FSP with Dynamic Speed Scaling

Maryam Elahi*, Carey Williamson† and Philipp Woelfel†
Department of Computer Science, University of Calgary
{bmelahi, carey, woelfel}@ucalgary.ca

ABSTRACT

The adverse effects of dynamic speed scaling on fairness is pointed out in the recent literature [2]. It is shown that while Processor-Sharing (PS) with speed scaling maintains constant slowdown, speed scaling magnifies unfairness under Shortest-Remaining-Processing-Time (SRPT). In search of a scheduling policy that can improve the average response time of PS without being unfair to any job, we look at the Fair-Sojourn-Protocol (FSP). We demonstrate that in the coupled speed scaling model where the speed scaler is a function of the number of jobs in the system, FSP does not work. We then propose the so-called decoupled speed scaling model wherein the speed scaling function is completely decoupled from the scheduling policy. Our initial simulation results suggest that FSP with decoupled speed scaling provides a considerable advantage over PS.

1. INTRODUCTION

Dynamic speed scaling has received significant attention in the recent literature as an approach for balancing energy consumption and other performance metrics in all levels of computer systems [1]. In speed scaling systems, the performance metrics of throughput and response time become secondary to energy consumption, or a weighted combination of energy consumption and response time. A typical formulation of the problem involves optimizing the cost

$$z = E[T] + E[\varepsilon]/\beta, \tag{1}$$

where T represents response time, ε reflects energy cost, and β is a positive relative weighting factor. Fairness in dynamic speed scaling design was first studied by Andrew, Lin and Wierman [2]. In their work, the authors demonstrated that there are inherent tradeoffs between optimality of z, fairness, and robustness in speed scaling systems. In particular, they show that while PS with speed scaling maintains its constant slowdown and is considered to be the criterion for fairness, speed scaling creates/magnifies unfairness under SRPT and non-preemptive policies such as FCFS. Although PS is fair, similar to the single speed model, it is suboptimal. In the same paper [2], the authors show that the optimal (coupled) policy for minimizing z is SRPT with a job-count-based speed scaling function $s = P^{-1}(n\beta)$, where $P(s)$ is the power consumption when the system runs at speed s, and n is the number of jobs in the system.

*Supported by the Alberta Innovates Technology Futures (AITF) in the Province of Alberta, Canada.
†Supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

2. IN SEARCH OF FAIRNESS

The concerns about unfairness towards large jobs in size-based scheduling prompted intensive work on fairness in the past decade [5]. In the non-speed-scaling model, a policy is considered fair if for all job sizes the slowdown is at most the slowdown of PS, which is $1/(1-\rho)$, where ρ is the load [6]. In the speed scaling model, although the slowdown of PS is no longer $1/(1-\rho)$, it remains constant for all job sizes (see Proposition 15 in [2]), and thus is suggested as the criterion for fairness.

We investigate whether we can approach the optimal performance of SRPT while maintaining the fairness of PS in the speed scaling world. As an initial step, we look at the Fair-Sojourn-Protocol (FSP) [4], which is known to dominate PS in the non-speed-scaling model.

2.1 FSP with Job-count Speed Scaling

FSP is a work-conserving preempt-resume policy, devised by Friedman and Henderson to improve upon the performance of PS, while guaranteeing that no job will finish any later than it finishes under PS [4]. In non-speed-scaling systems, FSP is interesting because of its strict dominance over PS. That is, all jobs except those ending busy periods finish earlier under FSP than under PS. FSP computes the time at which jobs would complete under PS and then devotes full service to the job with the earliest completion time.

A natural question is whether FSP can be used with speed scaling. The following example shows that in the coupled speed scaling model, the answer is no. For simplicity, we assume a simple job-count-based speed scaling function of the form $s(n) = n$, though the observations apply more generally. There are two fundamental problems with FSP in the coupled speed scaling model. First, the dominance of FSP over PS is not preserved. Second, the standard implementation of FSP is ill-defined in some scenarios. To understand these issues, consider a sample path with four jobs all of size 4 and all arriving at time 0. Figure 1 (a) shows the performances of PS and FSP in this scenario. Under PS, at any point in time, all jobs remaining in the system receive concurrent service and the aggregate service rate is equal to the number of jobs in the system. We see that all four jobs finish simultaneously at time 4. Under FSP, the first job receives full service at rate 4 and finishes at time 1. Then the second job receives full service at rate 3, and finishes $\frac{4}{3}$ time units later at time $2\frac{1}{3}$. Next, the third job receives full service at rate 2, and finishes at time $4\frac{1}{3}$. Note that this completion

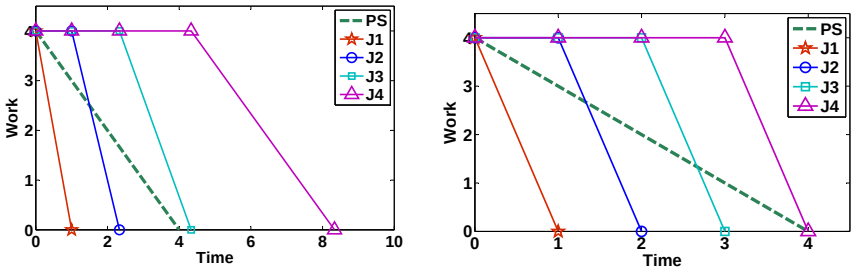

Figure 1: Scheduling under PS vs. FSP with (a) job-count-based speed scaling, and (b) decoupled speed scaling (in this example FSP runs at the same speeds as job-count-based PS)

time is later than it finishes under PS, and thus the strict dominance property of FSP is violated. Even more interesting, the fourth and final job never receives any service under FSP, since the "virtual PS" queue used to drive FSP's decision-making (see the original algorithm in [4]) contains *no jobs* at time $4\frac{1}{3}$. This anomaly arises because the start time for the fourth job under FSP is beyond its point of completion under PS. Hence the FSP policy is ill-defined in this scenario for job-count-based speed scaling.

The next question is whether we can devise another algorithm to achieve the properties of FSP (namely, efficiency and strict dominance over PS) in the speed scaling world. In [3], we prove that in the coupled speed scaling model, the answer is no and in fact *no policy can dominate PS*. However, if we allow the speed function to be completely decoupled from the scheduler, we show that FSP can maintain its strong dominance over PS.

2.2 FSP with Decoupled Speed Scaling

We notice that in the example in Section 2.1, FSP with coupled speed scaling loses its advantage over PS because the speed scaling function is sensitive to the scheduling decisions and reduces the speeds as soon as jobs leave the system. We argue that this reduction in speed happens prematurely. In [3] we show that if at all times FSP runs at the same speeds as PS, then FSP maintains its strict dominance over PS (see the example in Figure 1 (b)). Note that even though we use a shadow-PS to determine the speeds, the speed function only depends on the job arrivals and is not affected by the departures or order of service under FSP. Thus we introduce the notion of decoupled speed scaling, where the speeds of the system is decided based on an external speed function which is not sensitive to the scheduling decisions.

Decoupled speed scaling can level the playing field, and enable "apples to apples" comparisons between scheduling policies under fixed energy consumption. Furthermore, multiple playing fields are available, based on the external speed scaling function, which can be constant, derived from a shadow scheduling policy, or any non-decreasing function of the load. This approach provides great flexibility for the analysis and evaluation of speed scaling designs. In particular, with decoupled speed scaling, the response times of jobs can be altered without changing the energy consumption. That is, in the cost function (1) we can alter $E[T]$ without affecting $E[\varepsilon]$. Hence, if FSP runs at the speeds that PS with coupled speed scaling uses, it reduces the cost z, since it uses the same amount of energy but achieves better response times.

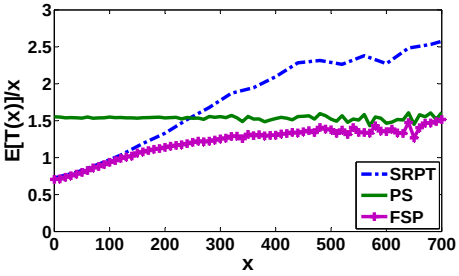

Figure 2: Simulation Results for PS, SRPT, and FSP Scheduling with Decoupled Speed Scaling. (Poisson Arrival Process ($\lambda = 1$) and Exponentially Distributed Jobs Sizes ($\mu = 100$)).

3. SIMULATION RESULTS

Figure 2 shows the simulation results for PS and FSP scheduling with decoupled speed scaling. We see that FSP dominates PS and provides visible advantage in the response times. Our initial simulation results suggest that under moderate/low load, FSP has about 30% advantage and as the load is increased the average response time advantage of FSP increases by more than 50%.

4. CONCLUSIONS

We demonstrate that the FSP scheduling policy, which dominates PS in the non-speed-scaling world, is ill-defined under coupled speed scaling based on job count. With decoupled speed scaling, however, FSP again dominates PS, and is provably efficient. Simulation results demonstrated a notable performance advantage for FSP, compared to PS. We propose decoupled speed scaling as a new paradigm for the analysis and evaluation of speed scaling systems.

5. REFERENCES

[1] S. Albers. Energy-efficient algorithms. *Commun. ACM*, 53:86–96, May 2010.

[2] L. L. Andrew, M. Lin, and A. Wierman. Optimality, fairness, and robustness in speed scaling designs. In *SIGMETRICS '10*.

[3] M. Elahi, C. Williamson, and P. Woelfel. Decoupled speed scaling: Analysis and evaluation. In *QEST '12 (to appear)*.

[4] E. J. Friedman and S. G. Henderson. Fairness and efficiency in web server protocols. In *SIGMETRICS '03*.

[5] A. Wierman. Fairness and classifications. *SIGMETRICS Perf. Eval. Rev.*, 34:4–12, March 2007.

[6] A. Wierman and M. Harchol-Balter. Classifying scheduling policies with respect to unfairness in an M/GI/1. In *SIGMETRICS '03*.