda \mathbb{E}\left[\frac{\sigma(\tau-\sigma)}{\tau}\right] \in [0, \Lambda\mathbb{E}[\sigma]]$$

where $\tau - \sigma$ is the slack time (the amount of time left at job completion if a job is served at its maximum service rate since it arrives).

3.2 Soft demand requirements

In this section, we relax the strict service requirements and characterize the optimal algorithm under soft demand constraints. Specifically, we consider the setting when the system needs to pay a cost δ for each unit of unsatisfied demands. When this unit cost is sufficiently large, we recover the case of strict service requirements. The optimal algorithm we identify is a generalization of Exact Scheduling:

$$u(x,y) = \begin{cases} \frac{x}{y}, & \text{if } \frac{x}{y} \le \frac{\delta}{2},\ y > 0, \\ \frac{\delta}{2}, & \text{if } \frac{x}{y} > \frac{\delta}{2},\ y > 0, \\ 0, & \textit{otherwise}. \end{cases} \tag{13}$$

We call (13) *Rate-limited Exact Scheduling.* This policy essentially sets $\delta/2$ to be the upper bound on service rates. Under this policy, job k receives its full service demand if $\sigma_k \le \delta\tau_k/2$ but otherwise is provided with the partial service demand of $\delta\tau_k/2$. To the best of our knowledge, this algorithm has not been proposed in the existing literature.

Theorem 2. *Rate-limited Exact Scheduling* (13) *is the optimal solution of* (6) *and achieves the optimal value*

$$\Lambda\mathbb{E}\left[\frac{\sigma^2}{\tau}\mathbf{1}_{\left\{\frac{\sigma}{\tau}\le\frac{\delta}{2}\right\}} + \delta\left(\sigma - \frac{\delta\tau}{4}\right)\mathbf{1}_{\left\{\frac{\sigma}{\tau}>\frac{\delta}{2}\right\}}\right]. \tag{14}$$

Theorem 2 shows the performance improvement gained by relaxing the demand requirements. If some demands do not have to be satisfied, the stationary variance can be reduced from $\mathrm{Var}(P) = \mathbb{E}\left[\sigma^2/\tau\right]$ to (14) when the service rate threshold is set to its optimal value $\delta/2$.

3.3 Soft deadline requirements

The previous section shows the optimal algorithm under soft demand requirements. In this section, we characterize the optimal distributed algorithm under soft deadline requirements. Specifically, we consider the setting when the system needs to pay a cost ϵ for each unit of deadline extensions. When the unit cost is sufficiently large, this setting recovers the case of strict deadline requirements. The resulting optimal algorithm is again a generalization of Exact Scheduling:

$$u(x,y) = \begin{cases} \frac{x}{y}, & \text{if } \frac{x}{y} \le \sqrt{\epsilon} \text{ and } y > 0, \\ \sqrt{\epsilon}\,\mathbf{1}_{\{x>0\}}, & \text{otherwise.} \end{cases} \tag{15}$$

We call (15) *Deadline-extended Exact Scheduling.* This policy essentially sets an upper bound $\sqrt{\epsilon}$ to service rates. Under this policy, the deadline of job k is extended when $\sigma_k > \sqrt{\epsilon}\tau_k$.

Theorem 3. *Deadline-extended Exact Scheduling* (15) *is the optimal solution of* (8) *and achieves the optimal value*

$$\Lambda\mathbb{E}\left[\frac{\sigma^2}{\tau}\mathbf{1}_{\left\{\frac{\sigma}{\tau}\le\sqrt{\epsilon}\right\}} + \left(2\sqrt{\epsilon}\sigma - \epsilon\tau\right)\mathbf{1}_{\left\{\frac{\sigma}{\tau}>\sqrt{\epsilon}\right\}}\right].$$

Theorem 3 shows the performance improvement by relaxing the deadline requirements. If all deadline must be satisfied, then $\mathrm{Var}(P) = \Lambda\mathbb{E}\left[\sigma^2/\tau\right]$ is the minimum stationary variance achievable. If some deadlines do not have to be satisfied, the stationary variance can be further reduced at the expense of paying a penalty for deadline extensions. The service rate threshold $\sqrt{\epsilon}$ strikes the optimal balance between minimizing $\mathrm{Var}(P)$ and minimizing $\mathbb{E}[\epsilon W]$.

3.4 Soft demand and deadline requirements

In this section, we consider relaxing both demand and deadline requirements simultaneously and characterize the optimal distributed algorithm. Specifically, we consider the setting when the system needs to pay a cost δ for each unit of demand extensions and a cost ϵ for each unit of deadline extensions. This setting recovers all previous settings as special cases.

Recall from previous sections that, under soft demand requirements, the optimal policy uses a constant service rate and reject partial demand requests if $\sigma/\tau > \delta/2$. Meanwhile, under soft deadline requirements, the optimal policy uses a constant service rate and extends the deadline if $\sigma/\tau > \sqrt{\epsilon}$. These two special cases suggest that, under soft demand and deadline requirements, a constant service rate combined with demand rejection and deadline extension may work well. This is indeed the case, as formalized below.

Theorem 4. *The optimal solution of* (9) *is*

$$u(x,y) = \begin{cases} \dfrac{x}{y}, & \text{if } y>0,\ \dfrac{x}{y} \le \min\left\{\dfrac{\delta}{2}, \sqrt{\epsilon}\right\}, \\ \dfrac{\delta}{2}, & \text{if } y>0,\ \dfrac{x}{y} > \dfrac{\delta}{2} \text{ and } \dfrac{\delta}{2} \le \sqrt{\epsilon}, \\ \sqrt{\epsilon}\,\mathbf{1}_{\{x>0\}}, & \text{otherwise}, \end{cases} \tag{16}$$

and it achieves the optimal value

$$\Lambda\mathbb{E}\left[\frac{\sigma^2}{\tau}\mathbf{1}_{\left\{\frac{\sigma}{\tau}\le\min\left\{\frac{\delta}{2},\sqrt{\epsilon}\right\}\right\}} + \delta\left(\sqrt{\epsilon} - \frac{\delta\tau}{4}\right)\mathbf{1}_{\left\{\frac{\sigma}{\tau}>\frac{\delta}{2}\ge\sqrt{\epsilon}\right\}} + \left(2\sqrt{\epsilon}\sigma - \epsilon\tau\right)\mathbf{1}_{\left\{\frac{\sigma}{\tau}>\sqrt{\epsilon}>\frac{\delta}{2}\right\}}\right].$$

Theorem 4 shows when one should extend the deadline to satisfy the demand or let the job depart at its deadline with unsatisfied demands. The solution (16) generalizes the optimal algorithms in Section 3.1-3.3, and we call (16) *Generalized Exact Scheduling.* Moreover, Generalized Exact Scheduling is also optimal for a more general problem (11), when the unit costs for unsatisfied demands and deadline extensions are allowed to be heterogeneous as in equation (11).

Corollary 1. *The optimal solution of* (11) *is*

$$\bar{u}(x,y,\delta,\epsilon) = \begin{cases} \dfrac{x}{y}, & \text{if } y>0,\ \dfrac{x}{y} \le \min\left\{\dfrac{\delta}{2}, \sqrt{\epsilon}\right\}, \\ \dfrac{\delta}{2}, & \text{if } y>0,\ \dfrac{x}{y} > \dfrac{\delta}{2},\ \dfrac{\delta}{2} \le \sqrt{\epsilon}, \\ \sqrt{\epsilon}\,\mathbf{1}_{\{x>0\}}, & \text{otherwise.} \end{cases}$$

4. PERFORMANCE BOUNDS

The focus of this work is on distributed algorithms, due to the importance of the algorithms being implementable in large-scale service systems. Given this focus, it is important to understand how much performance degradation is incurred due to restricting ourselves to distributed algorithms. To characterize the performance degradation, we compare the optimal distributed algorithm with the optimal centralized algorithms in this section by using both theoretical bounds and numerical comparisons. Specifically, we first provide an upper bound on the performance degradation. Then, we compare the optimal distributed online algorithms with existing centralized or offline algorithms using real Electric Vehicle charging instances [18].

Analytic bounds. To derive competitive-ratio-like bounds on the performance of optimal distributed policies, we first define centralized (online) policies and then bound their achievable performance. Then we compare this to performance bounds on the optimal distributed policies. The class of centralized algorithms we consider is of the form

$$r_k(t) = w(k,t,A_t), \qquad \forall k \in \mathcal{V}, \tag{17}$$

where $A_t = \{(a_k, \sigma_k, x_k(t), y_k(t)) : a_k \le t\}$ is the set that contains the information of jobs arriving before t, and $w(k,t.\cdot)$ is a deterministic mapping from A_t to a service rate $r_k(t)$.

Lemma 1. *Under any centralized policy of the form* (17)*, the stationary variance of $P(t)$ is lower-bounded by*

$$\mathrm{Var}(P) \ge \frac{\Lambda^2\mathbb{E}[\sigma^2]^2}{4\mathrm{Var}(X)},$$

where $X(t)$ is the total amount of remaining service demand of jobs arriving before t.

Lemma 1 characterizes the trade-off between achieving a small variance of $X(t)$ and achieving a small variance of $P(t)$. An immediate consequence of Lemma 1 is a competitive-ratio-like bound that compares Exact Scheduling.

Corollary 2. *Let* $\mathrm{Var}(P^*)$ *be the minimum stationary variance attainable by any centralized algorithm* (17) *with the same level of* $\mathrm{Var}(X)$ *as Exact Scheduling. Then, the stationary variance $P(t)$ that is attained by Exact Scheduling* (12) *satisfies*

$$\mathrm{Var}(P) \le \frac{\mathbb{E}[\sigma^2/\tau]\mathbb{E}[\sigma^2\tau]}{\mathbb{E}[\sigma^2]^2}\mathrm{Var}(P^*), \tag{18}$$

where the expectations on the right hand side are taken over the arrival distribution.

Corollary 2 bounds the ratio of $\mathrm{Var}(P)$, achievable by Exact Scheduling (the optimal distributed algorithm), and $\mathrm{Var}(P^*)$, achievable by any centralized algorithms. When the sojourn time τ is a deterministic random variable, (18) reduces to $\mathrm{Var}(P) \le \mathrm{Var}(P^*)$, implying that distributed algorithms can perform equally well compared to the centralized algorithms having the same $\mathrm{Var}(X)$. One such case is when service demands and sojourn times are deterministic variables, and the service demand of each job equals its sojourn time (arrival times a are random). In this case, due to the demand constraints (1) and the deadline constraints (2), $r_k(t) = 1_{\{t\in[a_k, a_k+\tau_k)\}}$ is trivially optimal both among centralized policies and among distributed policies.

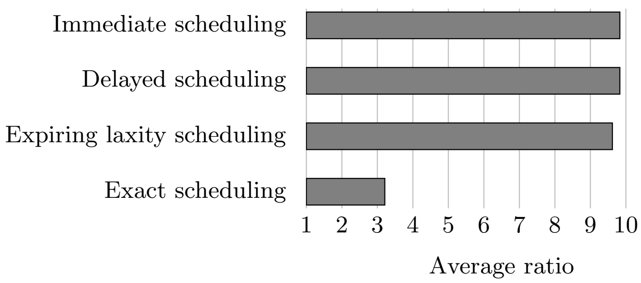

Figure 1: Performance under strict demand constraints.

Empirical performance. In order to further evaluate the performance of Exact Scheduling, we test it using data from an Electric Vehicle Charging Testbed [18] and compare the performance with existing scheduling algorithms. We employ a trace-driven simulation on a total of 92 charging instances from the testbed data in [18]. Each instance contains a set of jobs that are requested within a day. We compute the ratios between the empirical variance achieved by a few online algorithms and the empirical variance by the optimal centralized offline algorithm for all instances. The algorithms tested are Immediate scheduling ($u(x,y) = \mathbf{1}_{\{x>0\}}$), Delayed Scheduling ($u(x,y) = \mathbf{1}_{\{y\leq x\}}$), Expiring Laxity (serving jobs with positive remaining laxity equally and serving jobs with zero laxity at its maximum rate [12]), and Exact Scheduling. For each algorithm tested, we plot the mean ratio in Figure 1. The results highlight significant performance gains compared to other distributed algorithms and competitive performance with the optimal centralized offline algorithm.

5. REFERENCES

[1] F. Baccelli and B. Błaszczyszyn. *Stochastic Geometry and Wireless Networks, Volume I - Theory.* Now Publishers, 2009.

[2] N. Bansal, T. Kimbrel, and K. Pruhs. Speed scaling to manage energy and temperature. *Journal of the ACM (JACM)*, 54(1):3, 2007.

[3] M. Behrangrad. A review of demand side management business models in the electricity market. *Renewable and Sustainable Energy Reviews*, 47:270–283, 2015.

[4] P. P. Bhattacharya and A. Ephremides. Optimal scheduling with strict deadlines. *IEEE Transactions on Automatic Control*, 34(7):721–728, 1989.

[5] G. Binetti, A. Davoudi, D. Naso, B. Turchiano, and F. L. Lewis. Scalable real-time electric vehicles charging with discrete charging rates. *IEEE Transactions on Smart Grid*, 6(5):2211–2220, 2015.

[6] E. Boutin, J. Ekanayake, W. Lin, B. Shi, J. Zhou, Z. Qian, M. Wu, and L. Zhou. Apollo: Scalable and coordinated scheduling for cloud-scale computing. In *OSDI*, volume 14, pages 285–300, 2014.

[7] G. C. Buttazzo. *Hard real-time computing systems: predictable scheduling algorithms and applications*, volume 24. Springer Science & Business Media, 2011.

[8] S. Çelik and C. Maglaras. Dynamic pricing and lead-time quotation for a multiclass make-to-order queue. *Management Science*, 54(6):1132–1146, 2008.

[9] N. Chen, L. Gan, S. H. Low, and A. Wierman. Distributional analysis for model predictive deferrable load control. In *Proc. of the IEEE 53rd annual Conference on Decision and Control*, 2014.

[10] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam. Managing server energy and operational costs in hosting centers. In *ACM SIGMETRICS performance evaluation review*, volume 33, pages 303–314. ACM, 2005.

[11] J. Dean and L. A. Barroso. The tail at scale. *Communications of the ACM*, 56(2):74–80, 2013.

[12] A. Ferragut, F. Paganini, and A. Wierman. Controlling the variability of capacity allocations using service deferrals. *ACM Transactions on Modeling and Performance Evaluation of Computing Systems (TOMPECS)*, 2(3):15, 2017.

[13] L. Gan, U. Topcu, and S. H. Low. Optimal decentralized protocol for electric vehicle charging. *IEEE Transactions on Power Systems*, 28(2):940–951, 2013.

[14] A. Gandhi. *Dynamic server provisioning for data center power management.* PhD thesis, Carnegie Mellon University, 2013.

[15] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel. The cost of a cloud: research problems in data center networks. *ACM SIGCOMM computer communication review*, 39(1):68–73, 2008.

[16] H. C. Gromoll, Ł. Kruk, et al. Heavy traffic limit for a processor sharing queue with soft deadlines. *The Annals of Applied Probability*, 17(3):1049–1101, 2007.

[17] J. Hong, X. Tan, and D. Towsley. A performance analysis of minimum laxity and earliest deadline scheduling in a real-time system. *IEEE Transactions on Computers*, 38:1736–1744, 1989.

[18] G. Lee, T. Lee, Z. Low, S. H. Low, and C. Ortega. Adaptive charging network for electric vehicles. In *Signal and Information Processing (GlobalSIP), 2016 IEEE Global Conference on*, pages 891–895. IEEE, 2016.

[19] J. Lehoczky, L. Sha, and Y. Ding. The rate monotonic scheduling algorithm: Exact characterization and average case behavior. In *Real Time Systems Symposium, 1989., Proceedings.*, pages 166–171. IEEE, 1989.

[20] J. P. Lehoczky. Using real-time queueing theory to control lateness in real-time systems. *ACM SIGMETRICS Performance Evaluation Review*, 25(1):158–168, 1997.

[21] C. L. Liu and J. W. Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the ACM (JACM)*, 20(1):46–61, 1973.

[22] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. Andrew. Greening geographical load balancing. In *Proceedings of the ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*, pages 233–244. ACM, 2011.

[23] C. Maglaras and J. A. V. Mieghem. Queueing systems with leadtime constraints: A fluid-model approach for admission and sequencing control. *European journal of operational research*, 167(1):179–207, 2005.

[24] S. Melnik, A. Gubarev, J. J. Long, G. Romer, S. Shivakumar, M. Tolton, and T. Vassilakis. Dremel: interactive analysis of web-scale datasets. *Proceedings of the VLDB Endowment*, 3(1-2):330–339, 2010.

[25] P. Moyal. On queues with impatience: stability, and the optimality of earliest deadline first. *Queueing Systems*, 75(2-4):211–242, 2013.

[26] Y. Nakahira, N. Chen, L. Chen, and S. H. Low. Smoothed least-laxity-first algorithm for ev charging. In *Proceedings of the Eighth International Conference on Future Energy Systems*, pages 242–251. ACM, 2017.

[27] A. Nayyar, J. Taylor, A. Subramanian, K. Poolla, and P. Varaiya. Aggregate Flexibility of a Collection of Loads. In *Proc. of the 52nd IEEE Conference on Decision and Control*, 2013.

[28] S. S. Panwar, D. Towsley, and J. K. Wolf. Optimal scheduling policies for a class of queues with customer deadlines to the beginning of service. *Journal of the ACM (JACM)*, 35(4):832–844, 1988.

[29] E. Plambeck, S. Kumar, and M. J. Harrison. A multiclass queue in heavy traffic with throughput time constraints: Asymptotically optimal dynamic controls. *Queueing Systems*, 39(1):23–54, 2001.

[30] K. Spees and L. B. Lave. Demand response and electricity market efficiency. *The Electricity Journal*, 20(3):69–85, 2007.

[31] J. A. Stankovic and K. Ramamritham. What is predictability for real-time systems? *Real-Time Systems*, 2(4):247–254, 1990.

[32] A. Subramanian, M. Garcia, D. Callaway, K. Poolla, and P. Varaiya. Real-Time Scheduling of Distributed Resources. *IEEE Transactions on Smart Grid*, 4:2122–2130, 2013.

[33] L. M. Vaquero, L. Rodero-Merino, and R. Buyya. Dynamically scaling applications in the cloud. *ACM SIGCOMM Computer Communication Review*, 41(1):45–52, 2011.

[34] M. Zeballos, A. Ferragut, and F. Paganini. Achieving fairness for EV charging in overload: a fluid approach. In *IFIP Performance.* ACM, 2018.