Update on the Asymptotic Optimality of LPT

Anne Benoit, Louis-Claude Canon, Redouane Elghazi, Pierre-Cyrille Heam

Update on the Asymptotic Optimality of LPT

A. Benoit[1], L.-C. Canon[2], R. Elghazi[2], and P.-C. Héam[2]

1. LIP, ENS Lyon, France; 2. FEMTO-ST, U. Franche-Comté, France

Abstract. When independent tasks are to be scheduled onto identical processors, the typical goal is to minimize the makespan. A simple and efficient heuristic consists in scheduling first the task with the longest processing time (LPT heuristic), and to plan its execution as soon as possible. While the performance of LPT has already been largely studied, in particular its asymptotic performance, we revisit results and propose a novel analysis for the case of tasks generated through uniform integer compositions. Also, we perform extensive simulations to empirically assess the asymptotic performance of LPT. Results demonstrate that the absolute error rapidly tends to zero for several distributions of task costs, including ones studied by theoretical models, and realistic distributions coming from benchmarks.

1 Introduction

We revisit the classical problem of scheduling n independent tasks with costs $p_1, \ldots, p_n$ onto m identical processors. The goal is to minimize the total execution time, or *makespan*, usually denoted by $C_{\max}$. This problem, denoted $P||C_{\max}$ in Graham's notation [14], has been extensively studied in the literature, and greedy heuristics turn out to have theoretical guarantees and to perform well in practice. In particular, we focus on the *Longest Processing Time (LPT)* heuristic, where the longest task will be scheduled first, on the processor where it can start the earliest. This heuristic is very simple and has a low complexity, while exhibiting good worst-case performance [15], and excellent empirical one. With a large number of tasks, LPT appears to be almost optimal.

Since the worst-case performance exhibits cases where LPT is further from the optimal, many different approaches have tried to fill the gap between this worst-case performance and the excellent practical performance. The goal is to provide performance guarantees of different kinds, for instance by studying the average-case complexity, some generic-case complexity, or convergence results.

Hence, many convergence results have been proposed in the literature. They state that LPT ends up providing an optimal solution when the number of tasks grows towards infinity. Some of these results even provide asymptotic rates that quantify the speed with which LPT tends to optimally. These results depend on assumptions on the probability distribution of the costs of the tasks, and on the definition of distance to optimality. However, the literature lacks a definitive answer on the convergence to optimality and its rate when faced with difficult

cost distributions. In particular, this work is the first to consider dependent random costs with a constraint on the minimum cost.

First, Section 2 synthesizes the existing contributions and their limitations. Then, we revisit LPT and propose an update to these asymptotic optimality results, both from a theoretical perspective and from an empirical one. We also consider related heuristics, in particular a novel strategy recently proposed in [7]. Our contribution is twofold:

1. We derive a new convergence (in probability) result when the distribution of task costs is generated using uniform integer compositions, hence leading to a novel probabilistic analysis of the heuristics for this problem (Section 3);
2. We perform a thorough empirical analysis of these heuristics, with an extended range of settings to study particular distributions but also distributions coming from real applications (Section 4).

2 Related Work

Theoretical studies. There are several theoretical works studying the rate of convergence of LPT. Coffman et al. [3] analyze the average performance of LPT under the assumption that costs are uniformly distributed in the interval $(0, 1]$. They show that the ratio between the expected makespan obtained with LPT and the expected optimal one with preemption is bounded by $O(1 + \frac{m^2}{n^2})$, where m is the number of processors and n is the number of tasks.

Frenk and Rinnooy Kan [13] bound the *absolute error* (i.e., the difference between the achieved makespan and the optimal one) of LPT using order statistics of the processing times when the cost distribution has a cumulative distribution function of the form $F(x) = x^a$ with $0 < a < \infty$. The results also stand when this constraint is relaxed into $F(x) = \Theta(x^a)$. They prove that the absolute error goes to 0 with speed $O\left((\frac{\log\log(n)}{n})^{\frac{1}{a}}\right)$ as the number of tasks n grows. For higher moments, of order q, a similar technique gives a speed of $O\left((\frac{1}{n})^{\frac{a}{q}}\right)$.

Frenk and Rinnooy Kan [12] also study uniform machines ($Q||C_{\max}$) in the more general case where costs follow a distribution with finite moment and the cumulative distribution function is strictly increasing in a neighbourhood of 0. They show that LPT is asymptotically optimal almost surely in terms of absolute error. When it is the second moment that is finite instead, they show that LPT is asymptotically optimal in expectation. For the more specific cases where the costs follow either a uniform distribution or a negative exponential distribution, they provide additional convergence rates.

Another theoretical study is done by Loulou [19], providing a comparison between LPT and a less sophisticated heuristic, RLP (Random List Processing), also called LS (List Scheduling) in this paper. This heuristic is simpler than LPT because the jobs are considered in an arbitrary order instead of a sorted order. These algorithms are studied under the assumption that the costs are independent and identically distributed (i.i.d.) random variables with finite first

moment. Under this assumption, the absolute error of RLP with at least three processors and LPT are both stochastically bounded by a finite random variable. The author also proves that the absolute error of LPT converges in distribution to optimality with rate $O(1/n^{1-\epsilon})$.

Coffman et al. [5] list various results and techniques that are useful for the study of the problems of scheduling and bin packing. They consider both theoretical optimal results, and heuristic algorithm results. LPT is one of the algorithms they study, in terms of both relative error (LPT/OPT) and absolute error (LPT − OPT). They also reuse the specific probability distribution used by Frenk and Rinnooy Kan [13] of the form $F(x) = x^a$, with $0 < a < \infty$. They present a heuristic adapted from a set-partitioning problem with a better convergence on this distribution.

Piersma and Romeijn [21] have considered the $R||C_{\max}$ problem (with unrelated machines), and they propose an LP relaxation of the problem, followed by a Lagrange relaxation. Assuming that the processing times are i.i.d. random vectors of $[0,1]^m$, they prove that $\frac{1}{n}$OPT converges almost surely to a value θ that they give (it depends on the Lagrange relaxation). Using a previous convergence result [13], they infer that the makespan of LPT also converges a.s. to $n\theta$.

Dempster et al. [8] consider an objective function also depending on the machine cost, and they propose a heuristic in two steps, where they first choose the machines to be bought with knowledge of the distribution of the jobs, and then schedule the jobs on the machines that were bought in the first step. For identical machines, assuming that the processing times are i.i.d. random variables with finite second moment, they prove that the relative error of their heuristic converges to 0 in expectation and probability when the number of jobs goes to infinity. For uniform machines, they need more assumptions to reach results.

Table 1 summarizes the main results that are known about LPT.

Table 1. For each main result, the problem may consider uniform processors (P) or processors with speeds (Q or R). A result on the absolute difference is stronger than on the ratio. OPT is the optimal makespan, whereas OPT* is the optimal makespan with preemption.

	Problem	Distribution	Studied quantity	Convergence/rate
[3]	$P\|\|C_{\max}$	$\mathcal{U}(0,1)$	$E[\text{LPT}]/E[\text{OPT}^*]$	$1+O(m^2/n^2)$
[13]	$P\|\|C_{\max}$	$F(x)=x^a$, $0<a<\infty$	LPT − OPT	$O((\log\log(n)/n)^{\frac{1}{a}})$ almost surely (a.s.)
[13]	$P\|\|C_{\max}$	as above	$E[(\text{LPT}-\text{OPT})^q]$	$O((1/n)^{\frac{a}{q}})$
[12]	$Q\|\|C_{\max}$	finite 1st moment	LPT − OPT	a.s.
[12]	$Q\|\|C_{\max}$	finite 2nd moment	LPT − OPT	in expectation
[12]	$Q\|\|C_{\max}$	$\mathcal{U}(0,1)$ or $Exp(\lambda)$	LPT − OPT	$O(\log n/n)$ a.s.
[12]	$Q\|\|C_{\max}$	$\mathcal{U}(0,1)$	$E[\text{LPT}]-E[\text{OPT}]$	$O(m^2/n)$
[19]	$P\|\|C_{\max}$	finite 1st moment	LPT − OPT	bounding finite RV
[19]	$P\|\|C_{\max}$	$\mathcal{U}(0,1)$	LPT − OPT	$O(1/n^{1-\epsilon})$ in dist.
[5]	$P\|\|C_{\max}$	$\mathcal{U}(0,1)$	$E[\text{LPT}-\text{OPT}]$	$O(m/(n+1))$
[21]	$R\|\|C_{\max}$	$\mathcal{U}(0,1)$	OPT	$n\theta$ a.s.

Beyond LPT. Even though LPT has interesting properties in terms of convergence, other heuristics have been designed for the multiprocessor scheduling problem. For independent tasks and makespan minimization, the problem is actually close to a bin-packing problem, where one would like to create m bins of same size. Hence, the MULTIFIT heuristic [4] builds on techniques used in bin-packing, and it provides an improved worst-case bound.

Then, a COMBINE heuristic was proposed [18], combining MULTIFIT and LPT to get the best of these two heuristics. Another alternative, LISTFIT, was proposed in [16], still with the goal to minimize the makespan on identical machines.

Building on the Largest Differencing Method of Karmarkar and Karp [20], a novel heuristic was proposed, outperforming LPT and MULTIFIT from an average-case perspective.

More recently, Della Croce and Scatamacchia [7] revisit LPT to propose yet another heuristic, SLACK, by splitting the sorted tasks in tuples of m consecutive tasks (recall that m is the number of processors), and then sorting tuples by non-increasing order of the difference between the largest and smallest task in the tuple. A list-scheduling strategy is then applied with tasks sorted in this order. Moreover, LPT last step is enhanced to reach a better worst-case approximation ratio.

Empirical studies. An empirical comparison of LISTFIT with MULTIFIT, COMBINE and LPT is proposed in [16]. Several parameters are varied, in particular the number of machines, number of jobs, and the minimum and maximum values of a uniform distribution for processing times. No other distribution is considered. LISTFIT turns out to be robust and returns better makespan values than previous heuristics.

Behera and Laha [1] consider the three heuristics MULTIFIT, COMBINE and LISTFIT, and propose a comprehensive performance evaluation. While LISTFIT outperforms the two other heuristics, this comes at a price of an increased time complexity. They do not consider instances with more than 300 tasks, and no comparison with LPT is done.

An empirical evaluation of LPT was proposed in [17], showing that LPT consumes less computational time than the competitors (MULTIFIT, COMBINE, LISTFIT), but returns schedules with higher makespan values. However, here again, there is no study of the convergence, and no comparison of LPT with other simpler algorithms.

Finally, an evaluation of SLACK is done in [7]: this variant of LPT turns out to be much better than LPT on benchmark literature instances, and it remains competitive with the COMBINE heuristic that is more costly and more difficult to implement.

Beyond independent tasks. While we have been focusing so far on independent tasks, there have also been some empirical analysis of list scheduling for general directed acyclic graphs (DAGs), i.e., with dependencies. For instance,

Cooper et al. [6] evaluate various list schedulers on benchmark codes, pointing out cases where a basic list scheduling algorithm works well, and where more sophisticated approaches are helpful. In this paper, we focus on independent tasks to study the convergence of LPT and other heuristics.

3 Convergence Results for Integer Compositions

In this section, we derive new convergence results for four heuristics that are first described in Section 3.1. These results apply when the distribution of task costs is generated following an integer composition method. In contrast to related work where the number of tasks n is known beforehand, this consists in considering that the total amount of work W is fixed (costs are thus dependent random variables). We detail how tasks are generated among possible decompositions of this work (Section 3.2). We finally perform the probabilistic analysis in two different settings, depending whether the minimum cost of tasks is one (Section 3.3) or greater (Section 3.4).

The proofs of the results in this section are mainly based on combinatorics techniques. The reader is referred to [10] for more information. All the detailed proofs are available in the companion research report [2].

3.1 Algorithms

We consider four different list scheduling algorithms: they order the tasks in some way, and then successively assign tasks in a greedy manner, to the processor that has the lowest current finishing time (or makespan). Hence, tasks are always started as soon as possible, and for independent tasks, there is no idle time in the schedule.

The four algorithms differ in the way they first order the tasks:

- LS: List Scheduling is the basic list scheduling algorithm that does not order the tasks, but rather considers them in an arbitrary order. The time complexity of LS is $O(n \log m)$.
- LPT: Largest Processing Time orders the tasks from the largest to the smallest. The time complexity of LPT is $O(n \log n)$.
- MD: Median Discriminated is an attempt to find a intermediate solution between LPT and LS. The tasks are not completely sorted, but the median of the execution times is computed so that the first $\frac{n}{2}$ processed tasks are larger than the median, while the next $\frac{n}{2}$ are smaller. The time complexity of MD is $O(n \log m)$.
- SLACK: as defined by Della Croce and Scatamacchia in [7], it makes packs of m tasks and defines for each of these packs the slack, which is the difference between the largest and the smallest task of the pack. The packs are then sorted from the largest to the smallest slack and the tasks are ordered in the order incurred by the order of the packs. The time complexity of SLACK is $O(n \log n)$.

3.2 Tasks Random Generation

A W-composition is a finite sequence $p_1, \ldots, p_n$ of strictly positive integers such that $p_1 + \ldots + p_n = W$.

Let $\mathbb{D}_W$ be the uniform distribution over W-compositions and $\mathbb{D}_{W,p_{\min}}$ the uniform distribution over W-compositions satisfying for each i, $p_i \geq p_{\min}$. In particular, $\mathbb{D}_{W,1} = \mathbb{D}_W$. For instance, $\mathbb{D}_4$ is the uniform distribution over the eight elements $(1,1,1,1)$, $(1,1,2)$, $(1,2,1)$, $(1,3)$, $(2,1,1)$, $(2,2)$, $(3,1)$, (4); $\mathbb{D}_{4,2}$ is the uniform distribution over $(2,2)$ and (4). Note that for $\mathbb{D}_4$, the probability that $p_1 = 1$ is $1/2$ and the probability that $p_1 = 3$ is $1/8$.

In practice, random generation is performed using the recursive method [11].

For a list L of task costs, we denote by $\mathrm{LPT}(L,m)$ the makespan $C_{\max}$ returned by LPT on m machines. We define as well $\mathrm{LS}(L,m)$, $\mathrm{MD}(L,m)$ and $\mathrm{SLACK}(L,m)$ for the other heuristics. The optimal (minimum) $C_{\max}$ that can be obtained by any algorithm is similarly denoted $\mathrm{OPT}(L,m)$.

3.3 Probabilistic Analysis of List-Scheduling Heuristics for D_W

Ratio for $\mathbb{D}_W$. In this setting, we know the total workload W, but the number of tasks n is not fixed and there is no minimum task cost. Let $L[W] = (p_1, \ldots, p_n)$ be a sequence of positive integers such that $\sum_{i=1}^{n} p_i = W$, hence a W-composition.

According to [15],

$$\frac{\mathrm{LS}(L[W],m)}{\mathrm{OPT}(L[W],m)} \leq 1 + (m-1)\frac{p_{\max}}{\sum_{i=1}^{n} p_i} = 1 + (m-1)\frac{p_{\max}}{W},$$

where $p_{\max} = \max\{p_i\}$.

Following [10, page 310], for $\mathbb{D}_W$ and for any y,

$$\mathbb{P}\left(p_{\max} \geq 2\log_2 W + y\right) = O\left(\frac{e^{-2y}}{W}\right). \tag{1}$$

Since by definition of OPT, $\mathrm{OPT}(L[W],m) \leq \mathrm{LS}(L[W],m)$, for any fixed m,

$$\mathbb{P}\left(\frac{\mathrm{LS}(\mathbb{D}_W,m)}{\mathrm{OPT}(\mathbb{D}_W,m)} \leq 1 + 2(m-1)\frac{\log_2(W)}{W}\right) \underset{W\to+\infty}{\longrightarrow} 1. \tag{2}$$

It is also known, see [10, Proposition V.I.], that for the distribution $\mathbb{D}_W$, $E[p_{\max}] \sim \log_2 W$. By linearity of expectations, the following result holds:

$$E\left[\frac{\mathrm{LS}(\mathbb{D}_W,m)}{\mathrm{OPT}(\mathbb{D}_W,m)}\right] \underset{W\to+\infty}{\longrightarrow} 1.$$

The results also hold for LPT, MD and SLACK, which are particular list-scheduling heuristics.

Absolute error for $\mathbb{D}_{\boldsymbol{W}}$. The *absolute error* of a heuristic is the difference between its result and the optimal result. A first obvious upper bound is that $LS(L,m) - OPT(L,m) \leq p_{\max}$ (for any set of tasks L), and previous results on $p_{\max}$ can be used to bound the error (but not proving it tends to 0). Furthermore, we prove the following theorem:

Theorem 1. *Algorithms LPT and SLACK are optimal for $\mathbb{D}_W$, with probability $1 - O\left(\frac{1}{W}\right)$. For any fixed m, for L generated according to $\mathbb{D}_W$,*

$$\mathbb{P}(\mathrm{LPT}(L,m) = \mathrm{OPT}(L,m)) = 1 - O\left(\frac{1}{W}\right), \quad \textit{and}$$

$$\mathbb{P}(\mathrm{SLACK}(L,m) = \mathrm{OPT}(L,m)) = 1 - O\left(\frac{1}{W}\right), \quad \textit{and}$$

$$\mathbb{P}(\mathrm{MD}(L,m) \leq \mathrm{OPT}(L,m) + 1) = 1 - O\left(\frac{1}{W}\right).$$

The proof is building upon two lemmas, and can be found in the companion research report [2]. Note that the result for MD is almost similar, but up to 1 to the optimal.

Theorem 1 can be reformulated in a convergence in probability result:

Corollary 1. *For every $\varepsilon > 0$, for the distributions $\mathbb{D}_W$,*

$$\lim_{W \to +\infty} \mathbb{P}(|\mathrm{LPT}(L,m) - \mathrm{OPT}(L,m)| \geq \varepsilon) = 0, \quad \textit{and}$$

$$\lim_{W \to +\infty} \mathbb{P}(|\mathrm{SLACK}(L,m) - \mathrm{OPT}(L,m)| \geq \varepsilon) = 0, \quad \textit{and}$$

$$\lim_{W \to +\infty} \mathbb{P}(|\mathrm{MD}(L,m) - \mathrm{OPT}(L,m)| \geq 1 + \varepsilon) = 0.$$

Proof. $\mathbb{P}(|\mathrm{LPT}(L,m) - \mathrm{OPT}(L,m)| \geq \varepsilon) = \mathbb{P}(\mathrm{LPT}(L,m) - \mathrm{OPT}(L,m) \geq \varepsilon) \leq \mathbb{P}(\mathrm{LPT}(L,m) - \mathrm{OPT}(L,m) > 0) = 1 - \mathbb{P}(\mathrm{LPT}(L,m) - \mathrm{OPT}(L,m) = 0) = O\left(\frac{1}{W}\right)$. The proof is similar for SLACK and MD. □

3.4 Analysis for $\mathbb{D}_{\boldsymbol{W,p_{\min}}}$

Let $\min\{p_i\} = p_{\min} \geq 2$. Let $\alpha_{W,p_{\min}}$ be the number of p_i's equal to $p_{\min}$ in a decomposition $(p_1, \ldots, p_k)$ satisfying $\sum p_i = W$ and for every i, $p_i \geq p_{\min}$. The random variable $\alpha_{W,p_{\min}}$ is studied for the $\mathbb{D}_{W,p_{\min}}$ distribution. Let also $\gamma_{p_{\min},k}$ be the number of p_i's greater than or equal to k (with $k \geq p_{\min}$).

Theorem 2. *Let m be a fixed number of machines. One has, for L generated according to $\mathbb{D}_{W,p_{\min}}$, $\mathbb{P}(|\mathrm{LPT}(L,m) - \mathrm{OPT}(L,m)| \leq p_{\min}) \underset{W \to +\infty}{\longrightarrow} 1$ and $\mathbb{P}(|\mathrm{SLACK}(L,m) - \mathrm{OPT}(L,m)| \leq p_{\min}) \underset{W \to +\infty}{\longrightarrow} 1$.*

Here again, the proof, based on two lemmas, can be found in the companion research report [2]. Note that we do not have yet any theoretical results for MD for $\mathbb{D}_{W,p_{\min}}$, but experimental results explored in Section 4 are encouraging. Finally, we obtain the following corollary:

Corollary 2. *For every $\varepsilon > 0$, every $p_{\min} \geq 2$, for the distributions $\mathbb{D}_{W,p_{\min}}$,*

$$\lim_{W \to +\infty} \mathbb{P}(|\mathrm{LPT}(L,m) - \mathrm{OPT}(L,m)| < p_{\min} + \varepsilon) = 0, \quad \textit{and}$$

$$\lim_{W \to +\infty} \mathbb{P}(|\mathrm{SLACK}(L,m) - \mathrm{OPT}(L,m)| < p_{\min} + \varepsilon) = 0.$$

4 Empirical Study

The objective of this section is threefold: first, evaluate the tightness of the convergence rate proposed in [13] (Section 4.2); then, assess the performance of the four heuristics when generating costs with the integer composition approach (Section 4.3); finally, quantifying the convergence for realistic instance (Section 4.4). We first detail the experimental setting in Section 4.1. All the algorithms were implemented in Python 3, and the code is available on figshare.

4.1 Experimental Setting

Synthetic Instances. We consider two kinds of synthetic instances: (1) i.i.d. execution times with cumulative distribution function $F(x) = x^a$ for some $a > 0$. This distribution has an expected value of $\frac{a}{a+1}$ and a variance of $\frac{a}{(a+1)^2 \cdot (a+2)}$. These values can be seen as a function of a in Fig. 1. Note that for $a = 1$, this is a uniform distribution $\mathcal{U}(0,1)$. (2) The integer composition distribution considered in Section 3, that is to say a uniform distribution on all possible ways to decompose a total amount of work into integer values.

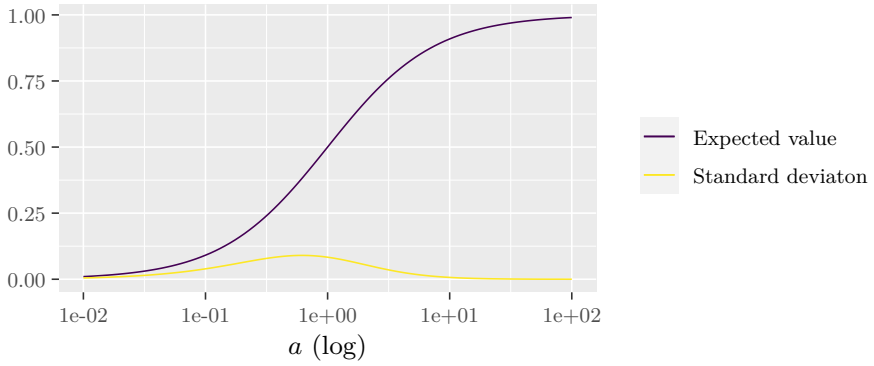

Fig. 1. Expected value and standard deviation of a random variable with cumulative distribution function $F(x) = x^a$ as a function of a with $0 < a < \infty$.

Realistic Instances. We also compare the four algorithms of Section 3.1 using real logs from the Parallel Workloads Archive, described in [9] and available at https://www.cs.huji.ac.il/labs/parallel/workload/. More specifically, we took the instances called KIT ForHLR II with 114 355 tasks and NASA Ames iPSC/860 with 18 239 tasks. The profiles of the task costs in these instances are presented in Fig. 2.

In order to also get instances for which the number of tasks n could change, we build new instances from these two instances. In the new instances, the tasks are i.i.d. random variables with an empirical cumulative distribution function that is computed from the distribution of the two original instances.

Optimality Transform. When studying the absolute error of an algorithm, we consider the difference of its makespan to the optimal one to measure the convergence when the number of tasks n goes to infinity. The optimal makespan is computationally hard to get, so as a first approach, we can take a lower bound instead of the actual optimal value. However, there is a risk of actually measuring the quality of the lower bound instead of the quality of the algorithm.

To address this problem, we transform the instances so that we know the optimal makespan. This transformation is described as follows:

- we take an instance with n tasks;
- we perform a random List Scheduling on this instance;
- from this schedule, we add a total of at most $m-1$ tasks so that all of the processors finish at the same time;
- we randomize the order of the tasks to avoid adding a bias to the heuristics;
- we end up with an instance with at most $n+m-1$ tasks such that the optimal makespan equals the sum of the execution times divided by the number of processors ($\mathrm{OPT} = \frac{W}{m}$).

As we are interested in the asymptotic behavior of the algorithms, m is small compared to n, and we expect this transformation to alter the task distribution only marginally. However, studying the precise distribution of the added tasks is left to future work.

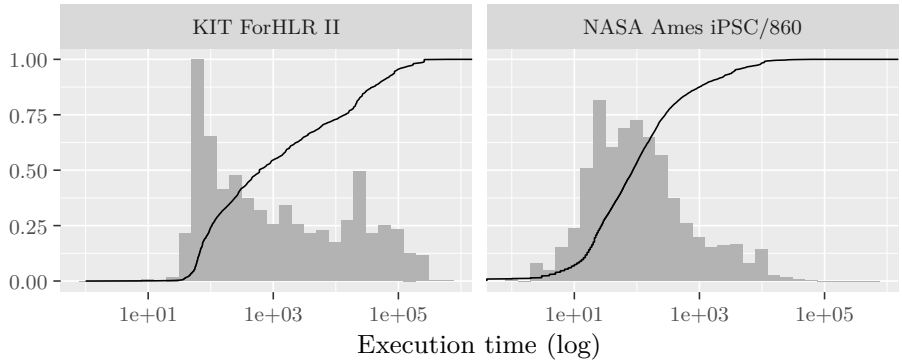

Fig. 2. Empirical cumulative distributions and histograms of task costs for the KIT ForHLR II and NASA Ames iPSC/860 instances.

4.2 Rate Tightness

We experimentally verify the bound given in [13]: if the tasks are independent and have cumulative distribution function $F(x) = x^a$ with $a > 0$, then the absolute error is a $O((\frac{\log\log(n)}{n})^{\frac{1}{a}})$ almost surely.

Fig. 3 depicts the absolute error of LPT and related heuristics (LS, MD and SLACK) for different values of n. The instance contains $n - m + 1$ costs generated with the considered distribution and is then completed with the optimality transform. Moreover, we plot $C \cdot (\frac{\log\log(n)}{n})^{\frac{1}{a}}$, where C is the lowest constant such that all of LPT values are under the bound.

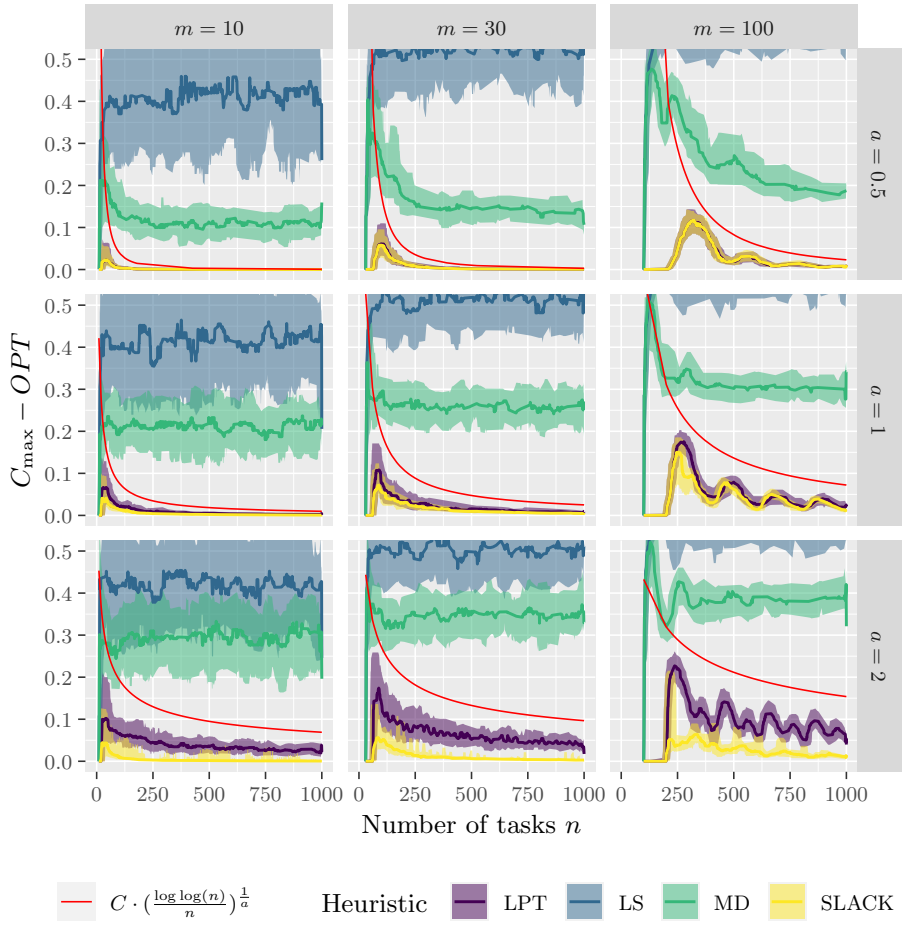

Fig. 3. Absolute error with a distribution of the form $F(x) = x^a$ with $a > 0$ (instances are transformed to obtain OPT). Smoothed lines are obtained by using a rolling median with 45 values (each value is set to the median of the 22 values on the left, the 22 on the right and the current one).

We can see that the bound seems to be rather tight for LPT, which confirms that the convergence rate of [13] is strong. Also, we can see that the absolute error of SLACK seems to converge to 0 at a similar rate than LPT, but with a lower multiplicative constant. On the other side, the absolute errors of LS and MD do not seem to converge to 0 at all, but MD performs significantly better than LS.

4.3 Uniform Integer Compositions

Some experiments have been performed for the distributions described in Section 3: a total workload W is fixed as well as a fixed number m of machines. Then, the list of task costs is uniformly picked among all the possible lists for the distribution $\mathbb{D}_W$; and among all the possible lists with a minimum cost $p_{\min}$ for the distribution $\mathbb{D}_{W,p_{\min}}$.

For $\mathbb{D}_W$, an instance has been generated for all W from 10 to 9999, for $m = 10$, $m = 30$, and $m = 100$. Instances are not transformed to avoid changing the total work W. Thus, we compare the makespan obtained by the heuristics to the lower bound on the optimal value OPT: $\max(\lceil \frac{W}{m} \rceil, p_{\max})$. In all cases (about 30 000), LPT and SLACK always reach this bound, which indicates that they are both optimal and the bound is tight with these instances. Results for LS and MD are reported in Table 2. The average absolute error for MD is 0.35 for this experiment with a standard deviation of 0.6. Moreover MD is optimal in 67.6% of the samples and up to 1 from the optimum in 98% of the samples. LS is optimal in 3.3% of the cases and the average error is 3.75 (s.d. 2.15).

Similar tests have been done for $\mathbb{D}_{W,p_{\min}}$ with $W \in \{10, \ldots, 9999\}$, $p_{\min} \in \{3, 5, 7, 10\}$ and $m \in \{10, 30, 100\}$ (see Table 3). We now focus on the difference δ between $C_{\max}$ and the lower bound. In each case, the maximal value of δ is reported, as well as its average and standard deviation. Note that for each sample, both SLACK and LPT ensure that $\delta < p_{\min}$, and MD ensures it in 99% of cases. The LS heuristic is less effective since for $p_{\min} = 3$, only 41% of the samples satisfy $\delta < p_{\min}$; 49% for $p_{\min} = 5$, 53% for $p_{\min} = 7$ and 58% for $p_{\min} = 10$. Results for the SLACK and LPT heuristics are very close. Over the about 120 000 samples, SLACK is strictly better than LPT only 55 times, when LPT is strictly better than SLACK only 54 times. Each time, the difference is either 1 or 2. On these distributions, SLACK and LPT seem to compete equally.

Table 2. Distribution of the absolute errors observed for LS and MD with W from 10 to 9999 and $m \in \{10, 30, 100\}$.

abs. err.	LS	MD	abs. err.	LS	MD
0	3.3	67.6	6	8.9	0.04
1	10.8	30.5	7	5.0	0.02
2	17.0	0.88	8	2.7	0.01
3	18.4	0.56	9	1.2	0.01
4	17.4	0.23	10	0.6	< 0.01
5	14.0	0.09	> 10	0.7	< 0.01

Table 3. Results on the difference between the $C_{\max}$ computed by the heuristics and a lower bound of the optimal makespan OPT. Each line is related to different $D_{W,p_{\min}}$. The first number is the maximum difference observed for all the samples, the second one is the average difference, and the last one is the standard deviation of this difference. Each value is obtained with W from 10 to 9999 and for $m \in \{10, 30, 100\}$.

$p_{\min}$	LPT	LS	MD	SLACK
3	2 – 0.92 – 0.71	16 – 2.53 – 0.92	10 – 1.47 – 0.58	2 – 0.92 – 0.71
5	4 – 1.82 – 1.23	24 – 4.28 – 1.51	10 – 2.66 – 0.90	4 – 1.82 – 1.23
7	6 – 2.69 – 1.81	26 – 5.95 – 2.15	12 – 3.63 – 1.20	6 – 2.69 – 1.81
10	9 – 3.92 – 2.67	38 – 8.32 – 3.22	15 – 5.17 – 1.86	9 – 3.92 – 2.67

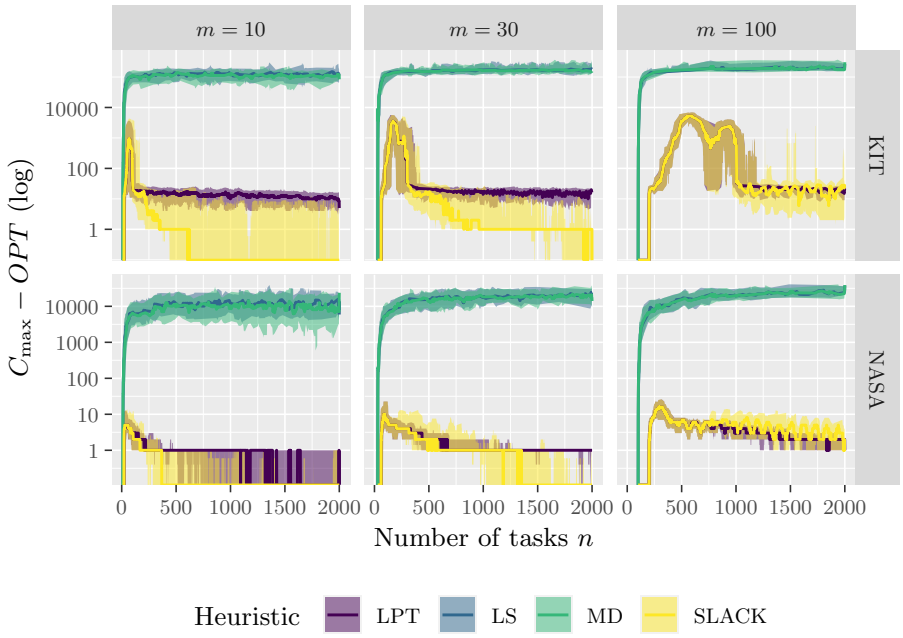

Fig. 4. Absolute error with costs derived from the KIT ForHLR II and NASA Ames iPSC/860 instances (after optimality transformation). Smoothed lines are obtained by using a rolling median with 45 values (each value is set to the median of the 22 values on the left, the 22 on the right and the current one). The ribbons represent the rolling 0.1- and 0.9-quantiles.

4.4 Realistic Workloads

In Fig. 4, we present experiments similar to those with synthetic instances in Section 4.2, but with the realistic instances.

As we can see when comparing LS and MD, treating the $\frac{n}{2}$ largest tasks first only marginally decreases the makespan of LS. We can also see that when n grows, the absolute error of LPT seems to be on par with the one of SLACK. In [7], SLACK was found to perform generally better than LPT for some synthetic instances, which differs from the results we get with more realistic instances. Finally, LPT seems to converge even faster to the optimal with these instances than with the synthetic ones, as the absolute error quickly becomes close to 0.

5 Conclusion

Given various probability distributions, we have evaluated the performance of four heuristics, among which the classical LPT heuristic and the more recent SLACK one. The literature already contains important theoretical results either in the form of different kinds of stochastic convergence to optimality or with a convergence rate. To the best of our knowledge, this paper is the first to empirically assess the tightness of a theoretical convergence rate for LPT. Furthermore, we focus on a novel definition of uniformity for the cost distribution: for a given total work, any integer composition can be drawn with the same probability, which leads to dependent random costs. This distribution is further enhanced by considering a subset of the decompositions that constrains the minimum cost. This paper proves the convergence in probability of LPT and similar heuristics with these distributions as well. Finally, we empirically analyze the convergence with realistic distributions obtained through traces. All these results contribute to understand the excellent performance of LPT in practice.

Future work will consist in obtaining stronger convergence theoretical results. For instance, existing results only consider that the number of tasks n tends to infinity. The impact of a varying number of processors m could be explored. Also, this work is the first attempt to consider dependent cost distributions, but many such distributions exist and could be explored. For instance, the same application consisting of a given set of tasks can be executed with different input size. The tasks could thus often have the same profile to a given multiplying factor. Finally, the novel distribution in this paper presents a minimum cost. Existing convergence results for independent distributions could probably be extended to consider costs with a similar minimum value. For instance, the worst-case ratio for LPT is achieved with costs $\frac{1}{3}$ and $\frac{1}{2}$. The uniform distribution $\mathcal{U}(\frac{1}{3}, \frac{1}{2})$ could thus present some challenges.

References

1. Behera, D.K., Laha, D.: Comparison of heuristics for identical parallel machine scheduling. Advanced Materials Research **488–489**, 1708–1712 (2012)
2. Benoit, A., Canon, L.C., Elghazi, R., Héam, P.C.: Update on the Asymptotic Optimality of LPT. Research report, Inria (2021), https://lccanon.github.io/report_LPT_conv.pdf
3. Coffman, E.G., Frederickson, G.N., Lueker, G.S.: Probabilistic analysis of the LPT processor scheduling heuristic. In: Deterministic and stochastic scheduling, pp. 319–331. Springer (1982)
4. Coffman, E.G., Garey, M.R., Johnson, D.S.: An application of bin-packing to multiprocessor scheduling. SIAM Journal of Computing **7**, 1–17 (1978)
5. Coffman Jr, E.G., Lueker, G.S., Rinnooy Kan, A.H.G.: Asymptotic methods in the probabilistic analysis of sequencing and packing heuristics. Management Science **34**(3), 266–290 (1988)
6. Cooper, K.D., Schielke, P.J., Subramanian, D.: An Experimental Evaluation of List Scheduling. Tech. Rep. 98-326, Rice Computer Science (1998)
7. Della Croce, F., Scatamacchia, R.: The longest processing time rule for identical parallel machines revisited. Journal of Scheduling **23**(2), 163–176 (2020)
8. Dempster, M.A.H., Fisher, M.L., Jansen, L., Lageweg, B.J., Lenstra, J.K., Rinnooy Kan, A.H.G.: Analysis of heuristics for stochastic programming: results for hierarchical scheduling problems. Mathematics of Op. Research **8**(4), 525–537 (1983)
9. Feitelson, D.G., Tsafrir, D., Krakov, D.: Experience with using the parallel workloads archive. J. of Parallel and Distributed Computing **74**(10), 2967–2982 (2014)
10. Flajolet, P., Sedgewick, R.: Analytic Combinatorics. Cambridge University Press (2009)
11. Flajolet, P., Zimmermann, P., Cutsem, B.V.: A calculus for the random generation of labelled combinatorial structures. Theor. Comput. Sci. **132**(2), 1–35 (1994)
12. Frenk, J.B.G., Rinnooy Kan, A.H.G.: The asymptotic optimality of the LPT rule. Mathematics of Operations Research **12**(2), 241–254 (1987)
13. Frenk, J.B.G., Rinnooy Kan, A.H.G.: The rate of convergence to optimality of the LPT rule. Discrete Applied Mathematics **14**(2), 187–197 (1986)
14. Graham, R.L., Lawler, E.L., Lenstra, J.K., Kan, A.H.G.R.: Optimization and approximation in deterministic sequencing and scheduling: a survey. Annals of Discrete Mathematics **5**, 287–326 (1979)
15. Graham, R.L.: Bounds on multiprocessing timing anomalies. SIAM journal on Applied Mathematics **17**(2), 416–429 (1969)
16. Gupta, J.N.D., Ruiz-Torres, A.J.: A LISTFIT heuristic for minimizing makespan on identical parallel machines. Production Planning & Control **12**(1), 28–36 (2001)
17. Laha, D., Behera, D.K.: A comprehensive review and evaluation of LPT, MULTIFIT, COMBINE and LISTFIT for scheduling identical parallel machines. Int. Journal of Information and Communication Technology (IJICT) **11**(2) (2017)
18. Lee, C.Y., David Massey, J.: Multiprocessor scheduling: combining LPT and MULTIFIT. Discrete Applied Mathematics **20**(3), 233–242 (1988)
19. Loulou, R.: Tight bounds and probabilistic analysis of two heuristics for parallel processor scheduling. Mathematics of Operations Research **9**(1), 142–150 (1984)
20. Michiels, W., Korst, J., Aarts, E., van Leeuwen, J.: Performance Ratios for the Karmarkar-Karp Differencing Method. Electronic Notes in Discrete Mathematics **13**, 71–75 (2003)
21. Piersma, N., Romeijn, H.E.: Parallel machine scheduling: A probabilistic analysis. Naval Research Logistics (NRL) **43**(6), 897–916 (1996)