Scheduling of Tasks with Effectiveness Precedence Constraints

Emily A. Heath[a,3,4], John E. Mitchell[b,1,*], Thomas C. Sharkey[c,1,2]

[a] *The MITRE Corporation, 202 Burlington Rd., Bedford, MA 01730*
[b] *Department of Mathematical Sciences, Rensselaer Polytechnic Institute, 110 8th St., Troy, NY 12180*
[c] *Department of Industrial and Systems Engineering, Rensselaer Polytechnic Institute, 110 8th St., Troy, NY 12180*

Abstract

We formally present the problem of scheduling tasks with effectiveness precedence relationships in order to achieve the minimum total weighted completion time. We provide the problem formulation and define the scope of the problem considered. We present computational complexity results for this problem and an approximation algorithm for it. We prove the theoretical performance of our algorithm and demonstrate its efficiency and practical performance through computational testing, which includes a comparison to the optimal results obtained with an integer programming formulation.

Keywords: scheduling, effectivness precedence constraints, single machine scheduling, total weighted completion time

1. Introduction

An effectiveness precedence relationship exists between two jobs if one job can be completed faster if the other job is completed first. Alternatively, this relationship describes the situation where one job is not as effective (and therefore has a longer processing time) unless another job is completed before it. For clarity, consider an example with job 1 and job 2. Job 1 has a processing time of p_1, and job 2 has a processing time of p_2 if job 1 is completed before it and a processing time of $p_2+\epsilon$, with $\epsilon > 0$ if job 2 is completed before job 1. This work examines scheduling pairs of jobs with effectiveness precedence relationships on a single machine to minimize the total weighted completion time.

An effectiveness precedence relationship for a pair of jobs is defined formally in the following manner:

- Let pair j denote two jobs with an effectiveness precedence relationship.
- Let the first job be denoted j_1 with weight w_{j_1} and processing time p_{j_1}.
- Let the second job be denoted j_2 with weight w_{j_2}.
- Let ϵ_j denote job j_2's excess processing time.
- Let job j_2 have processing time p_{j_2} if it is completed after job j_1 and processing time $p_{j_2} + \epsilon_j$ if it is completed before job j_1.

2. Literature Review

The class of scheduling problems with effectiveness precedence relationships is identified in Sharkey et al. [1] which was classifying the types of scheduling relationships between restoration tasks across infrastructures after Hurricane Sandy. An example of a pair of restoration tasks that have this effectiveness precedence relationship is the restoration of power to a specific area ("job 1" in the pair) and the pumping of floodwater from a subway ("job 2" in the pair). If power is first restored to the area, then mobile and permanent pumps can be used, so the pumping task takes a shorter time to complete. If the pumping task is completed before the restoration of power, then this task is impacted by electrical shortages and takes a longer time to complete because only mobile pumps can be used. Sharkey et al. [1] identifies this new problem in scheduling. Therefore, there are no additional works that refer to this class of scheduling problems, and there are no previous proofs of the complexity of this general class of problem or heuristic solution methods. Our work contributes to the study of this new type of scheduling problem by providing proofs of its complexity in certain cases and developing a fast and efficient approximation algorithm.

There are many related works in the area of single machine scheduling. For a general overview of scheduling,

* Corresponding author
Email address: mitchj@rpi.edu (John E. Mitchell)

[1] The work of these authors was supported in part by the U.S. National Science Foundation under grant number CMMI-1314350.
[2] The work of this author was supported in part by the U.S. National Science Foundation under grant number CMMI-1254258.
[3] The work of this author was supported in part by the Graduate Assistance in Areas of National Need (GAANN) Fellowship from the U.S. Department of Education under grant number P200A120055.
[4] The author's affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions or viewpoints expressed by the author.

the reader is directed to Pinedo [2]. In Graham et al. [3], the authors provide a survey of deterministic scheduling, including previous theoretical results and approximation methods. Additional work on single machine scheduling includes Akker et al. [4] and Waterer et al. [5]. In Akker et al. [4], facet inducing inequalities for the nonpreemptive single-machine scheduling problem are developed and incorporated into a branch-and-cut algorithm. In Waterer et al. [5], the authors relate the time-indexed formulation of the nonpreemptive single machine scheduling problem to the node packing problem. None of these previous works consider our single-machine scheduling problem of jobs with effectiveness precedence relationships.

The objective function considered in our scheduling problem is the minimum total weighted completion time, and there is a variety of previous work that considers this objective function. In Anderson and Potts [6], the authors consider an on-line single machine scheduling problem where the goal is to minimize the total weighted completion time. Anderson and Potts [6] develop an algorithm based on the weighted shortest processing time rule for this setting that has a competitive ratio of 2. Hall et al. [7] consider both on-line and off-line approximation algorithms for scheduling to minimize the average completion time. Hall et al. [7] also consider both the single-machine setting and the parallel machine environment for the problem where jobs have release dates or precedence constraints. Megow and Schulz [8] examine preemptive and nonpreemptive versions of on-line scheduling to minimize average completion time for the identical parallel machine environment. To date, there has been no work on how to schedule jobs with effectiveness precedence relationships in order to obtain the minimum total weighted completion time.

A closely-related area to effectiveness precedence relationship scheduling is precedence constraint scheduling. Correa and Schulz [9] consider scheduling traditional precedence-constrained jobs on a single machine to minimize average weighted completion time, and provide an approximation algorithm for the case where there are series-parallel precedence constraints. Several works also consider AND/OR precedence constraints, which is a more flexible environment where some jobs may be ready to be processed if only one of their predecessor jobs is completed. Examples of papers that consider this environment include Gillies and Liu [10] and Möhring et al. [11]. These works extend algorithms and insights from traditional precedence constraint scheduling and develop heuristic solution methods. Previous work has also considered the OR environment alone, such as Johannes [12] which proves that minimizing total weighted completion time on a single machine for the OR environment is strongly NP-hard. Although this work is closely related, an effectiveness precedence relationship cannot be described in a traditional AND or a traditional OR setting. Therefore, our work on this class of scheduling problems contributes to another area in precedence constraint scheduling.

3. Problem Statement

For the scope of this work, we also place several other restrictions on the overall scheduling problem:

- There are n pairs of jobs with effectiveness precedence relationships.
- Pairs of jobs have no precedence relationships between each other.
- All weights, processing times, and excess processing times are positive.
- Both jobs in a pair must be processed before another pair is processed, similar to chain restrictions in Pinedo [2].
- There is a single machine available to process the jobs.

For a single pair j of jobs, there are two options for scheduling: the jobs can be scheduled in the order $j_1 - j_2$, denoted as the "fast order," or the jobs can be scheduled in the order $j_2 - j_1$, denoted as the "slow order." The optimal option for this single pair depends upon the weights of the two jobs and their processing times. If pair j is one pair in a larger schedule of pairs of jobs, then selecting the "slow order" has additional costs if pair j is not the last pair in the schedule. All pairs scheduled after j will have their completion time pushed back by ϵ_j, so selecting the optimal option for j depends upon the weights of the jobs processed after it.

Scheduling pairs of jobs with effectiveness precedence relationships requires determining the order of the pairs of jobs, and whether each pair is processed in the fast or slow order. For notation, let σ denote an ordering of the pairs of jobs, and let $\sigma(k < j)$ denote the set of pairs of jobs processed before pair j. The notation $\sigma(k < j, \text{slow})$ refers to the set of pairs processed before pair j and processed in the slow order. The objective function of total weighted completion time can be written as the sum of four components $\alpha + \beta + \gamma + \delta$, where each component is defined as follows:

$$\alpha = \sum_{j=1}^{n} \sum_{\sigma(k<j)} (w_{j_1} + w_{j_2})(p_{k_1} + p_{k_2}) \quad (1)$$

$$\beta = \sum_{j=1}^{n} (w_{j_1} + w_{j_2})(p_{j_1} + p_{j_2}) \quad (2)$$

$$\gamma = -\sum_{j=1}^{n} \begin{cases} w_{j_1} p_{j_2} \text{ (if fast)} \\ w_{j_2} p_{j_1} - \epsilon_j (w_{j_1} + w_{j_2}) \text{ (if slow)} \end{cases} \quad (3)$$

$$\delta = \sum_{j=1}^{n} \sum_{\sigma(k<j,\text{slow})} (w_{j_1} + w_{j_2})\epsilon_k \quad (4)$$

The α component of the objective function is a standard formulation of the objective of weighted completion time, assuming the pairs have total weight $w_{j_1} + w_{j_2}$ and total processing time $p_{j_1} + p_{j_2}$. The β component of the objective function captures a required quantity in the objective function value, regardless of the order and the slow or fast decisions. The γ component provides an adjustment to the β quantity based on the slow or fast decision for each pair. The δ component provides the addition to the α component that is a result of the slow pairs in the schedule and their impact on the pairs that are processed after them. This definition of the objective function is used in the following Section 4, where we prove the computational complexity.

4. Computational Complexity

To determine the computational complexity of this class of problems, the following inequality must be satisfied in order to consider scheduling pair j in the slow order:

$$\epsilon_j < \frac{w_{j_2}p_{j_1} - w_{j_1}p_{j_2}}{w_{j_1} + w_{j_2}} \tag{5}$$

When this inequality holds for a pair of jobs, it is possible that it is better to process the pair in the slow order depending on where it is in the schedule. If equality holds for a pair, then it can be processed in either the fast order or the slow order if it is the last pair in the schedule. If an individual pair of jobs does not satisfy (5), then it is never better to process this pair in the slow order. This result allows us to determine the set of polynomially solvable cases.

Theorem 1. *If no more than $k\log_2(n)$ job pairs satisfy (5) for some constant k, then the optimal schedule can be determined in polynomial time.*

Proof. All job pairs that do not satisfy (5) will be scheduled in the fast order. For the remaining pairs of jobs, we consider both the fast and slow option, so there are $2^{k\log_2(n)} = n^k$ potential schedules. Each schedule is optimally ordered using the weighted shortest processing time (WSPT) rule, a well-known result in scheduling theory. The schedule with the lowest objective function value is selected as the final optimal solution. The WSPT rule is a polynomial time rule, and this rule is used n^k times, which is polynomial in n. Therefore, this problem with no more than $k\log_2(n)$ job pairs satisfying (5) for some constant k can be solved in polynomial time. □

From now on, we assume all pairs of jobs satisfy (5). We prove that the general problem of scheduling pairs of jobs with effectiveness precedence relationships that satisfy (5) on a single machine is NP-complete. To prove that this problem is NP-complete, we focus on the decision version of the effectiveness precedence relationship problem: given an input value M and pairs of jobs with effectiveness precedence relationships, does there exist a single machine schedule with an objective function value less than or equal to M. Our NP-complete proof reduces the partition problem to our problem.

Theorem 2. *Determining whether there exists a schedule of jobs with effectiveness precedence constraints that obtains an objective function value of weighted completion time no greater than M is NP-complete.*

Proof. We first prove that the effectiveness precedence scheduling problem is in NP. Given a schedule, we check whether all pairs are processed and whether both jobs in each pair are processed before the next pair is processed to determine if it is a feasible schedule. If it is a feasible schedule, we compute the objective function value of the schedule by finding the sum of weighted completion times for each job. We determine whether this objective function value is less than or equal to M. It is trivial to show that these two steps can be completed in polynomial time.

We prove the effectiveness precedence scheduling problem is NP-hard by reducing the partition problem to it. The partition problem is defined as follows: given n positive integers $a_1, a_2, \ldots, a_n$ with sum $C = \sum_{j=1}^{n} a_j$, does there exist a subset $I \subset \{1, \ldots, n\}$ with $\sum_{j\in I} a_j = \frac{1}{2}C$? Given an instance of the partition problem, we construct an instance of the effectiveness precedence scheduling problem in the following manner:

$$w_{j_1} = \zeta a_j \tag{6a}$$

$$w_{j_2} = (1-\zeta)a_j \tag{6b}$$

$$p_{j_1} = \frac{1}{2} + \frac{1}{2}\mu a_j \tag{6c}$$

$$p_{j_2} = (1 - \frac{1}{2}\mu)a_j - \frac{1}{2} \tag{6d}$$

$$\epsilon_j = \frac{1}{2}\mu a_j \tag{6e}$$

where

$$\mu = \frac{2}{C} \tag{7a}$$

$$\zeta = \frac{1}{2C} \tag{7b}$$

If $C < 2a_i$ for any i, then the partition instance is infeasible. If any $a_i = \frac{1}{2}C$, then the partition instance is feasible and trivial. Therefore, we assume without loss of generality that each $a_i < \frac{1}{2}C$. With the given choice of ζ and μ, we verify that (5) is satisfied:

$$\epsilon_j = \frac{1}{2}\mu a_j$$

$$\frac{w_{j_2}p_{j_1} - w_{j_1}p_{j_2}}{w_{j_1} + w_{j_2}} = \frac{\frac{1}{2} + \frac{1}{2}\mu a_j - \zeta a_j}{1}$$

$$\frac{1}{2}\mu a_j < \frac{\frac{1}{2} + \frac{1}{2}\mu a_j - \zeta a_j}{1} \text{ because } a_j < \frac{1}{2}C.$$

For each pair of jobs, we define two ratios: r_j, the ratio of weight to processing time for a pair processed in the fast order, and s_j, the ratio of weight to processing time for a pair processed in the slow order. For each pair of jobs, r_j and s_j are as follows:

$$r_j = \frac{w_{j_1} + w_{j_2}}{p_{j_1} + p_{j_2}} = 1 \quad \text{for } j = 1, \ldots, n \tag{8}$$

$$s_j \;=\; \frac{w_{j_1}+w_{j_2}}{p_{j_1}+p_{j_2}+\epsilon_j} = \frac{1}{1+\frac{1}{2}\mu} \quad \text{for } j = 1,\ldots,n \tag{9}$$

Recall that the objective function value of a schedule of pairs of jobs is given by $\alpha+\beta+\gamma+\delta$. Recall that σ denotes an ordering of the pairs of jobs. Note that β is independent of any particular ordering σ, and is equal to the following in our problem:

$$\beta \;=\; \sum_{j=1}^{n} a_j^2 \tag{10}$$

We also note that α is independent of any particular ordering σ with our reduction, as it corresponds to a standard formulation for the sum of weighted completion times over the pairs of jobs where each pair has weight $w_{j_1}+w_{j_2}$ and processing time $p_{j_1}+p_{j_2}$. The value of α therefore depends upon an ordering of the pairs based on the r_j ratios. Since every pair has an r_j value of 1 in our reduction, α is independent of any particular ordering. For our problem, it is equal to the following:

$$\alpha \;=\; \sum_{j=1}^{n} \sum_{\sigma(k<j)} a_j a_k \;=\; \frac{1}{2}C^2 \;-\; \frac{1}{2}\beta, \tag{11}$$

Let S denote the set of job pairs that are processed in the slow order. Let $\sigma(k \geq j)$ denote the set of jobs, including pair j, processed after pair j. The remainder of the objective function value $\gamma+\delta$ is as follows:

$$\gamma+\delta$$

$$= -\sum_{j\notin S} w_{j_1}p_{j_2} - \sum_{j\in S} w_{j_2}p_{j_1} + \sum_{j\in S} e_j \sum_{\sigma(k\geq j)} (w_{k_1}+w_{k_2}) \tag{12}$$

$$= -\sum_{j=1}^{n} w_{j_1}p_{j_2} - \sum_{j\in S}(w_{j_2}p_{j_1} - w_{j_1}p_{j_2}) + \sum_{j\in S} e_j \sum_{\sigma(k\geq j)} (w_{k_1}+w_{k_2}) \tag{13}$$

$$= -\sum_{j=1}^{n}\left(\zeta a_j^2 - \frac{1}{2}\mu\zeta a_j^2 - \frac{\zeta}{2}a_j\right) - \sum_{j\in S}\left(\frac{1}{2}a_j + \frac{\mu}{2}a_j^2 - \zeta a_j^2\right) + \frac{1}{2}\mu \sum_{\substack{j,k\in S\\ \sigma(k\geq j)}} a_j a_k \tag{14}$$

$$= -\sum_{j=1}^{n}\left(\zeta a_j^2 - \frac{1}{2}\mu\zeta a_j^2 - \frac{\zeta}{2}a_j\right) - \frac{1}{2}\sum_{j\in S} a_j - \left(\frac{\mu}{2}-\zeta\right)\sum_{j\in S} a_j^2 + \frac{1}{2}\mu \sum_{\substack{j,k\in S\\ \sigma(k\geq j)}} a_j a_k \tag{15}$$

$$= -\sum_{j=1}^{n}\left(\zeta a_j^2 - \frac{1}{2}\mu\zeta a_j^2 - \frac{\zeta}{2}a_j\right) - \frac{1}{2}\sum_{j\in S} a_j + \zeta\left(\sum_{j\in S} a_j\right)^2 \tag{16}$$

since $\zeta = \frac{1}{4}\mu$. This objective is fixed everywhere except in the term $\sum_{j\in S} a_j$. Considering that the objective function is a quadratic function of this term, the overall objective is minimized when $\sum_{j\in S} a_j = \frac{C}{2}$, with a value of

$$M := \frac{1}{2}C^2 + \left(\frac{C^2-C+1}{2C^2}\right)\sum_{j=1}^{n} a_j^2 - \frac{C}{8} + \frac{1}{4} \tag{17}$$

Therefore, the partition problem has a solution if and only if the effectiveness precedence problem has a solution with M defined in (17). Because we have already shown that the effectiveness precedence problem is in NP, it follows that it is NP-complete. □

5. Approximation Algorithm Results

We propose an approximation algorithm Pair First Scheduling (PFS). The algorithm works in two parts. The first part determines the order of the pairs of the jobs, and the second part determines the order of each pair. This algorithm determines a schedule with an objective function value guaranteed to be within a factor of two of the optimal. We present PFS formally by Algorithm 0.

1: Order Pairs: For all pairs j compute the following ratio: $\frac{w_{j_1}+w_{j_2}}{p_{j_1}+p_{j_2}}$ and order the pairs from greatest ratio to smallest ratio

2: Determine Order of Pairs: For each pair j, determine whether the following inequality holds:

$$w_{j_1}p_{j_1} + w_{j_2}(p_{j_1}+p_{j_2}) \leq w_{j_2}(p_{j_2}+e_j) + w_{j_1}(p_{j_2}+\epsilon_j+p_{j_1}) + \epsilon_j \sum_{\sigma(k>j)} (w_{k_1}+w_{k_2}) \tag{18}$$

If (18) holds, process the job in the fast order $j_1 - j_2$. Otherwise, process in the slow order $j_2 - j_1$.

Theorem 3. *PFS generates a schedule for pairs of jobs with effectiveness precedence constraints that obtains an objective function value of total weighted completion time within a factor of two of the optimal objective function value.*

Proof. Assume there are n pairs of jobs that satisfy the previously given assumptions, where each pair j has weights w_{j_1}, w_{j_2}, processing times p_{j_1}, p_{j_2}, and excess processing time ϵ_j. Recall that the objective function of total weighted completion time can be written as the sum of four components $\alpha+\beta+\gamma+\delta$. Let the schedule generated by PFS be denoted by $\widehat{PFS}$. To show that $\widehat{PFS}$ has an objective function value no more than two times the optimal objective function value, we first establish a lower bound on the objective function value for any feasible schedule. To establish a lower bound, we modify $\widehat{PFS}$ to obtain a new schedule $\overline{PFS}$. To obtain the schedule $\overline{PFS}$ from $\widehat{PFS}$, we use the same ordering of pairs but ensure that every pair j is processed in the slow order. Using $\overline{PFS}$, we can obtain a lower bound on the objective function value for any schedule.

Lower bound: Let $\bar{\alpha}, \bar{\beta}, \bar{\gamma}, \bar{\delta}$ be the objective function values for $\overline{PFS}$. By ordering the pairs according to step 1 in PFS, $\overline{PFS}$ minimizes $\alpha + \beta$ in the objective function value. This minimization is due to the well-known result in scheduling theory that the WSPT rule is optimal for a single machine weighted-completion time objective when scheduling jobs with chain precedence constraints. By processing all pairs using the extended processing times, $\overline{PFS}$ also minimizes γ. This minimization is a result of the fact that (5) holds for all pairs: $w_{j_2}p_{j_1} - \epsilon_j(w_{j_1} + w_{j_2}) < w_{j_1}p_{j_2}$ for all pairs. We have that $\bar{\alpha} + \bar{\beta} + \bar{\gamma}$ is a minimum over all $\alpha + \beta + \gamma$. Note that for any schedule, $\delta \geq 0$. Therefore, the value of any ordering is greater than or equal to $\bar{\alpha} + \bar{\beta} + \bar{\gamma}$ and this quantity provides a lower bound on the objective function value for any valid schedule.

To show that $\widehat{PFS}$ obtains an objective function value within a factor of two of the optimal objective function value, we first show that $\overline{PFS}$ obtains an objective function value within a factor of two of the lower bound and therefore of the optimal objective function value.

Upper bound: $\overline{PFS}$ is a valid schedule because it processes every pair and processes both jobs in all pairs before proceeding to the next pair. Therefore, its objective function value provides an upper bound of $\bar{\alpha} + \bar{\beta} + \bar{\gamma} + \bar{\delta}$. From (5), we know that $\epsilon_j < p_{j_1}$ for all pairs j. Therefore, we know that $\bar{\delta} \leq \bar{\alpha}$ because $(w_{j_1} + w_{j_2})\epsilon_k < (w_{j_1} + w_{j_2})(p_{k_1} + p_{k_2})$ for all pairs j and k. Because $\bar{\delta} \leq \bar{\alpha}$, we know that $\bar{\alpha} + \bar{\delta} \leq 2\bar{\alpha}$. Also, we note that $\beta + \gamma \geq 0$ for any ordering as we can rewrite this quantity in the following manner:

$$\sum_{j=1}^{n}(w_{j_1} + w_{j_2})(p_{j_1} + p_{j_2}) + \begin{cases} -w_{j_1}p_{j_2} \text{ (if fast)} \\ -w_{j_2}p_{j_1} + \epsilon_j(w_{j_1} + w_{j_2}) \text{ (if slow)} \end{cases}$$

Then we know that $\bar{\beta} + \bar{\gamma} \geq 0$, and that $\bar{\beta} + \bar{\gamma} \leq 2(\bar{\beta} + \bar{\gamma})$. From here, we can conclude that $\bar{\alpha} + \bar{\beta} + \bar{\gamma} + \bar{\delta} \leq 2(\bar{\alpha} + \bar{\beta} + \bar{\gamma})$. Since we have shown that $\bar{\alpha} + \bar{\beta} + \bar{\gamma}$ is a lower bound on the optimal objective function value, we have shown that $\overline{PFS}$ obtains an objective function value that is within a factor of two of the optimal objective function value. It now remains to show that $\widehat{PFS}$ obtains an objective function value that is at least as good as $\overline{PFS}$. If the schedule for $\widehat{PFS}$ is the same as the schedule for $\overline{PFS}$, then we are done. Assume without loss of generality that one pair of jobs, denoted by index k, is processed in the fast order in $\widehat{PFS}$. It is important to note that because we have assumed all pairs of jobs satisfy (5), then we know that this pair of jobs cannot be the last pair processed, but it can be in any other slot. If this pair of jobs is processed in the fast order, then we know that it satisfies (18). When comparing the objective function values for $\widehat{PFS}$ and $\overline{PFS}$, we note that since the order of the pairs of the jobs was determined in the same way for both schedules, $\hat{\alpha} + \hat{\beta} = \bar{\alpha} + \bar{\beta}$. The difference in the objective function values is determined by the differences in $\hat{\gamma} + \hat{\delta}$ and $\bar{\gamma} + \bar{\delta}$. Because pair k was processed in the fast order, $\hat{\gamma}$ is no longer minimized. However, $\hat{\delta}$ will contain fewer terms than $\bar{\delta}$ because pair k will not be a pair processed before any other pair in the slow order. By summing all of the terms in $\bar{\delta}$ that do not appear in $\hat{\delta}$ and considering the difference in $\hat{\gamma}$ and $\bar{\gamma}$, the differences in the objective function values for $\widehat{PFS}$ and $\overline{PFS}$ are as follows:

Objective terms in $\widehat{PFS}$: $\hat{\gamma} + \hat{\delta} = -w_{k_1}p_{k_2}$

Objective terms in $\overline{PFS}$: $\bar{\gamma} + \bar{\delta} = -(w_{k_2}p_{k_1} - \epsilon_k(w_{k_1} + w_{k_2})) + \epsilon_k \sum_{\sigma(j>k)}(w_{j_1} + w_{j_2})$

As stated previously, this pair k satisfies (18). With some rearranging of (18), we have that

$-w_{k_1}p_{k_2} \leq -(w_{k_2}p_{k_1} - \epsilon_k(w_{k_1} + w_{k_2})) + \epsilon_k \sum_{\sigma(j>k)}(w_{j_1} + w_{j_2})$.

Therefore,

$\hat{\gamma} + \hat{\delta} \leq \bar{\gamma} + \bar{\delta}$, and $\hat{\alpha} + \hat{\beta} + \hat{\gamma} + \hat{\delta} \leq \bar{\alpha} + \bar{\beta} + \bar{\gamma} + \bar{\delta}$.

We conclude that the objective function value for $\widehat{PFS}$ is at least as good as the objective function value for $\overline{PFS}$. Since the objective function value for $\overline{PFS}$ is within a factor of two of the optimal objective function value, it is also the case that the objective function value for $\widehat{PFS}$ is within a factor of two of the optimal objective function value. □

To show PFS's practical performance, we conducted computational testing. We generated 10 instances of problems with 100, 200, 400, and 800 pairs of jobs. We construct the weights and processing times using a method similar to the one used in the NP-complete proof. Note that in this generation procedure, each pair of jobs will have the same fast order r ratios and slow order s ratios as defined in the NP-complete proof, which also means that both the α and β components of the objective function value are independent of a specific ordering.

To compare PFS's objective function value and schedule, we computed the optimal objective function value and schedule by solving an integer programming (IP) formulation of the effectiveness precedence scheduling problem. The IP formulation is solved using CPLEX 12.6 with an overall time limit of six hours. After 6 hours, the best IP solution found is reported as the final solution even if it is not the optimal solution. To compare results, we report the averages across the 10 size instances of the PFS runtime, the IP runtime, difference in objective function values, and the percentage gap in difference between objective function values. Runtime is denoted by "RT." To obtain a more accurate depiction of the percentage gap between objective function values, we modified each objective function value for comparison by removing the α and β components from the IP and PFS solutions. The results for the different job sizes are given in Table 1.

Table 1: PFS Computational Testing Results

Size	PFS RT(s)	IP RT(s)	Obj. Diff	% Gap
100	0.02	3.17	44.76	0.1826294
200	0.049	395.12	183.89	0.37
400	0.007	14676.58	67.67	0.067
800	0.031	–	–	–

The dashes in the final row of the results indicate that the IP solution could not be determined within the six hour time limit. There are several conclusions to draw from these results. The runtime for PFS is always very quick, while the run for the IP increases dramatically with an increase in size. Additionally, its differences in terms of percentage gaps are very small. However, the PFS schedule is almost always completely wrong. This result is due to the fact that in this generation procedure, all pairs of jobs are tied in terms of their fast order r ratios. PFS therefore orders pairs from lowest index to highest index, and the optimal schedule is almost always completely different. It is also important to note that the percentage gaps are much larger than indicated by the differences in objective function values. This result is due to the fact that the α component of the objective is a substantial portion of the overall objective function value, and this component is subtracted before computing the percentage gaps. However, these percentage gaps are still very small. These results support the idea that PFS is a good approximation algorithm to use for this class of scheduling problems. Solving the IP formulation for large problem sizes can take a long time, and does not necessarily lead to a significantly better solution in terms of objective function value. PFS can be run very quickly and produces good quality solutions with an objective function value within a factor of 10^{-4} of optimal.

6. Conclusion

We provided significant contributions to a new class of scheduling problems with effectiveness precedence relationships. We have shown that this general class of problems, with a set of assumptions, is NP-complete using a reduction from the partition problem. We have developed an approximation algorithm and demonstrated that it performs very well in terms of percentage gap differences between the optimal solutions. There are many directions for future work, including considering a multi-machine setting, precedence or other relationships between pairs, and removing the restriction to process both jobs in a pair before considering the next pair.

References

[1] T. Sharkey, S. Nurre, H. Nguyen, J. Chow, J. Mitchell, W. Wallace, Identification and classification of restoration interdependencies in the wake of Hurricane Sandy, Journal of Infrastructure Systems 22 (1) (2016) 1–12.

[2] M. Pinedo, Scheduling: Theory, Algorithms, and Systems, Springer, New York, New York, 2012.

[3] R. Graham, E. Lawler, J. Lenstra, A. R. Kan, Optimization and approximation in deterministic sequencing and scheduling: A survey, Annals of Discrete Mathematics 5 (1979) 287–326.

[4] J. v. d. Akker, C. van Hoesel, M. Savelsbergh, A polyhedral approach to single-machine scheduling problems, Mathematical Programming 85 (3) (1999) 541–572.

[5] H. Waterer, E. Johnson, P. Nobili, M. Savelsbergh, The relation of time indexed formulations of single machine scheduling problems to the node packing problem, Mathematical Programming 93 (2002) 477–494.

[6] E. Anderson, C. Potts, On-line scheduling of a single machine to minimize total weighted completion time, Mathematics of Operations Research 29 (2004) 686–697.

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

[8] N. Megow, A. Schulz, On-line scheduling to minimize average completion time revisited, Operations Research Letters 32 (2004) 485–490.

[9] J. Correa, A. Schulz, Single-machine scheduling with precedence constraints, Mathematics of Operations Research 30 (2005) 1005–1021.

[10] D. Gillies, J. Liu, Scheduling tasks with AND/OR precedence constraints, SIAM Journal on Computing 24 (1995) 797–810.

[11] R. Mohring, M. Skutella, F. Stork, Scheduling with AND/OR precedence constraints, SIAM Journal on Computing 33 (2004) 393–415.

[12] B. Johannes, On the complexity of scheduling unit-time jobs with OR-precedence constraints, Operations Research Letters 33 (2005) 587–596.