Working Notes

Workshop on Recent Trends in Online Algorithms

The 50th EATCS International Colloquium on Automata, Languages and Programming

2023

Workshop Organizers

Thomas Erlebach (Durham University, UK)

Kazuo Iwama (Kyoto University, Japan)

Chung-Shou Liao (National Tsing Hua University, Taiwan)

Nicole Megow (University of Bremen, Germany)

Website

http://acolab.ie.nthu.edu.tw/icalp23-workshop/

Contents

Monday, July 10, 2023		
Session	Time (UTC+2)	Title
	08:30-09:00	Registration
	09:00-09:10	Opening
[Session 1] Scheduling and explorable or stochastic uncertainty	09:10-09:40	***Orienting (hyper)graphs under explorable stochastic uncertainty*** ***Christoph Dürr***
	09:40-10:10	***Configuration Balancing for Stochastic Requests*** ***Franziska Eberle***
	10:10-10:30	Scheduling with Explorable Uncertainty Revisited Hsiang-Hsuan Liu
	10:30-11:00	Coffee Break
[Session 2] Open problems	11:00-11:30	***The Randomized k-Server Conjecture is False!*** ***Christian Coester***
	11:30-12:30	Open Problems
	12:30-14:00	Lunch
[Session 3] Delays and revocable decisions	14:00-14:30	***List Update with Delays or Time Windows*** ***Yossi Azar***
	14:30-15:00	***Online Facility Location with Linear Delay*** ***Marcin Bieńkowski***
	15:00-15:20	Reservations in Online Algorithms Matthias Gehnen
	15:20-15:40	Relaxing the Irrevocability of Decisions for the Online Knapsack Problem Moritz Stocker
	15:40-16:10	Coffee Break
[Session 4] Learning-augmented algorithms and scheduling	16:10-16:40	***New Directions on Algorithms with Predictions*** ***Benjamin Moseley***
	16:40-17:00	Learning-Augmented Online TSP on Rings, Trees, Flowers and (almost) Everywhere Else Michalis Xefteris
	17:00-17:20	Speed-Oblivious Online Scheduling: Knowing (Precise) Speeds is not Necessary Alexander Lindermayr
	17:20-17:40	Online Dynamic Power Management: from One to Multiple Ya-Chun Liang

Orienting (hyper)graphs under explorable stochastic uncertainty

Christoph Dürr (Keynote Speech)
Sorbonne Université, CNRS, France

Abstract

Given a hypergraph with uncertain node weights following known probability distributions, we study the problem of querying as few nodes as possible until the identity of a node with minimum weight can be determined for each hyperedge. Querying a node has a cost and reveals the precise weight of the node, drawn from the given probability distribution. Using competitive analysis, we compare the expected query cost of an algorithm with the expected cost of an optimal query set for the given instance. For the general case, we give a polynomial-time $f(\alpha)$-competitive algorithm, where $f(\alpha) \in [1.618 + \epsilon, 2]$ depends on the approximation ratio α for an underlying vertex cover problem. We also show that no algorithm using a similar approach can be better than 1.5-competitive. Furthermore, we give polynomial-time 4/3-competitive algorithms for bipartite graphs with arbitrary query costs and for hypergraphs with a single hyperedge and uniform query costs, with matching lower bounds.

Configuration Balancing for Stochastic Requests

Franziska Eberle (Keynote Speech)
Department of Mathematics, LSE, United Kingdom

Abstract

The configuration balancing problem with stochastic requests generalizes well-studied resource allocation problems such as load balancing and virtual circuit routing. There are given m resources and n requests; each request has multiple possible configurations, each of which increases the load of each resource by some amount. The goal is to select one configuration for each request to minimize the makespan: the load of the most-loaded resource. In the stochastic setting, the amount by which a configuration increases the resource load is uncertain until the configuration is chosen, but we are given a probability distribution.

We develop both offline and online algorithms for configuration balancing with stochastic requests. When the requests are known offline, we give a non-adaptive policy for configuration balancing with stochastic requests that $O(\frac{\log m}{\log \log m})$-approximates the optimal adaptive policy, which matches a known lower bound for the special case of load balancing on identical machines. When requests arrive online in a list, we give a non-adaptive policy that is $O(\log m)$ competitive. Again, this result is asymptotically tight due to information-theoretic lower bounds for special cases (e.g., for load balancing on unrelated machines). Finally, we show how to leverage adaptivity in the special case of load balancing on related machines to obtain a constant-factor approximation offline and an $O(\log \log m)$-approximation online. A crucial technical ingredient in all of our results is a new structural characterization of the optimal adaptive policy that allows us to limit the correlations between its decisions.

This is joint work with Anupam Gupta, Nicole Megow, Ben Moseley, and Rudy Zhou.

The Randomized k-Server Conjecture is False!

Christian Coester (Keynote Speech)
Department of Computer Science, University of Oxford, United Kingdom

Abstract

The randomized k-server conjecture, which had been open for over three decades, states that there exists an $O(\log k)$-competitive randomized algorithm for the k-server problem. In this talk, I will present our recent joint work with Sébastien Bubeck and Yuval Rabani, where we refute this conjecture by giving a lower bound of $\Omega((\log k)^2)$. Our work also settles the competitive ratio of metrical task systems to be $\Theta((\log n)^2)$ on the hardest metric spaces and $\Theta(\log n)$ on the easiest metric spaces of n points. In particular, this yields the first improvement over the previous "coupon collector" lower bound since the introduction of the model in 1987.

List Update with Delays or Time Windows

Yossi Azar (Keynote Speech)
School of Computer Science, Tel-Aviv University, Israel

Abstract

We consider the problem of List Update one of the most fundamental problems in online algorithms and competitive analysis. Informally, we are given a list of elements and requests for these elements that arrive over time. Our goal is to serve these requests, at a cost equivalent to their position in the list, with the option of moving them towards the head of the list. Sleator and Tarjan introduced the famous "Move to Front" algorithm (wherein any requested element is immediately moved to the head of the list) and showed that it is 2-competitive. While this bound is excellent, the absolute cost of the algorithm's solution may be very large (e.g., requesting the last half elements of the list would result in a solution cost that is quadratic in the length of the list). Thus, we consider the more general problem wherein every request arrives with a deadline and must be served, not immediately, but rather before the deadline. We further allow the algorithm to serve multiple requests simultaneously - thereby allowing a large decrease in the solutions' total costs. We denote this problem as List Update with Time Windows. While this generalization benefits from lower solution costs, it requires new types of algorithms. In particular, for the simple example of requesting the last half elements of the list with overlapping time windows, Move-to-Front fails. In this work we show an $O(1)$ competitive algorithm. The algorithm is natural, but the analysis is a bit complicated, and a novel potential function is required. Thereafter we consider the more general problem of List Update with Delays in which the deadlines are replaced with arbitrary delay functions. This problem includes as a special case the prize collecting version in which a request might not be served (up to some deadline) and instead suffers an arbitrary given penalty. Here we also establish an $O(1)$ competitive algorithm for general delays. The algorithm for the delay version is more complex and its analysis is significantly more involved.

This is joint work with Shahar Lewkowicz and Danny Vainstein.

Online Facility Location with Linear Delay

Marcin Bieńkowski (Keynote Speech)
Institute of Computer Science, University of Wrocław, Poland

Abstract

In recent years many online problems have been studied in the setting "with delay", where incoming requests do not have to be served immediately, but can be delayed and served together at a reduced cost. Delaying comes with a certain penalty though: it incurs a waiting cost equal to the difference between the request arrival and its serving time.

The problems studied in this framework include many server variants and network design problems such as matching, Steiner forest, or facility location. While many clever approaches have been proposed, their competitive ratios in general metric spaces are at least logarithmic.

In this talk, I will present an algorithm for the online facility location with delays that beats this logarithmic barrier (albeit for linear delays only). It is inspired by greedy algorithms for the offline case and analyzed using a sequence of factor-revealing LPs.

New Directions on Algorithms with Predictions

Benjamin Moseley (Keynote Speech)
Carnegie Mellon University, United States

Abstract

This talk will discuss a model for augmenting algorithms with useful predictions to improve algorithm performance. The model ensures predictions are formally learnable and robust. Learnability guarantees that predictions can be efficiently constructed from past data. Robustness formally ensures a prediction is robust to modest changes in the problem input. This talk will discuss new directions on using predictions including for data structures and incorporating multiple forms of advice.

Scheduling with Explorable Uncertainty Revisited

Hsiang-Hsuan (Alison) Liu * Fu-Hong Liu † Prudence W.H. Wong ‡
Xiao-Ou Zhang §

In this work, we consider the *Scheduling with Explorable Uncertainty* (SEU) problem with the objective total completion time. In the model, the processing time of a job is uncertain, but only an *upper limit* of the processing time is known to the algorithm. By performing a *test* on a job, an algorithm gets the actual processing time of the job, which is at most the upper limit. After testing, the job can be processed by the actual processing time. Otherwise, the job can only be processed by the upper limit of the processing time. We aim to minimize the total completion time while finishing all jobs.

Formally, given n jobs, each job j has an *upper limit* u_j of its *actual processing time* p_j where $0 \le p_j \le u_j$ and a *testing time* t_j. Any job j can be executed after testing or executed without testing. In the former case, the time needed to process job j is $t_j + p_j$, while in the latter case, the time needed is u_j. Let c_j^A be the completion time of job j by a schedule A, the *cost* of schedule A is $\sum_{j=1}^{n} c_j^A$.

It looks like that by testing a job, the processing time may be shortened. However, the test also takes time, and it does not guarantee that the actual processing time of a job is shorter than the upper limit. Moreover, the decision to test or execute an untested job is irrevocable. Therefore, we consider the problem of scheduling with explorable uncertainty as an online problem. The aim is to design online algorithms that are competitive against the optimal solution that knows the actual processing time without actually testing a job. That is, the optimal solution knows exactly if a job's processing time can be reduced by testing. Yet, the optimal solution needs to pay a job's testing time in order to process the job with its actual processing time.

Previous work. The problem was first introduced by Dürr et al. [2, 3]. In this work, the authors considered a case where the testing time of jobs is uniform (that is, $t_j = 1$ for all j) and proposed a 2-competitive algorithm Threshold. They further studied a special case where all upper limits are uniform (that is, $u_j = 1$ for all j) and provided a Beat algorithm that is 1.9338-competitive. The authors also showed a lower bound of deterministic algorithms' competitive ratio of 1.8546.

Later, Albers and Eckl [1] consider a more general case where the testing times of jobs are variable. They proposed the (α, β)-Sort algorithm. Intuitively, the (α, β)-Sort algorithm tests a job j if and only if $u_j \ge \alpha \cdot t_j$. Depending on whether a job is tested or

*h.h.liu@uu.nl. Department of Information and Computing Sciences, Utrecht University, The Netherlands

†homer.fhliu@gmail.com.

‡P.Wong@liverpool.ac.uk. Department of Computer Science, The University of Liverpool, The United Kingdom

§xzhang1404@gmail.com. Department of Information and Computing Sciences, Utrecht University, The Netherlands

not, the job is transformed into one task (untesting task) or two tasks (testing task and execution task). These tasks are then maintained in a priority queue for the algorithm to decide their processing order. More specifically, a testing task has a weight of $\beta \cdot t_j$, an execution task has a weight of p_j, and an untesting task has a weight of u_j. When $\alpha = \beta = 1$, the competitive ratio of (α, β)-SORT is 4. On the other hand, the authors considered the *preemptive* case where jobs could be interrupted and resumed later and provided a $2\phi = 3.236$-competitive algorithm GOLDENROUNDROBIN.

Our contribution. In this work, we first revisit the techniques introduced by Albers and Eckl [1]. We show that by introducing amortization, the competitive ratio of SORT can be improved from 4 to $1 + \sqrt{2} \approx 2.414$.

Theorem 1 *The* $(\sqrt{2}, \sqrt{2})$-SORT *algorithm is* $1+\sqrt{2}$*-competitive for the* SEU *problem.*

Furthermore, we slightly modify the (α, β)-SORT algorithm. Instead of assigning the weight of p_j to an execution task, we set the weight $t_j + p_j$.

Theorem 2 *There exists a* 2.31652*-competitive algorithm for the* SEU *problem.*

Finally, we propose a potential function for the BEAT algorithm in [2, 3], which deals with a special case of the SEU problem, where all jobs have the same upper limit u and the same testing time 1. We focus on the case where $u = 2$. The potential function is based on Lemma 13 in [2, 3] that there exists a worst case instance where $p_j \in \{0, 1, 2\}$ for most of the jobs j.

Let A_i, B_i, and Γ_i be the set of 0-jobs, 1-jobs, and 2-jobs in the first i jobs touched by the algorithm, respectively. Let α_i, β_i, and γ_i be the cardinality of A_i, B_i, and Γ_i, respectively. We define the potential function after testing the $i+1$-th job as

$$\sum_{j\in A_i}(0.927\alpha_j - 0.073\beta_j - 0.573\gamma_j) + \sum_{j\in B_i}(0.927\alpha_j + 1.854\beta_j + 0.854\gamma_j) + \sum_{j\in\Gamma_i}(0.9\alpha_j + 1.854\beta_j + 0.354\gamma_j).$$

Note that the potential is always greater than or equal to 0. One can show that the change of the potential after testing a job always compensates the algorithm's cost, such that it is at most 1.927 times of the optimal cost. Therefore. the potential function provides an alternative proof that BEAT is 1.927-competitive when $u_j = 2$ for all j. A similar method can be extended to the case where $u = 1.9338$ and that BEAT is 1.9338-competitive in this case.

References

[1] S. Albers and A. Eckl. Explorable uncertainty in scheduling with non-uniform testing times. In C. Kaklamanis and A. Levin, editors, *WAOA*, volume 12806 of *Lecture Notes in Computer Science*, pages 127–142. Springer, 2020.

[2] C. Dürr, T. Erlebach, N. Megow, and J. Meißner. Scheduling with explorable uncertainty. In A. R. Karlin, editor, *ITCS*, volume 94 of *LIPIcs*, pages 30:1–30:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[3] C. Dürr, T. Erlebach, N. Megow, and J. Meißner. An adversarial model for scheduling with testing. *Algorithmica*, 82(12):3630–3675, 2020.

Reservations in Online Computation

Matthias Gehnen *

1 Reservation Model in Online Computation

A recent trend in online algorithms is to look into modifications of the classical model in order to get a more fine-grained analysis of online complexity for different problems, as well as studying problems that commonly do not admit a constant competitive ratio. Such modifications include delaying decisions of an online algorithm until its intermediate solution is no longer valid for the prefix of an instance in the *preemption* and *late accept* model [2], communicating information to the algorithm, either about the instance when modeling *predictions* or arbitrary information in the form of *advice* or to allow an online algorithm to use *randomization* [4].

1.1 Reservation Model

In this talk we want to present a recent modification which was first called *Online Algorithms with Reservations*. This model is an extension of the classical online model, adding a new option for an algorithm upon being presented the next element of the input sequence. For a fixed cost α, it has the option to ignore the current element of the input sequence and to delay the decision on this element until the end of the input sequence. Of course, the final decision on these *reserved* elements, together with the previously made decision has to result in an overall valid solution. The algorithm then has to deduct the accumulated costs from its gain in the calculation of the competitive ratio.

1.2 Knapsack

This model was first introduced by Böckenhauer et al. in 2021 [1], in which the Online Simple Knapsack Problem was studied. While the classical Online Simple Knapsack Problem does not admit any constantly bounded competitive ratio in the deterministic setting, they found that adding the possibility of reservation makes the problem constantly competitive, with varying competitive ratios depending on the value of the fraction α. They gave tight bounds for the whole range of reservation costs, which is split up into four connected areas, depending on the value of α.

*gehnen@cs.rwth-aachen.de. Department of Computer Science, RWTH Aachen, Germany

The option to reserve items has also been applied to the Online Simple Knapsack Problem with removals. In the paper, we combined reservations with *removability*, in which an item that was previously packed into the knapsack may be finally discarded at any point. If both are are permitted, we showed that the competitive ratio of the Online Simple Knapsack Problem rises depending on the relative reservation costs, with the competitive ratio being split into two segments.

1.3 Other Problems

Finally, this modification was also applied to the Secretary Problem [3], where a reservation fee can be paid to keep candidates on a short-list instead of rejecting them on the spot. We looked at two versions of paying these costs, one in which the fee has to be paid only once and the other in which the fee has to be paid in every round as long as the reservation is kept. We analyzed the competitive ratio for both variants and present optimal, relatively simple strategies.

In this talk, we want to discuss the results and strategies of these papers and show connections between the reservation model and Late Accept/Preemption models in the form of lower- and upper-bound reductions. We also discuss how to apply this model to the Vertex Cover, Feedback Vertex Set and other graph problems.

References

[1] Hans-Joachim Böckenhauer, Elisabet Burjons, Juraj Hromkovic, Henri Lotze, and Peter Rossmanith. Online simple knapsack with reservation costs. In Markus Bläser and Benjamin Monmege, editors, *38th International Symposium on Theoretical Aspects of Computer Science, STACS 2021, March 16-19, 2021, Saarbrücken, Germany (Virtual Conference)*, volume 187 of *LIPIcs*, pages 16:1–16:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[2] Joan Boyar, Lene M. Favrholdt, Michal Kotrbcík, and Kim S. Larsen. Relaxing the irrevocability requirement for online graph algorithms. *Algorithmica*, 84(7):1916–1951, 2022.

[3] Elisabet Burjons, Matthias Gehnen, Henri Lotze, Daniel Mock, and Peter Rossmanith. The secretary problem with reservation costs. In Chi-Yeh Chen, Wing-Kai Hon, Ling-Ju Hung, and Chia-Wei Lee, editors, *Computing and Combinatorics - 27th International Conference, COCOON 2021, Tainan, Taiwan, October 24-26, 2021, Proceedings*, volume 13025 of *Lecture Notes in Computer Science*, pages 553–564. Springer, 2021.

[4] Dennis Komm. *An Introduction to Online Computation - Determinism, Randomization, Advice.* Texts in Theoretical Computer Science. An EATCS Series. Springer, 2016.

Relaxing the Irrevocability of Decisions for the Online Knapsack Problem

Hans-Joachim Böckenhauer [*] Fabian Frei [†] Dennis Komm [‡]
Peter Rossmanith [§] Moritz Stocker [¶]

1 Introduction

Although competitive analysis is one of the major tools for analyzing online algorithms and has lead to many strong results, it has often been argued that this worst-case approach is overly pessimistic and too coarse-grained to always give a clear picture of the complexity of online problems. There have been many attempts to achieve a more fine-grained picture of the nature of online computation by relaxing the limitations of online algorithms. These limitations can be roughly described by the following characteristics of an online algorithm:

- (U) The complete instance is unknown.
- (I) The decisions are irrevocable.
- (A) The inputs are generated by a malicious adversary.

A lot of work has been done to relax the U-constraint, including several kinds of semi-online algorithms or the use of advice complexity for analyzing online problems, see for example the analysis for the knapsack problem [3]. Weakening the A-constraint, i.e., the power of the adversary, has been studied, for example, by introducing some kind of randomness to the model, e.g., in the random order model [1] or by drawing the requests from some fixed probability distribution [7].

2 Results

In this talk, we focus on relaxing the I-constraint, using the knapsack problem as an example. When filling a knapsack, there are several possibilities to relax the irrevocability of decisions. One version is to allow (unlimited) removal of already packed items, either for free [6] or at some cost [5]. Another possible model is that of recourse, where a limited number of decisions can be withdrawn. For the online knapsack problem with removal, for example, a constant number of previously removed items could be re-packed [2].

[*] hjb@inf.ethz.ch. Department of Computer Science, ETH Zürich, Switzerland
[†] fabian.frei@cispa.de. CISPA Helmholtz Center for Information Security, Germany
[‡] dennis.komm@inf.ethz.ch. Department of Computer Science, ETH Zürich, Switzerland
[§] rossmani@cs.rwth-aachen.de. Department of Computer Science, RWTH Aachen, Germany
[¶] moritz.stocker@inf.ethz.ch. Department of Computer Science, ETH Zürich, Switzerland

Removal can also be combined with analyzing the advice complexity, leading to a very interesting trade-off between competitive ratio and advice [4].

We will briefly review these different models in terms of competitive analysis and point out the sometimes surprisingly different behavior when compared to the classical online knapsack problem.

References

[1] Susanne Albers, Arindam Khan, and Leon Ladewig. Improved online algorithms for knapsack and GAP in the random order model. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019)*, volume 145, pages 22:1–22:23. Schloss Dagstuhl, 2019.

[2] Hans-Joachim Böckenhauer, Ralf Klasing, Tobias Mömke, Peter Rossmanith, Moritz Stocker, and David Wehner. Online knapsack with removal and recourse. In *Proceedings of the 34th International Workshop on Combinatorial Algorithms (IWOCA 2023)*. To appear.

[3] Hans-Joachim Böckenhauer, Dennis Komm, Richard Královič, and Peter Rossmanith. The online knapsack problem: Advice and randomization. *Theo. Comp. Sci.*, 527:61–72, 2014.

[4] Fabian Frei. *Beneath, Behind, and Beyond Common Complexity Classes*. PhD thesis, ETH Zurich, Zürich, Switzerland, 2021.

[5] Xin Han, Yasushi Kawase, and Kazuhisa Makino. Online unweighted knapsack problem with removal cost. *Algorithmica*, 70(1):76–91, 2014.

[6] Kazuo Iwama and Shiro Taketomi. Removable online knapsack problems. In Peter Widmayer, Francisco Triguero Ruiz, Rafael Morales Bueno, Matthew Hennessy, Stephan J. Eidenbenz, and Ricardo Conejo, editors, *Automata, Languages and Programming, 29th International Colloquium, ICALP 2002, Malaga, Spain, July 8-13, 2002, Proceedings*, volume 2380 of *Lecture Notes in Computer Science*, pages 293–305. Springer, 2002.

[7] Anna R. Karlin and Elias Koutsoupias. Beyond competitive analysis. In *Beyond the Worst-Case Analysis of Algorithms*, pages 529–546. Cambridge University Press, 2021.

Learning-Augmented Online TSP on Rings, Trees, Flowers and (almost) Everywhere Else

Evripidis Bampis [*] Bruno Escoffier [†] Themis Gouleakis [‡]
Niklas Hahn [§] Kostas Lakis [¶] Golnoosh Shahkarami [‖]
Michalis Xefteris [**]

We study the Online Traveling Salesperson Problem (OLTSP) with predictions. In OLTSP, introduced by Ausiello et al. in [1], a sequence of initially unknown requests arrive over time at points (locations) of a metric space. The goal is, starting from a particular point of the metric space (the origin), to serve all these requests while minimizing the total time spent. The server moves with unit speed or is "waiting" (zero speed) at some location. We consider two variants: in the *open* variant, the goal is achieved when the last request is served. In the *closed* one, the server additionally has to return to the origin. The time in which a request is communicated to the server is called its release time. Recent works have proposed a variety of approaches to tackle OLTSP with predictions [4, 6, 7, 5]. More related to our work is the one of Gouleakis et al. [6]. In [6], they studied a learning-augmented framework for OLTSP on the line. They introduced a prediction model in which the predictions correspond to the locations of the requests. They proposed algorithms for both the closed and the open variants that are consistent, smooth and robust. In this work, we adopt the prediction model of [6] using predictions concerning the locations of requests. The prediction error is defined as the normalized sum of distances between predicted and actual locations of the requests. We propose a general oracle-based framework that allows us to design consistent, smooth and robust learning-augmented algorithms for both the closed and open variants of the problem in general metrics. Moreover, we show how to get polynomial/FPT algorithms using this oracle-based framework in specific metrics, namely rings, trees and flowers.

Bampis et al. [2] gave a factorial time algorithm for general metrics with a competitive ratio of 3/2 for the case of perfect predictions (known locations), which is tight. First, we modify this algorithm (still under the assumption of perfect predictions) and introduce our main oracle-based 3/2-competitive framework. The main idea is to consider a suitable subset of permutations of the requests, referred to as *dominating permutations*, given by a so-called *domination oracle* instead of all the permutations. This allows for a reduction of the running time, since the bottleneck is located in the cardinality of the set of considered permutations. This restriction of the permutation set preserves the

[*] evripidis.bampis@lip6.fr. Sorbonne Université, Paris, France
[†] bruno.escoffier@lip6.fr. Sorbonne Université and Institut Universitaire de France, Paris, France
[‡] tgoule@nus.edu.sg. National University of Singapore, Singapore, Singapore
[§] niklas.hahn@lip6.fr. Sorbonne Université, Paris, France
[¶] klakis@student.ethz.ch. ETH Zürich, Zürich, Switzerland
[‖] gshahkar@mpi-inf.mpg.de. MPI für Informatik, Universität des Saarlandes, Saarbrücken, Germany
[**] michail.xefteris@lip6.fr. Sorbonne Université, Paris, France

consistency of 3/2. Using the oracle-based framework with predicted locations we introduce our main algorithm, which does not assume perfect predictions. Our algorithm is 3/2-consistent, smooth, and robust.

The main technical contribution of our work is the implementation of the domination oracles we have referred to. On a high level, we say that a permutation π_{dom} of requests dominates another permutation π at time t, if the following conditions hold. Assuming q is the first unreleased request in π at time t, the distance traveled up to q is not longer in π_{dom}, and also a superset of the requests preceding q in π is visited before q. Moreover, π_{dom} induces a not longer path than π. These two key facts allow us to preserve 3/2-consistency.

For general metrics, we achieve this domination using a very similar idea as the one employed in the definition of the $O(n^2 2^n)$ dynamic programming solution of the classical TSP [3]. That is, for any possible subset of released requests that might have been served by π before q, we simply build two optimal paths for the parts before and after q (without release times) and then we concatenate them to get a dominating permutation. Any permutation is dominated by the one corresponding to the correct guess of requests served up to q. This yields a single-exponential time algorithm overall.

For specific metrics we can reduce the runtime of the algorithm, since we do not really need to try all possible subsets of requests before serving q. Based on a structural result we give an FPT algorithm for trees parameterized by the number ℓ of leaves. Furthermore, we deal with the case of the ring and give a polynomial time algorithm. Finally, we combine the two previous sets of ideas to tackle flowers. Flowers are essentially comprised of a bunch of rings (petals) and a semi-line (stem), all of which are attached to the origin.

References

[1] Giorgio Ausiello, Esteban Feuerstein, Stefano Leonardi, Leen Stougie, and Maurizio Talamo. Algorithms for the on-line travelling salesman. *Algorithmica*, 29:560–581, 2001.

[2] Evripidis Bampis, Bruno Escoffier, Niklas Hahn, and Michalis Xefteris. Online TSP with Known Locations. *To appear in WADS*, 2023. CoRR abs/2210.14722.

[3] Richard Bellman. Dynamic programming treatment of the travelling salesman problem. *J. ACM*, 9(1):61–63, Jan 1962.

[4] Giulia Bernardini, Alexander Lindermayr, Alberto Marchetti-Spaccamela, Nicole Megow, Leen Stougie, and Michelle Sweering. A universal error measure for input predictions applied to online graph problems. In *NeurIPS*, 2022.

[5] Shuchi Chawla and Dimitris Christou. Online time-windows TSP with predictions. *CoRR*, abs/2304.01958, 2023.

[6] Themis Gouleakis, Konstantinos Lakis, and Golnoosh Shahkarami. Learning-Augmented Algorithms for Online TSP on the Line. *To appear in AAAI*, 2023. CoRR abs/2206.00655.

[7] Hsiao-Yu Hu, Hao-Ting Wei, Meng-Hsi Li, Kai-Min Chung, and Chung-Shou Liao. Online TSP with predictions. *CoRR*, abs/2206.15364, 2022.

Speed-Oblivious Online Scheduling: Knowing (Precise) Speeds is not Necessary

Alexander Lindermayr * Nicole Megow * Martin Rapp †

Heterogeneous processors are getting more and more common in various domains. For several years now, efficiency and performance gains in smartphone chips have depended crucially on the combination of high-performance and low-performance (but energy-efficient) cores [1]. These advances show the demand for schedulers that respect *job-dependent heterogeneity*. Formally, the *(processing) speed* s_{ij} of job j on machine i is the amount of processing that j receives when running on i for one time unit. Despite the relevance of values s_{ij} for high-performance scheduling, there is a big discrepancy between how theory and practice handle them: while scheduling theory most commonly assumes that speeds are known to an algorithm, this is typically not the case in practice. Hence, algorithms that perform well in theory are often not applicable in practice.

In this work, we propose new models and algorithms to bridge this gap. In particular, we introduce *speed-oblivious* algorithms, which do not rely on knowing (precise) speeds. Thereby we focus on (non-)clairvoyant scheduling subject to minimizing the total (weighted) completion time. One reason for the common assumption of knowing precise speeds in scheduling theory is that any speed-oblivious algorithm has a competitive ratio of at least $\Omega(m)$ for m machines.

To circumvent this pessimistic lower bound, we propose two (new) models which are motivated by data-driven machine-learned models and modern heterogeneous hardware architectures:

Speed predictions, motivated by a recent line of research, give algorithms access to values $\hat{s}_{ij}$ for every machine i at the release date of every job j. The *distortion error* μ is defined as $\mu = \mu_1 \cdot \mu_2$, where $\mu_1 = \max_{i,j} \left\{ \frac{\hat{s}_{ij}}{s_{ij}} \right\}$ and $\mu_2 = \max_{i,j} \left\{ \frac{s_{ij}}{\hat{s}_{ij}} \right\}$.

Speed-ordered machines assume an order on the machines such that for any two machines i and i' and any job j holds $s_{ij} \geq s_{i'j}$ if and only if $i \leq i'$. Algorithms are aware of this order.

In the following we highlight our results for both models.

Learning-augmented algorithms for speed predictions We provide the first learning-augmented algorithms with job-dependent speed predictions and prove error-dependent performance guarantees w.r.t. the distortion error μ. This gives formal evidence on why algorithms perform well in practice, even if the assumed speeds slightly diverge from the true speeds. We further show that a competitive ratio linear in μ is best possible, even for migratory algorithms and related machines.

*{linderal,nmegow}@uni-bremen.de. Faculty of Mathematics and Computer Science, University of Bremen, Germany.

†martin.rapp@kit.edu.Faculty for Informatics, Karlsruhe Institute of Technology, Germany.

Theorem 1. *For minimizing the total weighted completion time on unrelated machines, there exist speed-oblivious online algorithms with speed predictions that are*

(i) clairvoyant and 8μ*-competitive,*

(ii) clairvoyant, non-preemptive and $7.216\mu^2$*-competitive,*

(iii) non-clairvoyant and 27μ*-competitive.*

For (i), we design a novel and efficient clairvoyant algorithm, which might be of independent interest. It always schedules the subset of jobs that maximizes the total (predicted) density in a feasible job-to-machine assignment, where the density of a job j on machine i is equal to $\frac{w_j s_{ij}}{p_j}$. We show that it is 8-competitive in the speed-aware setting. Interestingly, this algorithm reduces to Smith's rule on a single machine and preemptive variants. [4].

On the technical side, we prove upper bounds on the competitive ratios using the *dual-fitting* technique . For (i), we present a new dual setup, which we believe could be helpful for future dual-fitting approaches. The algorithms and proofs for (ii) and (iii) are are inspired by previous work (Greedy WSPT [2], Proportional Fairness [3]). However, for (iii) we achieve better constants via an optimized analysis, which also improves the speed-aware case.

Algorithms for speed-ordered machines The strong lower bound of $\Omega(m)$ on the competitive ratio for speed-oblivious algorithms for m machines crucially relies on accelerating the machine that an algorithm tries last. This argument becomes infeasible in the speed-ordered setting, because the machines are distinguishable upfront. On the negative side, we show that any constant-competitive algorithm must migrate jobs. This is even true for clairvoyant algorithms and related machines.

Theorem 2. *There is a clairvoyant speed-oblivious online algorithm for minimizing the total weighted completion time on speed-ordered related machines with a competitive ratio of at most* 8*.*

We show that this algorithm is not competitive on unrelated machines. Somewhat surprisingly, our non-clairvoyant algorithm achieves non-trivial competitive ratios for both related and unrelated machines, as the following theorem states.

Theorem 3. *There is a non-clairvoyant speed-oblivious online algorithm for minimizing the total completion time*

(i) on speed-ordered related machines with a competitive ratio of at most 8*, and*

(ii) on m *speed-ordered unrelated machines with a competitive ratio in* $\Theta(\log(m))$*.*

References

[1] ARM Limited. big.LITTLE Technology: The Future of Mobile, 2013.

[2] Varun Gupta, Benjamin Moseley, Marc Uetz, and Qiaomin Xie. Greed works - online algorithms for unrelated machine stochastic scheduling. *Math. Oper. Res.*, 45(2):497–516, 2020.

[3] Sungjin Im, Janardhan Kulkarni, and Kamesh Munagala. Competitive algorithms from competitive equilibria: Non-clairvoyant scheduling under polyhedral constraints. *J. ACM*, 65(1):3:1–3:33, 2018.

[4] Nicole Megow and Andreas S. Schulz. On-line scheduling to minimize average completion time revisited. *Oper. Res. Lett.*, 32(5):485–490, 2004.

Online Dynamic Power Management: from One to Multiple

Ya-Chun Liang [*] Prudence W.H. Wong [†]

In the realm of the Internet of Things (IoT), it is crucial to design operating systems that can reduce power consumption and improve energy efficiency, especially with the increasing growth of portable devices such as smartphones, iPad, etc, in our daily life. A machine consumes energy and frequently transitions between two states: ON and OFF. The process of transitioning from the OFF state to the ON state can sometimes demand a significant amount of energy. For example, a laptop enters a sleep or shutdown (OFF) state after a certain period of inactivity, such as one hour, and resumes to the ON state upon pressing a key. However, this transition is actually energy consuming. Similarly, copy machines turn off automatically if idling for, e.g., ten minutes. Subsequently, when a new job arrives, it requires a large amount of energy to warm them up again. There are many similar scenarios that highlight the importance of developing efficient and effective algorithms to reduce energy usage when it comes to managing battery power.

More explicitly, the realistic problem can be depicted as an *online* model of the *dynamic power management (DPM)* problem, discussed by Irani et al. [5, 6] and Chen et al. [2], focusing on determining the optimal timings for transitioning a machine between different states to complete input jobs while minimizing energy consumption. Irani et al. [5, 6] initiated the study of combining the above power-down mechanism with *dynamic speed scaling*, where the latter technique has been widely explored in the past decades [1, 3, 4, 6, 8]. The feature of dynamic speed scaling is also referred to as *dynamic voltage frequency scaling (DVFS)*, which is a commonly used power management technique. It allows the speed of a machine to be adjusted dynamically, and the rate of energy consumption is typically described by a convex function of its processing speed. Irani et al. [6] proposed the first online algorithm with a constant competitive ratio of $\max\{c_1c_2+c_1+2, 4\}$, where c_1 and c_2 are some constant parameters in a given convex power function. Chen et al. [2] proposed a model without using dynamic speed scaling, where an online algorithm can utilize two machines M_1 and M_2 instead. The assumption is that all input jobs must be completed by the offline optimal scheduler using a single machine, which is known as the "single machine schedulability" condition. This condition is characterized by the earliest-deadline-first (EDF) principle, where the job with the earliest deadline is always selected for execution. Chen et al. provided an upper bound of 4 and a lower bound of 2.06 for the competitive ratio of this dual-machine problem. Very recently, Liang et al [7] improved both upper and lower bounds to 3 and 2.1, respectively, for the

[*] ya-chun.liang@liverpool.ac.uk.
Department of Industrial Engineering and Engineering Management, National Tsing Hua University, Hsinchu 30013, Taiwan. Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool L69 3BX, United Kingdom

[†] pwong@liverpool.ac.uk.
Department of Computer Science, University of Liverpool, Liverpool L69 3BX, United Kingdom

same dual-machine model in [2], narrowing the competitive ratio gap between lower and upper bounds from 1.94 [2] to 0.9 [7]. Their new turn-on technique attempts to delay jobs as far as possible but turn on a machine earlier than its due time by a margin. On the other hand, their turn-off strategy sets the idle time to twice the break even time, rather than the popular setting used for the well-known Ski Rental problem.

In this talk, we extend the study in [2, 7] to a more general model in which one can have a $(k+1)$ schedulability for the online scheduler while the offline adversary uses at most k machines based on the similar techniques used in [7]. We obtain the same ratios and thus generalize this problem from the original dual-machine model to a more general k-machine model.

References

[1] Nikhil Bansal, Tracy Kimbrel, and Kirk Pruhs. Speed scaling to manage energy and temperature. *Journal of the ACM (JACM)*, 54(1):3, 2007.

[2] Jian-Jia Chen, Mong-Jen Kao, DT Lee, Ignaz Rutter, and Dorothea Wagner. Online dynamic power management with hard real-time guarantees. *Theoretical Computer Science*, 595:46–64, 2015.

[3] Jian-Jia Chen and Tei-Wei Kuo. Procrastination for leakage-aware rate-monotonic scheduling on a dynamic voltage scaling processor. *ACM SIGPLAN Notices*, 41(7):153–162, 2006.

[4] Jian-Jia Chen and Tei-Wei Kuo. Procrastination determination for periodic real-time tasks in leakage-aware dynamic voltage scaling systems. In *Computer-Aided Design, 2007. ICCAD 2007. IEEE/ACM International Conference on*, pages 289–294. IEEE, 2007.

[5] Sandy Irani, Sandeep Shukla, and Rajesh Gupta. Online strategies for dynamic power management in systems with multiple power-saving states. *ACM Transactions on Embedded Computing Systems (TECS)*, 2(3):325–346, 2003.

[6] Sandy Irani, Sandeep Shukla, and Rajesh Gupta. Algorithms for power savings. *ACM Transactions on Algorithms (TALG)*, 3(4):41–es, 2007.

[7] Ya-Chun Liang, Kazuo Iwama, and Chung-Shou Liao. Improving the bounds of the online dynamic power management problem. In *33rd International Symposium on Algorithms and Computation (ISAAC 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

[8] Frances Yao, Alan Demers, and Scott Shenker. A scheduling model for reduced cpu energy. In *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*, pages 374–382. IEEE, 1995.