# Hardness of Approximating Flow and Job Shop Scheduling Problems

MONALDO MASTROLILLI, IDSIA, Lugano, Switzerland
OLA SVENSSON, EPFL, Switzerland

20

We consider several variants of the job shop problem that is a fundamental and classical problem in scheduling. The currently best approximation algorithms have worse than logarithmic performance guarantee, but the only previously known inapproximability result says that it is NP-hard to approximate job shops within a factor less than 5/4. Closing this big approximability gap is a well-known and long-standing open problem.

This article closes many gaps in our understanding of the hardness of this problem and answers several open questions in the literature. It is shown the first nonconstant inapproximability result that almost matches the best-known approximation algorithm for acyclic job shops. The same bounds hold for the general version of flow shops, where jobs are not required to be processed on each machine. Similar inapproximability results are obtained when the objective is to minimize the sum of completion times. It is also shown that the problem with two machines and the preemptive variant with three machines have no PTAS.

Categories and Subject Descriptors: F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Sequencing and scheduling*

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Job shop scheduling, hardness of approximation, machine scheduling

**ACM Reference Format:**
Mastrolilli, M. and Svensson, O. 2011. Hardness of approximating flow and job shop scheduling problems. J. ACM 58, 5, Article 20 (October 2011), 32 pages.
DOI = 10.1145/2027216.2027218 http://doi.acm.org/10.1145/2027216.2027218

## 1. INTRODUCTION

We consider the classical job shop scheduling problem together with the more restricted flow shop problem. In the *job shop problem* we have a set of $n$ jobs that must be processed on a given set $M$ of $m$ machines. Each job $j$ consists of a chain of $\mu_j$ operations $O_{1j}, O_{2j}, \ldots, O_{\mu_j j}$. Operation $O_{ij}$ must be processed during $p_{ij}$ time units without interruption on machine $m_{ij} \in M$. A feasible schedule is one in which each operation is scheduled only after all its preceding operations have been completed, and each machine processes at most one operation at a time. For any given schedule, let $C_j$ be the completion time of the last operation of job $j$. We consider the natural

This research is supported by Swiss National Science Foundation project N. 200020-122110/1 "Approximation Algorithms for Machine Scheduling through Theory and Experiments III" and by Hasler Foundation Grant 11099. O. Svensson was also supported by ERC Advanced Investigator Grants 226203 and 228021.
Different parts of this work have appeared in preliminary form in *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 583–592, and in *Proceedings of the 36th International Colloquium on Automata, Languages and Programming (ICALP)*, pp. 677–688.
Authors' addresses: M. Mastrolilli, IDSIA - Istituto Dalle Molle di Studi sull'Intelligenza Artificiale, Galleria 2 CH-6928 Manno, Switzerland; email: monaldo@idsia.ch; O. Svensson, EPFL-IC, Building BC (BC128), Station 14, CH-1015 Lausanne, Switzerland; email: ola.svensson@epfl.ch.

DOI 10.1145/2027216.2027218 http://doi.acm.org/10.1145/2027216.2027218

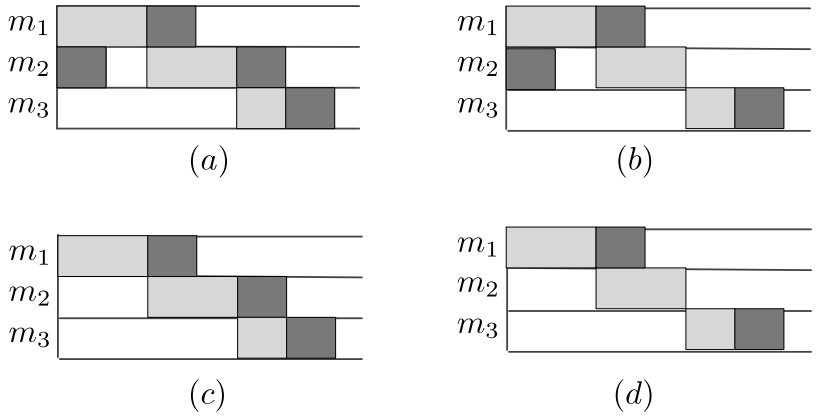


Fig. 1. Example of scheduling instances with three machines and two jobs depicted in light and dark gray. (a) Job shop instance – jobs may have several operations on each machine. (b) Acyclic job shop instance – a job has at most one operation per machine. (c) Flow shop instance – each job has exactly one operation per machine and all the jobs are processed on the machines in the same order. (d) Generalized flow shop instance – jobs have at most one operation per machine and jobs are processed on the machines in the same order.

and typically considered objectives of minimizing the *makespan* $C_{\max} = \max_j C_j$, and minimizing the *sum of weighted completion times* $\sum w_j C_j$. The goal is to find a feasible schedule which minimizes the considered objective function. In the notation of Graham et al. [1979], this problem is denoted as $\mathcal{J}||\gamma$, where $\gamma$ denotes the objective function to be minimized.

In the *flow shop scheduling problem* ($F||\gamma$), each job has exactly one operation per machine, and all jobs go through all the machines in the same order. A natural generalization of the flow shop problem is to not require jobs to be processed on all machines, that is, a job still requests the machines in compliance with some fixed order but may not require to be processed on some of them. We will refer to this more general version as *generalized flow shops* or *flow shops with jumps* ($F|jumps|\gamma$). Generalized flow shop scheduling (and thus flow shop) is a special case of acyclic job shop scheduling, which only requires that within each job all operations are performed on different machines, which in turn is a special case of job shop scheduling. See Figure 1 for example instances of the addressed problems.

### 1.1. Literature Review

Job and flow shops have long been identified as having a number of important practical applications and have been widely studied since the late 50's (the reader is referred to the survey papers of Lawler et al. [1993] and of Chen et al. [1998]). To find a schedule that minimizes the makespan, or one that minimizes the sum of completion times, was proved to be strongly NP-hard for flow shops (and thus job shops) in the 70's, even for severely restricted instances (see, e.g., Chen et al. [1998]).

From then, many approximation methods have been proposed.[1] Since the quality of an approximation algorithm is measured by comparing the returned solution value with a polynomial time computable lower bound on the optimal value, the latter is very important. For a given instance, let $C^*_{max}$ denote the minimum makespan taken over

[1]A $\rho$-approximation algorithm is a polynomial time algorithm that is guaranteed to return a solution that is within $\rho$ times the optimal value.

all possible feasible schedules. If $D$ denotes the length of the longest job (the *dilation*), and $C$ denotes the time units requested by all jobs on the most loaded machine (the *congestion*), then lb = max[$C$, $D$] is a known trivial lower bound on $C^*_{max}$. There is no known significantly stronger lower bound on $C^*_{max}$, and all the proposed approximation algorithms for flow shops, acyclic job shops, job shops, and the more constrained case of permutation flow shops have been analyzed with respect to this lower bound (see, e.g., Feige and Scheideler [2002], Goldberg et al. [2001], Leighton et al. [1994, 1999], Nagarajan and Sviridenko [2008], and Shmoys et al. [1994]).

Even though the trivial lower bound might seem weak, a surprising result by Leighton et al. [1994] says that for acyclic job shops, if all operations are of unit length, then $C^*_{max} = \Theta(\text{lb})$. If we allow operations of any length, then Feige and Scheideler [2002] showed that $C^*_{max} = O(\text{lb} \cdot \log \text{lb} \log \log \text{lb})$ for acyclic job shops. They also showed their analysis to be nearly tight by providing acyclic job shop instances with $C^*_{max} = \Omega(\text{lb} \cdot \log \text{lb}/ \log \log \text{lb})$. The proofs of the upper bounds in Feige and Scheideler [2002], and Leighton et al. [1994] are nonconstructive and make repeated use of (a general version) of the Lovàsz local lemma. Algorithmic versions of the Lovàsz local lemma [Beck 1991] and the general version [Czumaj and Scheideler 2000], have later been used to obtain constructive upper bounds yielding a constant approximation algorithm for unit time acyclic job shops [Leighton et al. 1999] and an $O(\log \text{lb}^{1+\epsilon})$-approximation algorithm for acyclic job shops [Czumaj and Scheideler 2000], where $\epsilon > 0$ is any constant.

Feige and Scheideler's [2002] upper bound for acyclic job shops is also the best upper bound for the special case of flow shops. As flow shops have more structure than acyclic job shops and no flow shop instances with $C^*_{max} = \omega(\text{lb})$ were known, one could hope for a significantly better upper bound for flow shops. The existence of such a bound was raised as an open question in Feige and Scheideler [2002]. The strength of the lower bound lb is better understood for the related *permutation* flow shop problem, that is a flow shop problem with the additional constraint that each machine processes all the jobs in the same order. Potts et al. [1991] gave a family of permutation flow shop instances with $C^*_{max} = \Omega(\text{lb} \cdot \sqrt{\min[m, n]})$. This lower bound was recently shown to be tight, by Nagarajan and Sviridenko [2008], who gave an approximation algorithm that returns a permutation schedule with makespan $O(\text{lb} \cdot \sqrt{\min[m, n]})$.

The best approximation algorithms known for general $J||C_{max}$ are an approximation algorithm by Shmoys et al. [1994] with performance guarantee $O((\log \text{lb})^2 / \log \log \text{lb})$; later improved by a $\log \log \text{lb}$ factor by Goldberg et al. [2001].

When preemption is allowed, every operation can be temporarily interrupted and resumed later without any penalty. For any $\varepsilon > 0$, it is well known that with only $\varepsilon$ loss in the approximation factor, the preemptive job shop scheduling problem is equivalent to the nonpreemptive job shop scheduling problem with unit processing times (see, e.g., Bansal et al. [2006]). For acyclic job shop and flow shop scheduling with preemption, the best-known result is due to Feige and Scheideler [2002] who showed that there always exists a preemptive schedule within a $O(\log \log \text{lb})$ factor of lb. For the general preemptive job shop problem, Bansal et al. [2006] showed an $O(\log m / \log \log m)$-randomized approximation algorithm, and a $(2 + \varepsilon)$-approximation for a constant number of machines. Bansal et al. [2006] also consider the preemptive two-machine job shop problem (denoted as $J2|pmtn|C_{max}$) and present an approximation algorithm with ratio of about 1.442 which improves the previous ratio of 1.5 [Anderson et al. 2001; Sevastianov and Woeginger 1998].

Whether these algorithms for $J||C_{\max}$ and $F||C_{\max}$ are tight or even nearly tight is a long-standing open problem (see "Open problems 6 and 7" in Schuurman and Woeginger [1999]). The only known inapproximability result is due to Williamson

et al. [1997], and states that when the number of machines and jobs are part of the input, it is NP-hard to approximate $F||C_{\max}$ with unit time operations and at most three operations per job, within a ratio better than $5/4$.

The situation is similar for the weighted sum of completions times objective. Queyranne and Sviridenko [2002] showed that an approximation algorithm for the above mentioned problems that produces a schedule with makespan a factor $O(\rho)$ away from the lower bound lb can be used to obtain an $O(\rho)$-approximation algorithms for other objectives, including the sum of weighted completion times. As a result, the previously mentioned approximation guarantees for the makespan criteria also hold for the sum of weighted completion times objective. The only known inapproximability result is by Hoogeveen et al. [2001], who showed that $F||\sum C_j$ is NP-hard to approximate within a ratio better than $1+\epsilon$ for some small $\epsilon > 0$.

Another open problem [Schuurman and Woeginger 1999] is to understand whether there is a PTAS for the general job shop problem with a constant number of machines. For those instances where the number of machines and the number of operations per job are constant, Jansen et al. [2003] gave a PTAS for the makespan criteria (see also Fishkin et al. [2008]). A similar result was obtained by Fishkin et al. [2003] for the weighted sum of completion times objective.

### 1.2. Results and Overview of Techniques

The main result of the this article gives an answer to "Open Problem 7" by Schuurman and Woeginger [1999]) by showing that it is NP-hard to approximate the generalized flow shop problem (and therefore job shops) within any constant factor. Under a stronger but widely believed assumption, the provided hardness result matches the best-known approximation algorithm for acyclic job shops [Feige and Scheideler 2002]. Similar inapproximability results are obtained when the objective is to minimize the sum of completion times.

Moreover, we solve negatively a question raised by Feige and Scheideler [2002], by showing that the optimal solution for flow shops can be a logarithmic factor away from $lb$. This implies that to obtain a better approximation guarantee for flow shops it is necessary to improve the used lower bound on the optimal makespan. When the number of machines and the number of operations per job are assumed to be constant the job shop problem is known to admit a PTAS [Jansen et al. 2003]. This article shows that both these restrictions are necessary to obtain a PTAS. Indeed we prove that the general job shop problem with two machines (denoted as $J2||C_{max}$) has no PTAS. The same limitation applies to the preemptive version with three machines (denoted as $J3|pmtn|C_{max}$). The latter answers negatively an open question raised by Bansal et al. [2006]. A more detailed presentation of the results is given in the following.

*1.2.1. Generalized Flow Shops.* Feige and Scheideler [2002] showed their analysis to be essentially tight for acyclic job shops. As flow shops are more structured than acyclic job shops, they raised as an open question whether the upper bound for flow shop scheduling can be improved significantly. Our first result resolves this question negatively.

THEOREM 1.1. *There exist flow shop instances with optimal makespan*

$$\Omega\left(\mathrm{lb}\cdot\frac{\log \mathrm{lb}}{\log\log \mathrm{lb}}\right).$$

PROOF OVERVIEW. The construction of job shop instances with "large" makespan is presented in Section 2.1 and serves as a good starting point for reading Section 2.2

where the more complex construction of flow shop instances with "large" makespan is presented.

The job shop construction closely follows the construction in Feige and Scheideler [2002] with the main difference being that we do not require all operations of a job to be of the same length, which leads to a slightly better analysis. The main idea is to introduce jobs of different "frequencies", with the property that two jobs of different frequencies essentially cannot be processed at the same time in any feasible schedule. Hence, a job shop instance with $d$ jobs of different frequencies, all of them of length $D$, has optimal makespan $\Omega(d \cdot D)$. Moreover, the construction satisfies $\text{lb} = C = D = 3^d$ and it follows that the job shop instance has optimal makespan $\Omega(d \cdot 3^d)$, which can be written as $\Omega(\text{lb} \cdot \log \text{lb})$.

The construction of flow shop instances with "large" makespan is more complicated, as each job is required to have exactly one operation for every machine, and all jobs are required to go through all the machines in the same order. The main idea is to start with the aforementioned job shop construction, which has "very cyclic" jobs, that is, jobs have many operations on a single machine. The flow shop instance is then obtained by "copying" the job shop instance several times and instead of having cyclic jobs we let the $i$th "long-operation" of a job to be processed by a machine in the $i$th copy of the original job shop instance. Finally, we insert additional zero-length operations to obtain a flow shop instance. By carefully choosing the different frequencies, we can show that the constructed flow shop instance will have essentially the same optimal makespan as the job shop instance we started with. The slightly worse bound, $\Omega(\text{lb} \cdot \log \text{lb}/ \log \log \text{lb})$ instead of $\Omega(\text{lb} \cdot \log \text{lb})$, arises from additional constraints on the design of frequencies. □

If we do not require a job to be processed on *all* machines, that is, generalized flow shops, we prove that it is hard to improve the approximation guarantee. Theorem 1.2 shows that generalized flow shops, with the objective to either minimize makespan or sum of completion times,[2] have no constant approximation algorithm unless $P = NP$.

THEOREM 1.2. *For all sufficiently large constants $K$, it is* NP*-hard to distinguish between generalized flow shop instances that have a schedule with makespan* $2K \cdot \text{lb}$ *and those that have no solution that schedules more than half of the jobs within* $(1/8)K^{\frac{1}{25}(\log K)} \cdot \text{lb}$ *time units. Moreover, this hardness result holds for generalized flow shop instances with bounded number of operations per job, that only depends on $K$.*

PROOF OVERVIEW. The reduction is from the NP-hard problem of deciding whether a graph $G$ with degree bounded by a constant $\Delta$ can be colored using "few" colors or has no "large" independent set (see Theorem 1.7). In Section 3.1, we present a relatively easy reduction to the job shop problem which serves as a good starting point for reading the more complex reduction for the generalized flow shop problem presented in Section 3.2. The main idea is as follows. Given a graph $G$ with bounded degree $\Delta$, we construct a generalized flow shop instance $S$, where all jobs have the same length $D$ and all machines the same load $C = D$. Hence, $\text{lb} = C = D$. Instance $S$ has a set of jobs for each vertex in $G$. By using jobs of different frequencies, as done in the gap construction, we have the property that "many" of the jobs corresponding to adjacent vertices *cannot* be scheduled in parallel in any feasible schedule. On the other hand, by letting jobs skip the machines corresponding to nonadjacent vertices, jobs corresponding to an independent set in $G$ can be scheduled in parallel (their operations can overlap in time) in a feasible schedule. For the reduction to be polynomial

[2]Note that Theorem 1.2 implies that for sufficiently large constants $K$, it is NP-hard to distinguish whether $\sum C_j \leq n \cdot 2K \cdot \text{lb}$ or $\sum C_j \geq \frac{n}{2} \cdot (1/8) \cdot K^{\frac{1}{25}(\log K)} \cdot \text{lb}$, where $n$ is the number of jobs.

it is crucial that the number of frequencies is relatively small. However, to ensure the desired properties, jobs corresponding to adjacent vertices must be of different frequencies. We resolve this by using the fact that $G$ has bounded degree. Since the graph $G$ has degree of at most $\Delta$ we can in polynomial time partition its vertices into $\Delta + 1$ independent sets. As two jobs only need to be assigned different frequencies if they correspond to adjacent vertices, we only need a constant $(\Delta + 1)$ number of frequencies.

The analysis follows naturally: a set of jobs corresponding to an independent set can be scheduled in parallel. Hence, if the graph $G$ can be colored with few, say $F$, colors then there is a schedule of $S$ with makespan $O(F \cdot \mathrm{lb})$. Finally, if there is a schedule where at least half the jobs finish within $L \cdot \mathrm{lb}$ time units then jobs corresponding to at least a fraction $\Omega(1/L)$ of the vertices overlap at some point in the schedule. As the jobs overlap, they correspond to a fraction $\Omega(1/L)$ of vertices that form an independent set. It follows that if the graph has no large independent set, then the generalized flow shop instance has no short schedule. □

By making a stronger assumption, we give a hardness result that essentially shows that the current approximation algorithms for generalized flow shops (and acyclic job shops), with both makespan and sum of weighted completion times objectives, are tight.

THEOREM 1.3. *Let $\epsilon > 0$ be an arbitrarily small constant. There is no $O\left((\log \mathrm{lb})^{1-\epsilon}\right)$-approximation algorithm for $F|jumps|C_{max}$ or $F|jumps|\sum C_j$, unless $NP \subseteq ZTIME(2^{\log n^{O(1/\epsilon)}})$.*

PROOF OVERVIEW. The construction of the generalized flow shop instance is the same as in the proof of Theorem 1.2. To obtain the stronger result, we use a stronger hardness result for graph coloring (see Theorem 1.8). The tricky part is that the graph no longer has a small bounded degree. We overcome this difficulty by a randomized process that preserves the desired properties of the graph with an overwhelming probability (see Lemma 3.1). □

For flow shops, the consequences of Theorem 1.1 and Theorem 1.3 are among others that in order to improve the approximation guarantee, it is necessary to (i) improve the used lower bound on the optimal makespan and (ii) use the fact that a job needs to be processed on all machines.

From Theorem 1.2 and Theorem 1.3, we have the following for the more general job shop problem.

THEOREM 1.4. *Job Shops are NP-hard to approximate within any constant and have no $O\left((\log \mathrm{lb})^{1-\epsilon}\right)$-approximation algorithm for any $\epsilon > 0$, unless $NP \subseteq ZTIME(2^{\log n^{O(1/\epsilon)}})$.*

*1.2.2. Job Shop with Bounded Number of Machines.* In Jansen et al. [2003], a PTAS was given for the (preemptive) job shop problem, where both the number of machines and the number of operations per job are assumed to be constant. Our second result shows that both these restrictions are necessary to obtain a PTAS (that one needs to restrict the number of machines follows from the work in Williamson et al. [1997]).

THEOREM 1.5. *The job shop problem with two machines ($J2||C_{\max}$) has no PTAS unless $NP \subseteq DTIME(n^{O(\log n)})$.*

PROOF OVERVIEW. In Section 4.1, we prove the result by presenting a gap-preserving reduction from the independent set problem in cubic graphs, that is, graphs where all vertices have degree three (see Theorem 1.9).

Given a cubic graph $G$ we construct an instance $S$ of $J2||C_{\max}$ as follows. The instance has a "big" job, called $j_b$, whose length will equal the makespan in the completeness case. Its operations are divided into four parts, called the *edge-, tail-, slack-,* and *remaining-part*. There is also a vertex job for each vertex. We again use the technique of introducing different frequencies of jobs; this time to ensure that, without delaying job $j_b$, two jobs corresponding to adjacent vertices cannot both complete before the end of the tail-part of job $j_b$.

The analysis now follows from selecting the lengths of the different parts of $j_b$ such that in the completeness case we can schedule all jobs, corresponding to a "big" independent set of $G$, in parallel with the edge- and tail-part of job $j_b$ and the remaining jobs are scheduled in parallel with the slack- and remaining-part of job $j_b$. In contrast, in the soundness case, as $G$ has no "big" independent set, we can, without delaying the schedule, only schedule relatively few jobs in parallel with the edge- and tail-part of job $j_b$. The remaining jobs, relatively many, will then require more time units than the total length of the slack- and remaining-part of job $j_b$ and it follows that the schedule will have makespan larger than the length of $j_b$.

The reduction runs in time $n^{O(f)}$, where $f$ is the number of frequencies. With our current techniques we need $O(\log n)$ frequencies and hence the assumption used in the statement of Theorem 1.5. □

For the general preemptive job shop problem, Bansal et al. [2006] showed a $(2+\varepsilon)$-approximation for a constant number of machines. According to Bansal et al. [2006], it is an an "outstanding open question" to understand whether there is a PTAS for the general preemptive job shop with a constant number of machines. We solve (negatively) the open question raised in Bansal et al. [2006].

THEOREM 1.6. *The preemptive job shop problem with three machines ($J3|pmtn|C_{\max}$) has no PTAS unless $NP \subseteq DTIME(n^{O(\log n)})$.*

The proof of Theorem 1.6 is more involved than the proof of Theorem 1.5, but it has a similar structure. We remark that the approximability of $J2|pmtn|C_{max}$ is still open.

### 1.3. Preliminaries

When considering a schedule, we shall say that two jobs (or operations) overlap or are scheduled in parallel for $t$ time units if $t$ time units of them are processed at the same time on different machines. For a given graph $G$, we let $\chi(G)$ and $\alpha(G)$ denote the chromatic number of $G$ and the size of a maximum independent set of $G$, respectively. We shall also denote the maximum degree of graph $G$ by $\Delta(G)$, where we sometimes drop $G$ when it is clear from the context.

Our reductions to the job shop problem with unbounded number of machines use results by Khot [2001], who proved that even though we know that the graph is colorable using $K$ colors it is NP-hard to find a coloring that uses less than $K^{\frac{1}{25}(\log K)}$ colors, for sufficiently large constants $K$. The result was obtained by presenting a polynomial-time reduction that takes as input a SAT formula $\phi$ together with a sufficiently large constant $K$ and outputs an $n$-vertex graph $G$ with degree at most $2^{K^{O(\log K)}}$ such that (completeness) if $\phi$ is satisfiable then $G$ can be colored using at most $K$ colors and (soundness) if $\phi$ is not satisfiable then $G$ has no independent set containing $n/K^{\frac{1}{25}(\log K)}$ vertices (see Section 6 in Khot [2001]). Note that the soundness case implies that any feasible coloring of the graph uses at least $K^{\frac{1}{25}(\log K)}$ colors. We let $\mathcal{G}[c, f]$ be the family of graphs that either can be colored using $c$ colors or have no independent set containing a fraction $f$ of the vertices. The following theorem (not stated in this form in Khot [2001]) summarizes the result obtained.

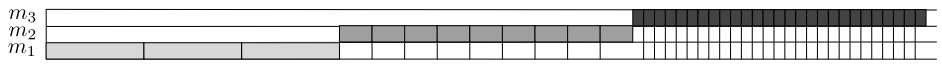


Fig. 2. An example of the construction with $d = 3$. The long-operations of jobs of frequency 1, 2 and 3 are depicted in light, medium, and dark gray, respectively. Job $j_i$ has $3^i$ long-operations on machine $m_i$, for $i \in \{1, 2, 3\}$.

THEOREM 1.7 [KHOT 2001]. *For all sufficiently large constants K, it is* NP*-hard to decide if a graph in* $\mathcal{G}[K, 1/K^{\frac{1}{25}(\log K)}]$ *can be colored using K colors or has no independent set containing a fraction* $1/K^{\frac{1}{25}(\log K)}$ *of the vertices. Moreover, this hardness result holds for graphs with degree at most* $2^{K^{O(\log K)}}$.

By using a stronger assumption, we can let $K$ be a function of the number of vertices. Again, the stronger statement (not explicitly stated in Khot [2001]) follows from the soundness analysis.

THEOREM 1.8 [KHOT 2001]. *There exists an absolute constant* $\gamma > 0$ *such that, for all* $K \leq 2^{(\log n)^\gamma}$*, there is no polynomial-time algorithm that decides if an n-vertex graph in* $\mathcal{G}[K, 1/K^{\Omega(\log K)}]$ *can be colored using K colors or has no independent set containing a fraction* $1/K^{\Omega(\log K)}$ *of the vertices, unless* $NP \subseteq DTIME(2^{O(\log n)^{O(1)}})$.

Our reduction to $J2||C_{\max}$ uses the following result by Alimonti and Kann [2000]. A cubic graph is a graph where all vertices have degree three.

THEOREM 1.9 [ALIMONTI AND KANN 2000]. *There exist positive constants* $\beta, \alpha$ *with* $\beta > \alpha$*, so that it is NP-hard to distinguish between n-vertex cubic graphs that have an independent set of size* $\beta \cdot n$ *and those that have no independent set of size* $\alpha \cdot n$.

## 2. JOB AND FLOW SHOP INSTANCES WITH LARGE MAKESPAN

We first exhibit a family of instances of general job shop scheduling for which it is relatively simple to show that any optimal schedule is of length $\Omega(\text{lb} \cdot \log \text{lb})$. This complements[3] and builds on the bound by Feige and Scheideler [2002], who showed the existence of job shop instances with optimal makespan $\Omega(\text{lb} \cdot \log \text{lb} / \log \log \text{lb})$. We then use this construction as a building block in the more complicated flow shop construction.

### 2.1. Job Shops with Large Makespan

*2.1.1. Construction.* For any integer $d \geq 1$, consider the job shop instance with $d$ machines $m_1, \ldots, m_d$ and $d$ jobs $j_1, \ldots, j_d$. We say that job $j_i$ has *frequency i*, which means that it has $3^i$ so-called *long-operations* on machine $m_i$, each one of them requires $3^{d-i}$ time units. Between any two consecutive long-operations, job $j_i$ has *short-operations* that require 0 time units on the machines $m_1, \ldots, m_{i-1}$. Note that the length of all jobs and the load on all machines are $3^d$, which we denote by lb. See Figure 2 for an example of the construction.

*2.1.2. Analysis.* Fix an arbitrary feasible schedule of the jobs. We shall show that the length of the schedule must be $\Omega(\text{lb} \cdot \log \text{lb})$.

LEMMA 2.1. *For* $i, j : 1 \leq i < j \leq d$*, the number of time units during which both* $j_i$ *and* $j_j$ *perform operations is at most* $\frac{\text{lb}}{3^{j-i}}$.

[3]In their construction, all operations of a job have the same length which is not the case for our construction.

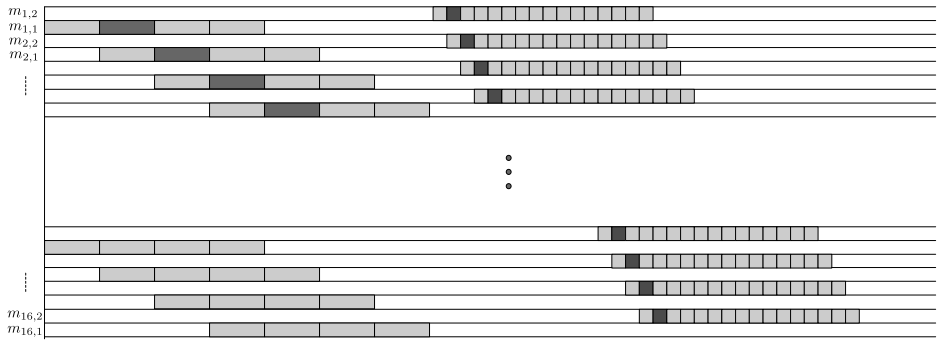


Fig. 3. An example of the construction for flow shop scheduling with $r = d = 2$. Only long-operations on the first 4 and last 4 groups of machines are depicted. The long-operations of one job of each frequency are highlighted in dark gray.

PROOF. During the execution of a long-operation of $j_i$ (that requires $3^{d-i}$ time units), job $j_j$ can complete at most one long-operation that requires $3^{d-j}$ time units (since its short-operation on machine $m_i$ has to wait). As $j_i$ has $3^i$ long-operations, the two jobs can perform operations at the same time during at most $3^i \cdot 3^{d-j} = \frac{3^d}{3^{j-i}} = \frac{\text{lb}}{3^{j-i}}$ time units. □

It follows that, for each $i = 1, \ldots, d$, at most a fraction $1/3 + 1/3^2 + \cdots + 1/3^i \leq 1/3 + 1/3^2 + \cdots + 1/3^d \leq \frac{1}{3-1} = 1/2$ of the time spent for long-operations of a job $j_i$ is performed at the same time as long-operations of jobs with lower frequency. Hence, a feasible schedule has makespan at least $d \cdot \text{lb}/2$. As $d = \Omega(\log \text{lb})$ (recall that $\text{lb} = 3^d$), the optimal makespan of the constructed job shop instance is $\Omega(\text{lb} \cdot \log \text{lb})$.

### 2.2. Flow Shops with Large Makespan

*2.2.1. Construction.* For sufficiently large integers $d$ and $r$, consider the flow shop instance defined as follows (we refer to Figure 3 for an example of the construction with $d = r = 2$).

— There are $r^{2d}$ groups of machines[4], denoted by $M_1, M_2, \ldots, M_{r^{2d}}$. Each group $M_g$ consists of $d$ machines $m_{g,1}, m_{g,2}, \ldots, m_{g,d}$ (one for each frequency). Finally the machines are ordered in such a way that $m_{g,i}$ is before $m_{h,j}$ if either (i) $g < h$ or (ii) $g = h$ and $i > j$. The latter case will ensure that, within each group of machines, long-operations of jobs with high frequency will be scheduled before long-operations of jobs with low frequency, a fact that will be used in the proof of Lemma 2.3.

In the example of Figure 3, there are $2^4$ groups of machines, where machines $\{m_{g,1}, m_{g,2}\}$ are those that belong to group $M_g$. The only long-operations that can be scheduled on machine $m_{g,i}$ are those belonging to jobs of frequency $i$. In top down order, note that machine $m_{g,2}$ is before $m_{g,1}$ and before $m_{h,i}$ for any $g < h$ and $i \in \{1, 2\}$.

— For each frequency $f = 1, \ldots, d$, there are $r^{2(d-f)}$ groups of jobs, denoted by $J_1^f, J_2^f, \ldots, J_{r^{2(d-f)}}^f$. Each group $J_g^f$ consists of $r^{2f}$ copies, referred to as $j_{g,1}^f, j_{g,2}^f, \ldots, j_{g,r^{2f}}^f$, of the job that must be processed during $r^{2(d-f)}$ time units on the machines

$$m_{a+1,f}, m_{a+2,f}, \ldots, m_{a+r^{2f},f} \text{ where } a = (g-1) \cdot r^{2f}$$

[4]These groups of machines "correspond" to copies of the job shop instance in Section 2.1.

and during 0 time units on all the other machines that are required to create a flow shop instance. Let $J^f$ be the set of jobs that correspond to frequency $f$, that is, $J^f = \{j^f_{g,a} : 1 \le g \le r^{2(d-f)}, 1 \le a \le r^{2f}\}$.

(In the example of Figure 3, there are four groups of jobs with frequency one, and one group with frequency two. Every group of frequency one has four identical jobs: in the figure, on machine $m_{1,1}$ the leftmost operations of equal length are the first operations of the jobs in group $J^1_1 = \{j^1_{1,1}, \ldots, j^1_{1,4}\}$; the four leftmost gray operations in cascade are the operations of job $j^1_{1,2}$; the leftmost operations on machines $m_{13,1}, m_{14,1}, m_{15,1}, m_{16,1}$ belong to the last group $J^1_4$ of frequency one. The rightmost (smaller) operations are those belonging to the unique group $J^2_1$ of frequency two; the rightmost gray operations in cascade are, respectively, the first and the last four operations out of the 16 operations of job $j^2_{1,2}$.)

Note that the length of any job and the load on any machine are $r^{2d}$, which equals lb. Moreover, the total number of machines and the total number of jobs are both $r^{2d} \cdot d$.

In the subsequent, we will call the operations that require more than 0 time units *long-operations* and the operations that only require 0 time units *short-operations*.

*2.2.2. Analysis.* We shall show that the length of any feasible schedule must be $\Omega(\text{lb} \cdot \min[r, d])$. As $\text{lb} = r^{2d}$, instances constructed with $r = d$ have optimal makespan $\Omega(\text{lb} \cdot \log \text{lb} / \log\log \text{lb})$.

Fix an arbitrarily feasible schedule for the jobs. We start by showing a useful property. For a job $j$, let $d_j(i)$ denote the delay between its $i$th and $(i+1)$-th long-operations, that is, the time units between the end of job $j$'s $i$th long-operation and the start of its $(i+1)$-th long-operation (let $d_j(i) = \infty$ for the last long-operation). We say that the $i$th long-operation of a job $j$ of frequency $f$ is *good* if $d_j(i) \le \frac{r^2}{4} \cdot r^{2(d-f)}$.

LEMMA 2.2. *If the schedule has makespan less than $r \cdot \text{lb}$, then the fraction of good long-operations of each job is at least $\left(1 - \frac{4}{r}\right)$.*

PROOF. Assume that the considered schedule has makespan less than $r \cdot \text{lb}$. Suppose toward contradiction that there exists a job $j$ of frequency $f$ so that $j$ has at least $\frac{4}{r} r^{2f}$ long-operations that are not good. But then the length of $j$ is at least $\frac{4}{r} r^{2f} \cdot \frac{r^2}{4} \cdot r^{2(d-f)} = r \cdot r^{2d} = r \cdot \text{lb}$, which contradicts that the makespan of the considered schedule is less than $r \cdot \text{lb}$. □

We continue by analyzing the schedule with the assumption that its makespan is less than $r \cdot \text{lb}$ (otherwise, we are done). In each group $M_g$ of machines, we will associate a set $T_{g,f}$ of time intervals for each frequency $f = 1, \ldots, d$. The set $T_{g,f}$ contains the time intervals corresponding to the *first half* of all good long-operations scheduled on the machine $m_{g,f}$. For intuition of the following lemma, see Figure 4.

LEMMA 2.3. *Let $k, \ell : 1 \le k < \ell \le d$ be two different frequencies. Then, the sets $T_{g,k}$ and $T_{g,\ell}$, for all $g : 1 \le g \le r^{2d}$, contain disjoint time intervals.*

PROOF. Suppose toward contradiction that there exist time intervals $t_k \in T_{g,k}$ and $t_\ell \in T_{g,\ell}$ that overlap, that is, $t_k \cap t_\ell \ne \emptyset$. Note that $t_k$ and $t_\ell$ correspond to good long-operations of jobs of frequencies $k$ and $\ell$, respectively. Let us say that the good long-operation corresponding to $t_\ell$ is the $a$th operation of some job $j$. Note that, since $j$ has a long-operation on machine $m_{g,\ell}$, job $j$ has frequency $\ell$. As $t_\ell$ and $t_k$ overlap, the $a$th long-operation of $j$ must overlap the first half of the long-operation corresponding to $t_k$. As job $j$ has a short operation on machine $m_{g,k}$ after its long-operation on machine

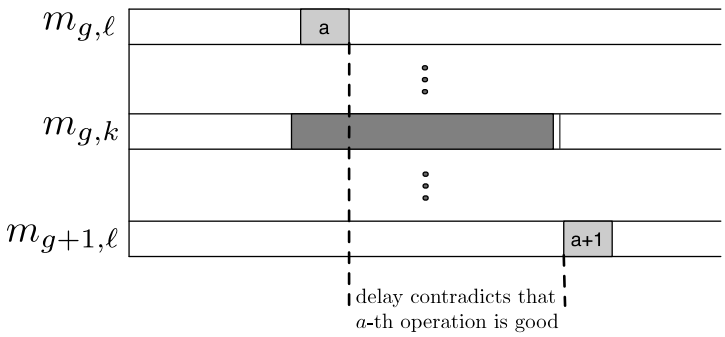


Fig. 4. The intuition behind the proof of Lemma 2.3.

$m_{g,\ell}$ (recall that machines are ordered $m_{g,d}, m_{g,d-1}, \ldots, m_{g,1}$ and $\ell > k$), job j's $(a+1)$-th operation must be delayed by at least $r^{2(d-k)}/2 - r^{2(d-\ell)}$ time units and thus $d_j(a) > r^{2(d-k)}/2 - r^{2(d-\ell)} > \frac{r^2}{4} r^{2(d-\ell)}$ (using $\ell > k$), which contradicts that the $a$th long-operation of job $j$ is good. □

Let $L(T_{g,f})$ denote the total time units covered by the time intervals in $T_{g,f}$. We continue by showing that there exists a $g$ such that $\sum_{f=1}^{d} L(T_{g,f}) \geq \frac{\text{lb}}{4} \cdot d$. With this in place, it is easy to see that any schedule has makespan $\Omega(d \cdot \text{lb})$ since all the time intervals $\{T_{g,f} : f = 1, \ldots, d\}$ are disjoint (Lemma 2.3).

LEMMA 2.4. *There exists a* $g \in \{1, \ldots, r^{2d}\}$ *such that*

$$\sum_{f=1}^{d} L(T_{g,f}) \geq \frac{\text{lb}}{4} \cdot d.$$

PROOF. As $\sum_{f=1}^{d} L(T_{g,f})$ adds up the time units required by the first half of each good long-operation scheduled on a machine in $M_g$, the claim follows by showing that there exists a group of machines $M_g$ from $\{M_1, M_2, \ldots, M_{r^{2d}}\}$ so that the total time units required by the good long-operations on the machines in $M_g$ is at least $\frac{\text{lb} \cdot d}{2}$.

By Lemma 2.2 we have that the good long-operations of each job require at least $\text{lb} \cdot \left(1 - \frac{4}{r}\right)$ time units. Since the total number of jobs is $r^{2d}d$ the total time units required by all good long-operations is at least $\text{lb} \cdot \left(1 - \frac{4}{r}\right) \cdot r^{2d}d$. As there are $r^{2d}$ many groups of machines, a simple averaging argument guarantees that in at least one group of machines, say $M_g$, the total time units required by the good long-operations on the machines in $M_g$ is at least $\text{lb} \cdot \left(1 - \frac{4}{r}\right) d > \text{lb} \cdot d/2$ for a sufficiently large $r$. □

## 3. HARDNESS OF JOB SHOPS AND GENERALIZED FLOW SHOPS

Theorem 1.2 and Theorem 1.3 are proved by presenting a gap-preserving reduction, $\Gamma$, from the graph coloring problem to the generalized flow shop problem, that has two parameters, $r$ and $d$. Given an $n$-vertex graph $G$ whose vertices are partitioned into $d$ independent sets, it computes in time polynomial in $n$ and $r^d$, a generalized flow shop instance $S(r, d)$ where vertices are mapped to jobs. By using jobs of different frequencies, as done in the gap construction, we have the property that "many" of the jobs corresponding to adjacent vertices cannot be scheduled in parallel in any feasible schedule. On the other hand, by letting jobs skip those machines corresponding to non-adjacent vertices, jobs corresponding to an independent set in $G$ *can* be scheduled

in parallel (their operations can overlap in time) in a feasible schedule. This ensures that the following completeness and soundness hold for the resulting generalized flow shop instance $S(r, d)$.

— *Completeness*. If $G$ can be colored using $L$ colors, then $C^*_{max} \leq \text{lb} \cdot 2L$.
— *Soundness*. For any $L \leq r$. Given a schedule where at least half the jobs finish within $\text{lb} \cdot L$ time units, we can find an independent set of $G$ of size $n/(8L)$, in time polynomial in $n$ and $r^d$.

In the proposed construction all jobs have the same length $r^{2d}$ and all machines the same load $r^{2d}$. Hence, $\text{lb} = r^{2d}$. Instance $S(r, d)$ has a set of $r^{2d}$ jobs and a set of $r^{2d}$ machines for each vertex in $G$. The total number of jobs and the total number of machines are thus both $r^{2d}n$. Moreover, each job has at most $dr^{2d}$ operations.

In Section 3.1, we present a reduction with somewhat stronger properties for the *job shop* problem. As the reduction is relatively simple, it serves as a good starting point before reading the similar but more complex reduction $\Gamma$ for the *generalized flow shop* problem. Before continuing, let us see how the reduction $\Gamma$, with the aforementioned properties, is sufficient for proving Theorem 1.2 and Theorem 1.3.

PROOF OF THEOREM 1.2. By Theorem 1.7, for sufficiently large $K$ and $\Delta = 2^{K^{O(\log K)}}$, it is NP-hard to decide if an $n$-vertex graph $G$ in $\mathcal{G}[K, 1/K^{\frac{1}{25}(\log K)}]$ with bounded degree $\Delta$ has

$$\chi(G) \leq K \quad \text{or} \quad \alpha(G) \leq \frac{n}{K^{\frac{1}{25}(\log K)}}.$$

As the vertices of a graph with bounded degree $\Delta$ can, in polynomial time, be partitioned into $\Delta + 1$ independent sets, we can use $\Gamma$ with parameters $d = \Delta + 1$ and $r = K^{\frac{1}{25}(\log K)}$ ($r$ is chosen such that the condition $L \leq r$ in the soundness case of $\Gamma$ is satisfied for $L = K^{\frac{1}{25}(\log K)}/8$). It follows by the completeness case and soundness case of $\Gamma$ that it is NP-hard to distinguish if the obtained scheduling instance has a schedule with makespan at most $\text{lb} \cdot 2K$ or no solution schedules more than half of the jobs within $\text{lb} \cdot K^{\frac{1}{25}(\log K)}/8$ time units. Moreover, each job has at most $dr^{2d}$ operations, which is a constant that only depends on $K$.

PROOF OF THEOREM 1.3. The proof is similar to the proof of Theorem 1.2 with the exception that the graphs have no longer bounded degree. Note that, since the construction is exponential in $d$, we aim for a "small" value of $d$ that ensures that two jobs corresponding to two adjacent nodes get different frequencies. This can be done if the graph can be colored with "few" colors. If the graph has "small" bounded degree $\Delta$, then $d = \Delta + 1$ suffices to ensure the above property. Otherwise, the following lemma says that we can construct another graph $G'$ with "similar" properties as $G$, but $G'$ is easily colorable with at most $(\log n)^\delta$ colors, which is sufficiently "small" for our purposes.

When using probabilistic arguments for graphs with $n$ vertices, we shall use the term *overwhelming* (*negligible*, respectively) to denote probability that tends to 1 (to 0, respectively) as $n$ tends to infinity.

LEMMA 3.1. *For any constant $\delta \geq 1$, given an $n$-vertex graph $G = (V, E)$, we can construct in randomized polynomial time a subgraph $G' = (V, E')$ of $G$ with $E' \subseteq E$ such that*

(1) *The vertices are partitioned into $(\log n)^\delta$ sets, each set forms an independent set in $G'$.*
(2) $\chi(G') \leq \chi(G)$.

(3) *With overwhelming probability the following holds: given an independent set of $G'$, with $\frac{n}{(\log n)^{\delta-1}}$ vertices, we can find an independent set of $G$ with $\frac{n}{(\log n)^{\delta}}$ vertices, in polynomial time.*

PROOF. Given an $n$-vertex graph $G = (V, E)$, we give a probabilistic construction of $G' = (V, E')$. Each vertex $v \in V$ is assigned independently, uniformly at random to one of the sets $I_1, I_2, \ldots, I_{(\log n)^{\delta}}$. Let $E' \subseteq E$ be those edges that are incident to vertices placed in different sets, that is, an edge is deleted from $E$ to yield $E'$ if and only if it is adjacent to two vertices $u \in I_i$ and $v \in I_i$ for some $i : 1 \leq i \leq (\log n)^{\delta}$.

The graph $G'$ obviously satisfies the first two properties in the lemma. We continue by showing that $G'$ satisfies property 3. In fact, we show that the following stronger property holds with overwhelming probability: any independent set $I'$ of $G'$, with $|I'| = \frac{n}{(\log n)^{\delta-1}}$, induces a subgraph of $G$ with at least $\frac{n}{(\log n)^{\delta}}$ maximal connected components. This is done by proving that if a subset $V'$ of $n/(\log n)^{\delta-1}$ vertices induces a subgraph of $G$ with less than $\frac{n}{(\log n)^{\delta}}$ maximal connected components, then the probability that $V'$ is an independent set of $G'$ is negligible.

Fix a set $V' \subseteq V$ of $n/(\log n)^{\delta-1}$ vertices and let $H$ be the subgraph of $G$ induced by $V'$. Assuming that $H$ can be partitioned into $s$ maximal connected components, with $s < \frac{n}{(\log n)^{\delta}}$, we calculate the probability that $V'$ forms an independent set in $G'$. Let $H_1, H_2, \ldots, H_s$ denote the maximal connected components of $H$. We use $|H_\ell|$, for $\ell : 1 \leq \ell \leq s$, to denote the number of vertices of $H_\ell$. If the vertices of $H$ form an independent set in $G'$ then all vertices of a connected component must be placed in the same set $I_i$, for some $i : 1 \leq i \leq (\log n)^{\delta}$. The probability that this happens, for a connected component with $k$ vertices, is at most $\left(\frac{1}{\log n}\right)^{\delta(k-1)}$. As the different maximal connected components are independent, the probability that $V'$ forms an independent set in $G'$ is at most

$$\left(\frac{1}{\log n}\right)^{\delta(|H_1|-1)} \left(\frac{1}{\log n}\right)^{\delta(|H_2|-1)} \cdots \left(\frac{1}{\log n}\right)^{\delta(|H_s|-1)} = \left(\frac{1}{\log n}\right)^{\delta\left(\sum_{i=1}^{s}|H_i|-s\right)}.$$

As $\sum_{i=1}^{s}|H_i| = |V'| = n/(\log n)^{\delta-1}$ and $s < \frac{n}{(\log n)^{\delta}}$, the probability that $V'$ forms an independent set in $G'$ is at most

$$\left(\frac{1}{\log n}\right)^{\delta \cdot n/(\log n)^{\delta-1} \cdot (1-1/\log n)}.$$

The number of ways to fix the set $V'$ is at most

$$\binom{n}{n/(\log n)^{\delta-1}} \leq \left(e \cdot \log n\right)^{(\delta-1)n/(\log n)^{\delta-1}}.$$

Hence, the union bound implies that the probability that graph $G'$ fails to satisfy property 3 is at most

$$\left(\frac{1}{\log n}\right)^{\delta \cdot n/(\log n)^{\delta-1} \cdot (1-1/\log n)} \cdot \left(e \cdot \log n\right)^{(\delta-1)n/(\log n)^{\delta-1}}$$

which tends to 0 as $n$ tends to infinity. □

Assuming $NP \not\subseteq DTIME(2^{O(\log n)^{O(1)}})$, Theorem 1.8 with $K = \log n$ says that there is no polynomial algorithm that decides if an $n$-vertex graph $G$ in $\mathcal{G}[\log n, 1/(\log n)^{\Omega(\log\log n)}]$ has

$$\chi(G) \leq \log n \quad \text{or} \quad \alpha(G) \leq \frac{n}{(\log n)^{\Omega(\log\log n)}}.$$

Given an $n$-vertex graph $G$ in $\mathcal{G}[\log n, 1/(\log n)^{\Omega(\log\log n)}]$, we construct graph $G'$ from $G$ by applying Lemma 3.1 with $\delta = 3/\epsilon$, where $\epsilon > 0$ is an arbitrarily small constant. We then obtain a generalized flow shop instance $S$ from $G'$ by using $\Gamma$ with parameters $r = (\log n)^{\delta-1}$ and $d = (\log n)^{\delta}$. The size of $S$ is $O(r^{2d}n \cdot \Delta r^{2d}) = O(2^{O(\log n)^{O(1/\epsilon)}})$ and $\text{lb} = r^{2d} = (\log n)^{2(\delta-1)(\log n)^{\delta}}$ and hence $\log \text{lb} \leq (\log n)^{\delta+1}$ (for large enough $n$).

The analysis is straightforward.

— *Completeness*. If $\chi(G) \leq \log n$, then $\chi(G') \leq \log n$ and, by the completeness case of $\Gamma$, there is a schedule of $S$ with makespan $\text{lb} \cdot 2\log n$.

— *Soundness*. Assuming that the probabilistic construction of $G'$ succeeded, we have

$$\alpha(G) \leq \frac{n}{n^{\Omega(\log\log n)}} \Rightarrow \alpha(G') \leq \frac{n}{(\log n)^{\delta-1}},$$

which in turn implies by the soundness case of $\Gamma$ that no solution schedules more than half the jobs within $\text{lb} \cdot (\log n)^{\delta-1}/8$ time units (recall that $r$ was chosen to be greater than $(\log n)^{\delta-1}/8$).

The probabilistic construction of $G'$ succeeds with overwhelming probability. Furthermore, given a schedule, we can detect such a failure in polynomial time and repeat the reduction. It follows that an approximation algorithm for $F|jumps|C_{max}$ with performance guarantee

$$\frac{(\log n)^{\delta-1}/8}{2\log n} = (\log n)^{\delta-2}/16$$

or an approximation algorithm for $F|jumps|\sum C_j$ with performance guarantee

$$\frac{(n/2)\cdot(\log n)^{\delta-1}/8}{2n\cdot\log n} = (\log n)^{\delta-2}/32$$

would imply that $NP \subseteq ZTIME(2^{O(\log n)^{O(1/\epsilon)}})$. Finally, we note that $\delta$ was chosen so that for large enough $n$ we have

$$(\log \text{lb})^{1-\epsilon} \leq (\log n)^{(1-\epsilon)(\delta+1)} = (\log n)^{3/\epsilon+1-3-\epsilon} =$$

$$(\log n)^{\delta-2-\epsilon} < 1/32 \cdot (\log n)^{\delta-2}.$$

### 3.1. Job Shops

In this section, we give and analyze a somewhat stronger reduction than $\Gamma$ for the *general* job shop problem. The number of operations per job is at most $(\Delta+1)r^d$, the number of jobs and the number of machines are $n$, and the soundness case says that, given a schedule with makespan $\text{lb} \cdot L$, we can, in time polynomial in $n$ and $r^d$, find an independent set of $G$ of size $(1-\frac{\Delta}{r})n/L$.[5] As the reduction is relatively simple, it serves as a good starting point for reading the more complex reduction to the generalized flow shop problem.

[5]The condition that $r$ needs to be greater than $\Delta$ can be removed as done in the analysis of generalized flow shops. In this section, we have chosen to provide a simpler analysis, which still describes most of the ideas used in the flow shop analysis.


Fig. 5. An example of the reduction with $r = 4, d = 2, I_1 = \{A\}$, and $I_2 = \{B, C\}$. Note that jobs only have short-operations on machines corresponding to adjacent vertices. (The jobs corresponding to $A$, $B$, and $C$ are depicted to the left, center, and right respectively).

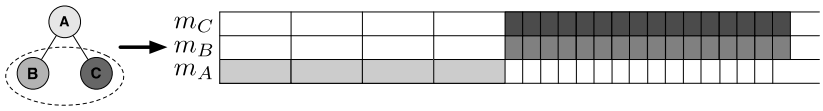


Fig. 6. An example of how the jobs are scheduled in the completeness case. Here, $V_1 = \{A\}$ and $V_2 = \{B, C\}$.

*3.1.1. Construction.* Given an $n$-vertex graph $G = (V, E)$ whose vertices are partitioned into $d$ independent sets, we create a job shop instance $S(r, d)$, where $r$ and $d$ are the parameters of the reduction. Instance $S(r, d)$ has a machine $m_v$ and a job $j_v$ for each vertex $v \in V$. We continue by describing the operations of the jobs. Let $I_1, I_2, \ldots, I_d$ denote the independent sets that form a partition of $V$. A job $j_v$ that corresponds to a vertex $v \in I_i$, for some $i : 1 \leq i \leq d$, has a chain of $r^i$ *long-operations* $O_{1,j_v}, O_{2,j_v}, \ldots, O_{r^i,j_v}$, each of them requiring $r^{d-i}$ time units, that must be processed on the machine $m_v$. Between two consecutive long-operations $O_{p,j_v}, O_{p+1,j_v}$, for $p : 1 \leq p < r^i$, we have a set of *short-operations* placed on the machines $\{m_u : \{u, v\} \in E\}$ (the machines corresponding to adjacent vertices) in some order. A short-operation requires time 0. For an example of the construction, see Figure 5.

*Remark* 3.2. The construction has $n$ machines and $n$ jobs. Each job has length $r^d$ and each machine has load $r^d$. Hence, $\text{lb} = r^d$. Moreover, the number of operations per job is at most $(\Delta + 1)r^d$.

*3.1.2. Completeness.* We prove that if the graph $G$ can be colored with $L$ colors then there is a "relatively short" solution to the job shop instance (see Figure 6).

LEMMA 3.3. *If $\chi(G) = L$, then there is a schedule of $S(r, d)$ with makespan* $\text{lb} \cdot L$.

PROOF. Let $V_1, V_2, \ldots, V_L$ be a partition of $V$ into $L$ independent sets. Consider one of these sets, say $V_i$. As the vertices of $V_i$ form an independent set, no short-operations of the jobs $\{j_v\}_{v \in V_i}$, are scheduled on the machines $\{m_v\}_{v \in V_i}$. Since short-operations require time 0 we can schedule the jobs $\{j_v\}_{v \in V_i}$ within lb time units. We can thus schedule the jobs in $L$-"blocks" in the order $\{j_v\}_{v \in V_1}, \{j_v\}_{v \in V_2}, \ldots, \{j_v\}_{v \in V_L}$. The total length of this schedule is $\text{lb} \cdot L$. □

*3.1.3. Soundness.* We prove that, given a schedule where many jobs are completed "early", we can, in polynomial time, find a "big" independent set of $G$.

LEMMA 3.4. *Given a schedule of $S(r, d)$ where at least half the jobs finish within* $\text{lb} \cdot L$ *time units, we can, in time polynomial in $n$ and $r^d$, find an independent set of $G$ of size at least* $(1 - \frac{\Delta}{r})n/(2L)$.

PROOF. First, we show that two jobs corresponding to adjacent vertices cannot be scheduled in parallel. (The proof of the following claim is similar to the proof of Lemma 2.1 in the gap construction.)

CLAIM 3.5. *Let $u \in I_i$ and $v \in I_j$ be two adjacent vertices in $G$ with $i < j$. Then, at most, a fraction $\frac{1}{r}$ of the long-operations of $j_v$ can overlap the long-operations of $j_u$.*

PROOF OF CLAIM. There are $r^i$ and $r^j$ long-operations of $j_u$ and $j_v$, respectively. As the vertices $u$ and $v$ are adjacent, job $j_v$ has a small-operation on machine $m_u$ between any two long-operations. Hence, at most one long-operation of $j_v$ can be scheduled in parallel with a long-operation of $j_u$ and the total number of such operations in any schedule is at most $r^i \leq \frac{r^j}{r}$ (using $i < j$). □

Now consider a schedule where at least half the jobs finish within $\text{lb} \cdot L$ time units. For each $i : 1 < i \leq d$ and for each $v \in I_i$, we disregard those long-operations of job $j_v$ that overlap long-operations of the jobs $\{j_u : \{u, v\} \in E \text{ and } u \in I_j \text{ for some } j < i\}$. After disregarding operations, no two long-operations corresponding to adjacent vertices will overlap in time. Furthermore, by applying Claim 3.5 and using that the maximum degree of $G$ is $\Delta$, we know that at most a fraction $\frac{\Delta}{r}$ of a job's long-operations have been disregarded. Thus the remaining long-operations of a job require at least $(1 - \frac{\Delta}{r}) \cdot \text{lb}$ time units. As at least half the jobs ($n/2$ many) finish within $L \cdot \text{lb}$ time units, we have that at least $(1 - \frac{\Delta}{r}) \cdot \text{lb} \cdot n/2$ time units are scheduled on the machines within $L \cdot \text{lb}$ time units. By a simple averaging argument we have that at least $(1 - \frac{\Delta}{r})n/(2L)$ of the remaining long-operations must overlap at some point within the first $L \cdot \text{lb}$ time units in the schedule. As the remaining long-operations that overlap correspond to different vertices that are nonadjacent, the graph has an independent set of size $(1 - \frac{\Delta}{r})n/(2L)$. Moreover, we can find such a point (corresponding to an independent set) in the schedule, for example, by considering the start and end points of all long-operations that were not disregarded. □

### 3.2. Generalized Flow Shops

Here, we present the reduction $\Gamma$ for the general flow shop problem where jobs are allowed to skip machines. The idea is similar to the reduction presented in Section 3.1 for the job shop problem. The main difference is to ensure, without using cyclic jobs, that jobs corresponding to adjacent vertices cannot be scheduled in parallel.

*3.2.1. Construction.* Given an $n$-vertex graph $G = (V, E)$ whose vertices are partitioned into $d$ independent sets, we create a generalized flow shop instance $S(r, d)$, where $r$ and $d$ are the parameters of the reduction. Let $I_1, I_2, \ldots I_d$ denote the independent sets that form a partition of $V$.

The instance $S(r, d)$ is very similar to the gap instance described in Section 2.2. The main difference is that in $S(r, d)$ distinct jobs can be scheduled in parallel if their corresponding vertices in $G$ are not adjacent. This is obtained by letting a job skip those machines corresponding to nonadjacent vertices. (The gap instance of Section 2.2 can be seen as the result of the following reduction when the graph $G$ is a complete graph with $d$ nodes.) For convenience, we give the complete description with the necessary changes.

— There are $r^{2d}$ groups of machines, denoted by $M_1, M_2, \ldots, M_{r^{2d}}$. Each group $M_g$ consists of $n$ machines $\{m_{g,v} : v \in V\}$ (one for each vertex in $G$). Finally, the machines are ordered in such a way that $m_{g,u}$ is before $m_{h,v}$ if either (i) $g < h$ or (ii) $g = h$ and $u \in I_k, v \in I_\ell$ with $k > \ell$. The latter case will ensure that, within each group of machines, long-operations of jobs with high frequency will be scheduled before long-operations of jobs with low frequency, a fact that is used to prove Lemma 3.11. For each $f : 1 \leq f \leq d$ and for each vertex $v \in I_f$ there are $r^{2(d-f)}$ groups of jobs, denoted by $J_1^v, J_2^v, \ldots, J_{r^{2(d-f)}}^v$. Each group $J_g^v$ consists of $r^{2f}$ copies, referred to as

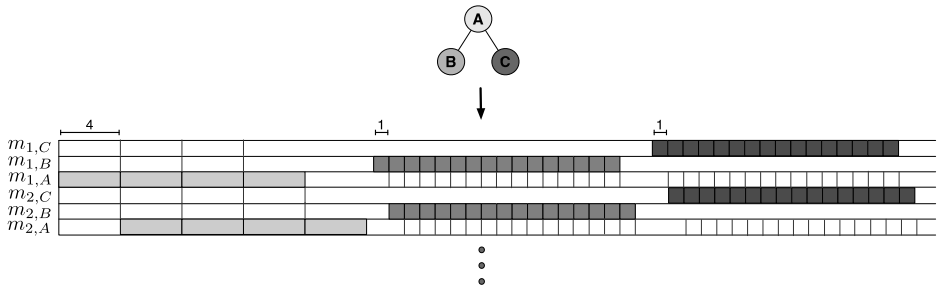


Fig. 7. An example of the reduction with $r = 2, d = 2, I_1 = \{A\}$ and $I_2 = \{B, C\}$. Only the first two out of $r^{2d} = 16$ groups of machines are depicted with the jobs corresponding to $A$,$B$, and $C$ to the left, center, and right, respectively.

$j^v_{g,1}, j^v_{g,2}, \ldots, j^v_{g,r^{2f}}$, of the job that must be processed during $r^{2(d-f)}$ time units on the machines

$$m_{a+1,v}, m_{a+2,v}, \ldots, m_{a+r^{2f},v} \text{ where } a = (g-1) \cdot r^{2f}$$

and during 0 time units on machines corresponding to adjacent vertices, that is, $\{m_{a,u} : 1 \leq a \leq r^{2d}, \{u, v\} \in E\}$ (in an order so that it results in a generalized flow shop instance).

Let $J^v$ be the set of jobs that correspond to the vertex $v$, that is, $J^v = \{j^v_{g,i} : 1 \leq g \leq r^{2(d-f)}, 1 \leq i \leq r^{2f}\}$.

*Remark* 3.6. The construction has $r^{2d}n$ machines and $r^{2d}n$ jobs. Each job has length $r^{2d}$ and each machine has load $r^{2d}$. Hence, lb $= r^{2d}$. Moreover, the number of operations per job is at most $(\Delta + 1)r^{2d}$.

In the subsequent, we will call the operations that require more than 0 time units *long-operations* and the operations that only require 0 time units *short-operations*. For an example of the construction, see Figure 7.

*3.2.2. Completeness.* We prove that if the graph $G$ can be colored with $L$ colors then there is a relatively "short" solution to the general flow shop instance.

LEMMA 3.7. *If $\chi(G) = L$, then there is a schedule of $S(r, d)$ with makespan* lb $\cdot 2L$.

PROOF. We start by showing that all jobs corresponding to nonadjacent vertices can be scheduled within $2 \cdot$ lb time units.

CLAIM 3.8. *Let $IS$ be an independent set of $G$. Then, all the jobs $\bigcup_{v \in IS} J^v$ can be scheduled within* $2 \cdot$ lb *time units.*

PROOF OF CLAIM. Consider the schedule defined by scheduling the jobs corresponding to each vertex $v \in IS$ as follows. Let $I_f$ be the independent set with $v \in I_f$. A job $j^v_{g,i}$ corresponding to vertex $v$ is then scheduled without interruption starting at time $r^{2(d-f)} \cdot (i-1)$.

The schedule has makespan at most $2 \cdot$ lb since a job is started at latest at time $r^{2(d-f)} \cdot (r^{2f} - 1) <$ lb and requires lb time units in total.

To see that the schedule is feasible, observe that no short-operations of the jobs in $\bigcup_{v \in IS} J^v$ need to be processed on the same machines as the long-operations of the jobs in $\bigcup_{v \in IS} J^v$ (this follows from the construction and from the fact that the jobs correspond to nonadjacent vertices). Moreover, two jobs $j^v_{g,i}, j^{v'}_{g',i'}$ with either $g \neq g'$

or $v \neq v'$ have no two long-operations that must be processed on the same machine. Hence, the only jobs that might delay each other are jobs belonging to the same vertex $v$ and the same group $g$, but these jobs are started with appropriate delays (depending on the frequency of the job). □

Assuming $\chi(G) = L$ we partition $V$ into $L$ independent sets $V_1, V_2, \ldots, V_L$. By this claim, the jobs corresponding to each of these independent sets can be scheduled within $2 \cdot \text{lb}$ time units. We can thus schedule the jobs in $L$-"blocks", one block of length $2 \cdot \text{lb}$ for each independent set. The total length of this schedule is $\text{lb} \cdot 2L$. □

*3.2.3. Soundness.* We prove that, given a schedule where many jobs are completed "early", we can find a "big" independent set of $G$, in polynomial time.

LEMMA 3.9. *For any $L \leq r$, given a schedule of $S(r, d)$ where at least half the jobs finish within $\text{lb} \cdot L$ time units, we can, in time polynomial in $n$ and $r^d$, find an independent set of $G$ of size at least $n/(8L)$.*

PROOF. Fix an arbitrarily schedule of $S(r, d)$ where at least half the jobs finish within $\text{lb} \cdot L$ time units. We now disregard the jobs that do not finish within $\text{lb} \cdot L$ time units. Note that the remaining jobs are at least $r^{2d}n/2$ many. As for the gap construction (see Section 2.2), we say that the $i$th long-operation of a job $j$ of frequency $f$ is *good* if the delay $d_j(i)$ between job $j$'s $i$th and $(i + 1)$-th long-operations is at most $\frac{r^2}{4} \cdot r^{2(d-f)}$. In each group $M_g$ of machines, we will associate a set $T_{g,v}$ of time intervals with each vertex $v \in V$. The set $T_{g,v}$ contains the time intervals corresponding to the *first half* of all good long-operations scheduled on the machine $m_{g,v}$. We also let $L(T_{g,v})$ denote the total time units covered by the time intervals in $T_{g,v}$. Scheduling instance $S(r, d)$ has similar structure and similar properties as the gap instances created in Section 2.2. By using the fact that all jobs (that were not disregarded) have completion time at most $L \cdot \text{lb}$, which is by assumption at most $r \cdot \text{lb}$, Lemma 3.10 follows from the same arguments as Lemma 2.2.

LEMMA 3.10. *The fraction of good long-operations of each job is at least $\left(1 - \frac{4}{r}\right)$.*

Consider a group $M_g$ of machines and two jobs corresponding to adjacent vertices that have long-operations on machines in $M_g$. Recall that jobs corresponding to adjacent vertices have different frequencies. By the ordering of the machines, we are guaranteed that the job of higher frequency has, after its long-operation on a machine in $M_g$, a short-operation on the machine in $M_g$ where the job of lower frequency has its long-operation. The following lemma now follows by observing, as in the proof of Lemma 2.3, that the long-operation of the high frequency job can only be good if it is *not* scheduled in parallel with the first half of the long-operation of the low-frequency job.

LEMMA 3.11. *Let $u \in I_k$ and $v \in I_l$ be two adjacent vertices in $G$ with $k > l$. Then, the sets $T_{g,u}$ and $T_{g,v}$, for all $g : 1 \leq g \leq r^{2d}$, contain disjoint time intervals.*

Finally, Lemma 3.12 is proved in the very same way as Lemma 2.4. Their different inequalities arise because in the gap instance we had $d \cdot r^{2d}$ jobs and here we are considering at least $r^{2d}n/2$ jobs that were not disregarded.

LEMMA 3.12. *There exists a $g \in \{1, \ldots, r^{2d}\}$ such that*

$$\sum_{v \in V} L(T_{g,v}) \geq \frac{\text{lb} \cdot n}{8}.$$

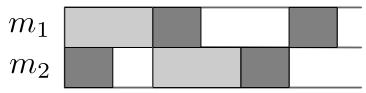


Fig. 8. An example of the representation. The light gray job is defined by $([m_1, 2], [m_2, 2])$ and the dark gray job is defined by $([m_2, 1], [m_1, 1])^2$.

We conclude by a simple averaging argument. Consider a $g$, such that $\sum_{v \in V} L(T_{g,v})$ is at least $\frac{\text{lb} \cdot n}{8}$. This is guaranteed to exist by the lemma above. As all jobs that were not disregarded finish within $L \cdot \text{lb}$ time units, at least $\frac{\text{lb} \cdot n}{8} / (L \cdot \text{lb}) = \frac{n}{8L}$ time intervals must overlap at some point during the first $L \cdot \text{lb}$ time units of the schedule, and, since they overlap, they correspond to different vertices that form an independent set in $G$ (Lemma 3.11). Moreover, we can find such a point in the schedule, for example, by considering all different blocks and in each block verify the start and end points of the time intervals. □

## 4. HARDNESS OF JOB SHOPS WITH FIXED NUMBER OF MACHINES

In this section, we prove Theorem 1.5 and 1.6. We show that problem $J2||C_{\max}$ ($J3|pmtn|C_{\max}$) has no PTAS by presenting a gap-preserving reduction from the NP-hard problem to distinguish between $n$-vertex cubic graphs that have an independent set of size $\beta \cdot n$ and those that have no independent set of size $\alpha \cdot n$, for some $\beta > \alpha$ (see Theorem 1.9). More specifically, given a cubic graph $G(V, E)$, we construct an instance $S$ of $J2||C_{\max}$ ($J3|pmtn|C_{\max}$) so that, for some $L$ defined later, we have the following completeness and soundness analyses.

— *Completeness*. If $G$ has an independent set of size $\beta n$, then $S$ has a schedule with makespan $L$.
— *Soundness*. If $G$ has no independent set of size $\alpha n$, then all schedules of $S$ have makespan at least $(1 + \Omega(1))L$.

Throughout this section, we will use the following notation to define jobs (see Figure 8 for an example). An operation is defined by a pair $[m_i, p]$, where $p$ is the processing time required on machine $m_i$. Let $s_1, \ldots, s_y$ be sequences of operations, and let $(s_1, \ldots, s_y)$ stand for the sequence resulting by their concatenation in the given order. We use $(s_1, \ldots, s_y)^x$ to denote the sequence obtained by repeating $(s_1, \ldots, s_y)$ $x$ times.

Before presenting the reductions and the corresponding analyses, we have the following lemma (whose standard proof is included for the sake of completeness), which will be useful in our constructions.

LEMMA 4.1. *For any sufficiently small fixed $\epsilon > 0$, we can, in time polynomial in $n$, construct a family of sets $\mathcal{C} = \{C_1, C_2, \ldots, C_{n^2}\}$ with the following properties.*

(1) *Each set $C_i \in \mathcal{C}$ is a subset of $\{1, 2, \ldots, (1/\epsilon)^{1/\epsilon} \log n\}$ and has size $\log n$.*
(2) *Two sets $C_i \in \mathcal{C}$ and $C_j \in \mathcal{C}$, with $i \neq j$, satisfy $|C_i \cap C_j| \leq \epsilon \log n$.*

PROOF. Consider the following procedure to obtain such a family $\mathcal{C}$.

(1) Initiate $S$ with all binary strings of length $(1/\epsilon)^{1/\epsilon} \log n$ with $\log n$ many 1's
(2) Let $\mathcal{C} = \emptyset$
(3) REPEAT
    (a) Pick a binary string $x \in S$, and add the set $\{i : x_i = 1\}$ to $\mathcal{C}$
    (b) Remove all the binary strings from $S$ that share at least $\epsilon \log n$ many 1's (elements) with $x$
(4) UNTIL $S$ is empty

It is clear that the family $\mathcal{C}$ returned by this procedure satisfies properties (1) and (2). We continue by analyzing the size of the returned $\mathcal{C}$. We will use that $\binom{a}{b}$ is bounded from above by both $\binom{a}{\lceil a/2 \rceil}$ and $\left(\frac{a \cdot e}{b}\right)^b$, where the latter follows by observing that $\binom{a}{b} \leq \frac{a^b}{b!}$ and $b! \geq (b/e)^b$ from Stirling's approximation. For simplicity, we assume all numbers to be integral. At the $i$th iteration we select a set $C_i$ with $\log n$ many 1's. At that iteration, a string $s$ from $S$ to be removed must have $j \in \{\epsilon \log n, \ldots, \log n\}$ many 1's as in $C_i$ among the $\log n$ 1's available in $s$ (there are at most $\binom{\log n}{j}$ different choices for this), and in the remaining $\log n - j$ places where $s$ has 1's the selected set $C_i$ has 0's (there are at most $\binom{((1/\epsilon)^{1/\epsilon}-1)\log n}{\log n - j}$ different choices for this).

So, at each iteration, the number of binary strings of $S$ that we remove is at most

$$\sum_{j=\epsilon \log n}^{\log n} \binom{\log n}{j} \cdot \binom{((1/\epsilon)^{1/\epsilon} - 1)\log n}{\log n - j} \leq$$

$$(1-\epsilon)\log n \binom{\log n}{\frac{\log n}{2}} \cdot \binom{(1/\epsilon)^{1/\epsilon} \cdot \log n}{(1-\epsilon)\log n} \leq$$

$$(1-\epsilon)\log n \left( (2e)^{\frac{\log n}{2}} \cdot \left( (1/\epsilon)^{1/\epsilon} \cdot \frac{e}{1-\epsilon} \right)^{(1-\epsilon)\log n} \right) \leq$$

$$(1-\epsilon)\log n \left(4e^2 \cdot (1/\epsilon)^{1/\epsilon}\right)^{(1-\epsilon)\log n}.$$

As the number of elements in $S$ at the beginning is $\binom{(1/\epsilon)^{1/\epsilon}\log n}{\log n} \geq (1/\epsilon)^{1/\epsilon \cdot \log n}$, the number of iterations and thus the size of $C$ at the end of the process is at least

$$\frac{(1/\epsilon)^{1/\epsilon \cdot \log n}}{(1-\epsilon)\log n \left(4e^2 \cdot (1/\epsilon)^{1/\epsilon}\right)^{(1-\epsilon)\log n}} =$$

$$\frac{1}{(1-\epsilon)\log n \left((4e^2)^{1-\epsilon}\right)^{\log n}} \cdot \frac{(1/\epsilon)^{1/\epsilon \cdot \log n}}{(1/\epsilon)^{1/\epsilon \log n(1-\epsilon)}} =$$

$$\frac{(1/\epsilon)^{\log n}}{(1-\epsilon)\log n \left((4e^2)^{1-\epsilon}\right)^{\log n}},$$

which is greater than $n^2$ for sufficiently small $\epsilon$. □

### 4.1. The (Non-Preemptive) Two Machines Case

The construction will be presented together with some useful properties that will later be used in the analysis. Before defining the jobs, we will define "types" and "blocks" of operations. The jobs will later be defined as concatenations of blocks, which in turn will be defined as concatenations of types.

*4.1.1. Types.* Let $d = O(\log n)$. For each frequency $f : 1 \leq f \leq d$, we define type $T_f$ and type $\bar{T}_f$ as

$$T_f := \left([m_1, n^{4(d-f)}], [m_2, 0]\right)^{n^{4f}}$$
$$\bar{T}_f := \left([m_2, n^{4(d-f)}], [m_1, 0]\right)^{n^{4f}}.$$

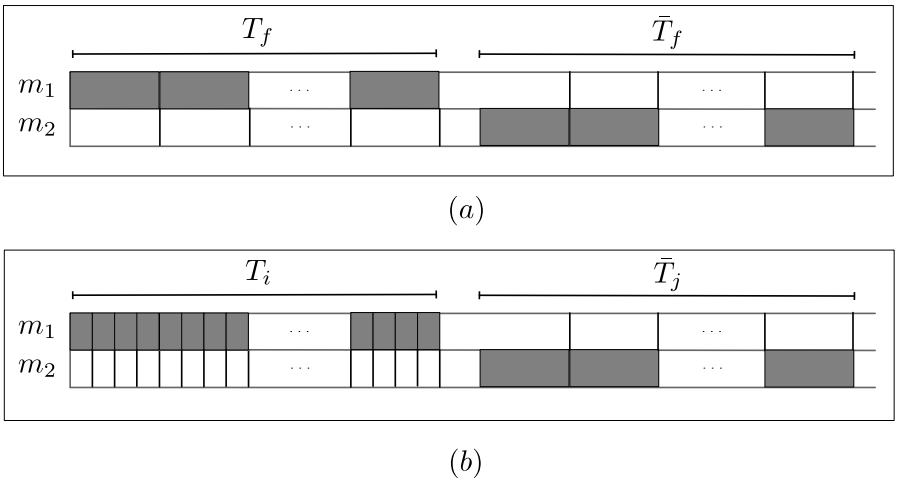


Fig. 9. a) An example of two compatible types $T_f$ and $\bar{T}_f$ that can be scheduled in parallel. b) Two types $T_i, \bar{T}_j$ with $i > j$, for which there are "many" time units during which they cannot be scheduled in parallel.

We will call the operations of $T_f$ and $\bar{T}_f$ that require $n^{4(d-f)}$ time units *long-operations* and the operations that require 0 time units *short-operations*. Note that a type requires time $n^{4f}n^{4(d-f)} = n^{4d}$. We say that the two types $T_f$ and $\bar{T}_f$ are *compatible*, for $f : 1 \leq f \leq d$. Note that two compatible types can be scheduled in parallel, that is, both can be scheduled within $n^{4d}$ time units (see Figure 9(a)). Moreover, we have the following lemma (for intuition see Figure 9(b)).

LEMMA 4.2. *Two types $T_i$ and $\bar{T}_j$ with $i \neq j$ can be scheduled in parallel during at most $n^{4d}/n^4$ time units in any feasible schedule.*

PROOF. If $i > j$, then, as $T_i$ has a short-operation on machine $m_2$ between any two consecutive long-operations on machine $m_1$ at most one long-operation of $T_i$ is scheduled in parallel with any long-operation of $\bar{T}_j$. Since $\bar{T}_j$ has $n^{4j}$ long-operations and each long-operation of $T_i$ requires $n^{4(d-i)}$ time units, it follows, by using $i > j$, that the operations of $\bar{T}_j$ and $T_i$ overlap at most $n^{4(d-1)}$ time units. The same result can be obtained when $i < j$ by using symmetric arguments. □

*4.1.2. Configurations.* For $i = 1, \ldots, |E|$, a *configuration* $C_i = (T_{\pi_{i,1}}, \ldots, T_{\pi_{i,\log n}})$ is an ordered sequence of $\log n$ types, where $\pi_{i,j} \in \{1, \ldots, d\}$ denotes the frequency of the $j$-th type of configuration $C_i$. Lemma 4.1 shows that we can define a set of configurations $\mathcal{C} = \{C_i : i = 1, \ldots, |E|\}$ such that any two configurations $C_i \in \mathcal{C}$ and $C_j \in \mathcal{C}$ with $i \neq j$ have at most $\varepsilon \log n$ types in common, for $\varepsilon > 0$ arbitrarily small. The set $\bar{\mathcal{C}} = \{\bar{C}_i : i = 1, \ldots, |E|\}$ is defined in a similar way by using the types $\bar{T}_i$, that is, for $i = 1, \ldots, |E|$ we have $\bar{C}_i = (\bar{T}_{\pi_{i,1}}, \ldots, \bar{T}_{\pi_{i,\log n}})$. Note that a configuration requires $n^{4d} \log n$ time units.

*4.1.3. Blocks.* We are now ready to define the different blocks. For $i = 1, \ldots, |E|$, *block* $B_i$ is obtained by concatenating $C_i$ for $n^2$-times, that is, $B_i := (C_i)^{n^2}$; similarly, $\bar{B}_i := (\bar{C}_i)^{n^2}$. Let $D = n^{4d+2} \log n$ be the length of a block. As in the case of compatible types, it is easy to see that two blocks $B_i$ and $\bar{B}_i$ can be scheduled in parallel. However, only a tiny fraction of the operations of a configuration $\bar{C}_i$ can overlap the operations of a block $B_j$, if $i \neq j$.

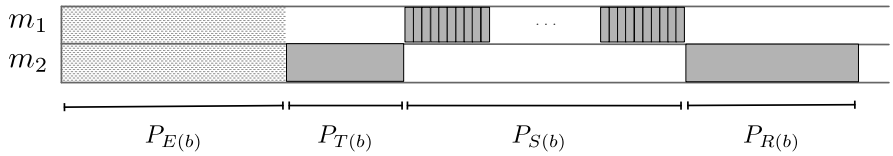


Fig. 10. An overview of $j_b$.

LEMMA 4.3. *The operations of a configuration $\bar{C}_i$ and a block $B_j$, with $i \neq j$, can be scheduled in parallel during at most $\epsilon n^{4d} \log n$ time units in any feasible schedule, where $\epsilon > 0$ can be made an arbitrarily small constant.*

PROOF. The block $B_j$ is composed of $n^2$ repetitions of $C_j$. Note that configuration $\bar{C}_i$ has at most $\epsilon \log n$ compatible types with configuration $C_j$, where $\epsilon > 0$ can be made an arbitrarily small constant. By Lemma 4.2 together with the fact that $B_j$ is a sequence of $n^2 \log n$ types, we have that the remaining $(1-\epsilon)\log n$ types of $\bar{C}_i$ can be scheduled in parallel with $B_j$ during at most $(1-\epsilon)\log n \cdot n^2 \log n \cdot n^{4d}/n^4 = o(n^{4d} \log n)$ time units. Hence, the operations of $\bar{C}_i$ and block $B_j$ can be scheduled in parallel during at most $\epsilon n^{4d} \log n + o(n^{4d} \log n) < \epsilon' n^{4d} \log n$ time units, for some $\epsilon' > 0$ that can be made an arbitrarily small constant. □

*4.1.4. Jobs.* The blocks are now used as building blocks for defining the *jobs*. We will define two kinds of jobs: a big job and vertex jobs. The *big job* $j_b$ is composed of an *edge-part* $P_{E(b)}$, followed by a *tail-part* $P_{T(b)}$, a *slack-part* $P_{S(b)}$, and finally a *remaining-part* $P_{R(b)}$, defined as follows:

$$\begin{aligned} P_{E(b)} &:= (B_1, B_2, \ldots, B_{|E|}) \\ P_{T(b)} &:= [m_2, D \cdot \beta n] \\ P_{S(b)} &:= ([m_1, 1])^{D \cdot 3(1-\beta)n} \\ P_{R(b)} &:= [m_2, D \cdot (1-\beta)n] \end{aligned}$$

Note that the length of job $j_b$ is $L := D(|E| + n + 3(1-\beta)n) = O(nD)$. A high level representation of the long job $j_b$ is given in Figure 10 (for simplicity the structure of the edge-part is omitted; the building blocks of this part have been previously described).

We have a *vertex job* $j_v$ for each vertex $v \in V$. Let $e_i, e_j, e_k$ be the 3 edges incident to $v$ with $i < j < k$. Job $j_v$ is composed of an *edge-part* $P_{E(v)}$ followed by a *tail-part* $P_{T(v)}$ defined as follows:

$$\begin{aligned} P_{E(v)} &:= (\bar{B}_i, \bar{B}_j, \bar{B}_k) \\ P_{T(v)} &:= [m_1, D] \end{aligned}$$

The length of a vertex job is $4D$.

The following fundamental lemma motivates our construction. It shows that for any pair $\{u, v\} \in E$ of adjacent vertices, either $j_u$ or $j_v$ cannot be completed before the end of $j_b$'s tail-part without delaying job $j_b$ $\Omega(D)$ time units. It follows that, without delaying job $j_b$, only jobs corresponding to vertices that form an independent set can be completed before the end of $j_b$'s tail-part.

LEMMA 4.4. *For any $i \in \{1, \ldots, |E|\}$, if there are two copies of block $\bar{B}_i$ to be scheduled, then at least $\gamma \cdot D$ time units of these two blocks cannot be scheduled in parallel with the edge-part of job $j_b$, for some $\gamma < 1$ that can be made arbitrarily close to* 1.

PROOF. Recall that $\bar{B}_i$ is composed of $n^2$ repetitions of the configuration $\bar{C}_i$. Hence, the two copies contain $2n^2$ copies of configuration $\bar{C}_i$. At most $n^2$ of these configurations,

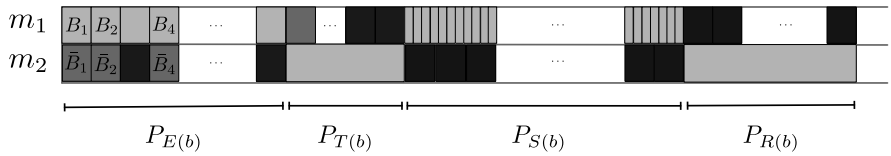


Fig. 11. Structure of a schedule that does not delay the completion time of the big job $j_b$ (depicted in light gray). The vertex jobs are depicted in a darker gray with one of them highlighted (in slightly lighter dark gray). As a block $\bar{B}_j$ of a vertex job only can be scheduled in parallel with either the block $B_j$ or the slack-part of $j_b$, the vertex jobs that complete before the slack-part of $j_b$ form an independent set. From that, it follows that the number of vertex jobs that need to be scheduled in parallel with the slack- and remaining-parts of $j_b$ depends on the size of a maximum independent set of the given graph which yields the desired gap.

that in total require $D$ time units, can be scheduled in parallel with block $B_i$ of job $j_b$'s edge-part. Let $\mathcal{R}$ denote the remaining configurations that may only be scheduled in parallel with the blocks $\{B_j : j \neq i\}$ of job $j_b$'s edge-part. Note that $|\mathcal{R}| \geq n^2$. By Lemma 4.3, we have that a configuration $\bar{C}_i$ can be scheduled in parallel with a block $B_j$, with $i \neq j$, during at most $\epsilon n^{4d} \log n$ time units, for an arbitrarily small constant $\epsilon > 0$. As the edge-part of $j_b$ consists of $|E| = 3n/2$ blocks, $|\mathcal{R}| \geq n^2$, and configurations belonging to a job must be scheduled in a fixed order, almost all configurations in $\mathcal{R}$ are scheduled in parallel with at most one block of job $j_b$'s edge-part. It follows that at most $D + n^2 \cdot n^{4d} \log n\epsilon = D(1+\epsilon)$ time units of the two copies (requiring $2D$ time units in total) can be scheduled in parallel with job $j_b$'s edge-part, where $\epsilon > 0$ can be made an arbitrarily small constant. □

*4.1.5. The Two Machines Analysis.* The intuition of the analysis is depicted in Figure 11 (for simplicity only blocks are depicted and not their operations).

*Completeness.* We will see that if graph $G$ has an independent set of size $\beta n$ then all vertex jobs can be scheduled in parallel with the big job $j_b$. Thus, the makespan of the schedule will equal $L$ (the length of job $j_b$).

Let $V' \subseteq V$ denote an independent set of $G$ with $|V'| = \beta n$. Since $V'$ forms an independent set no two vertices in $V'$ are incident to the same edge. Thus, the edge-parts $P_{E(v)}$ and $P_{E(u)}$ for two different nodes $v, u \in V'$ do not contain any common block. Recall that a block $\bar{B}_i$ can be scheduled in parallel with a block $B_i$, the tail-part of a vertex job requires time $D$ on machine $m_1$ and the tail-part $P_{T(b)}$ of $j_b$ requires time $D\beta n$ on machine $m_2$. It follows that the vertex jobs corresponding to the vertices in $V'$ can all be scheduled in parallel with the edge-part $P_{E(b)} = (B_1, B_2, \ldots, B_{|E|})$ and the tail-part $P_{T(b)}$ of $j_b$. As (i) the slack-part of job $j_b$ consists of $D \cdot 3(1-\beta)n$ unit time operations on $m_1$, (ii) a block $\bar{B}_i$ can be scheduled in parallel with $D$ slack-operations, and (iii) the remaining-part of job $j_b$ consists of one operation on machine $m_2$ that requires time $D(1-\beta)n$, the $(1-\beta)n$ jobs corresponding to the vertices of $V \setminus V'$ can be scheduled in parallel with the slack-part and remaining-part of $j_b$.

*Soundness.* As the makespan equals $L$ — the length of job $j_b$ — in the completeness case, we will analyze the soundness case by showing that there is a fraction of the operations belonging to the vertex jobs that are not scheduled in parallel with $j_b$. Then it follows that if graph $G$ has no independent set of size $\alpha n$ then the length of any schedule is at least $(1 + \Omega(1))L$.

For any given schedule, let $t_1$ be the time at which the tail-part $P_{T(b)}$ of $j_b$ is completed, and $t_2$ the time at which the remaining-part $P_{R(b)}$ of $j_b$ starts. Let $T := n \cdot D$ denote the sum of the lengths of tail-parts of the vertex jobs. Let $\tau_1$,$\tau_2$ and $\tau_3$ be the

fraction of $T$ spent to schedule tail-operations of the vertex jobs during time interval $[0, t_1)$, $[t_1, t_2)$ and $[t_2, \infty)$, respectively.

It is easy to observe that any positive value of $\tau_2$ implies that $\tau_2 \cdot T$ time units are not scheduled in parallel with job $j_b$. Similarly a positive value of $\tau_3$ implies that $\max\{0, (\tau_3 - (1-\beta))T\}$ time units are not scheduled in parallel with $j_b$. Finally, note that there are at least $\tau_1 \cdot n$ vertex jobs that complete their edge-part before time $t_1$. Since $G$ has no independent set of size $\alpha n$, it follows that there are at least $\max\{0, (\tau_1 - \alpha)n\}$ conflicting pairs of vertex jobs (i.e., corresponding to adjacent pairs of vertices). There are thus two "conflicting" copies of $\max\{0, (\tau_1 - \alpha)n\}$ different blocks from $\{\bar{B}_i : i = 1, \ldots, |E|\}$ to be scheduled before time $t_1$. By using Lemma 4.4, we can easily check that at least $(\tau_1 - \alpha)n \cdot \gamma \cdot D$ time units of these conflicting blocks cannot be scheduled in parallel with job $j_b$.

By these arguments, it follows that the makespan of the schedule is at least the length of job $j_b$ plus $(\beta - \alpha)\gamma \cdot n \cdot D$, where $\gamma < 1$ can be made arbitrarily close to 1. Hence, as $L = O(nD)$, the length of any schedule in the soundness case is at least $(1 + \Omega(1))L$.

### 4.2. The Preemptive Three Machines Case

In the following we consider the preemptive three machines case and prove Theorem 1.6. We show that problem $J3|pmtn|C_{\max}$ has no PTAS by presenting a gap-preserving reduction from the NP-hard problem to distinguish between $n$-vertex cubic graphs that have an independent set of size $\beta \cdot n$ and those that have no independent set of size $\alpha \cdot n$, for some $\beta > \alpha$ (see Theorem 1.9). In the reduction, the nodes of the cubic graph are mapped to jobs that are constructed as a concatenation of "well-structured" sequences. The definition and the analysis of these sequences are the arguments of the next subsections. The jobs will be defined subsequently.

In the following, we will use several times a well-known property that ensures that there exists an optimal schedule for the preemptive job shop problem where preemptions occur at integral times [Bansal et al. 2006].

*4.2.1. Sequences.* Let $d = O(\log n)$, for each frequency $f = 1, \ldots, d$, we define four basic *sequences* of operations:

$$S_{x,f} := \left([m_x, n^{4(d-f+1)}], [m_{x+1}, n^{4(d-f+1)}]\right)^{n^{4f}}$$
$$\bar{S}_{x,f} := \left([m_{x+1}, n^{4(d-f+1)}], [m_x, n^{4(d-f+1)}]\right)^{n^{4f}}$$

for $x \in \{1, 2\}$.

Note that all sequences have the same length, $\ell_s := 2n^{4(d+1)}$, and $S_{x,f}$ can be scheduled in parallel with $\bar{S}_{x,f}$, for $x \in \{1, 2\}$ and $f = 1, \ldots, d$.

The following lemma will be used several times. Informally, it says that if for "many" time units a sequence is scheduled in parallel with a lower frequency sequence, then the completion time of the latter is larger than its length by "many" time units. Figure 12(b) shows that only a "small" fraction of any given sequence can be scheduled in parallel with another sequence having lower frequency, and without delaying the latter; Figure 12(c) shows that any additional time unit spent in parallel would delay the lower frequency sequence by the same amount).

LEMMA 4.5. *For any $j > i$, consider a schedule for $\bar{S}_{x,j}$ and $S_{x,i}$, where $x \in \{1, 2\}$. If there are $t$ time units where sequences $\bar{S}_{x,j}$ and $S_{x,i}$ are scheduled in parallel, then the time required to complete $S_{x,i}$ is at least $\ell_s + t - o\left(\ell_s/n^3\right)$.*

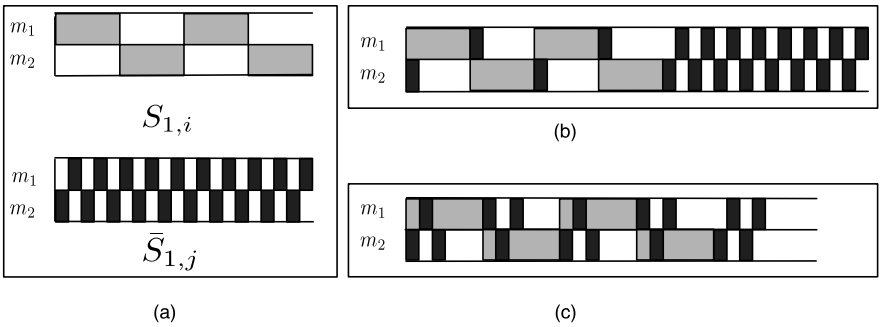


Fig. 12. Two sequences with different frequencies with $j > i$.

PROOF. Consider a single operation $O$ of $S_{x,i}$ that requires $n^{4(d-i+1)}$ time units on machine $m_x$. As $\bar{S}_{x,j}$ has an operation on machine $m_x$ between any two consequent operations on machine $m_{x+1}$, at most one operation of $\bar{S}_{x,j}$ (that requires $n^{4(d-j+1)}$ time units) can overlap $O$ without interrupting it. Moreover, if, for some $c > 1$, $c \cdot n^{4(d-j+1)}$ time units of $\bar{S}_{x,j}$ is scheduled in parallel with $O$, then operation $O$ must have been interrupted by at least $\lceil \frac{c \cdot n^{4(d-j+1)}}{n^{4(d-j+1)}} \rceil - 1 = \lceil c - 1 \rceil$ operations of $\bar{S}_{x,j}$, that in total requires $\lceil c-1 \rceil \cdot n^{4(d-j+1)}$ additional time units. As $S_{x,i}$ has $2n^{4i}$ operations in total and $j > i$, $\bar{S}_{x,j}$ can be scheduled in parallel with $S_{x,i}$ during at most $2n^{4i} \cdot n^{4(d-j+1)} \leq \ell_s/n^4 = o(\ell_s/n^3)$ time units without delaying $S_{x,i}$ and any additional time units spent in parallel will delay $S_{x,i}$ with at least the same amount. □

*4.2.2. Types.* For each $f = 1, \ldots d$, we define *type* $T_f$ and $\bar{T}_f$ as follows:

$$T_f := \left(S_{1,f}, S_{2,(d-f+1)}\right)^{n^4}$$
$$\bar{T}_f := \left(\bar{S}_{1,f}, \bar{S}_{2,(d-f+1)}\right)^{n^4}$$

Additionally, we define an extra type $T_0$ as a sequence of $2n^4$ "long" operations on machines $m_3$ and $m_1$ (this type will be part of the long job $j_b$ defined later):

$$T_0 := \left(S_{3,0}, S_{1,0}\right)^{n^4}$$
$$S_{3,0} := [m_3, \ell_s]$$
$$S_{1,0} := [m_1, \ell_s]$$

The total length of a type is $\ell_T := 2n^4\ell_s$. Note that the operations of $T_0$ can be scheduled in parallel with those of $T_f$ and $\bar{T}_f$, for $f = 1, \ldots, d$, and completed by time $\ell_T$.

We additionally observe that if $T_0$ and $T_f$ ($\bar{T}_f$) are completed by time $\ell_T$, then at any point of time interval $[0, \ell_T]$, either an occurrence of $S_{1,0}$, or $S_{1,f}$ (or $\bar{S}_{1,f}$) are processed (similarly, $S_{3,0}$, or $S_{2,(d-f+1)}$ (or $\bar{S}_{2,(d-f+1)}$)). Moreover, this structural property cannot be violated "many" times without increasing the completion time of either $T_0$, or $T_f$ ($\bar{T}_f$), by "many" time units.

*Definition* 4.6. For any given $T_f$ ($\bar{T}_f$) with $f = 1, \ldots, d$, we say that $T_f$ ( $\bar{T}_f$) *is not alternated with* $T_0$ *for* $t$ *time units* if at least one of the following two situations happens.

(1) There are $t$ units where neither an occurrence of $S_{1,0}$ nor an occurrence of $S_{1,f}$ ($\bar{S}_{1,f}$) is processed.

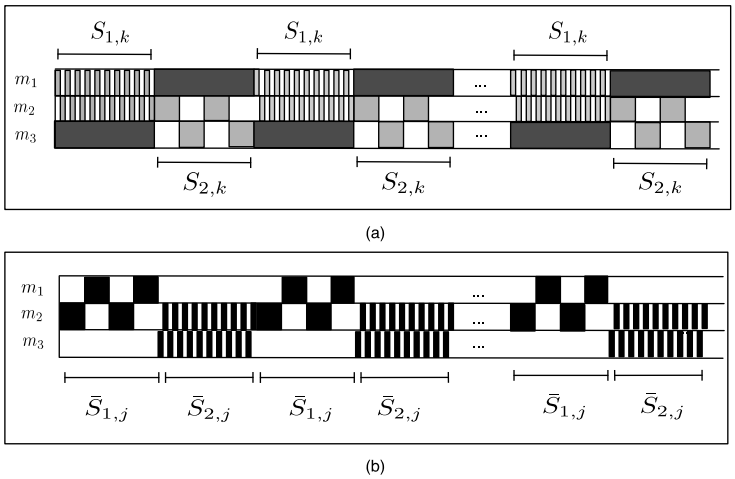


Fig. 13. Different types. The dark gray operations in (a) belong to type $T_0$ and can be scheduled in parallel with type $T_k$. In (b), the operations of $\bar{T}_j$ are depicted and Lemma 4.8 says that these operations cannot be scheduled in parallel with types $T_k$ and $T_0$, depicted in (a), without delaying one of them.

(2) There are $t$ units where neither an occurrence of $S_{3,0}$ nor an occurrence of $S_{2,(d-f+1)}$ ($\bar{S}_{2,(d-f+1)}$) is processed.

LEMMA 4.7. *Consider any feasible schedule for $T_0$ and $T_f$ ($\bar{T}_f$) and let $C_{max}$ be the completion time of the last operation. If for $t$ units within time interval $[0, C_{max}]$ $T_f$ ($\bar{T}_f$) is not alternated with $T_0$, then $C_{max} \geq \ell_T + \Omega(t) - o(\ell_T/n^3)$.*

PROOF. We provide the proof when $T_f$ is not alternated with $T_0$ for $t$ time units because there are $t$ units where, neither an occurrence of $S_{1,0}$, nor an occurrence of $S_{1,f}$ is processed (the other cases are similar).

Let $t_1$ be the total time units where either an occurrence of $S_{1,0}$ or an occurrence of $S_{1,f}$, but not both, is executed. Let $t_2$ be the total time units where occurrences of both, $S_{1,0}$ and $S_{1,f}$, are executed in parallel. Note that $\ell_T = t_1 + 2 \cdot t_2$, and therefore

$$C_{max} \geq t + t_1 + t_2 = t + \ell_T - t_2. \tag{1}$$

If, for some $c > 1$, $c \cdot n^{4(d-f+1)}$ time units of an occurrence $S_{1,f}$ is scheduled in parallel with an occurrence $O$ of $S_{1,0}$, then operation $O$ must have been interrupted by at least $\lceil \frac{c \cdot n^{4(d-f+1)}}{n^{4(d-f+1)}} \rceil - 1 = \lceil c - 1 \rceil$ operations of $S_{1,f}$; This situation increases the completion time of $T_0$ by $\lceil c - 1 \rceil \cdot n^{4(d-f+1)}$ additional time units. As $T_0$ has $n^4$ occurrences of $S_{1,0}$, $S_{1,f}$ can be scheduled in parallel with the occurrences of $S_{1,0}$ during at most $n^4 \cdot n^{4(d-f+1)} \leq n^{4(d+1)} = \ell_s/2 = o(\ell_T/n^3)$ time units without delaying $T_0$, and any additional time units spent in parallel will delay $T_0$ by at least the same amount. The latter means that

$$C_{max} \geq \ell_T + t_2 - o(\ell_T/n^3). \tag{2}$$

By summing inequalities (1) and (2), we conclude that $C_{max} \geq \ell_T + t/2 - o(\ell_T/n^3)$, and the claim follows. □

The following lemma shows that only a small fraction of a type $\bar{T}_j$ can be scheduled in parallel with $T_k$ without delaying the completion time of $T_0$ or $T_k$, whenever $j \neq k$. For an intuition see Figure 13.

LEMMA 4.8. *For $j \neq k$, consider a schedule of $\bar{T}_j$, $T_k$, and $T_0$. If $\bar{T}_j$ is scheduled in parallel with $T_k$ during $t$ time units then the completion time of either $T_k$ or $T_0$ is at least $\ell_T + \Omega(t) - o\left(\ell_T/n^3\right)$.*

PROOF. Recall that $\bar{T}_j = \left(\bar{S}_{1,j}, \bar{S}_{2,(d-j+1)}\right)^{n^4}$ and $T_k = \left(S_{1,k}, \bar{S}_{2,(d-k+1)}\right)^{n^4}$. We have that either $j > k$ or $d - j + 1 > d - k + 1$. We assume the former inequality (the other case is symmetric).

By Lemma 4.7, we assume that the amount of time units $t_0$ where neither $S_{1,0}$, nor $S_{1,k}$ are processed is "much smaller" than $t$, namely $t_0 = o(t)$, otherwise, the claim follows.

Then, by Lemma 4.5, a sequence $\bar{S}_{1,j}$ can be scheduled in parallel with a sequence $S_{1,k}$ during at most $o(\ell_s/n^3)$ time units without increasing the time needed to complete $S_{1,k}$ and any additional time units spent in parallel increases the time needed of the same amount. Since $T_k$ has $n^4$ occurrences of $S_{1,k}$ we have that if the occurrences of $\bar{S}_{i,j}$ are scheduled in parallel with the occurrences of $S_{1,k}$, during $t$ time units, then $T_k$ has completion time that is smaller than its length plus $t$, namely $\ell_t + t$, by at most $o(\frac{\ell_s}{n^3} \cdot n^4)$ units. It follows that $T_k$ has completion time at least $\ell_T + \Omega(t) - o(n \cdot \ell_s) = \ell_T + \Omega(t) - o(\ell_T/n^3)$ (recall that $\ell_T = 2n^4\ell_s$).

Moreover by a similar argument, if the occurrences of $\bar{S}_{1,j}$ are scheduled in parallel with the occurrences $S_{1,0}$, during $t$ time units, then, during at least $\Omega(t) - o(\ell_T/n^3)$ time units $T_0$ cannot be scheduled.

The statement now follows by observing that every second sequence of $\bar{T}_j$ is an occurrence of $\bar{S}_{1,j}$ and hence if $\bar{T}_j$ is scheduled in parallel with $T_k$ during $t > \ell_T/n^4$ time units, then $\Omega(t)$ time units of the occurrences of $\bar{S}_{1,j}$ must be scheduled. By these arguments, this would cause either $T_0$ or $T_k$ to be interrupted during at least $\Omega(t) - o(\ell_T/n^3)$ time units. □

*4.2.3. Configurations and Blocks.* For $i = 1, \ldots, |E|$, a *configuration* $C_i = (T_{\pi_{i,1}}, \ldots, T_{\pi_{i,\log n}})$ is an ordered sequence of $\log n$ types, where $\pi_{i,j} \in \{1, \ldots, d\}$ denotes the frequency of the $j$th type of configuration $C_i$. Lemma 4.1 shows that we can define a set of configurations $\mathcal{C} = \{C_i : i = 1, \ldots, |E|\}$ such that any two configurations $C_i \in \mathcal{C}$ and $C_j \in \mathcal{C}$ with $i \neq j$ have at most $\varepsilon \log n$ types in common, for $\varepsilon > 0$ arbitrarily small. The set $\bar{\mathcal{C}} = \{\bar{C}_i : i = 1, \ldots, |E|\}$ is defined in a similar way by using the types $\bar{T}_i$, that is, for $i = 1, \ldots, |E|$, we have $\bar{C}_i = (\bar{T}_{\pi_{i,1}}, \ldots, \bar{T}_{\pi_{i,\log n}})$

We are now ready to define the different *blocks*. For $i = 1, \ldots, |E|$, block $B_i$ is obtained by concatenating $C_i$ for $n^2$-times, that is, $B_i := (C_i)^{n^2}$; similarly $\bar{B}_i := (\bar{C}_i)^{n^2}$. Additionally, we define an extra configuration $C_0 := (T_0)^{\log n}$ and a corresponding block $B_0 := (C_0)^{n^2}$.

The length of any block is $D := \ell_T \cdot n^2 \log n$. Moreover, the operations of $B_0$ can always be scheduled in parallel with those of $B_i$ and $\bar{B}_i$, for $i = 1, \ldots, |E|$. Therefore, the blocks $B_0$, $B_i$, and $\bar{B}_i$, for $i = 1, \ldots, |E|$ can be completed by time $D$. However, only a small fraction of a block $\bar{B}_i$ can be scheduled in parallel with $B_j$ without delaying the completion time of $B_0$ or $B_j$, whenever $i \neq j$.

LEMMA 4.9. *For $i \neq j$, consider a schedule of $\bar{B}_i$, $B_j$, and $B_0$. If $\bar{B}_i$ is scheduled in parallel with $B_j$ during $t$ time units, then the completion time of either $B_j$ or $B_0$ is at least $D + \Omega(t) - o(D/n^{1.5})$.*

PROOF. The block $\bar{B}_i$ is composed of $n^2$ repetitions of $\bar{C}_i$ (consisting of $\log n$ types) and the block $B_j$ is composed of $n^2$ repetitions of $C_j$. Moreover, out of $\bar{B}_i$'s sequence

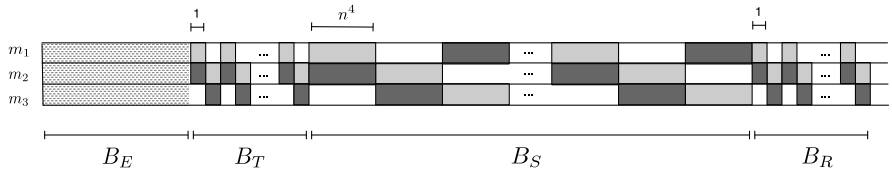


Fig. 14. Long jobs.

of $n^2 \log n$ types there is a subset $A$ of at least $(1-\epsilon)n^2 \log n$ types so that if $\bar{T}_k \in A$, then the sequence $B_i$ has no occurrence of $T_k$. By Lemma 4.8 together with that $B_j$ is a sequence of $n^2 \log n$ types and $B_0$ is $n^2 \log n$ repetitions of $T_0$, we have that if $t$ time units of the types in $A$ are scheduled in parallel with $B_j$ then either $B_j$ or $B_0$ has completion time at least $D + \Omega(t) - o(n^2 \log n \cdot \ell_T / n^3) = D + \Omega(t) - o(D/n^3)$ (recall that $D = \ell_T n^2 \log n$). The statement now follows from observing that if $t > D/n^{1.5}$ time units of $\bar{B}_i$ have been scheduled then $\Omega(t)$ time units of the types in $A$ have been scheduled. □

The remaining part of the construction has several similarities with the 2-machines case.

*4.2.4. Jobs.* There are two *long jobs*, $j_a$ and $j_b$. Job $j_a$ is composed of an *edge-part* $P_{E(a)}$, followed by a *tail-part* $P_{T(a)}$, a *slack-part* $P_{S(a)}$, and finally a *remaining-part* $P_{R(a)}$ defined as follows:

$$\begin{aligned}
P_{E(a)} &:= \big(B_1, B_2, \ldots, B_{|E|}\big) \\
P_{T(a)} &:= \big([m_1, 1], [m_2, 1]\big)^{\frac{\beta n D}{2}} \\
P_{S(a)} &:= \big([m_1, n^4], [m_2, n^4], [m_3, n^4]\big)^{\frac{3(1-\beta)nD}{n^4}} \\
P_{R(a)} &:= \big([m_1, 1], [m_2, 1]\big)^{\frac{(1-\beta)nD}{2}}.
\end{aligned}$$

Similarly, job $j_b$ is composed of an *edge-part* $P_{E(b)}$, followed by a *tail-part* $P_{T(b)}$, a *slack-part* $P_{S(b)}$, and finally a *remaining-part* $P_{R(b)}$ defined as follows:

$$\begin{aligned}
P_{E(b)} &:= (B_0)^{|E|} \\
P_{T(b)} &:= \big([m_2, 1], [m_3, 1]\big)^{\frac{\beta n D}{2}} \\
P_{S(b)} &:= \big([m_2, n^4], [m_3, n^4], [m_1, n^4]\big)^{\frac{3(1-\beta)nD}{n^4}} \\
P_{R(b)} &:= \big([m_2, 1], [m_3, 1]\big)^{\frac{(1-\beta)nD}{2}}.
\end{aligned}$$

In Section 4.2.3 we observed that the operations of $B_0$ can always be scheduled in parallel with those of $B_i$, for $i = 1, \ldots, |E|$. Thus, the edge-parts $P_{E(a)}$ and $P_{E(b)}$ of jobs $j_a$ and $j_b$ can be scheduled in parallel. Clearly, the same applies to the other parts. It follows that long jobs can be scheduled in parallel and the length of both is

$$L := \big(|E| + n + 9(1-\beta)n\big)\, D = O(nD). \tag{3}$$

A high level representation of the long jobs is given in Figure 14, where the operations of $j_a$ and $j_b$ are colored in white and gray, respectively (for simplicity, the structure of the edge part is omitted, the structure of this part is discussed in Section 4.2.3).

We have a *vertex job* $j_v$ for each vertex $v \in V$. Let $e_i, e_j, e_k$ be the three edges incident to $v$ with $i < j < k$. Job $j_v$ is composed of an *edge-part* $P_{E(v)}$ followed by a *tail-part* $P_{T(v)}$ defined as follows:

$$P_{E(v)} := \left(\bar{B}_i, \bar{B}_j, \bar{B}_k\right)$$
$$P_{T(v)} := \left([m_3, 1], [m_1, 1]\right)^{\frac{D}{2}}$$

The length of any vertex job is $4D$.

The following fundamental lemma motivates our construction. It shows that the jobs corresponding to adjacent vertices cannot be completed before the end of $j_a$'s tail-part in the schedule without delaying either $j_a$ or $j_b$. It follows that, without delaying job $j_a$ or $j_b$, only jobs corresponding to vertices that form an independent set can be completed before job $j_a$'s tail-part.

LEMMA 4.10. *For $I \subseteq \{1, \ldots, |E|\}$, consider a schedule of two copies of each block in $\{\bar{B}_i\}_{i\in I}$ together with $j_a$ and $j_b$. If the copies of $\{\bar{B}_i\}_{i\in I}$ are scheduled in parallel with $P_{E(a)}$ during $|I| \cdot D + t$ time units, then the completion time of either $j_a$ or $j_b$ is at least $L + \Omega(t) - o(D)$.*

PROOF. Consider a fixed $i \in I$. Clearly, as block $B_i$ requires $D$ time units, we can schedule at most $D$ time units of the two blocks $\bar{B}_i$ in parallel with block $B_i$. By Lemma 4.9, if $t$ time units of $\bar{B}_i$ is scheduled in parallel with any $B_h$, for $h = 1, \ldots, |E|$ and $h \neq i$, then either $B_h$ (belonging to $j_a$) or a block $B_0$ (belonging to $j_b$) is interrupted during at least $\Omega(t) - o(D/n^{1.5})$ time units. It follows that if $D + t$ time units of the two copies of $\bar{B}_i$ are scheduled in parallel with $P_{E(a)}$ then, as $P_{E(a)}$ consists of $3n/2$ blocks, either job $j_a$ or job $j_b$ are interrupted during $\Omega(t) - o(3n/2 \cdot D/n^{1.5}) = \Omega(t) - o(D)$ time units.

By observing that the copies of $\{\bar{B}_i\}_{i\in I}$ can be scheduled in parallel during very few time units, we have that if $|I| \cdot D + t$ time units of the copies of $\{\bar{B}_i\}_{i\in I}$ are scheduled in parallel with $P_{E(a)}$, then either job $j_a$ or $j_b$ are interrupted during $\Omega(t) - o(D)$ time units. □

#### *4.2.5. The Three Machines Unit Time Analysis.*

*Completeness.* We will see that if graph $G$ has an independent set of size $\beta n$ then all vertex jobs can be scheduled in parallel with the long jobs. The makespan of the resulting schedule will be equal to (3).

Long jobs are scheduled to complete at time $L$ (3) (see, e.g., Figure 14). Let $V' \subseteq V$ denote an independent set of $G$ with $|V'| = \beta n$. Since $V'$ forms an independent set no two vertices in $V'$ are incident to the same edge.

For $i = 1, \ldots, |E|$, observe that block $\bar{B}_i$ can be scheduled in parallel with block $B_i$. Moreover, we can schedule all the sequences from $\{P_{T(v)} : v \in V'\}$ in parallel with $P_{T(a)}$ and $P_{T(b)}$. It follows that all the vertex jobs corresponding to the vertices in $V'$ can be scheduled in parallel with the operations of long jobs from blocks $P_{E(a)}, P_{T(a)}, P_{E(b)}, P_{T(b)}$.

The operations from $\{P_{E(v)} : v \in V \setminus V'\}$ can be scheduled in parallel with the operations from $P_{S(a)}, P_{S(b)}$ without delaying or interrupting the execution of the slack parts of the long jobs (only the vertex job is preempted). Indeed, the total length of the operations in $\{P_{E(v)} : v \in V \setminus V'\}$ is at most $3(1-\beta)nD$, and each of these operations has length at least $n^4$. Moreover, the length of the slack-part of the long jobs is $9(1-\beta)nD$ and note that in the slack-part of the long jobs, during every $3n^4$ time units that are spent to process operations from $P_{S(a)}, P_{S(b)}$, we can schedule at least $n^4$ time units of an unscheduled operation from $\{P_{E(v)} : v \in V \setminus V'\}$, that is, $n^4$ time units are contiguously available on each machine (see Figure 14).

Finally, the remaining operations from the $(1-\beta)n$ tail-parts $\{P_{T(v)} : v \in V \setminus V'\}$, that in total require $(1-\beta)nD$ time units, can be scheduled in parallel with $P_{R(a)}$ and $P_{R(b)}$ (recall that both $P_{R(a)}$ and $P_{R(b)}$ require $(1-\beta)nD$ time units).

*Soundness.* We show the existence of a gap by proving that if the graph has no independent set of size $\alpha n$ then the length of any schedule is at least $L+\Omega(nD) = (1+\Omega(1))L$.

For any given schedule, let $t_1$ be the time at which operations from blocks $P_{T(a)}$ and $P_{T(b)}$ are completed. Let $t_2$ be the time at which the first operation from $P_{R(a)}$ and $P_{R(b)}$ starts. Let $\tau_1$,$\tau_2$ and $\tau_3$ be the fraction of $T = n \cdot D$ spent to schedule operations in $\{P_{T(v)} : v \in V\}$ during time interval $[0, t_1)$, $[t_1, t_2)$ and $[t_2, \infty)$, respectively. We consider the following cases.

*Case* 1. $\tau_1 = \alpha + c$ where $c = \Omega(1)$ is a positive constant.
*Case* 2. $\tau_2 = \Omega(1)$ is a positive constant.
*Case* 3. $c = o(1)$ and $\tau_2 = o(1)$.

For any $0 \le \tau_1 \le 1$, note that there are at least $\tau_1 \cdot n$ vertex jobs that complete their edge part before time $t_1$. In (Case 1), since $G$ has no independent set of size $\alpha n$, there are at least $(\tau_1 - \alpha)n = c \cdot n = \Omega(n)$ conflicting pairs of jobs (i.e., corresponding to edges of adjacent pairs of vertices). It follows that there are two "conflicting" copies of $\Omega(n)$ different blocks from $\{\bar{B}_i : i = 1, \ldots, |E|\}$ to be scheduled before time $t_1$. Let $t_0$ be the time the last operation from blocks $P_{E(a)}$ and $P_{E(b)}$ is completed. Observe that $t_0$ cannot be larger than $|E|D + o(nD)$, otherwise we would be done since this would create a schedule of length $L + \Omega(nD)$. For the same reason, we can assume that $t_1 = L_1 + o(nD)$, where $L_1$ denotes the total length of the sequence $(P_{E(a)}, P_{T(a)})$. It follows that in time interval $[t_0, t_1]$, machine $m_2$ is occupied for $t_1 - t_0 - o(nD)$ time units, by operations of the tail-part of long jobs. By these observations and by using Lemma 4.10, we have that at least $\Omega(nD)$ time units of these conflicting blocks cannot be scheduled within time $t_1$ without creating a schedule of length $L + \Omega(nD)$.

Now, let us consider (Case 2). Remember that we can assume that $t_1 = L_1 + o(nD)$, otherwise we are done. The latter assumption implies that "almost" all operations from $\{P_{S(a)}, P_{S(b)}\}$, that is, all but $o(nD)$, are not yet scheduled at time $t_1$. Let $T_{12}$ be the set of operations from $\{P_{T(v)} : v \in V\}$ that are scheduled during $[t_1, t_2)$. Under (Case 2), we have that $t_2 - t_1 = \Omega(nD)$. Moreover, we can assume that with the exception of $o(nD)$ time units, at every time point in interval $[t_1, t_2]$ an operation from $\{P_{S(a)}, P_{S(b)}\}$ is executed. Note that sequences from $\{P_{T(v)} : v \in V\}$ have higher frequency than those in $\{P_{S(a)}, P_{S(b)}\}$. By using similar arguments as in the proof of Lemma 4.5, it is easy to check that without delaying the long jobs we can only schedule a tiny fraction of $T_{12}$, that is, at most $o(nD)$ time units. Moreover, additional time units of operations from $T_{12}$ can be processed in parallel with a long job only if a long job is delayed by at least the same amount. Under Case 2, this implies a schedule of length $L + \Omega(nD)$.

Finally, in Case 3, we have $\tau_1 + \tau_2 = \alpha + o(1)$ and therefore $\tau_3 = 1 - \alpha - o(1) > 1 - \beta$. This means that we have to schedule $(\beta - \alpha - o(1))nD$ time units more than the space available to finish by time (3) (recall that the lengths of $P_{R(a)}$ and $P_{R(b)}$ are both $(1-\beta)nD$). This creates a schedule of length $L + \Omega(nD)$.

## 5. CONCLUSIONS

Schuurman and Woeginger [1999] highlighted the poor understanding of the approximability of job shop and flow shop scheduling as two of the ten most prominent open problems in the area of approximation algorithms for NP-hard machine scheduling problems.

In this article, we have resolved many of these questions by using strong hardness results for coloring by Khot [2001] together with novel "gap" constructions that build

upon previous work by Feige and Scheideler [2002]. The main results of our work can be summarized as follows.

(1) The $O((\log \mathrm{lb})^{1+\epsilon})$-approximation algorithm [Czumaj and Scheideler 2000; Feige and Scheideler 2002], where $\epsilon > 0$ can be made arbitrarily close to 0, for acyclic job shops and generalized flow shops is essentially the best possible.
(2) To improve the approximation guarantee for flow shops, one needs to (i) improve the polynomial time computable lower bound on the optimal makespan and (ii) use the fact that all jobs are processed on every machine.
(3) It is necessary to restrict both the machines *and* the number of operations per job to obtain a PTAS for the job shop problem with makespan objective. That it is sufficient follows from the work by Jansen et al. [2003].

With our current techniques we have been unable to address some shop scheduling problems, whose approximability remains poorly understood. In this article, we list three prominent problems, all of them with a significant "gap" between the best-known approximation guarantees and inapproximability results.

(1) The job shop problem admits an $O((\log \mathrm{lb})^2/(\log\log \mathrm{lb})^2)$-approximation algorithm [Goldberg et al. 2001]. Our results imply that it is unlikely to approximate job shops within a factor $O((\log \mathrm{lb})^{1-\epsilon})$, for any $\epsilon > 0$. To the best of our knowledge no instances of the job shop problem are known with optimal makespan a $\omega(\log \mathrm{lb})$ factor away from the lower bound lb. This leaves open the possibility that job shop scheduling might have an $O(\log \mathrm{lb})$-approximation algorithm.
(2) Job shop scheduling with preemption admits an $O(\log m/\log\log m)$-approximation algorithm [Bansal et al. 2006] and preemptive acyclic job shops admits an $O(\log\log \mathrm{lb})$-approximation algorithm [Feige and Scheideler 2002], which is currently also the algorithm of choice for flow shops with preemption. The only negative result [Williamson et al. 1997], says that it is NP-hard to approximate these problems within a factor less than 5/4. Moreover, it is still open whether or not the preemptive job shop problem with two-machines ($J2|pmtn|C_{max}$) admits a PTAS.
(3) The flow shop problem has an $O((\log \mathrm{lb})^{1+\epsilon})$-approximation algorithm, where $\epsilon > 0$ can be made arbitrarily close to 0 [Czumaj and Scheideler 2000; Feige and Scheideler 2002]. On the other hand, it is only known that it is NP-hard to approximate flow shops within a factor less than 5/4 [Williamson et al. 1997]. This leaves open the possibility that one can use the fact that all jobs have to be processed on every machine to even obtain a constant factor approximation algorithm for flow shop scheduling.

## ACKNOWLEDGMENTS

We are grateful to Maxim Sviridenko who introduced us to the problem and gave useful references. Also, many thanks to Subhash Khot for a kind explanation of his results. M. Mastrolilli would like to dedicate this work to Edoardo on the occasion of his birth.

## REFERENCES

ALIMONTI, P. AND KANN, V. 2000. Some APX-completeness results for cubic graphs. *Theoret. Comput. Science 237*, 1-2, 123–134.

ANDERSON, E. J., JAYRAM, T. S., AND KIMBREL, T. 2001. Tighter bounds on preemptive job shop scheduling with two machines. *Computing 67*, 1, 83–90.

BANSAL, N., KIMBREL, T., AND SVIRIDENKO, M. 2006. Job shop scheduling with unit processing times. *Math. Oper. Res. 31*, 381–389.

BECK, J. 1991. An algorithmic approach to the lovasz local lemma. *Random Struct. Algor. 2*, 4, 343–365.

CHEN, B., POTTS, C. N., AND WOEGINGER, G. J. 1998. A review of machine scheduling: Complexity, algorithms and approximability. *Hand. Combin. Optimiz. 3*, 21–169.

CZUMAJ, A. AND SCHEIDELER, C. 2000. A new algorithm approach to the general lovász local lemma with applications to scheduling and satisfiability problems (extended abstract). In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC)*. 38–47.

FEIGE, U. AND SCHEIDELER, C. 2002. Improved bounds for acyclic job shop scheduling. *Combinatorica 22*, 3, 361–399.

FISHKIN, A., JANSEN, K., AND MASTROLILLI, M. 2003. On minimizing average weighted completion time: A PTAS for the job shop problem with release dates. In *Proceedings of the 14th Annual International Symposium on Algorithms and Computation (ISAAC'03)*. Lecture Notes in Computer Science, vol. 2906, Springer, 319–328.

FISHKIN, A. V., JANSEN, K., AND MASTROLILLI, M. 2008. Grouping techniques for scheduling problems: Simpler and faster. *Algorithmica 51*, 2, 183–199.

GOLDBERG, L. A., PATERSON, M., SRINIVASAN, A., AND SWEEDYK, E. 2001. Better approximation guarantees for job-shop scheduling. *SIAM J. Disc. Math. 14*, 1, 67–92.

GRAHAM, R., LAWLER, E., LENSTRA, J., AND KAN, A. R. 1979. Optimization and approximation in deterministic sequencing and scheduling: A survey. *Ann. Disc. Math. 5*, 287–326.

HOOGEVEEN, H., SCHUURMAN, P., AND WOEGINGER, G. J. 2001. Non-approximability results for scheduling problems with minsum criteria. *INFORMS J. Comput. 13*, 2, 157–168.

JANSEN, K., SOLIS-OBA, R., AND SVIRIDENKO, M. 2003. Makespan minimization in job shops: A linear time approximation scheme. *SIAM J. Disc. Math. 16*, 2, 288–300.

KHOT, S. 2001. Improved inapproximability results for maxclique, chromatic number and approximate graph coloring. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science (FOCS)*. 600–609.

LAWLER, E. L., LENSTRA, J., KAN, A. R., AND SHMOYS, D. 1993. Sequencing and scheduling: Algorithms and complexity. *Handb. Oper. Res. Manage. Sci. 4*, 445–522.

LEIGHTON, F. T., MAGGS, B. M., AND RAO, S. B. 1994. Packet routing and job-shop scheduling in O(congestion + dilation) steps. *Combinatorica 14*, 2, 167–180.

LEIGHTON, F. T., MAGGS, B. M., AND RICHA, A. W. 1999. Fast algorithms for finding *O*(congestion + dilation) packet routing schedules. *Combinatorica 19*, 375–401.

NAGARAJAN, V. AND SVIRIDENKO, M. 2008. Tight bounds for permutation flow shop scheduling. In *Proceedings of the 13th Conference on Integer Programming and Combinatorial Optimization (IPCO)*. 154–168.

POTTS, C., SHMOYS, D., AND WILLIAMSON, D. 1991. Permutation vs. nonpermutation flow shop schedules. *Oper. Res. Lett. 10*, 281–284.

QUEYRANNE, M. AND SVIRIDENKO, M. 2002. Approximation algorithms for shop scheduling problems with minsum objective. *J. Sched. 5*, 4, 287–305.

SCHUURMAN, P. AND WOEGINGER, G. J. 1999. Polynomial time approximation algorithms for machine scheduling: ten open problems. *J. Sched. 2*, 5, 203–213.

SEVASTIANOV, S. V. AND WOEGINGER, G. J. 1998. Makespan minimization in preemptive two machine job shops. *Computing 60*, 1, 73–80.

SHMOYS, D., STEIN, C., AND WEIN, J. 1994. Improved approximation algorithms for shop scheduling problems. *SIAM J. Comput. 23*, 617–632.

WILLIAMSON, D. P., HALL, L. A., HOOGEVEEN, J. A., HURKENS, C. A. J., LENSTRA, J. K., SEVASTIANOV, S. V., AND SHMOYS, D. B. 1997. Short shop schedules. *Oper. Res. 45*, 288–294.

Received March 2010; revised April 2011; accepted September 2011