Optimal Scheduling of Multiclass Parallel Machines

Jay Sethuraman[*] Mark S. Squillante[†]

1 Introduction

Motivated by our study of a general stochastic scheduling optimization problem,[1] we examine various static instances where the stochastic problems reduce to the corresponding deterministic scheduling problems without loss of generality. Following our solution approach for these special cases leads us to rederive many known results in a fairly elegant manner, yielding simple approximation algorithms whose guarantees are shown to match the best known results. We believe that improvements in the approximation guarantees for some of these special cases will be possible by exploiting the convex programming techniques of our approach.

We formulate the nonlinear scheduling optimization problem, we provide a convex relaxation, and then we present a simple (randomized) rounding scheme. The interested reader is referred to our technical report[1] for additional details, including proofs and references.

2 Scheduling Problems

Consider a system with n jobs and m machines. We use i to index the jobs and j to index the machines, under the constraints $i = 1,\ldots,n$ and $j = 1,\ldots,m$ unless noted *otw.* The processing requirement of job i on machine j is p_{ij}; define $\mu_{ij} \equiv p_{ij}{}^{-1}$. Our formulation is general enough to handle machines that are *identical*, *uniform* and *unrelated*; we will primarily work in the setting of *unrelated* machines — the most general case.

The allocation question reduces to finding an assignment of jobs to machines that minimizes a weighted sum of job completion times, where $c_i \in \mathbb{N}$ and T_i denote the weight and completion time for job i. Let x_{ij} be an indicator variable such that $x_{ij} = 1$ if job i is assigned to machine j, and $x_{ij} = 0$ *otw.* Suppose for the moment that we know the optimal assignment of jobs to machines — this enables us to reduce the m machine scheduling problem to m independent single machine scheduling problems. For the objective under consideration, the single machine scheduling problem is solved by the $c\mu$ rule: at machine j, all the jobs assigned to it are scheduled in such a way that job k is scheduled before job ℓ *iff* $c_k\mu_{kj} > c_\ell\mu_{\ell j}$. Motivated by this, we assume that each machine orders all the n jobs in such a way that job k appears before job ℓ in the machine j ordering, denoted by $k <_j \ell$, if $c_k\mu_{kj} \geq c_\ell\mu_{\ell j}$.

We know the optimal completion time of a job given an assignment of jobs to machines. Thus, we can formulate our problem as that of finding the allocation vector which minimizes weighted completion time, subject to each job being allocated to a single machine.

Observe that an application of the $c\mu$ rule yields $T_i = \sum_{j=1}^{m} x_{ij}(p_{ij} + \sum_{k<_j i} x_{kj}p_{kj})$, and thus the optimal solution to our nonlinear integer program does yield an optimal solution to the scheduling problem we started with. This formulation, however, has two potentially complicating factors: (i) the integrality restrictions on the assignment variables, and (ii) the nonlinearity of the objective function. It is quite easy to see that (i) is not a serious problem: a straightforward (randomized) rounding scheme can be used to prove the integrality of the relaxation in which $0 < x_{ij} < 1$ (a proof of this is embedded in the proof of Theorem 4.1 in §4). Hence, the nonlinear optimization problem given by:

$$\text{(NLPR)}\quad \min \quad \sum_{i=1}^{n}\sum_{j=1}^{m} c_i x_{ij}\big(p_{ij} + \sum_{k<_j i} x_{kj}p_{kj}\big)$$

$$\text{s.t.} \quad \sum_{j=1}^{m} x_{ij} = 1, \quad x_{ij} \geq 0,$$

is an exact nonlinear formulation of our scheduling problem in the sense that one can always find an integral optimal solution to this nonlinear programming problem. Although this is an interesting property, it does not simplify the problem because our relaxation is a nonlinear programming problem and thus is difficult to solve. In §3 we will provide a convex relaxation which can be used to design approximation algorithms — of course, the known hardness results of this problem suggest that our convex relaxation is truly a relaxation and does not always provide the optimal solution to the scheduling problems considered.

3 Convex Relaxations

To clarify the exposition, we fix a machine j and assume that our n jobs are labeled such that $1 <_j 2 <_j \ldots <_j n$. Under these assumptions, the contribution F_j of machine j to the total cost can be written as $F_j = \sum_{i=1}^{n} c_i x_{ij}(p_{ij} + \sum_{k<i} x_{kj}p_{kj})$.

[*]Massachusetts Institute of Technology. jayc@mit.edu
[†]IBM T. J. Watson Research Center. mss@watson.ibm.com

When the x_{ij} are restricted to be either 0 or 1, we have $x_{ij}^2 = x_{ij}$. Simple algebraic manipulations yield

$$\sum_{k=1}^{n}\sum_{\ell=1}^{k-1} c_k x_{kj} p_{\ell j} x_{\ell j} = \sum_{k=1}^{n}\sum_{\ell=k+1}^{n} c_\ell x_{\ell j} p_{kj} x_{kj}. \tag{3.1}$$

We then can write the following two expressions for F_j:

$$F_j = \sum_{k=1}^{n} c_k p_{kj} x_{kj} + \sum_{k=1}^{n}\sum_{\ell=1}^{k-1} c_k x_{kj} p_{\ell j} x_{\ell j}, \tag{3.2}$$

$$= \sum_{k=1}^{n} c_k p_{kj} x_{kj}^2 + \sum_{k=1}^{n}\sum_{\ell=k+1}^{n} c_\ell x_{\ell j} p_{kj} x_{kj}. \tag{3.3}$$

Adding equations (3.2) and (3.3), we have

$$F_j = \frac{1}{2}\Big(\sum_{k=1}^{n} c_k p_{kj} x_{kj} + \sum_{k=1}^{n}\sum_{\ell=k+1}^{n} c_\ell p_{kj} x_{kj} x_{\ell j} + \sum_{k=1}^{n}\sum_{\ell=1}^{k-1} c_k p_{\ell j} x_{kj} x_{\ell j} + \sum_{k=1}^{n} c_k p_{kj} x_{kj}^2\Big). \tag{3.4}$$

To see why F_j given by equation (3.4) is convex, we first find the Hessian of F_j, $\mathbf{H_j}$, which is an $n \times n$ matrix with its $(k,l)^{\text{th}}$ entry given by: $(\mathbf{H_j})_{k\ell} = c_\ell p_{kj}$, if $\ell \geq k$; $(\mathbf{H_j})_{k\ell} = c_k p_{\ell j}$, if $\ell \leq k$. To prove that F_j is convex, it suffices to show that $\mathbf{H_j}$ is a positive semidefinite matrix. This is immediate from the ordering of the jobs; recall that all jobs were ordered so that $c_1\mu_{1j} \geq c_2\mu_{2j} \geq \ldots \geq c_n\mu_{nj}$, which readily yields a proof of the semidefiniteness of $\mathbf{H_j}$. In fact, if there are no ties, the Hessian is positive definite.

This simple change of using x_{ij}^2 instead of x_{ij} at appropriate places has helped us convert a nonconvex function to a convex one. While this change does not affect the integer program, it results in a weaker relaxation, which is convex. The convex relaxation of the parallel machine scheduling problem is then:

$$\begin{aligned}
\text{(UPM)}\quad \min \quad & \sum_{j=1}^{m} F_j \\
\text{s.t.} \quad & F_j \geq \frac{1}{2}\Big(\sum_{k=1}^{n} c_k p_{kj}(x_{kj} + x_{kj}^2) + 2\sum_{k=1}^{n}\sum_{l=k+1}^{n} c_l p_{kj} x_{kj} x_{lj}\Big), \\
& F_j \geq \sum_{i=1}^{n} c_i p_{ij} x_{ij}, \\
& \sum_{j=1}^{m} x_{ij} = 1, \quad x_{ij} \geq 0.
\end{aligned}$$

All of the constraints in the convex relaxation are straightforward and follow from our discussion. We have added the second constraint to the relaxation because it is a linear constraint that certainly yields a lower bound on the optimal value for the integer problem: this constraint does not make a difference to the quality of the relaxation in the case of identical or uniform machines, but strengthens the relaxation for the case of unrelated machines.

4 Rounding

The convex relaxation proposed in §3 can be solved efficiently using standard techniques. Given such a solution, we consider the following straightforward RR algorithm. *Step 1*: Solve the convex programming problem (UPM), and obtain the optimum routing matrix X^*. *Step 2*: Route job i to machine j with probability x_{ij}^*.

THEOREM 4.1. *Algorithm RR produces a schedule whose cost is within a factor of $\frac{3}{2}$ of the optimal cost for all three versions of the parallel machine scheduling problem.*[1]

Although our relaxation (UPM) is written as a general convex programming problem, we can rewrite it as a semidefinite programming problem using standard techniques. Moreover, we have described our rounding scheme assuming that we can find an optimal solution to the convex relaxation. In practice, however, we can only find an ϵ-approximate solution. The same rounding scheme can be used with an ϵ-approximate solution while retaining the same performance guarantee. Finally, our rounding scheme can be derandomized using the method of conditional probabilities. This derandomized rounding algorithm can be coupled with the ϵ-approximate solution to actually find an integer solution with value no worse than 3/2 times the optimum. This is an ϵ-improvement over the previously best known algorithm due to Schulz and Skutella.

For the special case of *identical machines*, the guarantee we can prove is still only 3/2. However, just as Schulz and Skutella point out, the derandomized version of algorithm RR is exactly the algorithm due to Kawaguchi and Kyan, who prove a guarantee of $\frac{\sqrt{2}+1}{2}$ using complicated arguments. Interestingly, for this special case, an optimal solution to the simpler convex relaxation (IPM), formulated exactly as (UPM) without the second constraint, can be solved in closed form where $\hat{x}_{ij} = \frac{1}{m}$ is an optimal solution. Moreover, we can still prove that algorithm RR applied to an optimal solution of (IPM) will yield an integer solution which is no worse than 3/2 times the optimum.

[1] J. Sethuraman, M. S. Squillante. Optimal scheduling in multiclass multiserver systems. Tech. Rep., IBM Res. Div., 1998.