Improved Bounds for Acyclic Job Shop Scheduling

Uriel Feige*
Dept. of Applied Mathematics and Computer Science
Weizmann Institute
76100 Rehovot, Israel

Christian Scheideler†
Heinz Nixdorf Institute and
Dept. of Mathematics and Computer Science
Paderborn University
33095 Paderborn, Germany

Abstract

In acyclic job shop scheduling problems there are n jobs and m machines. Each job is composed of a sequence of operations to be performed on different machines. A legal schedule is one in which within each job, operations are carried out in order, and each machine performs at most one operation in any unit of time. If D denotes the length of the longest job, and C denotes the number of time units requested by all jobs on the most loaded machine, then clearly $lb = \max[C, D]$ is a lower bound on the length of the shortest legal schedule. A celebrated result of Leighton, Maggs and Rao shows that if all operations are of unit length, then there always is a legal schedule of length $O(lb)$, independent of n and m. For the case that operations may have different lengths, Shmoys, Stein and Wein showed that there always is a legal schedule of length $\tilde{O}(lb(\log lb)^2)$, where $(\tilde{O})$ notation is used to suppress $\log\log(lb)$ terms. We improve the upper bound to $\tilde{O}(lb \log lb)$. We also show that our new upper bound is essentially best possible, by proving the existence of instances of acyclic job shop scheduling for which the shortest legal schedule is of length $\tilde{\Omega}(lb \log lb)$. This resolves (negatively) a known open problem of whether the linear upper bound of Leighton, Maggs and Rao applies to arbitrary job shop scheduling instances (without the restriction to acyclicity and unit length operations).

* Email address feige@wisdom.weizmann.ac.il. Incumbent of the Joseph and Celia Reskin Career Development Chair
† Email address chrsch@uni-paderborn.de. Supported in part by the DFG-Sonderforschungsbereich 376 “Massive Parallelität: Algorithmen, Entwurfsmethoden, Anwendungen” and the EU ESPRIT Long Term Research Project 20244 (ALCOM-IT). Research was done while staying at the Weizmann Institute, supported by a scholarship of the Minerva foundation.

1 Introduction

In job shop scheduling (JSS) problems there are n jobs and m machines. Each job is composed of a sequence of operations, where each operation has a machine and length (processing time) associated with it. An operation is scheduled by specifying a time step at which it begins, and then the operation is executed on its respective machine without interruption for a time period that is equal to its length. A legal schedule is one in which each operation is scheduled only after all operations preceding it in its job have been completed, and each machine performs at most one operation at any unit of time. The makespan of a legal schedule is the time by which all jobs have been completed. For a given JSS instance, let L denote the minimum makespan, taken over all possible legal schedules. Computing L is NP-hard. In this paper, we are interested in providing upper and lower bounds on L.

We shall concentrate on acyclic job shop scheduling, for which within each job, all operations are performed on different machines. A known example where acyclic JSS comes up naturally is that of message routing through a communication network. Each link of the network may be viewed as a machine, and each message may be viewed as a job. If a path is specified from the source to the destination of a message, traversing a link along the path can be viewed as an operation. Naturally, paths are acyclic. How should one schedule the transmissions of messages over links so as to minimize the time until all messages reach their destinations? If messages cannot be segmented and need to hop from vertex to vertex as a whole, and furthermore, while a message traverses a link no other message can use this link, then this question is an instance of acyclic JSS.

There are two trivial lower bounds on L. One of them is the length of the longest job, known as the dilation D. The other is the load on the most loaded machine, namely, the total number of time units that all jobs require on this machine, which is known as the congestion C. Clearly, $lb = \max[C, D]$ is a lower bound on L. Surprisingly, Leighton, Maggs and Rao [LMR94] show that for acyclic JSS, if all operations are of unit length, then in fact $L = \Theta(lb)$, regardless of any other parameter such as n and m. The proof given in [LMR94] is nonconstructive (does not provide an efficient algorithm for finding a schedule with makespan $O(lb)$), and makes repeated use of the Lovász Local Lemma. Attempts to generalize the $O(lb)$ upper bound to general JSS (with the assumptions on acyclicity or unit length removed) were unsuccessful. The best upper bounds known for general JSS are $L = O(lb(\log lb)^2/\log\log lb)$ by Shmoys, Stein and Wein [SSW94], later improved by a $\log\log lb$ factor by Goldberg, Paterson, Srinivasan and Sweedyk [GPSS97].

Our main results are as follows:

- For acyclic JSS, $L = O(lb \log lb \log\log lb)$. This improves previously known upper bounds for acyclic JSS by a factor of $\log lb/(\log\log lb)^3$.
- There are instances of acyclic JSS for which $L = \Omega(lb \log lb/\log\log lb)$. This resolves (negatively) the open question (see, *e. g.*, [SSW94, GPSS97]) of whether the linear upper bound of Leighton, Maggs and Rao applies to arbitrary job shop scheduling instances. Moreover, this shows that our upper bound for acyclic JSS is best possible up to a factor of $O((\log\log lb)^2)$.

Similar to the proof in [LMR94], the proof of our upper bound makes repeated use of the Lovász Local Lemma. However, the existence of operations of different lengths requires the use of the general version of the Local Lemma, rather than the uniform version used in [LMR94].

Our upper bound only proves the existence of short schedules. It does not provide an efficient algorithm for finding them. The main obstacle is the nonconstructive nature of the Local Lemma. There are algorithmic versions of the Local Lemma [Be91], and they have been used in order to provide a polynomial time algorithm for finding schedules of length $O(lb)$ for unit length acyclic JSS [LMR96]. Providing an algorithmic version of our upper bound is more complicated and is not discussed in this paper. We remark that for general JSS, the upper bounds of [SSW94, GPSS97] are algorithmic (though provide a weaker guarantee than our upper bound for acyclic JSS), and that it is NP-hard to approximate L within a ratio better than 5/4 [WHH+96] (even for flow shop scheduling, which is a special case of acyclic JSS). In the context of routing, one often seeks an algorithm that is not only polynomial time, but also distributed. Some work towards providing a distributed algorithmic version of [LMR94]'s schedule is presented in [LMR94, RT96, OR97].

We point out some additional consequences of our work.

- **Preemption provably helps:** In preemptive JSS one is allowed to temporarily suspend (interrupt) an operation on some machine, let the machine perform some other operation, and later resume the original operation (as if an operation of length l is composed of a sequence of l unit length operations). In some cases of JSS it makes sense to allow preemption, perhaps at some cost. Previous to our work, there was no proof that using preemption can significantly reduce the makespan. In Section 3.1 we show an example where this is indeed the case. More importantly, we are also able to prove a general upper bound of $L = O(lb \log \log lb)$ for acyclic JSS with preemption. Note that a similar upper bound does not hold if preemption is not allowed, by our lower bound of $L = \Omega(lb \log lb / \log \log lb)$.

- **Dilation versus congestion:** It has been observed that congestion and dilation influence the makespan differently. For example, Leighton, Maggs and Rao [LMR96] present a distributed algorithm for unit length acyclic JSS that produces schedules of length $O(C + D \log n)$. Our results (see Sections 2 and 3 for details) make similar distinctions between C and D. In particular, if $C = \Omega(D^{1+\epsilon})$ for some $\epsilon > 0$, then $L = \Theta(lb)$, even for general JSS.

- **Operation lengths that depend only on machine:** In a packet routing network, it may well be the case that all packets are of the same size, but that some links are slower than others (modems of different speeds on the communication lines). In this case, any two packets take the same time to cross the same link, but a packet may take more time to cross one link than another link. For acyclic JSS in which operation lengths depend only on the machine, our upper bound improves to $L = O(lb \log \log lb)$. (For general JSS, if operation lengths depend only on the machine, then $L = O(lb \log lb / \log \log lb)$. This follows from techniques of [LMR94, SSW94].)

- **Operation lengths that depend only on job:** Returning to the routing example, consider the case where each link has the same speed, but messages may be of different length. In this case, different messages may take different time to cross the same link, but any message takes the same time to cross two different links. This restriction of acyclic JSS does not seem to help, as in our negative example for which $L = \Omega(lb \log lb / \log \log lb)$, operation lengths depend only on the job. (Any instance of acyclic JSS can also be cast as an instance of message routing, though the paths that messages follow from source to destination may appear to be artificial.) This finding supports the practice of breaking messages into packets. By this we obtain an instance of unit length acyclic JSS, which by [LMR94] has a makespan of $O(lb)$. (We note that we may use here a convenient form of packet routing in which all packets corresponding to the same message follow the same path and arrive in order.)

- **A constant number of operation lengths:** If there are only a constant number of different operation lengths, then we can show that $L = O(lb)$ (the O notation hides a constant that is exponential in the number of different operation lengths).

Many questions remain open. Here are our favorite ones:

- **General JSS:** We have not been able to improve the upper bound of $L = O(lb(\log lb)^2 / (\log \log lb)^2)$ for general JSS. How tight is this upper bound?

- **Preemptive JSS:** For acyclic JSS with preemption we show that $L = O(lb \log \log lb)$. We do not know if this is best possible. Is $L = \Theta(lb)$? For general JSS with preemption we were unable to improve over the previously known upper bound of $L = O(lb \log lb / \log\log lb)$, nor provide any evidence that $L = \Theta(lb)$ is unachievable.

- **Flow shop scheduling:** An important special case of acyclic JSS is flow shop scheduling, where machines are ordered, and within each job, the order in which machines are requested respects the order of the machines. Previous to our work, the best upper bound known for flow shop scheduling was similar to that of general JSS, namely $L = O(lb(\log lb)^2/(\log\log lb)^2)$. Clearly, our upper bound of $L = O(lb \log lb \log\log lb)$ for acyclic JSS applies also to the special case of flow shop scheduling. However, this does not hold for our lower bound of $L = \Omega(lb \log lb / \log\log lb)$. Can the upper bound for flow shop scheduling be improved significantly?

1.1 Our results

We assume that the shortest operation length is 1 (this is achieved by scaling), and that all other operation lengths are a power of 2 (by rounding operation lengths up to the nearest power of 2, loosing a factor of at most 2 in congestion and dilation). Recall that C and D denote congestion and dilation (respectively), and that $lb = \max[C, D]$. Let P denote the length of the longest operation, and for general JSS, let Q denote the maximum number of time units that one job requests one machine (*i. e.*, the sum of the lengths of the operations that the job performs on that machine). Hence $P \leq Q \leq \min[C, D] \leq lb$. In the following, we assume that if $\log x$ or $\log\log x$ are smaller than 1, we always round them up to 1.

The following two theorems, which are proved in Section 2, describe our upper bounds on the length L of the shortest legal schedule.

Theorem 1.1 *For acyclic job shop scheduling, the following are upper bounds on the makespan:*

(1) For any acyclic JSS problem,

$$L = O\left((C + D\log\log P)\frac{\log P}{\log(\min[C/D + \log\log P, P])}\right),$$

implying that $L = O(lb \log lb \log\log lb)$. Observe that if $C \geq D \cdot P^{\epsilon}$ for some $\epsilon > 0$, then $L = \Theta(C)$.

(2) If there are only a constant number of operation lengths (independent of the values of C and D), then $L = \Theta(lb)$.

(3) If operation lengths depend only on the machine on which the operation is performed, then $L = O(C + D\log\log P)$.

(4) With preemption, $L = O(C + D\log\log P)$.

Theorem 1.2 *For general job shop scheduling, the following are upper bounds on the makespan:*

(1) For any JSS problem,

$$L = O\left(\left(C + D\frac{\log Q}{\log\left(2 + \frac{\log Q}{1 + C/D}\right)}\right)\frac{\log P}{\log(\min[C/D + \log Q, P])}\right).$$

Observe that if $C \geq D\max[P^{\epsilon}, \log Q]$ for some $\epsilon > 0$, then $L = \Theta(C)$.

(2) With preemption, $L = O(C + (D\log Q)/\log(2 + \frac{\log Q}{1 + C/D}))$.

The following two theorems, which are proved in Section 3, describe our lower bounds on L. For general JSS, we give a simple explicit construction showing:

Theorem 1.3 *For any integer $k > 1$, there is an instance of (general) JSS with k jobs, k machines, dilation and congestion each equal to $D = (k \log k)^k$, such that any legal schedule is of length more than $Dk(1 - \frac{1}{\log k})$. For large k this gives $L \simeq kD$ and $k \simeq \log D / \log \log D$.*

Using a probabilistic argument, the following result is shown for acyclic JSS.

Theorem 1.4 *For a large enough dilation D, there is an instance of acyclic JSS with $C < D$ and $L > D \frac{\log D}{16 \log \log D}$.*

2 The upper bounds

In this section we prove Theorems 1.1 and 1.2. Let us first give a high level overview of our approach and the new ingredients that it contains compared to [LMR94, SSW94, GPSS97]. For simplicity, consider an instance of acyclic job shop scheduling with $n = m = D = C$ and operations of varying lengths. The approach of [SSW94] for handling this instance has two phases:

(1) Use a simple randomized algorithm to construct an intermediate schedule of length $O(n)$. This intermediate schedule may be illegal in the sense that it contains contention conflicts: at certain time steps as many as $\alpha = O(\log n / \log \log n)$ jobs may want to operate on the same machine.

(2) Convert the intermediate schedule into a legal schedule of length $O(\alpha n \log n)$.

In [GPSS97], the second phase was improved to give a schedule of length $O(\alpha n \log n / \log \log n)$.

We also follow this two phase approach. In the intermediate schedule we allow for higher contention $\alpha = O(\log n \log \log n)$. However, this contention is distributed evenly over operations of different lengths, where for each value of t, operations of length t contribute at most $O(\log \log n)$ to the contention. This allows for a more efficient implementation of the second phase, which now gives a legal schedule of length essentially $O(\alpha n)$.

In order to construct our intermediate schedule, we follow the approach of [LMR94]. We start with the (illegal) schedule of running all jobs in parallel and make a sequence of refinement steps, each time obtaining a new (illegal) schedule. The feasability of each refinement step is proved using the Lovász Local Lemma. However, the existence of operations of different lengths (rather than just unit length operations) causes significant differences between our refinement steps and those of [LMR94]. We use many more refinement steps ($O(\log n / \log \log n)$ rather than $O(\log^* n)$), and we use the general form of the Local Lemma rather than the uniform (symmetric) version.

In Section 2.1, we give some definitions and establish some notation that is used throughout the proofs of the upper bounds. Section 2.2 reviews some probabilistic lemmas that we later use. In Section 2.3, we show how to convert suitable intermediate schedules for any JSS problem into legal schedules. Section 2.4 proves the existence of an efficient intermediate schedule for general JSS (used in the proof of Theorem 1.2), and Section 2.5 proves the existence of an efficient intermediate schedule for acyclic JSS (used in the proof of items (1), (3) and (4) in Theorem 1.1). In Section 2.6, we present the proof of Theorem 1.1 (2).

2.1 Definitions

Our proofs will use the following notation. A t-operation is an operation of length t. We assume that the shortest operation length is 1 (this is achieved by scaling), and that all other operation lengths are a power of 2 (by rounding operation lengths up to the nearest power of 2, loosing a factor of at most 2 in congestion and dilation). We use the notation $(\leq t)$-operation to denote an operation whose length is not more than t, and T-operation to denote an operation whose length belongs to the set T.

A time interval is composed of consecutive time units. A t-*interval* denotes a time interval of length t. Some intervals, called *frames*, are special. For a frame F, its length $|F|$ is always a power of two, and its starting time is an integer multiple of $|F|$. A schedule is called *well-structured* if every t-operation falls into a frame of length t.

For a (possibly illegal) schedule, the $(\leq t)$-*congestion* for machine M in time interval I is the sum over all $(\leq t)$-operations that are scheduled to begin execution on M within I, where each such t'-operation contributes t' to the sum. The T-*contention* of machine M is the maximum number of T-operations that are scheduled on it at one time unit (for a legal schedule, the contention of every machine is one).

As the length of the largest operation is P, there are at most $1 + \log P$ different operation lengths. Partition operation lengths into consecutive sets of cardinality c. These sets are denoted in the following by $T_1, T_2, \ldots, T_{(1+\log P)/c}$, where set T_i contains all operation lengths from $2^{(i-1)c}$ to 2^{ic-1}. (For acyclic JSS, we shall later fix $c = \log\log P$.) We remark that in Section 2.5 we shall use a slightly modified definition for T_i. This modified definition appears in the beginning of Section 2.5.

2.2 Probabilistic tools

In this section, we summarize the main probabilistic tools we will use in our proofs. Let us first recall the general form of the Lovász Local Lemma (see [ASE92]).

Lemma 2.1 (Lovász) *Let $A_1, \ldots, A_n$ be a set of "bad" events with dependency graph G. (That is, A_i is independent of the set of all events A_j with $(i,j) \notin G$.) Assume that there exist $p_1, \ldots, p_n \in [0,1)$ with*

$$\Pr[A_i] < p_i \prod_{(i,j)\in G} (1 - p_j)$$

for all i. Then

$$\Pr[\bigcap \bar{A}_i] > \prod_{i=1}^{n} (1 - p_i) > 0 ,$$

that is, with probability greater than zero no bad event occurs.

If all events have a similar probability of being true and roughly the same degree in the dependency graph, one can use the following uniform (or symmetric) form of the Local Lemma.

Lemma 2.2 *Let $A_1, \ldots, A_n$ be a set of "bad" events, each A_i occuring with probability at most p and independent of all but at most b other events in $\{A_1, \ldots, A_n\}$. If $\mathrm{e}p(b+1) \leq 1$, then with probability greater than zero no bad event occurs.*

Furthermore, we use the general Chernoff-Hoeffding bounds (see [Ho87]).

Lemma 2.3 (Chernoff-Hoeffding) *Let $X_1, \ldots, X_n \in [0,k]$ be independent random variables. Further let $X = \sum_{i=1}^{n} X_i$ and $\mu \geq \mathrm{E}[X]$. Then it holds for every $\epsilon \geq 0$ that*

$$\Pr[X \geq (1+\epsilon)\mu] \leq \left(\frac{\mathrm{e}^{\epsilon}}{(1+\epsilon)^{1+\epsilon}} \right)^{\mu/k} .$$

In case of $0 \leq \epsilon \leq 1$, the following bound holds

$$\Pr[X \geq (1+\epsilon)\mu] \leq \mathrm{e}^{-\epsilon^2 \mu/3k} .$$

2.3 Transforming suitable illegal into legal job shop schedules

The following lemma describes an intermediate schedule that may be illegal (in the sense that a machine may need to perform several operations at the same time), but can be transformed into a legal schedule with relatively short makespan. The lemma holds for general JSS (acyclicity is not required).

Lemma 2.4 *Let $\alpha, \beta, \gamma, \delta$ be nonnegative parameters that may depend on C and D. Assume a well-structured schedule of length δD with the following properties holding for every machine:*

- *For every set T_i, the T_i-contention is at most $\alpha + \beta C/D$.*
- *For any frame F, where $|F|$ is a power of 2^c, the $(< |F|/2^c)$-congestion is at most $\gamma|F|C/D$.*

Then the intermediate schedule can be transformed into

(1) a legal schedule of length $L \leq 2\delta D\phi \log P/\log(\min[\phi, P])$ with $\phi = 2\alpha + (2\beta + \gamma)C/D$ for the general case,

(2) a legal schedule of length $L \leq \delta(\alpha D + \beta C)$ for the case that the operation length depends only on the machine on which the operation is performed, or

(3) a preemptive schedule of length $L \leq 2\delta(\alpha D + (\beta + \gamma)C)$.

Assume that Lemma 2.4 is true. Then Theorem 1.1 (1), (3) and (4) would follow if we could prove the existence of a well-structured schedule for acyclic JSS with

$$c = \log\log D, \quad \alpha = O(\log\log P), \quad \gamma = O(1 + D/C), \quad \text{and} \quad \beta, \delta = O(1) \ .$$

Theorem 1.2 would follow if we could prove the existence of a well-structured schedule for general JSS with

$$c = \log D, \quad \alpha = O\left(\frac{\log Q}{\log\left(2 + \frac{\log Q}{C/D+1}\right)}\right), \quad \gamma = O(1 + D/C), \quad \text{and} \quad \beta, \delta = O(1) \ .$$

Let us first prove Lemma 2.4. In the following, let $\mathcal{S}$ denote a schedule that fulfills the requirements of Lemma 2.4.

2.3.1 Proof of Item 1

The proof extends a technique by Goldberg *et al.* [GPSS97]. Let us start with partitioning $\mathcal{S}$ into frames of length P. By the definition of P and the fact that $\mathcal{S}$ is well-structured, no operation crosses a frame border. We will show how to construct a legal schedule for the operations in each frame. Concatenating these schedules yields a legal schedule for the whole JSS problem.

Consider some fixed frame F. Let T be a rooted complete binary tree with P leaves labeled, from left to right, $0, 1, \ldots, P-1$, each leaf denoting a time step of the frame. Let u be a node in T and let $\ell(u)$ and $r(u)$ be the labels of the leftmost and rightmost leaves of the subtree rooted at u. For every machine M_i, we define $S_i(u)$ to be those operations that are scheduled on M_i in F for *precisely* the time interval $[\ell(u), r(u)]$. Hence each operation is in exactly one $S_i(u)$. In the following, we will describe a frame scheduling algorithm that produces a legal schedule for F with makespan $2\phi P\frac{\log P}{\log(\min[\phi,P])}$, where $\phi = 2\alpha + (2\beta + \gamma)C/D$.

Let us assume in the following that ϕ is a power of two. Mark a node u if it is at height $(0 \mod \log\phi)$ in T. (If u is a leaf, then its height is 0, the father of u has height 1, and so on.) Eliminating the edges between a marked node and its children partitions T into a collection of subtrees, each of height (at most) $\log\phi$.

Let T' be one of the subtrees of the partition. We move all operations stored in T' from the sets $S_i(v)$ to new sets $S'_i(v)$ for every machine M_i. Initially, each $S'_i(u)$ is empty for all $u \in T'$. We describe a procedure for building up the sets $S'_i(u)$. At each step of this procedure, $p_i(u)$ denotes the time needed to process all operations currently in $S'_i(u)$. We scan the nodes of T' from the bottom upwards, starting with all nodes at height 0, then continuing with all nodes at height 1, and so on until the root of T' is reached. For a node v in T' we scan its operations in an arbitrary order. For each such operation that has to be performed on machine M_i (*i. e.*, this is an operation belonging to $S_i(v)$) we place it in $S'_i(w)$, where w is a leaf of the subtree of T' rooted at v with the currently smallest $p_i(w)$. We update the respective $S'_i(w)$ and $p_i(w)$. In essence, this procedure amounts to:

Distribute all operations of $S_i(v)$ among the $S'_i(w)$ of its leaves w such that their $p_i(w)$'s are as balanced as possible.

We now view T once again as one complete binary tree. For each vertex u we define $p(u) = \max_i p_i(u)$. Let the nodes of T be numbered as $u_1, u_2, \ldots$ in the preorder traversal of T. Define $f(u_1) = 0$, and for any $j \geq 2$ let $f(u_j) = \sum_{k<j} p(u_k)$. Run the following scheduling algorithm to produce a schedule $\mathcal{S}_F$:

Schedule the operations in $S'_i(u)$ on machine M_i consecutively beginning at time step $f(u)$ and concluding before $f(u) + p(u)$.

It remains to show that $\mathcal{S}_F$ is legal and has a makespan of at most $2\phi P \frac{\log P}{\log(\min[\phi,P])}$. Consider any subtree T' of the partition. Assume the leaves of T' are at height j in T. Clearly, the algorithm for distributing the operations in the $S_i(u)$ among the $S'_i(u)$ ensures that all $S'_i(u)$ for all nodes in T' are empty except for the leaves of T'. We want to show by complete induction that for every height $h \in \{0, \ldots, \log \phi\}$, after the algorithm has distributed all operations in the S_i of nodes of height h in T' among the S'_i of their leaves, $p(u) \leq \phi 2^j + 2^{j+h}$ for every leaf u in T'.

For $h = 0$, this assumption follows directly from the contention bound in Lemma 2.4. Now suppose that the bound on the $p(u)$'s is true for some $h \geq 0$. Then we want to show that it also holds for $h + 1$.

Assume in the contrary that the assumption is not true for $h + 1$. Then there must exist a machine M_i and a node v at height $h + 1$ in T' with one of its leaves u having $p_i(u) > \phi 2^j + 2^{j+h+1}$. Since after the distribution of operations in nodes of height h, $p_i(u)$ still was at most $\phi 2^j + 2^{j+h}$, u must have an operation from $S_i(v)$. If this operation cannot be moved to another leaf to decrease the number of leaves w with $p_i(w) > \phi 2^j + 2^{j+h+1}$, $p_i(w)$ must be greater than $\phi 2^j$ for all leaves w under v. This, however, induces a total congestion of more than $2^{h+1} \cdot \phi 2^j$ by the $(\leq 2^{j+h+1})$-operations stored in the subtree of v, which is a contradiction to the contention and congestion bounds of Lemma 2.4 (two times the contention bound for the largest operations plus the congestion bound for the rest are sufficient to bound the total congestion in the subtree of v). Hence, the operations can be distributed in such a way that for all leaves u it holds

$$p_i(u) \leq \phi 2^j + 2^{j+\log \phi} = \phi 2^{j+1} .$$

There are $P/2^j$ nodes at this height in T. The sum of these $p(u)$'s is thus at most $2\phi P$. Each subtree layer therefore contributes at most $2\phi P$, and there are $\frac{\log P}{\log(\min[\phi,P])}$ layers. Thus, $\sum_{v \in T} p(v)$, the makespan of schedule $\mathcal{S}_F$, is at most $2\phi P \frac{\log P}{\log(\min[\phi,P])}$.

To show that $\mathcal{S}_F$ is legal we observe that by construction, no machine performs more than one operation at a time. We also need to show that each job performs its operations in order. For this, consider two operations O_1 and O_2 of the same job, where O_2 follows O_1. Then we have that O_1 finishes before O_2 begins under $\mathcal{S}$. Assume (the more difficult case) that $\mathcal{S}$ schedules O_1 and O_2 within the same frame F. Consider the respective tree T described above and let nodes u and v be such that $O_1 \in S_i(u)$, $O_2 \in S_j(v)$. Then u and v are roots of disjoint subtrees of T and u precedes v in the preorder traversal of T. Thus O_1 finishes before O_2 begins in $\mathcal{S}_F$, and the new schedule is feasible.

Concatenating all $\delta D/P$ legal frame schedules yields a legal schedule whose makespan is bounded as described in Lemma 2.4(1).

2.3.2 Proof of Item 2

For item 2 (operation length depends on machine), observe first that the contention is an integer, and hence $\alpha + \beta C/D$ is an integer. Modify the original schedule in a greedy way such that each l-operation on machine M that started at time t (recall that t is an integer multiple of l) now starts at a time step $(\alpha + \beta C/D)t + il$, where $0 \leq i < (\alpha + \beta C/D)$, and i is chosen such that no previously considered l-operation on machine M is scheduled to begin at the same time. Clearly, the length of the new schedule is at most $\delta D(\alpha + \beta C/D)$. The new schedule is legal for the following reasons. For each job, operations are performed in order, because they were performed in order in schedule $\mathcal{S}$, and every operation that started at time t_1 and ended at time t_2 in $\mathcal{S}$ now starts at time $t_1' \geq (\alpha + \beta C/D)t_1$ and ends at time $t_2' \leq (\alpha + \beta C/D)t_2$ in the new schedule. Furthermore, each machine performs at most one operation at every unit of time, because all operations in one machine are of the same length.

2.3.3 Proof of Item 3

Let us divide all operations into the following two sets:

- Set A consists of all T_i-operations, where $i \in \{1, \ldots, \frac{1+\log P}{c}\}$ is odd.
- Set B consists of all T_i-operations, where $i \in \{2, \ldots, \frac{1+\log P}{c}\}$ is even.

We first show that for each of the sets it is possible to produce a (still illegal) schedule $\mathcal{S}'$ with contention $\alpha + (\beta + \gamma)C/D$ such that no operation starts earlier than supposed to start according to $\mathcal{S}$ and ends later than supposed to end according to $\mathcal{S}$. We achieve this by suitably distributing the units of processing among the time steps, allowing several units of processing from the same operation to be placed at the same time step.

W.l.o.g., let us concentrate on scheduling operations in A. First, let us schedule the T_1-operations. According to the contention bound in Lemma 2.4, this can be done with contention $\alpha + \beta C/D$ without moving the operations.

Suppose now that for some odd $i \in \{3, \ldots, \frac{1+\log P}{c}\}$ all T_j-operations in A with $j < i$ can be scheduled with contention $\alpha + (\beta + \gamma)C/D$. Then we want to show that also the T_j-operations in A with $j \leq i$ can be scheduled with contention $\alpha + (\beta + \gamma)C/D$. Let us choose the strategy that we always try to fully exploit this contention bound, preferring insertions of units of processing from the operation that has to finish next in $\mathcal{S}$. Suppose that with this strategy there is an operation that cannot be inserted before it ends according to $\mathcal{S}$. We will show by contradiction that this cannot happen.

Let us choose the first operation, say O, where a bad event happens. Let us go back in time until we reach a point where a contention of $\alpha + (\beta + \gamma)C/D$ could not be fully exploited (or we reach the beginning of $\mathcal{S}$). This means that all T_i-operations that started before could already be inserted. Choose I as the interval that starts at the first point (forward in time) from this point, where a T_i-operation starts in $\mathcal{S}$, and ends where O is supposed to end in $\mathcal{S}$. Since the starting point of I cannot be later than the time step when O starts in $\mathcal{S}$, the length of I is at least $2^{(i-1)c}$. In fact, we chose I above such that its length is an integer multiple of $2^{(i-1)c}$ (which is the smallest length a T_i-operation can have). Since $\mathcal{S}$ is well-structured, all units of processing of T_j-operations with $j < i$ that are inserted in I belong to T_j-operations that start in I. From the congestion bound in Lemma 2.4 we know that within such an interval the congestion caused by T_j-operations in A with $j < i$ is at most $\gamma|I|\frac{C}{D}$. Hence, if a contention of $\alpha + (\beta + \gamma)C/D$ does not suffice to insert all T_i-operations that end no later than O, then, since all T_i-operations starting before I could be inserted before I, this means that somewhere in I there must have been a contention among the T_i-operations of more than $\alpha + \beta C/D$, which contradicts the contention bound in Lemma 2.4.

Hence, all operations in A can be scheduled with contention $\alpha + (\beta + \gamma)C/D$, and therefore all operations in A and B can be scheduled with contention $2(\alpha + (\beta + \gamma)C/D)$. Since a schedule of length t that has contention c can be simulated by a preemptive schedule of length $c \cdot t$ with contention 1, item 3 follows.

2.4 An intermediate schedule for general JSS

In this section, we prove the existence of an intermediate schedule that fulfills the requirements of Lemma 2.4 with

$$c = \log D, \quad \alpha = O\left(\frac{\log Q}{\log\left(2+\frac{\log Q}{1+C/D}\right)}\right), \quad \gamma = O(1+D/C), \quad \text{and} \quad \beta, \delta = O(1) \ .$$

We assume w.l.o.g. that $C \geq D$. (If $D > C$, just use D instead of C as an upper bound on the congestion.) Since $c = \log D$, all operations (except of the D-operations) are combined into one set T_i. Hence, it remains to show the following lemma.

Lemma 2.5 *For any job shop scheduling problem, there exists a well-structured schedule of length $O(D)$ such that for every machine the contention is bounded by*

$$O\left(\frac{C}{D} + \frac{\log Q}{\log\left(2+\frac{\log Q}{1+C/D}\right)}\right) \ .$$

Proof. Let us first assume that $Q \geq D^{1/4}$, *i. e.*, $\log Q = \Theta(\log D)$. Consider, for each job, the random experiment of choosing a delay independently and uniformly at random from a range of $[0, D-1]$. Each job that is assigned a delay of δ waits for δ time steps and then is processed by the machines in the prescribed order until it is completed. We use this random experiment to show with the help of the Lovász Local Lemma that delays exist such that Lemma 2.5 is satisfied. In order to simplify the proof, let us assume in the following that D is large enough (in fact, $D \geq 8$ suffices).

Fix a machine M_i. We want to bound the probability that the contention among the operations in M_i exceeds some limit. For this, let us consider some fixed time step t. Let q be the probability that at least k units of processing are scheduled on M_i at t. There are at most $\binom{C}{k}$ ways to choose k units of processing from the operations for M_i. For each such choice, the probability that all k units are scheduled at t is at most $(1/D)^k$. Hence we get with $k = 2\mathrm{e}\frac{C}{D} + 4\log(CD)/\log(2+\frac{\log(CD)}{1+C/D})$:

$$\begin{aligned} q &\leq \binom{C}{k}\left(\frac{1}{D}\right)^k \leq \left(\frac{\mathrm{e}C}{k \cdot D}\right)^k = \left(\frac{1}{2}\right)^{k\log\left(\frac{kD}{\mathrm{e}C}\right)} \\ &\leq \left(\frac{1}{2}\right)^{\frac{4\log(CD)}{\log\left(2+\frac{\log(CD)}{1+C/D}\right)} \cdot \log\left(2+\frac{\log(CD)}{C/D} \cdot \frac{1}{\log\left(2+\frac{\log(CD)}{1+C/D}\right)}\right)} \\ &\leq \left(\frac{1}{2}\right)^{2\log(CD)} = \frac{1}{(CD)^2} \ . \end{aligned}$$

Let us define event A_i to be true if M_i has a contention of at least k at some time step. Since there are at most $2D$ time steps to consider, the probability p that A_i is true is at most $1/(4CD)$.

In order to apply the Lovász Local Lemma, we have to bound the dependencies among all these events for all machines. Whether or not an event becomes true depends solely on the delays assigned to the jobs. Thus, two events are independent unless some job has operations in both of the corresponding machines. Since every machine processes at most C operations and each of these operations belongs to a job with at most D other operations, the dependence b among the events is at most $C \cdot D$. Hence, $\mathrm{e}p(b+1) < 1$ and therefore, by the Lovász Local Lemma, there exist delays such that the contention at every machine is bounded by

$$2\mathrm{e}\frac{C}{D} + \frac{4\log(CD)}{\log\left(2+\frac{\log(CD)}{1+C/D}\right)} = O\left(\frac{C}{D} + \frac{\log D}{\log\left(2+\frac{\log D}{1+C/D}\right)}\right) \ .$$

It remains to transform this schedule into a well-structured schedule. As was shown in [GPSS97], a non-well-structured schedule can be easily transformed into a well-structured schedule by stretching the makespan by a factor of two as follows:

For any l and t, operations of length l that start at time t in the original schedule start at a time step that is the least integer multiple of l that is at least as large as $2t$. Two operations overlap in the new schedule only if they overlap in the original schedule.

So for $Q \geq D^{1/4}$ the lemma is true. It remains to show what to do if $Q < D^{1/4}$.

Our strategy will be to make a succession of refinements to an initial schedule S_0 until we reach a schedule S_i of length $O(D)$ with the property that for every frame of size Q^3 the congestion at any machine is bounded by $O(\frac{C}{D} \cdot Q^3)$. Then we split S_i into frames of size Q^3 and schedule each of them independently as described above. This yields a final subschedule of length $O(Q^3)$ with contention

$$O\left(\frac{C}{D} + \frac{\log Q^3}{\log\left(2 + \frac{\log Q^3}{1+C/D}\right)}\right) .$$

Combining these subschedules into one schedule by executing them one after the other would therefore yield Lemma 2.5. In order to show how to do the refinements, we need the following lemma.

Lemma 2.6 *For any job shop scheduling problem with congestion C, dilation D and $Q < D^{1/4}$, there exists a schedule of length $O(D)$ such that for any Q^3-interval the congestion at any machine is bounded by $O(\frac{C}{D} \cdot Q^3)$.*

Proof. The proof uses the Chernoff-Hoeffding bounds and the Lovász local lemma at each refinement step. Let us start with an initial schedule S_0, in which each job is executed at every time step until it is completed. This initial schedule is as short as possible; its length is only D. Let $I_0 = D$ and $I_j = \max[\log I_{j-1}, Q]$ for all $j \geq 1$. Our aim is to show that for all $j \geq 1$ with $I_j \geq \max[\log I_{j-1}, Q, 36]$ there exists a refinement from schedule S_{j-1} to S_j such that the congestion at any machine for any I_j^3-interval is bounded by $O(\frac{C}{D} \cdot I_j^3)$. So when $I_j = Q$ or $\log I_j < 36$ for the first time, we reached the situation claimed in the lemma. (The condition $I_j \geq 36$ was chosen to simplify the presentation of the proof. Basically, our techniques can be used for any $I_j \geq Q \geq 1$.)

The first step is to assign an initial delay to each job, chosen independently and uniformly at random from the range $[0, \Delta_1 - 1]$, where $\Delta_1 = D$. In the resulting schedule, S_1, a job that is assigned a delay of δ waits for δ steps and afterwards is processed without waiting again until it is completed. The length of S_1 is at most $2D$. We use the Lovász Local Lemma to show that if the delays are chosen independently and uniformly at random and I_1 is sufficiently large, then with nonzero probability the congestion at any machine in any I_1^3-interval is at most $C_1 = \frac{C}{D}(1 + \frac{4}{\sqrt{I_1}})I_1^3$. Thus, such a set of delays must exist.

To apply the Local Lemma, we associate a bad event with each machine. For machine M and time interval I, we define the congestion of M at I to be the sum of the lengths of those operations on M that start within time interval I. (Hence, operations that start in an earlier time interval and extend into I are not counted, whereas for operations that start in I we also count their part that extends into subsequent intervals.) The bad event for machine M is that there is some I_1^3-interval with a congestion of more than C_1. We bound the dependence b among the bad events and the probability p that a bad event occurs.

We first bound the dependence. Whether or not a bad event occurs depends solely on the delays assigned to the jobs that have operations for the corresponding machine. Thus, two bad events are independent unless some job has operations in both of the corresponding sets. Since each machine processes at most C operations and each of the corresponding jobs has operations in at most D other sets, the dependence b of the bad events is at most $C \cdot D$.

Next we bound the probability that a bad event occurs. Consider some fixed machine M and I_1^3-interval I. Let m be the number of operations for M. For every $i \in \{1, \ldots, m\}$, let the random variable X_i be the length of operation i if operation i is started during I and 0 otherwise. Let $X = \sum_{i=1}^m X_i$. Since the jobs choose their delays uniformly at random from a range of size Δ_1, we get $\Pr[X_i \neq 0] \leq I_1^3/\Delta_1$ for all $i \in \{1, \ldots, m\}$. Hence,

$$\mathrm{E}[X] = \sum_{i=1}^m \mathrm{E}[X_i] \leq \frac{C}{D} \cdot I_1^3 \ .$$

As every job has operations for at most Q time steps at M, the X_i's can be grouped together to independent random variables of value at most Q. Thus, we get together with the Chernoff-Hoeffding bounds and $\epsilon = \frac{4}{\sqrt{I_1}}$ (note that $\epsilon < 1$ because $I_1 \geq 36$) that

$$\Pr[X \geq C_1] = \Pr\left[X \geq (1+\epsilon)\tfrac{C}{D} \cdot I_1^3\right] \leq \mathrm{e}^{-\left(\frac{4}{\sqrt{I_1}}\right)^2 \frac{C}{D} \cdot I_1^3/3Q} \leq \mathrm{e}^{-4\frac{C}{D} \cdot I_1} \leq \frac{1}{(CD)^2} \ .$$

For the last inequality we used $I_1 \geq \log D$, and $C/D \geq \log C/\log D$ for $C \geq D > 2$. Since for each set there are at most $D + \Delta_1 = 2D$ many I_1^3-intervals to consider, the probability that a bad event occurs for machine M is bounded by $p \leq 1/(4CD)$. Thus, the product $\mathrm{e}p(b+1)$ is less than 1 and therefore, by the Lovász Local Lemma, there is an assignment of delays such that the congestion at every machine in every I_1^3-interval is bounded by C_1.

We now break schedule S_1 into frames of length I_1^4. With each frame we associate those operations that S_1 schedules to start within the time interval of the frame. For each job that has at least one operation associated with the frame, we consider the fragment of the job composed of those operations associated with the frame. For each frame separately, for each fragment of a job, we choose an additional delay relative to the starting point of the frame. Schedule S_2 then results from executing the modified frames one after the other.

To simplify terminology, we treat each fragment of a job as a job by itself. The final schedule will guarantee that fragments of the same job are scheduled in order. Consider now some fixed I_1^4-frame F in S_1. Let the delays of the jobs in F be chosen in the range $[0, \Delta_2 - 1]$, where $\Delta_2 = I_1^3 - I_2^3$. Hence, the length of the resulting schedule for this frame is at most $I_1^4 + I_1^3 + P \leq (1 + 2/I_1)I_1^4$ (recall that $P \leq Q \leq I_1$). We use the Lovász Local Lemma to show that if the delays are chosen independently and uniformly at random and I_2 is sufficiently large, then with nonzero probability the congestion in any I_2^3-interval is at most

$$C_2 = \left(1 + \frac{6}{\sqrt{I_2}}\right)\left(\frac{C_1}{I_1^3 - I_2^3}\right) I_2^3 = \frac{C}{D}\left(1 + \frac{4}{\sqrt{I_1}}\right)\left(1 + \frac{6}{\sqrt{I_2}}\right)\left(\frac{1}{1 - (I_2/I_1)^3}\right) I_2^3 \ .$$

Thus, such a set of delays must exist.

To apply the Lovász Local Lemma, we associate a bad event with each machine. The bad event for machine M is that there is some I_2^3-interval I with a congestion of more than C_2. Let m be the number of operations on M. For every $i \in \{1, \ldots, m\}$, let the random variable X_i denote the length of operation i if operation i is started during I and 0 otherwise. Let $X = \sum_{i=1}^m X_i$. Since the jobs choose their delays uniformly at random from a range of size Δ_2, we get $\Pr[X_i \neq 0] \leq I_2^3/\Delta_2$ for all $i \in \{1, \ldots, m\}$. Hence,

$$\mathrm{E}[X] = \sum_{i=1}^m \mathrm{E}[X_i] \leq \frac{C_1}{I_1^3 - I_2^3} \cdot I_2^3 \ .$$

Since an event $X_i \neq 0$ influences the outcome of X by at most Q, we get together with the Chernoff-Hoeffding bounds and $\epsilon = \frac{6}{\sqrt{I_2}}$ (note that $\epsilon \leq 1$) that

$$\begin{aligned} \Pr[X \geq C_2] &= \Pr\left[X \geq (1+\epsilon)\frac{C_1}{I_1^3 - I_2^3} \cdot I_2^3\right] \\ &\leq \mathrm{e}^{-\left(\frac{5}{\sqrt{I_2}}\right)^2 \frac{C_1}{I_1^3 - I_2^3} \cdot I_2^3/3Q} \leq \mathrm{e}^{-12\frac{C}{D}I_2} \leq 2^{-17\frac{C}{D}I_2} . \end{aligned}$$

Since the total number of I_1^3-intervals to be considered is at most $|F| + \Delta_2 \leq 2I_1^4$, the probability that a bad event occurs for machine M is bounded by

$$p \leq 2I_1^4 \cdot 2^{-17\frac{C}{D}I_2} < \tfrac{D}{C} I_1^{-12} .$$

Next we bound the dependence among these events. Clearly, two bad events are independent unless some job has operations at both of the corresponding machines. Since each machine has at most $I_1 C_1$ operations and each of the corresponding jobs has operations in at most I_1^4 other machines, the dependence b of the bad events is at most $C_1 I_1^5 \leq 2\frac{C}{D} I_1^8$. Hence, the product $\mathrm{e}p(b+1)$ is less than 1 and therefore, by the Lovász Local Lemma, there is some assignment of delays such that the congestion at every machine in every I_2^3-interval is bounded by C_2.

We continue to refine each frame recursively in a way similar to above. That is, for frames of length I_j^4 we choose delays up to a value of $I_j^3 - I_{j+1}^3$ and bound the congestion in intervals of lengths I_{j+1}^3. Eventually, we reach a value k for which either $I_k = Q$ or $\log I_k < 36$ (implying that I_k is a constant). We end up with a schedule S_k with a total length of at most

$$2D \prod_{j=1}^{k-1} \left(1 + \frac{2}{I_j}\right) = O(D)$$

and with a congestion C_k in each I_k^3-frame of at most

$$\frac{C}{D} \cdot I_k^3 \left(1 + \frac{4}{\sqrt{I_1}}\right) \prod_{j=2}^{k} \left[\left(1 + \frac{6}{\sqrt{I_j}}\right)\left(\frac{1}{1 - (I_j/I_{j-1})^3}\right)\right] = O\left(\frac{C}{D} \cdot I_k^3\right) ,$$

which proves Lemma 2.6. □

This completes the proof of Lemma 2.5. □

2.5 An intermediate schedule for acyclic JSS

In this section we prove the existence of an intermediate schedule that fulfills the requirements of Lemma 2.4 with

$$c = \log\log D, \quad \alpha = O(\log\log P), \quad \gamma = O(1 + D/C), \quad \text{and} \quad \beta, \delta = O(1) .$$

As in Section 2.4 we assume w.l.o.g. that $C \geq D$. Also, we assume that D and $\log D$ are powers of two. It is not difficult to see that if this is not true, rounding these to the next higher power of two does not affect the bounds in this section. To simplify subsequent notation, we reverse the order of the T_i. That is, for every $i \in \{1, \ldots, \frac{\log D}{\log\log D}\}$ the set T_i is defined as $\{\frac{D}{\log^i D}, \ldots, \frac{D}{2\log^{i-1} D}\}$ and $T_0 = \{D\}$. Since there can be at most C/D many T_0-operations on each machine, we will not consider these operations in the following.

Lemma 2.7 *For any acyclic job shop scheduling problem, there exists a well-structured schedule of length $O(D)$ with the following properties holding for every machine:*

- *For every set T_i, the T_i-contention is at most $O(\log\log P + C/D)$.*
- *For every frame F, where $|F|$ is a power of $\log D$, the $(< |F|/\log D)$-congestion is at most $O(|F|C/D)$.*

Proof. We will only prove the existence of a non-well-structured schedule with the properties above. As noted in Lemma 2.5, it can be easily transformed into a well-structured schedule with the same properties.

In case that $P \geq D^{1/4}$, we have $\log\log P = \Theta(\log\log D)$ and therefore can replace the contention bound above by $O(\log\log D + C/D)$. If $P < D^{1/4}$, then we use the same strategy as described for $Q < D^{1/4}$ in Lemma 2.5 to create a schedule that can be partitioned into frames of size P^3 with congestion at most $O(\frac{C}{D} \cdot P^3)$, and continue to schedule each of these frames independently. It therefore suffices to consider the case $P \geq D^{1/4}$, *i. e.*, to show that there exists an intermediate schedule with a T_i-contention of $O(\log\log D + C/D)$ for every i.

Our strategy for constructing an efficient schedule is to make a succession of refinements to an initial schedule S_0. In S_0 each job is executed at every time step until it is completed. This initial schedule is as short as possible; its length is only D. Unfortunately, as many as C jobs may have to use a machine at a single time step in S_0. In order to refine S_0 to a schedule S_1, each job is assigned a suitable delay from the range $[0, D - \frac{D}{\log D} - 1]$. Every job that is assigned a delay of δ waits for δ time steps and then is processed by the machines in the prescribed order until it is completed. Hence, the length of S_1 is bounded by $2D - \frac{D}{\log D}$.

In order to refine S_1 to S_2, each job in S_1 is again assigned a suitable delay, but this time from the range $[0, \frac{D}{\log D} - \frac{D}{\log^2 D} - 1]$. Each job that is assigned a delay of δ waits for δ time steps in addition to the time steps it is already waiting according to S_1 and then is processed by the machines in the prescribed order until it is completed. Hence, the length of S_2 is bounded by $2D$.

For the refinement from S_2 to S_3, we break the schedule of S_2 into frames of length $\frac{D}{\log D}$. Each such frame F_j is considered separately from the other frames, and a subschedule $S_{3,j}$ is constructed for it. Then the subschedules $S_{3,j}$ are concatenated back together (with some overlap) to give schedule S_3.

Let us describe in a bit more detail how to refine a frame F_j to a subschedule $S_{3,j}$. Each operation is associated with the frame at which it starts. Within each frame, each job is given a suitable delay in the range $[0, \frac{D}{\log^2 D} - \frac{D}{\log^3 D} - 1]$. So a job starting at time t in F with delay δ now starts at time $t + \delta$ in $S_{3,j}$. There is no simple bound on the length of $S_{3,j}$, due to long operations that start at F_j and can extend well into subsequent frames. We consider the first $\frac{D}{\log D} + \frac{2D}{\log^2 D}$ steps of a subschedule to be its main part, and the rest of the steps as its extension. Note that all $(< \frac{D}{\log^2 D})$-operations are guaranteed to be completely processed during the main part, whatever their delay is. Schedule S_3 then consists of processing the main parts of $S_{3,1}, S_{3,2}, \ldots$ one after the other, letting the extension of a subschedule run in parallel to the main parts of subschedules that follow it.

The rest of the refinements is similar to the refinement from S_2 to S_3, with the difference that, for a refinement from schedule S_{r-1} to schedule S_r for some $r \geq 4$, S_{r-1} is broken into frames of size $\frac{D}{\log^{r-2} D}$ and within each frame, each job is given a suitable delay in the range $[0, \frac{D}{\log^{r-1} D} - \frac{D}{\log^r D} - 1]$. The main part of any resulting subschedule is considered to be its first $\frac{D}{\log^{r-2} D} + \frac{2D}{\log^{r-1} D}$ steps.

We continue to do these refinements until for the first time, frames of size at most $\log^3 D$ have been refined. Hence, the number of refinement steps is roughly $\frac{\log D}{\log\log D}$. Our refinements lead to the following claim.

Claim 2.8 *For any $r \geq 2$, the total length of schedule S_r is bounded by*

$$\left(1 + \frac{2}{\log D}\right)^{r-2} 2D \ .$$

This claim shows that the length of the final schedule is at most $3D$ (*i. e.*, $\delta \leq 3$). It remains to prove bounds on the congestion and contention of the operations. We want to achieve this by considering for each $S_{r,j}$ with $r \geq 1$ (let $S_{1,1}$ denote S_1 and $S_{2,1}$ denote S_2) the following problems for every machine:

- For every $i \leq r+1$, we want to bound the contention caused by T_i-operations, and
- for all $(< \frac{D}{\log^{r+1} D})$-operations we want to bound the congestion in any $\frac{D}{\log^r D}$-interval.

For this, let us introduce the following notation. For any time interval I and schedule S, let $S|_I$ denote S restricted to I (that is, $S|_I$ only contains operations that start in I). Fix some machine M and schedule S. Then the T_i-*congestion* C_i^S at M is defined as the congestion at M caused by T_i-operations in S. $C_{\geq i}^S$ denotes the congestion in S caused by all T_j-operations with $j \geq i$. Furthermore, the T_i-*contention* c_i^S at M is defined as the contention at M caused by T_i-operations in S.

In order to obtain the properties above, we prove the following claim.

Claim 2.9 *After the last refinement we end up with a schedule S such that for every machine it holds:*

(1) For every $i \geq 1$, $c_i^S = O(\frac{C}{D} + \log\log D)$, and

(2) for every $r \geq 2$, $C_{\geq r}^{S|_F} = O(\frac{C+D}{\log^{r-2} D})$ for every $\frac{D}{\log^{r-2} D}$-frame F in S.

This implies that $\alpha = O(\log\log D)$ and $\beta, \gamma = O(1)$. Therefore, it remains to show Claim 2.9.

Our refinement strategy above was chosen such that for every refinement step r we can basically use the same analysis for bounding the congestion and contention. Hence, let us consider in the following some fixed $r \geq 1$, and let us assume that for all $r' < r$ the claims below have already been shown to be true. Since our refinement strategy allows us to refine each frame of size $\frac{D}{\log^{r-2} D}$ in S_{r-1} independently, let us consider for the rest of the proof some fixed frame F of this size. Our goal is to refine schedule $S'_{r-1} = S_{r-1}|_F$ to some new schedule S'_r that is then used together with the refined schedules of other frames as described above to construct some schedule S_r.

2.5.1 Bounding the congestion

Let F, S'_{r-1} and S'_r be defined as above. Let $\mathcal{I}_j$ denote the set of all possible time intervals of length $\frac{D}{\log^j D}$ in F, and let $\ell_i = \frac{D}{\log^{i-1} D}$ denote twice the largest possible length of a T_i-operation. The following claim bounds the congestion of S'_r in terms of the congestion of S'_{r-1}.

Claim 2.10 *Let $\hat{C}_i^{r-1} = \max_{I \in \mathcal{I}_{r-1}} C_i^{S'_{r-1}|_I}$, and let $\hat{C}_i^r = \max_{I \in \mathcal{I}_r} C_i^{S'_r|_I}$. There is a schedule S'_r such that (in addition to the contention bounds stated in Claim 2.12) it holds for every $i \geq r+2$ and every machine:*

(1) If $\hat{C}_i^{r-1} \geq \ell_i \log^4 D$ then $\hat{C}_i^r = (1 + O(\frac{1}{\log D}))\frac{\hat{C}_i^{r-1}}{\log D}$.

(2) If $\ell_i \log^3 D \leq \hat{C}_i^{r-1} < \ell_i \log^4 D$ then $\hat{C}_i^r = (1 + O(\frac{1}{\sqrt{\log D}}))\frac{\hat{C}_i^{r-1}}{\log D}$.

(3) If $\ell_i \log^2 D \leq \hat{C}_i^{r-1} < \ell_i \log^3 D$ then $\hat{C}_i^r = O(\frac{\hat{C}_i^{r-1}}{\log D})$.

(4) If $\ell_i \log D \leq \hat{C}_i^{r-1} < \ell_i \log^2 D$ then $\hat{C}_i^r = O(\ell_i \log D)$.

(5) If $\hat{C}_i^{r-1} < \ell_i \log D$ then $\hat{C}_i^r \leq \hat{C}_i^{r-1}$.

Furthermore, it holds:

(6) $\max_{I \in \mathcal{I}_{r-1}} C_{r+1}^{S'_r|_I} \leq 2 \max_{I \in \mathcal{I}_{r-1}} C_{r+1}^{S'_{r-1}|_I}$ *and*

(7) $C_r^{S'_r} = C_r^{S'_{r-1}}$.

Obviously, Claim 2.10 (7) is always true, as in both cases we sum up the lengths of exactly the same set of operations. Claim 2.10 (5) and (6) are also true, since the delays for S'_r are chosen from the range $[0, \frac{D}{\log^{r-1} D} - \frac{D}{\log^r D} - 1]$. So it remains to prove Claim 2.10 (1)–(4). This will be done later.

Assume that Claim 2.10 (1)–(4) are true. Then we can show the following lemma.

Lemma 2.11 *For every $\frac{D}{\log^r D}$-interval I in S'_r and every machine, $C_{\geq r+2}^{S'_r|_I} = O(\frac{C+D}{\log^r D})$.*

Proof. Fix some machine M. We start by showing that for every $r \geq 1$ and $i \geq r+2$ it holds for every $\frac{D}{\log^r D}$-interval I in S'_r that

$$C_i^{S'_r|_I} = O\left(\frac{C_i^{S_0}}{\log^r D} + \ell_i \log D\right) .$$

Let us assume that $C_i^{S_0} = \ell_i \log^\alpha D$ for some $\alpha \geq 0$. Then we get for every $1 \leq r \leq \alpha - 3$ and every $\frac{D}{\log^r D}$-interval I in S'_r with the help of Claim 2.10 (1) that

$$C_i^{S'_r|_I} = \left(1 + O\left(\frac{1}{\log D}\right)\right)^r \frac{C_i^{S_0}}{\log^r D} .$$

For $\alpha - 3 < r \leq \alpha - 2$ and every $\frac{D}{\log^r D}$-interval I in S'_r, this yields with Claim 2.10 (2)

$$C_i^{S'_r|_I} = \left(1 + O\left(\frac{1}{\log D}\right)\right)^{r-1} \left(1 + O\left(\frac{1}{\sqrt{\log D}}\right)\right) \frac{C_i^{S_0}}{\log^r D} ,$$

and for $\alpha - 2 < r \leq \alpha - 1$ and every $\frac{D}{\log^r D}$-interval I in S'_r, this yields with Claim 2.10 (3)

$$\begin{aligned} C_i^{S'_r|_I} &= \left(1 + O\left(\frac{1}{\log D}\right)\right)^{r-2} \left(1 + O\left(\frac{1}{\sqrt{\log D}}\right)\right) O\left(\frac{C_i^{S_0}}{\log^r D}\right) \\ &= O\left(\frac{C_i^{S_0}}{\log^r D}\right) . \end{aligned}$$

Finally, for $\alpha - 1 < r \leq \alpha$ we get with Claim 2.10 (4) that $C_i^{S'_r|_I} = O\left(\ell_i \log D\right)$.

So for every $r \geq 1$ and every $\frac{D}{\log^r D}$-interval I in S'_r, $C_{\geq r+2}^{S'_r|_I}$ is bounded by

$$\begin{aligned} \sum_{i \geq r+2} C_i^{S'_r|_I} &= \sum_{i \geq r+2} O\left(\frac{C_i^{S_0}}{\log^r D} + \ell_i \log D\right) \\ &= \frac{1}{\log^r D} \sum_{i \geq r+2} O(C_i^{S_0}) + \log D \sum_{i \geq r+2} O\left(\frac{D}{\log^{i-1} D}\right) \\ &= O\left(\frac{C+D}{\log^r D}\right) . \end{aligned}$$

□

Hence, for every machine M, the total congestion in S'_r caused by $(< \frac{D}{\log^{r-1} D})$-operations in frame F is bounded by

$$\underbrace{(\log D)^2 \cdot O\left(\frac{C+D}{\log^r D}\right)}_{(< \frac{D}{\log^{r+1} D})\text{-operations}} + \underbrace{(\log D) \cdot O\left(\frac{C+D}{\log^{r-1} D}\right)}_{T_{r+1}\text{-operations}} + \underbrace{O\left(\frac{C+D}{\log^{r-2} D}\right)}_{T_r\text{-operations}} = O\left(\frac{C+D}{\log^{r-2} D}\right) .$$

The first term in the left hand side comes from Lemma 2.11 together with the fact that there are roughly $(\log D)^2$ disjoint intervals of length $D/\log^r D$ in frame F. The second term uses Lemma 2.11 and Claim 2.10 (6), and the third term uses Lemma 2.11, Claim 2.10 (6) and Claim 2.10 (7). Hence, under the assumption that Claim 2.10 is correct, Claim 2.9 (2) follows. It remains to prove Claim 2.9 (1).

2.5.2 Bounding the contention

In order to bound the contention, we will prove the following claim.

Claim 2.12 *There is a schedule S'_r such that (in addition to the congestion bounds stated in Claim 2.10)*

(1) $c_{r+1}^{S'_r} = O(\frac{C}{D} + \log\log D)$,

(2) $c_r^{S'_r} = O(c_r^{S'_{r-1}})$, *and*

(3) $c_i^{S'_r} \leq 2c_i^{S_i}$ *for all* $i \leq r-1$.

Claim 2.12 (3) is easy to show. Since the total amount of time steps by which the T_i-operations move for the refinements following schedule S_i is bounded by

$$\sum_{j>i} \left(\frac{D}{\log^{j-1} D} - \frac{D}{\log^j D}\right) < \frac{D}{\log^i D} = \frac{\ell_i}{\log D} ,$$

the contention of these operations can at most double afterwards. Suppose that Claim 2.12 (1) and (2) are also true. Then we can show the following lemma.

Lemma 2.13 *After the last refinement, for every $i \geq 1$ and machine, the contention among the T_i-operations is bounded by $O(\frac{C}{D} + \log\log D)$.*

Proof. Using Claim 2.12 it follows that, for any $r \geq 1$ and $i \leq r+1$, the contention among the T_i-operations is bounded by $O(\frac{C}{D} + \log\log D)$ in S_r. We mentioned earlier that we want to stop the refinements after for the first time frames of size at most $\log^3 D$ have been refined, that is, $\frac{D}{\log^{r-2} D} \leq \log^3 D$. In this case, T_{r+1} represents a set of t-operations with $t \in \{1, \ldots, \frac{1}{2}\log D\}$, and since their contention is bounded because of Claim 2.12 (1), the lemma follows. □

So if Claim 2.12 is true, then Claim 2.9 (1) follows.

2.5.3 Proofs of Claim 2.10 and Claim 2.12

Recall that we want to show that on frame F of length $D/\log^{r-2} D$ we can refine schedule S'_{r-1} to obtain a schedule S'_r that satisfies certain bounds on the congestion and contention. Our proof will use the general form of the Lovász Local Lemma. We associate bad events with each machine. For some fixed machine M, A_i denotes the event that the congestion of T_i-operations is too high in S'_r, and B_i denotes the event that the contention of T_i-operations is too high. We first bound the probabilities for the A_i and B_i events.

Let us consider some event A_i with $i \geq r+2$, denoted by A in the following. We distinguish between the following two cases:

- $\hat{C}_i^{r-1} \geq \ell_i \log^2 D$: Consider some fixed $\frac{D}{\log^{r-1} D}$-interval I in S'_{r-1}. The interval composed of the last $\frac{D}{\log^r D}$ time steps of I is called the *tail* of I. Let all T_i-operations on M that start in I be numbered from 1 to a. For every $j \in \{1, \ldots, a\}$, let the random variable X_j be t if operation j is a t-operation and chooses a delay such that it starts during the tail of I, and 0 if operation j chooses a delay such that it starts either before or after the tail of I. (Note that all operations that might be processed during the tail of I in S'_r start in S'_{r-1} during I.) Let $X = \sum_{j=1}^{a} X_j$. Since the jobs choose their delays independently and uniformly at random from a range of $[0, \frac{D}{\log^{r-1} D} - \frac{D}{\log^r D} - 1]$, we get

$$\Pr[X_j \neq 0] \leq \frac{D/\log^r D}{D/(\log^{r-1} D) - D/(\log^r D)} = \frac{1}{\log D - 1}$$

  for every j. Hence, $\mathrm{E}[X] \leq \hat{C}_i^{r-1}/(\log D - 1)$. Let $\mu = \hat{C}_i^{r-1}/(\log D - 1)$ and $\delta = \min[\sqrt{\hat{C}_i^{r-1}/(\ell_i \log^2 D)},\ \log D]$. Since the operations considered for A have a length of at most $k = \ell_i/2$, we get together with the Chernoff-Hoeffding bounds that, for $\epsilon = \frac{3}{\delta}$ if $\frac{3}{\delta} \leq 1$ and $\epsilon = (\frac{3}{\delta})^2$ otherwise,

$$\Pr\left[X \geq (1+\epsilon)\mu\right] \leq \mathrm{e}^{-(3/\delta)^2 \frac{\hat{C}_i^{r-1}}{\log D - 1}/3k} \leq \mathrm{e}^{-6\max\left[\log D,\ \frac{\hat{C}_i^{r-1}}{\ell_i \log^3 D}\right]}.$$

  In this bound, we replace the expression $\hat{C}_i^{r-1}/(\ell_i \log^3 D)$ as follows. Let m_A denote the number of T_i-operations on M in the frame F. Since the smallest T_i-operations have size $\ell_i/\log D$ and frame F has $\log D$ disjoint intervals of size $D/(\log n)^{r-1}$, we have that $m_A \leq (\log D \cdot \hat{C}_i^{r-1})/(\ell_i/\log D)$ and therefore $\hat{C}_i^{r-1} \geq m_A \cdot \ell_i/\log^2 D$. We conclude that for ϵ as defined in the two cases above

$$\Pr\left[X \geq (1+\epsilon)\frac{\hat{C}_i^{r-1}}{\log D - 1}\right] \leq \mathrm{e}^{-6\max\left[\log D,\ m_A/\log^5 D\right]}.$$

  Let us define the event A to be true if $X \geq (1+\epsilon)\frac{\hat{C}_i^{r-1}}{\log D - 1}$ for some $\frac{D}{\log^{r-1} D}$-interval in S'_{r-1}. Since at most $2\frac{D}{\log^{r-2} D}$ different $\frac{D}{\log^{r-1} D}$-intervals have to be considered, we get

$$\Pr[A] \leq \min\left[\frac{1}{D^5},\ D \cdot \mathrm{e}^{-5m_A/\log^5 D}\right].$$

- $\ell_i \log D \leq C_i^{r-1}(I) < \ell_i \log^2 D$: Let X, μ and k be defined as before. Then we get with $\epsilon = \frac{6\log D}{\mu/k}$ that

$$\Pr[X \geq (1+\epsilon)\mu] \leq \left(\frac{\mathrm{e}^{\epsilon}}{(1+\epsilon)^{1+\epsilon}}\right)^{\mu/k} \leq \mathrm{e}^{-6\log D} \leq D^{-6}.$$

  The rest is similar to above to obtain $\Pr[A] \leq \frac{1}{D^5}$.

Next we bound the probabilities for the contention events. For some fixed machine M and $i \in \{r, r+1\}$, let us consider the respective event B_i, which we will denote by B in the following.

First, we consider the case that $i = r+1$. Consider some fixed $\frac{D}{\log^{r-1} D}$-interval I in S'_{r-1}, and let t denote its last time step. Let O_I denote the set of all T_{r+1}-operations on M that start in I. (Note that O_I contains all T_{r+1}-operations that might be processed at time step t in schedule S'_r.) Since each unit of processing of an operation in O_I has a probability of at most $1/(\frac{D}{\log^{r-1} D} - \frac{D}{\log^r D})$ to be executed at time t, the probability that $k = \mathrm{e}(\mathrm{e}+1) \max[\hat{C}_{r+1}^{r-1}/|I|,\ \log\log D]$ operations are executed at time t is bounded by

$$\binom{\hat{C}_{r+1}^{r-1}}{k} \left(\frac{1}{\frac{D}{\log^{r-1} D} - \frac{D}{\log^r D}} \right)^k \leq \left(\frac{\mathrm{e}\hat{C}_{r+1}^{r-1}}{k\left(1 - \frac{1}{\log D}\right)|I|} \right)^k \leq \mathrm{e}^{-k} \ .$$

Let us define B to be true if this contention bound is true for the last time step of every $\frac{D}{\log^{r-1} D}$-interval I in S'_{r-1}. Using Lemma 2.11 as an inductive hypothesis for refinement step $r-1$, $\hat{C}_{r+1}^{r-1} = O(\frac{C+D}{\log^{r-1} D})$. Thus,

$$k = O\left(\frac{C+D}{\log^{r-1} D} \cdot \frac{1}{|I|} + \log\log D \right) = O(C/D + \log\log D) \ ,$$

as desired. Since the smallest operations in B have a length of at least $\ell_{r+1}/\log D$, the total number m_B of T_{r+1}-operations on M within frame F is at most $(\log D \cdot \hat{C}_{r+1}^{r-1})/(\ell_{r+1}/\log D)$ and therefore,

$$\hat{C}_{r+1}^{r-1} \geq \frac{m_B \cdot \ell_{r+1}}{\log^2 D} = m_B \cdot \frac{D}{\log^{r+2} D} \ .$$

Hence, the probability that B is true w.r.t. I is at most $\mathrm{e}^{-10\max[\log\log D,\ m_B/\log^3 D]}$. Since at most $2\log^3 D$ time steps have to be checked to be sure that nowhere in S'_{r-1} the contention caused by T_{r+1}-operations exceeds k, we get

$$\Pr[B] \leq \min\left[\frac{1}{\log^{10} D},\ 2\log^3 D \cdot \mathrm{e}^{-10 m_B/\log^3 D} \right] \ .$$

The calculations for the case that event B considers T_r-operations are similar: If we assume Claim 2.12 (1) to be true for these operations after refinement step $r-1$ and choose $k = O(\max[\hat{C}_r^{r-1}/\frac{D}{\log^{r-1} D},\ \log\log D])$ large enough, then we end up with the same probability and contention bounds as above.

Next we bound the dependencies among the A_i-events and the B_j-events. Whether or not a bad event A_i occurs depends solely on the delays assigned to the jobs that have a T_i-operation on the respective machine M. Each of these m_{A_i} jobs may have operations that influence up to D other A_j on other machines. Hence, each A_i depends on at most $m_{A_i} D$ other A_j-events. Furthermore, every such job may also have operations that influence at most $\log^3 D$ B_j-events on other machines. This holds, since schedule S'_{r-1} is based on a frame of size $\frac{D}{\log^{r-2} D}$ in S_{r-1} and the lengths of the operations influencing B_j-events are at least $\frac{D}{\log^{r+1} D}$. Hence, the maximum dependence of an A_i-event on B_j-events is bounded by $m_{A_i} \log^3 D$.

Whether or not a bad event B_i occurs also depends solely on the delays assigned to the jobs that have a T_i-operation on the respective machine M. Since each of these m_{B_i} jobs may have operations that influence up to $\log^3 D$ other B_j, each B_i depends on at most $m_{B_i} \log^3 D$ other B_j-events. Furthermore, every such job may also influence up to D A_j-events. Hence, the maximum dependence of a B_i-event on A_j-events is bounded by $m_{B_i} D$.

Before we use the Lovász Local Lemma, we state a simple claim

Claim 2.14 *For any* $0 \leq x \leq 1/2$, $1 - x \geq 2^{-2x}$.

Proof. Since $1 - 2x + 2x^2 \geq \mathrm{e}^{-2x}$ and $1 - x \geq 1 - 2x + 2x^2$ if and only if $0 \leq x \leq 1/2$, the claim follows. $\square$

Now we are ready to apply the Lovász Local Lemma. We will assume that D is sufficiently large. For every A_i, choose p_{A_i} as $p_{A_i} = \frac{1}{D^3}$, and for every B_i, choose p_{B_i} as $p_{B_i} = \frac{1}{\log^9 D}$. Let G be the dependency graph of the A_i-events and B_j-events. Then it holds for every A_i that

$$\begin{aligned} p_{A_i} \prod_{\{A_i,A_j\}\in G} (1 - p_{A_j}) \prod_{\{A_i,B_j\}\in G} (1 - p_{B_j}) &\geq \frac{1}{D^3}\left(1 - \frac{1}{D^3}\right)^{m_{A_i} D} \left(1 - \frac{1}{\log^9 D}\right)^{m_{A_i} \log^3 D} \\ &> \frac{1}{D^3} \cdot \mathrm{e}^{-2m_{A_i}/D^2} \cdot \mathrm{e}^{-2m_{A_i}/\log^6 D} \geq \Pr[A_i] , \end{aligned}$$

because for $m_{A_i} \leq \log^6 D$ we have

$$\frac{1}{D^3} \cdot \mathrm{e}^{-2m_{A_i}/D^2} \cdot \mathrm{e}^{-2m_{A_i}/\log^6 D} \geq \frac{1}{D^5}$$

and for $m_{A_i} > \log^6 D$ we have

$$\frac{1}{D^3} \cdot \mathrm{e}^{-2m_{A_i}/D^2} \cdot \mathrm{e}^{-2m_{A_i}/\log^6 D} \geq D \cdot \mathrm{e}^{-5m_{A_i}/\log^5 D} .$$

For every B_i we further get that

$$\begin{aligned} p_{B_i} \prod_{\{B_i,A_j\}\in G} (1 - p_{A_j}) \prod_{\{B_i,B_j\}\in G} (1 - p_{B_j}) &\geq \frac{1}{\log^9 D}\left(1 - \frac{1}{D^3}\right)^{m_{B_i} D} \left(1 - \frac{1}{\log^9 D}\right)^{m_{B_i} \log^3 D} \\ &> \frac{1}{\log^9 D} \cdot \mathrm{e}^{-2m_{B_i}/D^2} \cdot \mathrm{e}^{-2m_{B_i}/\log^6 D} \geq \Pr[B_i] , \end{aligned}$$

because for $m_{B_i} \leq \log^4 D$ we have

$$\frac{1}{\log^9 D} \cdot \mathrm{e}^{-2m_{B_i}/D^2} \cdot \mathrm{e}^{-2m_{B_i}/\log^6 D} \geq \frac{1}{\log^{10} D}$$

and for $m_{B_i} > \log^4 D$ we have

$$\frac{1}{\log^9 D} \cdot \mathrm{e}^{-2m_{B_i}/D^2} \cdot \mathrm{e}^{-2m_{B_i}/\log^6 D} \geq 2\log^3 D \cdot \mathrm{e}^{-10m_{B_i}/\log^3 D} .$$

Hence, according to the Lovász Local Lemma, there exists a set of delays that fulfills the requirements of Claims 2.10 and 2.12. So a refinement from S_{r-1} to S_r is possible as we claimed it, which proves Lemma 2.7. $\square$

2.6 Proof of Item 2 of Theorem 1.1

As before, let us assume that $C \geq D$. In order to simplify the proof, we first consider the situation that there are only two different operation lengths, P_1 and P_2, with $P_1 > P_2$ (recall that w.l.o.g. we can assume that P_1 and P_2 are powers of 2). Our strategy for constructing an efficient schedule is to successively refine the (possibly illegal) well-structured schedule S_0 that runs all jobs in parallel. Let $I_0 \cdot P_1$ with $I_0 = D/P_1$ be the frame size to be refined in schedule S_0 and $I_j^3 P_1$ with $I_j = \log I_{j-1}$ for all $j \geq 1$ be the frame size to be refined in schedule S_j. Our aim is to show that for all $j \geq 1$ with $I_j \geq 36$ there exists a refinement from schedule S_{j-1} to a well-structured schedule S_j such that the congestion at any machine for any frame of size $I_j^2 P_1$ is bounded by $O(\frac{C}{D} \cdot I_j^2 P_1)$. (The condition $I_j \geq 36$ was only chosen to simplify the proof.) So when $\log I_j < 36$ for the first time then the contention among the P_1-operations must be $O(C/D)$. Furthermore, the congestion caused by P_2-operations in any P_1-interval must be bounded by $O(\frac{C}{D} \cdot P_1)$. Our strategy therefore is to break schedule S_j into sub-schedules of length P_1 and continue to schedule each sub-schedule independently, using the same refinement strategy for the P_2-operations as before for the P_1-operations. At the end we want to arrive at a well-structured schedule of length $O(D)$ in which both P_1-operations and P_2-operations have a contention of $O(C/D)$. This schedule can then be transformed into a legal schedule of length $O(C+D)$ by using the strategy in the proof of Lemma 2.4 (1).

Let us describe now in detail, how to do the refinements. In the following, set $d_1 = D/P_1$ and $d_2 = D/P_2$. The first step is to assign an initial delay to each job, chosen independently and uniformly at random from $\{iP_1 \mid i \in [0, d_1 - 1]\}$. In the resulting schedule, S_1, a job that is assigned a delay of δ waits for δ steps and then is processed without waiting again until it is completed. The length of S_1 is at most $2D$. We show that if the delays are chosen independently and uniformly at random and I_1 is sufficiently large, then with nonzero probability the congestion at any machine in any time interval of size $I_1^2 P_1$ starting at an integer multiple of P_1 is at most $C_1 = \frac{C}{D}(1 + \frac{4}{\sqrt{I_1}}) I_1^2 P_1$ for both P_1- and P_2-operations. (Since the delays are multiples of P_1 and S_0 is well-structured, we only have to consider these $I_1^2 P_1$-intervals for our refinement step.) Thus, such a set of delays must exist.

To apply the Lovász Local Lemma, we associate two bad events with each machine. Fix some machine M. For $i \in \{1, 2\}$, the bad event $A_{M,i}$ for the P_i-operations at M is that there is some $I_1^2 P_1$-interval starting at an integer multiple of P_1 with a congestion of more than C_1. For this we have to bound the dependence among the bad events and the probability that a bad event occurs.

Let us start with bounding the probability. Consider some fixed event $A_{M,i}$, denoted by A, and $I_1^2 P_1$-interval I. Let m_i be the number of P_i-operations on M. For every $j \in \{1, \ldots, m_i\}$, let the random variable X_j be P_i if operation j is started in I and 0 otherwise. Let $X = \sum_{j=1}^{m_i} X_j$. Since the jobs choose their delays uniformly at random from $\{iP_1 \mid i \in [0, d_1 - 1]\}$, we get $\Pr[X_j = P_i] \leq I_1^2 / d_1$ for all j. Hence,

$$\mathrm{E}[X] = \sum_{j=1}^{m_i} \mathrm{E}[X_j] \leq \frac{C}{D} \cdot I_1^2 P_1 \ .$$

Let $c_1 = C/P_1$ and $c_2 = C/P_2$. Together with the Chernoff-Hoeffding bounds we get with $\epsilon = \frac{4}{\sqrt{I_1}}$ (note that $I_1 \geq 36$) that

$$\begin{aligned} \Pr[X \geq C_1] &= \Pr\left[X \geq (1+\epsilon)\tfrac{C}{D} \cdot I_1^2 P_1\right] \\ &\leq \mathrm{e}^{-\left(\frac{4}{\sqrt{I_1}}\right)^2 \frac{C}{D} I_1^2 P_1 / 3P_i} = \mathrm{e}^{-5 I_1 \frac{C/P_i}{D/P_1}} \\ &= \mathrm{e}^{-5 I_1 c_i / d_1} \ . \end{aligned}$$

The total number of $I_1^2 P_1$-intervals to be considered is at most $2d_1$. Hence, the probability that a bad event A occurs is bounded by

$$2d_1 \cdot \mathrm{e}^{-5 I_1 c_i / d_1} = 2d_1 \cdot \mathrm{e}^{-5 (\log d_1) c_i / d_1} \leq d_1^{-5 c_i / d_1} \ .$$

Now, we bound the dependencies among the events. Whether or not a bad event $A_{M,1}$ occurs depends solely on the delays assigned to the jobs whose operations influence $A_{M,1}$. Since each of the at most c_1 jobs regarded by $A_{M,1}$ may have operations that influence up to d_1 other $A_{M',1}$, each $A_{M,1}$ depends on at most $c_1 d_1$ other $A_{M',1}$-events. Furthermore, every job associated with $A_{M,1}$ may also have operations that influence up to d_2 $A_{M',2}$-events. Hence the maximum dependence of an $A_{M,1}$-event on $A_{M',2}$-events is bounded by $c_1 d_2$. Similarly, each $A_{M,2}$ depends on at most $c_2 d_2$ other $A_{M',2}$-events and on at most $c_2 d_1$ other $A_{M',1}$-events.

Now we are ready to apply the Lovász Local Lemma. Recall that, since $C \geq D$, $c_1 \geq d_1$ and $c_2 \geq d_2$. Let $p_1 = d_1^{-2c_1/d_1}$ and $p_2 = d_1^{-2c_2/d_1}$. Let G be defined as the dependency graph of the $A_{M,1}$-events and $A_{M,2}$-events. Then it holds for every $A_{M,1}$ that

$$\begin{aligned} & p_1 \prod_{\{A_{M,1},A_{M',1}\}\in G} (1-p_1) \prod_{\{A_{M,1},A_{M',2}\}\in G} (1-p_2) \\ \geq\ & d_1^{-2c_1/d_1}\left(1-d_1^{-2c_1/d_1}\right)^{c_1 d_1}\left(1-d_1^{-2c_2/d_1}\right)^{c_1 d_2} \\ >\ & d_1^{-2c_1/d_1}\cdot \mathrm{e}^{-2d_1^{-2c_1/d_1}c_1 d_1}\cdot \mathrm{e}^{-2d_1^{-2c_2/d_1}c_1 d_2} \\ \geq\ & d_1^{-2c_1/d_1}\cdot \mathrm{e}^{-2}\cdot \mathrm{e}^{-2c_1/d_1} \geq \Pr\left[A_{M,1}\right] \ , \end{aligned}$$

because

$$d_1^{-2c_1/d_1}\cdot c_1 d_1 \leq 1 \quad \Leftrightarrow \quad 2c_1 \geq d_1\left(1+\log_{d_1} c_1\right) \ ,$$

which is true for all $c_1 \geq d_1$, and

$$d_1^{-2c_2/d_1}\cdot c_1 d_2 \leq c_1/d_1 \quad \Leftrightarrow \quad 2c_2 \geq d_1\left(1+\log_{d_1} d_2\right) \ ,$$

which is also true since $c_2 \geq d_2 \geq d_1$. For every $A_{M,2}$ we further get that

$$\begin{aligned} & p_2 \prod_{\{A_{M,2},A_{M',1}\}\in G} (1-p_1) \prod_{\{A_{M,2},A_{M',2}\}\in G} (1-p_2) \\ \geq\ & d_1^{-2c_2/d_1}\left(1-d_1^{-2c_1/d_1}\right)^{c_2 d_1}\left(1-d_1^{-2c_2/d_1}\right)^{c_2 d_2} \\ >\ & d_1^{-2c_2/d_1}\cdot \mathrm{e}^{-2d_1^{-2c_1/d_1}c_2 d_1}\cdot \mathrm{e}^{-2d_1^{-2c_2/d_1}c_2 d_2} \\ \geq\ & d_1^{-2c_2/d_1}\cdot \mathrm{e}^{-2c_2/d_1}\cdot \mathrm{e}^{-2} \geq \Pr\left[A_{M,2}\right] \ , \end{aligned}$$

because

$$d_1^{-2c_1/d_1}\cdot c_2 d_1 \leq c_2/d_1 \quad \Leftrightarrow \quad c_1 \geq d_1 \ ,$$

which is true, and

$$d_1^{-2c_2/d_1}\cdot c_2 d_2 \leq 1 \quad \Leftrightarrow \quad 2c_2 \geq d_1\left(1+\log_{d_1} d_2\right) \ ,$$

which is also true since $c_2 \geq d_2 \geq d_1$. Hence, there is an assignment of delays to the jobs such that a refinement from schedule S_0 to S_1 is possible such that the congestion at any machine M_i in any $I_1^2 P_1$-interval is bounded by C_1.

Next, we break schedule S_1 into frames of size $I_1^3 P_1$ and schedule each frame independently. So each frame can be viewed as a separate scheduling problem where the jobs associated with a frame are those that have operations scheduled by S_1 within the frame, and each such job contains only its operations scheduled within that frame. For each frame, our refinement step will choose for each associated job a suitable initial delay. Schedule S_2 then results from executing the modified frames one after the other.

Consider some fixed $I_1^3 P_1$-frame F in S_1. Let each job randomly choose an (additional) initial delay in the range $\{iP_1 \mid i \in [0, I_1^2 - 1]\}$. Then the length of the resulting schedule for this frame is at most $I_1^3 P_1 + I_1^2 P_1 = (1 + \frac{1}{I_1}) I_1^3 P_1$. It is not difficult to see that we can use the Lovász Local Lemma in a similar way as above to show that if the delays are chosen independently and uniformly at random, then with nonzero probability the congestion in any $I_2^2 P_1$-interval in F starting at an integer multiple of P_1 is at most

$$C_2 = \frac{C}{D}\left(1 + \frac{4}{\sqrt{I_1}}\right)\left(1 + \frac{6}{\sqrt{I_2}}\right)\left(\frac{1}{1 - (I_2/I_1)^2}\right) I_2^2 P_1$$

for both P_1- and P_2-operations. Thus, such a set of delays must exist. (The analysis is a combination of the analysis above and the second refinement step in the proof of Lemma 2.6).

Continuing with the refinements until $\log I_j < 36$ completes the refinements for the P_1-operations. The processing of the P_2-operations is afterwards refined similar to Lemma 2.6. This proves the theorem for two different operation lengths.

Extending this refinement technique from two different operation lengths to any constant number of operation lengths by using it again and again from the largest to the smallest operations proves item 2 of Theorem 1.1.

3 Lower Bounds

We first exhibit an instance of general JSS for which it is relatively simple to show that $L = \Omega(lb \log lb / \log\log lb)$.

3.1 General JSS

Theorem 3.1 *For any integers $1 < k < r$, there is an instance of job shop scheduling with k jobs, k machines, dilation and congestion each equal to $D = kr^{k-1}$, such that any legal schedule is of length more than $Dk(1 - \frac{k-1}{r-1})$.*

Proof. We describe the JSS instance for arbitrary integers $1 < k < r$. There are k machines. Each job is composed of repetitions of the sequence of operations $M_1, M_2, \ldots, M_k$. For job i, $1 \leq i \leq k$, each operation requires time r^{i-1}, and the sequence as a whole is repeated r^{k-i} times. Hence, the length of each job (the dilation) is $r^{i-1} \cdot k \cdot r^{k-i} = kr^{k-1}$, which we denote by D. The congestion (the total demand for any individual machine) is also D, as each job requires time D/k on each machine.

Fix an arbitrary legal schedule S for the jobs, without preemption. We shall show that the length of the schedule must be more than $Dk(1 - \frac{k-1}{r-1})$.

Consider two arbitrary jobs, J_i and J_j. Relative to the schedule S, we define the overlap $O(i,j)$ as the number of time units in which both J_i and J_j perform operations.

Proposition 3.2 *For $1 \leq i < j \leq k$,*

$$O(i,j) \leq \frac{D(k-1)}{r^{j-i}}$$

Proof. During the execution of an operation of J_j on machine M, job J_i can complete at most $k-1$ operations (as operations on machine M have to wait). Each operation of J_i is shorter than the operation of J_j by a factor of r^{j-i}. □

We now use the first two terms of the exclusion-inclusion formula to bound $|S|$, the total length of schedule S. We obtain, $|S| \geq kD - \sum_{1 \leq i < j \leq k} O(i,j)$. But

$$\sum_{1 \leq i < j \leq k} O(i,j) = \sum_{2 \leq j \leq k} \sum_{1 \leq i < j} O(i,j) \leq \sum_{2 \leq j \leq k} \sum_{1 \leq i < j} \frac{D(k-1)}{r^{j-i}} < \sum_{1 \leq j \leq k} \frac{D(k-1)}{r-1} = \frac{k(k-1)D}{r-1}$$

Hence, $|S| > kD(1 - \frac{k-1}{r-1})$. $\square$

Observe that already for the case of three jobs and three machines, when r is sufficiently large we get that every job and every machine is idle for most of the duration of any legal schedule. Also, when $r \simeq k \log k$, then the length of the shortest schedule is nearly kD, and $k \simeq \log D / \log \log D$. This proves Theorem 1.3.

If preemption is allowed, then the jobs in the proof of Theorem 3.1 can be scheduled within $O(D)$ steps. This is a consequence of item 3 in Lemma 2.4.

3.2 Acyclic JSS

Here we prove Theorem 1.4, which is a direct consequence of Theorem 3.7 below.

Throughout this section we assume that m is sufficiently large. In order to avoid excessive use of floor and ceiling notation, we treat all functions of m (such as $\log m, \sqrt{m}$, $5m/6 \log m$, etc.) as if they give integer values. (The effect of rounding fractional values of these functions to the nearest integer is negligible for the purpose of our lower bound, when m is sufficiently large.) We start with preliminary definitions and a lemma.

Definition 3.3 *A* bucket configuration $B(m, k)$ *is a collection of* 2^k *buckets, where each bucket contains* $m - m/(\log m)^2$ *of the* m *machines.*

Definition 3.4 *A* partial cover *of job* $J = M_{i_1}, M_{i_2}, \ldots, M_{i_l}$ *by bucket configuration* $B(m, k)$ *is the assignment of the buckets of* $B(m, k)$ *to nonoverlapping intervals within* $i_1, \ldots i_l$*, such that within each interval, all operations of job J are on machines that are in the respective bucket. Job J* avoids *bucket configuration* $B(m, k)$ *if for any partial cover of J by* $B(m, k)$*, the number of operations of J that remain uncovered is at least* $l/2$*.*

Definition 3.5 *A* bucket avoiding family of jobs $F(l, m, k)$ *is a collection of* $m/\log m$ *jobs, where each job contains* 2^l *operations and the collection has the following properties:*

(1) Each machine participates in at most $2^{l+1}/\log m$ *jobs.*

(2) Each job is acyclic.

(3) For any bucket configuration $B(m, k)$*, at least half of the jobs avoid it.*

Lemma 3.6 *For* m *sufficiently large, for* $k \leq \frac{\log m}{2} - 4 \log \log m - 1$*, and for* $l = k + 4 \log \log m$*, there is a bucket avoiding family of jobs* $F(l, m, k)$*.*

Proof. We shall use the term *overwhelming* (*negligible*, respectively) to denote a probability that tends to 1 (to 0, respectively) as m tends to infinity.

Consider the following four step randomized procedure for constructing $F(l, m, k)$.

(1) For each of the $m/\log m$ jobs independently, each of the 2^l operations in a job is to be executed on a machine chosen uniformly at random, independently of all other choices.

(2) If there is some machine that participates in more than $2^{l+1}/\log m$ operations, abort the construction.

(3) If less than $5m/6 \log m$ jobs are acyclic, abort.

(4) Leave $5m/6 \log m$ acyclic jobs untouched. The total number of operations in the remaining jobs is $2^l m/6 \log m$. Globally rearrange these operations within the $m/6 \log m$ jobs so that each of these jobs becomes acyclic. (By item 2 above, each machine participates in at most $2^{l+1}/\log m$ operations. Observe that for sufficiently large m, the condition that $l < \frac{\log m}{2}$ implies that $2^{l+1}/\log m < m/6 \log m$. Hence, the rearranging is possible.)

First, let us show that the probability that the construction aborts is negligible.

View each operation as a ball and each machine as a bin. Then choosing an operation at random is the same as putting a ball in a random bin. We have $m2^l/\log m$ balls and m bins. For $l \geq 4\log\log m$ (as we have in the lemma), the expected number of balls per bin is at least $(\log m)^3$, and standard analysis of throwing balls into bins shows that the probability that some bin contains more balls than twice the expectation is negligible. Hence, the probability of aborting in step 2 of the construction is negligible.

Consider any particular job. The probability that it has two operations on the same machine is at most $\binom{2^l}{2}/m$, which is less than $1/8$, as $l \leq \frac{\log m}{2} - 1$. Hence, the expected number of acyclic jobs is at least $7m/8\log m$, and standard bounds on large deviations show that the probability that step 3 aborts is negligible.

We conclude that with overwhelming probability, the construction does not abort. Observe that in this case, conditions 1 and 2 of definition 3.5 hold, and it remains to verify that condition 3 holds with high probability. We shall check a stronger condition after step 1 of the construction, namely, that for any bucket configuration $B(m,k)$, at least two thirds of the jobs avoid it. This implies that condition 3 holds after step 4 of the construction, as only one sixth of the jobs are rearranged.

Fix a particular bucket configuration $B(m,k)$, and consider a job J composed of a sequence of 2^l machines chosen independently at random. If J does not avoid $B(m,k)$, then 2^{l-1} of its operations are covered by the 2^k buckets. Hence, there are 2^k disjoint sequences of $s = 2^{l-k-2}$ consecutive operations such that each one of them is completely covered by one bucket (though the same bucket can cover several such sequences). There are 2^l ways of choosing the starting point of a sequence, and for each sequence there is a choice of 2^k buckets that may cover it. For a particular starting point and a particular bucket, the probability that all of the s random machines in the sequence are among the $m - m/(\log m)^2$ machines in the bucket is $(1-(\log m)^{-2})^s$. Hence, we conclude that the probability that a random job fails to avoid a particular bucket configuration is at most $(2^{k+l}(1-(\log m)^{-2})^s)^{2^k}$. Substituting back in the value of s, setting $l = k + 4\log\log m$, and using the fact that $l \leq \frac{\log m}{2}$, this can be upper bounded by $(2^{2k}2^{4\log\log m}e^{-(\log m)^2+4})^{2^k} < e^{-\alpha(\log m)^2 2^k}$, for some α that can be made arbitrarily close to 1 when m is sufficiently large.

If there are $m/\log m$ random jobs as above, the probability that a third of them fail to avoid the bucket configuration is $2^{-\beta m\log m 2^k}$ for some $\beta > 0$. As there are less than $m^{m2^k/(\log m)^2} = 2^{m2^k/\log m}$ possible bucket configurations, the probability that a third of the jobs fail to avoid some bucket configuration is negligible. This takes care of condition 3 in definition 3.5.

As the randomized construction has overwhelming probability of succeeding, we have shown that a bucket avoiding family $F(l,m,k)$ exists. $\square$

Theorem 3.7 *For a large enough m, there is an instance of acyclic JSS, with m machines, less than m jobs, dilation roughly $\sqrt{m}$, congestion $o(\sqrt{m})$, for which any legal schedule requires at least $\frac{\sqrt{m}\log m}{32\log\log m}$ time steps. Expressed in terms of dilation (in our case, the congestion is much smaller than the dilation), any legal schedule has a length of at least $D\frac{\log D}{16\log\log D}$.*

Proof. Let m be sufficiently large. The instance is composed of $K = (\log m - 2)/4\log\log m + 1$ families of jobs, indexed by $k_1, k_2, \ldots, k_K$, where $k_1 = 1$, and for any $i < K$, $k_{i+1} = k_i + 4\log\log m$. The family k_1 contains $m/\log m$ jobs, where each job has only one operation, the length of each job is $\sqrt{m}$, and the operations are all distinct. For $1 < i \leq K$, family k_i is a bucket avoiding family $F(k_i, m, k_{i-1})$, where the length of each operation is $\sqrt{m}/2^{k_i}$. Hence, family k_i contains $m/\log m$ jobs, each job has 2^{k_i} operations, and the length of each job is $\sqrt{m}$.

Lemma 3.8 *Any legal schedule for the above jobs is of length at least $\frac{\sqrt{m}\log m}{32\log\log m}$.*

Proof. Fix any legal schedule, and assume that it is of length $c\sqrt{m}$, where $c < \frac{\log m}{32\log\log m}$. We shall derive a contradiction.

For each $k_i > 1$, we define a bucket configuration $B(m, k_i - 2)$ by induction relative to the assumed schedule. Each bucket configuration will be associated with $\sqrt{m}/8$ time steps of the schedule in which the jobs in family k_i are active, and the time steps associated with different bucket configurations will be disjoint. This will imply that the schedule must have a length of at least $\frac{\sqrt{m}}{8}\frac{\log m}{4\log\log m} = \sqrt{m}\log m/32\log\log m$.

Base case: For $k_1 = 1$, partition the schedule into $2c$ blocks of consecutive time steps, where each block is of length $\sqrt{m}/2$. As operations of family k_1 take $\sqrt{m}$ time units, each operation completely covers at least one block. As there are $m/\log m$ operations in family k_1, there is at least one block that is covered by at least $\frac{m}{2c\log m}$ operations. Take one such block arbirarily and $m/(\log m)^2$ of the machines that cover it. The bucket associated with k_1 contains the $m - m/(\log m)^2$ other machines. Note that for any $i > 1$, a sequence of operations of a job from family k_i can be scheduled at the same time period in which the block corresponding to the k_1 bucket was scheduled only if all machines that have operations within the sequence are in the bucket.

Inductive step: Suppose we have already defined the bucket configuration for all k_j with $j \leq i$. We shall now define the bucket configuration associated with k_{i+1}.

For each bucket in each bucket configuration up to k_i, mark on the schedule the time steps corresponding to the block that created each bucket. These time steps are disjoint (disjointness of these time steps is maintained by the inductive step). The total number of buckets that we have is $1 + \sum_{j=2}^{i} 2^{k_j - 2} \leq 2^{k_i}$, and each bucket has $m - m/(\log m)^2$ operations. Hence, this can be regarded as (a part of) a bucket configuration $B(m, k_i)$. Recall that the family k_{i+1} was chosen as a bucket avoiding family $F(k_{i+1}, m, k_i)$. This implies that if the schedule is legal, at least half of the jobs of this family avoid the bucket configuration $B(m, k_i)$. Hence, at least a quarter of the operations of the family are performed at time steps that are not covered by previous locations of buckets. Call these operations *good*.

Partition now the time steps of the schedule into $2c2^{k_{i+1}}$ blocks, each of size $\sqrt{m}/2^{k_{i+1}+1}$. Remove any block that overlaps the location of a previous bucket. Each good operation (which is of length $\sqrt{m}/2^{k_{i+1}}$) completely covers one of the remaining blocks. As there are at least $m2^{k_{i+1}}/4\log m$ good operations, and each block can contain at most $m/2\log m$ good operations (this is the number of jobs involved), at least $2^{k_{i+1}}/4$ of the blocks have $m/16c\log m > m/(\log m)^2$ good operations in them. These blocks make the bucket configuration associated with k_{i+1}, where the machines going into each bucket are those machines that do not cover the respective block. The total number of time steps associated with these buckets is $(2^{k_{i+1}}/4)(\sqrt{m}/2^{k_{i+1}+1}) = \sqrt{m}/8$. By construction, the time steps associated with these buckets are disjoint from those associated with previous buckets. □

To complete the proof of Theorem 3.7, observe that by construction, jobs are acyclic. The dilation (length of the longest job) is $\sqrt{m}$. The total number of jobs is $n = Km/\log m \simeq m/4\log\log m$. The average load per machine is $n\sqrt{m}/m < \sqrt{m}/4\log\log m$. By condition 1 in Definition 3.5 it follows that the congestion is at most $\sqrt{m}/2\log\log m$. □

Remark: For the particular instance of JSS that we construct, Lemma 3.8 is best possible up to constant factors. Each family of jobs in our instance can be scheduled to require $O(\sqrt{m})$ time (this follows from the main result of [LMR94], as all operations within a family are of the same length), and there are $O(\log m/\log\log m)$ families. The constant factors can be improved by tightening parameters of the construction (*e. g.*, making $k_{i+1} = k_i + 3\log\log m$, saying that a job avoids a bucket configuration if almost all of its operations remain uncovered) and tightening the analysis, giving a lower bound of $\sqrt{m}\log m/6\log\log m$, or $D\log D/3\log\log D$.

References

[ASE92] N. Alon, J. Spencer, P. Erdős. *The Probabilistic Method*. Wiley Interscience Series in Discrete Mathematics and Optimization, John Wiley & Sons, 1992.

[Be91] J. Beck. An algorithmic approach to the Lovász Local Lemma. *Random Structures and Algorithms* **2**(4), pp. 343-365, 1991.

[GPSS97] L. A. Goldberg, M. Paterson, A. Srinivasan, E. Sweedyk. Better approximation guarantees for job-shop scheduling. In *Proc. of the 8th Symp. on Discrete Algorithms*, pp. 599-608, 1997.

[Ho87] M. Hofri. *Probabilistic Analysis of Algorithms: On Computing Methodologies for Computer Algorithms Performance Evaluation*, Springer Verlag, 1987.

[LMR94] T. Leighton, B. Maggs, S. Rao. Packet routing and job-shop scheduling in O(congestion + dilation) steps. *Combinatorica* 14, pp. 167-186, 1994.

[LMR96] T. Leighton, B. Maggs, A. Richa. Fast algorithms for finding O(congestion + dilation) packet routing schedules. Technical Report CMU–CS–96–152, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA, 1996.

[OR97] R. Ostrovsky, Y. Rabani. Universal $O(\text{congestion}+\text{dilation}+\log^{1+\epsilon} N)$ local control packet switching algorithms. In *Proc. of the 29th Ann. ACM Symp. on Theory of Computing*, pp. 644-653, 1997.

[RT96] Y. Rabani, E. Tardos. Distributed packet switching in arbitrary networks. In *28th Ann. ACM Symp. on Theory of Computing*, pp. 366-375, 1996.

[SSW94] D. Shmoys, C. Stein, J. Wein. Improved approximation algorithms for shop scheduling problems. *SIAM J. on Comput.* **23**(3), pp. 617-632, 1994.

[WHH+96] D. Williamson, L. Hall, J. Hoogeveen, C. Hurkens, J. Lenstra, S. Sevastjanov, D. Shmoys. “Short shop schedules”. *Operations Research*, 1996.