Better Unrelated Machine Scheduling for Weighted Completion Time via Random Offsets from Non-Uniform Distributions [Extended Abstract]

Sungjin Im *
Electrical Engineering and Computer Science
University of California
5200 N. Lake Road, Merced, CA 95344, USA
sim3@ucmerced.edu

Shi Li †
Department of Computer Science and Engineering
University at Buffalo
1 White Road, Buffalo, NY 14260, USA
shil@buffalo.edu

***Abstract*—In this paper we consider the classic scheduling problem of minimizing total weighted completion time on unrelated machines when jobs have release times, i.e, $R|r_{ij}|\sum_j w_jC_j$ using the three-field notation. For this problem, a 2-approximation is known based on a novel convex programming (J. ACM 2001 by Skutella). It has been a long standing open problem if one can improve upon this 2-approximation (Open Problem 8 in J. of Sched. 1999 by Schuurman and Woeginger). We answer this question in the affirmative by giving a 1.8786-approximation. We achieve this via a surprisingly simple linear programming, but a novel rounding algorithm and analysis. A key ingredient of our algorithm is the use of random offsets sampled from non-uniform distributions.**

We also consider the preemptive version of the problem, i.e, $R|r_{ij}, pmtn|\sum_j w_jC_j$. We again use the idea of sampling offsets from non-uniform distributions to give the first better than 2-approximation for this problem. This improvement also requires use of a configuration LP with variables for each job's complete schedules along with more careful analysis. For both non-preemptive and preemptive versions, we break the approximation barrier of 2 for the first time.

***Keywords*-scheduling; completion time; release times; approximation algorithms;**

I. Introduction

Modern computing facilities serve a large number of jobs with different characteristics. To cope with this challenge, they are equipped with increasingly heterogeneous machines that are clustered and connected in networks, so that each job can be scheduled on a more suitable machine. Further, the large number of machines of different generations are deployed over a long period of time, increasing the heterogeneity. The scheduling decision must factor in the heterogeneity and communication overhead.

Unrelated machine scheduling is a widely studied classic model that captures various scenarios including the above. There is a set J of jobs to be scheduled on a set M of unrelated machines. Each job $j \in J$ can have an arbitrary processing time/size $p_{i,j}$ depending on the machine i it gets processed; if $p_{i,j} = \infty$, then job j cannot be scheduled on machine i. Furthermore, due to the communication delay, job j is available for service only from time $r_{i,j}$, which can be also arbitrary depending on the job j and the machine i the job gets assigned to. The parameter $r_{i,j}$ is often called as job j's arrival/release time.[1] Another parameter w_j is used to capture job j's importance.

Minimizing total (weighted) completion time is one of the most popular scheduling objectives that has been extensively studied, even dating back to 50's [1]. The scheduler must assign each job j to a machine i and complete it. We consider two settings, preemptive and non-preemptive schedules. In the non-preemptive setting, each job must be completed without interruption once it starts getting processed. On the other hand, in the preemptive setting, each job's processing can be interrupted to process other jobs and be resumed later. In both cases, job j's completion time is, if j is assigned to machine i, defined as the first time when the job gets processed for $p_{i,j}$ units of time. Then, the objective is to minimize $\sum_{j\in J} w_jC_j$. These two non-preemptive and preemptive versions can be described as $R|r_j|\sum_j w_jC_j$ and $R|r_j, pmtn|\sum_j w_jC_j$ respectively, using the popular three-field notation in scheduling literature. Both versions of the problem are strongly NP-hard even in the single machine setting [2], and are APX-hard even when all jobs are available for schedule at time 0 [3], in which case preemption does not help.

For the non-preemptive case, Skutella gave a 2-approximation based on a novel convex programming [4], which improved upon the $(2+\epsilon)$-approximation based on linear programming [5]. It has been an outstanding open problem if there exists a better than 2-approximation [4], [6], [5], [7], [8]. In particular, it is listed in [6] as one of the top 10 opens problems in the field of approximate scheduling algorithms; see the Open Problem 8. When jobs

* Supported in part by NSF grants CCF-1409130 and CCF-1617653.
† Supported in part by NSF grant CCF-1566356.

[1]For simplicity, we will mostly assume that job j's release time $r_{i,j}$ is the same for all machines. This will justify using a simpler notation r_j in place of $r_{i,j}$. Like most of previous works, extending our result to release dates with dependency on machines is straightforward.

have no arrival times, i.e. $r_{i,j} = 0$ for all i, j, very recently Bansal et al. [9] gave a better than 1.5-approximation in a breakthrough result, improving upon the previous best 1.5-approximations due to Skutella [4] and Sethuraman and Squillante [10]. In fact, the Open Problem 8 consists of two parts depending on whether jobs have release times or not. Bansal et al. [9] solved the first part of Open Problem 8, and the second part still remained open.

A. Our Results

In this paper, we answer the second part of the open problem in the affirmative by giving a better than 2-approximation.

Theorem 1 (Section II)**.** *For a constant $\alpha < 1.8786$, there exists an α-approximation for $R|r_j|\sum_j w_j C_j$.*

Surprisingly, we give this result by rounding a very simple and natural LP that has not been studied in previous works. Our LP can be viewed as a stronger version of the time-indexed LP in [5], by taking the non-preemption requirement into consideration. However, even with this stronger LP, the rounding algorithm in [5] does not yield a better than 2-approximation (see the discussion about use of uniform distribution in Section II-A), and we believe this is why the previous works overlooked this simple LP. Improving the 2-approximation ratio requires not only the stronger LP, but also novel rounding algorithm and analysis.

Our result also gives a positive answer to the conjecture made by Sviridenko and Wiese [8]. They considered a configuration LP where there is a variable for every machine $i \in M$ and subset of jobs $S \subseteq J$. The variable is associated with the optimal total weighted completion time of the jobs in S on machine i. They showed that one can solve their LP within a factor of $1 + \epsilon$, but could not give a better than 2-approximation, conjecturing that their LP have an integrality gap strictly less than 2.

Indeed, one can show that the configuration LP of [8] is the strongest among all convex programmings of the following form: minimize $\sum_{i \in M} f_i(x_i)$ subject to $\sum_{i \in M} x_{i,j} = 1$ for every $j \in J$ and $x_{i,j} \geq 0$ for every $i \in M, j \in J$, where $x_i = (x_{i,j})_{j \in J} \in [0,1]^J$ and f_i is some convex function over $[0,1]^J$ such that if $x_i \in \{0,1\}^J$, then $f_i(x_i)$ is at most the total weighted completion time of scheduling jobs $\{j : x_{i,j} = 1\}$ optimally on machine i. All results mentioned in this paper (including our results) are based on programmings of this form and thus the configuration LP is the strongest among them. Hence, our result gives a 1.8786 upper bound on the integrality gap of the configuration LP.

With a solution to the configuration LP, one can derive a natural independent rounding algorithm. For each job j, independently assign j to a machine i with probability $x_{i,j}$. Then for every machine i, we schedule all jobs assigned to i; this can be done optimally if all release times are 0 [1], and nearly optimally (within $(1 + \epsilon)$ factor) in general [11], [12]. When all jobs have release time 0, the algorithm gives a 1.5-approximation. However, [9] showed this independent rounding algorithm can not give a better than 1.5-approximation, which motivated them to develop a clever dependent rounding algorithm.

For $R|r_j|\sum_j w_j C_j$, the independent rounding algorithm is known to give a 2-approximation [5], [4]. In contrast to the status for $R||\sum_j w_j C_j$, no matching lower bound was known for this algorithm. Our result indirectly shows that the independent rounding can achieve 1.8786-approximation. Thus we do not need to apply the sophisticated dependence rounding scheme of [9], which only led to a tiny improvement on the approximation ratio for $R||\sum_j w_j C_j$. We complement our positive result by showing that the independent rounding algorithm can not give an approximation ratio better than $e/(e-1) \approx 1.581$. Due to the space limitation, the proof is deferred to the full version of this paper.

Theorem 2. *There is an instance for which the independent rounding gives an approximation ratio worse than $e/(e-1) - \epsilon \geq 1.581 - \epsilon$ for any $\epsilon > 0$.*

We continue to study the preemptive case. In the preemptive case, two variants were considered in the literature depending on whether jobs can migrate across machines or must be completed scheduled on one of the machines. If migration is not allowed, the work in [5] still gives a $(2 + \epsilon)$-approximation since the LP therein is a relaxation for preemptive schedules but the rounding outputs a non-preemptive schedule. If migration is allowed, [4] gives a 3-approximation. Our main result for the preemptive case is the first better than 2-approximation when migration is not allowed.

Theorem 3 (Section III)**.** *For a constant $\alpha < 1.99971$, there exists an α-approximation for $R|r_j, pmtn|\sum_j w_j C_j$.*

We note that our algorithm is based on a stronger linear programming relaxation. The configuration LP of [8] is for non-preemptive schedules hence not usable for preemptive schedules. Our LP is a different type of configuration LP where there are variables for each job's complete schedules. While we use an LP for preemptive schedules, we output a non-preemptive schedule.

B. Our Techniques

As mentioned before, we give a better than 2-approximation for the non-preemptive case based on a very simple LP. In this LP, we have an indicator variable $y_{i,j,s}$ which is 1 if job j starts at time s on machine i. Then, we add an obvious constraint that no more than one job can be processed at any time on any machine. This LP has a pseudo-polynomial size but can be reduced to a polynomial size using standard techniques with a loss of $(1 + \epsilon)$ factor in approximation.

As mentioned earlier, our algorithm falls into the independent rounding framework: we assign each job j to machine i with probability $x_{i,j} = \sum_s y_{i,j,s}$ independently following the optimal LP solution. Then, it remains to schedule jobs assigned to each machine.[2] Any solution to our LP is also a solution to the LP in [5]. When restricted to a solution to our LP, the rounding algorithm of [5] works as follows. For every j that is assigned to i, we choose $s_j = s$ randomly with probability proportional to $y_{i,j,s}$. Then we choose τ_j uniformly at random from $[s_j, s_j + p_{i,j}]$; here $\tau_j - s_j$ can be viewed as a random extra offset applied to j. We schedule jobs assigned to i non-preemptively in increasing order of τ_j values. While this gives a 2-approximation, this is the best one can obtain using their LP since it has a matching integrality gap. Even with our stronger LP, the algorithm only gives a 2-approximation.

We use a more sophisticated distribution to sample τ_j for individual jobs. Discovering such a distribution and showing how it helps improve the approximation ratio requires a novel analysis. We are not the first that use non-uniform distributions for scheduling problems. Goemans et al. [13] used non-uniform distributions in their α-point rounding for the single machine scheduling, i.e. $1|r_j|\sum_j w_j C_j$ to give a 1.6853-approximation. However, their analysis does not lend itself to multiple machines. The LP objective considered in [13] uses the notion of fractional completion time, which views a job j of size p_j as consisting of p_j unit pieces with weight w_j/p_j. In this view, the optimal schedule trivially follows from the simple greedy Smith rule. [13] heavily uses this special structure to get a better than 2 approximation. However, this relaxation inherently loses a factor 2 when applied to multiple machines even with some correction terms [5], [4]. Hence to overcome the 2-approximation barrier, one has to deviate from this relaxation and the special structure used in [13], which calls for use of a stronger LP along with new algorithms and/or analysis. Intuitions on the effect of non-uniform distributions can be found in Section II, particularly in discussion of the limitations of uniform distributions.

As mentioned before, the preemptive result requires an even stronger LP where there is a variable for each job's complete schedule. Since preemption is allowed, even when all parameters are polynomially bounded, the LP has exponentially many variables. We solve this LP by solving its dual with help of a separation oracle. While the algorithm for the non-preemptive case naturally extends to the preemptive case, the analysis doesn't. At a high level, the analysis for both cases needs to carefully handle the interaction between busy times and idle times which both can contribute jobs delays. Non-preemptive schedules possess better structural properties which allow us to break down the analysis into that for each time step. However, preemptive schedules lack such properties and require a different analysis of a somewhat amortized flavor.

[2]Since $1|r_j|\sum_j w_j C_j$ admits a PTAS, given the set of jobs assigned to i, one can find a $(1+\epsilon)$-approximately optimal schedule on i. However, it is hard to directly relate this schedule to the fractional solution.

C. Other Related Work

The first non-trivial $O(\log^2 n)$-approximation for $R|r_{ij}|\sum_j w_j C_j$ was given by Stein et al. [14] using a hypergraph matching. Then, subsequent works [12], [5], [4] gave constant approximations, culminating in a 2-approximation [4] which was the best known prior to our work. The work in [12] uses the celebrated rounding for the generalized assignment problem [15] to round an LP with intervals of doubling lengths, thereby giving a 16/3-approximation. As mentioned before, [5] gives a $(2+\epsilon)$-approximation, and there is an easy instance of matching integrality gap for their LP. Subsequently, Skutella gave a 2-approximation using a convex programming [4], which is tight since the CP has an integrality gap of 2. When machines are identical, uniformly related, or a special case of unrelated machines, PTASes are known [11], [16], [17].

Minimizing makespan or equivalently the maximum completion time is a closely related objective. For this problem when all jobs arrive at time 0, Lensta et al. gave a 2-approximation and showed it does not admit a better than 1.5 approximation unless P = NP [18]. Reducing this gap remains open. Svensson showed that one can estimate the optimal makespan within a factor of $33/17+\epsilon$ for the special case of restricted assignment [19]. For the dual objective of maximizing the minimum load on any machine, see [20], [21], [22]. For the minimizing ℓ_p norms of completion times, see [23], [24].

For the objective of minimizing total flow time, i.e. $\sum_j (C_j - r_j)$, a poly-logarithmic approximation is known [25]. For earlier works for the restricted assignment case, see [26], [27]. Due to the vast literature on scheduling, our discussion on related work is necessarily incomplete. For a nice survey and more pointers, see [28].

II. Non-Preemptive Scheduling

We begin by giving an LP for the non-preemptive case. To present our algorithm and analysis more transparently, we assume that all parameters are polynomially bounded, i.e. all $w_j, r_j, p_{i,j}$ are $\texttt{poly}(|J|, |M|)$. Although we can also handle the case when $p_{ij} = \infty$ by not allowing j to be scheduled on machine i, we assume such a case does not happen since the extension is straightforward. Due to the space constraints, in this paper we omit how one can remove these simplifying assumptions.

Define $T := \sum_{i,j} p_{i,j} + \max_j r_j$ so that any 'reasonable' scheduler can complete all jobs by the time T. Throughout this section, s is always an integer.

$$\min \sum_{i\in M,\ j\in J,\ s\in[0,T)} w_j y_{i,j,s}(s+p_{i,j}) \qquad (\mathsf{LP}_{\mathsf{interval}})$$

$$\text{s.t.} \sum_{s\in[0,T)} y_{i,j,s} = x_{i,j}, \quad \forall i\in M, j\in J \qquad (1)$$

$$\sum_{i\in M} x_{i,j} = 1, \quad \forall j\in J \qquad (2)$$

$$\sum_{j\in J,\ s\in[\max\{0,t-p_{i,j}\},t)} y_{i,j,s} \le 1, \quad \forall i\in M, t\in[T] \qquad (3)$$

$$y_{i,j,s} \ge 0, \quad \forall i\in M, j\in J, s\in[0,T)$$

$$y_{i,j,s} = 0, \quad \forall i\in M, j\in J, s < r_j \text{ or } s > T - p_{i,j}$$

To see this is a valid LP relaxation for non-preemptive schedules, assume that all variables can only take integer values. Then, the first two constraints require that each job j must be assigned to exactly one machine, which is captured by the indicator variable $x_{i,j}$. The variable $y_{i,j,s} = 1$ if and only if j starts getting processed at time s on machine i. The constraints (3) ensure that only one job gets processed at a time on any machine. The last constraint prohibits jobs from getting processed before their arrival times. We obtain a valid LP relaxation by allowing variables to have fractional values.

Rounding. We now describe how to round the LP, which consists of two steps. The first step is to define a 'pseudo' arrival time $\tau_j \ge r_j$ for each job j. For each job j, we can view $\{y_{i,j,s}\}_{i,s}$ as a probability distribution over pairs (i,s) due to Constraint (1), and choose a pair (i_j, s_j) according to the distribution randomly and independently. Job j will be scheduled on machine i_j. Let $\mathbf{\Theta}$ be some distribution over real numbers in $[0,1]$ where no number in the distribution occurs with positive probability; $\mathbf{\Theta}$ will be fixed later. We randomly and independently choose a number θ_j from $\mathbf{\Theta}$. Define $\tau_j = s_j + \theta_j \cdot p_{i_j,j}$. We assume w.l.o.g. that all jobs have different τ_j values since this event happens almost surely.

In the second step, we finalize each machine's schedule. For each $i \in M$, let $J_i = \{j \in J : i_j = i\}$ be the set of jobs that are assigned to i. Let π be the ordering of J_i according to increasing order of τ_j values. We schedule jobs in J_i on machine i according to π, pretending that τ_j is job j's actual arrival time. That is, job $j \in J_i$ starts when all jobs in J_i ahead of j in the ordering of π complete, or at time τ_j, whichever comes later.

Notice that if we use the actual arrival times r_j instead of the pseudo ones for scheduling, we can obtain the optimum schedule on i respecting the ordering π – that is, each job $j \in J_i$ starts when all jobs in J_i before j according to π complete, or at time r_j, whichever comes later. The schedule given by our algorithm might be worse than this optimum schedule respecting π. However, for the sake of analysis, it is more convenient to use our schedule, rather than the optimum one. Our schedule on machine i might have fractional starting times, but it is not an issue since we can convert the schedule to the optimum one respecting π, in which all starting times are integral.

A. Analysis

It will be convenient to think of the LP solution as a set $\mathcal{R}_i$ of rectangles for each machine i. For each pair of j and s with $y_{i,j,s} > 0$, we have a rectangle $R_{i,j,s}$ of length $p_{i,j}$ and height $y_{i,j,s}$ in $\mathcal{R}_i$. Horizontally, the rectangle $R_{i,j,s}$ covers the time interval $(s, s+p_{i,j}]$. For any machine i, the total height of rectangles in $\mathcal{R}_i$ covering any time point $t \in (0,T]$ is at most 1.

We will analyze the expected completion time of each job j and upper bound it by the corresponding LP quantity, $\sum_{i,s} y_{i,j,s}(s+p_{i,j})$. Towards this end, henceforth we fix a job $j \in J$, the machine $i \in M$ job j is assigned to, and a value of $\tau \in (0,T]$ job j is given. We consider $\mathbb{E}[C_j | i_j = i, \tau_j = \tau]$, i.e, the expected completion time of j, conditioned on the event that $i_j = i$ and $\tau_j = \tau$. For notational convenience, we use $\widehat{\mathbb{E}}[\cdot]$ to denote $\mathbb{E}[\cdot | i_j = i, \tau_j = \tau]$ and $\widehat{\Pr}[\cdot]$ to denote $\Pr[\cdot | i_j = i, \tau_j = \tau]$. After bounding $\widehat{\mathbb{E}}[C_j]$ by τ and $p_{i,j}$, we will get the desired bound on $\mathbb{E}[C_j]$ by deconditioning.

The key issue we have to handle when jobs have arrival times is that there can be idle times before job j starts. Hence we have to consider not only the volume of jobs scheduled before job j, but also the total length of idle times.

Definition 4. *For a time point $t \in (0,T]$, we say that t is* idle*, if there are no jobs scheduled at time t on machine i in our schedule. Let* $\mathsf{idle}(t)$ *indicate whether the time point t is idle or not.*

With this definition, we are ready to formally break down C_j into several quantities of different characteristics.

$$C_j = \sum_{j'\in J_i : \tau_{j'} < \tau_j} p_{i,j'} + \int_0^{\tau_j} \mathsf{idle}(t)\mathsf{d}t + p_{i,j}. \qquad (4)$$

The first term is the total length of jobs scheduled before j on machine i and the second is the total length of idle times before τ_j. Notice that there are no idle points in $[\tau_j, C_j)$ since all jobs $j' \in J_i$ scheduled before j have $\tau_{j'} < \tau_j$.

Uniform Distribution and its Limitations. Before we present a better than 2-approximation, we take a short detour to discuss how we recover a simple 2-approximation by setting $\mathbf{\Theta}$ to be the uniform distribution over $[0,1]$. To compute $\widehat{\mathbb{E}}[C_j]$, we first consider $\widehat{\mathbb{E}}\big[\sum_{j'\in J_i, \tau_{j'}<\tau} p_{i,j'}\big]$. If some $j' \in J_i$ has $\tau_{j'} < \tau = \tau_j$, we say that the pair $(j', s_{j'})$ contributed $p_{i,j'}$ to the sum. For each $j' \ne j$ and integer $s < \tau$, the expected contribution of the pair (j', s) to the sum is $\widehat{\Pr}\big[i_{j'} = i, s_{j'} = s\big]\widehat{\Pr}\big[\tau_{j'} < \tau | i_{j'} = i, s_{j'} = s\big] p_{i,j'} =$

$y_{i,j',s} \cdot \min\{1, (\tau - s)/p_{i,j'}\} \cdot p_{i,j'} = y_{i,j',s} \cdot \min\{p_{i,j'}, \tau - s\}$. This is exactly the area of the portion of the rectangle $R_{i,j',s}$ before time point τ. Summing up over all pairs $(j' \neq j, s < \tau)$, $\widehat{\mathbb{E}}\big[\sum_{j' \in J_i, \tau_{j'} < \tau} p_{i,j'}\big]$ is at most the total area of the portions of $\mathcal{R}_i$ before τ, which is at most τ. The total length of idle slots before τ is obviously at most τ. Thus, $\widehat{\mathbb{E}}[C_j] \leq 2\tau + p_{i,j}$. Since $\mathbb{E}[\tau_j | i_j = i, s_j = s] = s + p_{i,j}/2$, we have $\mathbb{E}[C_j | i_j = i, s_j = s] \leq 2(s + p_{i,j}/2) + p_{i,j} = 2(s + p_{i,j})$. Since $\Pr[i_j = i, s_j = s] = y_{i,j,s}$, we have that $\mathbb{E}[C_j] \leq 2\sum_{i,s} y_{i,j,s}(s + p_{i,j})$, which is exactly twice the unweighted contribution of j to the $\mathsf{LP}_{\mathsf{interval}}$ objective. Thus, we obtain a 2-approximation for the problem.

However, uniform distribution does not yield a better than 2-approximation. To see this, consider the following instance and LP solution. There are $1/\epsilon + 1$ machines indexed by $1, 2, ..., 1/\epsilon + 1$. There is one unit-sized job j^* with arrival time r and it is scheduled on each of machines $1, 2, ..., 1/\epsilon$ by ϵ fraction during $(r, r+1]$; j^* is not allowed to be scheduled on machine $1/\epsilon + 1$. There are $1/\epsilon$ big jobs of sizes $p \gg r$ with arrival time 0, which are indexed by $j_1, j_2, ..., j_{1/\epsilon}$. Each big job j_k can be assigned to either machine k or machine $1/\epsilon + 1$. The job j_k starts on machine k at time 0 by $1 - \epsilon$ fraction, and on machine $1/\epsilon + 1$ by ϵ fraction. For simplicity, say the unit-sized job has a unit weight and the big jobs have zero (or infinitesimally small) weights so that the objective is essentially dominated by the unit sized job j^*'s completion time. Clearly, j^* has completion time $r + 1$ in the LP solution.

We now show that the above rounding makes j^*'s completion time arbitrarily close to $2r$ in expectation. Fix the machine j^* is assigned to by the above algorithm; w.l.o.g. assume that the machine is 1. With $1 - \epsilon$ probability, job j_1 is assigned to machine 1; under this event, j_1 has a smaller τ value than j^* with probability r/p. Hence j^* starts at time p with probability $(1 - \epsilon)r/p$, otherwise at time r, meaning that j^*'s expected starting time is at least $(1 - \epsilon)(r/p) \times p + (1 - (1 - \epsilon)r/p) \times r$ which tends to $2r$ as $\epsilon \to 0$ and $p \to \infty$. This shows one cannot get a better than 2-approximation using uniform distribution.

Finding a Better Distribution. The above example is simple yet illuminating. We first observe that pushing back the small job a lot due to big job might be a sub-optimal choice. Intuitively, a bigger job is less sensitive to delay since the delay can be charged to the job's processing time. We could try to shift mass in the distribution $\mathbf{\Theta}$ to the right. Then, big jobs will be less likely to have smaller τ values than the small job. However, this could increase τ_j values in expectation, thereby increasing the objective. We would like to avoid increasing the offset added to τ_j which was $p_{i,j}/2$ (assuming that job j goes to machine i). To satisfy both requirements, we shall shift the mass from both ends to the middle. In the above example, the job j^* overlaps the left-end of the big job j_1. Shifting the mass from the left to the middle will decrease the probability that $\tau_{j_1} < \tau_{j^*}$. On the other hand, shifting the mass from the right to the middle will decrease the expectation of τ_{j^*}.

The remainder of this section is devoted to studying the effect of using different distributions on the approximation ratio. Let $f : [0, 1] \to \mathbb{R}_{\geq 0}$ be the probability density function (PDF) of $\mathbf{\Theta}$ and $F(t) = \int_0^t f(t')\mathsf{d}t'$ be the cumulative distribution function (CDF) of $\mathbf{\Theta}$. Recall that we fixed a job $j \in J$, the machine $i \in M$ job j is assigned to, and a value of $\tau \in [0, T)$ job j is given. For every $j' \in J \setminus j, t \in (0, T]$ and integer s, we shall use $s \lhd_{j'} t$ to indicate that $s \in [\max\{0, t - p_{i,j'}\}, t)$. In other words, $s \lhd_{j'} t$ means that if j' starts at s, then it must get processed at time t. For every $t \in (0, T]$, define

$$g(t) = \sum_{j' \in J \setminus j,\ s \lhd_{j'} t} y_{i,j',s} \cdot f\left(\frac{t - s}{p_{i,j'}}\right)$$

$$\text{and} \qquad h(t) = \sum_{j' \in J \setminus j,\ s \lhd_{j'} t} y_{i,j',s} \cdot F\left(\frac{t - s}{p_{i,j'}}\right).$$

It is worth mentioning that $\sum_{s \lhd_{j'} t} y_{i,j',s} \cdot f\left(\frac{t-s}{p_{i,j'}}\right) \frac{1}{p_{i,j'}}$ is the density of the probability that $\tau_{j'} = t$. Thus, integrating $g(t)$ from time 0 to $\tau = \tau_j$ will give the expected volume of work done before job j, which is the first term of (4) in expectation. The usefulness of $h(t)$ will be discussed shortly.

Lemma 5. $\widehat{\mathbb{E}}\left[\sum_{j' \in J_i : \tau_{j'} < \tau} p_{i,j'}\right] = \int_0^\tau g(t)\mathsf{d}t.$

Proof:
$$\text{LHS} = \sum_{j' \in J \setminus j} p_{i,j'} \cdot \widehat{\Pr}[i_{j'} = i, \tau_{j'} < \tau]$$
$$= \sum_{j' \in J \setminus j} p_{i,j'} \sum_{s \in [0,\tau)} y_{i,j',s} \cdot \widehat{\Pr}\big[\tau_{j'} < \tau | i_{j'} = i, s_{j'} = s\big]$$
$$= \sum_{j' \in J \setminus j} p_{i,j'} \sum_{s \in [0,\tau)} y_{i,j',s} \cdot \int_0^{\min\{(\tau - s)/p_{i,j'}, 1\}} f(\theta)\mathsf{d}\theta$$
$$= \sum_{j' \in J \setminus j} p_{i,j'} \sum_{s \in [0,\tau)} \frac{y_{i,j',s}}{p_{i,j'}} \int_s^{\min\{\tau, s + p_{i,j'}\}} f\left(\frac{t - s}{p_{i,j'}}\right)\mathsf{d}t$$
$$= \int_{t=0}^{\tau} \sum_{j' \in J \setminus j} \sum_{s \lhd_{j'} t} y_{i,j',s} \cdot f\left(\frac{t - s}{p_{i,j'}}\right)\mathsf{d}t$$
$$= \int_0^\tau g(t)\mathsf{d}t. \qquad \blacksquare$$

We now shift our attention to bounding the second term in (4) using the function $h(t)$. As we observed when using uniform distributions, the obvious upper bound on the second term is τ. To improve upon this, we need to show a considerable fraction of times are not idle. We note that $\sum_{s \lhd_{j'} t} y_{i,j',s} \cdot F\left(\frac{t-s}{p_{i,j'}}\right)$ is the probability that job j' is processed at time t when starting at $\tau_{j'}$. If such an event

occurs, then time t will be shown to be non-idle, hence we get some credits.

Claim 6. *$h(t) \le 1$ for every $t \in [0,T)$.*

Proof: Since F is a CDF, we have $F(t') \le 1$ for every $t' \in [0,1]$. Thus, $h(t) \le \sum_{j' \in J \setminus j,\ s \lhd_{j'} t} y_{i,j',s} \le 1$ by Constraint (3). ■

Lemma 7. *For every $t \in (0,\tau]$, we have $\widehat{E}[\mathsf{idle}(t)] \le e^{-h(t)}$.*

Proof: We say t' is *empty* if there are no jobs $j' \in J_i$ such that $t' \in (\tau_{j'}, \tau_{j'} + p_{i,j'}]$; let $\mathsf{empty}(t')$ denote the indicator variable that is 1 iff t' is empty. We first observe that if some $t' \in (0,T]$ is not empty, then t' is not idle. This is because a job j' such that $t' \in (\tau_{j'}, \tau_{j'} + p_{i,j'}]$ is not processed at time t' only when other jobs are. Thus,

$$\begin{aligned}
&\widehat{\mathbb{E}}[\mathsf{idle}(t)] \le \widehat{\mathbb{E}}[\mathsf{empty}(t)] \\
&= \prod_{j' \in J \setminus j} \left(1 - \widehat{\Pr}[i_{j'} = i, t \in (\tau_{j'}, \tau_{j'} + p_{i,j'}]]\right) \\
&\le \exp\Big(-\sum_{j' \in J \setminus j} \widehat{\Pr}[i_{j'} = i, t \in (\tau_{j'}, \tau_{j'} + p_{i,j'}]]\Big) \\
&\le \exp\Big(-\sum_{j' \in J \setminus j} \widehat{\Pr}\left[i_{j'} = i, t \in (s_{j'}, s_{j'} + p_{i,j'}], \theta_{j'} < \frac{t - s_{j'}}{p_{i,j'}}\right]\Big) \\
&= \exp\Big(-\sum_{j' \in J \setminus j} \sum_{s \lhd_{j'} t} y_{i,j',s} \cdot F\left(\frac{t-s}{p_{i,j'}}\right)\Big) \\
&= e^{-h(t)}.
\end{aligned}$$

■

Lemma 8. $\displaystyle\int_0^\tau \widehat{\mathbb{E}}[\mathsf{idle}(t)]\mathsf{d}t \le \tau - \left(1 - \frac{1}{e}\right)\int_0^\tau h(t)\mathsf{d}t.$

Proof: By Lemma 7, we have $\int_0^\tau \widehat{\mathbb{E}}[\mathsf{idle}(t)]\mathsf{d}t \le \int_0^\tau e^{-h(t)}\mathsf{d}t$. Notice that $h(t) \in [0,1]$ for every $t \in [0,1]$ by Claim 6. Thus by the convexity of the function e^{-x}, we have that $e^{-h(t)} \le (1-h(t))e^0 + h(t)e^{-1} = 1 - (1-1/e)h(t)$. Taking the integral from $t = 0$ to τ gives the lemma. ■

Lemma 9. *Let $\rho = \sup_{\phi \in (0,1]} \frac{1}{\phi}\left(F(\phi) - \left(1 - \frac{1}{e}\right)\int_0^\phi F(\theta)\mathsf{d}\theta\right)$, $\beta = \int_0^1 f(\theta)\theta\mathsf{d}\theta$ and $\alpha = 1 + \max\{\rho, (1+\rho)\beta\}$. Then our algorithm is an α-approximation algorithm.*

To prove Lemma 9, we first upper bound $\widehat{\mathbb{E}}[C_j]$ in terms of $\tau = \tau_j$ and $p_{i,j}$, then obtain an upper bound on $\mathbb{E}[C_j]$ by deconditioning.

Lemma 10. $\widehat{\mathbb{E}}[C_j] \le (1+\rho)\tau + p_{i,j}$.

Proof: By applying the bounds in Lemmas 5 and 8 to Eq. (4), we have

$$\begin{aligned}
&\widehat{\mathbb{E}}[C_j] - \tau - p_{i,j} \le \int_0^\tau g(t)\mathsf{d}t - \left(1 - \frac{1}{e}\right)\int_0^\tau h(t)\mathsf{d}t \\
&= \int_0^\tau \sum_{j' \ne j, s \lhd_{j'} t} y_{i,j',s}\left(f\left(\frac{t-s}{p_{i,j'}}\right) - \left(1 - \frac{1}{e}\right)F\left(\frac{t-s}{p_{i,j'}}\right)\right)\mathsf{d}t \\
&= \sum_{j' \ne j, s \in [0,\tau)} y_{i,j',s} \\
&\quad \cdot \int_s^{\min\{\tau, s+p_{i,j'}\}} \left(f\left(\frac{t-s}{p_{i,j'}}\right) - \left(1 - \frac{1}{e}\right)F\left(\frac{t-s}{p_{i,j'}}\right)\right)\mathsf{d}t \\
&= \sum_{j' \ne j, s \in [0,\tau)} y_{i,j',s} \cdot p_{i,j'} \\
&\quad \cdot \int_0^{\min\{(\tau-s)/p_{i,j'}, 1\}} \left(f(\theta) - \left(1 - \frac{1}{e}\right)F(\theta)\right)\mathsf{d}\theta.
\end{aligned}$$

By the definition of ρ and that $\int_0^\phi f(\theta)\mathsf{d}\theta = F(\phi)$ for $\phi \in [0,1]$, we have

$$\begin{aligned}
\widehat{\mathbb{E}}[C_j] &\le \sum_{j' \ne j, s \in [0,\tau)} y_{i,j',s} \cdot p_{i,j'} \cdot \rho \cdot \min\{(\tau - s)/p_{i,j}, 1\} + \tau + p_{i,j} \\
&= \rho \sum_{j' \ne j, s \in [0,\tau)} y_{i,j',s} \cdot \min\{\tau - s, p_{i,j'}\} + \tau + p_{i,j} \\
&\le \rho\tau + \tau + p_{i,j} = (1+\rho)\tau + p_{i,j},
\end{aligned}$$

where the last inequality holds because the sum is the total area of the portions of rectangles in $\mathcal{R}_i$ before time τ. ■

Lemma 11. $\mathbb{E}[C_j] \le \alpha \sum_{i \in M, s \in [0,T)} y_{i,j,s}(s + p_{i,j})$.

Proof: Now, we consider all machines $i \in M$. Then $\mathbb{E}[C_j]$ equals to

$$\begin{aligned}
&\sum_{i \in M, s \in [0,T)} \Pr[i_j = i, s_j = s] \\
&\qquad \cdot \int_0^1 f(\theta)\,\mathbb{E}[C_j | i_j = i, s_j = s, \tau_j = s + \theta p_{i,j}]\mathsf{d}\theta \\
&\le \sum_{i \in M, s \in [0,T)} y_{i,j,s} \int_0^1 f(\theta)\left((1+\rho)(s + \theta p_{i,j}) + p_{i,j}\right)\mathsf{d}\theta \\
&= \sum_{i \in M, s \in [0,T)} y_{i,j,s}\Big(\int_0^1 f(\theta)((1+\rho)s + p_{i,j})\mathsf{d}\theta + \int_0^1 f(\theta)(1+\rho)\theta p_{i,j}\mathsf{d}\theta\Big) \\
&\le \sum_{i \in M, s \in [0,T)} y_{i,j,s} \max\{1+\rho, 1 + (1+\rho)\beta\}(s + p_{i,j}) \\
&= \alpha \sum_{i \in M, s \in [0,T)} y_{i,j,s}(s + p_{i,j}).
\end{aligned}$$

■

We are now ready to complete the proof of Lemma 9. Summing up $\mathbb{E}[w_j C_j]$ over all jobs $j \in J$, we have

$$\mathbb{E}\Big[\sum_{j \in J} w_j C_j\Big] \le \alpha \sum_{i \in M, j \in J, s \in [0,T)} w_j y_{i,j,s}(s + p_{i,j}).$$

Notice that the right-hand-side is exactly α times the cost of the LP solution. Thus, our algorithm is an α-approximation.

To complete the proof of Theorem 1, we only need to find a distribution $\mathbf{\Theta}$ whose α value is no greater than the approximation ratio claimed in the theorem. We note that we first used a factor revealing LP to find out the best distribution that minimizes α. Then we discovered a truncated quadratic function is the best fit for the obtained discretized PDF. To find the best coefficients, we ran another program and obtained a distribution that yields a slightly better approximation ratio than one we could using the factor revealing LP. We set the PDF f as follows:

$$f(\theta) = \begin{cases} 0.1702\theta^2 + 0.5768\theta + 0.8746 & 0 \le \theta \le 0.85897 \\ 0 & \text{otherwise} \end{cases}.$$

Notice that $f(\theta)$ increases as θ goes from 0 to 0.85897 and becomes 0 when $\theta > 0.85897$. This is consistent with the previous discussion that we shift the probability mass from both ends to the middle. Then, by easy calculation one can show that $\beta < 0.46767$ and $\rho < 0.8785$. Thus $\alpha = 1 + \max\{\rho, (1+\rho)\beta\} < 1.8786$.

III. Preemptive Scheduling

This section is devoted to proving Theorem 3, which claims a better than 2-approximation for the preemptive case. Note that migration is not allowed, i.e. each job must be processed on only one of the machines. In the preemptive setting, a job's processing may be interrupted, so we need to choose $p_{i,j}$ unit-length time slots on machine i to schedule job j on machine i. This motivates the following definition.

Definition 12 (Chains)**.** *A chain A for job $j \in J$ on machine $i \in M$ is a sequence $(t_1, t_2, \cdots, t_{p_{i,j}})$ of integers such that $r_j < t_1 < t_2 < \cdots < t_{p_{i,j}} \le T$. Equivalently, we may view A as the set $\{t_1, t_2, \cdots, t_{p_{i,j}}\}$, or as a function from $(0, p_{i,j}]$ to $(0, T]$ such that $A(\vartheta) = t_{\lceil\vartheta\rceil} + \vartheta - \lceil\vartheta\rceil$ for all $\vartheta \in (0, p_{i,j}]$. For all $t \in (0, T]$, let $A^{-1}(t) = \sup\{\vartheta \in (0, p_{i,j}] : A(\vartheta) \le t\}$.*

A chain $A = (t_1, t_2, \cdots, t_{p_{i,j}})$ completely describes j's schedule on machine i: we schedule j on slots $(t_1 - 1, t_1], (t_2 - 1, t_2], \cdots, (t_{p_{i,j}} - 1, t_{p_{i,j}}]$. Thus, $A(\vartheta)$ is the time at which we have run j for ϑ units of time. In particular, $A(p_{i,j})$ is the completion time of j. We may use $C_A := A(p_{i,j})$ to denote j's completion time under the schedule A of job j. Notice that $A^{-1}(t)$ is the amount of time in which j is processed before t in A. Let $\mathcal{A}^{i,j}$ denote the set of all chains for job j on machine i.

Linear Programming. We are now ready to present our LP using the notion of chains. For notational convenience, when we refer to a chain A, we assume it is associated with a machine i and a job j satisfying $A \in \mathcal{A}^{i,j}$.

$$\min \sum_{i \in M, j \in J} \sum_{A \in \mathcal{A}^{i,j}} w_j C_A \cdot z_A \qquad (\mathsf{LP}_{\mathsf{chain}})$$

$$\text{s.t} \sum_{i \in M} \sum_{A \in \mathcal{A}^{i,j}} z_A \ge 1 \qquad \forall j \in J \tag{5}$$

$$\sum_{j \in J} \sum_{A \in \mathcal{A}^{i,j} : t \in A} z_A \le 1 \qquad \forall i \in M, t \in [T] \tag{6}$$

$$z_A \ge 0 \qquad \forall i \in M, j \in J, A \in \mathcal{A}^{i,j}$$

To see $\mathsf{LP}_{\mathsf{chain}}$ is a valid relaxation, assume that variables can only take integer values. In $\mathsf{LP}_{\mathsf{chain}}$ we have an indicator variable z_A for every possible chain $A \in \mathcal{A}^{i,j}$ for all i and j, which is 1 if and only if j is scheduled following the chain description A. The first constraint requires that every job must complete; note that we do not need equality here since the optimal solution will satisfy equality. It is also worth mentioning that job j never gets processed before its arrival time r_j since j's chains don't allow it. Finally, the second constraint ensures that every machine is used by at most one job at any point in time – there is at most one chain that schedules a job at any time. Thus we get a valid LP relaxation by allowing variables to have fractional values.

Although the LP has exponentially many variables, we can solve it using standard techniques – we solve the dual using the Ellipsoid method with a separation oracle. Due to the space constraints, we omit the details.

Algorithm. Our rounding is a natural generalization of the rounding for non-preemptive scheduling. To see this, suppose that a chain $A \in \mathcal{A}^{i,j}$ is a sequence of $p_{i,j}$ consecutive integers. Then A corresponds to an interval. If every chain in the support of z corresponds to an interval, then the fractional solution is a valid solution to $\mathsf{LP}_{\mathsf{interval}}$ for non-preemptive scheduling. In this scenario, our rounding works exactly in the same way as that for non-preemptive scheduling. Thus, we can generalize the former rounding by generalizing intervals to chains.

More specifically, our rounding algorithm works as follows. Let $\mathbf{\Theta}$ be some distribution over $[0, 1]$. For every $j \in J$, we randomly and independently choose a pair (i_j, A_j) such that $\Pr[(i_j, A_j) = (i, A)] = z_A$ for every $i \in M, A \in \mathcal{A}^{i,j}$. As $\sum_{i \in M, A \in \mathcal{A}^{i,j}} z_A = 1$ for every j, the random procedure is well-defined. For each j, we randomly and independently choose a number θ_j from $\mathbf{\Theta}$. Let $\tau_j = A_j(\theta_j \cdot p_{i_j,j})$. We assume that all jobs have different τ_j values since the event happens almost surely. As in the algorithm for the non-preemptive scheduling, we let $J_i = \{j \in J : i_j = i\}$ and schedule all jobs in J_i on machine i in increasing order of τ_j. We schedule the jobs as early as possible, maintaining the property that job j starts no earlier than τ_j. Notice that the schedule our algorithm constructed is non-preemptive, even though the problem allows preemption.

Overview of the Analysis. The analysis is more involved than the one for the non-preemptive case. To see this, let's recall how we gave a better than 2-approximation for the non-preemptive case. We can still break down a job's completion time as in Eq. (4) where job j's completion time is decomposed into three quantities: total volume of jobs with smaller τ values, total length of idle times before τ_j, and the size of job j itself. As we observed, if we use a uniform distribution for $\mathbf{\Theta}$, it is easy to get a 2-approximation by showing that both quantities are bounded by τ_j, which is j's starting time plus half of its size in expectation. Then, by using a non-uniform distribution $\mathbf{\Theta}$ with more mass around the center, we could have the following benefits: (i) if a job $j' \neq j$ is processed a little before τ_j, it is less likely to have a smaller τ value; and (ii) otherwise, a considerable fraction of job j' is processed before τ_j, thus contributes to reducing the number of idle times. Then, using the non-preemptive structure of the schedule, we were able to analyze each time's contribution to the first and second

quantities in Eq. (4).

While the high-level idea is the same, we have to take a different analysis route for the preemptive case since each job's schedule is scattered over time, which keeps us from defining h. Note that many jobs may contribute to making a time t busy since we don't have a nice structural property given by the intervals but not by the chains. In particular, when a lot of jobs are partially processed around time t, the time will highly likely to become non-idle. This create an issue for the analysis since we don't get enough idle times compared to the volume of jobs we used.

Hence we have to bound C_j by taking a more global view of the schedule. In the analysis, we will consider two cases. Let W denote the volume of work done by LP before τ_j. If W mostly comes from jobs that are processed very little before τ_j, we can reduce the first quantity in (4) using the non-uniform distribution. Otherwise, we can show that a large fraction of W comes from jobs that are processed a lot by the LP by time $(9/10)\tau_j$. Then, either a lot of jobs complete by time τ_j or the entire interval $[(9/10)\tau_j, \tau_j]$ becomes non-idle. In either case, we can have a better bound on the second quantity in Eq. (4) than the trivial τ_j. Somewhat subtle definitions are needed for the analysis, but this is a high-level overview.

A. Analysis

We fix $j \in J, i \in M$ and $\tau \in (0, T]$ and condition on the event that $i_j = i$ and $\tau_j = \tau$. Using the same notations as before, let $\widehat{\mathbb{E}}[\cdot]$ denote $\mathbb{E}[\cdot | i_j = i, \tau_j = \tau]$ and $\widehat{\Pr}[\cdot]$ denote $\Pr[\cdot | i_j = i, \tau_j = \tau]$. For any $t \in (0, T]$, we say t is idle if there are no jobs scheduled at time t on machine i, and use $\mathsf{idle}(t)$ to indicate whether t is idle or not. We will again use Eq. (4) for the analysis of $\mathbb{E}\, C_j$.

We do not try to optimize the approximation ratio. Rather we will use a distribution $\boldsymbol{\Theta}$ that is very close to the uniform distribution to make the analysis more transparent. The probability density function (PDF) of $\boldsymbol{\Theta}$ is $f(\theta) = 1/(1-2\lambda)$ if $\theta \in (\lambda, 1-\lambda)$ and $f(\theta) = 0$ otherwise, where $\lambda \in (0, 0.005)$ is some constant to be decided later. Let $F(t) = \int_0^t f(\theta)\mathsf{d}\theta$ be its cumulative distribution function (CDF). Note that this is a uniform distribution with small portion of both ends clipped out. It is not hard to show that if $\lambda = 0$ then we can still obtain a 2-approximation. The following claims easily follow from elementary algebra.

Claim 13. *For any θ, θ' such that $\lambda \le \theta \le \theta' \le 1$, we have $\frac{F(\theta)}{\theta} \le \frac{\theta' - \lambda}{\theta'(1-2\lambda)}$.*

Claim 14. *For any θ, θ' such that $\lambda \le \theta' \le 1/2$ and $\theta' \le \theta \le 1$, we have $\frac{F(\theta)}{\theta} \ge \frac{\theta' - \lambda}{\theta'(1-2\lambda)}$.*

We start by defining heavy and light chains. Roughly speaking, a chain $A \in \mathcal{A}^{i,j}$ is said to be heavy if a considerable fraction of the corresponding job j is processed before $\tau_j = \tau$, otherwise light.

Definition 15. *Given a chain $A \in \mathcal{A}^{i,j'}$ for some job $j' \neq j$, we say A is heavy if $A^{-1}(\tau) \ge p_{i,j'}/15$ and light otherwise. Let $\mathcal{A}^{i,j'}_{\mathsf{h}}$ and $\mathcal{A}^{i,j'}_{\mathsf{l}}$ be the sets of heavy and light chains in $\mathcal{A}^{i,j'}$, respectively.*

Let $W_{\mathsf{h}} = \sum_{j' \neq j, A \in \mathcal{A}^{i,j'}_{\mathsf{h}}} z_A A^{-1}(\tau)$ and $W_{\mathsf{l}} = \sum_{j' \neq j, A \in \mathcal{A}^{i,j'}_{\mathsf{l}}} z_A A^{-1}(\tau)$. Let $W = W_{\mathsf{h}} + W_{\mathsf{l}} = \sum_{j' \neq j, A \in \mathcal{A}^{i,j'}} z_A A^{-1}(\tau)$. Then, W is the total area of the portions of the rectangle chains before τ; here we view z_A fraction of chain A as a chain of rectangles with height z_A on times in A. In the light of this view, we immediately have $W \le \tau$.

We continue our analysis by considering two cases depending on how much light/heavy chains contribute to W.

1) Case 1: $W_{\mathsf{l}} \ge W/3$.: In this case, we focus on the expected total length of jobs scheduled on machine i before j. For a light chain, a large portion is after τ. Since the non-uniform distribution Θ moves the mass to the middle, it will give smaller expected total length if many chains are light. In the following, the first inequality is due to Claim 13 with $\theta' = 1/15$ and $\theta = \frac{A^{-1}(\tau)}{p_{i,j'}}$.

$$\widehat{\mathbb{E}}\Big[\sum_{j' \in J_i : \tau_{j'} < \tau} p_{i,j'}\Big] = \sum_{j' \neq j, A \in \mathcal{A}^{i,j'}} z_A F\left(\frac{A^{-1}(\tau)}{p_{i,j'}}\right) p_{i,j'}$$

$$\le \frac{1}{1-2\lambda} \sum_{j' \neq j, A \in \mathcal{A}^{i,j'}_{\mathsf{h}}} z_A \left(\frac{A^{-1}(\tau)}{p_{i,j'}}\right) p_{i,j'} + \frac{(1/15 - \lambda)}{(1/15)(1-2\lambda)} \sum_{j' \neq j, A \in \mathcal{A}^{i,j'}_{\mathsf{l}}} z_A \left(\frac{A^{-1}(\tau)}{p_{i,j'}}\right) p_{i,j'}$$

$$= \frac{1}{1-2\lambda} \sum_{j' \neq j, A \in \mathcal{A}^{i,j'}} z_A A^{-1}(\tau) - \left(\frac{1}{1-2\lambda} - \frac{1-15\lambda}{1-2\lambda}\right) \sum_{j' \neq j, A \in \mathcal{A}^{i,j'}_{\mathsf{l}}} z_A A^{-1}(\tau)$$

$$= \frac{W}{1-2\lambda} - \frac{15\lambda}{1-2\lambda} W_{\mathsf{l}} \le \frac{1-5\lambda}{1-2\lambda} W \le \frac{1-5\lambda}{1-2\lambda} \tau.$$

Thus, we have:

$$\widehat{\mathbb{E}}[C_j] \le \Big(2 - \frac{3\lambda}{1-2\lambda}\Big)\tau + p_{i,j}. \tag{7}$$

2) Case 2: $W_{\mathsf{h}} \ge 2W/3$.: In this case, we shall further divide heavy chains into good and bad ones. Roughly speaking, a good chain doesn't process the corresponding job too much very close to τ. Intuitively, good chains will likely lead to the job being processed considerably before time τ. We will show that there are 'enough' good chains that will make a lot of times before τ non-idle. Due to the space constraints, we omit most of the proofs.

Definition 16. *We say a heavy chain $A \in \mathcal{A}^{i,j'}$ for some $j' \neq j$ is good, if $A^{-1}(9\tau/10) \ge A^{-1}(\tau)/2$, and bad otherwise. Let $\mathcal{A}^{i,j'}_{\mathsf{hg}}$ and $\mathcal{A}^{i,j'}_{\mathsf{hb}}$ be the sets of good and bad heavy chains in $\mathcal{A}^{i,j'}_{\mathsf{h}}$, respectively.*

Let $W_{\mathsf{hg}} = \sum_{j' \neq j, A \in \mathcal{A}^{i,j'}_{\mathsf{hg}}} z_A A^{-1}(\tau)$ and $W_{\mathsf{hb}} =$

$\sum_{j' \neq j, A \in \mathcal{A}_{\mathsf{hb}}^{i,j'}} z_A A^{-1}(\tau)$ respectively. So, $W_{\mathsf{h}} = W_{\mathsf{hg}} + W_{\mathsf{hb}}$. Next we show that there are not many bad chains.

Claim 17. $W_{\mathsf{hb}} \leq \tau/5$.

Thus, we have secured lots of good chains, precisely $W_{\mathsf{hg}} \geq 2W/3 - \tau/5$.

Lemma 18. $\int_0^\tau (1 - \mathsf{idle}(t))\mathsf{d}t \geq \min\Big\{\tau/10, \sum_{j' \in J_i : \tau_{j'} < 9\tau/10} p_{i,j'}\Big\}$.

We will lower bound the expected value of the RHS in Lemma 18 as follows. Note that we only use jobs j' that have good chains since other jobs are not very useful for deriving a lower bound. The proof is somewhat technical, so we first derive an upper bound on $\widehat{\mathbb{E}} C_j$ assuming that the bound is true.

Lemma 19. $Q := \widehat{\mathbb{E}} \min\left\{\frac{\tau}{10}, \sum_{j' \in J_i : A_{j'} \in \mathcal{A}_{\mathsf{hg}}^{i,j'}, \tau_{j'} < 9\tau/10} p_{i,j'}\right\} \geq \frac{\gamma W_{\mathsf{hg}}}{10}$ *where* $\gamma = \left(1 - \frac{1}{e}\right)\frac{(1-30\lambda)}{30(1-2\lambda)}$.

Lemmas 18 and 19 will give us an upper bound on the length of idle times before τ. To bound the total expected volume of jobs with smaller τ values than job j, we use the following obvious bound.

$$\widehat{\mathbb{E}}\Big[\sum_{j' \in J_i : \tau_{j'} < \tau} p_{i,j'}\Big] = \sum_{j' \neq j, A \in \mathcal{A}^{i,j'}} z_A F\left(\frac{A^{-1}(\tau)}{p_{i,j'}}\right) p_{i,j'}$$
$$\leq \frac{1}{1-2\lambda} \sum_{j' \neq j, A \in \mathcal{A}^{i,j'}} z_A \left(\frac{A^{-1}(\tau)}{p_{i,j'}}\right) p_{i,j'} = \frac{1}{1-2\lambda} W.$$

Applying these two bound to Eq. (4) and using the fact that $W \leq \tau$, we have

$$\begin{aligned}\hat{\mathbb{E}}[C_j] &\leq \frac{1}{1-2\lambda} W + \left(\tau - \frac{\gamma W_{\mathsf{hg}}}{10}\right) + p_{i,j} \\ &\leq \frac{1}{1-2\lambda} W + \tau - \frac{\gamma}{10}\left(\frac{2W}{3} - \frac{\tau}{5}\right) + p_{i,j} \\ &\leq \frac{1}{1-2\lambda}\tau + \tau - \frac{\gamma}{10} \cdot \frac{7\tau}{15} + p_{i,j} \\ &= \frac{2 - 2\lambda - (1-1/e)(1/30 - \lambda)(7/150)}{1-2\lambda}\tau + p_{i,j}. \end{aligned} \tag{8}$$

This bound (8) will be combined with (7) for Case 1 in the following section to complete the analysis.

The remainder of this section is devoted to proving Lemma 19. The main difficulty in lower bounding Q is no matter how big the second term in Q is, the quantity is capped at $\tau/10$. Hence if jobs are very large compared to the cap, Q can be very small. Fortunately, we have found lots of good chains. Good chains process their corresponding jobs considerably before τ. This implies that such jobs cannot be very large compared to τ.

For formal proof, we define a random function $\Psi(\alpha, p')$ over a vector $\alpha \in [0,1]^{J \setminus j}$ and $p' \in \mathbb{R}_{\geq 0}^{J \setminus j}$ as follows. Initially let $S \leftarrow 0$. Then for every $j' \neq j$, with probability $\alpha_{j'}$, we let $S = S + p_{j'}$. Then let $\Psi(\alpha, p') = \min\{\tau/10, S\}$. We define $\alpha^*_{j'} = \widehat{\Pr}[j' \in J_i, A_{j'} \in \mathcal{A}_{\mathsf{hg}}^{i,j'}, \tau_{j'} < 9\tau/10]$ and $p^*_{j'} = p_{i,j'}$ for every $j' \neq j$. Then Q is exactly $\mathbb{E}\,\Psi(\alpha^*, p^*)$. The following lemma will allow us to increase job sizes while keeping their expected contribution to S the same.

Lemma 20. *If for some job $j' \neq j$ and some real number $a \geq 1$, we update $\alpha_{j'}$ to $\alpha_{j'}/a$ and $p'_{j'}$ to $ap'_{j'}$, then $\mathbb{E}\,\Psi(\alpha, p')$ can only decrease.*

The next step is to show that a large fraction of the second quantity in Q (or $\sum_{j' \neq j} \alpha^*_{j'} p^*_{j'}$) comes from good chains.

Lemma 21. $\sum_{j' \neq j} \alpha^*_{j'} p^*_{j'} \geq \frac{1-30\lambda}{2(1-2\lambda)} W_{\mathsf{hg}}$.

Notice if $\alpha^*_{j'} > 0$ then $\mathcal{A}_{\mathsf{hb}}^{i,j} \neq \emptyset$. Thus, taking an arbitrary $A \in \mathcal{A}_{\mathsf{hb}}^{i,j}$, we have $p^*_{j'} = p_{i,j'} \leq 15A^{-1}(\tau) \leq 15\tau$. Initially let $\alpha = \alpha^*$ and $p' = p^*$. Then we apply Lemma 20: for every j' such that $\alpha_{j'} > 0$, we scale down $\alpha_{j'}$ and scale up $p'_{j'}$ by the same factor so that $p'_{j'}$ becomes 15τ. After the update, we have $\mathbb{E}\,\Psi(\alpha, p') \leq Q$. Moreover, $\sum_{j' \neq j} \alpha_{j'} p'_{j'} \geq \frac{1-30\lambda}{2(1-2\lambda)} W_{\mathsf{hg}}$ as the operations maintained the left-hand-side in the bound of Lemma 21. Thus, $\sum_{j' \neq j} \alpha_{j'} \geq \frac{1-30\lambda}{30(1-2\lambda)} \frac{W_{\mathsf{hg}}}{\tau}$. Now, consider the process for computing $\Psi(\alpha, p')$. The probability that we add some $p'_{j'} = 15\tau$ to S is

$$\begin{aligned} &1 - \prod_{j' \neq j}(1 - \alpha_{j'}) \geq 1 - \prod_{j' \neq j} e^{-\alpha_{j'}} \\ &= 1 - \exp\Big(-\sum_{j' \neq j} \alpha_{j'}\Big) \geq 1 - \exp\left(-\frac{(1-30\lambda)W_{\mathsf{hg}}}{30(1-2\lambda)\tau}\right) \\ &\geq \left(1 - \frac{1}{e}\right)\frac{(1-30\lambda)W_{\mathsf{hg}}}{30(1-2\lambda)\tau} = \gamma \frac{W_{\mathsf{hg}}}{\tau}. \end{aligned}$$

Thus, $\mathbb{E}\,\Psi(\alpha, p') \geq \frac{\gamma W_{\mathsf{hg}}}{\tau} \cdot \frac{\tau}{10} = \frac{\gamma W_{\mathsf{hg}}}{10}$. Note that the expectation is lower bounded by the probability multiplied by $\tau/10$ since the total size of S is capped at $\tau/10$ in Ψ. This completes the proof of Lemma 19.

3) Wrapping up: Combining the Two Cases: We set $\lambda = 1/5100$, then in both cases (Eq. (7) and (8)), we have $\widehat{\mathbb{E}}[C_j] < 1.99942\tau + p_{i,j}$. For a chain $A \in \mathcal{A}^{i,j}$, $\mathbb{E}[\tau_j | i_j = i, A_j = A]$ is at most $A(p_{i,j}) - p_{i,j}/2$; this is where $\mathsf{LP}_{\mathsf{chain}}$ with each chain's cost associated with the corresponding job's completion time plays a crucial role.

$$\begin{aligned}\mathbb{E}[C_j] &< \sum_{i \in M, A \in \mathcal{A}^{i,j}} z_A \left(1.99942(C_A - p_{i,j}/2) + p_{i,j}\right) \\ &= \sum_{i \in M, A \in \mathcal{A}^{i,j}} z_A (1.99942 C_A + 0.00029 p_{i,j}) \\ &\leq 1.99971 \sum_{i \in M, A \in \mathcal{A}^{i,j}} z_A C_A. \end{aligned}$$

This is exactly 1.99971 times the (unweighted) contribution of j to the LP solution. Thus, our algorithm is a 1.99971-approximation, implying Theorem 3.

REFERENCES

[1] W. E. Smith, "Various optimizers for single-stage production," *Naval Research Logistics Quarterly*, vol. 3, no. 1-2, pp. 59–66, 1956.

[2] J. K. Lenstra, A. R. Kan, and P. Brucker, "Complexity of machine scheduling problems," *Annals of discrete mathematics*, vol. 1, pp. 343–362, 1977.

[3] H. Hoogeveen, P. Schuurman, and G. J. Woeginger, "Non-approximability results for scheduling problems with minsum criteria," *INFORMS Journal on Computing*, vol. 13, no. 2, pp. 157–168, 2001.

[4] M. Skutella, "Convex quadratic and semidefinite programming relaxations in scheduling," *J. ACM*, vol. 48, no. 2, pp. 206–242, Mar. 2001. [Online]. Available: http://doi.acm.org/10.1145/375827.375840

[5] A. S. Schulz and M. Skutella, "Scheduling unrelated machines by randomized rounding," *SIAM J Discrete Math*, vol. 15, no. 4, pp. 450–469, 2002.

[6] P. Schuurman and G. J. Woeginger, "Polynomial time approximation algorithms for machine scheduling: Ten open problems," *J. Scheduling*, vol. 2, no. 5, pp. 203–213, 1999.

[7] V. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan, "Minimum weighted completion time," in *Encyclopedia of Algorithms*. Springer, 2008, pp. 544–546.

[8] M. Sviridenko and A. Wiese, "Approximating the configuration-lp for minimizing weighted sum of completion times on unrelated machines," in *Integer Programming and Combinatorial Optimization*. Springer, 2013, pp. 387–398.

[9] N. Bansal, A. Srinivasan, and O. Svensson, "Lift-and-round to improve weighted completion time on unrelated machines," in *STOC*, 2016, pp. 156–167.

[10] J. Sethuraman and M. S. Squillante, "Optimal scheduling of multiclass parallel machines," in *SODA*, 1999, pp. 963–964.

[11] F. Afrati, E. Bampis, C. Chekuri, D. Karger, C. Kenyon, S. Khanna, I. Milis, M. Queyranne, M. Skutella, C. Stein *et al.*, "Approximation schemes for minimizing average weighted completion time with release dates," in *FOCS*. IEEE, 1999, pp. 32–43.

[12] L. A. Hall, A. S. Schulz, D. B. Shmoys, and J. Wein, "Scheduling to minimize average completion time: Off-line and on-line approximation algorithms," *Mathematics of operations research*, vol. 22, no. 3, pp. 513–544, 1997.

[13] M. X. Goemans, M. Queyranne, A. S. Schulz, M. Skutella, and Y. Wang, "Single machine scheduling with release dates," *SIAM J Discrete Math*, vol. 15, no. 2, pp. 165–192, 2002.

[14] C. Phillips, C. Stein, and J. Wein, "Task scheduling in networks," *SIAM J Discrete Math*, vol. 10, no. 4, pp. 573–598, 1997.

[15] D. B. Shmoys and É. Tardos, "An approximation algorithm for the generalized assignment problem," *Math program*, vol. 62, no. 1-3, pp. 461–474, 1993.

[16] M. Skutella and G. J. Woeginger, "A ptas for minimizing the total weighted completion time on identical parallel machines," *Mathematics of Operations Research*, vol. 25, no. 1, pp. 63–75, 2000.

[17] C. Chekuri and S. Khanna, "A ptas for minimizing weighted completion time on uniformly related machines," in *ICALP*. Springer, 2001, pp. 848–861.

[18] J. K. Lenstra, D. B. Shmoys, and É. Tardos, "Approximation algorithms for scheduling unrelated parallel machines," *Math program*, vol. 46, no. 1-3, pp. 259–271, 1990.

[19] O. Svensson, "Santa claus schedules jobs on unrelated machines," *SIAM Journal on Computing*, vol. 41, no. 5, pp. 1318–1341, 2012.

[20] N. Bansal and M. Sviridenko, "The santa claus problem," in *STOC*. ACM, 2006, pp. 31–40.

[21] D. Chakrabarty, J. Chuzhoy, and S. Khanna, "On allocating goods to maximize fairness," in *FOCS*, 2009, pp. 107–116.

[22] B. Haeupler, B. Saha, and A. Srinivasan, "New constructive aspects of the lovász local lemma," *J. ACM*, vol. 58, no. 6, p. 28, 2011. [Online]. Available: http://doi.acm.org/10.1145/2049697.2049702

[23] Y. Azar and A. Epstein, "Convex programming for scheduling unrelated parallel machines," in *STOC*, 2005, pp. 331–337.

[24] V. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan, "A unified approach to scheduling on unrelated parallel machines," *J. of ACM*, vol. 56, no. 5, p. 28, 2009.

[25] N. Bansal and J. Kulkarni, "Minimizing flow-time on unrelated machines," in *STOC*, 2015, pp. 851–860.

[26] N. Garg and A. Kumar, "Minimizing average flow-time : Upper and lower bounds," in *FOCS*, 2007, pp. 603–613.

[27] N. Garg, A. Kumar, and V. N. Muralidhara, "Minimizing total flow-time: The unrelated case," in *ISAAC*, 2008, pp. 424–435.

[28] C. Chekuri and S. Khanna, "Approximation algorithms for minimizing average weighted completion time," 2004.