# THE ASYMPTOTIC OPTIMALITY OF THE SPT RULE FOR THE FLOW SHOP MEAN COMPLETION TIME PROBLEM

**PHILIP KAMINSKY**

*Industrial Engineering and Operations Research, University of California, Berkeley, California 94720, kaminsky@ieor.berkeley.edu*

**DAVID SIMCHI-LEVI**

*Massachusetts Institute of Technology, Cambridge, Massachusetts 02139-4307, dslevi@mit.edu*

(Received August 1997; revisions received January 1999, October 1999; accepted February 2000)

In the flow shop mean completion time problem, a set of jobs has to be processed on $m$-machines. Every machine has to process each one of the jobs, and every job has the same routing through the machines. The objective is to determine a sequence of the jobs on the machines so as to minimize the sum of the completion times of all jobs on the final machine. In this paper, we prove the asymptotic optimality of the Shortest Processing Time algorithm for any continuous, independent, and identically distributed job processing times.

In the *m-machine flow shop problem*, a set of jobs, each consisting of $m$ operations, must be sequentially processed on $m$ machines. Each machine can handle at most one job at a time, and a job can only be processed on one machine at a time. The jobs have to be processed on each of the machines without preemption, and every machine serves the arriving jobs in a first-come, first-served fashion. Given the processing times of each of the jobs on each of the machines, the *Flow Shop Mean Completion Time Problem* involves determining a sequence of the jobs on the machines that minimizes the average or, equivalently, the sum, of the completion times of the jobs on the *final* machine in the sequence. It is well known (see Garey et al. 1976) that this problem is NP-hard, even in the two-machine case.

Much previous research on the Flow Shop Mean Completion Time problem has focused on optimal solutions to small-size problems, sometimes with as many as 10 machines and 50 jobs but most often with only 2 machines. Some of these approaches were adapted to find heuristic solutions. For example, Kohler and Steiglitz (1975) combined branch and bound techniques with local search heuristics to, approximately, solve 2-machine problems for up to 50 jobs.

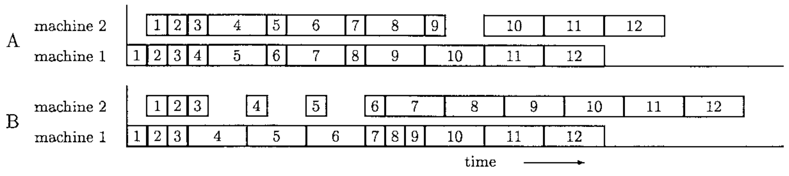
For larger problems, dispatch rules are typically used to find reasonable sequences. Bhaskaran and Pinedo (1992) suggest a variety of simple and compound dispatch rules useful for larger-problem instances. Morton and Pentico (1993) compare bottleneck dynamics and OPT-like rules, which adjust the schedule based on the perceived bottlenecks, to dispatch rule-based heuristic scheduling approaches. Simulation experiments demonstrated the relative advantage of each of the approaches, although the absolute performance of each approach was not known. Most work involving the use of dispatch rules for the flow shop model and, indeed, for multiple-machine models in general, is experimental in nature.

Recently, Kaminsky and Simchi-Levi (1998) used a different approach to analyze this model. Utilizing the tools of probabilistic analysis, they characterized the underlying structure of the asymptotic optimal solution to the Flow Shop Completion Time Problem (weighted, in that case) as the number of jobs increases to infinity and developed a simple algorithm for the problem based on this analysis. Their algorithm is a modification of the Shortest Processing Time (SPT) sequence, which is defined as follows: Sequence the jobs in increasing order of their sum of the processing times on all of the machines. Indeed, they show that a specific Shortest Processing Time sequence of the jobs is optimal for a special discretized version of the problem. This special discretized version has many different SPT sequences, however, so the question of the asymptotic optimality of any Shortest Processing Time algorithm for general flow shop problems, and in particular for problems in which processing times are continuous random variables, remained open.

The difficulty in answering this question is illustrated by the following example. Consider a two-machine scheduling problem with the same number, 3, of four types of jobs. The first type, which we call the $(1, 3)$ type, has processing time of 1 on Machine 1 and 3 on Machine 2; the second type, the $(3, 1)$ type, has a processing time of 3 on Machine 1 and 1 on Machine 2; and the remaining two types, $(1, 1)$ and $(3, 3)$, have a processing time of either 1 or 3 on both machines. Clearly, in an SPT sequence, the $(1, 1)$ jobs will be scheduled first, and the $(3, 3)$ jobs will be scheduled last. However, any sequence of $(1, 3)$ and $(3, 1)$ jobs is an SPT sequence. Interestingly,

*Subject classifications:* Production/scheduling: multiple machine sequencing, Flow shop weighted completion time problem, shortest processing time dispatch rule. Analysis of algorithms: probabilistic analysis.
*Area of review:* Optimization.

**Figure 1.** Comparing SPT sequences.



alternating between $(1, 3)$ and $(3, 1)$ jobs reduces total completion time relative to other SPT sequences, as illustrated in Figure 1. Both sequences in this Figure are SPT, but Sequence $A$ has an objective value of 222, whereas Sequence $B$ has an objective value of 252. Thus, the SPT sequence can either be very good (and in fact, asymptotically optimal, as in Kaminsky and Simchi-Levi 1998) or, depending on *which* SPT sequence is used, far from optimal.

Previous computational results (see Kaminsky and Simchi-Levi 1998) indicate that the SPT rule is in fact effective for various randomly generated flow shop problems, particularly as the number of jobs gets to be large. For example, for 5,000 jobs with processing times generated from a uniform distribution, the SPT sequence has a cost which, on average, is between 1% and 7% larger than that of a lower bound, depending on the number of machines in the flow shop. Specifically, in the case of a three-machine flow shop problem, the average increase in cost is about 8% when the number of jobs is 500.

Our objective in this paper is to *characterize the conditions* under which any Shortest Processing Time sequence is asymptotically optimal, as the number of jobs tends to infinity, for the Flow Shop Mean Completion Time Problem, and thus provide an analytical, rather than an experimental, understanding of the effectiveness of this particular dispatch rule. Indeed, the key assumption in the probabilistic model described in the next section is that processing times are generated from a continuous distribution. This implies that, with high probability, all job processing times are different, and thus, the model avoids the difficulty of having to select "the correct SPT" that arises in the example shown above.

To put our work into perspective, we highlight other research relating to probabilistic analysis of scheduling problems. Much of this work has focused on parallel machine problems, including the work of Coffman et al. (1982), Loulou (1984), and Frenk and Rinnooy Kan (1987), who analyze the parallel machine scheduling problem when the objective is to minimize the makespan, and Spaccamela et al. (1992) and Webster (1993), who analyze the parallel machine *weighted completion time* model. The only work we are aware of related to flow shop models is the recent work by Ramudhin et al. (1996), who analyze the two-machine flow shop makespan model. Finally, Hall (1997) surveys the development of algorithms with guaranteed worst-case bounds for various related scheduling models.

## 1. THE MODEL AND THE MAIN RESULT

To formally present our model, which is similar to the model presented in Kaminsky and Simchi-Levi (1998), consider a set of $n$ jobs that have to be processed on $m$ machines. Job $i, i = 1, 2, \ldots, n$, has a processing time $t_i^l$ on Machine $l, l = 1, 2, \ldots, m$. The processing times are drawn from an identical and bounded distribution with nonzero density $\phi(\cdot)$, defined on the interval $(0, 1]$.

Each job must be processed without preemption on each of the machines sequentially. That is, each job must be processed on Machine 1 through Machine $m$ in that order. Jobs are available for processing at time zero, and with the exception of the first machine all other machines process the jobs in a first-come, first-served manner, a so-called *permutation* schedule. Also, there is unlimited intermediate storage between successive machines, and we are interested in *semiactive* schedules, or schedules in which no operation on any machine can be completed earlier without altering the processing sequence on any of the machines (see Pinedo 1995). For the objective we are about to describe, there is always an optimal *semiactive schedule*.

The objective is to determine a *schedule*, or sequence of jobs, such that the total completion times of all the jobs on the final machine is minimized. Note that although the processing times of each of the $n$ jobs are drawn from a random distribution as described above, all of the processing times are available *before the schedule is determined.* We call this problem Problem $P$, and use $Z^*$ to denote its optimal objective function value. That is, $Z^*$ is the minimum possible total completion time of all jobs in Problem $P$. Similarly, given a heuristic $H$ for the Flow Shop Completion Time Problem, we use $Z^H$ to denote the sum of the completion time in the resulting schedule. Specifically, $Z^{SPT}$ represents the sum of completion times of the jobs when they are sequenced from smallest to largest total processing times, an SPT sequence.

In this paper we prove the following.

Theorem 1. *Let the processing times* $t_i^1, t_i^2, \ldots, t_i^m$, $i = 1, 2, \ldots, n$, *be independent random variables having the same continuous and bounded distribution* $\phi(\cdot)$ *with nonzero density defined on* $(0, 1]$. *Then with probability one we have*

$$\lim_{n\to\infty} \frac{Z^*}{n^2} = \lim_{n\to\infty} \frac{Z^{SPT}}{n^2}.$$

Theorem 1 thus implies that the objective value of the solution generated by the SPT sequence converges to the optimal objective value as the number of jobs tends to infinity. Unfortunately, the rate of convergence remains an open question. However, the computational results cited above (Kaminsky and Simchi-Levi 1998) provide some insight as to the rate at which the objective value of the SPT solution approaches the optimal objective value. For example, consider the three-machine flow shop, in which processing times of each job are generated from a uniform distribution. For 500 jobs, the SPT sequence has a cost which, on average, is about 8% higher than that of a lower bound on the optimal solution. This decreases to 5% for 1,000 jobs, 3% for 2,500 jobs, and 1% for 5,000 jobs.

To prove Theorem 1, we start in § 2 by presenting a simplified discrete model. For this simplified model, we prove a result analogous to our main result. This proof helps to provide some of the intuition for the proof of Theorem 1 in §3, which uses, among other results, the simplified discrete result. Certain Lemmas and Properties to which we refer throughout the analysis are included in Appendix A.

For the subsequent analysis, it is useful to define the concept of an *associated single-machine model* to the flow shop model we have defined. In particular, consider the following single-machine model, associated with Problem $P$ defined above. Given Job $i$, $i = 1, 2, \dots, n$, with processing times $t_i^1, t_i^2, \dots, t_i^m$ on Machine $1, 2, \dots, m$, respectively, let $t_i = \sum_{l=1}^m t_i^l$. Consider a single-machine scheduling problem with $n$ tasks each having a processing time $t_i$, $i = 1, 2, \dots, n$. As with the original flow shop problem, the objective of the single-machine problem is to sequence the tasks so as to minimize the sum of their completion times.

## 2. THE DISCRETE MODEL

To prove our main theorem, Theorem 1, we begin by introducing a discrete model, first introduced in Kaminsky and Simchi-Levi (1998), with a finite number of different possible processing times and a carefully defined relationship between certain subsets of the jobs. The following analysis of this *Cyclic Discrete Model* has two purposes. It provides some intuition as to why Theorem 1 is true, and it provides an upper bound that is useful in the proof of Theorem 1.

### 2.1. The Model

Consider an $m$-machine flow shop model for which the objective is to minimize the sum of the completion times. Each job has an associated vector $(t^1, t^2, \dots, t^m)$, where $t^i$ is the processing time on the $i$th machine. The *total processing time* of a job is the quantity $\sum_{l=1}^m t^l$. We say that two jobs are *identical* when the vectors representing each job are equal, element wise, and we call a set of identical jobs, which can all be represented by the same vector $(t^1, t^2, \dots, t^m)$, a *job type*.

Given a job type represented by $(t^1, t^2, \dots, t^m)$, we construct a number of new jobs types through a *cyclic shift* of the processing times. That is, given the job type represented by vector $(t^1, t^2, \dots, t^m)$, new *job types* are created by shifting the processing times over one machine in a cyclic manner. In that process we create the following job types, represented by the vectors

$$(t^2, t^3, \dots, t^m, t^1), (t^3, t^4, \dots, t^m, t^1, t^2), \dots, (t^m, t^1, t^2, \dots, t^{m-1}).$$

Of course, when some of the processing times $t^l, l = 1, 2, \dots, m$, are equal, some of the job types in the process may be identical. If, on the other hand, the processing times $t^l$, $l = 1, 2, \dots, m$, are all different, the shifted cyclic process will generate $m-1$ new nonidentical job types. To simplify the exposition, in this model we will restrict ourselves to job types for which all of the processing times are different, although the results can be quite easily generalized to include job types for which two or more processing times are the same.

We define a *group type* $g$ to be a job type, which we call $j_1^g$, and its $m-1$ *cyclic shifted job types*, $j_2^g, j_3^g, \dots, j_m^g$, where job type $j_2^g$ is shifted left one position from $j_1^g$, $j_3^g$ is shifted left two positions from $j_1^g$, and so on. Thus, each *group type* $g$ consists of $m$ job types, each of which has the same *total* processing time, $t_g$. In addition, when we refer to the *next job type* within a group, we are referring to the job type that is *shifted* one additional time to the left. That is, $j_2^g$ is the *next job type* after $j_1^g$, $j_3^g$ is the *next job* after $j_2^g$, and so on, noting especially that $j_1^g$ is the *next job type* after $j_m^g$. Finally, we refer to all groups with the same total processing time as a *family* and define $\tilde{t}_d$ to equal the total processing time of each job in Family $d$.

Now, consider a model in which there is a *finite* number, $G$, of group types and, thus, a *finite* number, $f$, of *families*. Let $n_g$ be the number of jobs of type $j_k^g$, for $k = 1, 2, \dots, m$, and $g = 1, 2, \dots, G$. Thus, all job types in a group have the same number of jobs assigned to each one of them, so $n = m\sum_{g=1}^G n_g$ is the total number of jobs, out of which $mn_g$ are associated with *group type* $g$. Also, let $\tilde{n}_l$, $l = 1, 2, \dots, f$ be the number of jobs in Family $l$. Clearly, there is a strong relationship between groups and families. That is, $\tilde{n}_l = m\sum_{i|t_i=\tilde{t}_l} n_i$.

Let $Z^*$ be the optimal solution to this $m$–machine flow shop problem, where the objective is to minimize total completion times of all jobs. In what follows, we refer to this problem as the *original Cyclic Discrete* problem.

In the probabilistic analysis that follows, we consider a Cyclic Discrete model in which groups of $m$ jobs are added to the model by selecting a group type $g$ with probability $p_g$, for $g = 1, 2, \dots, G$, and then generating $m$ jobs, one for each job type within that group. That is, with probability one, we have $p_g = \lim_{n\to\infty} n_g / \sum_{l=1}^G n_l$ for $g = 1, 2, \dots, G$. This implies that the probability that a job belongs to Family $l$, $\tilde{p}_l$, equals almost surely $\lim_{n\to\infty} \tilde{n}_l / \sum_{j=1}^f \tilde{n}_j$ for $l, l = 1, 2, \dots, f$.

Finally, given an instance of this Cyclic Discrete model, define an associated single-machine model as described in Section 1, with optimal objective value $Z_1^*$.

### 2.2. The Main Discrete Result

Consider *any* SPT ordering of $n$ jobs in the Cyclic Discrete problem described above. Associated with such an ordering is a $K_n$ *value*, which we determine as follows: Starting with the first job in the sequence, determine its *forward match* by finding in the sequence the first job that is the next job type, as defined in Section 2.1. Continue through the remaining jobs, noting that to find the forward match of job $j$, find the first job in the sequence following job $j$ that has *not been the forward match* of a job preceding job $j$ and is the next job type in the same group as job $j$. When a job has no forward match, we define the last job in its family in the SPT sequence to be its forward match. In particular, the last job in every family is the forward match of itself. Consequently, with the possible exception of the last job, a job can only be the forward match of a single other job. However, each job must have a forward match, regardless of whether or not it *is* the forward match of a preceding job.

Define the *distance between two jobs* as one plus the number of jobs between these two jobs. The $K_n$ *value* associated with a particular SPT ordering of $n$ jobs is defined as the largest distance between any job and its forward match.

Now, let $Z^{SPT_K}$ be the total completion time of all jobs in an $n$-job SPT ordering with its associated $K_n$ *value*. We note that a particular set of jobs may have more than one associated SPT ordering (depending on how ties are broken) and thus may have more than one possible $Z^{SPT_K}$ value. Finally, recall our definition of *family*, which implies that in any SPT ordering, all jobs within the same family are sequenced consecutively.

We prove the following.

THEOREM 2. *For any SPT sequence whose associated $K_n$ value satisfies almost surely*

$$\lim_{n\to\infty} \frac{K_n}{n} = 0,$$

*we have with probability one*

$$\lim_{n\to\infty} \frac{Z^*}{n^2} = \lim_{n\to\infty} \frac{Z^{SPT_K}}{n^2} = \lim_{n\to\infty} \frac{Z_1^*}{mn^2} = \theta,$$

*for some constant $\theta > 0$.*

This theorem implies that regardless of how ties are broken in a particular SPT sequence, as long as the $K_n$ value of that sequence meets the condition required above, the sequence is asymptotically optimal.

PROOF OF THEOREM 2. We prove the theorem by finding a lower bound on the asymptotic value of $Z^*/n^2$ and an upper bound on $Z^{SPT_K}/n^2$ which converge to the same value. First we apply Lemma A.1 to obtain the following lower bound, which also characterizes the value of $\theta$ in Theorem 2.

LEMMA 1. *We have almost surely*

$$\lim_{n\to\infty} \frac{Z^{SPT_K}}{n^2} \geqslant \lim_{n\to\infty} \frac{Z^*}{n^2} \geqslant \lim_{n\to\infty} \frac{Z_1^*}{mn^2} = \frac{1}{m}\left[\sum_{j=1}^{f} \tilde{t}_j \frac{\tilde{p}_j^2}{2} + \sum_{k=2}^{f} \tilde{p}_k \sum_{i=1}^{k-1} \tilde{p}_i \tilde{t}_i\right]. \quad (1)$$

PROOF. Consider the original Cyclic Discrete problem and an associated single machine scheduling problem constructed as described above. In the latter model, we have $\tilde{n}_j$ jobs each having a processing time $\tilde{t}_j$, for $j = 1, 2, \ldots, f$. The minimum total completion time of $n$ jobs on a single machine is obtained using the SPT first rule. Let

$$G_j = \tilde{t}_j \frac{(\tilde{n}_j + 1)\tilde{n}_j}{2}, \quad \forall j = 1, 2, \ldots, f.$$

The optimal objective value of the single machine problem is clearly

$$Z_1^* = \sum_{j=1}^{f} G_j + \sum_{k=2}^{f} \tilde{n}_k \left(\sum_{i=1}^{k-1} \tilde{n}_i \tilde{t}_i\right).$$

Dividing by $mn^2$, taking the limit as the number of jobs, $n$, tends to infinity, and noting that with probability one,

$$\tilde{p}_j = \lim_{n\to\infty} \tilde{n}_j / n, \quad \forall j = 1, 2, \ldots, f,$$

we get with probability one,

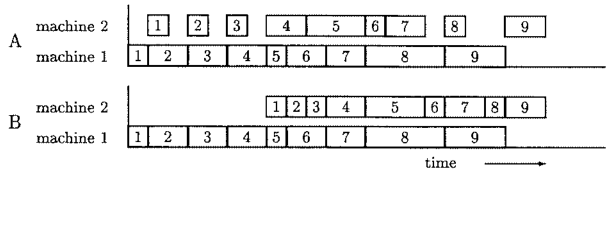
$$\lim_{n\to\infty} \frac{Z_1^*}{mn^2} = \frac{1}{m}\left[\sum_{j=1}^{f} \tilde{t}_j \frac{\tilde{p}_j^2}{2} + \sum_{k=2}^{f} \tilde{p}_k \sum_{i=1}^{k-1} \tilde{p}_i \tilde{t}_i\right]. \quad (2)$$

This, together with Lemma A.1 completes the proof. □

We now construct an upper bound on $Z^{SPT_K}$ and show that asymptotically this upper bound converges to the asymptotic lower bound from Equation (1). For this purpose, consider the original Cyclic Discrete problem. We schedule the jobs using an arbitrary SPT ordering, and determine the $K_n$ value associated with this ordering. To simplify exposition, we round up $K_n$ to the nearest multiple of $m$, the number of machines. We also index the jobs from 1 to $n$, according to their appearance in the sequence.

To construct an upper bound we hold that part of the processing time of the final job departing from Machine $m$ stationary and shift all the other jobs on Machine $m$ as far to the right as possible, with the sequence remaining the same. In other words, we keep the starting time of job $n$ on Machine $m$ the same, and beginning with job $n-1$ and going backwards to job 1, we increase the starting times of each job *on machine m only* as much as possible, without overlapping jobs and while maintaining the same order. Figure 2 provides a simple two-machine example of this shifting procedure. Sequence $A$ represents the original sequence, whereas Sequence $B$ is the shifted sequence.

**Figure 2.** Comparing the original and shifted sequences.



Let $Z^{SHIFT}$ be the total completion time of all jobs in the above *shifted* strategy. Clearly,

$$Z^{SPT_K} \leqslant Z^{SHIFT}. \tag{3}$$

Note that all of the idle time on machine $m$ in this new *shifted* sequence occurs *before the first job is processed on that machine*. We define the length of the idle time on Machine $m$ to be $I^m$.

To construct an upper bound on $Z^{SHIFT}$, consider each family, $f_j$, $j = 1, 2, \ldots, f$, and divide the family into $s_j - 1$ sets of exactly $K_n$ consecutive jobs and one additional set that contains at most $K_n$ consecutive jobs. Number the sets consecutively within a family and let $S_j^i$, $j = 1, \ldots, f$, $i = 1, \ldots, s_j$ be the $i$th set in the $j$th family.

We note the following important observations:

• All sets, with the possible exception of the last set within each family, contain $K_n$ jobs.

• An upper bound on the total processing time of any set on Machine $m$ is $K_n$, because all processing times are bounded by 1.

• Any job within Family $j$ and its $(m-1)$ *forward matches* (that is, a job, its *forward match*, the *forward match* job's *forward match*, and so on for a total of $m$ jobs) have a total processing time on Machine $m$ of $\tilde{t}_j$. This is true because this collection of jobs *must belong to one group* within Family $j$.

• For every family with $s_j > m$, consider the time Machine $m$ completes processing the last job in the set $S_j^i$, $i = m, \ldots, s_j$, $j = 1, \ldots, f$ according to the shifted sequence. This time can be divided into three components. The first is the idle time of Machine $m$; the second is the time it takes for Machine $m$ to process all jobs prior to the first job in Family $j$; the third is the total processing time of *completed jobs within the same family*, Family $j$. The latter is no more than

$$\frac{K_n}{m}\tilde{t}_j(i+1-m)+(m-1)K_n.$$

This is true because, for each $i = m, \ldots, s_j$, at least $(i+1-m)K_n$ jobs are part of collections of $m$ *forward matched* jobs, as described in the previous point; each collection has a total processing time of $\tilde{t}_j$. On the other hand, at most $(m-1)K_n$ jobs are not part of any collection and, therefore, the only thing we can say is that each one of these $(m-1)K_n$ jobs has a processing time no greater than 1.

To find an upper bound on $Z^{SHIFT}$, round the completion time of each job within a set $S_j^i$, $i = 1, 2, \ldots, s_j$, $j = 1, 2, \ldots, f$ on Machine $m$ up to the completion time of the entire set on Machine $m$. Thus, we get that

$$\begin{aligned} Z^{SHIFT} \leqslant{}& nI^m + \sum_{j=2}^{f}\tilde{n}_j\sum_{k=1}^{j-1}\left[\frac{K_n}{m}\tilde{t}_k(s_k+1-m)+(m-1)K_n\right] \\ &+ f(m-1)^2K_n^2 + K_n \\ &\cdot \sum_{j=1}^{f}\sum_{i=m}^{s_j}\left[\frac{K_n}{m}\tilde{t}_j(i+1-m)+(m-1)K_n\right], \end{aligned}$$

where the first component in the above upper bound represents total idle time, the second represents total processing time until a specific family is processed, the third is an upper bound on the sum of completion times on Machine $m$ of all jobs in $S_j^i$, $i = 1, 2, \ldots, m-1$ and $j = 1, 2, \ldots, f$, and the last component in the above upper bound represents the total processing times on Machine $m$ of all jobs in $S_j^i$, $i = m, \ldots, s_j$ and $j = 1, 2, \ldots, f$. Hence,

$$\begin{aligned} Z^{SHIFT} \leqslant{}& nI^m + \sum_{j=2}^{f}\tilde{n}_j\sum_{k=1}^{j-1}\left[\frac{K_n}{m}\tilde{t}_k s_k+(m-1)K_n\right] \\ &+ f(m-1)^2K_n^2 + K_n\sum_{j=1}^{f}\sum_{i=m}^{s_j}\left[\frac{K_n}{m}\tilde{t}_j i+(m-1)K_n\right] \\ \leqslant{}& nI^m + \frac{K_n}{m}\sum_{j=2}^{f}\tilde{n}_j\sum_{k=1}^{j-1}\tilde{t}_k s_k + (m-1)K_n n\frac{f(f-1)}{2} \\ &+ f(m-1)^2K_n^2 + \frac{K_n^2}{m}\sum_{j=1}^{f}\sum_{i=1}^{s_j}\tilde{t}_j i + (m-1)K_n n \\ \leqslant{}& nI^m + \frac{1}{m}\sum_{j=2}^{f}\tilde{n}_j\sum_{k=1}^{j-1}\tilde{t}_k(\tilde{n}_k+K_n) + (m-1)K_n n\frac{f(f-1)}{2} \\ &+ f(m-1)^2K_n^2 + \frac{K_n^2}{m}\sum_{j=1}^{f}\tilde{t}_j s_j\frac{(s_j+1)}{2} + (m-1)K_n n. \end{aligned}$$

Dividing by $n^2$, taking the limit as the number of jobs, $n$, tends to infinity, recalling the assumption that

$$\lim_{n\to\infty}\frac{K_n}{n} = 0$$

and noting that with probability one,

$$\tilde{p}_j = \lim_{n\to\infty} s_j K_n/n = \lim_{n\to\infty}\tilde{n}_j/n, \quad \forall j = 1, 2, \ldots, f,$$

we get that with probability one,

$$\lim_{n\to\infty}\frac{Z^{SHIFT}}{n^2} \leqslant \lim_{n\to\infty}\frac{I^m}{n} + \frac{1}{m}\left[\sum_{j=1}^{f}\tilde{t}_j\frac{\tilde{p}_j^2}{2} + \sum_{k=2}^{f}\tilde{p}_k\sum_{i=1}^{k-1}\tilde{p}_i\tilde{t}_i\right]. \tag{4}$$

Finally, using Lemma 1, Equation (3), and Equation (4) we get that with probability one

$$\begin{aligned} \lim_{n\to\infty}\frac{Z_1^*}{mn^2} &\leqslant \lim_{n\to\infty}\frac{Z^*}{n^2} \leqslant \lim_{n\to\infty}\frac{Z^{\mathrm{SPT_K}}}{n^2} \\ &\leqslant \lim_{n\to\infty}\frac{I^m}{n} + \lim_{n\to\infty}\frac{Z_1^*}{mn^2}. \end{aligned} \tag{5}$$

Thus, the difference between the lower and upper bounds developed is a function of $I^m$, the idle time on Machine $m$ obtained in the shifted strategy. We characterize this idle time below.

Lemma 2. *For any SPT schedule in the original Cyclic Discrete flow shop problem described above with an associated $K_n$ value, the total idle time on Machine m satisfies $I^m = O(K_n)$.*

Proof. The proof proceeds by induction on the number of machines. We begin with the two-machine case.

Lemma A.2, combined with the definition of a semi-active schedule, tells us that

$$I^2 = \max\left\{0, \max_{l=2,3,\ldots,n} \sum_{k=2}^{l} \left(t_k^1 - t_{k-1}^2\right)\right\}.$$

Given $l$, we analyze the function

$$\sum_{k=2}^{l} \left(t_k^1 - t_{k-1}^2\right).$$

Because the distance between any job and its forward match is no more than $K_n$, the sequence of jobs $1, 2, 3, \ldots, l$ has no more than $fK_n$ jobs, each of which has the property that its forward match is the last job in its family. That is, every job, except for at most $fK_n$ jobs, has a forward match that is its next job type, as defined in Section 2.1.

This, together with the fact that the processing time on a machine is no more than one, implies that

$$\sum_{k=2}^{l} \left(t_k^1 - t_{k-1}^2\right)$$

can never be larger than $fK_n$. Hence, accounting now for the idle time before the first job begins processing, the idle time on the second machine is no larger than $fK_n + 1$.

Next, we assume that $I^{m-1} = O(K_n)$, and we analyze $I^m$. Lemma A.2 tells us that

$$I^m = \max\left\{0, \max_{l=2,3,\ldots,n} \left(I_l^{m-1} + \sum_{k=2}^{l} \left(t_k^{m-1} - t_{k-1}^m\right)\right)\right\}.$$

By the induction hypothesis, we know that the $I^{m-1} = O(K_n)$. Also, by the same argument as above,

$$\sum_{k=2}^{l} \left(t_k^{m-1} - t_{k-1}^m\right)$$

can never be larger than $fK_n$. Hence, accounting for the idle time on Machine $m$ *before* the first job is processed, idle time on Machine $m$ can never be larger than $fK_n + (m-1) + O(K_n) = O(K_n)$. $\square$

To complete the proof of Theorem 2, we utilize Equation (5), Lemma 2, and the assumption,

$$\lim_{n\to\infty} \frac{K_n}{n} = 0.$$

## 3. PROOF OF THE MAIN THEOREM

We prove Theorem 1 by constructing a number of discretized versions of Problem $P$. We begin by discretizing the original problem and removing just enough jobs to obtain a Cyclic Discrete model, described in the previous section. These discretized models allow us to develop an expression for an upper bound on the asymptotic objective value of the SPT ordering associated with an instance of Problem $P$. We show that under the condition stated in Theorem 1, this upper bound on the SPT sequence converges to a lower bound on the optimal value of Problem $P$ developed in Lemma A.1.

### 3.1. Discretization

First, we take the original continuous problem, Problem $P$, and discretize it so there are a finite number of possible *job types*. When discretizing the problem, however, we need to ensure that given an instance of Problem $P$ and an SPT sequence, this sequence remains an SPT ordering in the discretized model. That is, suppose $t_i$ and $t_j$ represent the total processing times of Jobs $i$ and $j$ in Problem $P$, and $t_i \leqslant t_j$. Also, suppose $t_{i_d}$ and $t_{j_d}$ represent the total processing times of the discretized versions of Jobs $i$ and $j$. We must ensure that $t_{i_d} \leqslant t_{j_d}$ for all $i$ and $j$, such that $t_i \leqslant t_j$. To do this, we round up each of the processing times, using the following two-step process.

In Step 1, we begin by subdividing the (0, 1] interval into $s$ subintervals, each of length $\epsilon$. We use $A_l$, $l = 1, 2, \ldots, s$, to denote the $l$th subinterval, that is, $A_l = ((l-1)\epsilon, l\epsilon]$. For every Job $i$ in Problem $P$, $i = 1, 2, \ldots, n$, and Machine $k$, $k = 1, 2, \ldots, m$, such that $t_i^k \in A_l$ for some $l$, $l = 1, 2, \ldots, s$, we round its processing time, $t_i^k$, up to the value $l\epsilon$, and call this new processing time $\bar{t}_i^k$. Let $\bar{t}_i$ be equal to the sum of the rounded processing times of Job $i$. Clearly, this step is not sufficient to ensure that an SPT ordering of the discretized job set is the same as the original sequence.

In Step 2, we utilize the following technique to ensure that the SPT ordering remains the same. For every Job $i$, $i = 1, 2, \ldots, n$, in the original Problem $P$, let $t_i$ be the total processing time of the job *before rounding*. Next, subdivide the interval $(0, m]$ into $ms$ subintervals, each of length $\epsilon$, and define $B_l$ to be the $l$th interval, $l = 1, 2, \ldots, ms$. That is, $B_l = ((l-1)\epsilon, l\epsilon]$. For every Job $i$ in Problem $P$, $i = 1, 2, \ldots, n$, such that $t_i \in B_l$ for some $l, l = 1, 2, \ldots, ms$, let $\tilde{t}_i = l\epsilon$. Clearly, $\tilde{t}_i \leqslant \bar{t}_i$.

To maintain the SPT sequence, every job that has the same associated time $\tilde{t}_i$ as defined above must have the same total processing time in the discretized problem. Define

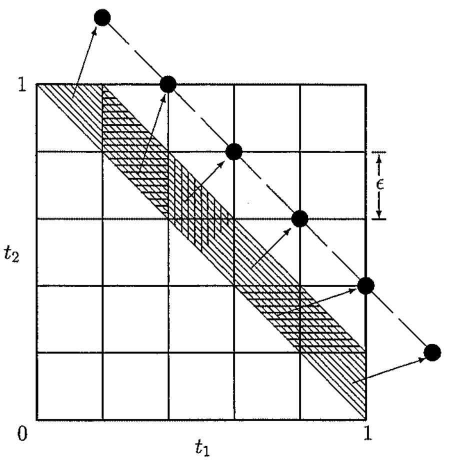
$$r_i = (\bar{t}_i - \tilde{t}_i)/\epsilon,$$

and for each job $i$ in the rounded problem created in the first step, add *an additional $\epsilon$ to the $(m-1) - r_i$ largest unrounded processing times* within that job. That is, given a Job $i$, order its processing time in Problem $P$ on the $m$ machines from the largest to smallest. Now, for the $(m-1) - r_i$ largest processing times of this job in Problem $P$, increase the corresponding processing times in the rounded problem by exactly $\epsilon$. This process is illustrated in Figure 3 for $m = 2$. In this figure, each job is represented by a point on the graph, where the $x$-axis represents processing time on Machine 1, and the $y$ axis represents processing time on Machine 2. At the end of the two-step rounding process described above, all of the points in the shaded area are rounded up to points on the dotted line, as illustrated by the arrows. The remainder of the processing time pairs are rounded in a similar fashion.

Thus, for every Job $i$, the total processing time of the rounded version of that job is,

$$t_{i_d} = \bar{t}_i + ((m-1) - r_i)\epsilon = \tilde{t}_i + (m-1)\epsilon.$$

**Figure 3.** The rounding strategy.



We call this new rounded problem Problem $P_D$, whose optimal objective value is $Z_D^*$.

Given an instance of Problem $P$ and an SPT sequence, construct Problem $P_D$ using the rounding technique described above. This rounding procedure implies that the (single) original SPT sequence associated with Problem $P$ is also one of possibly many SPT orderings of Problem $P_D$. We use $Z_D^{SPT}$ to denote the objective value of this SPT sequence when applied to Problem $P_D$. Clearly,

$$Z^* \leqslant Z^{SPT} \leqslant Z_D^{SPT}. \tag{6}$$

Because in Problem $P_D$, processing times take only discrete values, we can construct an associated *Cyclic Discrete* problem called Problem $P_{CD}$. As in the previous section, let a *job type* be represented by a vector $(t^1, t^2, \ldots, t^m)$. Observe that in Problem $P_D$, every job type has a corresponding vector whose elements $t^k$ satisfy $t^k = l\epsilon$ for every $k$, $k = 1, 2, \ldots, m$, and for some $l$, $l = 1, 2, \ldots, (s+1)$.

In Problem $P_{CD}$, we consider *only* job types from Problem $P_D$ represented by vectors that have *no two equal elements*. We partition the set of all job types from Problem $P_D$ with the above property into groups $g_1, g_2, \ldots, g_G$ and in addition create the remaining job types necessary so that all of these groups are complete. That is, each group must include *all* of the job types that are obtained by a cyclic shift of each one of the others. Clearly, each such group consists of exactly $m$ job types, and all of the job types within a single group correspond to the job types defined in Section 2.1.

Let $n_{g_i}^l$ be the number of jobs in Problem $P_D$ whose processing times are represented by the $l$th job type of group $g_i$, $l = 1, \ldots, m$, and $i = 1, 2, \ldots, G$. Let

$$\tilde{n} = n - \sum_{j=1}^{G}\sum_{l=1}^{m} n_{g_i}^l,$$

that is, $\tilde{n}$ is the number of jobs in Problem $P_D$, each of which has at least two machines on which its processing times are equal.

In the new problem, Problem $P_{CD}$, we assign exactly

$$n_{g_i} = \min_{l=1,\ldots,m} \{n_{g_i}^l\}$$

jobs to each one of the job types associated with group $g_i$. Each job in Problem $P_{CD}$ has a corresponding job in Problem $P_D$.

Let $Z_{CD}^*$ be the optimal solution value of the resulting problem, let $Z_{CD}^{SPT}$ be the objective value of the resulting problem when jobs are sequenced *in the same order* as their corresponding jobs in the SPT sequencing of jobs in Problem $P_D$ and observe that this problem is a *Cyclic Discrete* model, as defined in Section 2.

We note the following relationship between Problem $P_{CD}$ and Problem $P_D$. For each job deleted from the SPT sequencing of Problem $P_D$ to obtain Problem $P_{CD}$, the completion time of each subsequent job in the sequence decreases by no more than $m(1+\epsilon)$. This is true because the processing time on each machine is bounded by one. Because a total of

$$\tilde{n} + \sum_{i=1}^{G}\sum_{l=1}^{m} (n_{g_i}^l - n_{g_i})$$

jobs are deleted, the following relationship holds:

$$Z_{CD}^{SPT} \geqslant Z_D^{SPT} - nm(1+\epsilon)\left[\tilde{n} + \sum_{i=1}^{G}\sum_{l=1}^{m}(n_{g_i}^l - n_{g_i})\right]. \tag{7}$$

Dividing Equation (7) by $n^2$, taking the limit as $n$ goes to infinity, and using Equation (6) we obtain

$$\lim_{n\to\infty} \frac{Z^*}{n^2} \leqslant \lim_{n\to\infty} \frac{Z_{CD}^{SPT}}{n^2} + \lim_{n\to\infty} \frac{1}{n^2} nm(1+\epsilon)\left[\tilde{n} + \sum_{i=1}^{G}\sum_{l=1}^{m}(n_{g_i}^l - n_{g_i})\right].$$

A similar argument to the one employed in Kaminsky and Simchi-Levi (1998) can be used to show that the second term in the above upper bound is almost surely $O(\epsilon)$, and thus, almost surely,

$$\lim_{n\to\infty} \frac{Z^*}{n^2} \leqslant \lim_{n\to\infty} \frac{Z_{CD}^{SPT}}{n^2} + O(\epsilon). \tag{8}$$

Because Problem $P_{CD}$ is a *Cyclic Discrete* problem, we can utilize Theorem 2 to prove the following Lemma.

Lemma 3. *Consider Problem $P_{CD}$, and its SPT ordering. We have with probability one*

$$\lim_{n\to\infty} \frac{Z_{CD}^*}{n^2} = \lim_{n\to\infty} \frac{Z_{CD}^{SPT}}{n^2}.$$

Of course, to apply Theorem 2, the $K_n$ value associated with the specific SPT must have the property that almost surely

$$\lim_{n\to\infty} \frac{K_n}{n} = 0.$$

Indeed, in Appendix B we prove the following result:

Lemma 4. *Consider an arbitrary sequence of jobs whose processing times are generated according to Theorem* 1. *Order the jobs according to the SPT schedule and construct Problem $P_{CD}$ and its associated SPT as described above. The $K_n$ value associated with this SPT satisfies $K_n = o(n)$ almost surely.*

### 3.2. Completing the Proof

To complete the proof we utilize Lemma 3 and Equation (6) and (8) to get

$$\lim_{n\to\infty} \frac{Z^*}{n^2} \leqslant \lim_{n\to\infty} \frac{Z^{SPT}}{n^2} \leqslant \lim_{n\to\infty} \frac{Z^*_{CD}}{n^2} + O(\epsilon). \tag{9}$$

To find an upper bound on $Z^*_{CD}$, recall that every instance of the flow shop mean completion time has an associated single-machine model, as defined in §1. Starting with problem $P_{CD}$, generate a single-machine model, Problem $P_{1CD}$ with optimal objective value $Z^*_{1CD}$, in exactly the same way that Problem $P_1$ was generated from Problem $P$. It follows from Theorem 2 and Lemma 4 that with probability one,

$$\lim_{n\to\infty} \frac{Z^*_{1CD}}{mn^2} = \lim_{n\to\infty} \frac{Z^*_{CD}}{n^2}. \tag{10}$$

Next, we relate $Z^*_{1CD}$ to $Z^*_1$, the optimal solution to Problem $P_1$, the single-machine model associated with Problem $P$. For this purpose, note that each task in Problem $P_{1CD}$ has a corresponding task in Problem $P_1$ (although the opposite is not true). Furthermore, the total processing time of each task in Problem $P_{1CD}$ is no more than $m\epsilon$ larger than the total processing time of its corresponding task in Problem $P_1$. Consequently,

$$Z^*_{1CD} \leqslant Z^*_1 + \frac{n(n+1)}{2} m\epsilon,$$

and this, together with Equations (9) and (10), shows that almost surely:

$$\begin{aligned} \lim_{n\to\infty} \frac{Z^*}{n^2} &\leqslant \lim_{n\to\infty} \frac{Z^{SPT}}{n^2} \leqslant \lim_{n\to\infty} \frac{Z^*_{1CD}}{mn^2} + O(\epsilon) \\ &\leqslant \frac{Z^*_1}{mn^2} + O(\epsilon). \end{aligned} \tag{11}$$

On the other hand, Lemma A.1 tells us that

$$\lim_{n\to\infty} \frac{Z^*}{n^2} \geqslant \frac{Z^*_1}{mn^2}. \tag{12}$$

Thus, combining Equations (11) and (12) and choosing $\epsilon$ small enough show that with probability one,

$$\lim_{n\to\infty} \frac{Z^*}{n^2} = \lim_{n\to\infty} \frac{Z^{SPT}}{n^2}.$$

This completes the proof of Theorem 1. □

## 4. EXTENSIONS AND CONCLUDING REMARKS

To illustrate the effectiveness of the SPT rule, it is important to point out that in Kaminsky and Simchi-Levi (1998), we consider some industrial data that clearly do not conform to all of the parameters of this model. We applied the SPT rule to two-, three-, and six-machine problems with 169, 143, and 112 jobs, respectively. These are small instances so it is not surprising that SPT does not work as well as for larger instances. Indeed, for the two- and six-machine instances, SPT yields solutions with cost about 40% higher than that of a lower bound. However, as we discuss in more detail in the paper by Kaminsky and Simchi-Levi, we suspect that at least some of this gap is attributable to the weakness of the lower bound. For the three-machine instance, SPT performs better, yielding a solution that is about 5% larger than that of the lower bound.

Finally, we note that the analysis performed in this paper can be carried over to a more general version of the Flow Shop Weighted Completion Time Problem in which one is allowed to process jobs on different machines in different sequences, a *nonpermutation* schedule. In addition, the tools of probabilistic modeling have only been applied in a limited way to scheduling problems. In the future, we hope to extend the kinds of approaches demonstrated in this paper to more complex scheduling models.

## APPENDIX A. PRELIMINARY RESULTS

In this section we present several Lemmas and Properties that we refer to throughout the paper.

### A.1. A Lower Bound

Given Problem $P$ as defined in §1, we define its associated single-machine problem as defined in §1. We call this single-machine scheduling problem Problem $P_1$, with optimal solution value $Z^*_1$, the minimum total completion time of all of the tasks. This optimal solution is achieved by sequencing the tasks in Shortest Processing Time first order (see, for example, Pinedo 1995).

Problem $P$ and Problem $P_1$ are related through the following lower bound, whose proof is given in Kaminsky and Simchi-Levi (1998).

Lemma A.1. *Consider Problem $P$, the general Flow Shop Mean Completion Time Problem, and its associated single-machine scheduling problem, Problem $P_1$. For every instance we have*

$$\frac{1}{m} Z^*_1 \leqslant Z^*.$$

### A.2. Total Idle Time

Consider any semiactive permutation sequence of the jobs in the $m$ machine flow shop problem and index the jobs according to their appearance in that sequence. Our objective is to characterize $I^a_j$, $a = 2, \ldots, m$, $j = 2, \ldots, n$, the total idle time incurred on Machine $a$, between the time the first job starts on that machine and the time Job $j$ departs from that machine. We show

Lemma A.2. *For every $j$, $j \geqslant 2$ we have*

$$I_j^a = \max\left\{0, \max_{l=2,3,\dots,j}\left(I_l^{a-1} + \sum_{k=2}^{l}(t_k^{a-1} - t_{k-1}^a)\right)\right\}.$$

We note that because there is no idle time on Machine 1, i.e., $I_j^1 = 0$ for every $j$,

$$I_j^2 = \max\left\{0, \max_{l=2,3,\dots,j}\sum_{k=2}^{l}(t_k^1 - t_{k-1}^2)\right\}.$$

Proof. Define $i_j^a$, $a = 1, \dots, m$, $j = 2, \dots, n$ to be equal to the idle time on Machine $a$ between the completion of Job $j-1$ on Machine $a$ and the completion of Job $j$ on Machine $a$. By definition,

$$I_j^a = \sum_{k=2}^{j} i_k^a.$$

The proof proceeds by induction on $j$. Clearly,

$$I_2^a = \max\{0, I_2^{a-1} + t_2^{a-1} - t_1^a\}.$$

Assume

$$I_i^a = \max\left\{0, \max_{l=2,3,\dots,i}\left(I_l^{a-1} + \sum_{k=2}^{l}(t_k^{a-1} - t_{k-1}^a)\right)\right\},$$

for all $i$, $2 \leqslant i \leqslant j$. We distinguish between two cases.

*Case 1.* Job $j$ starts on Machine $a$ immediately after finishing on Machine $a-1$. Obviously, if $t_{j+1}^{a-1} + i_{j+1}^{a-1} \leqslant t_j^a$, then

$$\begin{aligned} I_{j+1}^a &= I_j^a \\ &= \max\left\{0, \max_{l=2,3,\dots,j}\left(I_l^{a-1} + \sum_{k=2}^{l}(t_k^{a-1} - t_{k-1}^a)\right)\right\} \\ &\geqslant I_{j+1}^{a-1} + \sum_{k=2}^{j+1}(t_k^{a-1} - t_{k-1}^a), \end{aligned}$$

because $t_{j+1}^{a-1} \leqslant t_j^a$. On the other hand, if $t_{j+1}^{a-1} + i_{j+1}^{a-1} > t_j^a$, then,

$$I_{j+1}^a = I_{j+1}^{a-1} + \sum_{k=2}^{j+1}(t_k^{a-1} - t_{k-1}^a).$$

To see why this is true, note that Job 1 starts processing on Machine $a$ immediately after it completes on Machine $a-1$ and that Job $j+1$ starts processing on Machine $a$ immediately after it completes on Machine $a-1$. Thus, the total elapsed time on Machine $a$ between the start of processing of Job 1 and the start of processing of Job $j+1$ is exactly

$$I_{j+1}^{a-1} + \sum_{k=2}^{j+1} t_k^{a-1}.$$

Subtracting the time devoted to processing yields the idle time.

Hence, we need to show that

$$I_{j+1}^{a-1} + \sum_{k=2}^{j+1}(t_k^{a-1} - t_{k-1}^a) \geqslant \max_{l=2,3,\dots,j}\left(I_l^{a-1} + \sum_{k=2}^{l}(t_k^{a-1} - t_{k-1}^a)\right).$$

For this purpose, we identify the latest Job $i$, $i \leqslant j$, whose processing time on Machine $a$ starts after this machine has incurred a delay. By the induction assumption and the fact that no additional idle time is incurred after Job $i$ starts processing until Job $j$ completes processing,

$$\begin{aligned} I_i^a &= \max\left\{0, \max_{l=2,3,\dots,i}\left(I_l^{a-1} + \sum_{k=2}^{l}(t_k^{a-1} - t_{k-1}^a)\right)\right\} \\ &= \max\left\{0, \max_{l=2,3,\dots,j}\left(I_l^{a-1} + \sum_{k=2}^{l}(t_k^{a-1} - t_{k-1}^a)\right)\right\}. \end{aligned}$$

Similarly, because the total time that elapses between the start of processing of Job $i$ on Machine $a$ and the start of processing of Job $j+1$ on Machine $a$ is

$$\sum_{k=i+1}^{j+1} i_k^{a-1} + t_k^{a-1},$$

and it is clear that

$$\sum_{k=i+1}^{j+1}(i_k^{a-1} + t_k^{a-1} - t_{k-1}^a) > 0,$$

and, hence,

$$\begin{aligned} I_{j+1}^a &= I_{j+1}^{a-1} + \sum_{k=2}^{j+1}(t_k^{a-1} - t_{k-1}^a) \\ &= I_i^a + \sum_{k=i+1}^{j+1}(i_k^{a-1} + t_k^{a-1} - t_{k-1}^a) \\ &> I_i^a \\ &\geqslant \max_{l=2,3,\dots,j}\left(I_l^{a-1} + \sum_{k=2}^{l}(t_k^{a-1} - t_{k-1}^a)\right), \end{aligned}$$

where the second equality follows because the second term in the addition captures the idle time that occurs after Job $i$ completes processing on Machine $a$, and the final inequality follows from the induction assumption.

*Case 2.* Job $j$ has to wait in front of Machine $a$ before its processing starts. Let $\delta$ be the amount of time Job $j$ has to wait after finishing on Machine $a-1$ and before being processed on Machine $a$. We consider two cases depending on the value $\delta + t_j^a$. If $\delta + t_j^a \leqslant t_{j+1}^{a-1} + i_{j+1}^{a-1}$, then

$$I_{j+1}^a = I_{j+1}^{a-1} + \sum_{k=2}^{j+1}(t_k^{a-1} - t_{k-1}^a),$$

which, following the same argument as in the second part of the previous case, implies that

$$I^a_{j+1} = \max\left\{0, \max_{l=2,3,\ldots,j+1}\left(I^{a-1}_l + \sum_{k=2}^{l}(t^{a-1}_k - t^a_{k-1})\right)\right\}.$$

On the other hand, if $\delta + t^a_j > t^{a-1}_{j+1} + i^{a-1}_{j+1}$, then

$$I^a_{j+1} = I^a_j.$$

Again, we identify the latest Job $i, i < j$ whose processing time on Machine $a$ starts after this machine has incurred a delay and we use a similar approach to the previous case. Note that in this case, there is no additional idle time on Machine $a$ between the time that Job $i$ starts processing and the time that Job $j+1$ completes processing. By the induction assumption and the choice of Job $i$,

$$I^a_i = \max\left\{0, \max_{l=2,3,\ldots,i}\left(I^{a-1}_l + \sum_{k=2}^{l}(t^{a-1}_k - t^a_{k-1})\right)\right\}$$
$$= \max\left\{0, \max_{l=2,3,\ldots,j}\left(I^{a-1}_l + \sum_{k=2}^{l}(t^{a-1}_k - t^a_{k-1})\right)\right\}.$$

On the other hand, by comparing elapsed times on Machine $a$ and $a-1$ as before,

$$\sum_{k=i+1}^{j+1}(i^{a-1}_k + t^{a-1}_k - t^a_{k-1}) < 0$$

and, hence,

$$I^a_{j+1} = I^a_j$$
$$= \max\left\{0, \max_{l=2,3,\ldots,j}\left(I^{a-1}_l + \sum_{k=2}^{l}(t^{a-1}_k - t^a_{k-1})\right)\right\}$$

by the induction assumption. Hence,

$$I^a_{j+1} = I^a_i$$
$$= I^{a-1}_l + \sum_{k=2}^{i}(t^{a-1}_k - t^a_{k-1}) > I^{a-1}_{j+1} + \sum_{k=2}^{j+1}(t^{a-1}_k - t^a_{k-1}),$$

where the first equality follows from the choice of $i$, and the second equality follows by comparing elapsed time on both machines, as in the previous case. □

## A.3. Useful Inequalities

The following two properties, given here without proof, are used throughout the paper.

PROPERTY A.1. BOOLE'S INEQUALITY *(See, for example, Rohatgi 1976). Consider event $E_i$, $i = 1, 2, \ldots, b$, for some positive integer $b \geqslant 2$. We have*

$$Pr\left(\bigcap_{i=1}^{b} E_i\right) \geqslant 1 - \sum_{i=1}^{b}(1 - Pr(E_i)).$$

PROPERTY A.2. *For any numbers $a, b,$ and $c$ such that $a \geqslant 0$, $b \geqslant 0$, and $0 \leqslant c \leqslant 1$,*

$$1 - (1-a)(1-b)(1-c) \leqslant a + b + c.$$

## APPENDIX B. PROOF OF LEMMA 4

To prove the Lemma, we find for every $n$ large enough a lower bound on the probability that $K_n$, the *maximum distance between any job and its forward match* in the SPT sequence consisting of $n$ jobs, is no more than $D$, where $D = o(n)$. In particular, we show that for $D = o(n)$,

$$\sum_{n=B}^{\infty} Pr(K_n \geqslant D) < \infty,$$

where $B$ is an arbitrary constant. Hence, by the Borel-Cantelli Lemma, we have almost surely $K_n = o(n)$.

Our strategy in calculating the probability $Pr(K_n \geqslant D)$ is to consider three different random variables and then combine them to find our bound. The first random variable concerns $\overline{X}$, the maximum distance between two consecutive jobs in the same group in the SPT sequence.

PROPERTY A.3. *For every $x$ we have*

$$Pr(\overline{X} \leqslant x) \geqslant 1 - n(1-c)^x, \tag{13}$$

*for some constant $c$.*

PROOF. Given Job $i$ in Group $g_j$, let $X^{g_j}_i$ be the distance between Job $i$ and the first job in Group $g_j$ that follows it in the SPT sequence. Given a Job $k$ with total processing time larger than that of Job $i$, and whose family is the one associated with Group $g_j$, let $p^{g_j}_k$ be the conditional probability that this job is a member of Group $g_j$. Because the job processing times are continuous random variables with nonzero densities, there exists a constant $c$ such that $c \leqslant p^{g_i}_k$ for all $k$ and $j$. Thus, we have that

$$Pr(X^{g_j}_i \leqslant x) \geqslant 1 - (1-c)^x.$$

Using Property A.1 we have

$$Pr(\overline{X} \leqslant x) = Pr(X^{g_j}_i \leqslant x, \forall i, j) \geqslant 1 - \sum_{j=1}^{G}\sum_{i=1}^{n_{g_j}}(1-c)^x.$$

Hence,

$$Pr(\overline{X} \leqslant x) \geqslant 1 - n(1-c)^x. \quad \square$$

To introduce the second random variable, employed only when $m \geqslant 3$, we break each group into *subgroups*, one for each pair of job types within a group. Each Subgroup $r$ consists of all jobs within that group that are a job type and its *next job type*, as defined in Section 2.1. This implies that if there are $m$ different machines, there will be exactly $m$ subgroups within a group. Also, each *job type*, and therefore each job, will be in two subgroups; in one it will be the first type, and in another it will be the second type.

We number the jobs within a subgroup consecutively and let $Y^{g_j}_{r_i}$ represent the number of jobs in Group $g_j$ that are sequenced between Job $i$ and Job $i+1$ in Subgroup $r$. Note that we are counting only jobs in Group $g_j$; for this purpose, we ignore all jobs in other groups in the family.

Observe that all job types within a group occur with the same probability. Hence, the conditional probability that a particular job within a group is in a subgroup, equals $2/m$.

Defining $\overline{Y} = \max_{j,r,i} Y_{r_i}^{g_j}$, and using Property A.1 exactly as we did above, we get

PROPERTY A.4.

$$Pr(\overline{Y} \leqslant y) \geqslant 1 - n(1-2/m)^y. \tag{14}$$

For the third random variable, note that given a job, say $i$, in Subgroup $r$ and Group $g_j$, it may be followed by many jobs from the two job types in $r$ until its forward match arrives. Let $W_{r_i}^{g_j}$ represent the number of jobs in Subgroup $r$ that are in between Job $i$ and its *forward match*, as defined in Section 2. Let $\overline{W} = \max_{g_j,r,i} W_{r_i}^{g_j}$. We show

PROPERTY A.5.

$$Pr(\overline{W} \leqslant w) \geqslant 1 - 4Gme^{-\frac{w^2}{8n}}. \tag{15}$$

PROOF. Let $\overline{W}_r^{g_j} = \max_i W_{r_i}^{g_j}$. We start by calculating a lower bound on the probability

$$Pr(\overline{W}_r^{g_j} \leqslant w).$$

For this purpose, consider the job types in the $r$th subgroup of Group $g_j$. We refer to one type as a plus type and the other type as a minus type. Index all the jobs in this subgroup according to their appearance in the SPT sequence. Associated with each such Job $l$ is a random variable $V_l$. The random variable $V_l$ equals 1 when it belongs to the plus job type, and it is equal to $-1$ when it belongs to the minus job type. Let $S_l = \sum_{k=1}^{l} V_l$. It is easy to see that

$$\overline{W}_r^{g_j} = \max_l^{n_{g_j}} |S_l|.$$

The random variable $S_l$ is well understood, see Theorem 2.7 in Coffman and Lueker (1991). They show that

$$Pr\left(\max_{1\leqslant l\leqslant n_{g_j}} S_l \geqslant w\right) \leqslant 2e^{-\frac{w^2}{8n_{g_j}}},$$

and similarly

$$Pr\left(\min_{1\leqslant l\leqslant n_{g_j}} S_l \leqslant -w\right) \leqslant 2e^{-\frac{w^2}{8n_{g_j}}}.$$

Hence, because $n \geqslant n_{g_j}$, we have

$$Pr(\overline{W}_r^{g_j} \leqslant w) \geqslant 1 - 4e^{-\frac{w^2}{8n}}.$$

Finally, using Property A.1 again, we have

$$\begin{aligned} Pr(\overline{W} \leqslant w) &= Pr(\max_{r,g_j} \overline{W}_r^{g_j} \leqslant w) \\ &\geqslant 1 - \sum_{r,g_j}(1 - Pr(W_r^{g_j} \leqslant w) \\ &\geqslant 1 - 4Gme^{-\frac{w^2}{8n}}. \quad \square \end{aligned} \tag{16}$$

To finish the proof, we combine the three random variables as follows. The above upper bounds developed in Equations (13), (14), and (16) imply that for any $x, y$, and $w$ such that $D = xyw$ we have

$$\begin{aligned} Pr(K_n \leqslant D) &\geqslant Pr(\overline{W} \leqslant w)Pr(\overline{Y} \leqslant y)Pr(\overline{X} \leqslant x) \\ &\geqslant (1 - Gm4e^{-\frac{w^2}{8n}})(1 - n(1-2/m)^y) \\ &\quad \cdot (1 - n(1-c)^x), \end{aligned}$$

and thus,

$$\begin{aligned} Pr(K_n \geqslant D) \leqslant 1 - (1 - Gm4e^{-\frac{w^2}{8n}}) \\ \cdot (1 - n(1-2/m)^y)(1 - n(1-c)^x). \end{aligned} \tag{17}$$

Choosing

$$x = C_1 n^{1/10}, y = C_2 n^{1/10}, w = \sqrt{16n \ln n},$$

for some Constants $C_1$ and $C_2$, noting that for these particular values, $D = o(n)$, and utilizing Property A.2, we get

$$\begin{aligned} Pr(K_n \geqslant D) &\leqslant 1 - \left(1 - \frac{4Gm}{n^2}\right) \\ &\quad + 1 - \left(1 - n(1-2/m)^{C_2 n^{1/10}}\right) \\ &\quad + 1 - \left(1 - n(1-c)^{C_1 n^{1/10}}\right) \\ &\leqslant \left(\frac{4Gm}{n^2}\right) + \left(n(1-2/m)^{C_2 n^{1/10}}\right) \\ &\quad + \left(n(1-c)^{C_1 n^{1/10}}\right). \end{aligned}$$

Finally, taking the sum over all $n, n \geqslant B$

$$\begin{aligned} \sum_{n=B}^{\infty} Pr(K_n \geqslant D) \leqslant \sum_{n=B}^{\infty} \frac{4Gm}{n^2} + \sum_{n=B}^{\infty} \left(n(1-2/m)^{C_2 n^{1/10}}\right) \\ + \sum_{n=B}^{\infty} \left(n(1-c)^{C_1 n^{1/10}}\right). \end{aligned}$$

Because each of the terms on the right hand side is finite, the proof is complete. $\square$

## ACKNOWLEDGMENTS

Research in this study was supported in part by ONR Contracts N00014-90-J-1649 and N00014-95-1-0232, NSF Contracts DDM-9322828 and DMI-9732795, and a grant from S&C Electric Corporation.

## REFERENCES

Bhaskaran, K., M. Pinedo. 1992. Dispatching. G. Salvendy, ed. *Handbook of Industrial Engineering*. Wiley, New York, 2184–2198.

Coffman, E. G., G. N. Frederickson, G. S. Lueker. 1982. Probabilistic analysis of the LPT processor scheduling heuristic. M. A. H. Dempster et al., eds. *Deterministic and Stochastic Scheduling*. D. Reidel Publishing Company, 319–331.

Coffman, E. G., G. S. Lueker. 1991. *Probabilistic Analysis of Packing and Partitioning Algorithms* (Interscience series in discrete mathematics and optimization). Wiley, New York.

Frenk, J. B. G., A. H. G. Rinnooy Kan. 1987. The asymptotic optimality of the LPT rule. *Math. Oper. Res.* **12** 241–254.

Garey, M. R., D. S. Johnson, R. Sethi. 1976. The complexity of flowshop and jobshop scheduling. *Math. Oper. Res.* **1** 117–129.

Hall, L. 1997. Approximation algorithms for scheduling. D. Hochbaum, ed. *Approximation Algorithms for NP-Hard Problems*. PWS Publishing, Boston, MA.

Kaminsky, P., D. Simchi-Levi. 1998. Probabilistic analysis and practical algorithms for the flow shop weighted completion time problem. *Oper. Res.* **46** 872–882.

Kohler, W., K. Steiglitz. 1995. Exact, approximate, and guaranteed accuracy algorithms for the flow shop problem n/2/$F$/$\overline{F}$. *J ACM* **22** 106–114.

Krone, M. J., K. Steiglitz. 1974. Heuristic programming solutions of a flowshop scheduling problem. *Oper. Res.* **22** 629–638.

Loulou, R. 1984. Tight bounds and probabilistic analysis of two heuristics for parallel processor scheduling. *Math. Oper. Res.* **9** 142–150.

Morton, T., D. Pentico. 1993. *Heuristic Scheduling Systems*. Wiley, Interscience, New York.

Pinedo, M. 1995. *Scheduling: Theory, Algorithms and Systems*. Prentice Hall, Inc., Englewood Cliffs, NJ.

Ramudhin, A., J. J. Bartholdi, J. Calvin, J. H. Vande Vate, G. Weiss. 1996. A probabilistic analysis of 2-machine flowshops. *Oper. Res.* **44** 889–908.

Rohatgi, V. K. 1976. *An Introduction to Probability Theory and Mathematical Statistics*. Wiley & Sons, New York.

Spaccamela, A. M., W. S. Rhee, L. Stougie, S. van de Geer. 1992. Probabilistic analysis of the minimum weighted flowtime scheduling problem. *Oper. Res. Lett.* **11** 67–71.

Webster, S. 1993. Bounds and asymptotic results for the uniform parallel processor weighted flow time problem. *Oper. Res. Lett.* **41** 186–193.