Scheduling with a processing time oracle

Fanny Dufossé[1], Christoph Dürr[2], Noël Nadal[3], Denis Trystram[1], Óscar C. Vásquez[4]

[1] Univ. Grenoble-Alpes, Inria, CNRS, LIG, France.
Fanny.Duffose@inria.fr, Denis.Trystram@univ-grenoble-alpes.fr

[2] Sorbonne University, CNRS, LIP6, Paris, France. Christoph.Durr@LIP6.fr

[3] Ecole Normale Supérieure Paris-Saclay, France. Noel.Nadal@ens-cachan.fr

[4] Industrial Engineering Department, Universidad de Santiago de Chile, Chile.
Oscar.Vasquez@usach.cl

To illustrate the problem studied in this work, suppose that you need to schedule on a single machine four jobs called A, B, C, D. Job B has processing time 5, while jobs A, C, D have processing times 0.3, see Figure 1. The cost of a schedule (also called *objective value*) is the total completion time of the jobs. Hence one possible optimal execution order is A, C, D, B, which has objective value 7.7. However the processing times are initially unknown to you, the jobs are indistinguishable, and you only know that the processing times are either 0.3 or 5. So you could schedule them in an arbitrary order, for example the execution order A, B, C, D would yield an objective value of 17.1, which is roughly 2.22 larger than the optimum. Luckily you have access to a processing time oracle, which allows you to query (or *test*) the processing time of a particular job, and this operation takes 1 time unit. Such an oracle can be thought as a machine learning black box predictor, which was trained on large number of jobs. You could query this oracle for all 4 jobs, providing full information, which allows you to schedule the jobs in the optimal order. However you don't have to wait until you know the processing time of all jobs before you start executing them. For example if a query reveals a short job, then it could be scheduled immediately, and if a query reveals a long job, then its execution could be postponed towards the end of the schedule. This means that there is no benefit to query the last job, as the position of its execution would be the same, regardless of the outcome. In summary, for this example you have the possibility to query between 0 and 3 jobs. The resulting schedules are depicted in Figure 1. Testing the first 2 jobs would lead to the smallest objective value of 14.7, which is only roughly 1.91 larger than the optimum.

Formally, we study a single machine scheduling problem with the objective of minimizing the sum of completion times. Each of the given jobs is either short or long. However the processing

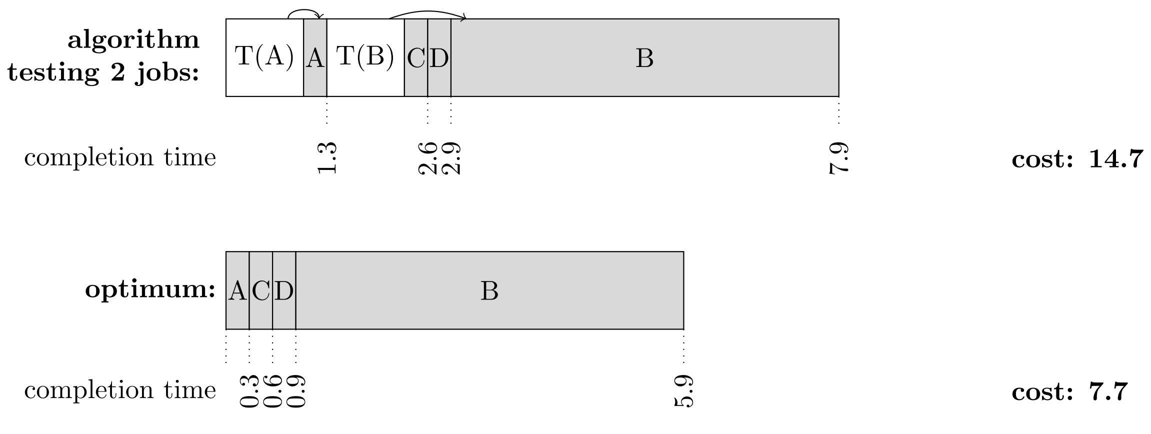

FIG. 1 – Some schedules with four jobs A, B, C, D and short processing times either 0.3 or 5. White boxes represent job tests, while gray boxes represent job executions. The cost is the sum of the completion times.

times are initially hidden to the algorithm, but can be tested. This is done by executing a processing time oracle, which reveals the processing time of a given job. Each test occupies a time unit in the schedule, therefore the algorithm must decide for which jobs it will call the processing time oracle. The objective value of the resulting schedule is compared with the objective value of an optimal schedule, which is computed using full information. The resulting competitive ratio measures the price of hidden processing times, and the goal is to design an algorithm with minimal competitive ratio.

This model falls into the paradigm of *optimizing under explorable uncertainty*, which has been introduced in 1991 [4], and started to be applied to scheduling problems in 2016 [5]. In this approach, a problem instance consists of a set of numerical parameters, the algorithm obtains as input only an uncertainty interval for each one. The algorithm knows for each parameter that it belongs to the given interval and has the possibility to make a query in order to obtain the precise value. Clearly a compromise has to be found between the number of queries an algorithm makes and the quality of the solution it produces. This setting differs from a probabilistic one, studied in [5, 6, 7], where jobs have weights and processing times drawn from known distributions, and the algorithm can query these parameters. A seemingly similar problem has been studied in [2, 3, 1], where the term *testing* has a different meaning than in this paper.

Two models are studied in this paper. In the *non-adaptive* model, the algorithm needs to decide beforehand which jobs to test, and which jobs to execute untested. However in the *adaptive* model, the algorithm can make these decisions adaptively depending on the outcomes of the job tests. In both models we provide optimal polynomial time *two-phase* algorithms, which consist of a first phase where jobs are tested, and a second phase where jobs are executed obliviously. Experiments give strong evidence that optimal algorithms have this structure. Proving this property is left as an open problem.

Mots-clés : *scheduling ; uncertainty ; competitive ratio ; processing time oracle*

Références

[1] Susanne Albers and Alexander Eckl. Explorable Uncertainty in Scheduling with Non-Uniform Testing Times. In *Proc. of the 18th Workshop on Approximation and Online Algorithms*, September 2020.

[2] Christoph Dürr, Thomas Erlebach, Nicole Megow, and Julie Meißner. Scheduling with Explorable Uncertainty. In *The 9th Innovations in Theoretical Computer Science Conference (ITCS)*, 2018.

[3] Christoph Dürr, Thomas Erlebach, Nicole Megow, and Julie Meißner. An Adversarial Model for Scheduling with Testing. *Algorithmica*, pages 3630–3675, 2020.

[4] Simon Kahan. A model for data in motion. In *Proceedings of the twenty-third annual ACM symposium on Theory of computing - STOC '91*, pages 265–277, New Orleans, Louisiana, United States, 1991. ACM Press.

[5] Retsev Levi. Scheduling (Talk at Dagstuhl Seminar 16081). In *Dagstuhl Reports*, volume 6. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016. Issue : 2.

[6] Retsev Levi, Thomas Magnanti, and Yaron Shaposhnik. Scheduling with Testing. *Management Science*, 65(2), 2018.

[7] Yaron Shaposhnik. *Exploration vs. exploitation : reducing uncertainty in operational problems.* PhD Thesis, Massachusetts Institute of Technology, 2016.