Scheduling Jobs with Machine-dependent Benefits to Machines with Different Capacity Limits under a Consideration of Fairness

Ling-Chieh Kung*, Chi-Wei Liu, Yi-An Lin, and Ling-Hsuan Chen

Department of Information Management, National Taiwan University, Taiwan

Abstract

We consider a problem of allocating jobs to multiple machines. Each job has an amount of workload and benefits collected upon completion. Job benefits may be different while assigned to different machines, since qualities of machines may differ. Each machine has its capacity and can afford only a certain amount of workloads. Under the no-splitting constraint of jobs and capacity constraint of machines, the problem is to maximize the minimum benefit per unit capacity among all machines. Our problem is thus a capacitated job allocation problem with a consideration of fairness. After showing that this problem is NP-hard, we propose an approximation algorithm modified from the longest processing time (LPT) rule. The algorithm is proved to have a worst-case performance guarantee 1/2 when job benefits are linear to workloads. Different approximation factors are also derived for convex and concave relationships. Finally, numerical studies illustrate the average performances of the algorithm and demonstrates that this algorithm works well when the jobs exhibit economy of scale but not so well when they exhibit diminishing marginal benefits.

Keywords: Scheduling, approximation algorithm, benefit-workload relationship, capacitated machine, fairness.

1 Introduction

Job allocation and scheduling has been widely discussed over decades and applied in many fields (Pinedo, 2012). When one assigns jobs to multiple factories, machines, or agents, jobs bring in workloads as well as benefits upon completion. While traditionally these benefits all go to the

*lckung@ntu.edu.tw; corresponding author.

central decision maker, in many cases an individual factory/machine/agent also care about the benefits generated by itself. The issue of allocation fairness then emerges.

An anonymous leading LED chip manufacturer in Taiwan, who owns around twenty factories in Taiwan and China, faces exactly this issue. The company accepts orders for producing different kinds of LED chips from its customers. Once per month, the company assigns these orders to its factories. Because factory managers are evaluated according to the amount of revenues generated by completing orders (as well as other performance indicators), they indeed would fight for easy-to-complete jobs of high margins. From this perspective, jobs are actually valuable *resources* that should be fairly assigned to factories. Unlike typical resource allocation problems, however, here a factory manager cannot accept too many jobs due to its capacity limitation. Job allocation considering fairness is thus of new challenges.

Some social enterprises nowadays also face a similar problem. With the idea of "giving them jobs, not money," a social enterprise (or sometimes a government) may hire jobless people to do part-time jobs to earn their livings. For example, "The Sock Mob Homeless Volunteer Network" in London finds that many homeless people are familiar with the street and may provide their own perspective to view a city. As a result, it hires and trains homeless people to be London city tour guides.[1] As another example, the magazine company "The Big Issue Taiwan" hires the homeless to sell magazines.[2] For these social enterprises, the objective is not limited to job completion and revenue maximization. Instead, a more important objective is to distribute benefits generated by job completion to help as many people as possible. Again, the first priority is not on *efficiency* but on *fairness*.

In this study, we consider the aforementioned job allocation problem with the fairness issue. For ease of exposition, we will call those working agents as machines. The most important feature of our job allocation problem is that the decision maker considers not only the overall profitability but also fairness among machines. In the environment, each job has an associated workload and different benefits depending on the quality of the machine to which this job was assigned. Each machine has its limited capacity and can afford only a certain amount of total workloads. As long as a machine has enough capacity to complete jobs assigned to it, it will prefer to be assigned more jobs so as to earn more benefits. We assume that all jobs cannot be split; otherwise, the problem can be solved trivially. While we consider fairness as

[1]For more information, please see `http://www.meetup.com/thesockmob/`.

[2]For more information, please see `http://www.bigissue.tw/`.

the decision maker's most critical target, naturally it also should not sacrifice profitability too much. Therefore, we set up the objective function in our problem as to maximize the benefit per unit capacity generated by the machine generating the least benefit per unit capacity. While adopting this objective takes care the performance of the poorest machine, it also intuitively encourages the whole system to complete as many high-benefit jobs as possible.

As our problem is strongly NP-hard, we focus on designing a polynomial-time approximation algorithm.We follow the scheduling literature and modify the famous longest processing time first (LPT) algorithm to propose a greedy algorithm for our problem. In each iteration, we choose the machine with the lowest benefit per unit capacity and find the remaining unassigned jobs which generates the most benefit to that machine. This simple algorithm, which is called capacitated highest benefit first (CHBF), is then proved to possess worst-case performance guarantee in various cases.

The approximation factor critically depends on the relationship between job benefits and workloads. When benefits are linear to workloads, CHBF is shown to be a $\frac{1}{2}$-approximation algorithm.[3] When the relationship is convex (i.e., the marginal benefit increases in workload), the factor is a function depending on the number of jobs, number of machines, and degree of convexity. When the relationship is concave (i.e., the marginal benefit decreases in workload), the factor becomes depending on the machine capacity and degree of concavity. Both factors under the convex and concave scenarios are shown to approach $\frac{1}{2}$ when the convex and concave functions approach a linear one.

Beside our theoretical investigation, we also conduct numerical studies. These numerical studies serve for multiple purposes. First, it demonstrates that CHBF's average-case performance is much better than its worst-case performance guarantee in all situations. Moreover, it delivers insightful managerial implications of CHBF. Numerically we observe that CHBF's average-case performance is better under the convex benefit-workload relationship than under the linear one, which is then better than the concave one. This implies that a decision maker facing our job allocation problem may consider whether to apply CHBF according to the characteristics of its jobs/products. If its production environment is of economy of scale, e.g., due to cost saving through standardized mass production, CHBF would be a good choice. On the

[3]Throughout this paper, we follow the convention of Williamson and Shmoys (2011) and say that an α-approximation algorithm (or an algorithm of factor $\alpha \in (0,1)$) for a maximization problem always attains an objective value z such that $z \geq \alpha z^*$, where z^* is the objective value of an optimal solution.

contrary, if the marginal benefit of producing products is decreasing, which is typical when the market-clearing price decreases as more products are put onto the market, CHBF may be less effective.

In the next section, we will review some relevant literature about job scheduling and allocation, approximation algorithms, and allocation fairness. In Section 3, we will formulate our job allocation problem into an integer program and demonstrate its NP-hardness. The algorithm CHBF will then be introduced in Section 4. More importantly, the worst-case performance guarantees of CHBF will be proved in the same section. We conduct numerical studies in Section 5. Section 6 concludes.

2 Literature Review

All kinds of job allocation and scheduling problems have been widely studied in the literature. A unique feature of our job allocation problem is that each job has two attributes, workload and benefit, which may have various kinds of relationships. A special case of our problem is for machines to be uncapacitated. In this case, job workloads do not matter. If we consider job benefits as processing times, the problem reduces to maximizing minimum completion time on parallel identical machines, which is denoted as $P||C_{\min}$ according to the three-field notation in the scheduling literature (Pinedo, 2012; Walter et al., 2017). Applications of $P||C_{\min}$ include replacement part sequencing (Friesen and Deuermeyer, 1981), regional allocations of investments (Haouari and Jemmali, 2008), public transportation advertisement scheduling (Kuo et al., 2019), among others.

Most of the scheduling problems are NP-hard. By the definition of NP-hardness, unless P = NP, we cannot solve the problem in polynomial time. As a result, some researchers tackle $P||C_{\min}$ by proposing exact algorithms by improving the generic branch-and-bound algorithms (Mokotoff, 2004; Haouari and Jemmali, 2008; Walter et al., 2017). More research are devoted to approximation algorithms or approximation schemes. As the "dual" version of $P||C_{\min}$, makespan minimization is one of the earliest problems for which approximation algorithms are developed (Williamson and Shmoys, 2011). It has been shown by Graham (1966, 1969) that the longest processing time first (LPT) algorithm, a listing algorithm that sorts all jobs by their processing times in the descending order, then assigns jobs by this order once at a time to the currently least loaded machine, has an asymptotic approximation factor $\frac{4}{3}$. Deuermeyer et al.

(1982) show that the same algorithm can be adopted for $P||C_{\min}$ with a factor $\frac{3}{4}$. Csirik et al. (1992) improves the performance guarantee to $\frac{3m-1}{4m-2}$, where m is the number of machines. Walter (2013) propose a similar approximation algorithm called restricted LPT (RLPT) and prove that it has an approximation fact $\frac{1}{m}$. The author also shows that LPT always outperforms RLPT. Alon et al. (1998) prove that under some mild conditions a polynomial time approximation scheme (PTAS) can be constructed. **Popenko et al. (2020) consider three kinds of job scheduling problems on uniform parallel machines, including the makespan minimization, machine covering problem and the distribution of jobs among the machines maximization. Base on their mixed-integer programs, they determine the sufficient conditions of the schedule optimality for their problems. Lee et al. (2022) consider the scheduling problem with m parallel identical machines and want to minimize makespan. Each job's processing time is less than equal to $\frac{1}{k}$ times the optimal makespan, which means the size of job's processing time are small. Base on the assumption, they derive an exact upper bound of approximation ratio of the LPT algorithm with different parameters k and m. Barman and Krishnamurthy (2020) consider the problem of allocating indivisible goods to n agents fairly and want to maximize the smallest benefit of each agent. They shows a approximation algorithm and a round-robin algorithm to solve their problem.** All the above works consider only one attribute of jobs (i.e., processing time), while our work considers two attributes (i.e., workload and benefit).

To the best of our knowledge, currently only two works in the scheduling literature considering multiple attributes of jobs. Ji et al. (2022) consider a problem of parallel-machine scheduling that is very similar to our study. In their problem, each job has a resource consumption requirement, a processing time, and each machine has the same resource capacity limit. The objective is to maximize the minimum machine load, and they propose two approximation algorithms and analyse their worst-case ratios. While the resource consumption and processing time correspond to the workload and benefit in our work, our work is different from theirs because in our problem machines may have different capacity limits and benefit are machine-dependent. Lawrinenko et al. (2018) consider a problem of scheduling n jobs to m parallel machines for maximizing the minimum machine completion time. Moreover, the number of jobs on each machine cannot exceed a machine-dependent cardinality limit. They solve the problem by an effective dynamic programming based improvement procedure. The algorithm can finds the (near-)optimal solutions on a large set of random instances quickly. While the cardinality constraint is similar to

our capacity constraint, it is clear that our setting is a generalization. In addition, benefits in our study are machine-dependent, where the corresponding processing times in their study are identical among all machines.

Sometimes researchers look for *a posteriori* bounds for NP-hard problems. For makespan minimization, Blocher and Sevastyanov (2015) improves the *a posteriori* Coffman-Sethi bound of LPT by considering the maximum number of jobs scheduled on a machine. Massabò et al. (2016) derive a tight *a posteriori* bound of LPT for the two-machine makespan minimization problem. This bound depends on the index of the last job assigned to the critical machine. Following their ideas, the bounds provided in our study are proved by considering the first job that cannot be directly assigned due to the capacity constraints. Nevertheless, for all scenarios we provide *a priori* bounds, not only *a posteriori* ones.

3 Model

We consider a problem of assigning n jobs in set $J = \{1, 2, ..., n\}$ to m machines in set $I = \{1, 2, ..., m\}$. The capacity of machine i is $K_i > 0$, for all $i = 1, ..., m$. For job j assigned to machine i, there is a workload $c_j > 0$ and a benefit $b_{ij} > 0$. That is, machine i has to spend c_j amount of time in order to earn b_{ij} for completing job j. With the assumption that a job cannot be split to multiple machines, we try to assign jobs to machines which maximizes the minimum score among the machines. Here, we define the *score* of a machine as the total benefit of this machine divided by its capacity. More precisely, let

$$x_{ij} = \begin{cases} 1 & \text{if job } j \text{ is assigned to machine } i \\ 0 & \text{otherwise} \end{cases} , i \in I, j \in J,$$

be our decision variables, our job allocation problem can be formulated as

$$\begin{aligned} \max \quad & \min_{i \in I} \left\{ \frac{\sum_{j \in J} b_{ij} x_{ij}}{K_i} \right\} \\ \text{s.t.} \quad & \sum_{j \in J} c_j x_{ij} \leq K_i \quad \forall i \in I \\ & \sum_{i \in I} x_{ij} \leq 1 \quad \forall j \in J \\ & x_{ij} \in \{0, 1\} \quad \forall i \in I, j \in J. \end{aligned} \tag{1}$$

The first constraint ensures that the total workload of jobs assigned to machine i will not exceed its capacity K_i, the second constraint ensures that a job can only be assigned once, and the last

Parameter	
I	The set of machines
J	The set of jobs
n	The number of jobs (in set J)
m	The number of machines (in set I)
K_i	The capacity of machine i ($K_i > 0$)
c_j	The finite workload of job j ($c_j > 0$)
b_{ij}	The finite benefit of job j assigned to machine i ($b_{ij} > 0$)

Decision variable

$$x_{ij} = \begin{cases} 1 & \text{if job } j \text{ is assigned to machine } i \\ 0 & \text{otherwise} \end{cases}, i \in I, j \in J.$$

Table 1: List of notations

constraint guarantees integrality. The objective function is to maximize the lowest score among all machines. Table 1 lists the notations used in this model.

Two special cases of our problem have been shown to be NP-hard. First, when all the machines are uncapacitated, our problem becomes to $P||C_{\min}$, for which many researchers design approximation algorithms for (Alon et al., 1998; Csirik et al., 1992; Deuermeyer et al., 1982; Walter, 2013). Second, when there is only one machine, our problem is exactly a knapsack problem. Using either reduction shows that our problem is NP-hard.

Given a pair of jobs j_1 and j_2, we say that job j_1 is *larger* (respectively, *smaller*) than job j_2 if $c_{j_1} > c_{j_2}$. Without loss of generality, we may assume that $c_1 \geq c_2 \geq \cdots \geq c_n$. If the workload of the smallest job c_n is below 1, we may without loss of generality normalize it to 1 and adjust job workloads and machine capacity proportionally. After the normalization, if machine capacity is less than 2, one will be able to solve the problem in polynomial time (as a machine can only be assigned one job). Therefore, we exclude this uninteresting case and further assume that machine capacity is at least 2. Finally, it is clear that jobs whose workloads are above the machine capacity can be removed without affecting an optimal solution. We formally state these assumptions below.

Assumption 1. *$K_i \geq c_1 \geq c_2 \geq \cdots \geq c_n \geq 1$ and $K_i \geq 2$ for all $i \in I$.*

The difficulty of our problem naturally depends on the relationship between job benefits b_{ij}

and workloads c_j. Intuitively, the problem is more complicated if benefits and workloads are not regulated by any relationship. Fortunately, in practice, benefits generated by a job often correlate with machine quality and job complexity, i.e.,

$$b_{ij} = q_i h c_j^t,$$

where q_i denotes machine quality, h is a scale factor and $t > 0$ elaborates the relationship between benefit and workload. We further consider three special relationships between b_{ij} and c_j: In the *linear* relationship, $b_{ij} = q_i h c_j$ for some $h > 0$; in the *convex* relationship, $b_{ij} = q_i h c_j^t$ for some $h > 0$ and $t > 1$; in the *concave* relationship, $b_{ij} = q_i h c_j^t$ for some $h > 0$ and $t < 1$. These assumptions are formalized below.

Assumption 2. *$b_{ij} = q_i h c_j^t$ for all $i \in I, j \in J$ for some $h > 0$ and $t > 0$.*

Moreover, each machine has its own quality q_i and capacity K_i. If we multiply K_i by q_i, the value of product $K_i q_i$ may differ, and there must be a maximum and a minimum among these products. Without loss of generality, we have following assumption.

Assumption 3. $K_1 q_1 \geq K_2 q_2 \geq \cdots \geq K_m q_m$.

4 Analysis

4.1 Capacitated highest benefit first (CHBF) algorithm

To solve the problem, we propose a new algorithm called CHBF. We want the algorithm to increase our objective value the most every time we assign a job, without violating the capacity constraints. Below we detail the steps.

In each iteration, we try to maximize the objective value, so we choose the machine with the lowest score and find the remaining unassigned jobs which generates the most benefit to that machine. If there are multiple machines with the same lowest score, we choose the one with minimum residual capacity. If the job cannot be assigned to the machine because the residual capacity on that machine is not enough to complete that job, we try the next job that generates the next most benefit to that machine. If all the remaining jobs cannot be assigned to that machine, then we choose the machine with the next lowest score and repeat the process. We reiterate this process until there is no machine with enough capacity to be assigned the remaining jobs. The pseudocode of CHBF is in Algorithm 1. We define $s_i = \frac{\sum_{j \in J} b_{ij} x_{ij}}{K_i}$ as the

score for all machine i, l as the currently lowest score machine and h as the highest benefit job for machine l.

Algorithm 1 CHBF

$J^{'} \leftarrow \emptyset$
repeat
  if $J^{'} = \emptyset$ **then**
    $l \leftarrow$ findLowestMachine(I)
  end if
  $h \leftarrow$ findHighestJob($J \setminus J', l$)
  $J^{'} \leftarrow J^{'} \cup h$
  if $K_l^{'} \geq c_h$ **then**
    $x_{lh} \leftarrow 1$, $K_l^{'} \leftarrow K_l^{'} - c_h$, $J \leftarrow J \setminus h$, $J^{'} \leftarrow \emptyset$
  end if
  if $J^{'} = J$ **then**
    $I \leftarrow I \setminus l$, $J^{'} \leftarrow \emptyset$
  end if
until $I = \emptyset$ **or** $J = \emptyset$
return $\min_{i \in I}\{s_i\}$

Algorithm 2 findLowestMachine(I'')

min $\leftarrow \infty, l \leftarrow 0$
for $i \in I''$ **do**
  if $s_i <$ min **or** $s_i =$ min & $K_i < K_l$ **then**
    min $\leftarrow s_i$, $l \leftarrow i$
  end if
end for
return l

Below we will discuss the CHBF with three different worst-case performance guarantees for three different benefit-workload relationships: linear, convex, or concave. Linear benefit-workload relationship (R1) is for situations that benefits are proportional to workloads. Convex benefit-workload relationship (R2) occurs when jobs exhibit economy of scale, and thus a larger job generates relatively more benefits by the same machine. This is more likely the case if these jobs are manufacturing or production tasks. Finally, concave benefit-workload relationship (R3)

Algorithm 3 findHighestJob(J'', l)

```
max ← 0, h ← 0
for j ∈ J'' do
    if b_il > max then
        max ← b_il, h ← j
    end if
end for
return h
```

models the situation in which the marginal benefit of a machine decreases in a unit of workload increase. This is a typical feature if the resulting products/services are to be sold to the end market. Chapter 4.5 is the time complexity of CHBF.

Relationship	Linear	Convex	Concave
Section	§4.2	§4.3	§4.4

Table 2: Section table

4.2 Linear benefit-workload relationship

We first prove that CHBF is a factor-$\frac{1}{2}$ approximation algorithm when job benefit and workload are proportional in the form $b_{ij} = q_i h c_j$ for some $h > 0$. Our algorithm can generate a solution whose objective value is at least $\frac{1}{2}$ of that from an optimal solution. We denote s^* as the objective value of an optimal solution and s^{A} as that of the CHBF solution.

Theorem 1. *If $b_{ij} = q_i h c_j$ for all $i \in I, j \in J$ for some given $h > 0$,*

$$s^{\mathrm{A}} \geq \frac{1}{2} s^*.$$

Proof. First, let $q_{\min}$ denotes the minimum of q_i for all $i \in I$. We know that an upper bound of s^* is $\min\left\{\frac{q_1 h K_1}{K_1}, \frac{q_2 h K_2}{K_2}, \cdots, \frac{q_m h K_m}{K_m}\right\} = q_{\min} h$, as all machines are assigned one job whose workload equals to its capacity. This bound is restricted to the lowest quality machine. If we can prove that $s^{\mathrm{A}} \geq \frac{1}{2} q_{\min} h$, we will have $s^{\mathrm{A}} \geq \frac{1}{2} q_{\min} h \geq \frac{1}{2} s^*$, and the proof is complete. We prove by contradiction, suppose that $s^{\mathrm{A}} < \frac{1}{2} q_{\min} h$, and job j is the first job not assigned to its first-priority machine, say, machine i, the one with the lowest score currently. The current score of machine i is also lower than $\frac{1}{2} q_{\min} h$ at the time of assigning job j, and the cumulative

benefit of machine i is then lower than $\frac{1}{2}q_{\min}hK_i$. This implies that $b_{ij} < \frac{1}{2}q_{\min}hK_i$, since every job assigned before job j are with higher benefit, the cumulative benefit of machine i is also greater than b_{ij}. Therefore, $c_j < \frac{q_{\min}}{2q_i}K_i \leq \frac{K_i}{2}$ according to $b_{ij} = q_i h c_j$ and that $q_{\min}$ is the minimum of q_i for all $i \in I$. However, if the cumulative benefit of machine i must also be lower than $\frac{1}{2}q_{\min}hK_i$, the total workloads of jobs that have been assigned to machine i must be less than $\frac{K_i}{2}$ as well. This means the residual capacity of machine i is greater than $\frac{K_i}{2}$, and there is a room to assign job j. This violates our assumption that job j cannot be assigned to machine i. With this contradiction, s^{A} must be greater or equal to half of the optimal s^*. □

The following example shows that the bound $\frac{1}{2}$ is tight. Assume ϵ be a small positive number, $q_i = 1$ for all i, $h = 1$, $K_i = K$ for all i, where K is a constant, and we have $2m$ jobs with workload $\frac{1}{2}K$ and 1 job with workload $\frac{1}{2}K + \epsilon$. In this case, $b_{ij} = c_j$ for all i and j. CHBF will result in $m - 1$ machines being allocated 2 jobs but one machine leaving with only 1 job. The machine with only 1 job will earn score $s^{\mathrm{A}} = \frac{1}{2} + \frac{\epsilon}{K}$. However, the optimal solution is to allocate each machine with 2 jobs by ignoring the largest workload job. This results in $s^* = 1$. As a result, $\frac{s^{\mathrm{A}}}{s^*} = \frac{\frac{1}{2}+\frac{\epsilon}{K}}{1}$, which approaches $\frac{1}{2}$ as ϵ approaches 0.

4.3 Convex benefit-workload relationship

Next, we prove that the CHBF has a performance guarantee when job benefits are convex in workloads in the form $b_{ij} = q_i h c_j^t$ for some $h > 0$ and $t > 1$. We define $\beta_{\mathrm{H}} \in (0, 1]$ according to the largest-workload job and smallest-capacity machine as

$$\beta_{\mathrm{H}} = \frac{c_1}{K_m},$$

Recall that K is defined by

$$K_1 q_1 \geq K_2 q_2 \geq \cdots \geq K_m q_m.$$

We rely on two lemmas to build our main result. The first lemma shows the upper bound of the optimal value, while the second lemma shows the lower bound of the CHBF. Combines these two algorithm enables us to come up with the worst-case performance guarantee of the algorithm.

Lemma 1. *If $b_{ij} = q_i h c_j^t$ for all $i \in I, j \in J$ for some given $h > 0$ and $t > 1$, we have*

$$s^* \leq \beta_{\mathrm{H}}^{t-1} q_m h K_m^{t-1}.$$

Proof. Intuitively, we know that $s^* \leq \min\left\{\frac{q_1 h K_1^t}{K_1}, \frac{q_2 h K_2^t}{K_2}, \cdots, \frac{q_m h K_m^t}{K_m}\right\} = q_m h K_m^{t-1}$, where all machines are fully loaded with one job whose workload equals its capacity. This is clearly an upper bound. When $\beta_{\mathrm{H}} = 1$, we can easily understand that $s^* \leq q_m h K_m^{t-1} = \beta_{\mathrm{H}}^{t-1} q_m h K_m^{t-1}$.

When $0 < \beta_{\mathrm{H}} < 1$, the statement is still correct. For any machine, we know that the best way to consume machine capacity is to fill it by large workloads so it earns higher cumulative benefits according to $b_{ij} = q_i h c_j^t$. If all machines are filled with the same workloads, the final score will be restricted by the smallest-capacity machine. By our notation, the largest job's workload is $c_1 = \beta_{\mathrm{H}} K_m$. In this case, the final score will not be greater than assigning $\frac{1}{\beta_{\mathrm{H}}}$ jobs with the largest workload c_1 to the least capacitated machine. Thus, an upper bound of s^* is $\frac{\frac{1}{\beta_{\mathrm{H}}} \beta_{\mathrm{H}}^t q_m h K_m^t}{K_m} = \beta_{\mathrm{H}}^{t-1} q_m h K_m^{t-1}$. □

Lemma 2. *If $b_{ij} = q_i h c_j^t$ for all $i \in I, j \in J$ for some given $h > 0$ and $t > 1$, we have*

$$s^{\mathrm{A}} \geq \min\left\{\frac{1}{2^t}, \frac{n-m-1}{(n-m)^t}\right\} q_m h K_m^{t-1}.$$

Proof. Suppose that job j is the first job that cannot be assigned to its first-priority machine, say, machine i, the one with the lowest score currently. We define p as the number of jobs that have been assigned to that minimum-benefit machine. For ease of exposition, let $\alpha = \min\{\frac{1}{2^t}, \frac{n-m-1}{(n-m)^t}\}$ be the bound to be proved.

By contradiction, we assume that $s^{\mathrm{A}} < \alpha q_m h K_m^{t-1} \leq \alpha q_i h K_i^{t-1}$. The cumulative benefit on machine i is thus less than $\alpha q_i h K_i^t$. We know that the job j cannot be assigned to machine i, which has already been assigned p jobs by CHBF, we have $b_j \leq \frac{\alpha q_i h K_i^t}{p}$, where the equality holds if and only if all the p jobs and job j are equally large. As $b_{ij} = q_i h c_j^t$, this implies that $c_j \leq \sqrt[t]{\frac{\alpha}{p}} K_i$. Therefore, the consumed capacity on machine i is at least $K_i - \sqrt[t]{\frac{\alpha}{p}} K_i$, otherwise machine i would have enough capacity for job j. Since benefits of jobs are now convex in their workloads, the least possible cumulative benefit of machine i is for the p jobs to be equally large. In this case, the cumulative benefit is $ph(\frac{K_i - \sqrt[t]{\frac{\alpha}{p}} K_i}{p})^t$.

We now show $ph(\frac{K_i - \sqrt[t]{\frac{\alpha}{p}} K_i}{p})^t \geq \alpha q_i h K_i^t$ to establish a contradiction. Some arithmetic shows that $ph(\frac{K_i - \sqrt[t]{\frac{\alpha}{p}} K_i}{p})^t \geq \alpha q_i h K_i^t$ if and only if $\alpha \leq \frac{p}{(p+1)^t}$. As the function $\frac{p}{(p+1)^t}$ is quasi-concave in p for any $t > 1$, and the smallest and largest possible numbers of jobs on machine i is 1 and $n - m - 1$, it implies that the global minimum of $\frac{p}{(p+1)^t}$ is either 1 or $n - m - 1$. Thus, $\alpha = \min\{\frac{1}{2^t}, \frac{n-m-1}{(n-m)^t}\} \leq \frac{p}{(p+1)^t}$ will always be satisfied. The proof is then complete. □

The lower bound of s^{A} is the minimum of two values. Whether $\frac{1}{2^t}$ or $\frac{n-m-1}{(n-m)^t}$ will be the

smaller one depends on t. It can be easily verified that if $t \geq 2$, we have $\frac{n-m-1}{(n-m)^t} < \frac{1}{2^t}$ because $\frac{p}{(p+1)^t}$ is decreasing in that case. However, as long as $t < 2$, $\frac{p}{(p+1)^t}$ is increasing at $p = 1$, and it is possible that $\frac{n-m-1}{(n-m)^t} > \frac{1}{2^t}$. Note that when t approaches 1, eventually $\frac{1}{2^t}$ will be the smaller one (as long as $n - m > 1$), and this bound converges to $\frac{1}{2}$ as Theorem 1 suggests for the linear benefit-workload relationship (where $t = 1$).

Our main result, Theorem 2, is a direct combination of the above two lemmas.

Theorem 2. *If $b_{ij} = q_i h c_j^t$ for all $i \in I, j \in J$ for some given $h > 0$ and $t > 1$, we have*

$$s^{\mathrm{A}} \geq \frac{\min\{\frac{1}{2^t}, \frac{n-m-1}{(n-m)^t}\}}{\beta_{\mathrm{H}}^{t-1}} s^*.$$

Proof. According to Lemma 2, we have $s^{\mathrm{A}} \geq \min\{\frac{1}{2^t}, \frac{n-m-1}{(n-m)^t}\} q_m h K_m^t$. Also, we know that $s^* \leq h \beta_{\mathrm{H}}^{t-1} K_m^t$ by Lemma 1. Thus, the proof can be completed by combining all the two lemmas.

The following example shows that the bound $\frac{\min\{\frac{1}{2^t}, \frac{n-m-1}{(n-m)^t}\}}{\beta_{\mathrm{H}}^{t-1}}$ is tight. First, we set $K_i = K$ for all i, $b_j = c_j^t$ for all j, and the number of jobs and their workloads just like what we did in the linear case. The CHBF algorithm will result in $m - 1$ machines being allocated 2 jobs but one machine leaving with only 1 job. The machine with only 1 job will earn score $s^{\mathrm{A}} = \frac{(\frac{K}{2}+\epsilon)^t}{K}$. However, the optimal solution is to allocate each machine with 2 jobs by ignoring the largest workload job, which results in $s^* = \frac{2(\frac{K}{2})^t}{K}$. As a result, $\frac{s^{\mathrm{A}}}{s^*} = \frac{(\frac{K}{2}+\epsilon)^t}{2(\frac{K}{2})^t}$, which approaches $\frac{1}{2}$ as ϵ approaches 0. In this case, $\frac{\min\{\frac{1}{2^t}, \frac{2m+1-m-1}{(2m+1-m)^t}\}}{\left(\frac{\frac{K}{2}+\epsilon}{K}\right)^{t-1}} = \frac{\frac{1}{2^t}}{\left(\frac{\frac{K}{2}+\epsilon}{K}\right)^{t-1}}$, which also approaches $\frac{1}{2}$ as ϵ approaches 0. □

4.4 Concave benefit-workload

Here we prove the performance guarantee when the benefit is concave in workload in the form $b_{ij} = q_i h c_j^t$ for all $j \in J$ for some $h > 0$ and $t < 1$.

Theorem 3. *If $b_{ij} = q_i h c_j^t$ for all $i \in I, j \in J$ for some given $h > 0$ and $t > 1$, we have*

$$s^{\mathrm{A}} \geq \frac{K_{\min}^{t-1}}{2^t} s^*,$$

where $K_{\min}$ is the minimum of K_i for all $i \in I$.

Proof. The same as the proof in Theorem 1 and Theorem 2. First, we establish an upper bound for s^*. When the relationship between job benefits and workloads are concave, the most

beneficial way to consume all the capacity of machine i is to use K_i jobs with unit workload. If all machines are assigned jobs of workload 1 until there is no more capacity, we will have $\min\left\{\frac{K_1 q_1 h 1^t}{K_1}, \frac{K_2 q_2 h 1^t}{K_2}, \cdots, \frac{K_m q_m h 1^t}{K_m}\right\} = q_{\min} h$ as the objective value, which is clearly an upper bound of s^*.

Then, we consider the moment of that the CHBF algorithm fails to assign a job to its first-priority machine. Let that job be job j and that machine be machine i. We prove by contradiction, by assuming that $s^{\mathrm{A}} < \frac{K_{\min}^{t-1}}{2^t} s^* \leq \frac{K_{\min}^{t-1}}{2^t} q_{\min} h$. This implies that at this moment, machine i is with the lowest score, which is also lower than $\frac{K_{\min}^{t-1}}{2^t} q_{\min} h$. Thus, the cumulative benefit of this machine i is lower than $\frac{K_{\min}^{t-1}}{2^t} q_{\min} h K_i$, and therefore the job benefit b_{ij} should be less than that as well (otherwise, it should have been assigned by CHBF already). As a result, we know $c_j < \sqrt[t]{\frac{K_i K_{\min}^{t-1}}{2^t} \frac{q_{\min}}{q_i}} \leq \frac{K_i}{2} \sqrt[t]{\frac{q_{\min}}{q_i}} \leq \frac{K_i}{2}$, and the residual capacity of machine i is less than $K_i - \frac{K_i}{2} = \frac{K_i}{2}$ (otherwise, there would be a room to assigned job j to machine i). For machine i, however, if its current benefit is lower than $\frac{K_{\min}^{t-1}}{2^t} q_{\min} h K_i$, then it must be also lower than $\frac{K_i^t}{2^t} q_i h$. This implies that the currently occupied capacity must be at most $\frac{K_i}{2}$ (which happens when machine i is assigned only one job), and the residual capacity is above $\frac{K_i}{2}$, which contradicts with the fact derived above. Therefore, we have $s^{\mathrm{A}} \geq \frac{K_{\min}^{t-1}}{2^t} s^*$. □

Note that $t < 1$ and $K \geq 2$ imply that $\frac{K^{t-1}}{2^t} = \frac{1}{2^t K^{1-t}} \leq \frac{1}{2^t \cdot 2^{1-t}} = \frac{1}{2}$, which means that the performance guarantee is no greater than $\frac{1}{2}$. Moreover, as t approaches 1, the guarantee approaches $\frac{1}{2}$. This indicates that Theorem 3 actually extends Theorem 1 from $t = 1$ to $t \leq 1$.

4.5 Time complexity analysis

In this section, we briefly derive the time complexity of CHBF. In each iteration, CHBF finds the lowest score machine and the job benefits this machine the most. That is, scanning through m machines and n jobs. We repeat the iteration n times so that all jobs are assigned. The time complexity of CHBF is then $O(mn^2)$. Numerical results of time are provided in Section 5.5.

5 Numerical Study

5.1 Experiment settings

In our numerical study, we check both solution performance and time performance of CHBF on this fair job allocation problem by running experiments on a personal computer with Windows

10, 8GB RAM and Intel i7-6770 3.4GHz CPU. In all experiments, we set $c_j \sim U(0, 100)$, i.e., c_j is a randomly chosen real number between 0 and 100. We set up several factors to see how the solution varies in different settings. The first factor is the relationship between job benefits and workloads. We consider four scenarios, in which the job benefit is linear in, non-decreasing and convex in, non-decreasing and concave in, and unrelated with the job workload. The second factor is machine quality. We consider two scenarios, where all machines are with the same quality, or there will be a variation. The third factor is capacity tightness. We consider two scenarios, one with loose machine capacity, and one with tight machine capacity. The fourth factor is capacity variation. We consider two scenarios, where all machines are with the same capacity, or there will be a variation. We adopt the following settings for each factor:

- Benefit-workload relationship (labelled as T): scenario L (for linear): $b_{ij} = q_i c_j$; scenario X (for convex): $b_{ij} = q_i c_j^2$; scenario A (for concave): $b_{ij} = q_i \sqrt{c_j}$; scenario R (for unrelated): $b_{ij} \sim U(0, 100)$.
- Machine quality (labelled as Q): scenario I (for identical): $q_i = 1$; scenario D (for diverse): $q_i \sim U(0.8, 1.2)$.
- Capacity tightness (labelled as C): scenario L (for loose): $K_i = a_i \left(\frac{\sum_{j \in J} c_j}{m} \right)$; scenario T (for tight): $K_i = \frac{3}{4} a_i \left(\frac{\sum_{j \in J} c_j}{m} \right)$, where a_i are defined in the next factor.
- Capacity variation (labelled as A): scenario I (for identical) $a_i = 1$; scenario D (for diverse): $a_i \sim U(0.8, 1.2)$.

For the comparison of solution quality, we combine the above factors with several m and n ($m = 5$ with $n = 25,\ 50,\ 150$ and $m = 20$ with $n = 100,\ 200,\ 600$), as a total of 192 scenarios, each with 100 instances. For the comparison of computation time, we randomly select 100 instances for each different problem scales ($m = 5$ with $n = 20,\ 40,\ 60, ...,\ 400$ and $n = 400$ with $n = 5,\ 10,\ 15, ...,\ 40$).

5.2 Benchmark algorithms

We compare CHBF with IP or LP and genetic algorithm to see its performance. An IP solution is an optimal solution to our problem; we solve it with AMPL using the solver CPLEX. However, due to problem complexity and memory limitation, only one scenario in our study is IP solvable: $m = 5$ with $n = 20$ for computation time. When we are unable to get an IP solution, we release the binary constraint of our problem and solve an LP relaxation.

For genetic algorithm, we implement it as follows. A chromosome is a list of n machine IDs, where the jth machine ID is the ID of the machine that will process job j. If a job is not processed by any machine, we denote its "assigned machine ID" by 0. In the initialization phase, we randomly create a pool of 100 feasible solutions. In each iteration we choose two pairs of parents (four chromosomes) from the best half of our pool and one pair of parents (two chromosomes) from the other half. Then we perform a crossover on each pair of parents by randomly select a cross-point which divides the selected solutions into the head part and tail part. Six child solutions are then created by connecting the head part to the another tail part between each pair of parents. All child solutions are given a 1% chance to mutate. When mutation happens, one job will change its destined machine randomly. We then check the feasibility of child solutions. At the end of each iteration, we compare feasible child solutions with those in the pool and replace the lower ones so that the pool remain 100 feasible solutions. This step makes the solutions in the pool better and better after iterations. The above process will be repeated 2000 times. At the end, the algorithm will report the best solution in the pool.

5.3 Efficiency problem

While solving the fair allocation problem in (1), we further investigate on how much efficiency is sacrificed when we maximize fairness. To see this, we consider the efficiency problem by replacing the objective function in (1) by

$$\max \quad \sum_{i\in I}\sum_{j\in J} b_{ij}x_{ij},$$

which is the total benefits generated by all machines. We call our main problem in (1) the *fairness problem* and the one with the alternative objective function the *efficiency problem*. We denote the solution to our algorithm as x_{ij}^{A}, the solution to genetic algorithm as x_{ij}^{G}, the optimal solution to the fairness problem as x_{ij}^{F}, and the optimal solution to the efficiency problem as x_{ij}^{E}. Let

$$s^{\mathrm{A}} = \min_{i\in I}\left\{\frac{\sum_{j\in J} b_{ij}x_{ij}^{\mathrm{A}}}{K_i}\right\}, \quad s^{\mathrm{G}} = \min_{i\in I}\left\{\frac{\sum_{j\in J} b_{ij}x_{ij}^{\mathrm{G}}}{K_i}\right\}, \quad s^{*} = \min_{i\in I}\left\{\frac{\sum_{j\in J} b_{ij}x_{ij}^{\mathrm{F}}}{K_i}\right\},$$

$$\text{and} \quad z^{*} = \sum_{i\in I}\sum_{j\in J} b_{ij}x_{ij}^{\mathrm{E}}.$$

By plugging x_{ij}^{A} and x_{ij}^{G} separately into the objective function of the efficiency problem, we get $z^{\mathrm{A}} = \sum_{i\in I}\sum_{j\in J} b_{ij}x_{ij}^{\mathrm{A}}$ and $z^{\mathrm{G}} = \sum_{i\in I}\sum_{j\in J} b_{ij}x_{ij}^{\mathrm{G}}$, respectively. Besides, if the scenario is not

IP solvable, we will solve an LP relaxation, as we mentioned in 5.2. In these cases, the objective values of the fairness problem and the efficiency problem will be denoted as s^L and z^L. We will report how the objective value of our algorithm deviates from a fairest solution by calculating $\frac{s^{\mathrm{A}}}{s^*}$ or $\frac{s^{\mathrm{A}}}{s^L}$ as well as the comparison of total benefits $\frac{z^{\mathrm{A}}}{s^*}$ or $\frac{z^{\mathrm{A}}}{s^L}$. We will also report how genetic algorithm performs by calculating $\frac{s^{\mathrm{G}}}{s^*}$ or $\frac{s^{\mathrm{G}}}{s^L}$, and $\frac{z^{\mathrm{G}}}{s^*}$ or $\frac{z^{\mathrm{G}}}{s^L}$, to see if our algorithm is effective.

5.4 Comparison of solution quality

To see how the solution quality is, we compare the solution between LP, CHBF and GA. They are then compared to generate Tables 10-15 in Appendix. For the remaining, we use LP solutions because IP solutions are not attainable. To understand how each factor affects the performance of CHBF, we calculate the average performance of each scenario and then generate Tables 3, 4, 5, and 6. Moreover, Table 7 shows how CHBF is affected by different m and n.

In Table 3, in the fairness problem, CHBF and GA perform the best when benefits are linear in workloads. For the CHBF algorithm, the performance falls when it comes to convex and unrelated relationship between benefits and workloads, and performs the least desirable in concave relationship. The reason behind this observation could be the way CHBF assign jobs, it chooses jobs based on jobs' benefit. If benefits are linear in workloads, the value of each job tends to be the same. It means that it is easier for CHBF to assign jobs to achieve fairness and efficiency at the same time. When benefits are convex in workload, high-benefit jobs and low-benefit jobs tend to have relatively similar workloads, and thus machine capacity does not introduce a huge difficulty to the performance of CHBF. If the relationship between benefit and workload is concave, job benefits are lower when workloads are higher. CHBF puts the low cost-performance jobs first to the currently inferior machine, that might be why it performs worse than in linear and convex. However, for genetic algorithm, concave relationship is in its second best, while convex and unrelated relationship get the third and fourth, respectively.

In the efficiency problem, CHBF ranks benefit-workload relationship in the same sequence as that of fairness version. For genetic algorithm, the result is slightly different from that in fairness version, GA now performs better in concave than convex. In any case, data shows that CHBF performs relatively well than GA in both fairness problem and efficiency problem. We can also see that while pursuing fairness, efficiency remains in a high level and did not sacrifice too much using the CHBF algorithm.

Note that the convexity or concavity of the relationship between benefit and workload

has important managerial implication. When the relationship is convex, basically the problem environment is of significant economy of scale so that marginal benefit increases as workloads increase. On the contrary, when the relationship is concave, the product is of diminishing marginal benefit as workloads increase. This understanding will help managers to decide whether to choose the CHBF algorithm as their solution tool when they face the fair job allocation problem studied in this paper.

T	$\frac{s^{\mathrm{A}}}{s^{\mathrm{L}}}$	$\frac{s^{\mathrm{G}}}{s^{\mathrm{L}}}$	$\frac{z^{\mathrm{A}}}{z^{\mathrm{L}}}$	$\frac{z^{\mathrm{G}}}{z^{\mathrm{L}}}$
L	0.992	0.982	0.994	0.979
X	0.932	0.807	0.971	0.879
A	0.812	0.888	0.896	0.944
R	0.931	0.766	0.991	0.905

Table 3: Impact of the benefit-workload relationship (T)

Table 4 shows that CHBF has a better performance when machine qualities are the same, which is when the conversion rate is stable among machines. This will be helpful for CHBF to put the highest benefit job in the lowest score machine in each iteration, because it consumes a constant capacity no matter assigned to which machine.

Q	$\frac{s^{\mathrm{A}}}{s^{\mathrm{L}}}$	$\frac{s^{\mathrm{G}}}{s^{\mathrm{L}}}$	$\frac{z^{\mathrm{A}}}{z^{\mathrm{L}}}$	$\frac{z^{\mathrm{G}}}{z^{\mathrm{L}}}$
I	0.944	0.866	0.969	0.935
D	0.890	0.855	0.957	0.919

Table 4: Impact of machine quality (Q)

Table 5 shows that CHBF performs better when the capacity is loose comparing to tight. This is an intuitive result, because jobs can be assigned more appropriately by CHBF when it is more flexible to assign jobs.

C	$\frac{s^{\mathrm{A}}}{s^{\mathrm{L}}}$	$\frac{s^{\mathrm{G}}}{s^{\mathrm{L}}}$	$\frac{z^{\mathrm{A}}}{z^{\mathrm{L}}}$	$\frac{z^{\mathrm{G}}}{z^{\mathrm{L}}}$
L	0.945	0.899	0.985	0.965
T	0.888	0.822	0.941	0.889

Table 5: Impact of capacity tightness (C)

Table 6 is the results of capacity variation. The data reports that there is no big difference if there is a variation of capacity among machines or not.

Furthermore, by examining Table 7, we find that the performance ratios of CHBF is higher

A	$\frac{s^{\mathrm{A}}}{s^{\mathrm{L}}}$	$\frac{s^{\mathrm{G}}}{s^{\mathrm{L}}}$	$\frac{z^{\mathrm{A}}}{z^{\mathrm{L}}}$	$\frac{z^{\mathrm{G}}}{z^{\mathrm{L}}}$
I	0.917	0.860	0.963	0.962
D	0.917	0.861	0.963	0.928

Table 6: Impact of capacity variance (A)

when $\frac{n}{m}$ increases. We believe that with more options of job candidates to be selected, our algorithm can assign more valuable jobs to machines.

$\frac{n}{m}$	$\frac{s^{\mathrm{A}}}{s^{\mathrm{L}}}$	$\frac{s^{\mathrm{G}}}{s^{\mathrm{L}}}$	$\frac{z^{\mathrm{A}}}{z^{\mathrm{L}}}$	$\frac{z^{\mathrm{G}}}{z^{\mathrm{L}}}$
5	0.896	0.820	0.957	0.910
10	0.922	0.867	0.966	0.931
30	0.932	0.896	0.967	0.940

Table 7: Impact of number of jobs and machines ($\frac{n}{m}$)

In summary, the average performance of all instances of CHBF is larger than 0.9, which is better than the average performance of GA. Numerical experiments show that CHBF is a good algorithm for this problem with high robustness. In particular, CHBF performs better when job benefits are linear or convex to workloads, machine qualities are stable, machine capacities are loose, and $\frac{n}{m}$ is large. These results will suggest decision makers when to apply the CHBF algorithm is suitable.

5.5 Comparison of computation time

We compare the average computation time for different problem scales in Tables 8 and 9. Figures 1 and 2 shows that the numerical experiments fit our analysis of time complexity in Section 4.5. With a fixed m, when n becomes larger, the computation time of CHBF increases in a quadratic manner. On the other hand, with a fixed n, when m becomes larger, the computation time of CHBF increases linearly. Moreover, we compared with CHBF and LP in Figures 3 and 4 and see that our algorithm runs fast while LP spends much longer time in solving when n and m are larger.

n	IP	LP	CHBF	GA
20	55503.3	2.0	0.6	1372.5
40	-	3.8	1.2	2373.0
60	-	7.2	1.7	3427.3
80	-	12.7	2.2	4450.7
100	-	16.7	3.2	5418.3
120	-	26.1	4.2	6505.0
140	-	31.5	4.7	7441.4
160	-	35.2	5.9	8425.3
180	-	46.3	6.8	9572.8
200	-	52.5	8.2	10536.4
220	-	59.6	10.2	11743.7
240	-	60.3	10.5	12750.1
260	-	72.4	12.2	13884.4
280	-	81.7	13.9	14907.3
300	-	96.4	15.1	15741.2
320	-	107.9	16.6	17012.4
340	-	126.5	17.9	18429.6
360	-	133.6	19.7	19206.7
380	-	150.7	21.5	20269.4
400	-	160.6	23.3	21052.2

Table 8: Computation time (milliseconds) with fixed $m = 5$

m	LP	CHBF	GA
5	160.6	23.3	21052.2
10	548.0	39.0	21276.6
15	953.9	51.1	21232.9
20	1655.2	69.6	21059.2
25	2606.2	83.9	21368.3
30	3875.1	104.5	21182.0
35	5500.3	117.8	21379.0
40	7894.7	135.6	21686.0

Table 9: Computation time (milliseconds) with fixed $n = 400$

6 Conclusions

In this study, we consider a job allocation problem with allocation fairness as the main focus. We formulate the problem as an integer program to maximize the lowest benefit among all machines

subject to capacity constraints. Meanwhile, the problem also take the machine quality and the job complexity into consideration, meaning that the benefit may alter from workload of jobs to the different machines. By modifying the classic LPT rule, which is known to be effective for the uncapacitated version of our problem, we develop our own algorithm CHBF. We then prove the performance guarantee is $\frac{1}{2}$ when the job benefit is linear to workload. For convex and concave relationships, we also establish different performance guarantees depending on the number of jobs, number of machines, degree of convexity (or concavity), machine capacity, machine quality, and the job complexity. Their worst-case performance guarantee is also $\frac{1}{2}$. The average-case performance is demonstrated to be much better than the worst-case performance guarantee in each scenario through numerical studies. Moreover, according to the numerical studies, CHBF is more appropriate than the genetic algorithm on almost all scenario, the benefit-workload relationship, the quality of machines, the different number of jobs and machines, and the capacity tightness and variance.

Some further investigations may further improve this study. One promising direction is to look for better worst-case performance guarantees of our problem, either for the general problem or under some conditions. Another direction is to consider a different algorithm. For example, as we have pinpointed the reason for CHBF to be not so effective when job benefit is convex to workload, we may want to sort jobs in the descending order of their benefit-workload ratio. This calls for further investigation.

References

Alon, N., Y. Azar, G.J. Woeginger, T. Yadid. 1998. Approximation schemes for scheduling on parallel machines. *Journal of Scheduling* **1**(1) 55–66.

Barman, S., S. K. Krishnamurthy. 2020. Approximation algorithms for maximin fair division. *Association for Computing Machinery* **8**(1) 1–28.

Blocher, J.D., S. Sevastyanov. 2015. A note on the Coffman-Sethi bound for LPT scheduling. *Journal of Scheduling* **18**(3) 325–327.

Csirik, J., H. Kellerer, G. Woeginger. 1992. The exact LPT-bound for maximizing the minimum completion time. *Operations Research Letters* **11**(5) 281–287.

Deuermeyer, B.L., D.K. Friesen, M.A. Langston. 1982. Scheduling to maximize the minimum

processor finish time in a multiprocessor system. *SIAM Journal on Algebraic Discrete Methods* **3**(2) 190–196.

Friesen, D.K., B.L. Deuermeyer. 1981. Analysis of greedy solutions for a replacement part sequencing problem. *Mathematics of Operations Research* **6**(1) 74—-87.

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

Graham, R.L. 1969. Bounds on multiprocessing timing anomalies. *SIAM Journal on Applied Mathematics* **17**(2) 416–429.

Haouari, M., M. Jemmali. 2008. Maximizing the minimum completion time on parallel machines. *4OR: A Quarterly Journal of Operations Research* **6**(4) 375—392.

Ji, M., S. Hu, Y. Zhang, T. C. E. Cheng, Y. Jiang. 2022. Parallel-machine scheduling with identical machine resource capacity limits and dejong's learning effect. *International Journal of Production Research* **60**(9) 2753–2765.

Kuo, Yun-Hsin, Fa-Xuan Xiao, Cheng-Wei Lu, Chia-Hua Chang, Ling-Chieh Kung. 2019. Public transportation advertisement scheduling: algorithms and a case study in taiwan. *Proceedings of the 2019 Pacific Asia Conference on Information Systems (PACIS)*.

Lawrinenko, A., S. Schwerdfeger, R. Walter. 2018. Reduction criteria, upper bounds, and a dynamic programming based heuristic for the max–min k_i-partitioning problem. *Journal of Heuristics* **24**(2) 173–203.

Lee, M., K. Lee, M. Pinedo. 2022. Tight approximation bounds for the lpt rule applied to identical parallel machines with small jobs. *Journal of Scheduling* **25**(6) 721–740.

Massabò, I., G. Paletta, A.J. Ruiz-Torresh. 2016. A note on longest processing time algorithms for the two uniform parallel machine makespan minimization problem. *Journal of Scheduling* **19**(2) 207–211.

Mokotoff, E. 2004. An exact algorithm for the identical parallel machine scheduling problem. *European Journal of Operational Research* **152**(3) 758–769.

Pinedo, M. 2012. *Scheduling Theory, Algorithms, and Systems*. Springer, Berlin, Germany.

Popenko, V., M. Sperkach, O. Zhdanova, Zbigniew Kokosiński. 2020. On optimality conditions for job scheduling on uniform parallel machines. *Advances in Computer Science for Engineering and Education II*. 103–112.

Walter, R. 2013. Comparing the minimum completion times of two longest-first scheduling-heuristics. *Central European Journal of Operations Research* **21**(1) 125—139.

Walter, R., M. Wirth, A. Lawrinenko. 2017. Improved approaches to the exact solution of the machine covering problem. *Journal of Scheduling* **20**(3) 147—164.

Williamson, D.P., D.B. Shmoys. 2011. *The Design of Approximation Algorithms*. Cambridge University Press, London, UK.

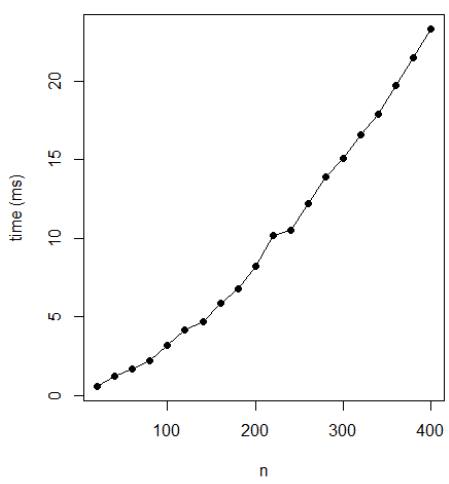

Figure 1: Computation time (ms) of CHBF with fixed $m = 5$

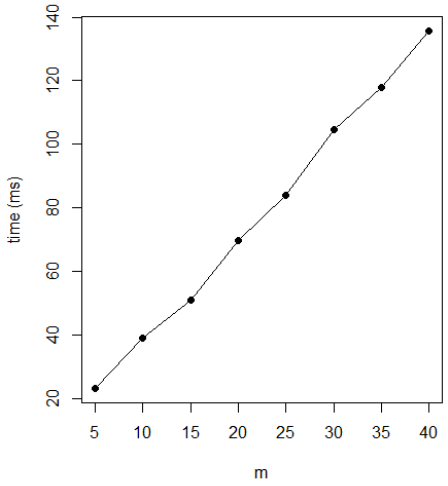

Figure 2: Computation time (ms) of CHBF with fixed $n = 400$

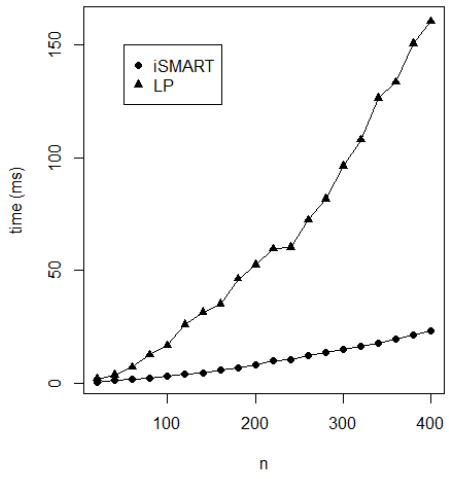

Figure 3: Computation time (ms) of CHBF and LP with fixed $m = 5$

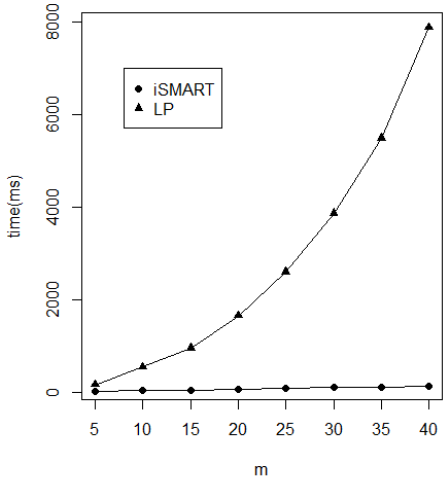

Figure 4: Computation time (ms) of CHBF and LP with fixed $n = 400$

Appendix: the complete computational results

						average				minimum			
m	n	T	Q	C	A	$\frac{s^{\mathrm{A}}}{s^{\mathrm{L}}}$	$\frac{s^{\mathrm{G}}}{s^{\mathrm{L}}}$	$\frac{z^{\mathrm{A}}}{z^{\mathrm{L}}}$	$\frac{z^{\mathrm{G}}}{z^{\mathrm{L}}}$	$\frac{s^{\mathrm{A}}}{s^{\mathrm{L}}}$	$\frac{s^{\mathrm{G}}}{s^{\mathrm{L}}}$	$\frac{z^{\mathrm{A}}}{z^{\mathrm{L}}}$	$\frac{z^{\mathrm{G}}}{z^{\mathrm{L}}}$
5	25	L	I	L	I	0.957	0.937	0.972	0.953	0.902	0.882	0.927	0.909
5	25	L	I	L	D	0.975	0.960	0.988	0.974	0.927	0.897	0.953	0.914
5	25	L	I	T	I	0.964	0.962	0.977	0.976	0.882	0.929	0.916	0.945
5	25	L	I	T	D	0.970	0.963	0.984	0.975	0.902	0.899	0.938	0.910
5	25	L	D	L	I	0.992	0.993	0.981	0.937	0.958	0.947	0.912	0.874
5	25	L	D	L	D	0.990	0.994	0.986	0.962	0.961	0.951	0.945	0.884
5	25	L	D	T	I	0.989	0.997	0.979	0.955	0.950	0.969	0.929	0.889
5	25	L	D	T	D	0.988	0.997	0.981	0.957	0.942	0.974	0.933	0.892
5	25	A	I	L	I	0.914	0.914	0.958	0.949	0.861	0.890	0.908	0.887
5	25	A	I	L	D	0.927	0.935	0.965	0.968	0.791	0.870	0.868	0.916
5	25	A	I	T	I	0.827	0.890	0.854	0.935	0.709	0.823	0.776	0.874
5	25	A	I	T	D	0.798	0.897	0.842	0.936	0.711	0.850	0.751	0.866
5	25	A	D	L	I	0.812	0.906	0.955	0.921	0.669	0.858	0.928	0.862
5	25	A	D	L	D	0.842	0.920	0.947	0.936	0.723	0.852	0.860	0.867
5	25	A	D	T	I	0.738	0.881	0.839	0.905	0.568	0.819	0.743	0.854
5	25	A	D	T	D	0.735	0.883	0.820	0.906	0.619	0.806	0.756	0.854
5	25	X	I	L	I	0.967	0.885	0.991	0.939	0.894	0.712	0.970	0.825
5	25	X	I	L	D	0.965	0.888	0.993	0.948	0.885	0.743	0.955	0.790
5	25	X	I	T	I	0.890	0.732	0.934	0.799	0.757	0.502	0.854	0.668
5	25	X	I	T	D	0.894	0.736	0.947	0.797	0.796	0.532	0.888	0.623
5	25	X	D	L	I	0.913	0.873	0.959	0.899	0.833	0.756	0.934	0.805
5	25	X	D	L	D	0.919	0.886	0.959	0.907	0.777	0.702	0.925	0.741
5	25	X	D	T	I	0.836	0.734	0.918	0.779	0.705	0.564	0.828	0.656
5	25	X	D	T	D	0.844	0.737	0.935	0.778	0.719	0.606	0.862	0.641
5	25	R	I	L	I	0.937	0.872	0.990	0.962	0.805	0.716	0.969	0.841
5	25	R	I	L	D	0.928	0.879	0.991	0.959	0.765	0.739	0.964	0.829
5	25	R	I	T	I	0.863	0.756	0.969	0.848	0.675	0.605	0.916	0.699
5	25	R	I	T	D	0.871	0.760	0.969	0.853	0.722	0.558	0.930	0.672
5	25	R	D	L	I	0.925	0.877	0.991	0.958	0.743	0.729	0.962	0.902
5	25	R	D	L	D	0.920	0.875	0.991	0.956	0.700	0.741	0.958	0.832
5	25	R	D	T	I	0.867	0.758	0.971	0.851	0.658	0.613	0.938	0.704
5	25	R	D	T	D	0.855	0.758	0.968	0.847	0.595	0.610	0.922	0.653

Table 10: The average and worst-case performance of CHBF and GA - $m = 5, n = 25$

m	n	T	Q	C	A	average				minimum			
						$\frac{s^{\mathrm{A}}}{s^{\mathrm{L}}}$	$\frac{s^{\mathrm{G}}}{s^{\mathrm{L}}}$	$\frac{z^{\mathrm{A}}}{z^{\mathrm{L}}}$	$\frac{z^{\mathrm{G}}}{z^{\mathrm{L}}}$	$\frac{s^{\mathrm{A}}}{s^{\mathrm{L}}}$	$\frac{s^{\mathrm{G}}}{s^{\mathrm{L}}}$	$\frac{z^{\mathrm{A}}}{z^{\mathrm{L}}}$	$\frac{z^{\mathrm{G}}}{z^{\mathrm{L}}}$
5	50	L	I	L	I	0.990	0.967	0.993	0.975	0.969	0.945	0.974	0.957
5	50	L	I	L	D	0.994	0.983	0.998	0.991	0.985	0.951	0.990	0.956
5	50	L	I	T	I	0.989	0.988	0.993	0.993	0.951	0.969	0.977	0.978
5	50	L	I	T	D	0.993	0.987	0.996	0.992	0.985	0.970	0.990	0.977
5	50	L	D	L	I	0.998	0.999	0.995	0.968	0.993	0.973	0.986	0.925
5	50	L	D	L	D	0.998	0.998	0.993	0.977	0.986	0.974	0.964	0.937
5	50	L	D	T	I	0.997	0.999	0.996	0.980	0.987	0.991	0.983	0.908
5	50	L	D	T	D	0.998	0.999	0.996	0.981	0.984	0.996	0.990	0.928
5	50	A	I	L	I	0.969	0.956	0.985	0.975	0.950	0.936	0.965	0.937
5	50	A	I	L	D	0.969	0.967	0.982	0.988	0.897	0.938	0.903	0.958
5	50	A	I	T	I	0.810	0.923	0.827	0.946	0.780	0.897	0.783	0.913
5	50	A	I	T	D	0.813	0.920	0.830	0.944	0.725	0.874	0.759	0.892
5	50	A	D	L	I	0.814	0.930	0.975	0.946	0.709	0.875	0.958	0.914
5	50	A	D	L	D	0.834	0.941	0.954	0.952	0.707	0.853	0.861	0.916
5	50	A	D	T	I	0.713	0.897	0.810	0.914	0.608	0.843	0.761	0.855
5	50	A	D	T	D	0.717	0.896	0.808	0.914	0.606	0.837	0.745	0.854
5	50	X	I	L	I	0.996	0.936	0.999	0.970	0.989	0.895	0.992	0.926
5	50	X	I	L	D	0.993	0.933	0.998	0.965	0.966	0.838	0.988	0.888
5	50	X	I	T	I	0.971	0.796	0.980	0.830	0.937	0.715	0.947	0.738
5	50	X	I	T	D	0.964	0.791	0.980	0.827	0.938	0.673	0.959	0.715
5	50	X	D	L	I	0.947	0.918	0.964	0.932	0.870	0.865	0.940	0.877
5	50	X	D	L	D	0.945	0.921	0.961	0.929	0.849	0.828	0.927	0.806
5	50	X	D	T	I	0.898	0.781	0.964	0.807	0.818	0.685	0.943	0.696
5	50	X	D	T	D	0.887	0.785	0.964	0.810	0.825	0.678	0.939	0.690
5	50	R	I	L	I	0.967	0.923	0.998	0.978	0.877	0.866	0.991	0.918
5	50	R	I	L	D	0.966	0.922	0.998	0.974	0.764	0.824	0.989	0.898
5	50	R	I	T	I	0.921	0.780	0.987	0.839	0.792	0.614	0.960	0.677
5	50	R	I	T	D	0.926	0.781	0.988	0.844	0.722	0.620	0.896	0.720
5	50	R	D	L	I	0.968	0.919	0.998	0.978	0.879	0.855	0.992	0.929
5	50	R	D	L	D	0.969	0.927	0.998	0.978	0.885	0.814	0.988	0.901
5	50	R	D	T	I	0.922	0.791	0.988	0.851	0.737	0.697	0.969	0.735
5	50	R	D	T	D	0.916	0.787	0.988	0.850	0.746	0.655	0.955	0.740

Table 11: The average and worst-case performance of CHBF and GA - $m = 5, n = 50$

						average				minimum			
m	n	T	Q	C	A	$\frac{s^{\mathrm{A}}}{s^{\mathrm{L}}}$	$\frac{s^{\mathrm{G}}}{s^{\mathrm{L}}}$	$\frac{z^{\mathrm{A}}}{z^{\mathrm{L}}}$	$\frac{z^{\mathrm{G}}}{z^{\mathrm{L}}}$	$\frac{s^{\mathrm{A}}}{s^{\mathrm{L}}}$	$\frac{s^{\mathrm{G}}}{s^{\mathrm{L}}}$	$\frac{z^{\mathrm{A}}}{z^{\mathrm{L}}}$	$\frac{z^{\mathrm{G}}}{z^{\mathrm{L}}}$
5	150	L	I	L	I	0.999	0.990	0.999	0.993	0.996	0.982	0.997	0.987
5	150	L	I	L	D	0.999	0.995	1.000	0.998	0.998	0.985	0.999	0.988
5	150	L	I	T	I	0.999	0.998	0.999	0.999	0.996	0.996	0.998	0.997
5	150	L	I	T	D	0.999	0.998	1.000	0.999	0.998	0.996	0.999	0.997
5	150	L	D	L	I	1.000	1.000	1.000	0.990	0.999	0.999	0.999	0.976
5	150	L	D	L	D	1.000	0.999	0.995	0.992	0.998	0.983	0.966	0.969
5	150	L	D	T	I	1.000	1.000	1.000	0.997	0.999	1.000	0.999	0.981
5	150	L	D	T	D	1.000	1.000	1.000	0.997	0.998	0.999	0.999	0.993
5	150	A	I	L	I	0.994	0.982	0.997	0.992	0.989	0.972	0.992	0.986
5	150	A	I	L	D	0.965	0.983	0.970	0.993	0.878	0.961	0.888	0.978
5	150	A	I	T	I	0.809	0.931	0.812	0.944	0.778	0.905	0.783	0.916
5	150	A	I	T	D	0.806	0.930	0.811	0.942	0.754	0.889	0.761	0.899
5	150	A	D	L	I	0.804	0.928	0.984	0.965	0.694	0.851	0.974	0.947
5	150	A	D	L	D	0.816	0.940	0.949	0.965	0.673	0.843	0.855	0.938
5	150	A	D	T	I	0.693	0.882	0.795	0.918	0.590	0.796	0.763	0.876
5	150	A	D	T	D	0.691	0.878	0.796	0.917	0.594	0.798	0.747	0.862
5	150	X	I	L	I	1.000	0.972	1.000	0.990	0.999	0.956	1.000	0.976
5	150	X	I	L	D	0.999	0.963	1.000	0.979	0.995	0.884	0.997	0.906
5	150	X	I	T	I	0.993	0.830	0.994	0.849	0.988	0.791	0.990	0.810
5	150	X	I	T	D	0.991	0.831	0.994	0.848	0.986	0.782	0.990	0.798
5	150	X	D	L	I	0.952	0.933	0.963	0.953	0.903	0.871	0.933	0.918
5	150	X	D	L	D	0.958	0.936	0.962	0.945	0.884	0.859	0.914	0.896
5	150	X	D	T	I	0.910	0.805	0.976	0.830	0.854	0.751	0.963	0.781
5	150	X	D	T	D	0.906	0.804	0.975	0.829	0.849	0.725	0.959	0.753
5	150	R	I	L	I	0.989	0.961	1.000	0.993	0.958	0.937	0.999	0.981
5	150	R	I	L	D	0.990	0.955	1.000	0.983	0.950	0.849	0.997	0.887
5	150	R	I	T	I	0.952	0.788	0.992	0.824	0.862	0.694	0.980	0.726
5	150	R	I	T	D	0.951	0.786	0.992	0.823	0.863	0.691	0.969	0.731
5	150	R	D	L	I	0.990	0.961	1.000	0.993	0.965	0.926	0.999	0.976
5	150	R	D	L	D	0.988	0.954	1.000	0.984	0.941	0.850	0.998	0.898
5	150	R	D	T	I	0.952	0.786	0.991	0.821	0.870	0.729	0.979	0.764
5	150	R	D	T	D	0.952	0.789	0.992	0.828	0.894	0.709	0.972	0.757

Table 12: The average and worst-case performance of CHBF and GA - $m = 5, n = 150$

m	n	T	Q	C	A	average				minimum			
						$\frac{s^{\mathrm{A}}}{s^{\mathrm{L}}}$	$\frac{s^{\mathrm{G}}}{s^{\mathrm{L}}}$	$\frac{z^{\mathrm{A}}}{z^{\mathrm{L}}}$	$\frac{z^{\mathrm{G}}}{z^{\mathrm{L}}}$	$\frac{s^{\mathrm{A}}}{s^{\mathrm{L}}}$	$\frac{s^{\mathrm{G}}}{s^{\mathrm{L}}}$	$\frac{z^{\mathrm{A}}}{z^{\mathrm{L}}}$	$\frac{z^{\mathrm{G}}}{z^{\mathrm{L}}}$
20	100	L	I	L	I	0.965	0.902	0.978	0.953	0.936	0.806	0.951	0.902
20	100	L	I	L	D	0.982	0.905	0.993	0.960	0.958	0.852	0.976	0.917
20	100	L	I	T	I	0.961	0.928	0.977	0.967	0.918	0.880	0.945	0.933
20	100	L	I	T	D	0.984	0.930	0.993	0.969	0.963	0.895	0.983	0.945
20	100	L	D	L	I	0.998	0.980	0.994	0.942	0.993	0.925	0.979	0.909
20	100	L	D	L	D	0.998	0.982	0.993	0.951	0.979	0.934	0.975	0.897
20	100	L	D	T	I	0.997	0.992	0.992	0.957	0.987	0.968	0.961	0.926
20	100	L	D	T	D	0.997	0.992	0.994	0.959	0.984	0.965	0.983	0.935
20	100	A	I	L	I	0.916	0.854	0.963	0.953	0.883	0.811	0.934	0.930
20	100	A	I	L	D	0.935	0.856	0.981	0.959	0.860	0.816	0.932	0.924
20	100	A	I	T	I	0.832	0.829	0.867	0.932	0.725	0.785	0.789	0.901
20	100	A	I	T	D	0.785	0.824	0.840	0.932	0.747	0.791	0.800	0.899
20	100	A	D	L	I	0.769	0.810	0.967	0.923	0.716	0.770	0.951	0.895
20	100	A	D	L	D	0.772	0.815	0.958	0.927	0.679	0.759	0.902	0.889
20	100	A	D	T	I	0.701	0.784	0.847	0.901	0.587	0.742	0.790	0.870
20	100	A	D	T	D	0.678	0.786	0.819	0.901	0.603	0.738	0.758	0.870
20	100	X	I	L	I	0.981	0.774	0.995	0.931	0.946	0.704	0.976	0.889
20	100	X	I	L	D	0.975	0.768	0.997	0.927	0.943	0.652	0.985	0.855
20	100	X	I	T	I	0.905	0.629	0.931	0.786	0.821	0.533	0.905	0.722
20	100	X	I	T	D	0.917	0.639	0.966	0.792	0.853	0.548	0.932	0.714
20	100	X	D	L	I	0.887	0.752	0.956	0.892	0.801	0.654	0.947	0.834
20	100	X	D	L	D	0.882	0.749	0.955	0.893	0.810	0.659	0.940	0.832
20	100	X	D	T	I	0.809	0.619	0.918	0.769	0.724	0.532	0.888	0.710
20	100	X	D	T	D	0.801	0.611	0.946	0.763	0.716	0.527	0.930	0.678
20	100	R	I	L	I	0.922	0.649	0.996	0.960	0.807	0.541	0.991	0.915
20	100	R	I	L	D	0.925	0.651	0.996	0.959	0.801	0.560	0.987	0.868
20	100	R	I	T	I	0.847	0.530	0.982	0.836	0.624	0.428	0.968	0.756
20	100	R	I	T	D	0.852	0.524	0.982	0.831	0.652	0.417	0.963	0.739
20	100	R	D	L	I	0.928	0.658	0.996	0.958	0.809	0.576	0.988	0.922
20	100	R	D	L	D	0.926	0.649	0.995	0.954	0.750	0.547	0.982	0.880
20	100	R	D	T	I	0.847	0.522	0.982	0.834	0.645	0.411	0.967	0.765
20	100	R	D	T	D	0.854	0.530	0.982	0.835	0.642	0.396	0.969	0.739

Table 13: The average and worst-case performance of CHBF and GA - $m = 20, n = 100$

						average				minimum			
m	n	T	Q	C	A	$\frac{s^{\mathrm{A}}}{s^{\mathrm{L}}}$	$\frac{s^{\mathrm{G}}}{s^{\mathrm{L}}}$	$\frac{z^{\mathrm{A}}}{z^{\mathrm{L}}}$	$\frac{z^{\mathrm{G}}}{z^{\mathrm{L}}}$	$\frac{s^{\mathrm{A}}}{s^{\mathrm{L}}}$	$\frac{s^{\mathrm{G}}}{s^{\mathrm{L}}}$	$\frac{z^{\mathrm{A}}}{z^{\mathrm{L}}}$	$\frac{z^{\mathrm{G}}}{z^{\mathrm{L}}}$
20	200	L	I	L	I	0.992	0.953	0.995	0.979	0.984	0.930	0.991	0.964
20	200	L	I	L	D	0.997	0.960	0.999	0.988	0.987	0.890	0.992	0.967
20	200	L	I	T	I	0.992	0.977	0.995	0.990	0.977	0.966	0.985	0.979
20	200	L	I	T	D	0.997	0.977	0.999	0.990	0.990	0.962	0.997	0.984
20	200	L	D	L	I	1.000	0.998	0.999	0.973	0.997	0.984	0.997	0.955
20	200	L	D	L	D	0.999	0.997	0.996	0.980	0.998	0.965	0.980	0.960
20	200	L	D	T	I	0.999	0.999	0.999	0.985	0.998	0.994	0.997	0.971
20	200	L	D	T	D	0.999	0.999	0.999	0.985	0.996	0.994	0.997	0.969
20	200	A	I	L	I	0.971	0.910	0.988	0.977	0.960	0.894	0.974	0.967
20	200	A	I	L	D	0.969	0.914	0.986	0.984	0.910	0.884	0.926	0.969
20	200	A	I	T	I	0.801	0.865	0.818	0.941	0.785	0.846	0.797	0.926
20	200	A	I	T	D	0.800	0.870	0.824	0.942	0.769	0.835	0.787	0.918
20	200	A	D	L	I	0.752	0.843	0.977	0.948	0.689	0.812	0.970	0.921
20	200	A	D	L	D	0.766	0.853	0.962	0.953	0.667	0.817	0.904	0.927
20	200	A	D	T	I	0.662	0.812	0.798	0.913	0.628	0.772	0.775	0.885
20	200	A	D	T	D	0.670	0.816	0.803	0.915	0.603	0.769	0.764	0.889
20	200	X	I	L	I	0.998	0.864	0.999	0.968	0.994	0.825	0.998	0.946
20	200	X	I	L	D	0.995	0.859	0.999	0.971	0.985	0.808	0.996	0.928
20	200	X	I	T	I	0.977	0.720	0.984	0.825	0.960	0.665	0.964	0.789
20	200	X	I	T	D	0.968	0.718	0.985	0.826	0.948	0.665	0.979	0.783
20	200	X	D	L	I	0.919	0.819	0.955	0.927	0.875	0.784	0.939	0.907
20	200	X	D	L	D	0.912	0.817	0.955	0.928	0.856	0.757	0.932	0.878
20	200	X	D	T	I	0.852	0.689	0.966	0.802	0.803	0.631	0.953	0.758
20	200	X	D	T	D	0.852	0.692	0.964	0.806	0.810	0.633	0.957	0.756
20	200	R	I	L	I	0.959	0.757	0.999	0.982	0.866	0.696	0.998	0.966
20	200	R	I	L	D	0.959	0.755	0.999	0.977	0.869	0.679	0.996	0.938
20	200	R	I	T	I	0.896	0.620	0.993	0.836	0.773	0.545	0.985	0.795
20	200	R	I	T	D	0.908	0.614	0.992	0.834	0.727	0.535	0.984	0.765
20	200	R	D	L	I	0.957	0.756	0.999	0.981	0.878	0.691	0.997	0.963
20	200	R	D	L	D	0.961	0.759	0.999	0.979	0.842	0.677	0.996	0.934
20	200	R	D	T	I	0.902	0.623	0.992	0.837	0.768	0.558	0.981	0.785
20	200	R	D	T	D	0.904	0.610	0.992	0.832	0.748	0.507	0.983	0.753

Table 14: The average and worst-case performance of CHBF and GA - $m = 20, n = 200$

m	n	T	Q	C	A	average				minimum			
						$\frac{s^{\mathrm{A}}}{s^{\mathrm{L}}}$	$\frac{s^{\mathrm{G}}}{s^{\mathrm{L}}}$	$\frac{z^{\mathrm{A}}}{z^{\mathrm{L}}}$	$\frac{z^{\mathrm{G}}}{z^{\mathrm{L}}}$	$\frac{s^{\mathrm{A}}}{s^{\mathrm{L}}}$	$\frac{s^{\mathrm{G}}}{s^{\mathrm{L}}}$	$\frac{z^{\mathrm{A}}}{z^{\mathrm{L}}}$	$\frac{z^{\mathrm{G}}}{z^{\mathrm{L}}}$
20	600	L	I	L	I	0.999	0.988	0.999	0.995	0.998	0.976	0.937	0.991
20	600	L	I	L	D	1.000	0.978	1.000	0.998	0.999	0.886	1.000	0.993
20	600	L	I	T	I	0.999	0.997	1.000	0.999	0.998	0.995	0.999	0.998
20	600	L	I	T	D	1.000	0.997	1.000	0.999	0.999	0.994	1.000	0.997
20	600	L	D	L	I	1.000	1.000	1.000	0.992	1.000	0.998	1.000	0.985
20	600	L	D	L	D	1.000	0.999	0.996	0.994	1.000	0.972	0.978	0.980
20	600	L	D	T	I	1.000	1.000	1.000	0.998	1.000	1.000	1.000	0.996
20	600	L	D	T	D	1.000	1.000	1.000	0.998	1.000	1.000	1.000	0.995
20	600	A	I	L	I	0.994	0.956	0.998	0.993	0.991	0.947	0.995	0.989
20	600	A	I	L	D	0.977	0.951	0.983	0.995	0.908	0.897	0.915	0.984
20	600	A	I	T	I	0.809	0.900	0.812	0.941	0.797	0.887	0.800	0.926
20	600	A	I	T	D	0.805	0.900	0.812	0.941	0.786	0.875	0.792	0.920
20	600	A	D	L	I	0.743	0.860	0.982	0.966	0.678	0.816	0.979	0.957
20	600	A	D	L	D	0.748	0.862	0.963	0.966	0.685	0.826	0.897	0.956
20	600	A	D	T	I	0.650	0.812	0.790	0.914	0.596	0.771	0.776	0.901
20	600	A	D	T	D	0.652	0.812	0.789	0.913	0.606	0.754	0.767	0.894
20	600	X	I	L	I	1.000	0.931	1.000	0.991	1.000	0.915	1.000	0.983
20	600	X	I	L	D	0.999	0.918	1.000	0.989	0.997	0.857	0.999	0.953
20	600	X	I	T	I	0.994	0.787	0.995	0.848	0.991	0.755	0.991	0.820
20	600	X	I	T	D	0.991	0.789	0.995	0.849	0.987	0.746	0.993	0.821
20	600	X	D	L	I	0.928	0.859	0.956	0.953	0.879	0.821	0.946	0.944
20	600	X	D	L	D	0.927	0.852	0.955	0.948	0.890	0.812	0.940	0.913
20	600	X	D	T	I	0.871	0.734	0.973	0.829	0.824	0.695	0.965	0.809
20	600	X	D	T	D	0.870	0.734	0.973	0.831	0.837	0.690	0.964	0.800
20	600	R	I	L	I	0.983	0.863	1.000	0.995	0.913	0.825	1.000	0.988
20	600	R	I	L	D	0.984	0.855	1.000	0.989	0.927	0.799	0.999	0.951
20	600	R	I	T	I	0.930	0.694	0.991	0.818	0.840	0.638	0.986	0.773
20	600	R	I	T	D	0.936	0.696	0.992	0.820	0.842	0.642	0.983	0.766
20	600	R	D	L	I	0.986	0.861	1.000	0.995	0.942	0.833	1.000	0.988
20	600	R	D	L	D	0.984	0.858	1.000	0.992	0.923	0.803	0.999	0.947
20	600	R	D	T	I	0.935	0.697	0.992	0.819	0.836	0.645	0.983	0.778
20	600	R	D	T	D	0.930	0.695	0.992	0.821	0.822	0.646	0.982	0.767

Table 15: The average and worst-case performance of CHBF and GA - $m = 20, n = 600$