(Acyclic) Job Shops are Hard to Approximate

Monaldo Mastrolilli and Ola Svensson

IDSIA

Lugano, Switzerland,

{monaldo,ola}@idsia.ch

Abstract

For every $\epsilon > 0$, we show that the (acyclic) job shop problem cannot be approximated within ratio $O(\log^{1-\epsilon} lb)$, unless NP has quasi-polynomial Las-Vegas algorithms, and where lb denotes a trivial lower bound on the optimal value. This almost matches the best known results for acyclic job shops, since an $O(\log^{1+\epsilon} lb)$-approximate solution can be obtained in polynomial time for every $\epsilon > 0$.

Recently, a PTAS was given for the job shop problem, where the number of machines and the number of operations per job are assumed to be constant. Under $P \neq NP$, and when the number μ of operations per job is a constant, we provide an inapproximability result whose value grows with μ to infinity. Moreover, we show that the problem with two machines and the preemptive variant with three machines have no PTAS, unless NP has quasi-polynomial algorithms. These results show that the restrictions on the number of machines and operations per job are necessary to obtain a PTAS.

In summary, the presented results close many gaps in our understanding of the hardness of the job shop problem and resolve (negatively) several open problems in the literature.

1 Introduction

In the job shop scheduling problem there is a set of n jobs that must be processed on a given set M of machines. Each job J_j consists of a sequence of μ operations $O_{1j}, O_{2j}, \ldots, O_{\mu j}$ that need to be processed in this order. Operation O_{ij} must be processed without interruption on machine $m_{ij} \in M$, during $p_{ij} \in \mathbb{Z}^+$ time units. Each machine can process at most one operation at a time, and each job may be processed by at most one machine at any time. A job shop instance is *acyclic* if each job has at most one operation per machine. For any given schedule, let C_j be the completion time of the last operation of job J_j. The goal is to find a feasible schedule which minimizes the *makespan* $C_{\max} = \max_j C_j$. In standard scheduling notation [8], this

problem is denoted as $J||C_{\max}$ (and $J|acyclic|C_{\max}$).

The job shop scheduling problem is a widely studied combinatorial optimization problem (see e.g. [11]). It is strongly NP-hard even for two machines [7]. If D denotes the length of the longest job (the dilation), and C denotes the time units requested by all jobs on the most loaded machine (the congestion), then $lb = max[C, D]$ is a lower bound on the shortest makespan. For unbounded number of machines, Shmoys et al. [17] and Goldberg et al. [6] obtained the best approximation algorithms known with performance guarantee $\tilde{O}((\log lb)^2)$ for general jobs shops[1], where the $\tilde{O}$ notation is used to suppress $\log \log lb$ terms. For acyclic job shops, Feige & Scheideler [4] and Czumaj & Scheideler [3] improved this result to an $\tilde{O}(\log lb)$-approximation algorithm. In the case of acyclic job shops with unit processing times for every operation, the famous paper by Leighton, Maggs, and Rao [12] shows the existence of solutions with makespan $O(lb)$. Leighton, Maggs, and Richa [13] later gave an algorithmic variant yielding a constant factor approximation algorithm.

It is a long standing open problem if the above algorithms for $J||C_{\max}$ and $J|acyclic|C_{\max}$, are tight or even nearly tight (see "Open problem 7" in [16]). The only known inapproximability result is due to Williamson et al. [18], and states that when the number of machines and jobs are part of the input, it is NP-hard to approximate the acyclic job shop scheduling problem with unit time, and at most three operations per job, within a ratio better than $5/4$.

In the preemptive variant of the problem (denoted $J|pmtn|C_{\max}$), every operation can be temporarily interrupted and resumed later without any penalty. For any $\varepsilon > 0$, it is well-known that with only ε loss in the approximation factor, the preemptive job shop scheduling problem is equivalent to the nonpreemptive job shop scheduling problem with unit processing times (see e.g. [2]), and therefore the $5/4$-inapproximability in [18] applies to the preemptive version as well. For the acyclic job shop schedul-

---

[1] We note that we can assume $\log lb = O(\log m\mu)$ [17], where m is the number of machines and μ is the maximum number of operations per job.

ing with preemption, the best known result is due to Feige & Scheideler [4] who showed that there always exists a preemptive schedule within a $O(\log \log lb)$ factor of lb. For the general preemptive job shop problem, Bansal et al. [2] showed an $O(\log |M|/ \log \log |M|)$-randomized approximation algorithm, and a $(2 + \varepsilon)$-approximation for a constant number of machines. It is another open problem [2, 16] to understand whether there is a PTAS for the general nonpreemptive and preemptive job shop with a constant number of machines. For those instances where the number of machines and μ are constant, polynomial time approximation schemes are known [9, 5] for both, the preemptive and nonpreemptive case.

In this paper we give an answer to "Open problem 7" raised in [16]. More precisely, let $\epsilon > 0$ be an arbitrarily small constant. We show that the (acyclic) job shop problem cannot be approximated within ratio $O(\log^{1-\epsilon} lb)$, unless $NP \subseteq ZTIME(n^{\text{polylog } n})$. This almost matches (up to smaller terms) the best known results for $J|acyclic|C_{\max}$ [4, 3], since an $O(\log^{1+\epsilon} lb)$-approximate solution can be obtained in polynomial time for every $\epsilon > 0$. If one is only willing to believe that $P \neq NP$ then, for fixed number of operations per job, we provide an inapproximability result whose value grows with the number of operations per job to infinity.

Finally, we show that the job shop problem with two machines ($J2||C_{\max}$), and the preemptive variant with three machines ($J3|pmtn|C_{\max}$) have no PTAS, unless $NP \subseteq DTIME(n^{O(\log n)})$. These results show that the restrictions in [9], on the number of machines and operations, are necessary to obtain a PTAS, and solve (negatively) an open question raised in [2].

Many questions remain open. For the following two well-known problems our understanding is especially weak. The flow shop scheduling problem is a variant of the acyclic job shop problem where each job has exactly one operation for every machine, and all jobs go through all the machines in the same order. The best known approximation algorithm for this problem is the algorithm provided for the acyclic job shop [4], and the best inapproximability result says that it cannot be approximated better than $5/4$ [18]. [2] A similar situation holds for the general preemptive job shop scheduling problem.

1.1 Preliminaries

When considering a job shop instance we shall use C^*_{max} to denote the minimum makespan over all feasible schedules. For a given graph G, we let $\chi(G)$ and $\alpha(G)$ denote

---

[2]Preliminary results of the authors show that the more general variant of the flow shop problem where each job is not required to go through all the machines has similar inapproximability results as for the acyclic job shop.

the chromatic number of G and the size of a maximum independent set of G, respectively. We shall also denote the maximum degree of graph G by $\Delta(G)$, where we sometimes drop G when it is clear from the context.

Our reductions to the job shop problem with unbounded number of machines use results by Khot [10], who proved that it is NP-hard to color a K-colorable graph with $K^{\frac{1}{25}(\log k)}$ colors, for sufficiently large constants K. In fact the following stronger statement is a direct consequence of the soundness analysis presented in the same paper.

Theorem 1.1 ([10]) *For all sufficiently large constants K, it is* NP-*hard to decide if a graph can be colored using K colors or has no independent set containing a fraction $1/K^{\frac{1}{25}(\log K)}$ of the vertices. Moreover this hardness result holds for graphs with bounded degree, in fact graphs with degree at most $2^{K^{O(\log K)}}$.*

By using a stronger assumption we can let K be a function of the number of vertices. Again the stronger statement (not explicitly stated in [10]) follows from the soundness analysis.

Theorem 1.2 ([10]) *There exists an absolute constant $\gamma > 0$ such that for all $K \leq 2^{(\log n)^\gamma}$, it is hard to decide if an n-vertex graph can be colored using K colors or has no independent set containing a fraction $1/K^{\Omega(\log K)}$ of the vertices, unless $NP \subseteq DTIME(2^{O(\log n)^{O(1)}})$.*

Our reductions to $J2||C_{\max}$ and $J3|pmtn|C_{\max}$ use the following result by Alimonti and Kann [1].

Theorem 1.3 ([1]) *There exist positive constants α, β with $\alpha > \beta$, so that it is NP-hard to decide whether an n-vertex cubic graph has an independent set of size $\alpha \cdot n$ or has no independent set of size $\beta \cdot n$.*

1.2 Results and Proof Ideas

Our first result shows that acyclic job shops have no constant approximation algorithm, unless $P = NP$.

Theorem 1.4 *For all sufficiently large constants K, it is NP-hard to decide if an acyclic job shop instance can be scheduled with makespan $K \cdot lb$ or has no schedule with makespan $(1/8)K^{\frac{1}{25}(\log K)} \cdot lb$. Moreover this hardness result holds for acyclic job shop instances with bounded μ, that only depends on K.*

The main idea of the reduction is as follows. Given a graph G with bounded degree Δ, we construct a job shop instance S, where all jobs have the same length D and all machines the same load $C = D$. Hence, $lb = C = D$. Instance S has a set of jobs for each vertex in G with the property that two jobs *can* be scheduled in parallel, i.e., their operations can overlap in time, if and only if their corresponding

vertices are *not* adjacent. This property is achieved by introducing different "types" of jobs, a technique previously used in [4]. For the reduction to be polynomial it is crucial that the number of types is relatively few, However, to ensure the desired properties, jobs corresponding to adjacent vertices must be of different types. We resolve this by using that G has bounded degree. Since the graph G has degree at most Δ we can in polynomial time partition its vertices into $\Delta + 1$ independent sets. As two jobs only need to be assigned different types if they correspond to adjacent vertices, we only need a constant $(\Delta + 1)$ number of types.

The analysis follows naturally: A set of jobs corresponding to an independent set can be scheduled in parallel. Hence, if the graph G can be colored with K colors then there is a schedule of S with makespan $K \cdot lb$. Finally, if there is a schedule with makespan $K^{\frac{1}{25}(\log K)} \cdot lb$ then at least a fraction $\Omega(1/K^{\frac{1}{25}(\log K)})$ of the jobs overlap. As the jobs overlap, they correspond to a fraction $\Omega(1/K^{\frac{1}{25}(\log K)})$ of vertices that form an independent set.

By using a similar reduction but using Theorem 1.2 and setting $K = \log n$ we give a hardness result that almost matches the $O(\log lb \log \log lb)$-approximation algorithm for acyclic job shops.

Theorem 1.5 *Let $\epsilon > 0$ be an arbitrarily small constant. There is no $(\log lb)^{1-\epsilon}$-approximation algorithm for the acyclic job shop problem, unless $NP \subseteq ZTIME(2^{O(\log n)^{O(1/\epsilon)}})$.*

The tricky part is that the graph has no longer small bounded degree. We overcome this difficulty by a randomized process that preserves the desired properties of the graph with an overwhelming probability (see Lemma 2.1).

Remarks. The analyses of Theorem 1.4 and Theorem 1.5 are straightforward to extend to the job shop problem with objective to minimize the *sum* of completion times. Hence, we have that $J|acyclic|\sum C_j$ has no constant approximation algorithm, unless $P = NP$, and no $(\log lb)^{1-\epsilon}$-approximation algorithm, unless $NP \subseteq ZTIME(2^{O(\log n)^{O(1/\epsilon)}})$. The latter result is almost tight, since an $\tilde{O}(\log lb)$-approximation algorithm for $J|acyclic|\sum w_j C_j$ was presented in [15].

Theorem 1.4 and Theorem 1.5 establish a nice relationship between the job shop problem and the coloring problem. It is tempting to believe that a short schedule also implies that the associated graph has a small chromatic number. This is not the case. However, a short schedule implies that the associated graph has a small *fractional* chromatic number. □

In [9], a PTAS was given for the job shop problem, where the number of machines and the number of operations per job are both assumed to be constant. Our second result

shows that both these restrictions are necessary to obtain a PTAS, and solve (negatively) an open question raised in [2].

Theorem 1.6 *Problems $J2||C_{\max}$ and $J3|pmtn|C_{\max}$ have no PTAS unless $NP \subseteq DTIME(n^{O(\log n)})$.*

The reductions are from the independent set problem in cubic graphs. We give a high level description of the reduction to $J2||C_{\max}$. The reduction to $J3|pmtn|C_{\max}$ is more involved, but the basic structure is the same. Due to space limitations the reduction for $J3|pmtn|C_{\max}$ can be found in the full version of the paper [14].

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

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

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

2 Unbounded Number of Machines

Here, we prove Theorem 1.4 and Theorem 1.5. When using probabilistic arguments for graphs with n vertices, we shall use the term *overwhelming* (*negligible*, respectively) to denote probability that tends to 1 (to 0, respectively) as n tends to infinity.

We present a gap-preserving reduction, Γ, from the graph coloring problem to the acyclic job shop problem, that has two parameters r and d. Given an n-vertex graph G whose vertices are partitioned into at most d independent sets, it computes deterministically if both r and d are constants and probabilistically otherwise, in time polynomial in n and r^d, an acyclic job shop instance such that

- The number of jobs and the number of machines are both $r^{11d}n$;
- The number of operations per job is at most Δr^{4d};
- Each job has length r^d and each machine has load r^d. Hence, $lb = r^d$;
- Completeness case: If G can be colored using L colors then $C^*_{max} \leq lb \cdot L$;
- Soundness case: Given a schedule with makespan $lb \cdot L$, we can, in time polynomial in n and r^d, find an independent set of G of size $(1/4 - \frac{5\Delta L}{r})n/L$.

When the parameters r and d are functions of n, i.e., the reduction is probabilistic, the above properties hold with probability 1 with the exception that the soundness analysis might fail with negligible probability. Since the soundness case is constructive we can, given a schedule, detect such a failure in polynomial time.

In Section 2.1, we present a deterministic reduction with somewhat stronger properties for the *general* job shop problem. As the reduction is relatively simple, it serves as a good starting point before reading the similar but more complex reduction Γ for the *acyclic* job shop problem.

Before continuing, let us see how Γ is sufficient for proving Theorem 1.4 and Theorem 1.5.

Proof of Theorem 1.4. By Theorem 1.1, for sufficiently large K and $\Delta = 2^{K^{O(\log K)}}$, it is NP-hard to decide if an n-vertex graph G with bounded degree Δ has

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

As the vertices of a graph with bounded degree Δ can, in polynomial time, be partitioned into $\Delta + 1$ independent sets, we complete the proof by using Γ with parameters $d = \Delta + 1$ and $r = 40\Delta K^{\frac{1}{25}(\log K)}$. We note that the construction is deterministic as both d and r are constants. □

Proof of Theorem 1.5. The proof is similar to the proof of Theorem 1.4 with the exception that the graphs have no longer bounded degree. To this end the following lemma will be useful, whose proof can be found in Appendix A.

Lemma 2.1 *For any constant $\delta \geq 1$, we can, given an n-vertex graph $G = (V, E)$, construct in randomized polynomial time, a subgraph $G' = (V, E')$ of G with $E' \subseteq E$ such that*

1. *The vertices are partitioned into $(\log n)^\delta$ sets, each set forms an independent set in G'.*
2. *We have that $\chi(G') \leq \chi(G)$.*
3. *With overwhelming probability: Given an independent set of G', with $\frac{n}{(\log n)^{\delta-1}}$ vertices, we can, in polynomial time, find an independent set of G with $\frac{n}{(\log n)^\delta}$ vertices.*

Assuming $NP \not\subseteq DTIME(2^{O(\log n)^{O(1)}})$, Theorem 1.2 with $K = \log n$ says that it is hard to decide if an n-vertex graph G has

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

Given an n-vertex graph G, we construct graph G' from G by applying Lemma 2.1. We then obtain a job shop instance S from G' by using Γ with parameters $d = (\log n)^\delta$ and $r = n^3$, where $\delta = 3/\epsilon$. The size of S is $O(r^{11d}n \cdot \Delta r^{4d}) = O(2^{O(\log n)^{O(1/\epsilon)}})$ and $lb = n^{3(\log n)^\delta}$.

The analysis is straightforward: If $\chi(G) \leq \log n$ then $\chi(G') \leq \log n$ and, by the completeness case of Γ, there is a schedule of S with makespan $\log n \cdot lb$. On the other hand, assuming that the probabilistic constructions of G' and S succeeded, we have that if $\alpha(G) \leq \frac{n}{(\log n)^{\Omega(\log \log n)}}$ then $C^*_{max} > lb \cdot 1/8(\log n)^{\delta-1}$. Since otherwise if $C^*_{max} \leq lb \cdot 1/8(\log n)^{\delta-1}$ then G' has an independent set of size $n/(\log n)^{\delta-1}$ (soundness case of Γ) and thus G has an independent set of size $n/(\log n)^\delta$ (Lemma 2.1).

The probabilistic constructions of G' and S succeed with overwhelming probability. Furthermore, as the properties of G' and S that might fail are constructive, we can, given a schedule, detect such a failure in polynomial time and repeat the reduction. It follows that a $(1/8 \cdot (\log n)^{\delta-2})$-approximation algorithm for the acyclic job shop problem would imply that $NP \subseteq ZTIME(2^{O(\log n)^{O(1/\epsilon)}})$. Finally we note that δ was chosen such that

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

$$3 \cdot (\log n)^{\delta-2-\epsilon} < 1/8 \cdot (\log n)^{\delta-2}.$$

□

2.1 General Job Shops

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

Construction. Given an n-vertex graph $G = (V, E)$ whose vertices are partitioned into d independent sets, we create a job shop instance $S(r, d)$, where r and d are the parameters of the reduction. There is a machine M_v and a job J_v for each vertex $v \in V$. We continue by describing the

operations of the jobs. Let $I_1, I_2, \ldots I_d$ denote the independent sets that form a partition of V. To simplify notation, we shall use $I_{<i}$ to denote $\bigcup_{k:1 \leq k < i} I_k$. A job J_v that corresponds to a vertex $v \in I_i$, for some $i : 1 \leq i \leq d$, has a chain of r^{i-1} *long-operations* $O_1, O_2, \ldots, O_{r^{i-1}}$, each of them requires r^{d-i+1} time units, that must be processed on the machine M_v. Between two consecutive long-operations O_p, O_{p+1} of J_v, for $p : 1 \leq p < r^{i-1}$, we have a set of *short-operations* placed on the machines $\{M_u : \{u, v\} \in E, u \in I_{<i}\}$ in some order. A short-operation requires time 0.

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

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

Lemma 2.2 *If $\chi(G) = L$ then then there is a schedule of $S(r, d)$ with makespan $lb \cdot L$.*

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

Soundness. We prove that, given a "short" schedule, we can, in polynomial time, find a "big" independent set of G.

Lemma 2.3 *Given a schedule of $S(r, d)$ with makespan $lb \cdot L$, we can, in time polynomial in n and r^d, find an independent set of G of size at least $(1 - \frac{\Delta}{r})n/L$.*

Proof. First we show that two jobs corresponding to adjacent vertices cannot be scheduled in parallel.

Claim 2.4 *Let $u \in I_i$ and $v \in I_j$ be two adjacent vertices in G with $i < j$. Then at most a fraction $\frac{1}{r}$ of the long-operations of J_v can overlap the long-operations of J_u.*

Proof of Claim. There are r^{i-1} and r^{j-1} long-operations of J_u and J_v, respectively. As the vertices u and v are adjacent, job J_v has a small-operation on Machine M_u between any two long-operations. Hence at most one long-operation of J_v can be scheduled in parallel with a long-operation of J_u and the total number of such operations in any schedule is at most $r^{i-1} \leq \frac{r^{j-1}}{r}$. □

Now consider a schedule with makespan $lb \cdot L$. For each $i : 1 < i \leq d$ and for each $v \in I_i$, we disregard those long-operations of job J_v that overlap long-operations of the jobs $\{J_u : \{u, v\} \in E \text{ and } u \in I_{<i}\}$. After disregarding operations, no two long-operations corresponding to adjacent vertices will overlap in time. Furthermore, by applying Claim 2.4 and using that the maximum degree of G is Δ, we know that at most a fraction $\frac{\Delta}{r}$ of a job's long-operations have been disregarded. The statement now follows by observing that the remaining long-operations of each job require time at least $(1 - \frac{\Delta}{r}) \cdot lb$ and by a simple averaging argument we have that at least $(1 - \frac{\Delta}{r})n/L$ long-operations must overlap at some point in the schedule. Moreover, we can find such a point in the schedule by, for example, considering the start and end points of all remaining long-operations. □

2.2 Acyclic Job Shops

Here, we present the reduction Γ for the acyclic job shop problem. The idea is similar to the reduction presented in Section 2.1 for the general job shop problem. The main difference is to ensure, without using cyclic jobs, that jobs corresponding to adjacent vertices cannot be scheduled in parallel. To this end we need some preliminary definitions and a lemma that are slight variations of the techniques developed in [4] for proving the existence of acyclic job shop instances with $C^*_{max} = \Omega(lb \cdot \log lb / \log \log lb)$. We shall use $[m]$ for the set $\{0, 1, \ldots, m-1\}$.

Definition 2.5 (Bucket and List)

1. *A bucket $B(m, b)$ is a subset of $[m]$ with $|B(m, b)| = (1 - 1/b)m$.*
2. *A list of type $L(m, \ell, s)$ is a family of ℓ mutually disjoint subsets of $[m]$, where each subset has cardinality s.*
3. *A list avoids a bucket $B(m, b)$ if each of its ℓ subsets has a non-empty intersection with $[m] \setminus B(m, b)$.*

Definition 2.6 (Conflict Family) *A set of m lists of type $L(m, \ell, s)$ is called a conflict family $C(m, \ell, s)$ if for any bucket $B(m, \ell)$ at least $(1 - 1/\ell^2)m$ lists avoid it.*

The proof of the following lemma is similar to the proof of Lemma 3.6 in [4] and can be found in Appendix B.

Lemma 2.7 *For ℓ sufficiently large, for $m = \ell^{11}$ and for $s = \ell^3$, we can in $O(m\ell s)$ time with overwhelming probability construct a conflict family $C(m, \ell, s)$. Moreover, this can be done deterministically when ℓ is a constant.*

Construction. Given an n-vertex graph $G = (V, E)$ whose vertices are partitioned into d independent sets, we create an acyclic job shop instance $S(r, d)$, where r and

d are the parameters of the reduction. Let $I_1, I_2, \ldots I_d$ denote the independent sets that form a partition of V. To simplify notation, we shall again use $I_{<i}$ to denote $\bigcup_{k:1\leq k<i} I_k$. Construct a conflict family $C(m, \ell, s)$, where $\ell = r^d$, $s = \ell^3$, and $m = \ell^{11}$. By Lemma 2.7, the construction always succeeds if ℓ is a constant, and otherwise, if ℓ is non-constant, it succeeds with overwhelming probability. The m different lists in the conflict family will be used when we describe the jobs and will be referred to as $\ell_1, \ell_2, \ldots, \ell_m$. The instance $S(r, d)$ has a set of m machines $M_v = \{M_v^0, M_v^1, \ldots, M_v^{m-1}\}$ and a set of m jobs $J_v = \{J_v^1, J_v^2, \ldots, J_v^m\}$ for each vertex $v \in V$. We continue by describing the operations of the jobs. A set of jobs $J_v = \{J_v^1, J_v^2, \ldots, J_v^m\}$ that correspond to a vertex $v \in I_i$, for some $i : 1 \leq i \leq d$, has the following operations. The job J_v^j, for $j : 1 \leq j \leq m$, has a chain of r^{i-1} *long-operations* $O_1, O_2, \ldots, O_{r^{i-1}}$ that must be processed on the respectively machines $M_v^j, M_v^{j+1}, \ldots, M_v^{j+r^{i-1}}$, where the superscript is $mod\ m$. A long-operation of J_v^j requires time r^{d-i+1}. Between two consecutive long-operations O_p, O_{p+1} of J_v^j, for $p : 1 \leq p < r^{i-1}$, we have a set of short-operations that require 0 time units. These operations are defined as follows. Let $s_1, s_2, \ldots, s_\ell$ denote the ℓ subsets of size s in the list ℓ_j. For each $u \in I_{<i}$ with $\{u, v\} \in E$ we have s short-operations placed on the machines $\{M_u^k\}_{k \in s_p}$ in some order.

Remarks.

- The instance is acyclic as its operations are scheduled on different machines. This follows by first observing that a job has at most r^{d-1} long-operations and $m > r^{d-1}$. And secondly, by the definition of a list, two short-operations of a job is not scheduled on the same machine.

- For each $v \in V$, the jobs in J_v are the only jobs that have long-operations on the machines in M_v. Moreover, the long-operations of the jobs in J_v are placed in such a way so that all jobs in J_v can be scheduled in parallel.

- The number of jobs and the number of machines are both $m \cdot n = r^{11d} n$. Each job has length r^d and each machine has load r^d. Hence, $lb = r^d$. Moreover, the number of operations per job is at most $s\Delta r^{d-1} + r^{d-1} \leq \Delta r^{4d}$.

$\square$

Completeness. By similar arguments as in the proof of Lemma 2.2, if graph G can be colored with L colors then there is a relatively short solution to the job shop instance.

Lemma 2.8 *If $\chi(G) = L$ then there is a schedule of $S(r, d)$ with makespan $lb \cdot L$.*

Soundness. We carry out the analysis in this section by assuming that the construction of the conflict family $C(m, \ell, s)$ succeeded. We prove that, given a short schedule, we can, in polynomial time, find a big independent set of G.

Lemma 2.9 *Given a schedule of $S(r, d)$ with makespan $lb \cdot L$, we can, in time polynomial in n and r^d, find an independent set of G of size at least $(1/4 - \frac{5\Delta L}{r})n/L$.*

Proof. For the statement to be interesting we can assume that $r > 20\Delta L$. The main idea of the proof is as follows. Each vertex $v \in V$ will be associated to $lb/4$ time steps of the given schedule in which the jobs in J_v have many long-operations. We then use the properties of the conflict family to prove that the time steps associated to adjacent vertices will be almost disjoint and the analysis then follows in the same way as for general job shops.

Throughout the proof we consider any fixed schedule with makespan $lb \cdot L$. We start by defining the time steps associated to each vertex. A vertex $v \in V$, with $v \in I_i$ for some $i : 1 \leq i \leq d$, is associated to a set T_v that consists of $r^{i-1}/2$ disjoint time intervals of length $r^{d-i+1}/2$ so that each time interval in T_v is completely covered by at least $m/(4L)$ long-operations of the jobs in J_v. The set T_v can be selected as follows. Partition the schedule into $2Lr^{i-1}$ blocks of consecutive time steps, where each block is of length $r^{d-i+1}/2$. As long-operations of jobs in J_v require r^{d-i+1} time units, each long-operation completely covers at least one block. Since there are $m \cdot r^{i-1}$ long-operations of the jobs in J_v and each block can be completely covered by at most m long-operations, at least $r^{i-1}/2$ of the blocks are completely covered by at least $m/4L$ long-operations of the jobs in J_v. Let T_v contain $r^{i-1}/2$ of those blocks.

We continue by showing that the time steps associated to two adjacent vertices are almost disjoint. We call a job $j \in J_v$ *bad*, with $v \in I_i$ for some $i : 1 \leq i \leq d$, if there exists an adjacent vertex u, with $u \in I_{<i}$, so that at least two long-operations of j overlap some interval $t \in T_u$.

Claim 2.10 *For each vertex $v \in V$ at most a fraction $1/\ell$ of the jobs in J_v are bad.*

Proof of Claim. Consider a vertex $v \in V$, with $v \in I_i$ for some $i : 1 \leq i \leq d$. For a job $J_v^j \in J_v$ to be bad, there must exist a vertex $u \in I_{<i}$, adjacent to v, so that at least two of J_v^j's long-operations overlap some time interval $t \in T_u$. This means that list ℓ_j of the conflict family $C(m, \ell, s)$ fails to avoid the bucket B, defined by the superscripts of the machines in M_u that have idle-time during the time interval t. By the definition of T_u, t is completely covered by at least $m/(4L)$ long-operations of jobs in J_u. It follows that the bucket B has size at most $m - m/(4L)$ and, as $4L \leq \ell$, at most m/ℓ^2 lists fail to avoid it (see Lemma 2.7). In other

words, at most m/ℓ^2 jobs in J_v have two long-operations that overlap with the time interval t.

Finally, as vertex v has at most Δ adjacent vertices and each vertex is associated with at most $r^{d-1}/2$ time intervals, we have that at most $\Delta r^{d-1}m/(2\ell^2) < \Delta m/(r \cdot \ell) < m/\ell$ of the jobs in J_v are bad. $\quad\square$

Remove all the bad jobs. Since we for each $v \in V$ remove at most m/ℓ jobs in J_v, the intervals in T_v are still completely covered by at least $m(\frac{1}{4L} - \frac{1}{\ell}) > m(\frac{1}{4L} - \frac{1}{(20\Delta L)^d}) \geq \frac{m}{5L}$ long-operations. The remaining arguments are similar to the soundness analysis of the general job shop problem. The main difference is that, for each vertex, we now consider a set of *time intervals*, instead of only considering the long-operations of a single job as done in Section 2.1. A key ingredient for the remaining part of the proof is the following observation. After removing the bad jobs no job in J_v with $v \in I_i$, for some $i : 1 \leq i \leq d$, will have two long-operations that overlap some time interval $t \in T_u$, where $u \in I_{<i}$ and $\{u,v\} \in E$. Furthermore, as each interval of T_v are completely covered by at least $m/(5L)$ operations, at most $5L$ time intervals in T_v can overlap a time interval in T_u (otherwise some of the remaining jobs would be bad). More specifically, we have

Claim 2.11 *Let $u \in I_i$ and $v \in I_j$ be two adjacent vertices in G with $i < j$. Then at most a fraction $\frac{5L}{r}$ of the intervals in T_v are overlapping with the intervals in T_u.*

Proof of Claim. The sets T_u and T_v contain $r^{i-1}/2$ and $r^{j-1}/2$ intervals, respectively. Since we removed the bad jobs, at most one long-operation of some job in J_v can be scheduled in parallel with an interval in T_u. As each interval of T_v are completely covered by at least $m/(5L)$ operations, at most $5L$ time intervals in T_v can overlap a time interval in T_u (otherwise some of the remaining jobs would be bad). We conclude that the total number of intervals in T_v that overlaps with intervals in T_u is at most $5L \cdot r^{i-1}/2$. The statement now follows by recalling that T_v contains $r^{j-1}/2$ intervals and $j > i$. $\quad\square$

For each $i : 1 < i \leq d$ and for each $v \in I_i$, we disregard those intervals of T_v that overlap intervals in the set $\bigcup_{\{u,v\}\in E, u \in I_{<i}} T_u$. After disregarding those intervals, no two intervals corresponding to adjacent vertices will overlap in time. Furthermore, by applying Claim 2.11 and using that the maximum degree of G is Δ, we know that at most a fraction $\frac{\Delta 5L}{r}$ of a job's time intervals have been disregarded. The statement now follows by observing that the remaining time intervals of each job require time at least $(1-\frac{\Delta 5L}{r}) \cdot lb/4$ and by a simple averaging argument we have that at least $(1/4 - \frac{\Delta 5L}{4r})n/L$ time intervals must overlap at some point in the schedule. Moreover, we can find such a point in the schedule by, for example, considering the start and end points of all remaining time intervals. $\quad\square$

3 Fixed number of Machines

In this section we show that $J2||C_{\max}$ has no PTAS by presenting gap preserving reductions from the independent set problem in cubic graphs. Given a cubic graph $G = (V, E)$, with $n = |V|$ and thus $|E| = 3n/2$, it is well known (see Theorem 1.3) that it is NP-hard to distinguish whether G has an independent set (IS) of size $\alpha \cdot n$ or no independent set of size $\beta \cdot n$, for some $\alpha > \beta$. For any given G, we construct an instance of $J2||C_{\max}$, so that, for some $f(n)$ and a constant $\epsilon > 0$, we have

$$IS \geq \alpha \cdot n \quad \Rightarrow \quad C_{\max} \leq f(n) \qquad (1)$$
$$IS \leq \beta \cdot n \quad \Rightarrow \quad C_{\max} \geq (1+\epsilon)f(n) \qquad (2)$$

The following lemma will be useful in the construction.

Lemma 3.1 *For any small fixed $\epsilon > 0$, we can in time polynomial in n, construct a family $\mathcal{C} = \{C_1, C_2, \ldots, C_{n^2}\}$ with the following properties:*
  1. *Each set $C_i \in \mathcal{C}$ is a subset of $\{1, 2, \ldots, (1/\epsilon)^{1/\epsilon} \log n\}$ and has size $\log n$.*
  2. *Two sets $C_i \in \mathcal{C}$ and $C_j \in \mathcal{C}$, with $i \neq j$, satisfy $|C_i \cap C_j| \leq \epsilon \log n$.*

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

Construction. Before defining the jobs we will define "blocks" of operations. The jobs will later be defined as a concatenation of these blocks.

Let $d = O(\log n)$, for each $i : 1 \leq i \leq d$ we define type T_i and type $\bar{T}_i$ as

$$T_i := \left([M_1, n^{4(d-i+1)}], [M_2, 0]\right)^{n^{4(i-1)}}$$
$$\bar{T}_i := \left([M_2, n^{4(d-i+1)}], [M_1, 0]\right)^{n^{4(i-1)}}.$$

We will call the operations of T_i and $\bar{T}_i$, that require $n^{4(d-i+1)}$ time units, for *long-operations* and the operations, that require 0 time units, for *short-operations*. Note that a type requires time $n^{4(i-1)}n^{4(d-i+1)} = n^{4d}$ and two types T_i and $\bar{T}_j$ are *compatible*, i.e., they can be scheduled in parallel, if and only if $i = j$. For $i = 1, \ldots, |E|$, a *configuration* $C_i = (T_{\pi_{i,1}}, \ldots, T_{\pi_{i,\log n}})$ is an ordered sequence of $\log n$ types, where $\pi_{i,j} \in \{1, \ldots, d\}$ denotes the "frequency" of the j-th type of configuration C_i. Lemma 3.1 shows that we can define a set of configurations $\mathcal{C} = \{C_i : i = 1, \ldots, |E|\}$ such that any two configurations $C_i \in \mathcal{C}$ and $C_j \in \mathcal{C}$ with $i \neq j$ have at most

$\varepsilon \log n$ types in common, for $\varepsilon > 0$ arbitrarily small. The set $\bar{\mathcal{C}} = \{\bar{C}_i : i = 1, \ldots, |E|\}$ is defined in a similar way by using the types $\bar{T}_i$, i.e., for $i = 1, \ldots, |E|$ we have $\bar{C}_i = (\bar{T}_{\pi_{i,1}}, \ldots, \bar{T}_{\pi_{i,\log n}})$. Note that a configuration requires $n^{4d} \log n$ time units. We are now ready to define the different blocks. For $i = 1, \ldots, |E|$, block B_i is obtained by concatenating C_i for n^2-times, i.e. $B_i := (C_i)^{n^2}$; similarly $\bar{B}_i := (\bar{C}_i)^{n^2}$. Let $D = n^{4d+2} \log n$ be the length of a block.

The blocks are now used as building blocks for defining the jobs. We have a *big job* J_b that is composed of an *edge-part* $B_E = (B_1, B_2, \ldots, B_{|E|})$, followed by a *tail-part* $O_{IS} = [M_2, D \cdot \alpha n]$, a *slack-part* $B_S = ([M_1, 1])^{D \cdot 3(1-\alpha)n}$, and finally a *remaining-part* $O_{VC} = [M_2, D \cdot (1-\alpha)n]$.

We have a *vertex job* J_v for each vertex $v \in V$. Let e_i, e_j, e_k be the 3 edges incident to v with $i < j < k$. Job J_v is composed of the sequence $(\bar{B}_i, \bar{B}_j, \bar{B}_k)$ (*edge part*) followed by a *tail operation* $O_v = [M_1, D]$. Note that the length of job J_b is $D(|E| + n + 3(1-\alpha)n)$ and the length of a vertex job is $4D$.

The following fundamental lemma motivates our construction. It shows that for any pair $\{u, v\} \in E$ of adjacent vertices, either J_u or J_v cannot be completed before the end of O_{IS} without delaying job J_b of $\gamma \cdot D$ time units. It follows that, without delaying job J_b, only jobs corresponding to vertices that form an independent set can be completed before the end of O_{IS}.

Lemma 3.2 *For any $i \in \{1, \ldots, |E|\}$, if there are two copies of block $\bar{B}_i$ to be scheduled, then at least $\gamma \cdot D$ time units of these two blocks cannot be scheduled in parallel with the edge-part of job J_b, for some $\gamma < 1$ that can be made arbitrarily close to 1.*

Proof. Let A_1 and A_2 be the two copies of block $\bar{B}_i$. Recall that $\bar{B}_i$ is composed of n^2 repetitions of the configuration $\bar{C}_i$. Hence, A_1 and A_2 contain $2n^2$ copies of configuration $\bar{C}_i$. We say that a configuration $\bar{C}_i$ is contained in some block B_k of job J_b if the first operation of $\bar{C}_i$ starts not before the first operation of B_k starts and the last operation of $\bar{C}_i$ ends not later than the last operation of B_k ends. For $k = 1, \ldots, |E|$, let λ_k denote the number of configurations of A_1 and A_2 that are contained in block B_k of job J_b. Note that, for $k = 1, \ldots, |E|$, at most one configuration of A_i might start before and end after the first operation of B_k, where $i \in \{1, 2\}$. Similarly, at most one configuration of A_i might start before and end after the last operation of B_k, where $i \in \{1, 2\}$. It follows that at most four configurations of A_1 and A_2, that are not contained in some block B_k, can overlap that block's operations. Hence, by considering the configurations not contained in some block and by recalling that a configuration requires $n^{4d} \log n$ time units, we have that at least $\max[0, (2n^2 - \sum_j \lambda_j - 4|E|) \cdot n^{4d} \log n]$ time

units of A_1 and A_2 are not scheduled in parallel with the edge-part of job J_b. Furthermore, as at most D time units of A_1 and A_2 can be scheduled in parallel with block B_i we have that the configurations completely contained in block B_i contribute with at least $\max[0, \lambda_i \cdot n^{4d} \log n - D]$ time units of A_1 and A_2, that are not scheduled in parallel with the edge-part of job J_b. For the other configurations, i.e., the ones that are completely contained in some block B_j with $j \neq i$, we have

Claim 3.3 *The operations of a configuration $\bar{C}_i$ and a block B_j, with $i \neq j$, can overlap at most $\epsilon n^{4d} \log n + o(n^{4d} \log n)$ time units, for any arbitrarily small $\epsilon > 0$.*

Proof of Claim. The block B_j is composed of n^2 repetitions of C_j. Consider types $\bar{T}_k \in \bar{C}_i$ and $T_\ell \in C_j$ with $k \neq l$. If $k > \ell$ then, as $\bar{T}_k$ has a short-operation on machine M_1 between any two consequent long-operations on machine M_2, at most one long-operation of $\bar{T}_k$ overlaps a long-operation of T_ℓ. Since T_ℓ has $n^{4(\ell-1)}$ long-operations and each long-operation of $\bar{T}_k$ requires $n^{4(d-k+1)}$ time units, it follows, by using $k > l$, that the operations of T_ℓ and $\bar{T}_k$ overlap at most $n^{4(d-1)}$ time units. The same result can be obtained when $k < \ell$ by using symmetric arguments. Furthermore, if $T_k \notin C_j$ then the operations of $\bar{T}_k$ and B_j overlap at most $n^2 \log n \cdot n^{4(d-1)} < n^{4d-1}$ time units. The claim now follows since the two configurations $\bar{C}_i$ and C_j have at most $\epsilon \log n$ compatible types. $\qquad \square$

Adding up the two above observations with the above claim give us that at least

$$\max[0, (2n^2 - \sum_j \lambda_j - 4|E|) \cdot n^{4d} \log n] +$$

$$\max[0, \lambda_i \cdot n^{4d} \log n - D] + (1 - \epsilon - o(1)) n^{4d} \log n \sum_{j \neq i} \lambda_j$$

time units of these blocks cannot be scheduled in parallel with the edge-part of job J_b. The statement now follows from that the graph has relatively few edges, $|E| = 3/2n$, and since ϵ can be made arbitrarily small we can pick γ arbitrarily close to 1 and disregard the terms of order $o(n^{4d+2} \log n) = o(D)$. $\qquad \square$

Completeness. We will see that all vertex jobs can be scheduled in parallel with the long job J_b. Thus the makespan of the schedule will be equal to the length of J_b.

Let $V' \subseteq V$ denote an independent set of G with $|V'| = \alpha n$. Since V' forms an independent set no two vertices are incident to the same edge. Recall that a block $\bar{B}_i$ can be scheduled in parallel with a block B_i, the last operation of a vertex job requires time D on machine M_1 and operation O_{IS} of J_b requires time $D\alpha n$ on machine

M_2. It follows that the vertex jobs corresponding to the vertices in V' can all be scheduled in parallel with the blocks $B_1, B_2, \ldots, B_{|E|}$ and operation O_{IS} of J_b. As job J_b has $D \cdot 3(1-\alpha)n$ slack-operations, a block $\bar{B}_i$ can be scheduled in parallel with D slack-operations, and operation O_{VC} requires time $D(1-\alpha)n$, the jobs corresponding to the vertices of $V \setminus V'$, $(1-\alpha)n$ many, can be scheduled in parallel with the slack-operations and operation O_{VC} of J_b.

Soundness. As the makespan equals the length of job J_b in the completeness case, we will analyze the soundness case by showing that there is a fraction of the operations belonging to the vertex jobs that are not scheduled in parallel with J_b.

For any given schedule, let t_1 be the time at which operation O_{IS} is completed, and t_2 be the time at which operation O_{VC} starts. Let $T := n \cdot D$ denote the sum of the tail operation lengths. Let τ_1, τ_2 and τ_3 be the fraction of T spent to schedule tail operations during time interval $[0, t_1)$, $[t_1, t_2)$ and $[t_2, \infty)$, respectively.

It is easy to observe that any positive value of τ_2 creates a delay of job J_b of value $\tau_2 \cdot T$, whereas τ_3 a delay of $\max\{0, (\tau_3 - (1-\alpha))T\}$. Finally, note that there are at least $\tau_1 \cdot n$ jobs that complete their edge-part before time t_1. Since $IS \leq \beta n$, it follows that there are at least $\max\{0, (\tau_1 - \beta)n\}$ conflicting pairs of jobs (i.e., corresponding to adjacent pairs of vertices) that delay job J_b by at least $(\tau_1 - \beta)n \cdot \gamma \cdot D$ time units by Lemma 3.2. It is not difficult to check that the delay of job J_b is at least $(\alpha - \beta)\gamma \cdot n \cdot D$.

Acknowledgments.

References

[1] P. Alimonti and V. Kann. Some APX-completeness results for cubic graphs. *Theor. Comput. Sci.*, 237(1-2):123–134, 2000.

[2] N. Bansal, T. Kimbrel, and M. Sviridenko. Job shop scheduling with unit processing times. *Mathematics of Operations Research*, 31:381–389, 2006.

[3] A. Czumaj and C. Scheideler. A new algorithm approach to the general lovász local lemma with applications to scheduling and satisfiability problems (extended abstract). In *STOC*, pages 38–47, 2000.

[4] U. Feige and C. Scheideler. Improved bounds for acyclic job shop scheduling. *Combinatorica*, 22(3):361–399, 2002.

[5] A. V. Fishkin, K. Jansen, and M. Mastrolilli. Grouping techniques for scheduling problems: Simpler and faster. *Algorithmica*, 51(2):183–199, 2008.

[6] L. Goldberg, M. Paterson, A. Srinivasan, and E. Sweedyk. Better approximation guarantees for job-shop scheduling. *SIAM Journal on Discrete Mathematics*, 14(1):67–92, 2001.

[7] T. Gonzalez and S. Sahni. Flowshop and jobshop schedules: complexity and approximation. *Operations Research*, 26:36–52, 1978.

[8] R. Graham, E. Lawler, J. Lenstra, and A. R. Kan. Optimization and approximation in deterministic sequencing and scheduling: A survey. In *Annals of Discrete Mathematics*, volume 5, pages 287–326. North–Holland, 1979.

[9] K. Jansen, R. Solis-Oba, and M. Sviridenko. Makespan minimization in job shops: A linear time approximation scheme. *SIAM J. Discrete Math.*, 16(2):288–300, 2003.

[10] S. Khot. Improved inaproximability results for maxclique, chromatic number and approximate graph coloring. In *FOCS*, pages 600–609, 2001.

[11] E. Lawler, J. Lenstra, A. R. Kan, and D. Shmoys. Sequencing and scheduling: Algorithms and complexity. *Handbook in Operations Research and Management Science*, 4:445–522, 1993.

[12] F. T. Leighton, B. M. Maggs, and S. B. Rao. Packet routing and job-shop scheduling in O(congestion + dilation) steps. *Combinatorica*, 14(2):167–186, 1994.

[13] F. T. Leighton, B. M. Maggs, and A. W. Richa. Fast algorithms for finding O(congestion + dilation) packet routing schedules. *Combinatorica*, 19:375–401, 1999.

[14] M. Mastrolilli and O. Svensson. (acyclic) job shops are hard to approximate. http://www.idsia.ch/~monaldo, 2008.

[15] M. Queyranne and M. Sviridenko. Approximation algorithms for shop scheduling problems with minsum objective. *Journal of Scheduling*, 5(4):287–305, 2002.

[16] P. Schuurman and G. J. Woeginger. Polynomial time approximation algorithms for machine scheduling: ten open problems. *Journal of Scheduling*, 2(5):203–213, 1999.

[17] D. Shmoys, C. Stein, and J. Wein. Improved approximation algorithms for shop scheduling problems. *SIAM Journal on Computing*, 23:617–632, 1994.

[18] D. Williamson, L. Hall, J. Hoogeveen, C. Hurkens, J. Lenstra, S. Sevastianov, and D. Shmoys. Short shop schedules. *Operations Research*, 45:288–294, 1997.

A Proof of Lemma 2.1

Given an n-vertex graph $G = (V, E)$, we give a probabilistic construction of G'. Each vertex $v \in V$ is assigned, independently, uniformly at random to one of the

sets $I_1, I_2, \ldots, I_{(\log n)^\delta}$. Let $E' \subseteq E$ be those edges that are incident to vertices placed in different sets, i.e., an edge is deleted if and only if it is adjacent to two vertices $u \in I_i$ and $v \in I_i$ for some $i : 1 \leq i \leq (\log n)^\delta$.

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

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

$$\left(\frac{1}{\log n}\right)^{\delta(|H_1|-1)} \left(\frac{1}{\log n}\right)^{\delta(|H_2|-1)} \cdots \left(\frac{1}{\log n}\right)^{\delta(|H_s|-1)}$$

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

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

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

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

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

which tends to 0 as n tends to infinity.

B Proof of Lemma 2.7

We will give a probabilistic construction when ℓ is depending on n, i.e., is not a constant. The other case when ℓ is a constant then also m and s are constants and since the arguments below show that there is at least one such conflict family $C(m, \ell, s)$ exists (for large enough ℓ), we can find it by, for example, using brute force.

Consider the following three step randomized procedure for constructing $C(m, \ell, s)$.

1. For each of the m lists independently, each of the ℓ subsets are chosen independently, and each of the s elements of a subset is chosen uniformly at random, independently of all other choices. After this step the subsets can contain duplicates, i.e., be multi-sets, and the different subsets of a list may not be mutually disjoint.

2. If more than a fraction $1/(2\ell^2)$ of the lists created in step 1 contains multi-sets or two subsets with non-empty intersection, abort.

3. Leave the $(1 - 1/(2\ell^2))m$ valid lists untouched. Rearrange the subsets of the remaining lists so that each of them becomes valid.

First, let us show that the probability that the construction aborts is negligible. Consider any particular list, the probability that it has two elements that take the same value is at most $\binom{\ell s}{2}/m$, which is less than $1/\ell^3$ as $m = \ell^{11}$ and $s = \ell^3$. Hence, the expected number of acyclic jobs is at least $(1 - 1/\ell^3)m$ and the standard bounds on large deviations show that the probability that step 2 aborts is negligible.

We conclude that with overwhelming probability, the construction does not abort. It remains to verify that for any bucket $B(m, \ell)$ at least $(1 - 1/\ell^2)m$ lists avoid it. We shall check a stronger condition after step 1 of the construction, namely, that for any bucket $B(m, \ell)$, at least a fraction of $(1 - 1/(2\ell^2))$ of the lists avoid it. This implies that the construction is valid after step 3, as only a fraction $1/(2\ell^2)$ of the jobs are rearranged.

Fix a particular bucket $B(m, \ell)$, and consider a list L composed of ℓ subsets of $[m]$, each of these subsets contains s elements chosen independently at random. If L fails to avoid $B(m, \ell)$, then at least one out of its ℓ subsets is completely contained in $B(m, \ell)$. The probability that s randomly picked elements belongs to $B(m, \ell)$ is $(1 - \frac{1}{\ell})^s$. Since there are ℓ different subsets we conclude that the probability that a random list fails to avoid a particular bucket is at most $\ell(1 - \frac{1}{\ell})^s$. Substituting back in the value of s, this can be upper bounded by $\ell \cdot e^{-\ell^2} < e^{-\alpha\ell^2}$, for some $\alpha < 1$ that can be made arbitrarily close to 1 when ℓ is sufficiently large.

If there are m random lists as above, the probability that a fraction $1/(2\ell^2)$ of them fail to avoid the bucket configuration is at most $\left(e^{-\alpha\ell^2}\right)^{m/(2\ell^2)} \cdot \binom{m}{m/(2\ell^2)} \leq 2^{-\beta m}$, for some $\beta > 0$. As there are less than $m^{m/\ell} = 2^{m\log m/\ell}$ possible buckets, the probability that a fraction $1/(2\ell^2)$ of lists fail to avoid some bucket configuration is negligible.

It follows that a conflict family can be created in time $O(m\ell s)$ with overwhelming probability.