SIGACT News Online Algorithms Column 21: APPROX and ALGO

Rob van Stee
Max Planck Institute for Informatics
Saarbrücken, Germany

For this column, I would like to report on some conferences that I attended recently, with special attention given (of course) to results in online algorithms. Most of the papers discussed below consider online scheduling, another favorite problem area of mine. Generally, for a job j we will denote its processing time by p_j, its weight by w_j (if applicable), its release time by r_j, and its deadline by d_j. In the discussion below, I have tried to point out interesting open problems wherever possible. Enjoy!

1 APPROX

The first conference I will discuss is the APPROX part of APPROX-RANDOM, which this year took place at MIT in Boston. The conference was held in a highly unusual building as you can see on the photo, which does not appear to have any straight walls at all. Many of you might know this building already (especially if you are in the US), but this was actually my first visit to MIT, and I thought it was really quite a sight to behold. Most of the papers there focused on approximation algorithms, but there were a few nice papers about online algorithms as well—at least I like to think so!

Minimizing flow time with preemptions Ho-Leung Chan, Tak-Wah Lam and Rongbin Li [5] revisited the problem of minimizing the total flow time of online jobs. This is a notoriously hard problem and it is clear that no algorithm can be competitive without preemptions. With preemptions, at least on a single machine, it is suddenly possible to be 1-competitive by using the Shortest Remaining Processing Time (SRPT) algorithm. However, this assumes that preemptions are free, in the sense that they are instantaneous and do not cause any overhead.

The authors consider the case where preemptions are not free but cause extra work for the processor. This model was introduced by Heydari et al. [12], after Bartal et al. [4] first considered a

Figure 1: The Ray and Maria Stata Center

model where preemptions cause an abstract amount of overhead, which is accounted for separately but does not delay the running of the jobs. In the latter model, a constant competitive algorithm for minimizing the total flow time is known. Surprisingly, Chan et al. [5] show that with this small change, it is no longer possible for an online algorithm to be competitive. Specifically, they give a lower bound of $\Omega((\frac{\delta}{1+\delta})^{1/4}n^{1/4})$ for the competitive ratio of any online algorithm on a single machine, where n is the number of jobs and δ is the ratio of preemption overhead to the minimum job size.

Using resource augmentation on the speed, the authors give a $(1+\varepsilon)$-speed $(1+1/\varepsilon)$-competitive algorithm. This algorithm is a variation of SRPT, called QSRPT, which processes *quantums* of jobs (which have a certain fixed size) before possibly preempting a job for a smaller one. This algorithm is then generalized to deal with parallel machines, where it is assumed that there is a certain overhead cost for a preemption and another, larger cost if the job is later started on a different machine.

Jobs with fixed start times Together with Leah Epstein, Łukasz Jeż, and Jiří Sgall, I considered the problem of scheduling jobs that have fixed starting times [8]. That is, it is not possible to store jobs that arrive, but they must be assigned to a machine immediately upon arrival, and each machine can handle only one job at a time. Hence, if a job is preempted, it is lost forever. Previously this problem had been considered on one machine [16] and on parallel machines [10], where it equates to scheduling intervals with weights. We extend it to related machines.

The most general problem, with arbitrary weights and lengths for the jobs, does not admit a bounded competitive ratio, and we consider some special cases that do. To begin with, we show that even the case of unit-weight jobs (and arbitrary sizes) does not admit a competitive ratio below m. It is simple to achieve this ratio by using the fastest machine esclusively.

Even for the simplest case, where all jobs have size and weight equal to 1, it is not possible to maintain a competitive ratio of 1 (unlike for identical machines), since an online algorithm needs to choose machines for arriving jobs and may make mistakes. It is easy, however, to achieve a competitive ratio of 2, simply by using a greedy algorithm which schedules each arriving job on an arbitrary idle machine, if there is such a machine at this time. The ratio of 2 follows because during every job that the greedy algorithm runs, at most one job can finish in the optimal solution and at most one job can start. Assigning both these jobs to the job of the greedy algorithm proves the ratio of 2 (note that no job is ever preempted). We could show a lower bound of 1.56 for a version of the greedy algorithm which prefers the fastest idle machine, and a general lower bound of 1.5, but we were unable to break the trivial upper bound of 2. This is an intriguing open question.

We also considered several other standard cases, in particular weighted jobs with unit sizes and jobs with proportional weight ($w_j = p_j$ for every job j). We give a 4-competitive algorithm for both these cases. The algorithm uses an arbitrary idle machine if one is available, and otherwise preempts a job that has less than half the size of the new job, if such a job is running on some machine. This generalizes the results for one machine for these job classes by Gerhard Woeginger [16], where the ratio of 4 is tight. For related machines, the best lower bound is 1.693 [9, 10].

2 ALGO

In September, I attended ALGO in Ljubljana, the capital of Slovenia. For the excursion we went to an impressive set of caves called the Postojna Cave, which is the second largest cave system in the country and was very impressive. We took a train ride inside the cave and then got a tour. I guess the photo might not work so well if you are reading a printed version of this, but at least online it should be nice.

I had hoped to also include some pictures of participants here, but unfortunately I dropped my camera on the rock floor of the cave soon after I made this picutre, which prevented me from making any further pictures. However, many pictures of participants can be found at the ALGO website: http://algo12.fri.uni-lj.si/?file=gallery

2.1 ESA

Jiří Sgall gave a very nice invited talk about several open problems in throughput scheduling. He wrote an accompanying paper for the ESA proceedings which I encourage you to read [15]. It would be nice if more invited talks were also published in a written form like this one, but it seems to be rather the exception to the norm. I mentioned one open problem already above—designing a better than 2-competitive online algorithm for jobs with fixed start times on related machines, where for each job we have $w_j = p_j = 1$.

There are in fact several open problems involving this apparently simple class of jobs. To be more precise, let us consider jobs with equal size p and integer release times and deadlines (so this is not the same as having unit-sized jobs). The start times are no longer fixed, but a job needs to complete by its deadline in order for the online algorithm to gain a profit from it. It is again simple to get a competitive ratio of 2 just as before (using a greedy algorithm). Moreover, this is the best you can do with a deterministic algorithm, at least on a single machine. But what about *randomized* algorithms? For such algorithms, there is a lower bound of 4/3 and an upper bound of 5/3. The algorithm uses only one bit of randomness and chooses with equal probability one of two

Figure 2: The Postojna Cave

deterministic algorithms to use at the start of the execution (a *barely random* algorithm). Can we do better?

It would be pointless to repeat Jiří's paper here in full, so I'll just stop here and refer you to his paper [15] for more information on the open problems and for references.

The value of job migration Susanne Albers and Matthias Hellwig [1] considered what I suppose is the most fundamental problem in online scheduling: minimizing the makespan on parallel machines. Here jobs arrive in a list instead of over time, and each job needs to be assigned to one of the m machines before the next job becomes known. This problem has received a lot of attention in the past and it has been known since 2001 that the optimal competitive ratio lies in the interval $[1.88, 1.92]$. There has been no further progress since then, and indeed it seems very hard to see how either the upper or the lower bound should be improved.

We only have tight bounds for two and three machines; for four machines, I conjecture that the optimal competitive ratio is $\sqrt{3}$, which is the current lower bound [13]. I expect that the optimal algorithm would focus on input sequences like the lower bound sequences, which consist of groups of four approximately equal-sized jobs, with the sizes increasing exponentially from one group to the next. Most likely the algorithm would have to behave differently based on whether there are already one, two, or three jobs of the most recent group, and work towards maintaining a certain profile (load distribution) of the jobs. While I would be very interested to see this conjecture being proved or disproved, I should also point out that the best known upper bound for four machines is $1.73\overline{3}$, so we are already very close to optimal here.

Susanne and Matthias [1] look at how the competitive ratio changes if you allow migration.

Obviously, if an online algorithm can completely rearrange the schedule in every step, it becomes an offline algorithm, so typically, some limits are placed on how much you can migrate in any given step of the input. In the current paper, an algorithm with competitive ratio α_m is presented, where α_m is the solution of an equation representing load in an ideal machine profile for a subset of the jobs. The resulting competitive ratio tends to 1.4659 (from above) as m tends to infinity. The algorithm uses at most $7m$ migrations for $m \geq 11$, and at most $10m$ migrations for smaller m. The authors show that this algorithm is optimal in the sense that no deterministic algorithm that uses $o(n)$ job migrations can do better. They also give a family of algorithms that achieve competitive ratios between 5/3 and 2, using increasing migration to get a better ratio, maxing out at $4m$ migrations to achieve the ratio of 5/3.

Interestingly, the ratio of α_m was also found by Matthias Englert, Deniz Özmen and Matthias Westermann [6] for a somewhat similar problem, where the online algorithm has the option of storing some jobs in a buffer before assigning them to machines (instead of the power to migrate jobs). The authors of the new paper put as an open question whether the two models are equivalent, writing that while they can transform their algorithms into algorithms that use a buffer without a loss in the performance guarantee, it is not clear how to translate the algorithms from [6] into algorithms that use migration.

2.2 WAOA

As will surprise absolutely no one, there were of course various papers on online algorithms at this workshop. Here I will discuss just a few of them.

Online labeling Suppose n items arrive online to be stored in an array of size $m > n$. The items have an intrinsic order and need to be stored in the correct order, but of course when an item arrives we do not know where in the global ordering it belongs. In order to maintain a stored order, sometimes we will have to move items around, let us say at a cost of 1 per item. This is known as the online labeling problem (the location of an item in the array can be seen as a label). It is intuitively clear that as m grows, you will need less relabeling operations (moving items around), but how much less exactly?

This problem has been studied by various authors for different values of m. Babka et al. [2] consider the case where $m = \Omega(n^C)$ for $C > 1$. They fix a gap in the previous proof of the lower bound construction and also give a simpler and more precise lower bound. Their final result is a lower bound of $\Omega((n \log n)/(\log\log m - \log\log n))$ for m between $n^{1+\varepsilon}$ and 2^{n^ε}. For polynomially many labels, this reduces to $\Omega(n \log n)$, and a matching upper bound is known.

An interesting question that is left open is whether it is possible to design a better online labeling algorithm if the n items are chosen from a relatively small set, say of size $m \log n$. The current lower bound construction requires that the set from which the n items are picked has size exponential in n. It is also not known whether randomization can help to improve the upper bound.

Page migration In this problem, requests for a (fixed) page appear at several nodes in a network. The page has a location which may be changed for a cost, and the question that an online algorithm faces for each request is whether to move the page to a node closer to the request point, or serve it from the current location. It is assumed that moving a page over a certain distance costs D times as much as serving it over the same distance. This is one of the classic problems in competitive analysis.

Black and Sleator were the first to analyze the competitive ratio of this problem. They gave a 3-competitive algorithm for several types of networks and a matching lower bound, and conjectured that a 3-competitive algorithm existed for every network. This was disproved a couple of years later, already for networks with four nodes, with lower bound constructions using $D = 1$. At the same time, it was shown that networks with three nodes do admit 3-competitive algorithms, at least if $D = 1$. Later, a general upper bound of 4.086 was shown.

These results left open the possibility that for networks with three nodes, there might be an online algorithm with asymptotic competitive ratio 3 as D tends to infinity. Akira Matsubayashi [14] presented a work function algorithm at WAOA with competitive ratio $3 + 1/D$, along with a lower bound of $3 + \Omega(1/D)$ for every $D \geq 3$. For $D = 2$, he gave a 3-competitive algorithm.

It is left open by this work what happens for larger networks and larger D. Neither a $3 + o(1)$-competitive algorithm nor a $3 + \Omega(1)$ lower bound are known. The work function algorithm used in this paper is known not to be $3 + o(1)$-competitive, but it is 3-competitive on roughly uniform networks, where all edge weights are between 1 and 4/3.

Black and white bin packing János Balogh et al. [3] considered a version of bin packing (see also my previous column on this) where every item has a color (black or white) and items need to be packed into the bins in such a way that the colors alternate in each bin. For the offline version, an efficient 2.5-approximation algorithm and a nontrivial APTAS are presented, and for the online version, the authors give an algorithm with an absolute competitive ratio of 3.

The algorithm works by first packing the items under the assumption that all their sizes are 0 (using Any Fit), and then repacking items into new bins whenever the total size packed into a bin exceeds 1. Clearly the second part of this can be done online. Note that the first part can easily require many bins, in the case that many items of the same color arrive in sequence. For the competitive analysis, the online algorithm is compared against an optimal solution which is required to pack the items in the same order as they arrive, since otherwise the situation is clearly hopeless: by reordering the sequence, it can be that you need arbitrarily many fewer bins. Thus, the adversary is of the restricted offline kind.

It is shown that classical algorithms like various *-Fit algorithms and Harmonic have competitive ratio of at least 3 (Next Fit and Harmonic are in fact not even constant competitive), and a general lower bound of 1.72 is presented, showing that this problem has a higher competitive ratio than standard bin packing (where an upper bound of 1.59 is known).

Of course, the most obvious open question here is to reduce the gap between the upper and the lower bound. Personally I would think that the upper bound in particular is a good candidate for improvement, mostly because the lower bound construction appears to be significantly more complicated than the analysis of the algorithm. But of course, this is only a feeeling and such things can very well be deceptive.

References

[1] Susanne Albers and Matthias Hellwig. On the value of job migration in online makespan minimization. In Epstein and Ferragina [7], pages 84–95.

[2] Martin Babka, Jan Bulánek, Vladimír Cunát, Michal Koucký, and Michael Saks. On online labeling with polynomially many labels. In Epstein and Ferragina [7], pages 121–132.

[3] János Balogh, József Békési, Gyorgy Dosa, Hans Kellerer, and Zsolt Tuza. Black and white bin packing. In *Proc. 10th Workshop on Approximation and Online Algorithms (WAOA 2012)*. To appear.

[4] Yair Bartal, Stefano Leonardi, Gil Shallom, and René Sitters. On the value of preemption in scheduling. In Josep Díaz, Klaus Jansen, José D. P. Rolim, and Uri Zwick, editors, *APPROX-RANDOM*, volume 4110 of *Lecture Notes in Computer Science*, pages 39–48. Springer, 2006.

[5] Ho-Leung Chan, Tak Wah Lam, and Rongbin Li. Online flow time scheduling in the presence of preemption overhead. In Gupta et al. [11], pages 85–97.

[6] Matthias Englert, Deniz Özmen, and Matthias Westermann. The power of reordering for online minimum makespan scheduling. In *FOCS*, pages 603–612. IEEE Computer Society, 2008.

[7] Leah Epstein and Paolo Ferragina, editors. *Algorithms - ESA 2012 - 20th Annual European Symposium, Ljubljana, Slovenia, September 10-12, 2012. Proceedings*, volume 7501 of *Lecture Notes in Computer Science*. Springer, 2012.

[8] Leah Epstein, Łukasz Jeż, Jiří Sgall, and Rob van Stee. Online scheduling of jobs with fixed start times on related machines. In Gupta et al. [11], pages 134–145.

[9] Leah Epstein and Asaf Levin. Improved randomized results for the interval selection problem. *Theoretical Computer Science*, 411(34-36):3129–3135, 2010.

[10] Stanley P. Y. Fung, Chung Keung Poon, and Duncan K. W. Yung. On-line scheduling of equal-length intervals on parallel machines. *Inf. Process. Lett.*, 112(10):376–379, 2012.

[11] Anupam Gupta, Klaus Jansen, José D. P. Rolim, and Rocco A. Servedio, editors. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques - 15th International Workshop, APPROX 2012, and 16th International Workshop, RANDOM 2012, Cambridge, MA, USA, August 15-17, 2012. Proceedings*, volume 7408 of *Lecture Notes in Computer Science*. Springer, 2012.

[12] Mehdi Heydari, Seyed Sadjadi, and Emran Mohammadi. Minimizing total flow time subject to preemption penalties in online scheduling. *The International Journal of Advanced Manufacturing Technology*, 47:227–236, 2010. 10.1007/s00170-009-2190-9.

[13] John F. Rudin III and R. Chandrasekaran. Improved bounds for the online scheduling problem. *SIAM J. Comput.*, 32(3):717–735, 2003.

[14] Akira Mtsubayashi. Optimal online page migration on three points. In *Proc. 10th Workshop on Approximation and Online Algorithms (WAOA 2012)*. To appear.

[15] Jiří Sgall. Open problems in throughput scheduling. In Epstein and Ferragina [7], pages 2–11.

[16] Gerhard J. Woeginger. On-line scheduling of jobs with fixed start and end times. *Theoret. Comput. Sci.*, 130:5–16, 1994.