Machine scheduling and re-optimization
(extended abstract)

Nicolas Boria  Federico Della Croce  Fabio Salassa
D.A.I., Politecnico di Torino
{nicolas.boria,federico.della.croce,fabio.salassa}@polito.it

We present approximation algorithms for various machine scheduling problems in the re-optimization setting, which can be described as follows: considering an instance $I$ of a given problem $P$ for which an optimal solution $OPT$ is provided, and an instance $I'$ which results from a local perturbation of $I$, can the information provided by $OPT$ be used to solve $I'$ in a more efficient way (i.e., with a lower complexity and/or with a better approximation ratio) than if this information wasn’t available?

Although the first results regarding re-optimization were provided on a parallel machines scheduling problem with forbidden sets in [2], since then, the relevant literature was mainly devoted to graph problems rather than attacking known paradigmatic $NP$–$Hard$ machine scheduling problems.

This is however somehow in contrast with the important literature available on reactive scheduling (see [3]), that is what refers to the schedule modifications that may have to be made during project execution starting from a baseline schedule. In some cases, the reactive scheduling effort may rely on very simple techniques aimed at a quick schedule consistency restoration. To this purpose, by considering an optimal schedule as baseline schedule and some jobs additions or deletions as schedule disruptions, we can see that scheduling re-optimization may be considered as strictly linked to reactive scheduling, particularly with simple reoptimization strategies so that the baseline schedule is only mildly modified.

Here, we consider the re-optimization versions of various strongly $NP$–$Hard$ machine scheduling problems, under modification of the job set (insertion or deletion of jobs). We analyze the approximation ratios of simple re-optimization strategies that keep unchanged the baseline schedule, namely that merely insert the new job (jobs) into the initial optimum under job insertion, or remove the deleted job (jobs) from the initial optimum under job deletion. We denote by $k_{i,j}$ ($1 \leq i \leq j \leq n+1$) the reoptimization strategy that consists of computing $j - i + 1$ candidate solutions, by inserting the new job in the initial optimum in the $k_{i,j}$ position for all possible $k$'s between $i$ and $j$, while leaving the rest of the scheduling unchanged. Yet, despite the extreme simplicity of such classes of algorithms, we show that the proposed strategies ensure constant approximation ratios, and also provide tight lower bounds for some of them.

As a simple example, consider the $F3 \parallel C_{\max}$ problem (according to the three-field notation of [1]) that can be stated as follows. A set of $n$ jobs is available at time 0 to be processed on 3 machines where each job is processed on machine $M_1$ first, then on machine $M_2$ and finally on machine $M_3$. We denote by $p_{kj}$ the processing time of job $j$ on machine $k$ ($j = 1, \ldots, n$, $k = 1, \ldots, 3$). Also, we denote by $C_{kj}$ the completion time of job $j$ on machine $M_k$. Preemption on either machine is not allowed. The objective is the minimization of the maximum completion time on the third machine, that is the makespan. The solution is required to be a permutation schedule, that is the jobs must share the same sequence on all machines. This problem is known to be $NP$-hard in the strong sense. We denote by $F3 \parallel C_{\max}$ the re-optimization of version
of $F3 \parallel C_{\text{max}}$ where a single job $x$ with processing times $p_{1x}$, $p_{2x}$, and $p_{3x}$ is added to an instance already solved to optimality. Without loss of generality, the initial optimum is given by a schedule $\text{OPT} = [1, \ldots, n]$ and has value $f(\text{OPT}) = C_{3n}$. On the other hand, the unknown optimal sequence on the modified instance is denoted by $\text{OPT}'$ and its value is denoted by $f(\text{OPT}')$.

The following proposition holds.

**Proposition 1.** $F3 \parallel C_{\text{max}}^+$ is approximable within ratio 2 by algorithm $\text{A}_{n+1,n+1}$ and this bound is asymptotically tight both for that algorithm but also for algorithm $\text{A}_{1,n+1}$.

**Proof.** To prove the upper bound on the approximation ratio, notice that, as job $x$ is added to the original jobset and all processing times are $\geq 0$, clearly $f(\text{OPT}') \geq f(\text{OPT})$. Also, $f(\text{OPT}') \geq p_{1x} + p_{2x} + p_{3x}$ obviously holds. Besides, in sequence $[1, \ldots, n, x]$ provided by algorithm $\text{A}_{n+1,n+1}$, we have $f([1, \ldots, n, x]) = C_{3x} \leq f(\text{opt}) + p_{1x} + p_{2x} + p_{3x}$ as from time $t = f(\text{OPT})$ all machines are available for processing and hence job $x$ starts processing on machine $M_1$ not later than $t$. But then $f([1, \ldots, n, x]) \leq f(\text{opt}) + p_{1x} + p_{2x} + p_{3x} \leq 2f(\text{OPT}')$. For the asymptotic tightness, consider the following original instance with $n = 2m$ jobs where job $2i - 1$ ($i = 1, \ldots, m$) has processing times $p_{1i} = 0$, $p_{2i} = 0$ and $p_{3i} = 1$, while job $2j$ ($j = 1, \ldots, m$) has processing times $p_{1j} = 1$, $p_{2j} = 0$ and $p_{3j} = 0$. Finally, for the new instance, consider adding job $x$ with processing times $p_{1x} = 0$, $p_{2x} = n$ and $p_{3x} = 0$. Sequence $[1, \ldots, 2m]$ is optimal (actually one of the optimal sequences) for the original problem with value $n$, while all sequences provided by algorithm $\text{A}_{1,n+1}$ on the new problem have either value $2n$ or value $2n - 1$. On the other hand, the optimal sequence for the new problem is $[1, 3, \ldots, 2m - 1, x, 2, 4, \ldots, 2m]$. Correspondingly, the approximation ratio is $\frac{2n-1}{n}$ that asymptotically tends to 2 for $n \to \infty$. \quad \blacksquare

In all, similarly to the above result, we provide in this work approximation results of re-optimization strategies such as algorithm $\text{A}_{i,j}$ for various paradigmatic machine scheduling problems with different regular performance measures and different machines settings (single machine, parallel machines, flow shop, job shop) under job insertion. These paradigmatic machine scheduling problems are then considered in the re-optimization framework also under job deletion, where we analyze the performance guaranteed by the basic strategy of leaving the remaining schedule as is, that is, not switching any pair of jobs with respect to the initial optimal solution.

**References**

