Existence Theorems, Lower Bounds and Algorithms for Scheduling to Meet Two Objectives

April Rasala* Cliff Stein† Eric Torng‡ Patchrawat Uthaisombut §

Abstract

We give general results about the existence of schedules which simultaneously minimize two criteria. Our results are general in that (i) they apply to any scheduling environment and (ii) they apply to all pairs of metrics in which the first metric is one of maximum flow time, makespan, or maximum lateness and the second metric is one of average flow time, average completion time, average lateness, or number of on-time jobs. For most of the pairs of metrics we consider, we show the existence of near-optimal schedules for both metrics as well as some lower bound results. For some pairs of metrics such as (maximum flow time, average weighted flow time) and (maximum flow time, number of on-time jobs), we prove negative results on the ability to approximate both criteria within a constant factor of optimal. For many other criteria we present lower bounds that match or approach our bicriterion existence results.

1 Introduction

For years, scheduling algorithms have been designed to optimize many optimality criteria in a wide variety of scheduling models[15, 6, 12, 19]. Two features are common to an overwhelming majority of this scheduling research. First, algorithms are designed for a particular job-machine environment (e.g. one machine and jobs with release dates or parallel machines with precedence constraints). Second, algorithms are designed to minimize one objective (e.g. makespan, average completion time, or minimum lateness), without explicitly considering other metrics simultaneously.

There has been some research on scheduling to simultaneously optimize two criteria[24, 10, 17, 16, 8, 9, 21, 22, 11]. These works all choose two particular criteria and focus on designing algorithms which work well with respect to these criteria. Similarly, there have been a few works that try to characterize good schedules that apply to a variety of environments[14, 23, 1]. (See Section 2 for details.)

1.1 Our Results

In this paper, we give very general results about the existence of schedules which simultaneously minimize two criteria. Our results are general in the ways described above. First, they apply to "any" scheduling environment. Second, they apply to all pairs of metrics in which the first metric is one of maximum flow time, makespan, or maximum lateness and the second metric is one of average flow time, average completion time, average lateness, or number of on-time jobs. We will show that for almost all pairs of objectives, there exist schedules that are simultaneously within a small constant factor of optimal for both objectives. This generalizes the work of Stein and Wein [23] and Aslam et al [1], who gave results only for the problem of simultaneously minimizing makespan and average completion time. We also give lower bounds on the simultaneous approximability of these metrics. A summary of the results appears in Tables 1 and 2. A precise definition of "any" appears in Section 2. Roughly, our model captures almost all of the standard combinatorial scheduling environments.

Upper bounds. We give upper bounds on simultaneously approximating two objectives in Table 1. For all pairs of objectives, except for two involving maximum flow time, we show the existence of schedules which simultaneously optimize both criteria. The techniques used to obtain these results are a combination of extensions of previous techniques along with some new ideas. For several results, we use the framework developed by

*MIT Laboratory for Computer Science, Cambridge, MA 02139. arasala@theory.lcs.mit.edu. Supported by a Lucent Technologies GRPW fellowship. Part of this work was done while at Dartmouth College.

†Department of IEOR, Columbia University, New York, NY 10027. cliff@ieor.columbia.edu. Research partially supported by NSF Career Award CCR-9624828, NSF Grant EIA-98-02068, NSF Grant DMI-9970063 and an Alfred P. Sloan Foundation Fellowship.

‡Dept. of Computer Science and Engineering, 3115 Engineering Building, Michigan State University, East Lansing, MI 48824-1226. torng@cse.msu.edu. FAX: (517) 432-1061. Supported in part by NSF grant CCR-9701679.

§Department of Computer Science University of Pittsburgh, Pittsburgh, PA 15260. utp@cs.pitt.edu. Part of thie work was done while at Michigan State University.

Aslam et al. [1], in which we map schedules optimizing a minsum objective to probability density functions. By using continuous probability density functions, finding an upper bound on the existence of bicriterion schedules can be viewed as finding the probability density function that maximizes a particular analytic expression involving minimums, maximums and integrals. Our work demonstrates that this technique of mapping schedules to probability density functions is more general than originally thought. For other pairs of metrics, we introduce new techniques. For example, in Section 4 we consider the problem of simultaneously minimizing the makespan and average weighted flow time of a schedule. Typically, techniques that work well for approximately minimizing the average completion time of a schedule do not extend to approximately minimizing the average flow time. However, by demonstrating a relationship between the optimal makespan schedule and the optimal average flow time schedule, we prove the existence of schedules that are simultaneously a constant factor approximation for both criteria.

Lower bounds. We also give lower bounds on the simultaneous approximability of these metrics. Many of these lower bounds match the upper bounds presented. Furthermore, the techniques used to prove the lower bounds demonstrate another use of probability density functions. A summary of the lower bound results appear in Table 2. These lower bounds have two important implications. First, they demonstrate that many of the upper bound analyses are tight, or close-to-tight. Second, they demonstrate that in order to provide stronger upper bounds on bicriterion approximation, we must consider more specific scheduling models in which the lower bounds no longer hold.

Algorithms. The techniques used to prove the existence of schedules that are simultaneously a good approximation for two criteria extend, in a natural way, to providing bicriterion algorithms for problems where good approximation algorithms are known for both single criterion problems. We discuss this in Section 6. In addition we present an optimal algorithm for the specific case of scheduling a set of jobs with release dates to simultaneously minimize the makespan and average completion time of the resulting schedule.

1.2 Previous Work

Some previous research has focused on scheduling to simultaneously optimize two criteria[24, 10, 17, 16, 8, 9, 21, 22, 11]. These works all choose two particular criteria and focus on designing algorithms which work well with respect to these criteria. Similarly, there have been several papers that try to characterize good schedules that apply to a variety of environments[14, 23, 1]. (See Section 2 for details.) This paper, however, is the first to simultaneously consider many pairs of metrics and many environments simultaneously.

1.3 Motivation

Often, the "objective" of a particular real-world scheduling problem is best expressed as the minimization of several expressions. Suppose that the resources being scheduled are a group of people with a variety of skills and multiple bosses. This group of people wants to keep all the bosses happy by completing requests as promptly as possible and also wants to insure that the total time needed to complete all requests will not exceed the end of the work day. They are concerned with both the average completion time of all the jobs received over the course of the day and also with the total length of time it will take to complete the set of jobs.

In the interests of space, we have omitted the details of most of the results which are claimed in Tables 1 and 2. Complete details appear in the undergraduate thesis of the first author [20].

2 Background

2.1 Notation and Optimality Criteria

Very generally, this paper deals with scheduling a set J of n jobs on a set of m machines. Each job $j \in J$ requires some non-negative processing time, p_j, on one of the machines. In addition, a job j may have an associated *release date* r_j, *deadline* d_j or *weight* w_j. The release date is the time that the job becomes available; it cannot be processed before the release date. The deadline is a time by which we would like the job to complete, and a weight is an input parameter that assigns a relative importance to a job.

We define a *valid schedule* S to be an assignment of jobs to machines over time such that each job j starts no earlier than its release date r_j and runs for p_j units of time. A job may or may not complete before its deadline d_j. For a schedule S^x we will denote the *completion time* of job j in S^x as C_j^x. We also define three values related to the completion time. The *flow time* F_j^x of job j in schedule S^x is the amount of time that passes between when job j is released and when it is processed, i.e. $F_j^x = C_j^x - r_j$. The *lateness* of a job L_j^x is the amount of time between its deadline and completion time, and $L_j^x = C_j^x - d_j$. Finally we define an indicator variable U_j^x which denotes whether a job is late, so $U_j^x = 1$ if $C_j > d_j$ and 0 otherwise.

There are two natural types of objective functions that can be defined on these parameters. The first type are called *minmax* objectives. In these we measure the maximum value achieved over all jobs. So for any $O \in \{C, F, L\}$, and for a schedule S^x, we define

		β			
	(α,β)	ACT $\sum w_jC_j$	AFT $\sum w_jF_j$	AL $\sum w_jL_j$	$\sum(1-U_j)$
α	Makespan ($C_{\max}$)	$(1+\rho, \frac{e^\rho}{e^\rho-1})$[1]	$(2+\rho, \frac{e^\rho}{e^\rho-1})$	$(1+\rho, \frac{e^\rho}{e^\rho-1})$	$(2,2)$
	Maximum Flow Time($F_{\max}$)	$(1+\rho, 2+\frac{1}{\rho})$	NO	$(1+\rho, 2+\frac{1}{\rho})$	NO
	Maximum Lateness ($L_{\max}$)	$(1+\rho, \frac{e^\rho}{e^\rho-1})$	$(2+\rho, \frac{e^\rho}{e^\rho-1})$	$(1+\rho, \frac{e^\rho}{e^\rho-1})$	$(2,2)$

Table 1: Existence results for many criteria for any $\rho \in [0,1]$. All results expect the first are new.

		β			
	(α,β)	ACT $\sum w_jC_j$	AFT $\sum w_jF_j$	AL $\sum w_jL_j$	$\sum(1-U_j)$
α	Makespan ($C_{\max}$)	$(1+\rho, \frac{e^\rho}{e^\rho-1})$	$(1+\rho, \frac{e^\rho}{e^\rho-1})$	$(1+\rho, \frac{e^\rho}{e^\rho-1})$	$(2,2)$
	Max Flow Time ($F_{\max}$)	?	$(\frac{\sqrt{N}}{4}, \frac{\sqrt{N}}{4})$	?	$(N^{\frac{1}{4}}, N^{\frac{1}{4}})$
	Max Lateness ($L_{\max}$)	$(1+\rho, \frac{e^\rho}{e^\rho-1})$	$(1+\rho, \frac{e^\rho}{e^\rho-1})$	$(1+\rho, \frac{e^\rho}{e^\rho-1})$	$(2,2)$

Table 2: Lower bounds on bicriterion scheduling for any $\rho \in [0,1]$. All the results are new.

$O^x_{\max} = \max_j O^x_j$. For any particular instance, we denote the objective value of the optimal schedule as

$$O^*_{\max} = \min_{\text{schedules } S^x} O^x_{\max}.$$

The second type of objectives are called *minsum* objectives. In this case, for any $O \in \{C, F, L, U\}$, and for a schedule S^x, we define $O^x_{\text{sum}} = \sum_j w_j O^x_j$. For any particular instance, we denote the objective value of the optimal schedule as

$$O^*_{\text{sum}} = \min_{\text{schedules } S^x} O^x_{\text{sum}}.$$

For bicriterion optimization, Stein and Wein [23] introduced the following notation. Suppose we have criteria (A, B), where A is a minmax criteria and B is a minsum criteria. Then we say that S^x is an (α,β)-schedule if S^x is simultaneously at most an α-approximation for A and a β-approximation for B, i.e. $A^x_{\max} \le \alpha A^*_{\max}$ and $B^x_{\text{sum}} \le \beta B^*_{\text{sum}}$. Similarly an (α,β)-approximation algorithm is an algorithm which, in polynomial time, returns an (α,β)-schedule. A negative result for a bicriterion problem with objective(A, B) will be any result that shows that for some α and β instances exist for which no (x,y)-schedule exists with $x < \alpha$ and simultaneously $y < \beta$.

There are two technical issues associated with these metrics. The first is that the lateness of a job can be 0 or negative. Therefore, as is standard in the literature, we work with the delivery time model in which each job has an associated positive delivery time instead of a deadline[12]. Let q_j be the non-negative delivery time required by job j. We overload notation and let $L_j = C_j + q_j$, with the remaining objectives defined as before. This model is equivalent to one in which the deadlines are actually non-positive. Second, as is customary in the literature, we approximate the metric of maximizing the number of on-time jobs, denoted $\max \sum_j (1 - U_j)$.

2.2 Environments

Our results will apply to a large number of machine environments, and we will take the liberty of saying that they apply to *any* scheduling problem. We borrow the definition of *any* from Stein and Wein[23] to mean any problem that meets the following two conditions.

- **Truncation:** If we take a valid schedule S and remove from it all jobs that complete after time t, the schedule remains a valid schedule for those jobs that remain.
- **Composition:** Given two valid schedules S^1 and S^2 for two sets J_1 and J_2 of jobs (where $J_1 \cap J_2$ is potentially nonempty), the composition of S^1 and S^2, obtained by appending S^2 to the end of S^1, and removing from S^2 all jobs that are in $J_1 \cap J_2$, is a valid schedule for $J_1 \cup J_2$.

These two conditions encompass most known scheduling environments, and all cases in which the processing time of a job is independents of the time at which it runs. They encompass environments that are more general than those defined in this section. For example, they capture preemptive scheduling environments,

COMBINE(S^1, S^2, t)
1. Let $K = \{j : C_j^2 > t\}$ be the set of jobs that complete after time t in schedule S^2.
2. Create schedule $S^{2'}$ from schedule S^2 by removing from S^2 all jobs in K.
3. Create schedule $S^{1'}$ by removing from S^1 all jobs in $J - K$.
4. Return schedule S^3, formed by appending $S^{1'}$ to the end of $S^{2'}$.

Figure 1: Procedure Combine

precedence constraints between jobs, and also unrelated machines environments, in which the processing time of a job on a machine is dependent on the machine-job pair. In the interests of space, we do not discuss these types of environments in this extended abstract.

The Combine operation: A basic part of many of our constructions will be to combine two schedules via truncation and composition. Given two valid schedules S^1 and S^2 for a set of jobs J, we use the procedure COMBINE to create a valid schedule S^3 for J. This procedure removes from schedule S_2 the jobs that complete after time t. It then schedules the remaining jobs in S_2 first, followed by the removed jobs, scheduled as they appeared in schedule S_1. We call t the *breakpoint*. Our two conditions on any scheduling problem are enough to show that S^3 is a valid schedule. The details appear in Figure 1.

2.3 Related Work

While some papers have explicitly set out to address bicriterion scheduling problems, other results have been the byproduct of work on single criterion scheduling problems. For instance, Graham showed in 1966 that using any list scheduling algorithm for the problem of scheduling jobs on parallel identical machines will produce a schedule of length at most twice optimal[5]. One list-scheduling algorithm schedules jobs according to non-increasing ratio of weight to processing time. This turns out to produce a schedule with average weighted completion time at most $(\sqrt{2}+1)/2$ times the optimal average weighted completion time[13], and hence is a $(2, \sqrt{2}+1)/2)$-approximation algorithm. In the special case where the weights are all equal this actually achieves the optimal value[4].

A set of schedules is said to be "Pareto optimal" if no schedule exists that is simultaneously better, in terms of both criteria, than any of the schedules in that set. Various bicriterion scheduling problems have been approached using the idea of Pareto optimal schedules[24, 10, 17, 16, 8, 9]. Other papers have approached bicriterion scheduling by fixing the value of one of the criteria and then optimizing the other criterion[21, 22, 11]. By considering schedules that were only close to optimal for both criteria, Chakrabarti et al.[2] were able to outline general techniques for creating algorithms to optimize the makespan and average weighted completion time simultaneously of a set of jobs. More recently, Stein and Wein[23] were able to show that for any scheduling problem there exists schedules that are $(1.88, 1.88)$-approximations for the makespan and the average weighted completion time, and Aslam et al [1] improved these results to, for example, $(2, e/(e-1))$-approximations.

3 Previous Existence Results and Techniques

Stein and Wein[23] showed that for any scheduling problem there exists schedules that are simultaneously good approximations for the makespan and total weighted completion time. Their method is encapsulated in COMBINE, where S^1 is the optimal schedule for makespan and S^2 is the optimal average completion time schedule. The following lemma is implicit in their work[23]:

LEMMA 3.1. *[23] For any scheduling problem, let S^1 and S^2 be two valid schedules. Then for any $\lambda \geq 0$, let S^λ =COMBINE$(S^1, S^2, \lambda C^1_{\max})$. Then $C^\lambda_{\max} \leq (1+\lambda)C^1_{\max}$, and for each job j, $C^\lambda_j \leq (1+\frac{1}{\lambda})C^2_j$.*

By considering the distribution of weight in the optimal average completion time schedule and choosing the best breakpoint out of 3 different possibilities, Stein and Wein were able to show the existence of $(2, 1.735)$, $(1.785, 2)$ and $(1.88, 188)$-schedules for the makespan and average weighted completion time. Aslam et al.[1] extended these results by considering infinitely many breakpoints and choosing the best one according to the distribution of weight in the optimal average completion time schedule, thereby proving, for example, the existence of $(2, 1.582)$, $(1.695, 2)$, and $(1.806, 1.806)$-schedules for $(C_{\max}, \sum w_j C_j)$. We outline their technique below, for more details, see [1, 20].

Given an optimal average completion time schedule S^{ACT}, if we normalize the weights so that $C^{ACT}_{\text{sum}} = 1$, we can interpret the average completion time schedule as a continuous probability density function(pdf). In particular, we can use the pdf with cumulative distribution function

$$\Pr(X \leq x) = \sum_{j: C_j^{ACT} \leq x} w_j C_j.$$

The advantage of probability density functions is that they are continuous and we can more easily consider

infinitely many breakpoints and then calculate an upper bound on the average completion time of the resulting schedule based on the best possible breakpoint for a given pdf. By considering all possible distributions f, Aslam et al.[1] showed that finding the worst case instance to maximize the average completion time of S^λ is equivalent to finding the pdf which maximizes the following expression:

$$1 + \max_f \min_{0\le\alpha\le\rho} \int_\alpha^\infty \frac{1+\alpha-x}{x} f(x)\,dx.$$

Here α is restricted to be in $[0,\rho]$ to guarantee that any breakpoint that is chosen to minimize the average completion time, will still result in a schedule of length no more than $(1+\rho)C^1_{\max}$. Aslam et al.[1] then presented the following upper bound.

LEMMA 3.2. *[1] If f ranges over all pdf's,*

$$\max_f \min_{0\le\alpha\le\rho} \int_\alpha^\infty \frac{1+\alpha-x}{x} f(x)dx \le \frac{1}{e^\rho - 1}.$$

The direct result of Lemmas 3.1 and 3.2 is a simultaneous upper bound on the makespan and average completion time of the best possible S^λ.

THEOREM 3.1. *[1] For any $\rho \in [0,1]$, for any scheduling problem, there exists a $\left(1+\rho, \frac{e^\rho}{e^\rho-1}\right)$-approximation for $(C_{\max}, \sum w_j C_j)$.*

4 New Upper Bounds for Other Criteria

In this section, we give bicriterion existence results for most pairs of minmax and minsum criteria. Table 1 summarizes our results. For 10 of the possible pairs, we give constant factor upper bounds, but in two cases instances exist for which no schedule is within a constant factor of optimal for both criteria. These two cases are designated by a "NO" and will be discussed in greater detail in Section 5. The various results are obtained by using some of the machinery developed in [1] along with several new techniques that are specific to the particular objectives.

In this extended abstract, we give the details of two of these upper bound results: simultaneously minimizing the makespan and average weighted flow time and simultaneously minimizing the maximum flow time and average weighted completion time. Details on the remaining metrics can be found in the undergraduate thesis of the first author [20].

4.1 Makespan and Average Weighted Flow Time

If S^{AFT} is the optimal average flow time schedule and S^{C} is the optimal makespan schedule of length $C^*_{\max}$, we will create the schedule $S^\lambda = \text{COMBINE}(S^{\text{C}}, S^{\text{AFT}}, \lambda C^*_{\max})$. Then by choosing the best $\lambda \in [0,\rho]$ we arrive at the following result.

THEOREM 4.1. *For any $\rho \in [0,1]$, for any scheduling problem, there exists a $(2+\rho, e^\rho/(e^\rho-1))$ approximation for the makespan and average weighted flow time.*

Proof Sketch. Let $r_{\max} = \max_j r_j$ and let $K = C^*_{\max}$ denote the length of S^C. Since all jobs must be released before the end of schedule S^C, $r_{\max} \le K$. Therefore, for each job j,

(4.1)
$$F_j^{\text{AFT}} = C_j^{\text{AFT}} - r_j \ge C_j^{\text{AFT}} - r_{\max} \ge C_j^{\text{AFT}} - K\ .$$

In order to use this lower bound, we consider the results of $S^\lambda = \text{COMBINE}(S^C, S^{\text{AFT}}, \lambda K)$, where $\lambda \in [1,(1+\rho)]$. By Lemma 3.1, for any choice of $\lambda \in [1, 1+\rho]$ we know that the length of the resulting schedule will be at most $(2+\rho)K$. Therefore, once we have ρ we need to choose λ to minimize F^λ_{sum}, the average flow time of our new schedule S^λ. In this schedule, a job j with $C_j^{\text{AFT}} < \lambda K$ is before the breakpoint, and so $C_j^\lambda = C_j^{\text{AFT}}$ which implies that $F_j^\lambda = F_j^{\text{AFT}}$. Now consider a job j with $C_j^{\text{AFT}} \ge \lambda K$. Since $C^\lambda_{\max} \le (1+\lambda)K$, the new flow time of job j in our schedule can be characterized as

$$\begin{aligned} F_j^\lambda &= (C_j^\lambda - C_j^{\text{AFT}}) + F_j^{\text{AFT}} \\ &\le (1+\lambda)K - C_j^{\text{AFT}} + F_j^{\text{AFT}} \\ &\le \left(\frac{\lambda K}{C_j^{\text{AFT}} - K}\right) F_j^{\text{AFT}}\ , \end{aligned}$$

where the last inequality follows from (4.1) above. Now that we have an upper bound on the total increase of the flow time of each job for a given choice of λ, we need to find a lower bound on the optimal average weighted flow time schedule. We do this using the same lower bound. Notice that for a particular time y, $\sum_{(j:C_j^{\text{AFT}}=y)} w_j F_j^{\text{AFT}} \ge \sum_{(j:C_j^{\text{AFT}}=y)} w_j(y-K)$ is a lower bound on the total weighted flow time of all jobs completing at time y in the optimal average weighted flow time schedule. We will now express this lower bound as a continuous function $g(y) = \sum_j w_j(C_j^{\text{AFT}} - K)\delta(C_j^{\text{AFT}} - y)$, where $\delta(\cdot)$, Dirac's delta function, is defined to be the function that satisfies the conditions $\delta(x) = 0$ for all $x \ne 0$ and $\int_{-\infty}^\infty \delta(x)dx = 1$. This means that for a function $f(x)$, Dirac's delta function has the property that $\int_{-\infty}^\infty f(x)\delta(x-z)dx = f(z)$. Given this representation of the weight completing at a certain time, we can now integrate over all time to arrive at a lower bound for our schedule. Since we are concerned

with the worst case schedule for our analysis we note that the worst case pdf, $g(y)$, will have $\int_0^K g(y)dy = 0$. This means that we can assume wlog that the weights have been normalized so that $\sum_j w_j(C_j^{\text{AFT}} - K) = 1$ and that all jobs have $C_j^{\text{AFT}} \geq K$. At this point we can write $\sum_j w_j F_j^\lambda \leq \int_0^{\lambda K} g(y)\,dy + \int_{\lambda K}^\infty \frac{\lambda K}{y-K} g(y)\,dy = 1 + \int_{\lambda K}^\infty \frac{(1+\lambda)K-y}{y-K} g(y)\,dy$. For a particular $g(y)$ we can find the best λ in the range $[1, 1+\rho]$ to minimize the this integral. To arrive at an upper bound on the average flow time of a schedule with a makespan at most $(2+\rho)K$, we find an upper bound on the pdf that maximizes this calculation. This corresponds to solving the problem

$$\max_g \min_{1\leq\lambda\leq 1+\rho} \int_{\lambda K}^\infty \frac{(1+\lambda)K - y}{y-K} g(y)\,dy,$$

where g is a probability distribution over $[0,\infty)$. This is equivalent to

$$\max_f \min_{1\leq\lambda\leq 1+\rho} \int_\lambda^\infty \frac{1+\lambda-x}{x-1} f(x)\,dx,$$

where now f ranges over all distributions.

To solve this we choose $\alpha = \lambda - 1$ from the range $[0,\rho]$. To make the corresponding shift in the summation we let $n = x - 1$. The result of this set of transformations is

$$\max_f \min_{0\leq\alpha\leq\rho} \int_\alpha^\infty \frac{1+\alpha-n}{n} f(n)\,dn$$

which was shown by Aslam et al.[1] to be at most $1/(e^\rho - 1)$. Therefore choosing the best breakpoint to minimize the average flow time of S^λ results in a schedule of length at most $(2+\rho)C_{\max}^*$ with average flow time at most $(e^\rho/(e^\rho-1))F_{\text{sum}}^*$. □

4.2 Maximum Flow Time and Average Weighted Completion Time

Now we will look at the problem of simultaneously minimizing the maximum flow time of a schedule and the average weighted completion time.

THEOREM 4.2. *For any $\rho \in [0,1]$, for any scheduling problem, there exists a $(1+\rho, 2+\frac{1}{\rho})$-schedule for the maximum flow time and average weighted completion time.*

Proof. Let S^{F} be the optimal maximal flow time schedule, let S^{ACT} be the optimal average weighted completion time schedule, and consider the schedule $S^\lambda = \text{COMBINE}(S^{\text{F}}, S^{\text{ACT}}, \lambda F_{\max}^*)$.

First we claim that $F_{\max}^\lambda \leq (1+\lambda)F_{\max}^*$. Any job that is run according to the optimal average completion time schedule S^{ACT} must complete before time $\lambda F_{\max}^*$ and therefore $F_j^\lambda \leq \lambda F_{\max}^*$. All other jobs are run according to their order in the optimal maximum flow time schedule. When this schedule was started at time 0, this ordering guaranteed a maximum flow time of $F_{\max}^*$. Since any job j that is run in this portion of S^λ is delayed by at most $\lambda F_{\max}^*$ with respect to its start time in S^{F}, its flow time can increase by at most $\lambda F_{\max}^*$. Therefore, $F^\lambda \leq (1+\lambda)F_{\max}^*$, for all jobs j.

Now we need to analyze the average completion time of S^λ. All jobs j that complete before time $\lambda F_{\max}^*$ in S^{ACT} will have $C_j^\lambda = C_j^{\text{ACT}}$. By the above argument, we know that $F_j^\lambda \leq (1+\lambda)F_{\max}^*$. This leads to the following upper bound on C_j^λ for jobs with $C_j^{\text{ACT}} \geq \lambda F_{\max}^*$:

$$\begin{aligned} C_j^\lambda &\leq r_j + (1+\lambda)F_{\max}^* \\ &\leq C_j^{\text{ACT}} + (1+\lambda)F_{\max}^* \\ &\leq C_j^{\text{ACT}} + (1+\lambda)C_j^{\text{ACT}}/\lambda \\ &= (2+1/\lambda)C_j^{\text{ACT}}, \end{aligned}$$

and hence, $\sum_j w_j C_j^\lambda \leq (2+\frac{1}{\lambda})\sum_j w_j C_j^{\text{ACT}} = (2+\frac{1}{\lambda})C_{\text{sum}}^*$. □

5 Lower Bounds

Table 2 gives lower bounds on the approximability of the same problems considered in the previous section. As an example of the techniques used to derive these lower bounds we will consider two bicriterion scheduling problems.

5.1 Makespan and Average Completion Time

For the problem of scheduling unweighted jobs with release dates on one machine to minimize the makespan and average completion time we prove a lower bound matching the result of Theorem 3.1.

THEOREM 5.1. *For $0 < \rho < 1$, there exists an (infinite-sized) instance such that there is no (x,y)-schedule with $x < 1+\rho$ and $y < \frac{e^\rho}{e^\rho - 1}$ for $(C_{\max}, \sum C_j)$.*

Proof Sketch. Consider an instance in which there are $n+1$ total jobs, n of which are jobs with $p_j = 0$ and the other one job has $p_j = 1$. The job with processing time 1 is released at time 0. The following pdf f specifies, for some fixed ρ where $0 < \rho < 1$, the release dates and therefore also optimal average completion times of the

n jobs with $p_j = 0$:

$$f(t) = \delta(\rho - t)\frac{1}{e^\rho} + \begin{cases} \frac{1}{e^t} & 0 < t < \rho \\ 0 & t \geq \rho \,. \end{cases}$$

Let $C^s_{\max}$ and C^s_{sum} denote the makespan and average completion time of a schedule created with the job with $p_j = 1$ starting at s. Notice that the only schedules of interest occur with $0 \leq s \leq \rho$. We know that the makespan will be $C^s_{\max} = (1 + s)$. Since the optimal makespan is 1, we have that $C^s_{\max} = (1 + s)C^*_{\max}$. Next, when analyzing the effect of s on C^s_{sum} we can ignore the completion time of the job with processing time 1 since as n approaches infinity, it becomes negligible. Thus, we obtain

$$C^s_{\text{sum}} = \int_0^s xf(x)dx + (s+1)\left[1 - \int_0^s f(x)dx\right]$$
$$= \begin{cases} 1 & 0 \leq s < \rho \\ \frac{e^\rho - 1}{e^\rho} & s = \rho \,. \end{cases}$$

All schedules that start the one-unit job before time ρ are no better than $\frac{e^\rho}{e^\rho - 1}$-approximations for the optimal average completion time and all other schedules must start the unit job after time ρ and therefore have length at least $1 + \rho$ times the optimal makespan. □

5.2 Maximum Flow Time and Average Flow Time

Now we consider scheduling a set of jobs to simultaneously minimize the maximum flow time and the average weighted flow time of the resulting schedule. The following example shows that instances exist for which no schedule will be a constant factor approximation for both criteria simultaneously.

THEOREM 5.2. *If $F^*_{\max}$ is the optimal maximum flow time and F^*_{sum} is the optimal total flow time of a set of N jobs, then instances exist for which there is no (α, β)-schedule with $1 \leq \alpha < \frac{\sqrt{N}}{4}$ and $1 \leq \beta < \frac{\sqrt{N}}{4}$ for $(F_{\max}, \sum F_j)$.*

Proof. Consider the following example on one machine with release dates. Let j_0 be released at $t = 0$ with processing time $p_0 = \sqrt{N}$. Let jobs $j_1, j_2, \ldots, j_{N-1}$ be jobs of length 1. Let $r_i = i$ for all $j_1, j_2, \ldots, j_{N-1}$. Since j_0 has the earliest release date and the longest processing time, the schedule S^{F} that achieves $F^*_{\max}$ runs j_0 at time $t = 0$ when it is released. All $N - 1$ small jobs are delayed by $\sqrt{N}$ time units. $F^*_{\max}$ of this schedule is $F^*_{\max} = \sqrt{N}$. The total flow time of this schedule is $\sum F^F_j = N^{\frac{3}{2}}$.

On the other hand, the optimal total flow time schedule, S^{TFT}, will run all small jobs $j_1, j_2, \ldots, j_{N-1}$ as they are released and run j_0 starting at time N. Since all jobs with $p_j = 1$ complete 1 time unit after they are started, $F^{\text{TFT}}_i = 1, \forall i > 0$. The larger job will now have to wait until all the small jobs complete and therefore will have $F^{\text{TFT}}_0 = N + \sqrt{N}$. The total flow time of this schedule will be $\sum F^{\text{TFT}}_j = 2N + \sqrt{N} - 1$.

Any schedule in which j_0 starts before $t = N/4$ will have at least $3N/4$ small jobs with $F_j = \sqrt{N}$ and therefore a total flow time of $\sum_j F_j \geq= 3N^{\frac{3}{2}}/4$ which is at least $\sqrt{N}/4$ times the optimal total flow time. However, if j_0 starts after $t = N/4$ then $F_{\max} \geq N/4$. Which is also at least $\sqrt{N}/4$ times the optimal maximum flow time. Therefore no (α, β)-schedule exists with $1 \leq \alpha < \sqrt{N}/4$ and $1 \leq \beta < \sqrt{N}/4$. □

6 Algorithms

The existence results in this paper lead to algorithms in a natural way. Suppose that, for a particular problem, we have shown the existence of an (α, β)-approximation for objectives A and B, and we have an x-approximation algorithm for objective A and a y-approximation algorithm for metric B. Then using our constructions, we obtain an $(\alpha x, \beta y)$-approximation algorithm for objectives A and B.

In certain special cases, we can obtain better bounds. Consider the problem of scheduling jobs on 1 machine with release dates in which the objectives are minimum makespan and average completion time. For these objectives, we have tight upper and lower bounds of $(1+\rho, \frac{e^\rho}{e^\rho - 1})$ on the existence of bicriterion schedules. In this section, we give an algorithm which actually matches these bounds. Using ideas from α-scheduling we first present a randomized algorithm that achieves these bounds and then describe a deterministic polynomial time algorithm.

The idea behind α-scheduling is to use an optimal preemptive schedule to obtain an ordering of jobs for the non-preemptive case[18, 7, 3]. Consider the schedule P created by scheduling jobs preemptively by the shortest remaining processing time($\mathcal{SRPT}$). For some $0 < \alpha \leq 1$, let $C^P_j(\alpha)$ be the time at which an α-fraction of j completes in schedule P. A non-preemptive schedule can then be created by list scheduling jobs according to non-decreasing $C^P_j(\alpha)$. Chekuri et al. [3] provide the following two lemmas for this approach to α-scheduling.

LEMMA 6.1. *[3] The makespan of any α-schedule is at most $1 + \alpha$ times the optimal makespan.*

The following lemma refers to the randomized algorithm

$\mathcal{RAND}$ that chooses α randomly from a distribution $f(x)$ and then uses that α to create a non-preemptive α-schedule.

LEMMA 6.2. *[3] The expected average completion time of $\mathcal{RAND}$ is at most $1+\delta$ times the optimal preemptive average completion time where*

$$\delta = \max_{0<t\leq 1} \int_0^t \frac{1+\alpha-t}{t} f(\alpha)d\alpha.$$

The randomized algorithm $\mathcal{RAND}-\beta$ chooses α randomly from the probability distribution

$$f(\alpha) = \begin{cases} \frac{e^{\alpha}}{e^{\beta}-1} & 0 \leq \alpha < \beta \\ 0 & \alpha > \beta \,. \end{cases}$$

THEOREM 6.1. *For $0 < \beta \leq 1$, $\mathcal{RAND}-\beta$ is a randomized $(1+\beta, \frac{e^{\beta}}{e^{\beta}-1})$-approximation algorithm for $1|r_j|(C_{\max}, \sum C_j)$.*

Finally using the observation by Chekuri et al.[3] that $\mathcal{SRPT}$ creates a preemptive schedule with at most $n-1$ preemptions, we know that there are at most $n-1$ interesting choices of α. This means that there are at most n distinct non-preemptive schedules that can be derived by using α-scheduling to convert the preemptive schedule to a non-preemptive schedule. Chekuri et al.[3] use this to show that by searching all n possible schedules and choosing the best one, we can in polynomial time, find a non-preemptive schedule that matches the expected bounds for the randomized algorithm. For $0 < \beta \leq 1$, let $\mathcal{BEST}-\beta$ be the deterministic algorithm that tries all possible α-schedules with $0 < \alpha \leq \beta$ and chooses the one with the smallest total completion time.

THEOREM 6.2. *For $0 < \beta \leq 1$, $\mathcal{BEST}-\beta$ is a deterministic $(1+\beta, \frac{e^{\beta}}{e^{\beta}-1})$-approximation algorithm for $1|r_j|(C_{\max}, \sum C_j)$.*

References

[1] J. A. Aslam, A. Rasala, C. Stein, and N. Young. Improved bicriteria existence theorems for scheduling. In *Proceedings of the 10th ACM-SIAM Symposium on Discrete Algorithms*, 1999. To appear.

[2] S. Chakrabarti, C. A. Phillips, A. S. Schulz, D. B. Shmoys, C. Stein, and J. Wein. Improved scheduling algorithms for minsum criteria. In F. Meyer auf der Heide and B. Monien, editors, *Automata, Languages and Programming*, number 1099 in Lecture Notes in Computer Science. Springer, Berlin, 1996. Proceedings of the 23rd International Colloquium (ICALP'96).

[3] C. Chekuri, R. Motwani, B. Natarajan, and C. Stein. Approximation techniques for average completion time scheduling. In *Proceedings of the 8th ACM-SIAM Symposium on Discrete Algorithms*, pages 609–618, January 1997. To appear in SIAM J. Computing.

[4] R.W. Conway, W.L. Maxwell, and L.W. Miller. *Theory of Scheduling*. Addison-Wesley, 1967.

[5] R.L. Graham. Bounds for certain multiprocessor anomalies. *Bell System Technical Journal*, 45:1563–1581, 1966.

[6] R.L. Graham, E.L. Lawler, J.K. Lenstra, and A.H.G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics*, 5:287–326, 1979.

[7] L. A. Hall, A. S. Schulz, D. B. Shmoys, and J. Wein. Scheduling to minimize average completion time: Off-line and on-line approximation algorithms. *Mathematics of Operations Research*, 22:513–544, August 1997.

[8] J. A. Hoogeveen. Minimizing maximum promptness and maximum lateness on a single machine. *Mathematics of Operations Research*, 21:100–114, 1996.

[9] J. A. Hoogeveen. Single-machine scheduling to minimize a function of two or three maximum cost criteria. *Journal of Algorithms*, 21(2):415–433, 1996.

[10] J. A. Hoogeveen and S.L. van de Velde. Minimizing total completion time and maximum cost simulatneously is solvable in polynomial time. *Operations Research Letters*, 17:205–208, 1995.

[11] C.A.J. Hurkens and M.J. Coster. On the makespan of a schedule minimizing total completion time for unrelated parallel machines. Unpublished manuscript, 1996.

[12] David Karger, Cliff Stein, and Joel Wein. *CRC Handbook on Algorithms*, chapter Scheduling Algorithms. CRC Press, 1998.

[13] T. Kawaguchi and S. Kyan. Worst case bound of an LRF schedule for the mean weighted flow-time problem. *SIAM Journal on Computing*, 15:1119–1129, 1986.

[14] E.L. Lawler. Optimal sequencing of a single machine subject to precedence constraints. *Management Science*, 19:544–546, 1973.

[15] E.L. Lawler, J.K. Lenstra, A.H.G. Rinooy Kan, and D.B. Shmoys. Sequencing and scheduling: Algorithms and complexity. In S.C. Graves, A.H.G. Rinnooy Kan, and P.H. Zipkin, editors, *Handbooks in Operations Research and Management Science, Vol 4., Logistics of Production and Inventory*, pages 445–522. North-Holland, 1993.

[16] S.T. McCormick and M.L. Pinedo. Scheduling n independent jobs on m uniform machines with both flow time and makespan objectives: A parametric approach. *ORSA Journal of Computing*, 7:63–77, 1992.

[17] R.T. Nelson, R.K. Sarin, and R.L. Daniels. Scheduling with multiple performance measures: the one-machine case. *Management Science*, 32:464–479, 1986.

[18] C. Phillips, C. Stein, and J. Wein. Minimizing average completion time in the presence of release dates.

Mathematical Programming, 82:199–223, 1998.

[19] M. Pinedo. *Scheduling: Theory, Algorithms and Systems*. Prentice Hall, 1995.

[20] A. Rasala. Existence theorems for scheduling to meet two objectives. Technical Report PCS-TR99-347, Department of Computer Science, Dartmouth College, 1999.

[21] D. B. Shmoys and E. Tardos. An approximation algorithm for the generalized assignment problem. *Mathematical Programming A*, 62:461–474, 1993.

[22] W.E. Smith. Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, 3:59–66, 1956.

[23] C. Stein and J. Wein. On the existence of schedules that are near-optimal for both makespan and total weighted completion time. *Operations Research Letters*, 21, 1997.

[24] L.N. Van Wassenhove and F. Gelders. Solving a bicriterion scheduling problem. *European Journal of Operations Research*, 4:42–48, 1980.