On the Hardness of Scheduling With Non-Uniform Communication Delays

Sami Davies* Janardhan Kulkarni† Thomas Rothvoss*
Sai Sandeep‡ Jakub Tarnawski† Yihao Zhang*

Abstract

In the problem of scheduling with non-uniform communication delays, the input is a set of jobs with precedence constraints. Associated with every precedence constraint between a pair of jobs is a communication delay, the time duration the scheduler has to wait between the two jobs if they are scheduled on different machines. The objective is to assign the jobs to machines to minimize the makespan of the schedule. Despite being a fundamental problem in theory and a consequential problem in practice, the approximability of scheduling problems with communication delays is not very well understood. One of the top ten open problems in scheduling theory, in the influential list by Schuurman and Woeginger and its latest update by Bansal, asks if the problem admits a constant-factor approximation algorithm. In this paper, we answer this question in the negative by proving a logarithmic hardness for the problem under the standard complexity theory assumption that NP-complete problems do not admit quasi-polynomial-time algorithms.

Our hardness result is obtained using a surprisingly simple reduction from a problem that we call Unique Machine Precedence constraints Scheduling (UMPS). We believe that this problem is of central importance in understanding the hardness of many scheduling problems and we conjecture that it is very hard to approximate. Among other things, our conjecture implies a logarithmic hardness of related machine scheduling with precedences, a long-standing open problem in scheduling theory and approximation algorithms.

1 Introduction

We study the problem of scheduling jobs with precedence and non-uniform communication delay constraints on identical machines to minimize the makespan objective function. This classic model was first introduced by Rayward-Smith [RS87] and Papadimitriou and Yannakakis [PY90]. In this problem, we are given a set J of n jobs, where each job j has a processing length $p_j \in \mathbb{Z}_+$. The jobs need to be scheduled on m identical machines. The jobs have precedence and communication delay constraints, which are given by a partial order $\prec$. A constraint $j \prec j'$ encodes that job j' can only start after job j is completed. Moreover, if $j \prec j'$ and j, j' are scheduled on different machines, then j' can only start executing at least $c_{jj'}$ time units after j had finished. On the other hand, if j and j' are scheduled on the same machine, then j' can start executing immediately after j finishes. The goal is to schedule jobs *non-preemptively* to minimize the makespan objective function, which is defined as the completion time of the last job. In a non-preemptive schedule, each job j needs to be assigned to a single machine i and executed during a contiguous time interval of length p_j. In the classical scheduling notation, the problem is denoted by $\mathsf{P} \mid \mathsf{prec}, c_{jk} \mid C_{\max}$.[1] A closely related problem is $\mathsf{P}\infty \mid \mathsf{prec}, c_{jk} \mid C_{\max}$, where the scheduler has access to an unbounded number of machines.

Scheduling jobs with precedence and communication delays has been studied extensively over many years [RS87, PY90, MK97, HM01, TY92, HLV94, GKMP08]. Furthermore, due to its relevance in datacenter scheduling problems and large-scale training of ML models, there has been a renewed interest in more applied communities; we refer the readers to [CZM+11, GFC+12, HCG12, SZA+18, ZZC+12, ZCB+15, LYZ+16, NHP+19, MPL+17, GCL18, JZA19, TPD+20]. However, from a theoretical standpoint, besides NP-hardness results, very little was known in terms of the algorithms for the problem until the recent work by Maiti et al. [MRS+20] and Davies et al. [DKR+20, DKR+21]. These very recent papers designed polylogarithmic approximation algorithms for

*University of Washington, Seattle. Email: {daviess,rothvoss,yihaoz93}@uw.edu. Sami Davies was an intern at MSR Redmond when this work was done. Thomas Rothvoss is supported by NSF CAREER grant 1651861 and a David & Lucile Packard Foundation Fellowship.

†Microsoft Research, Redmond. Email: {jakul,jatarnaw}@microsoft.com.

‡Carnegie Mellon University, spallerl@andrew.cmu.edu. Research supported in part by NSF grants CCF-1563742 and CCF-1908125.

[1] We adopt the convention of [GLLK79, VLL90], where the respective fields denote: **(1) machine environment:** Q for related machines, P for identical machines, **(2) job properties:** prec for precedence constraints; c_{jk} for communication delays; c when all the communication delays are equal to c; $p_j = 1$ for the unit-length case, **(3) objective:** $C_{\max}$ for minimizing makespan.

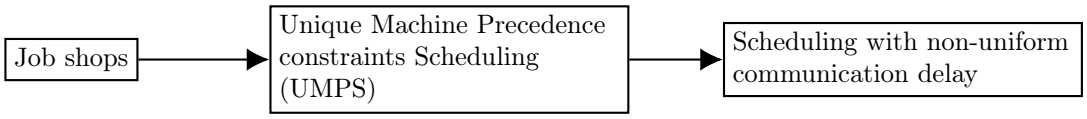

Figure 1: Role of the UMPS problem in our hardness reduction.

the special case when all the communication delays are *equal*. We survey these results in Section 1.2. In fact, the problems of scheduling jobs with communication delays are some of the well-known open questions in approximation algorithms and scheduling theory, and have resisted progress for a long time. For this reason, the influential survey by Schuurman and Woeginger [SW99] and its recent update by Bansal [Ban17] list understanding the approximability of the problems in this space as one of the top-10 open questions in scheduling theory.

In particular, an open problem in these surveys asks if the non-uniform communication delay problem on identical machines, even assuming an unbounded number of machines ($\mathsf{P}\infty \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}}$), admits a constant-factor approximation algorithm. The main result of this paper resolves this question.

THEOREM 1.1. *For every constant $\epsilon > 0$, assuming $\mathsf{NP} \not\subseteq \mathsf{ZTIME}\left(n^{(\log n)^{O(1)}}\right)$, the non-uniform communication delay problem ($\mathsf{P}\infty \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}}$) does not admit a polynomial-time $(\log n)^{1-\epsilon}$-approximation algorithm.*

We remark that our hard instances contain only two distinct values of communication delays (essentially 0 and ∞). Furthermore, as $\mathsf{P}\infty \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}}$, the problem with an unbounded number of machines, is a special case of $\mathsf{P} \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}}$, where the number of machines is specified, our theorem also implies the same hardness for $\mathsf{P} \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}}$.

1.1 Our Techniques Our hardness result is obtained using a reduction via a problem we call Unique Machines Precedence constraints Scheduling (UMPS). In this problem, there are m machines and n jobs $j_1, j_2, \ldots, j_n$ with precedence relations between them. Each job j_l has length $p(l)$ and can be scheduled only on a *unique* machine $M(l) \in [m]$. The objective is to schedule the jobs non-preemptively on the corresponding unique machines, respecting the precedence relations, so as to minimize the makespan objective function. Our proof of Theorem 1.1 proceeds via two steps:

1. First we show a reduction from an instance I of the UMPS problem to an instance I' of the non-uniform communication delay problem. The key step is to make sure that the set of jobs $J(i)$ that need to be scheduled on machine i in I do not get scheduled on multiple machines in I'. We achieve this by introducing a dummy job j_i^* and introducing precedence constraints from all jobs in $J(i)$ to j_i^* and a very large communication delay. This ensures that $J(i)$ and j_i^* are scheduled on the same machine in I', although this machine need not be i. Despite this, we show that any valid schedule of I' can be mapped back to a feasible schedule of I, with almost the same makespan. Our reduction creates only two types of communication delays and works for the unit-length case.

2. Next we observe that the UMPS problem generalizes the classical job shops problem (see e.g. [LLKS93, LMR94, MS11]), whose approximation is well understood [SSW94, CS00, GPSS01, FS02]. The logarithmic hardness result for the acyclic job shops problem by Mastrolilli and Svensson [MS11] implies a logarithmic hardness of the UMPS problem. We remark that the hardness result of [MS11] only works when jobs have lengths, and hence our Theorem 1.1 only applies to the setting where jobs have lengths.

In hindsight, our proof of Theorem 1.1 is surprisingly simple. However, the main conceptual contribution of our proof is in identifying the UMPS problem as a central problem that has implications for the hardness of various scheduling problems. Furthermore, the UMPS problem, which can be viewed as a generalization of the job shop scheduling model or as a highly restricted version of multidimensional scheduling with precedences, or as a restricted assignment problem with precedence constraints, is a fundamental problem to study on its own, both from a theoretical perspective and also from a practical point of view. We believe the UMPS problem is a key intermediate step towards resolving several long-standing open problems in scheduling theory. We make the following two conjectures regarding the approximability of UMPS.

CONJECTURE 1.1. *There exists a constant $\epsilon < 1$ such that it is NP-hard to approximate UMPS within a factor of n^{ϵ}, even when all jobs have unit lengths, where n is the number of jobs.*

CONJECTURE 1.2. *There exists an absolute constant $C \geq 1$ such that the following holds. For every constant $\epsilon > 0$, it is NP-hard to approximate UMPS within a $(\log n)^{1-\epsilon}$-factor, even when the number of machines m is at most $(\log n)^{C}$ and all the jobs have unit lengths, where n is the number of jobs.*

Our second main contribution is to show that the above conjectures imply hardness results for various problems. In particular, Conjecture 1.2 implies logarithmic hardness for scheduling with precedences on related machines, another top-10 problem in scheduling theory [SW99, Ban17] and in the approximation algorithm book of Shmoys and Williamson [WS11].

THEOREM 1.2. *Assuming Conjecture 1.2 and $\mathsf{NP} \not\subseteq \mathsf{DTIME}\left(n^{(\log n)^{O(1)}}\right)$, there exists an absolute constant $\gamma > 0$ such that the problem of scheduling related machines with precedences $(\mathsf{Q} \mid \mathsf{prec} \mid C_{\mathsf{max}})$ has no polynomial-time $O\left((\log m)^{\gamma}\right)$-factor approximation algorithm.*

Previously, Bazzi and Norouzi-Fard [BN15] introduced a k-partite hypergraph partition problem whose hardness implies a superconstant hardness for scheduling with precedences on related machines. Our reduction uses the same idea of job replication as [BN15], while our soundness analysis is technically more involved. We also show that the hypothesis of [BN15] implies a superconstant hardness of the UMPS problem. Thus, our problem can be viewed as a weaker version of the hypothesis of [BN15] with the same implication towards the hardness of related machines. Furthermore, stronger hardness of the UMPS problem implies better (almost optimal) hardness results for the related machines scheduling problem.

Finally, we note that Conjecture 1.1 implies that precedence-constrained scheduling (even without communication delays) is very hard to approximate when generalized to the restricted assignment setting or unrelated machines.

Our confidence in the above conjectures stems from the fact that existing techniques, both the classical jobshops algorithms [LMR94] and the recent LP-hierarchies-based algorithms [MRS+20, DKR+20] fail to give non-trivial approximation guarantees for the UMPS problem. Furthermore, a candidate hard instance for the problem is a *layered* instance, where there are precedences between jobs $j_1 \prec j_2$ only if j_1 can be scheduled on the machine i and j_2 can be scheduled on the machine $i+1$. These layered instances are closely related to the k-partite partitioning hypothesis of [BN15] and the integrality gap instances [MRS+20] for the problem of scheduling with uniform communication delays.

1.2 A Brief History of the Communication Delay Problem In this subsection, we give a brief overview of the literature on the problem of scheduling with communication delays.

Scheduling with precedences. Scheduling with precedences to minimize makespan $(\mathsf{P}|\mathsf{prec}|C_{\mathsf{max}})$ is a classic combinatorial optimization problem and is a special case of the communication delay problem with $c = 0$ for all pairs of jobs. In one of the earliest results in the scheduling theory, Graham's list scheduling algorithm [Gra66] was shown to be a 2-factor approximation for the problem. Recently, Svensson [Sve10] gave a matching hardness of approximation result assuming (a variant of) the Unique Games Conjecture [BK09]. When the number of machines is a constant, a series of recent works have obtained $(1+\epsilon)$-approximation in nearly quasi-polynomial time [LR16, Gar18, KLTY20, Li21].

Uniform communication delay setting. The problem becomes much harder with communication delays, even when all the communication delays are equal. This problem is denoted by $\mathsf{P} \mid \mathsf{prec}, c \mid C_{\mathsf{max}}$ and is referred to as scheduling with *uniform* communication delays. In this setting, Graham's list scheduling algorithm obtains a $(c+1)$-factor approximation. This was improved to $2/3 \cdot (c+1)$ by Giroudeau et al. [GKMP08] in the case when the jobs have unit lengths $(\mathsf{P}\infty \mid \mathsf{prec}, p_j = 1, c \geq 2 \mid C_{\mathsf{max}})$. In recent concurrent and independent works, poly-logarithmic-factor approximation algorithms have been obtained for the uniform communication delays problem $\mathsf{P} \mid \mathsf{prec}, c \mid C_{\mathsf{max}}$ by Maiti et al. [MRS+20] and Davies et al. [DKR+20, DKR+21].

On the hardness front, when $c = 1$, Hoogeveen, Lenstra and Veltman [HLV94] showed that the problem $\mathsf{P}\infty \mid \mathsf{prec}, p_j = 1, c = 1 \mid C_{\mathsf{max}}$ is NP-hard to approximate to a factor better than 7/6. The result has been generalized for $c \geq 2$ to $(1 + 1/(c+4))$-hardness [GKMP08].[2]

[2] Papadimitriou and Yannakakis [PY90] claim a 2-hardness for $\mathsf{P}\infty \mid \mathsf{prec}, p_j = 1, c \mid C_{\mathsf{max}}$, but give no proof. Schuurman and

Scheduling with non-uniform communication delay. We do not know of any algorithm for the non-uniform communication delays ($\mathsf{P}\infty \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}}$) problem. On the hardness side, the best hardness known is the above small constant hardness of the uniform communication delay setting. While our main result shows logarithmic hardness for this problem, it is conceivable that it admits a polylog-approximation algorithm, although our conjectures suggest otherwise.

Duplication model. Scheduling with communication delays problem has also been studied in the duplication model, where we allow jobs to be duplicated (replicated), i.e., executed on more than one machine to avoid communication delays. In this easier model, for the general $\mathsf{P}\infty \mid \mathsf{prec}, p_j, c_{jk}, \mathsf{dup} \mid C_{\mathsf{max}}$ problem, there is a simple 2-factor approximation algorithm by Papadimitriou and Yannakakis [PY90]. On the other hand, [PY90] also show the NP-hardness of $\mathsf{P}\infty \mid \mathsf{prec}, p_j = 1, c, \mathsf{dup} \mid C_{\mathsf{max}}$. Note that the $O(1)$-approximation algorithm for the version with duplication is in sharp contrast to our hardness result (Theorem 1.1) illustrating that the problem is significantly harder without duplication.

1.3 Discussion and Open Problems

While we make progress on the hardness of approximation of scheduling with non-uniform communication delay, the main conceptual contribution of this work is initiating the formal study of the UMPS problem. When jobs have lengths, the problem does not admit a polylogarithmic approximation. However, much less is known for the unit-length case. We now mention a few open problems in this direction.

1. The key open problem is to prove (or disprove) Conjecture 1.2. A positive resolution of the conjecture would prove the hardness of scheduling related machines with precedences, a long-standing open problem in scheduling theory. By the same reduction as in the proof of Theorem 1.1, Conjecture 1.2 also implies a logarithmic hardness of approximation for the non-uniform communication delay problem even when the jobs have unit lengths ($\mathsf{P}\infty \mid \mathsf{prec}, p_j = 1, c_{jk} \mid C_{\mathsf{max}}$).

2. On the other hand, obtaining good approximation algorithms for the UMPS problem would be even more exciting. Is Conjecture 1.1 true, or is there a polylog-factor approximation algorithm for the unit-length case?

1.4 Organization

The rest of the paper is organized as follows. We first formally define the UMPS problem and relate it to the jobshops problem in Section 2. We then use the hardness of the UMPS problem to prove Theorem 1.1 in Section 3. Finally, in Section 4, we show that Conjecture 1.2 implies an improved hardness of related machine scheduling with precedences and that the hypothesis of [BN15] implies a superconstant hardness of the UMPS problem with unit lengths.

2 Unique Machine Precedence Constraints Scheduling problem

We first formally define the Unique Machine Precedence constraint Scheduling (UMPS) problem.

DEFINITION 2.1. *(Unique Machine Precedence constraint Scheduling) In the Unique Machine Precedence constraint Scheduling (UMPS) problem, the input is a set of m machines and n jobs $j_1, j_2, \ldots, j_n$ with precedence relations between them. Furthermore, each job j_l can be scheduled only on a fixed machine $M(l) \in [m]$, and takes $p(l)$ time to complete. The jobs should be scheduled non-preemptively, i.e., once a machine starts processing a job j_l, it has to finish it before processing other jobs. The objective is to schedule the jobs on the corresponding machines in this non-preemptive manner while respecting the precedence relations, so as to minimize the makespan.*

We note that the UMPS problem is a generalization of the classical jobshops problem that we formally define below.

DEFINITION 2.2. *(Job shops) In the jobshops problem, the input is a set of n jobs to be processed on a set M of m machines. Each job j consists of μ_j operations $O_{1,j}, O_{2,j}, \ldots, O_{\mu_j,j}$. Operation $O_{i,j}$ must be processed for $p_{i,j}$ units of time without interruptions on the machine $m_{i,j} \in M$, and can only be scheduled if all the preceding operations $O_{i',j}, i' < i$ have finished processing. The objective is to schedule all the operations on the corresponding machines to minimize the makespan.*

Woeginger [SW99] remark that "it would be nice to have a proof for this claim".

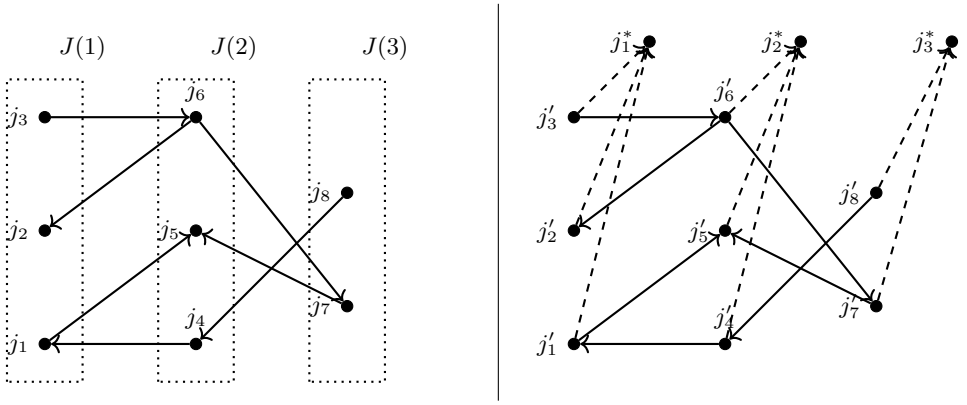

Figure 2: Illustration of the reduction from UMPS to non-uniform communication delays. In the communication delay instance on the right, the dashed arrow precedences have communication delay C_∞ while the normal arrow precedences have communication delay 0.

Note that jobshops problem is a special case of the UMPS problem, corresponding to the case when the precedence DAG is a disjoint union of chains. The jobshops problem has received a lot of attention and played an important role in the development of key algorithmic techniques [LLKS93, LMR94]. On the hardness front, Mastrolilli and Svensson showed almost optimal hardness results for the problem in a breakthrough result [MS11].

THEOREM 2.1. *For every constant $\epsilon > 0$, assuming $\mathsf{NP} \not\subseteq \mathsf{ZTIME}\left(n^{(\log n)^{O(1)}}\right)$, there is no polynomial-time $(\log n)^{1-\epsilon}$-factor approximation algorithm for the jobshops problem, where n is the total number of operations in the given jobshops instance.*

As a corollary, we obtain the following hardness result.

COROLLARY 2.1. *For every constant $\epsilon > 0$, assuming $\mathsf{NP} \not\subseteq \mathsf{ZTIME}\left(n^{(\log n)^{O(1)}}\right)$, there is no polynomial-time $(\log n)^{1-\epsilon}$-factor approximation algorithm for the UMPS problem.*

3 Hardness of Scheduling With Non-Uniform Communication Delays

We now give a reduction from the UMPS problem to the non-uniform communication delay problem, thereby proving the hardness of the non-uniform communication delay problem. We restate the theorem for convenience.

THEOREM 3.1. *For every constant $\epsilon > 0$, assuming $\mathsf{NP} \not\subseteq \mathsf{ZTIME}\left(n^{(\log n)^{O(1)}}\right)$, the non-uniform communication delay problem $(\mathsf{P}\infty \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}})$ does not admit a polynomial-time $(\log n)^{1-\epsilon}$-approximation algorithm.*

Reduction. Let I be an instance of the UMPS problem with n jobs $j_1, j_2, \ldots, j_n$, and m machines. Furthermore, each job j_l has a processing time $p(l)$ and can be scheduled only on the machine $M(l) \in [m]$. For an index $i \in [m]$, let $J(i) \subseteq \{j_1, j_2, \ldots, j_n\}$ denote the set of jobs that can be scheduled on the machine i.

Roughly speaking, our idea in the reduction is to output a non-uniform communication delay instance where we force the jobs in $J(i)$ to be scheduled on the same machine, for every $i \in [m]$. We achieve this by adding a set of m dummy jobs $j_1^*, j_2^*, \ldots, j_m^*$ and adding precedences with very large communication delay from all the jobs in $J(i)$ to j_i^* for every $i \in [m]$. More formally, we define an instance I' of the non-uniform communication delay problem as follows. First, we choose a large integer $C_\infty = n \sum_{l=1}^{n} p(l)$. There are $n+m$ jobs in I': a set of n jobs $j_1', j_2', \ldots, j_n'$ such that for each $l \in [n]$, the processing time of j_l' is equal to $p(l)$, and a set $\{j_1^*, j_2^*, \ldots, j_m^*\}$ of m dummy jobs, each with processing time 1. For every precedence relation $j_u \prec j_v$ in the original instance I, there is a precedence relation $j_u' \prec j_v'$ in I' with communication delay 0. Finally, for every i, and every job $j_l \in J(i)$, there is a precedence relation $j_l' \prec j_i^*$ with communication delay C_∞.

Completeness. Suppose that there is a schedule for I with makespan at most L. Then, we claim that there is a schedule for I' with makespan at most $L+1$. We use m machines and schedule the job j_l' on the machine $M(l)$

in the same time slot used by the schedule for I. As the communication delay of the precedences among the jobs $\{j'_1, j'_2, \ldots, j'_n\}$ is zero, we can schedule these jobs using m machines with makespan at most L. Now, after all the jobs $\{j'_1, j'_2, \ldots, j'_n\}$ have been scheduled, we schedule the job j^*_i in the machine i, for every $i \in [m]$. As we are scheduling all the jobs in $J(i)$ and j^*_i on the same machine for every $i \in [m]$, we incur no communication delay when we are scheduling the dummy jobs, and we can schedule all the dummy jobs $j^*_1, j^*_2, \ldots, j^*_m$ simultaneously in the time slot between L and $L+1$.

Soundness. Suppose that there is a schedule for I' with makespan at most L. Then, we claim that there is a schedule for I with makespan at most L as well.

Note that there is a trivial schedule for I where we schedule each job one by one after topologically sorting them, that has a makespan of $\sum_{j=1}^{n} p(j)$. Thus, henceforth, we assume that $L \leq \sum_{j=1}^{n} p(j)$. For an index $i \in [m]$, let $J'(i)$ be the subset of jobs in I' whose corresponding jobs in I are to be scheduled on the machine i:

$$J'(i) := \{j'_l : M(l) = i\}.$$

We claim that in the schedule for I' with makespan at most L, for every $i \in [m]$, all the jobs in $J'(i)$ must be scheduled on the same machine. Suppose for the sake of contradiction that this is not the case. If there are jobs j'_{l_1} and j'_{l_2} such that $M(l_1) = M(l_2) = i$ are scheduled on different machines i_1, i_2 in the schedule for I', at least one of j'_{l_1} and j'_{l_2} is scheduled on a different machine than j^*_i. However, as there are precedence relations $j'_{l_1} \prec j^*_i$ and $j'_{l_2} \prec j^*_i$ with communication delay C_∞, at least one of the precedence relations has to wait for the communication delay, and thus, the makespan is at least $C_\infty > L$, a contradiction.

Thus, for every $i \in [m]$, all the jobs in $J'(i)$ are processed on the same machine in I'. This implies that at any point of time, at most one job from $J'(i)$ is being processed, for every $i \in [m]$. Using this observation, we output a schedule for I: for every job $j_l \in J(i)$, we schedule j_l in the same time slot used by the job j'_l in the schedule for I'. By the above observation, every machine $i \in [m]$ is used at most once at any time point. Furthermore, as the schedule for I' respects the precedence conditions, the new schedule for I also respects the precedence conditions. Note that the makespan of this schedule for I is equal to L. This completes the proof that there exists a schedule for I with makespan at most L, if there exists a schedule for I' with makespan at most L.

This completes the proof of Theorem 1.1. We remark that the same reduction also proves a $(\log n)^{1-\epsilon}$-factor inapproximability of the bounded-machines version $\mathsf{P} \mid \mathsf{prec}, c_{jk} \mid C_{\mathsf{max}}$ of the non-uniform communication delay problem.

4 Conditional Hardness of Scheduling With Precedence Constraints on Related Machines

In this section, we first prove that Conjecture 1.2 implies improved hardness of scheduling related machines with precedences.

We begin by formally defining the scheduling related machines with precedences problem $(\mathsf{Q} \mid \mathsf{prec} \mid C_{\mathsf{max}})$.

Definition 4.1. *(Scheduling related machines with precedences) In the scheduling related machines with precedences problem, the input is a set of m machines $\mathcal{M}$ and a set of n jobs $\mathcal{J}$ with precedences among them. Furthermore, each machine i has speed $s_i \in \mathbb{Z}^+$, and each job j has processing time $p_j \in \mathbb{Z}^+$, and scheduling the job j on machine i takes $\frac{p_j}{s_i}$ units of time. The objective is to schedule the jobs on the machines non-preemptively respecting the precedences constraints, to minimize the makespan.*

An algorithm with $O(\log m)$ approximation guarantee for the problem was given independently by Chudak and Shmoys [CS99], and Chekuri and Bender [CB01]. On the hardness side, a hardness factor of 2 follows from the identical machines setting [Sve10], assuming a variant of the Unique Games Conjecture. Furthermore, Bazzi and Norouzi-Fard [BN15] put forth a hypothesis on the hardness of a k-partite graph partitioning problem, which implies a super constant hardness of the scheduling related machines with precedences problem.

We now prove that Conjecture 1.2 implies poly logarithmic hardness of scheduling related machines with precedences problem.

Theorem 4.1. *Assuming Conjecture 1.2 and $\mathsf{NP} \not\subseteq \mathsf{DTIME}\left(n^{(\log n)^{O(1)}}\right)$, there exists an absolute constant $\gamma > 0$ such that the problem of scheduling related machines with precedences $(\mathsf{Q} \mid \mathsf{prec} \mid C_{\mathsf{max}})$ has no polynomial-time $O\left((\log m)^{\gamma}\right)$-factor approximation algorithm.*

Reduction. Our reduction is essentially the same reduction as in [BN15] where the authors obtained conditional hardness of the related machine scheduling with precedences problem assuming the hardness of a certain k-partite graph partitioning problem. However, our soundness analysis needs more technical work.

We start with an instance I of the UMPS problem with n unit sized jobs $j_1, j_2, \ldots, j_n$ and m machines, and every job j_l can only be scheduled on the machine $M(l) \in [m]$. Furthermore, we let $J(i) \subseteq [n], i \in [m]$ denote the set of all the jobs that can be scheduled on the machine i.

We now output an instance I' of the related machine scheduling problem. We choose a parameter $\kappa = 10n^3m$. For every $l \in [n]$, we have a set $\mathcal{J}_l$ of $\kappa^{2(m-M(l))}$ jobs in I'. The processing time of each of these jobs is equal to $\kappa^{M(l)-1}$. For every $i \in [m]$, we have $\mathcal{M}_i$, a set of $\kappa^{2(m-i)}$ machines, each with speed κ^{i-1}. Furthermore, for every precedence constraint $j_u \prec j_v$ in I, we have $j'_{l_1} \prec j'_{l_2}$ for every $j'_{l_1} \in \mathcal{J}_u$ and $j'_{l_2} \in \mathcal{J}_v$.

Completeness. Suppose that there is a scheduling of I with makespan equal to L. Then, we claim that there is a scheduling of I' with makespan at most L as well. Note that all the jobs in $\mathcal{J}_l$ can be scheduled on the machines $\mathcal{M}_{M(l)}$ in unit time. We obtain a scheduling of I' by assigning the jobs $\mathcal{J}_l$ to the machines $\mathcal{M}_{M(l)}$ in the time slot used in I to schedule the job j_l. This scheduling of I' is indeed a valid scheduling, and has a makespan of at most L.

Soundness. We prove the soundness part in the lemma below.

LEMMA 4.1. *Suppose that there is a scheduling of I' with makespan L. Then, we will show that there is a scheduling of I with makespan at most $2L$.*

Proof. Note that there is a trivial scheduling of I where we schedule jobs in a topological sort one by one, with makespan equal to n. Thus, henceforth, we assume that $L \leq n$.

Let $\gamma = \frac{1}{10n^2}$. We claim that for every $l \in [n]$, at most $\gamma\kappa^{2(m-M(l))}$ jobs in $\mathcal{J}_l$ are processed by machines that do not belong to $\mathcal{M}_{M(l)}$ in the scheduling I'. The proof of this claim follows from Lemma 1 of [BN15], and we present it here for the sake of completeness. Fix an index $l \in [n]$, and for ease of notation, let $i = M(l)$. First, as each job in $\mathcal{J}_l$ has length κ^{i-1}, and the processing speed of each machine in $\mathcal{M}_j, j < i$ is at most $\kappa^{j-1} \leq \kappa^{i-2}$, no job in $\mathcal{J}_l$ is scheduled on machines in $\mathcal{M}_j$, $j < i$, as the makespan of I' is at most $n < \kappa$. Now, consider an integer $j \in [m]$, $j > i$. There are $\kappa^{2(m-j)}$ machines in $\mathcal{M}_j$, and they have a processing speed of κ^{j-1}. Thus, in time $L \leq n$, they can process at most

$$\frac{n \cdot \kappa^{2(m-j)} \cdot \kappa^{j-1}}{\kappa^{i-1}} \leq \frac{n}{\kappa} \cdot \kappa^{2(m-i)}$$

jobs of $\mathcal{J}_l$. Taking union over all $j > i$, we get that at most

$$\frac{nm}{\kappa} \cdot \kappa^{2(m-i)} \leq \frac{1}{10n^2}\kappa^{2(m-i)}$$

jobs in $\mathcal{J}_l$ are processed by machines outside $\mathcal{M}_i$. In other words, for every job j_l of I, at most γ fraction of the jobs in $\mathcal{J}_l$ are processed by machines outside $\mathcal{M}_{M(l)}$.

Now, consider a scheduling of the jobs in I' where for every $l \in [n]$, we get rid of the jobs in $\mathcal{J}_l$ that are processed by machines outside $\mathcal{M}_{M(l)}$. After removing the jobs processed by other machines, we still have that for every $l \in [n]$, at least $1-\gamma$ fraction of the jobs in $\mathcal{J}_l$ are processed. Also observe that since we are only deleting some jobs, the makespan of the new scheduling is at most L as well. Recall that processing each job in $\mathcal{J}_l$ takes unit time on the machines in $\mathcal{M}_{M(l)}$.

Using this observation, we obtain a fractional scheduling of I in time L as follows. For every $l \in [n]$ and $t \in [L]$, define the variable $x_{l,t}$ to be the fraction of the jobs of $\mathcal{J}_l$ that are scheduled by the machines $\mathcal{M}_{M(l)}$ in the time slot t. By the above discussion, we get the following properties of this fractional scheduling.

1. Every job $l \in [n]$ is almost fully processed. For every $l \in [n]$, we have

$$\sum_{t=1}^{L} x_{l,t} \geq 1 - \gamma$$

2. Every machine is used only for processing a single unit of job in a time slot.

$$\sum_{l \in J(i)} x_{l,t} \leq 1 \ \forall i \in [m], t \in [L].$$

3. If there is a precedence constraint $j_{l_1} \prec j_{l_2}$ in I, l_2's processing is done only in the time slots after l_1 is fully processed. More formally,

$$x_{l_1,t} > 0 \Rightarrow x_{l_2,t'} = 0 \, \forall t' \leq t$$

We will now show that the fractional scheduling implies that the instance I has an integral scheduling with makespan at most $O(L)$, thereby proving the Lemma. We will prove this in two steps: first, we modify the fractional scheduling to obtain another fractional scheduling with better structure, and then next, we use this to obtain the integral scheduling.

For a job $l \in [n]$, define the starting time t^s_l and the end time t^e_l as the minimum and the maximum times at which l is being processed.

$$t^s_l = \min\{t : x_{l,t} > 0\} \, , \, t^e_l = \max\{t : x_{l,t} > 0\}$$

Note that if we have $j_{l_1} \prec j_{l_2}$, $t^s_{l_2} > t^e_{l_1}$. We now modify the fractional scheduling to ensure that each machine processes the job with the lowest ending time first, from the available set of the jobs. More formally, for a machine $i \in [m]$, consider the pair of jobs $l_1, l_2 \in J(i)$ and time slot $t \in [L]$ satisfying the following conditions.

(C1) The job l_1 has lower ending time: $t^e_{l_1} < t^e_{l_2}$, or $t^e_{l_1} = t^e_{l_2}$ and $l_1 < l_2$.

(C2) The job l_1 can be processed on the time slot t, but the job l_2 is processed instead of finishing the job l_1:

$$t^s_{l_1} \leq t < t^e_{l_1} \, , \, x_{l_2,t} > 0$$

If there are jobs l_1, l_2 and time slot t satisfying these conditions, we swap the processing times, and process the job l_1 in the time slot t instead of l_2. More formally, let $t' > t$ be such that $x_{l_1,t'} > 0$. Let $y = \min(x_{l_1,t'}, x_{l_2,t})$. We obtain a new fractional scheduling by setting

$$x_{l_1,t'} = x_{l_1,t'} - y \, , \, x_{l_1,t} = x_{l_1,t} + y$$
$$x_{l_2,t'} = x_{l_2,t'} + y \, , \, x_{l_2,t} = x_{l_2,t} - y$$

Note that the operation does not increase the ending time of either job and does not decrease the starting time of either job and thus, results in a valid fractional scheduling respecting the precedence conditions. We repeat the swapping operations until there is no triple i, j, t left where both (C1) and (C2) are true. We also update the starting and ending times of the jobs t^s_l and t^e_l appropriately when we apply the swapping operations.

Next, we apply another transformation to the fractional scheduling by filling the empty slots in the machines, if there are any. More formally, consider a time slot $t \in [L]$ and job $l \in [n]$ such that the following hold.

(D1) The time slot t is not fully utilized:

$$\sum_{l' \in J(M(l))} x_{l',t} < 1$$

(D2) The job l can be scheduled on the time slot t instead of leaving the machine idle: $t^s_l \leq t < t^e_l$.

If there is a time slot t and job l such that the above two conditions hold, we fill the empty slot in the time slot t by processing the job l. Let $t' > t$ be such that $x_{l,t'} > 0$. Let $y = \min(x_{l,t'}, 1 - \sum_{l' \in J(M(l))} x_{l',t})$. We set

$$x_{l,t} = x_{l,t} + y \, , \, x_{l,t'} = x_{l,t'} - y$$

We repeat these operations iteratively until no empty slots can be filled. Similar to the previous case, we update the starting and ending times of the jobs appropriately.

After the two types of operations, we obtain a fractional scheduling with the following property: at every time slot t, for a machine $i \in [m]$, let $S_{i,t}$ be the set of jobs that can be scheduled on i in the time slot t:

$$S_{i,t} := \{l \in J(i) : t^s_l \leq t \leq t^e_l\}$$

We sort the jobs in $S_{i,t}$ as $\{l_1, l_2, \ldots, l_k\}$ by increasing order of ending times, and breaking ties based on the index. The fractional scheduling greedily schedules the jobs $l_1, l_2, \ldots,$ in that order. More formally, we have

$$x_{l_1,t} = \sum_{t'=1}^{L} x_{l_1,t'} - \sum_{t'=1}^{t-1} x_{l_1,t'}$$

and

$$x_{l_2,t} = \min\left(1 - x_{l_1,t}, \sum_{t'=1}^{L} x_{l_2,t'} - \sum_{t'=1}^{t-1} x_{l_2,t'}\right)$$

and so on.

Our goal is to show that in this final fractional scheduling that we obtained, each machine schedules at most two jobs in any time slot. In order to prove this, we first define the following parameter, $P_{i,t}$, the amount of jobs *partially* completed in the machine i by the time t.

$$P_{i,t} = \sum_{l \in J(i): t_l^e > t} \sum_{t'=1}^{t} x_{l,t'}$$

We claim that for every $i \in [m], t \in [L]$, we have $P_{i,t} \leq \gamma t$. Fix a machine $i \in [m]$. We will prove the claim by induction on t.

1. Base case when $t = 1$. If no job is processed by the machine i in the time slot $t = 1$, the claim is trivially satisfied. Else, let l_1 be the job in $J(i)$ with the lowest ending time, breaking ties by the lowest index. Note that the fractional scheduling fully schedules the job l_1 in the time slot $t = 1$. As each job is processed for at least $1 - \gamma$ duration, we get that $P_{i,1}$ is at most γ.

2. Inductive proof. Suppose that the claim holds for all $t' \leq t$ and consider the time slot $t + 1$. For ease of notation, let $S = S_{i,t+1}$ be the set of jobs that can be processed on the machine i in the time slot $t + 1$. If S is empty, the inductive claim trivially holds. Else, let $l \in S$ be the job with the lowest ending time (breaking ties by the least index). Note that the modified fractional scheduling finishes the job l in the time slot $t + 1$. Let $x'_{l,t}$ denote the amount of the job l that is processed by time t i.e., $x'_{l,t} = \sum_{t'=1}^{t} x_{l,t}$. The amount of jobs that are partially finished at the end of time slot $t + 1$ is at most

$$\begin{aligned} P_{i,t+1} &\leq P_{i,t} - x'_{l,t} + (1 - x_{l,t+1}) \\ &\leq P_{i,t} + \gamma \leq (t+1)\gamma \end{aligned}$$

We will now show that every machine processes at most 2 jobs in a time slot. Consider a machine $i \in [m]$ and time slot $t \in [L]$. Let $S_{i,t} := \{l_1, l_2, \ldots, l_k\}$. By the previous claim, we know that at most $(t-1)\gamma$ portion of the job l_u is finished before time t, for every $u \in [k]$. Note that $(t-1)\gamma \leq L\gamma \leq \frac{1}{10n}$. Thus, the greedy fractional scheduling can only schedule at most two jobs, as each of them takes at least $1 - \frac{1}{10n}$ time. Finally, using this observation, we can duplicate every time slot to obtain an integral scheduling of I with makespan at most $2L$. ∎

Parameter analysis. The number of machines in the related machines scheduling instance is $M = \kappa^{O(m)} = n^{O(m)}$, while the hardness gap is $(\log n)^{1-\epsilon'}$ for every $\epsilon' > 0$. By setting ϵ' appropriately, we get a hardness of $(\log M)^{\Omega(1)}$ for the scheduling related machines with precedences problem.

4.1 Hypothesis of [BN15] implies superconstant hardness of the UMPS problem with unit lengths

Bazzi and Norouzi-Fard [BN15] introduced the following hypothesis and proved that it implies a superconstant hardness for scheduling related machines with precedences.

HYPOTHESIS 4.1. ([BN15]) *For every $\epsilon, \delta > 0$ and constant integers $k, Q > 0$, the following problem is NP-hard. Given a k-partite graph $G = (V_1, V_2, \ldots, V_k, E_1, E_2, \ldots, E_{k-1})$ with $|V_i| = n$ for all $1 \leq i \leq k$, and E_i being the set of edges between V_i and V_{i+1} for every $1 \leq i < k$, distinguish between the two cases:*

1. *(YES case) Every V_i can be partitioned into $V_{i,0}, V_{i,1}, \ldots, V_{i,Q-1}$ such that*
   - *There is no edge between V_{i,j_1} and V_{i+1,j_2} for all $1 \leq i < k, j_1 > j_2 \in [Q]$.*
   - *$|V_{i,j}| \geq \frac{(1-\epsilon)}{Q} n$ for all $i \in [k], j \in [Q]$.*
2. *(NO case) For every $1 < i \leq k$, and any two sets S, T with $S \subseteq V_i$, $T \subseteq V_{i-1}$, $|S| = |T| = \delta n$, there is an edge between S and T.*

We now prove that the above hypothesis implies that it is NP-hard to obtain a constant factor approximation algorithm for the UMPS problem, even when all the jobs have unit length.

Reduction. Given an instance of k-partite problem I, we output an instance I' of the UMPS problem as follows: there are $n' = nk$ unit sized jobs in I', one job corresponding to each vertex of G. There are k machines, and all the jobs in $V_i, i \in [k]$ can only be scheduled on the machine i. For every edge $e = (u, v)$ in the graph such that $u \in V_i, v \in V_{i+1}$, we have a precedence condition $u \prec v$ in I'. We choose the parameter $Q = k$, and $\delta = \epsilon = \frac{1}{k}$.

Completeness. Suppose that the YES case of Hypothesis 4.1 holds i.e., there is a partition of V_i into $V_{i,0}, V_{i,1}, \ldots, V_{i,Q-1}$ respecting the two conditions above. Then, we claim that there is a scheduling of I' with makespan at most $3n$. For every machine $i \in [k]$, we schedule the jobs in $V_{i,0}$ (in arbitrary order), and then the jobs in $V_{i,1}$ (in arbitrary order) and so on. However, we start the execution of the jobs in $V_{i,0}$ at time t_i, and then, execute the jobs in $V_{i,l}$ immediately after the execution of the jobs in $V_{i,l-1}$ for all $l \geq 1$. The parameters t_i, $i \in [k]$ are chosen such that for every pair of jobs u, v with $u \prec v$, u is guaranteed to have scheduled before v. In particular, we choose $t_i = (i-1)n\left(\epsilon + \frac{1}{Q}\right)$.

We now prove that this results in a valid scheduling that respects the precedence conditions. Consider a pair of jobs u, v such that $u \in V_i, v \in V_{i+1}$ such that $u \prec v$. As the k-partite graph satisfies the YES condition, we have integers j_1, j_2 such that $u \in V_{i,j_1}$ and $v \in V_{i+1,j_2}$, and $j_1 \leq j_2$. Note that u is processed by time at most

$$t_u = t_i + |V_{i,0}| + |V_{i,1}| + \ldots + |V_{i,j_1}|$$

Furthermore, v is processed only after time

$$t_v = t_{i+1} + |V_{i+1,0}| + |V_{i+1,1}| + \ldots + |V_{i+1,j_2-1}|$$

We have

$$\begin{aligned}
t_v - t_u &= t_{i+1} + |V_{i+1,0}| + |V_{i+1,1}| + \ldots + |V_{i+1,j_2-1}| - (t_i + |V_{i,0}| + |V_{i,1}| + \ldots + |V_{i,j_1}|) \\
&= n\left(\epsilon + \frac{1}{Q}\right) + |V_{i+1,0}| + |V_{i+1,1}| + \ldots + |V_{i+1,j_2-1}| - (n - |V_{i,j_1+1}| + |V_{i,j_1+2}| + \ldots + |V_{i,Q-1}|) \\
&\geq n\left(\epsilon + \frac{1}{Q}\right) + j_2\frac{(1-\epsilon)n}{Q} - \left(n - \frac{(Q - j_1 - 1)(1-\epsilon)n}{Q}\right) \\
&\geq n\left(\epsilon + \frac{1}{Q}\right) + j_1\frac{(1-\epsilon)n}{Q} - \left(j_1\frac{(1-\epsilon)n}{Q} + \epsilon n + \frac{(1-\epsilon)n}{Q}\right) \geq 0
\end{aligned}$$

Thus, the schedule is a valid scheduling of I'. The makespan of this scheduling is at most $t_k + n = (k-1)n\left(\epsilon + \frac{1}{Q}\right) + n \leq 3n$.

Soundness. Suppose that the NO case of Hypothesis 4.1 holds. We claim that in this case, the makespan of I' is at least $(1-2\delta)kn$. For every $i \in [k]$, let s_i denote the time at which the machine i has finished $(1-\delta)n$ jobs of V_i. For an index $i \in [k]$, let $S(i) \subseteq V_i$ denote the set of jobs that are not processed by the time s_i. By the definition of s_i, we have $|S(i)| \geq \delta n$. By the NO case of Hypothesis 4.1, we get that there are at least $(1-\delta)n$ jobs in V_{i+1} that have dependencies in $S(i)$. Note that all these jobs can be scheduled only after s_i. Thus, we get

$$s_{i+1} \geq s_i + (1-2\delta)n \ \forall i \in [k-1]$$

Summing over all i, we get that the makespan of the scheduling is at least $(1-2\delta)kn$, which is at least $\frac{kn}{2}$ when $k \geq 4$. By choosing k large enough, this completes the proof that assuming Hypothesis 4.1, it is NP-hard to obtain a $O(1)$ factor approximation algorithm for the UMPS problem when the jobs have unit lengths.

References

[Ban17] Nikhil Bansal. Scheduling open problems: Old and new. MAPSP 2017. http://www.mapsp2017.ma.tum.de/MAPSP2017-Bansal.pdf, 2017.

[BK09] Nikhil Bansal and Subhash Khot. Optimal long code test with one free bit. In *50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, October 25-27, 2009, Atlanta, Georgia, USA*, pages 453–462. IEEE Computer Society, 2009.

[BN15] Abbas Bazzi and Ashkan Norouzi-Fard. Towards tight lower bounds for scheduling problems. In *Algorithms - ESA 2015 - 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*, volume 9294 of *Lecture Notes in Computer Science*, pages 118–129. Springer, 2015.

[CB01] Chandra Chekuri and Michael A. Bender. An efficient approximation algorithm for minimizing makespan on uniformly related machines. *J. Algorithms*, 41(2):212–224, 2001.

[CS99] Fabián A. Chudak and David B. Shmoys. Approximation algorithms for precedence-constrained scheduling problems on parallel machines that run at different speeds. *J. Algorithms*, 30(2):323–343, 1999.

[CS00] Artur Czumaj and Christian Scheideler. A new algorithm approach to the general lovász local lemma with applications to scheduling and satisfiability problems (extended abstract). In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 38–47. ACM, 2000.

[CZM+11] Mosharaf Chowdhury, Matei Zaharia, Justin Ma, Michael I. Jordan, and Ion Stoica. Managing data transfers in computer clusters with orchestra. In *Proceedings of the ACM SIGCOMM 2011 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, Toronto, ON, Canada, August 15-19, 2011*, pages 98–109. ACM, 2011.

[DKR+20] Sami Davies, Janardhan Kulkarni, Thomas Rothvoss, Jakub Tarnawski, and Yihao Zhang. Scheduling with communication delays via LP hierarchies and clustering. *To appear at 61st Annual IEEE Symposium on Foundations of Computer Science (FOCS 2020), 16-19 November 2006, Durham, North Carolina, USA, Proceedings*, 2020.

[DKR+21] Sami Davies, Janardhan Kulkarni, Thomas Rothvoss, Jakub Tarnawski, and Yihao Zhang. Scheduling with communication delays via LP hierarchies and clustering II: weighted completion times on related machines. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 2958–2977. SIAM, 2021.

[FS02] Uriel Feige and Christian Scheideler. Improved bounds for acyclic job shop scheduling. *Comb.*, 22(3):361–399, 2002.

[Gar18] Shashwat Garg. Quasi-PTAS for scheduling with precedences using LP hierarchies. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, pages 59:1–59:13, 2018.

[GCL18] Yuanxiang Gao, Li Chen, and Baochun Li. Spotlight: Optimizing device placement for training deep neural networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1676–1684, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[GFC+12] Zhenyu Guo, Xuepeng Fan, Rishan Chen, Jiaxing Zhang, Hucheng Zhou, Sean McDirmid, Chang Liu, Wei Lin, Jingren Zhou, and Lidong Zhou. Spotting code optimizations in data-parallel pipelines through periscope. In *Presented as part of the 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI 12)*, pages 121–133, 2012.

[GKMP08] Rodolphe Giroudeau, Jean-Claude König, Farida Kamila Moulai, and Jérôme Palaysi. Complexity and approximation for precedence constrained scheduling problems with large communication delays. *Theor. Comput. Sci.*, 401(1-3):107–119, 2008.

[GLLK79] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: a survey. *Ann. Discrete Math.*, 4:287–326, 1979.

[GPSS01] Leslie Ann Goldberg, Mike Paterson, Aravind Srinivasan, and Elizabeth Sweedyk. Better approximation guarantees for job-shop scheduling. *SIAM J. Discret. Math.*, 14(1):67–92, 2001.

[Gra66] R. L. Graham. Bounds for certain multiprocessing anomalies. *Bell System Technical Journal*, 45(9):1563–1581, 1966.

[HCG12] Chi-Yao Hong, Matthew Caesar, and P Brighten Godfrey. Finishing flows quickly with preemptive scheduling. *ACM SIGCOMM Computer Communication Review*, 42(4):127–138, 2012.

[HLV94] J. A. Hoogeveen, Jan Karel Lenstra, and Bart Veltman. Three, four, five, six, or the complexity of scheduling with communication delays. *Oper. Res. Lett.*, 16(3):129–137, 1994.

[HM01] C. Hanen and A. Munier. An approximation algorithm for scheduling dependent tasks on m processors with small communication delays. *Discrete Applied Mathematics*, 108(3):239 – 257, 2001.

[JZA19] Zhihao Jia, Matei Zaharia, and Alex Aiken. Beyond data and model parallelism for deep neural networks. In *Proceedings of the 2nd SysML Conference, SysML '19*, Palo Alto, CA, USA, 2019.

[KLTY20] Janardhan Kulkarni, Shi Li, Jakub Tarnawski, and Minwei Ye. Hierarchy-based algorithms for minimizing makespan under precedence and communication constraints. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 2770–2789. SIAM, 2020.

[Li21] Shi Li. Towards ptas for precedence constrained scheduling via combinatorial algorithms. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2991–3010. SIAM, 2021.

[LLKS93] Eugene L Lawler, Jan Karel Lenstra, Alexander HG Rinnooy Kan, and David B Shmoys. Sequencing and scheduling: Algorithms and complexity. *Handbooks in operations research and management science*, 4:445–522, 1993.

[LMR94] Frank Thomson Leighton, Bruce M. Maggs, and Satish Rao. Packet routing and job-shop scheduling in O(congestion + dilation) steps. *Comb.*, 14(2):167–186, 1994.

[LR16] E. Levey and T. Rothvoss. A (1+epsilon)-approximation for makespan scheduling with precedence constraints using LP hierarchies. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 168–177, 2016.

[LYZ+16] Shouxi Luo, Hongfang Yu, Yangming Zhao, Sheng Wang, Shui Yu, and Lemin Li. Towards practical and near-optimal coflow scheduling for data center networks. *IEEE Transactions on Parallel and Distributed Systems*, 27(11):3366–3380, 2016.

[MK97] A. Munier and J.C. König. A heuristic for a scheduling problem with communication delays. *Operations Research*, 45(1):145–147, 1997.

[MPL+17] Azalia Mirhoseini, Hieu Pham, Quoc V Le, Benoit Steiner, Rasmus Larsen, Yuefeng Zhou, Naveen Kumar, Mohammad Norouzi, Samy Bengio, and Jeff Dean. Device placement optimization with reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2430–2439. JMLR. org, 2017.

[MRS+20] Biswaroop Maiti, Rajmohan Rajaraman, David Stalfa, Zoya Svitkina, and Aravindan Vijayaraghavan. Scheduling precedence-constrained jobs on related machines with communication delay. *To appear at 61st Annual IEEE Symposium on Foundations of Computer Science (FOCS 2020)*, 2020.

[MS11] Monaldo Mastrolilli and Ola Svensson. Hardness of approximating flow and job shop scheduling problems. *J. ACM*, 58(5):20:1–20:32, 2011.

[NHP+19] D. Narayanan, A. Harlap, A. Phanishayee, V. Seshadri, N. Devanur, G. Ganger, P. Gibbons, and M. Zaharia. Pipedream: Generalized pipeline parallelism for dnn training. In *Proc. 27th ACM Symposium on Operating Systems Principles (SOSP)*, Huntsville, ON, Canada, October 2019.

[PY90] Christos H. Papadimitriou and Mihalis Yannakakis. Towards an architecture-independent analysis of parallel algorithms. *SIAM J. Comput.*, 19(2):322–328, 1990.

[RS87] V.J. Rayward-Smith. Uet scheduling with unit interprocessor communication delays. *Discrete Applied Mathematics*, 18(1):55 – 71, 1987.

[SSW94] David B. Shmoys, Clifford Stein, and Joel Wein. Improved approximation algorithms for shop scheduling problems. *SIAM J. Comput.*, 23(3):617–632, 1994.

[Sve10] Ola Svensson. Conditional hardness of precedence constrained scheduling on identical machines. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 745–754. ACM, 2010.

[SW99] P. Schuurman and G. J. Woeginger. Polynomial time approximation algorithms for machine scheduling: Ten open problems, 1999.

[SZA+18] Ayan Shymyrbay, Arshyn Zhanbolatov, Assilkhan Amankhan, Adilya Bakambekova, and Ikechi A Ukaegbu. Meeting deadlines in datacenter networks: An analysis on deadline-aware transport layer protocols. In *2018 International Conference on Computing and Network Communications (CoCoNet)*, pages 152–158. IEEE, 2018.

[TPD+20] Jakub Tarnawski, Amar Phanishayee, Nikhil R. Devanur, Divya Mahajan, and Fanny Nina Paravecino. Efficient algorithms for device placement of dnn graph operators, 2020.

[TY92] R. Thurimella and Y. Yesha. A scheduling principle for precedence graphs with communication delay. *International Conference on Parallel Processing*, 3:229–236, 1992.

[VLL90] B. Veltman, B.J. Lageweg, and J.K. Lenstra. Multiprocessor scheduling with communication delays. *Parallel Computing*, 16(2):173 – 182, 1990.

[WS11] David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms.* Cambridge University Press, USA, 2011.

[ZCB+15] Yangming Zhao, Kai Chen, Wei Bai, Minlan Yu, Chen Tian, Yanhui Geng, Yiming Zhang, Dan Li, and Sheng Wang. Rapier: Integrating routing and scheduling for coflow-aware data center networks. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pages 424–432. IEEE, 2015.

[ZZC+12] Jiaxing Zhang, Hucheng Zhou, Rishan Chen, Xuepeng Fan, Zhenyu Guo, Haoxiang Lin, Jack Y Li, Wei Lin, Jingren Zhou, and Lidong Zhou. Optimizing data shuffling in data-parallel computation by understanding user-defined functions. In *Presented as part of the 9th USENIX Symposium on Networked Systems Design and Implementation*

(NSDI 12), pages 295–308, 2012.