Autoscaling Effects in Speed Scaling Systems

Maryam Elahi Carey Williamson
Department of Computer Science, University of Calgary
Calgary, Alberta, Canada T2N 1N4
{bmelahi, carey}@cpsc.ucalgary.ca

***Abstract*—In this paper, we study the dynamics of coupled speed scaling systems, in which service rate is a function of system occupancy. We focus on both Processor Sharing (PS) and Shortest Remaining Processing Time (SRPT) as scheduling disciplines, and study their speed scaling dynamics under heavy load. Using a combination of Markov chain analysis and discrete-event simulation, we identify several important properties of speed scaling systems, which we call the autoscaling effect, the α effect, and the saturation effect. We also identify different overload regimes for PS and SRPT. In particular, SRPT exhibits a starvation effect that differs from the compensation effect of PS. These dynamics lead to different stability, fairness, and robustness properties for PS and SRPT under heavy load.**

***Index Terms*—Speed scaling; PS; SRPT; Markov chain; analysis; simulation**

I. Introduction

Dynamic speed scaling systems provide an inherent tradeoff between response time and energy consumption in computer systems [2]. For example, running the CPU faster improves job response time, but also consumes more energy, because of the higher voltage levels and clock frequencies required. Finding a suitable balance between these two opposing metrics is the main focus in the design of speed scaling systems.

Within a speed scaling system, the two most important design decisions are the speed scaling function and the scheduler. First, the speed scaling function determines *at what speed* jobs are to be executed. This function adjusts the aggregate service rate dynamically based on the offered load. One approach is job-count-based speed scaling, wherein the service rate is a function of the instantaneous system occupancy [3], [9], [23]. This approach optimizes the linear combination of response time and energy consumption in the worst case [23]. With this design, the highest CPU rates are used when the system backlog is the largest, and the lowest CPU rates are used when the load is light. Second, the scheduler determines *which job* is executed next. Classic examples include First-Come-First-Serve (FCFS), Processor Sharing (PS), and Shortest-Remaining-Processing-Time (SRPT). These scheduling algorithms have different properties with respect to queueing delay, response time, and fairness.

In this paper, we focus on the interplay between the speed scaling function and the scheduler. We consider *coupled* speed scaling systems, wherein the system speed is a direct function of system occupancy (i.e., job-count-based). We restrict our attention to PS and SRPT as two representative examples of schedulers. PS dynamically shares the CPU amongst all active jobs in a time-slicing fashion. This scheduling algorithm is attractive because of its fairness, and its ability to approximate the time-sharing and multiplexing characteristics of practical computer systems. SRPT devotes service to the smallest remaining job in the system. Although SRPT can be unfair, this scheduling algorithm is attractive because it optimizes the average response time over any fixed-speed sample path [24].

The primary motivation for our work is a desire to better understand the autoscaling properties of speed scaling systems. As one example, consider a subtle but important property that we call the *compensation effect*. Given an arbitrary set of jobs in a speed scaling system, suppose that SRPT makes a "good" decision to complete a small job more quickly than PS does. By doing so, however, SRPT reduces the system occupancy and service rate relative to PS. This might be good for short-term energy consumption, but might be a disadvantage in the longer term, since it may take SRPT longer to complete the remaining jobs. We want to understand this subtle tradeoff in speed scaling systems.

A novel aspect of our study is considering speed scaling systems under "overload". By overload, we mean sustained offered loads that far exceed the capacity of single-speed systems. If there is no limit to the maximum service rate, then a speed scaling system will automatically adjust (i.e., autoscale) its service rate to accommodate whatever load is presented to it. We use mathematical analysis (for PS) and discrete-event simulation (for PS and SRPT) to study the dynamics of speed scaling systems under a wide range of offered loads, and establish invariant properties of such systems that can be used to validate our results.

The main observations and insights that emerge from our work are the following. First, PS is amenable to Markov chain analysis, and has elegant system occupancy distributions in speed scaling systems, consistent with classical works [13], [14]. Second, SRPT is not amenable to Markov chain analysis, since it is not a symmetric queueing discipline, and it violates the memory-less property. Nonetheless, it has interesting system occupancy dynamics that can be understood reasonably well. Third, the "overload" regimes for PS and SRPT are different, which lead to different behavioural properties under very high load. Among these properties are the autoscaling effect, the saturation effect, the compensation effect, and the starvation effect. Our work identifies and illustrates these dynamics in speed scaling systems.

The rest of this paper is organized as follows. Section II reviews prior literature on speed scaling systems. Section III presents our system model. Section IV presents our analytical results. Section V presents simulation results. Finally, Section VI concludes the paper.

II. Background and Related Work

The literature on scheduling policies dates back to the early days of computer systems, in which single-speed systems were the norm. The most common metric used to evaluate scheduling policies is *mean response time* (also known as sojourn time). In single-speed systems, SRPT is optimal for this metric [18]. SRPT is a preemptive policy that always selects for service the pending job in the system with the least remaining work. Although SRPT minimizes the mean response time, it is rarely used in practice, since it needs advance knowledge of job sizes. Furthermore, it can be unfair; large jobs may starve if small jobs have precedence.

The unfairness of SRPT has been studied extensively [4], [11], [22]. Under heavy load, SRPT may be unfair to certain jobs [4]. However, in many cases, SRPT can provide lower expected response times than PS for *all* job sizes [4], [22]. Under heavy load, the jobs that receive unfair treatment are intermediate-size jobs, and not the largest jobs [12]. In single-speed systems, all scheduling policies asymptotically converge to the same slowdown value for very large jobs [12]. It is not known if this property holds in speed scaling systems. In single-speed systems, one nice property of SRPT under (transient) overload is that it minimizes the number of jobs that are starved [19].

Prior research on speed scaling systems appears in two different research communities: theory and systems. The theoretical work typically focuses on formal mathematical proofs of the properties of speed scaling systems (e.g., optimality, fairness, worst-case bounds, competitive analysis) under some simplifying assumptions (e.g., unbounded service rates, known job sizes, idealized schedulers, zero context-switching costs). Systems research typically aims at "good" rather than optimal solutions, based on practical considerations [8], [9] (e.g., limited range of discrete system speeds, unknown job sizes, threshold-based control, leakage power, realistic schedulers, non-zero context-switching costs). In this literature review, we focus primarily on the theoretical work, which provides the relevant background context for our paper.

In speed scaling systems, there are many tradeoffs between service rate, response time, and energy consumption. Yao *et al.* [26] conducted one of the first analytical studies of dynamic speed scaling systems, assuming that jobs have explicit deadlines, and the service rate is unbounded. Bansal *et al.* [5] considered an alternative approach that minimizes system response time, within a fixed energy budget. Some work focuses on energy-efficient algorithms [2], while others focus on finding the optimal fixed rate at which to serve jobs in a system with dynamically-settable speeds [10], [23], [25].

Several studies indicate that energy-proportional speed scaling is nearly optimal [3], [7]. In this model, the power consumption $P(s)$ of the system depends only on the speed s, which itself depends on the number of jobs in the system (i.e., $P(s) = n$). Bansal, Chan, and Pruhs [7] showed that SRPT with the speed scaling function $P^{-1}(n+1)$ is 3-competitive for an arbitrary power function P. Andrew *et al.* [3] showed that the optimal policy is SRPT with a job-count-based speed scaling function of the form $s = P^{-1}(n\beta)$.

TABLE I
Model Notation

Symbol	Description
λ	Mean job arrival rate
μ	Service rate
μ_n	Service rate in state n
$E[X]$	Average size (work) for each job
ρ	Offered load $\rho = \lambda/\mu = \lambda E[X]$
p_n	Steady-state probability of n jobs in the system
U	System utilization $U = 1 - p_0$
n	Number of jobs
$f(n)$	CPU speed as a function of number of jobs
t	Time in seconds
$n(t)$	Number of jobs in system at time t
$s(t)$	CPU speed at time t
$P(s)$	Power consumption when running at speed s
α	Exponent in power consumption function $P(s) = s^\alpha$

III. System Model

We consider a single-server system with dynamically adjustable service rates. Service rates are changed only at job arrival and departure points (i.e. when the system occupancy changes). There is no cost incurred for changing the service rate, and no limit on the maximum possible service rate.

The workload presented to the server is a sequence of tuples specifying job arrival times and sizes. We assume that the arrival process is Poisson. That is, the interarrival times are exponentially distributed and independent, with mean arrival rate λ. The size (work) of a job represents the time it takes to complete the job when the service rate $\mu = 1$. We assume that job sizes are exponentially distributed and independent. Unless stated otherwise, we assume that the mean job size is $E[X] = 1$. Table I summarizes our model notation.

In this paper, we consider two specific work-conserving scheduling policies, namely PS and SRPT. We assume that the schedulers know all job sizes upon arrival. A job in execution may be preempted and later resumed without any context-switching overhead.

A speed scaling function, $s(t)$, specifies the speed of the system at time t. Let $P(s)$ denote the power required to run at speed s. For coupled speed scaling, the speed at time t depends on the number of jobs in the system, denoted by $n(t)$, and thus is influenced by the scheduling policy. The best known policy uses the speed function $s(t) = P^{-1}(n(t)\beta)$ [3]. In this paper, we assume $\beta = 1$. We also consider $P(s) = s^\alpha$, which is commonly used in the literature to model CPU power consumption. Therefore, in the coupled speed scaling model, we use $s(t) = \sqrt[\alpha]{n(t)} = n(t)^{1/\alpha}$, where $\alpha \geq 1$.

IV. Analytical Results

In this section, we present our analytical results for speed scaling under PS scheduling, which treats all jobs equally.

A. Markov Chain Formulation

The starting point for our analysis is the classic M/M/∞ queueing model [14]. In this model, a new server is always

available for each arriving customer, so there is no waiting time, and the response time depends only on the service requirements of each job. In our speed scaling context, there is only a single server, so there can be waiting times for jobs (depending on the service discipline), but the service rate scales with occupancy, and can be much higher than that associated with a single dedicated server. The analysis of these two systems is identical [14].

For the M/M/∞ model, the system occupancy follows a Poisson distribution [14]. Specifically, $p_n = \frac{(\lambda/\mu)^n}{n!}e^{(-\lambda/\mu)}$, and the mean system occupancy is $\bar{N} = \frac{\lambda}{\mu} = \rho$. Like most queueing models, the analysis implicitly assumes a First Come First Serve queue, in which only the front job receives service, while the other jobs (if any) wait. However, the steady-state occupancy of the system is the same under PS [13], [23].

B. Autoscaling Effect

In single-speed systems, there is a fundamental ergodicity requirement for a queueing system to be stable. In an M/M/1 queue, for example, the mean queue size is given by $\bar{q} = \frac{\rho}{1-\rho}$, where $\rho = \lambda/\mu$ represents the relative offered load to the system, or equivalently the system utilization $U = \rho = 1 - p_0$ (i.e., the proportion of time that the server is busy with at least one job). Clearly, the expression for $\bar{q}$ is only well-defined for $\rho < 1$, which is the ergodicity requirement for the queue. Specifically, the average arrival rate λ must not exceed the average service rate μ, if the queue is to be stable.

In dynamic speed scaling systems, we are no longer restricted to $\rho < 1$ as the operating regime, since the system automatically scales its service rate to accommodate the offered load. The only technical requirement is that $\rho < \infty$ [14].

If there is no upper bound on the maximum service rate, and infinite storage capacity in the queue for jobs, then the steady-state average service speed $\bar{s}$ must match the offered load (i.e., $\bar{s} = \rho$). This is an invariant property of any stable speed scaling system. We refer to this property as the *autoscaling effect*.

An important subtlety in speed scaling systems is that the notions of offered load ρ and system utilization U differ [25]. At sufficiently high load, the system rarely empties, and $U \rightarrow 1$. However, the offered load ρ at which this occurs is typically $\rho > 1$, and sometimes $\rho >> 1$, depending on the parameters of the speed scaling system.

Despite this difference, there is still a fundamental ergodicity property underlying the dynamics of speed scaling systems. Specifically, the steady-state occupancy has a central tendency toward the lowest-occupancy state in which the service rate μ_n is at least as large as the (fixed) arrival rate λ. This property can be used to analytically calculate the mean system occupancy in steady state. Much like the M/M/∞ model, the system occupancy for linear speed scaling follows a Poisson distribution with mean $\bar{N} = \lambda/\mu = \rho$.

C. Effect of α

We next consider the dynamics of a system with sub-linear speed scaling. Specifically, we consider running the system at speed $s = n^{1/\alpha}$ when the system occupancy is n jobs.

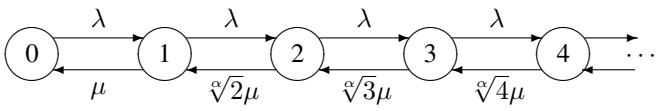


Fig. 1. Markov Chain Model for Dynamic Speed Scaling System

We consider $1 \leq \alpha \leq 3$, which is the relevant range of interest for Dynamic Voltage and Frequency Scaling (DVFS) on commercial processors [20], [25].

The parameter α determines the set of distinct speeds available in our speed scaling system. For the special case $\alpha = 1$, the speeds scale linearly with occupancy, much like the M/M/∞ queue, which provides a natural validation point for our model. For $\alpha = 2$, speeds scale less than linearly with system occupancy, following the "square root speed scaling" approach recommended in the literature (i.e., the system speed when there are n jobs in the system is $\sqrt{n} = n^{1/2}$). For $\alpha = 3$, speeds scale even more slowly with growing system occupancy: the system speed when there are n jobs in the system is $\sqrt[3]{n} = n^{1/3}$). In the limiting case of $\alpha = \infty$, the speeds scale so slowly that they are effectively constant (i.e., single-speed system). This provides another validation point for our model.

The Markov chain for our generalized speed scaling system is shown in Figure 1. Analysis of this chain produces steady-state probabilities p_n that are analogous to those for the M/M/∞ chain, except for the effect of the $1/\alpha$ exponent on all of the service rates.

The general expression for the steady-state probabilities is:

$$p_n = p_0 \prod_{i=0}^{n-1} \frac{\lambda}{(i+1)^{1/\alpha}\mu} = p_0 \frac{(\lambda/\mu)^n}{(n!)^{1/\alpha}}$$

and the idle probability p_0 has the following form:

$$p_0 = \frac{1}{1 + \sum_{i=1}^{\infty} \frac{\rho^i}{\sqrt[\alpha]{i!}}}$$

From these steady-state probabilities, it is straightforward (but tedious) to obtain the mean occupancy (and higher moments) of the system. We currently do so numerically, since we don't have a closed form for the general case. For the special case $\alpha = 2$, analysis leads to a bound $E[N] \leq \sqrt{2}\rho^2 + p_0\rho$. This bound becomes tight as ρ increases.

The impact of the parameter α on system occupancy is illustrated in Figure 2. The graphs on the left-hand side are for PS, while those on the right are for SRPT, which will be discussed later in Section V-B. Figure 2(a) is for $\rho = \lambda = 1$, while Figures 2(b) and (c) show $\rho = \lambda = 2$ and $\rho = \lambda = 3$, respectively. In each graph, the horizontal axis shows the system occupancy, while the vertical axis shows the probability of observing that occupancy. The three lines on the graph show the analytical results for system occupancy for $\alpha = 1$, $\alpha = 2$, and $\alpha = 3$. The points show simulation results for the same settings. The latter results come from two different simulators: a Markov chain simulation with the same assumptions as the analytical model (i.e., memory-less

property for service times), and a discrete-event simulation of a speed scaling system (which is job-count based, and remembers the remaining size of any pre-empted job). All simulation results agree extremely well with the analytical model, for both PS and FCFS scheduling.

The primary effect of increasing α is to shift the occupancy distribution to the right. This makes sense intuitively, since the slower service rates lead to a larger queue of jobs in the system. However, the larger backlog of jobs leads to an increased service rate, which eventually stabilizes the system. The system stabilizes around a mean occupancy of $\bar{N} = \rho^{\alpha}$. This is another invariant property of any stable coupled speed scaling system under sufficiently heavy offered load, as confirmed by the results for PS and SRPT in Figure 2.

A secondary effect of the parameter α is the flattening and spreading of the (formerly Poisson) distribution for system occupancy. While the structure of the distribution is similar to Poisson, the state probabilities degenerate, and the Coefficient of Variation (CoV) is greater than that for a Poisson distribution. The particular relationship observed here is $Var = \alpha\bar{N}$. In the limiting case of $\alpha = \infty$, this distribution degenerates to an equal but negligible probability for all states, indicating an unstable (infinite) queue.

A tertiary effect of α is the decline in p_0, which is the probability of having an empty system. Since $U = 1 - p_0$, this is simply another manifestation of the saturation effect.

D. Insights and Observations

The following insights emerge from our analysis:

- In stable speed scaling systems, $\bar{s} = \rho$.
- PS with $\alpha = 1$ behaves like the M/M/∞ queue.
- Increasing α alters the Poisson structure of PS.
- In stable speed scaling systems, $lim_{\rho\to\infty}\frac{\bar{N}}{\rho^{\alpha}} = 1$.

V. Simulation Results

In this section, we use discrete-event simulation to study PS and SRPT scheduling in coupled speed scaling systems.

A. Saturation Effect

In our first experiment, we use our simulator to explore the busy period structure in speed scaling systems. As the load offered to a speed scaling system is increased, the number of busy periods diminishes until there is a single massive busy period that includes all jobs. We refer to this phenomenon as the *saturation effect*, since $U \to 1$.

Despite the saturation of the utilization U, the speed scaling system still remains stable, even if the load is further increased. In particular, $lim_{\rho\to\infty}\frac{\bar{N}}{\rho^{\alpha}} = 1$. The probability of the system returning to the empty state becomes very small, but the system is still recurrent.

A key observation from simulation is that the saturation point for SRPT is different than under PS scheduling. In particular, the saturation point for SRPT is lower than that for PS. One implication of this observation is that there exist load levels at which SRPT is beyond saturation, while PS is not. We explore one consequence of this in the next subsection.

B. Starvation Effect

In single-speed systems, SRPT is optimal, since it minimizes the mean waiting time and the mean response time. However, SRPT can sometimes be unfair, since it favours short jobs over long jobs. While there is often fear that long jobs will starve, this is not true in a stable system that has adequate capacity to (eventually) serve all the jobs. Under overload, however, starvation can occur [19].

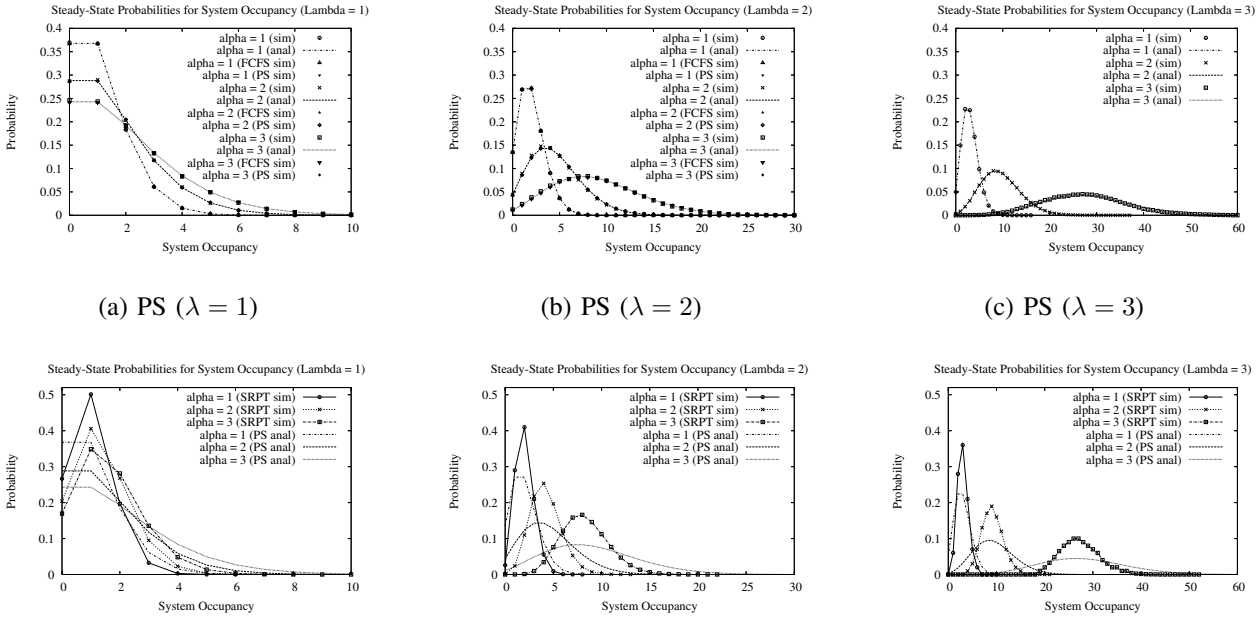
In a speed scaling system, the effects of SRPT scheduling are subtle and interesting. By serving the smallest jobs to completion, SRPT improves response time, and minimizes system occupancy. However, by reducing system occupancy, SRPT inherently reduces the service rate, perhaps leading to worse response times for the remaining jobs. Understanding this tradeoff is one of the key motivations for our work.

The right-hand side of Figure 2 shows simulation results for the system occupancy in an SRPT system. In each graph, the simulation results for SRPT are shown using points (connected with lines for easier visual reference), while the other lines in the background show the PS analytical results (for comparison purposes). These simulation results lead to three specific observations about SRPT-based speed scaling systems.

The first observation is that the system occupancy distribution under SRPT has much lower variance than the PS system. That is, there is a much greater central tendency toward the mean occupancy $\bar{N} = \rho^{\alpha}$ than under PS. This observation is particularly evident in Figure 2(e) and Figure 2(f), although it also holds for Figure 2(d), as confirmed in Tables II to IV. In fact, for $\alpha = 1$, the variance is even less than for a Poisson distribution. Intuitively, this phenomenon makes sense: if there are many jobs in the system, then it is likely that some of them are small, and will depart soon, since the speed is high. Conversely, if the system occupancy is low, then the speed is low, and departures will be sluggish, especially for large jobs.

The sluggishness leads to our second observation, which we call the starvation effect. Under SRPT-based speed scaling, large jobs that enter the system tend to remain a very long time before they receive service. This is another manifestation of SRPT's unfairness, which is magnified under speed scaling [3]. Furthermore, when they do receive service, it is at widely different rates (i.e., very small when they are still large, but faster and faster as they become smaller). This phenomenon leads to higher variability of response time for SRPT compared to PS (see Table IV, for example).

The third observation from simulation is that the sizes of the starved jobs tend to increase over time. This observation is not evident from the graphs or the table, but does emerge from post-processing and visualization of our simulation output. Loosely stated, a large job is more likely to get service only when an even larger job arrives. While the number of large jobs in the queue remains finite (since the queue itself is finite), their average size tends to grow over time. The latter observation suggests a difference between the notions of "job stability" and "work stability" under SRPT scheduling in a speed scaling system, when $U \to 1$. We are investigating the



(a) PS ($\lambda = 1$) (b) PS ($\lambda = 2$) (c) PS ($\lambda = 3$)

(d) SRPT ($\lambda = 1$) (e) SRPT ($\lambda = 2$) (f) SRPT ($\lambda = 3$)

Fig. 2. System Occupancy Results for PS and SRPT Scheduling (PS analysis/simulation; SRPT simulation)

details of this phenomenon as part of our ongoing work.

C. Compensation Effect

In single-speed systems, it is well-known that SRPT provides better response time than PS, and the differences between the two can be very large, depending on system load and job size distribution. However, in speed scaling systems, the differences between the two schedulers is not as dramatic, because of the compensation effect. To elaborate, if SRPT makes a decision that completes a specific job sooner than under PS, the side effect of this decision is to reduce the system rate for the next job in service. In other words, PS now has an advantage over SRPT in terms of service rate.

Tables II to IV present results exploring the tradeoffs between PS and SRPT, including the compensation effect. Table II is for $\alpha = 1$, while Table III and Table IV are for $\alpha = 2$ and $\alpha = 3$, respectively. In each table, we present results for mean and variance of response time, mean and variance of occupancy, and system utilization, with the offered load ranging from $\rho = \lambda = 1$ to $\rho = \lambda = 3$.

The results in these tables show elegant autoscaling properties for PS scheduling. For the special case of linear speed scaling ($\alpha = 1$) in Table II, the mean occupancy has a Poisson distribution with mean $\bar{N} = \rho$, and the mean response time matches the expected exponential distribution with mean $E[X] = 1$. For $\alpha = 2$ in Table III, the mean occupancy grows with load according to our bound, and the variance of occupancy increases as well. These effects translate into higher mean and variance for response time. Similar observations apply for $\alpha = 3$ in Table IV, as the PS system approaches its saturation point.

TABLE II
SIMULATION RESULTS FOR PS AND SRPT ($\alpha = 1$)

Scheduling Policy	Load λ	Response Time		Occupancy		System Util. U
		Mean	Var	Mean	Var	
PS	1	1.005	1.016	1.005	1.003	0.634
	2	1.005	1.016	2.010	2.006	0.867
	3	1.005	1.016	3.014	3.014	0.952
SRPT	1	1.005	2.281	1.005	0.621	0.734
	2	1.005	6.208	2.010	0.922	0.974
	3	1.005	21.123	3.016	1.222	0.9996

The results for SRPT in these tables are also interesting. For linear speed scaling (Table II), the mean occupancy and mean response time match those for PS. However, the variance of occupancy is much lower than for PS, and the variance of response time much higher. The other notable difference is the relative system utilization, which is much higher than under PS. In Table III for $\alpha = 2$, the response time advantages of SRPT are evident, but the advantage is relatively modest, due to the compensation effect. The mean system occupancy is slightly lower than under PS, as is the variance. However, as the system utilization approaches saturation, the variance of response time grows dramatically. For $\alpha = 3$ in Table IV, similar observations apply. At saturation, the mean occupancy approaches $\bar{N} = \rho^{\alpha}$ as expected, and the variance of occupancy is much lower than under PS. The mean response time is still slightly better than PS, but not by much. Furthermore, the variance of response time becomes unwieldly, reflecting the starvation effect.

TABLE III
SIMULATION RESULTS FOR PS AND SRPT ($\alpha = 2$)

Scheduling Policy	Load λ	Response Time		Occupancy		System Util. U
		Mean	Var	Mean	Var	
PS	1	1.537	2.615	1.537	2.192	0.714
	2	2.297	5.836	4.595	8.159	0.958
	3	3.203	11.001	9.611	18.342	0.997
SRPT	1	1.341	5.579	1.341	1.020	0.796
	2	2.103	147.859	4.208	2.611	0.998
	3	2.906	16,760.8	9.175	4.742	1.0

TABLE IV
SIMULATION RESULTS FOR PS AND SRPT ($\alpha = 3$)

Scheduling Policy	Load λ	Response Time		Occupancy		System Util. U
		Mean	Var	Mean	Var	
PS	1	2.013	5.039	2.013	3.650	0.759
	2	4.600	25.053	9.201	24.522	0.989
	3	9.474	97.044	28.267	81.642	0.99994
SRPT	1	1.602	10.233	1.602	1.393	0.831
	2	4.127	5,779.02	8.262	5.971	1.0
	3	8.967	856,635	27.302	16.717	1.0

D. Insights and Observations

Several insights emerge from our simulation results:

- Under heavy load, busy periods coalesce, and $U \to 1$.
- The overload regimes for PS and SRPT differ.
- SRPT suffers from starvation under very high load.

VI. CONCLUSIONS

In this paper, we have used mathematical analysis and simulation to explore the autoscaling properties of dynamic CPU speed scaling systems. Our study has focused primarily on the interplay between schedulers and speed scalers. We have assumed coupled (i.e., job-count-based) speed scaling, with PS and SRPT as representative schedulers.

The main conclusions from our work are several. First, we show that PS is amenable to Markov chain analysis, and has elegant system occupancy distributions in speed scaling systems. Insights from this model establish several invariant properties that are useful for the study of SRPT, which is not directly amenable to Markov chain analysis. Second, our simulation results show that SRPT has interesting speed scaling dynamics, with system occupancy distributions that are distinctly different than PS. These dynamics produce different response time and fairness properties. Third, the "overload" regimes for PS and SRPT are different, which lead to different behavioural properties under very high load. Among these properties are the autoscaling effect, the saturation effect, the compensation effect, and the starvation effect. Our work identifies and illustrates these dynamics in speed scaling systems.

ACKNOWLEDGEMENTS

Financial support for this work was provided by Canada'a Natural Sciences and Engineering Research Council (NSERC). The authors thank Philipp Woelfel for many discussions about speed scaling, and Jennifer Williamson for building a Java-based visualization tool for our simulator.

REFERENCES

[1] S. Albers, F. Mueller, and S. Schmelzer, "Speed Scaling on Parallel Processors", *Proceedings of ACM Symposium on Parallelism in Algorithms and Architectures* (SPAA), pp. 289-298, 2007.

[2] S. Albers, "Energy-Efficient Algorithms", *Communications of the ACM*, Vol. 53, No. 5, pp. 86-96, May 2010.

[3] L. Andrew, M. Lin, and A. Wierman, "Optimality, Fairness, and Robustness in Speed Scaling Designs", *Proceedings of ACM SIGMETRICS*, pp. 37-48, June 2010.

[4] N. Bansal and M. Harchol-Balter, "Analysis of SRPT Scheduling: Investigating Unfairness", *Proceedings of ACM SIGMETRICS Conference*, Cambridge, MA, pp. 279-290, June 2001.

[5] N. Bansal, T. Kimbrel, and K. Pruhs, "Speed Scaling to Manage Energy and Temperature", *Journal of the ACM*, Vol. 54, 2007.

[6] N. Bansal, K. Pruhs, and C. Stein, "Speed Scaling for Weighted Flow Time", *Proceedings of ACM-SIAM Symposium on Discrete Algorithms*, 2007.

[7] N. Bansal, H. Chan, and K. Pruhs, "Speed Scaling with an Arbitrary Power Function", *Proceedings of ACM-SIAM Symposium on Discrete Algorithms*, 2009.

[8] M. Dell'Amico, D. Carra, M. Pastorelli, and P. Michiardi, "Revisiting Size-based Scheduling with Estimated Job Sizes", *Proceedings of IEEE MASCOTS*, Paris, France, pp. 411-420, September 2014.

[9] M. Dell'Amico, D. Carra, M. Pastorelli, and P. Michiardi, "PSBS: Practical Size-Based Scheduling", to appear, *IEEE Transactions on Computers*, 2016.

[10] J. George and J. Harrison, "Dynamic Control of a Queue with Adjustable Service Rate", *Operations Research*, Vol. 49, No. 5, pp. 720-731, September-October 2001.

[11] M. Gong and C. Williamson, "Revisiting Unfairness in Web Server Scheduling", *Computer Networks*, Vol. 50, pp. 2183-2203, 2006.

[12] M. Harchol-Balter, K. Sigman, and A. Wierman, "Asymptotic Convergence of Scheduling Policies with Respect to Slowdown", *Proceedings of IFIP Performance 2002*, Rome, Italy, pp. 241-256, September 2002.

[13] F. Kelly, *Reversibility and Stochastic Networks*, Wiley, 1979.

[14] L. Kleinrock, *Queueing Systems, Volume 1: Theory*, Wiley, 1975.

[15] D. Lu, H. Shen, and P. Dinda, "Size-based Scheduling Policies with Inaccurate Scheduling Information", *Proceedings of IEEE/ACM MASCOTS*, Volendam, Netherlands, pp. 31-38, October 2004.

[16] I. Rai, G. Urvoy-Keller, and E. Biersack, "Analysis of LAS Scheduling for Job Size Distributions with High Variance", *Proceedings of ACM SIGMETRICS Conference*, San Diego, CA, pp. 218-238, June 2003.

[17] D. Raz, H. Levy, and B. Avi-Itzhak, "RAQFM: A Resource Allocation Queueing Fairness Measure", *Proceedings of ACM SIGMETRICS Conference*, New York, NY, June 2004.

[18] L. Schrage, "A Proof of the Optimality of the Shortest Remaining Processing Time Discipline", *Operations Research*, Vol. 16, pp. 678-690, 1968.

[19] B. Schroeder and M. Harchol-Balter, "Web Servers Under Overload: How Scheduling Can Help", *ACM Transactions on Internet Technology*, Vol. 6, No. 1, pp. 20-52, February 2006.

[20] D. Snowdon, E. Le Sueur, S. Petters, and G. Heiser, "Koala: A Platform for OS-level Power Management", *Proceedings of ACM EuroSys*, pp. 289-302, 2009.

[21] M. Weiser, B. Welch, A. Demers, and S. Shenker, "Scheduling for Reduced CPU Energy", *Proceedings of USENIX Operating System Design and Implementation* (OSDI), 1994.

[22] A. Wierman and M. Harchol-Balter, "Classifying Scheduling Policies with Respect to Unfairness in an M/GI/1", *Proceedings of ACM SIGMETRICS Conference*, San Diego, CA, pp. 238-249, June 2003.

[23] A. Wierman, L. Andrew, and A. Tang, "Power-Aware Speed Scaling in Processor Sharing Systems", *Proceedings of IEEE INFOCOM*, April 2009.

[24] A. Wierman, L. Andrew, and M. Lin, "Speed Scaling: An Algorithmic Perspective", book chapter in *Handbook of Energy-Aware and Green Computing*, CRC Press, 2012.

[25] A. Wierman, L. Andrew, and A. Tang, "Power-Aware Speed Scaling in Processor Sharing Systems: Optimality and Robustness", *Performance Evaluation*, Vol. 69, pp. 601-622, 2012.

[26] F. Yao, A. Demers, and S. Shenker, "A Scheduling Model for Reduced CPU Energy", *Proceedings of ACM Foundations of Computer Systems* (FOCS), pp. 374-382, 1995.