Special Issue on New Perspectives in Scheduling

Mor Harchol-Balter
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15217
harchol@cs.cmu.edu

I would like to introduce this issue by telling a story. Sometime back in 1997, I wrote a paper on a new idea for improving the response times of http requests at a Web server. The idea was to schedule the HTTP requests so as to favor requests for small files, in accordance with the well-known scheduling policy Shortest Remaining Processing Time (SRPT). The paper was rejected, for many reasons, but the review that stuck in my mind was the one that said, *"Why is this person writing about scheduling? Scheduling is dead."* According to this reviewer, everything that would ever be known about scheduling was already described in the beautiful *Theory of Scheduling* book, written in 1967, by Conway, Maxwell, and Miller.

While scheduling research may have seemed "dead" back then, we were actually only a few years away from a new awakening in scheduling research. This resurrection of scheduling has involved both a large number of *applied* papers on using scheduling to improve the performance of computer systems, but also a tremendous number of new *theoretical* results on scheduling. What's especially interesting is that these theoretical results don't even look like the old theoretical results. The metrics are different, the workloads are different, the system models are different, and the proof techniques are often entirely new.

The renewed interest in scheduling is immediately evident when one looks at the annual *ACM Sigmetrics Conference on Measurement and Modeling of Computer Systems*, the highly selective conference, known for bridging theoretical and empirical research in computer systems performance and design. Every year since 2002, the Sigmetrics conference has devoted at least one entire session exclusively to scheduling papers, which now occupy about 1/8th of the conference.

How did this change come to pass? What are some of the new results that spurred such interest? How do the techniques, models, and metrics of today differ from previous eras? This special issue is devoted to answering these questions. Herein is a collection of seven papers, each surveying a different new area in scheduling research, written by some of the people who made it happen.

It is interesting to observe that one theme that is evident in all the papers is the emphasis on the workload, specifically the distribution of job sizes (service requirements). The last couple decades has seen an explosion in studies of job size distributions in a variety of computing contexts including UNIX job sizes, Web file sizes, FTP file transfers, supercomputing job sizes, Internet flow durations, and more. In all these cases, the job size distribution has been shown to exhibit heavy tails, and be well-modeled by a Pareto distribution, or some other distribution with a power-law tail. This finding was very important in the world of scheduling because it meant that scheduling policies had to be reevaluated in terms of their behavior on these highly-variable, heavy-tailed workloads, often yielding counterintuitive results and "new perspectives" on scheduling. As we briefly overview the papers in this Special Edition, the large role that heavy-tailed workloads play in scheduling will become evident.

The first survey paper in this issue, "Fairness and Classifications," written by Adam Wierman is motivated by the following story: While it has long been understood that scheduling policies that bias towards short jobs improve mean response time, one of the roadblocks that has kept people from implementing such policies is a fear that long jobs will suffer greatly as a consequence of the bias towards short jobs. This fear is challenged directly in Wierman's survey, which describes new theorems, starting around 2001, proving that, counter to intuition, policies like SRPT need not be unfair to large jobs. In fact, when job sizes follow an (unbounded) Pareto distribution, *all* jobs prefer SRPT to Processor-Sharing (PS) with respect to their expected response time. When the job size distribution is bounded, the *largest* job still prefers SRPT to PS, provided load is not too high. After discussing the *fairness* properties of all of today's common scheduling policies, the Wierman survey goes on to introduce the notion of scheduling *classifications*, such as the SMART class of policies. Whereas prior work on scheduling has always dealt with "idealized" scheduling policies, these policies are not always consistent with what can be implemented in practice. A classification like SMART contains a broad range of policies, all of which favor short jobs. Hence it is far more practical to prove fairness or performance guarantees on the whole SMART class, as Wierman does, rather than on individual policies.

Another very important collection of recent theoretical results with practical influence is described in the survey paper titled "Tails in scheduling," authored by Onno Boxma and Bert Zwart. This paper surveys work studying the tail of response time, and provides additional support for SRPT and other SMART policies. While much prior work in scheduling theory focuses on moments of response time, this paper describes recent work on the probability that response time exceeds some value, and the influence of scheduling on the tail behavior. Observe that the tail behavior of response time is of great practical importance in ensuring QoS guarantees, where the service provider must pay a price every time response time exceeds the agreed-upon value. Using a beautiful new theory, which they call *catastrophe theory*, Boxma and Zwart are able to show that large delays in the case of heavy-tailed job sizes are typically due to one very large job arriving, whereas large delays in the case of light-tailed job sizes are typically due to many small jobs arriving at once. This theory enables them to prove many new scheduling results, including a result showing that for heavy-tailed job size dis-

tributions, SMART policies have very good behavior with respect to the response time tail. Specifically, their response time tail mimics the tail of the service time, as compared to other policies like First Come First Served with much worse response time tail.

With all these new theoretical results showing off the powers of SMART policies, it is no wonder that new *computer systems implementations* have begun sprouting up, whose goal is to put these scheduling policies into practice and experience the huge performance gains. The paper, "Scheduling in Practice," written by Ernst Biersack, Bianca Schroeder, and Guillaume Urvoy-Keller, surveys computer systems implementations during the past five years, which are based on some of the above theoretical results. For the case of Web servers, the authors describe implementations of SRPT-based policies. For routers, the authors explain that the size of a flow is not known a priori. Hence they propose instead to use the Least Attained Service (LAS, or equivalently FB) policy, which favors flows that have sent the fewest bytes so far. It turns out that many of the good theoretical properties of SRPT carry over to the LAS policy as well. The authors describe various implementations of LAS for routers and SRPT for web servers and show tremendous improvements in mean response time over existing implementations based on Processor Sharing (PS) scheduling. The paper ends with a discussion of the effect of scheduling in closed versus open system models.

Another area that has been both a source of new scheduling problems and a beneficiary of scheduling solutions is bandwidth sharing in networks. The paper titled, "Scheduling Network Traffic," by Thomas Bonald and James Roberts, presents a survey of algorithms used for sharing bandwidth under a multitude of different conditions, and the analysis of these bandwidth sharing algorithms, in the context of wired and wireless networks. The authors look at two different types of traffic: *elastic traffic* (e.g., data traffic), which is not delay sensitive, and *streaming traffic* (e.g., video or voice traffic), which is delay-sensitive, as well as situations where the two types of traffic appear in combination. The authors show that using scheduling to enforce fair bandwidth sharing is very effective, even when flows are unresponsive. The authors point out that priority scheduling and size-based scheduling can sometimes be quite effective for elastic traffic, however there are many problems that occur under streaming traffic that are still unsolved.

The scheduling algorithm that we see most often in computer systems is Processor Sharing (PS), which is used in both time-shared computer systems, and as a convenient abstraction for modeling flow-level bandwidth-sharing protocols, like TCP, in networks. In recent years, three important variants of PS have become popular. The paper, "Beyond Processor-Sharing," by Samuli Aalto, Urtzi Ayesta, Sem Borst, Vishal Misra, Rudesindo Nunez-Queija, surveys the abundant research on PS variants. Discriminatory Processor Sharing (DPS) and Generalized Processor Sharing (GPS) are both are weighted generalizations of PS, where the resource is split using class-dependent weights. While both DPS and GPS are highly practical, their analysis, unfortunately, is still largely open, although solutions in certain asymptotic regimes are surveyed. One particularly interesting metric is the level of *immunity* of an individual class from the effects of other, perhaps heavier-tailed, classes. The paper surveys several counter-intuitive results on this topic. Multilevel Processor Sharing (MLPS) is actually a very old scheduling policy, typically used in UNIX operating systems. The idea here is that there are multiple levels of queues, labeled highest priority down to lowest priority, each scheduling via PS, where a job moves from the highest priority queue downwards as it ages (receives service) – thus older jobs receive lower priority. The paper surveys many recent theorems, which show, among other things, that when the job size distribution has decreasing hazard rate, MLPS is far superior to a single PS queue. The advantages of changing the scheduling at the queues from PS to FB are also explored.

One of the most interesting and also most challenging areas of scheduling analysis deals with multiserver systems, surveyed in the paper titled, "Stochastic Analysis of Multiserver Systems," by Mark Squillante. Multiserver systems come in many forms. A well-known case is the *server farm* architecture, consisting of a front-end dispatcher, whereby each incoming job is immediately routed to one of the servers in the server farm. Typical dispatching policies used include Join-the-Shortest-Queue (JSQ), which has received much attention by researchers, but whose analysis is still largely open for more than two queues. Other interesting examples of multiserver architectures include *cycle stealing*, or *coupled-processor models*, whereby one server can "help" another server when its own queue is empty. Like JSQ, such problems with dependencies between queues suffer from the curse of high dimensionality, since any Markov model tracking the state of the system needs to track the state at all queues, and hence grows unboundedly in two (or more) dimensions, depending on the problem. Squillante does an excellent job of surveying the latest techniques used to deal with these high-dimensional models, including boundary value techniques, dimensionality reduction techniques, and others. He also surveys heavy-traffic and light-traffic results where the analysis is more tractable. Multiserver analysis is likely to be an area of huge growth in scheduling research in the future.

While all of the above papers cover scheduling research in the stochastic arena, where there is a stochastic process governing job arrivals and job sizes, there has simultaneously been a plethora of research dealing with *worst-case* performance bounds. In his paper titled, "Competitive Online Scheduling for Server Systems," Kirk Pruhs surveys work predominantly by the theoretical computer science community, appearing in conferences like STOC, FOCS, and SPAA, where scheduling policies are evaluated based on their *competitive ratio*. A scheduling policy which is "2-competitive" will perform within a factor of 2 of the optimal possible performance on every possible arrival sequence of jobs (where what is "optimal" naturally changes for different arrival sequences). While there is little overlap between the stochastic community and the worst-case community, I feel that these two different communities could learn a lot from each other. As an example of a difference between the results in the two communities, consider an SRPT queue, being served by two servers, where at any moment of time the two jobs being served are those two jobs having the shortest remaining processing times. While such a multiserver system is not tractable in any common stochastic setting, the competitive ratio for this problem is well known. The competitive ratio is also known in the case of a server farm where each of the servers maintains an SRPT queue and clever immediate dispatching is used. Many such interesting competitive ratio results are covered in Pruhs' survey.

I sincerely thank each of the authors that contributed to this issue. It is always difficult to put together a survey paper on results that are still quite new. The authors all did an excellent job of writing very readable and comprehensive papers. Finally, I am extremely grateful to Evgenia Smirni, for her much-appreciated guidance and help in putting together this issue.