Single machine scheduling to minimize a modified total late work function with multiple due dates

R. Bryan Kethley\textsuperscript{a*}, Bahram Alidaee\textsuperscript{b} and Haibo Wang\textsuperscript{c}

\textsuperscript{a}Management and Marketing Department, Jones College of Business, Middle Tennessee State University, MTSU, P.O. Box X75, Murfreesboro, TN 37132, USA; \textsuperscript{b}School of Business Administration, University of Mississippi, 325 Holman, University, MS 38677, USA; \textsuperscript{c}Sanchez School of Business, Texas A&M International University, PH 313 M, Laredo, TX 78045, USA

(Received 4 November 2013; accepted 24 July 2014)

In this paper, the problem of scheduling $n$ jobs on a single machine is evaluated. We introduce a Modified Total Late Work function that allows a declining piecewise linear penalty over time. Multiple due dates for each job are identified and at each due date, the penalty is reassessed. Many instances occur when delivery urgency leads to progress metrics or dates being added to a contract. Over the life of the contract, a loss of incentives or a price decrease can occur at each of the negotiated due dates. After the final due date, the product is valueless and the penalty ceases to accrue. The number and placement of contract due dates are part of the initial negotiations. Two metaheuristics, Simulated Annealing and the Noising Method, are empirically evaluated to determine their performance when measured by this new loss function.

**Keywords:** deferral costs; total late work; Simulated Annealing; Noising Method

**Introduction**

**Literature review**

The Total Late Work problem was initially presented by Błażewicz (1984) as information loss. Błażewicz (1984) considered the problem of sensors collecting data for parallel processors. If the control system receives the data after a set deadline, it results in a complete data loss. Potts and Van Wassenhove (1992a) suggested the term Total Late Work because of the function’s generalizability to other applications. Applications include information loss in parallel processors (Błażewicz, 1984), identical and uniform processors (Błażewicz & Finke, 1987), single machine scheduling (Hariri, Potts, & Van Wassenhove, 1995; Kethley & Alidaee, 2002; Potts & Van Wassenhove, 1992a), multiple machine scheduling (Błażewicz, Pesch, Sterna, & Werner, 2005a, 2005b, 2007, 2008; Lin, Lin, & Lee, 2006), open-shop scheduling (Błażewicz, Pesch, Sterna, & Werner, 2004), and flexible manufacturing system scheduling (Sterna, 2007).

Kethley and Alidaee (2002) modified the late work function to include end user determined deadlines and due dates. The penalty still progresses linearly but the length of time the penalty can be assessed is not bounded by the job’s processing time. Another application is the harvesting of crops (Potts & Van Wassenhove, 1992a). Crops are harvested according to a deadline and after the deadline, the crops are no longer

\*Corresponding author. Email: bkethley@mtsu.edu

© 2014 The Author(s). Published by Taylor & Francis. This is an Open Access article distributed under the terms of the Creative Commons Attribution License http://creativecommons.org/licenses/by/3.0/, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. The moral rights of the named author(s) have been asserted.
viable. A recent application of the late work function is by Sterna (2007). Sterna (2007) indicates that scheduling to minimize late work is appropriate in a flexible manufacturing system (FMS) to foster customer goodwill. A customer always desires delivery on time, so a planner can use the late work function in FMS shift scheduling to help achieve this goal.

Another penalty function that models a constant linear penalty is the Deferral Costs loss function (McNaughton, 1959). In the Deferral Costs function, there are relative and absolute deadlines. A job progresses without penalty up to the relative deadline, and then a linear penalty is assigned between the relative and absolute deadlines. The maximum penalty is accrued at the absolute deadline. McNaughton’s (1959) research included single processors and a linear penalty function. Lawler (1964) extended McNaughton’s (1959) Deferral Costs work to include multiprocessor and nonlinear cases.

Traditional late work function for machine scheduling

Potts and Van Wassenhove (1992a) make the following assumptions regarding the late work problem. There are ‘n’ jobs (1 … n) which include each job to be processed. For example, a scheduling problem with 50 jobs would be contained in the set (1 … 50). For each job within the set, a processing time ‘pi’ and a due date ‘di’ would be assigned. An individual job is considered early if the job is completed before the due date and no penalty is assigned. The job is considered partially early if the job is completed after the due date but before the due date plus processing time, and a progressive penalty is assigned. A job is considered late if the job is completed after the due date plus the processing time. At this point, the penalty ceases and the job is considered unusable. The sum of each individual job penalty is considered the total penalty for the set of jobs is determined (Equation (1)).

$$g_i(V_i) = \begin{cases} 
0, & C_i \leq d_i \\
C_i - d_i, & d_i < C_i < d_i + p_i \\
p_i, & d_i + p_i \leq C_i 
\end{cases} \quad (1)$$

where $g_i(V_i)$ = late work penalty for job $i$; $C_i$ = completion date for job $i$; $d_i$ = due date for job $i$; and $p_i$ = processing time for job $i$.

In the function identified by Equation (1), the maximum penalty is set as the processing time of job $i$ and the penalty increases linearly until the job is late.

Pinedo (1995) indicates that in practice the penalty function associated with a scheduling policy may not follow the function in Equation 1. Jobs that are completed before their due dates are assigned no penalty, and jobs that are finished after their due dates are assigned a penalty that increases at a given rate. Within the penalty function, the job reaches a point where the penalty assignment changes and increases at a much slower pace. Pinedo’s (1995) function is theoretical and is undefined.

Two functions that react similarly are the deferral cost function (Lawler, 1964) and the Total Late Work function depicted by Figure 1 (Błażewicz, 1984; Błażewicz & Finke, 1987; Potts & Van Wassenhove 1992a, 1992b; Sterna, 2007). In the following research, the penalty assessed was either as a linear fixed rate (Błażewicz, 1984; Błażewicz & Finke, 1987; McNaughton, 1959; Potts & Van Wassenhove, 1992a, 1992b; Sterna, 2007) or at a nonlinear rate (Lawler, 1964). Both functions utilize a single absolute deadline with a single penalty structure.
In this research, we propose a Modified Total Late Work (MTLW) function that applies a penalty in linear segments with multiple due dates. A feature of the MTLW is the inclusion of different penalty schedules that are assigned to each due date. In the MTLW function, if the completion time is before due date 1, then no penalty is assessed. At each of the following due dates, the penalty is assessed in decreasing order (see Equation (2) for an example of a five stage function) (Figure 2).

**Proposed Modified Total Late Work function for machine scheduling**

![Total Late Work penalty function.](image1)

Figure 1. Total Late Work penalty function.

![MTLW penalty function.](image2)

Figure 2. MTLW penalty function.
Previous research (Blążewicz, 1984; Blążewicz & Finke, 1987; Kethley & Alidaee, 2002; McNaughton, 1959; Potts & Van Wassenhove, 1992a, 1992b; Sterna, 2007) has suggested that the penalty accrued is constant through the processing of the good. We propose that the penalty in some cases may not be fixed but can vary over the processing time of the good or service. Delivery urgency may lead to the inclusion of progress points or checks within a contract, where the price decreases or a negotiated penalty may change. Loss of this incentive can occur throughout the life of the contract. In the proposed loss function, the penalty decreases linearly and is adjusted at preselected points until the last due date, at which time the product is no longer viable, and the penalty ceases. In the MTLW loss function given in Equation (2), there are four separate due dates but in practice, the number of due dates are set by the model user.

Lawler (1977) has shown the Total Weighted Tardiness (TWT) problem to be strongly NP-hard. Since the TWT is a special case of the more general function evaluated within this paper, this function is also assumed to be strongly NP-hard. While the theorems and proofs developed by McNaughton (1959) and Lawler (1964) are elegant, they only address solving small problems and do not consider the application of weights for jobs.

The purpose of the modified function presented in this research is to provide an alternative to the basic late function that only considers one processing time, due date, and a single linear penalty. It is the intent of our research to provide a function that will allow many changes in the penalty function with multiple due dates that approximates the shape of the functions presented by Pinedo (1995) and Lawler (1964). This function could easily be incorporated as an objective function in various heuristics, algorithms, and metaheuristics.

Other potential applications

In addition to the machine scheduling context proposed within this research, it is possible to visualize other applications for the function. One such application could be to maximize a function such as profit (Chen & Hall, 2008; Yang, 2009). The proposed function could also be modified to model the increase in productivity. In traditional learning curve literature, the time per unit decreases as more units are built and experience is gained. Time per unit decreases can be modeled as productivity increases. Carlson (1973) suggests that this learning is not constant but may follow a cubic model. This could indicate that productivity is not a linear function but could progress in a decreasing piecewise function similar to the function proposed within this research.

Research motivation

Production & Manufacturing Research: An Open Access Journal

627
Learning does not only impact productivity but may also impact quality (Levin, 2000). As learning increases over time, the quality of the product also increases. Schneiderman (1988) suggests that quality or continuous improvement increases with the application of a ‘QIP’ or quality improvement process, and he further indicates that the rate of improvement changes over time.

Some other applications, where the MTLW may be appropriate, include the depreciation of equipment that may be accelerated with the majority of the depreciation occurring during the initial period of depreciation, supplier payment schedules that include incentives for early repayment, and investment instruments or structured payments that are adjusted periodically.

Metaheuristic evaluation and selection
Kethley (1997) performed multiple simulations with various problem sets to determine the efficacy of several different algorithms within a single machine context. He used the Total Weighted Late Work function as the objective function for the various algorithms. Heuristics and Algorithms, studied by Kethley (1997), included the Genetic Algorithm using random keys (Bean, 1994), Simulated Annealing (SA), (Matsuo, Suh, & Sullivan, 1989), Neighborhood Job Search (Golden & Stewart, 1985; Pinedo, 1995), Space Smoothing Gu and Huang (1994) with Neighborhood Job Search, and the Noising Method (NM; Charon & Hudry, 1993) with Neighborhood Job Search. With each simulation instance, the weighting factor is reduced until the weighted parameters reach the original parameter values.

Kethley (1997) documented the minimum penalty, maximum penalty, average penalty, and standard deviation. While the Neighborhood Job Search performed well, SA and the NM with the Neighborhood Job Search were by far superior in the majority of instances. Because the proposed function given here is a modified version of the Total Weighted Late Work function used by Kethley (1997), the Authors chose the SA and the NM with the Neighborhood Job Search for analysis. In depth, descriptions of these methods follow.

**Neighborhood Job Search**

Neighborhood search techniques attempt, through an iterative process, to find a solution that is progressively better than the previous solution (Golden & Stewart, 1985; Pinedo, 1995). Pinedo (1995) states that ‘two schedules are neighbors if one can be obtained through a well-defined modification of the other.’ A new schedule is achieved by a pairwise exchange of jobs from the initial schedule and the search is continued until a local minimum is reached. The steps of our Neighborhood Job Search are as follows.

*Step 1.* Generate an initial solution seed using the shortest processing time heuristic (Smith, 1956). For an example of the Shortest Processing Time heuristic, consider the following set of job processing times (8,1,5,4,3,2,7,6). With the Shortest Processing Time heuristic the order would be (1,2,3,4,5,6,7,8), (Smith, 1956). Processing times are placed in ascending order.

*Step 2.* Calculate a loss or penalty using the MTLW loss function.  
*Step 3.* Perform a neighborhood exchange of jobs and calculate a penalty for the resulting job sequence. If the new job sequence results in a lower penalty, the new sequence replaces the old sequence.  
*Step 4.* Repeat step 3 until no improvement is possible.
A shortcoming of descending methods is the local optimal may not be equal to the global optimal. The metaheuristics evaluated in this research apply processes in addition to the local search to help the search move from a local minimum to other problem search spaces.

The solution obtained by the local search is used in two ways within this paper. First, the local minimum solution is used in conjunction with SA and NM to develop solutions during the parameter development process and secondly, the solution is used as an initial seed solution for the SA and NM during the 50, 150, and 250 job simulations.

**Noising Method**

The NM has been applied with good results to various combinatorial problems including graph portioning and the Traveling Salesman Problem and scheduling (Charon & Hudry, 2001). The NM, when applied with a local search algorithm, can improve the solution given by the local search alone. Charon and Hudry’s (1993) approach to the local optima problem is to identify an initial solution and through an iterative process, modify or add ‘noise’ to the parameters of the current solution to obtain a new search space. At each step, a decent procedure, that can be applied to the original or noised parameters, is invoked and the best solution is retained. As the number of steps increase, the amount of noise is decreased until at the last iteration the noising factor is at zero. The noising of individual parameters is accomplished using Equation (3).

\[ P'_i = P_i + r \times \text{rate} \times \text{maxval} \]  

\((3)\)

\(P'_i\) is the parameter that is noised. In our research, we apply noise to different parameter combinations to evaluate the effectiveness of the noising schedule. \(P_i\) is the original parameter, \(r\) is a random number, \(\text{rate}\) is the rate of noising perturbation, in our case 0–1, and \(\text{maxval}\) is the maximum value of perturbation. The noising rate decreases from 1 to 0 in a series of steps until the noise applied is zero and the original parameter is obtained. The steps to our NM algorithm are as follows:

**Step 1.** Generate initial solution using the Shortest Processing Time heuristic.

**Step 2.** Using the Neighborhood Job Search described above, search the job sequence until a local minimum solution is achieved.

**Step 3.** Apply noise to the parameters and using a neighborhood selection technique, search the job sequence until a local minimum solution is achieved. Using the noised sequence, calculate a penalty with non-noised parameters. If the new penalty is lower than the best penalty retained, then the noised sequence is the new best penalty. Note that as the noise is decreased by the noising schedule identified in the Methodology section, the parameters reach their original values.

**Step 4.** Decrease noise according to noising schedule and perform Step 3. Continue Step 4 for the number of steps specified by the noising schedule.

**Step 5.** Using the Neighborhood Job Search described, above search the job sequence until a local minimum solution is achieved.

**Simulated Annealing**

Another approach that can be used to navigate through local optima is SA. SA was developed independently by Kirkpatrick, Gelatt, and Vecchi (1983) and Černý (1985).
The method is based on the process of solid material changes as a material cools over time, Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller (1953).

SA has been widely used in a scheduling context. A recent search for SA scheduling applications on ABI Inform resulted in over 200 citations from the past 20 years. SA is a local search technique that allows a solution to move from a local minimum to another search space. At each stage of the algorithm, the objective function is evaluated. If there is an improvement over the previous value, the new solution is retained. If there is no improvement, an acceptance probability is generated that allows retention of an inferior solution. At the initial stages, the probability is set higher to allow a greater chance of retaining inferior solutions but as the algorithm progresses, the probability of keeping a poor solution approaches zero. This sampling of other solution spaces allows SA to escape from a local minimum and theoretically given enough time, SA will progress to a global minimum. A common acceptance probability is given in Equation (4), where $C_j - C_i$ indicates the change in the objective function, $k$ is the stage of the search, and $\beta$ is a control parameter (Matsuo, Suh, & Sullivan, 1989).

$$AP_{ij}(k) = \min\{1, \exp(-\frac{(C_j - C_i)}{\beta k})\}$$

(4)

Traditional SA using a random seed and a high probability of acceptance based on the change in the objective function may lead to high computational time and slow convergence. Matsuo et al. (1989) propose a controlled search that begins with a good solution from a heuristic instead of a random initial solution and acceptance probabilities that are not tied to a change in the objective function. Matsuo et al. (1989) found that for the TWT problem, the linear acceptance probability shown in Equation (5) outperformed the other probabilities studied.

$$AP_{ij}(k) = p - (\text{step} \times k)$$

(5)

Matsuo et al. (1989) set the acceptance probability $p$ at .5, decreasing the probability over 25 steps by .02 with the final probability equal to zero. In our analysis, we will empirically test different probability and iteration combinations to select input parameters for $p$ and the number of steps or stages. The steps of our SA algorithm are as follows:

**Step 1.** Generate initial solution using the Shortest Processing Time heuristic (Smith, 1956).

**Step 2.** Using the Neighborhood Job Search described above, search the job sequence until a local minimum solution is achieved.

**Step 3.** Select a new solution by performing the Neighborhood Job Search described above. If the reordering results in a better solution, retain the new sequence. If the reordering provides an inferior solution, select an acceptance probability. Retain the inferior solution if a random number is less than the acceptance probability.

**Step 4.** Decrease acceptance probability by the empirically derived step and perform Step 3. Continue Step 4 for the number of steps specified.

**Step 5.** Using the Neighborhood Job Search described above, search the job sequence until a local minimum solution is achieved.

**Methodology**

**Problem parameter space**

The development of the input parameters for processing time, weights, and due dates are as follows. The processing time is randomly selected according to a uniform distribution
between 30 and 80. Weight 1 is selected from a range between 12 and 15, weight 2 is between 6 and 9 and weight 3 is set between 2 and 5. These parameters were set by the Authors. Due date 1 was developed from a uniform distribution of Equation (6) (Potts & Van Wassenhove, 1982).

\[
(MS * (1 - T + R/2)), (MS * (1 - T - R/2))
\]  

(6)

MS is the make span of the problem, T is the tardiness factor, and R is the Range. The tardiness factors used in this research are .2, .6, and 1.0 and the range factors used are .2, .4, .6, and .8, giving 12 shop conditions that vary in level of difficulty (Potts and Van Wassenhove, 1982). Subsequent due dates are developed using the last due date plus a random portion of the due date range. For example, Due date 2 is derived using Due date 1 + Random Number * (Due date 1 – Processing time 1), and Due date 3 is calculated using Due date 2 + Random Number * (Due date 2 – Due date 1). This process continues according to the number of due dates identified by the user. The loss function included within this paper has three weights and four due dates that were developed by the Authors using the methods described above. In practice, the number of weights and due dates could vary and would be set by the model end user.

Codenotti, Manzini, Margara, and Resta (1996) suggest that a good strategy for developing a good solution is in choosing ‘(i) the length of \( t \) close to the length of \( s \) and (ii) \( t \) is quite different from a local optimum (in particular from \( s \)).’ It is important to strike a balance between modifying the parameter search space \( t \) to the point that it no longer a representative of the initial parameter \( s \), and a search space that is too close to the original search space of \( s \).

In the NM, we attempt to accomplish this balance by empirically analyzing the performance of the algorithm as different parameter noising schedules and the number of steps or stages is applied. In the SA, the parameters of interest are the acceptance probability and the number of annealing stages. In the following sections, the process of developing parameters for the NM and SA is discussed.

**NM parameter selection**

Seven parameter, combinations consisting of (1) due date, (2) processing time and due date, (3) weight and due date, (4) processing time, weight and due date, (5) processing time, (6) weight and processing time, and (7) weight, were each simulated for 25, 50, 75, 100, 125, 150, 175, 200, 225, and 250 iterations. This resulted in 70 combinations and each combination was evaluated using 30 different problem sets. Preliminary results indicated that values from 0, −1 did not perform well as a noising range. Noised parameters were developed using the noising range of 0, 1 with a uniform distribution. The results of the 30 simulations were averaged and the improvement over the Neighborhood Job Search was computed. The average improvement over the local minimum that was reached with the Neighborhood Job Search was collected and plotted in Figure 3.

Figure 3 suggests that noising the due dates, alone and in various combinations with other parameters, produces a poor solution. Applying noise to the processing times and the weights produces better improvement with the best improvement occurring at 175 noising iterations and noise applied to the weight parameter. From the initial simulations, it was decided to set the noising parameters at 175 iterations and applying noise only to the weights for the evaluation of the 50, 150, and 250 job problems.
SA parameter selection

Five parameter combinations consisting of the probabilities .5, .6, .7, .8, and .9 were simulated for 5, 10, 15, 20, 25, 30, 35, 40, 45, and 50 iterations each. This resulted in 50 combinations and each combination was evaluated using 30 different problem sets. The results of the 30 simulations were averaged and the improvement over the

Figure 3. NM improvement over Neighborhood Job Search.

Figure 4. SA improvement over Neighborhood Job Search.

SA parameter selection

Five parameter combinations consisting of the probabilities .5, .6, .7, .8, and .9 were simulated for 5, 10, 15, 20, 25, 30, 35, 40, 45, and 50 iterations each. This resulted in 50 combinations and each combination was evaluated using 30 different problem sets. The results of the 30 simulations were averaged and the improvement over the
Table 1. Simulation results for 50 Jobs.

| $T$ | $R = 0.2$ |     | $T$ | $R = 0.6$ |     | $T$ | $R = 1.0$ |     |
|-----|-----------|-----|-----|-----------|-----|-----|-----------|-----|
|     | SA        | NM  |     | SA        | NM  |     | SA        | NM  |
| .2  | Minimum   | 3072| .2  | Minimum   | 0   | .2  | Minimum   | 0   |
|     | Maximum   | 12,292|    | Maximum   | 1949|    | Maximum   | 0   |
|     | Std Dev   | 2034|    | Std Dev   | 356 |    | Std Dev   | 0   |
|     | Average   | 6914|    | Average   | 65  |    | Average   | 0   |
| .4  | Minimum   | 26,716|   | Minimum   | 1251|   | Minimum   | 0   |
|     | Maximum   | 57,300|  | Maximum   | 25,269|  | Maximum   | 7723|
|     | Std Dev   | 7650|    | Std Dev   | 4415|    | Std Dev   | 1704|
|     | Average   | 40,286|  | Average   | 8428|  | Average   | 1677|
| .6  | Minimum   | 52,027|   | Minimum   | 20,485|   | Minimum   | 2335|
|     | Maximum   | 102,066|  | Maximum   | 52,351|  | Maximum   | 22,789|
|     | Std Dev   | 14,974|  | Std Dev   | 8901 |  | Std Dev   | 4762|
|     | Average   | 77,752|  | Average   | 34,384|  | Average   | 9062|
| .8  | Minimum   | 55,970|   | Minimum   | 26,590|   | Minimum   | 10,992|
|     | Maximum   | 100,146|  | Maximum   | 67,944|  | Maximum   | 48,944|
|     | Std Dev   | 9673 |   | Std Dev   | 10,942|  | Std Dev   | 8205|
|     | Average   | 77,421|  | Average   | 43,015|  | Average   | 26,884|
Table 2. Simulation results for 150 Jobs.

| R | T   | SA  | NM  | R | T   | SA  | NM  | R  | T   | SA  | NM  |
|---|-----|-----|-----|---|-----|-----|-----|----|-----|-----|-----|
|   | .2  | 34,569 | 34,569 | .6 | 32,952 | 38,280 | .1 | 0 | 0 | 0 | 0 |
|   | Maximum | 63,805 | 63,926 |   | Maximum | 100,666 | 90,528 |   | Maximum | 112,445 | 121,031 |
|   | Std Dev | 7728 | 7796 |   | Std Dev | 13,827 | 13,781 |   | Std Dev | 27,147 | 16,393 |
|   | Average | 45,262 | 45,227 |   | Average | 61,552 | 66,772 |   | Average | 10,168 | 57,420 |
|   | .4  | 267,108 | 263,317 | .8 | 294,921 | 285,724 | .6 | 36,872 | 33,009 |
|   | Maximum | 441,712 | 435,488 |   | Maximum | 376,014 | 421,494 |   | Maximum | 21,590 | 23,780 |
|   | Std Dev | 39,407 | 39,758 |   | Std Dev | 39,128 | 42,756 |   | Std Dev | 7634 | 4572 |
|   | Average | 334,331 | 331,616 |   | Average | 61,552 | 66,772 |   | Average | 10,168 | 57,420 |
|   | .6  | 565,397 | 563,902 | .8 | 598,642 | 598,642 | .8 | 124,612 | 128,710 |
|   | Maximum | 922,639 | 918,209 |   | Maximum | 537,014 | 546,059 |   | Maximum | 252,028 | 260,908 |
|   | Std Dev | 84,580 | 85,068 |   | Std Dev | 39,128 | 42,756 |   | Std Dev | 27,237 | 33,563 |
|   | Average | 733,679 | 732,619 |   | Average | 282,416 | 307,111 |   | Average | 73,953 | 67,446 |
Table 3. Simulation results for 250 Jobs.

| T  | R = .2 |        | T  | R = .6 |        | T  | R = 1.0 |
|----|--------|--------|----|--------|--------|----|--------|
|    | SA     | NM     |    | SA     | NM     |    | SA     |
| .2 | Minimum| 94,913 | 93,575 | .2 | Minimum| 0     | 0     | .2 | Minimum| 0     | 0     |
|    | Maximum| 180,424| 176,370 |    | Maximum| 0     | 1908  |    | Maximum| 0     | 4014  |
|    | Std Dev| 19,818 | 20,070 |    | Std Dev| 0     | 444   |    | Std Dev| 0     | 982   |
|    | Average| 123,529| 122,301 |    | Average| 0     | 224   |    | Average| 0     | 508   |
| .4 | Minimum| 703,639| 703,301 | .4 | Minimum| 110,378| 140,126| .4 | Minimum| 1511  | 56    |
|    | Maximum| 1,136,523| 1,135,321 |    | Maximum| 203,767| 288,798|    | Maximum| 68,498| 44,488|
|    | Std Dev| 101,550| 99,490 |    | Std Dev| 30,022| 34,712 |    | Std Dev| 22,422| 12,760|
|    | Average| 933,664| 929,063 |    | Average| 155,880| 198,785|    | Average| 28,587| 12,170|
| .6 | Minimum| 1,729,967| 1,720,229 | .6 | Minimum| 569,131| 632,915| .6 | Minimum| 107,319| 109,086|
|    | Maximum| 2,254,768| 2,214,570 |    | Maximum| 949,801| 1,011,790|    | Maximum| 342,368| 282,109|
|    | Std Dev| 136,061| 135,179 |    | Std Dev| 80,374| 84,588 |    | Std Dev| 46,055| 45,160|
|    | Average| 1,991,244| 1,983,672 |    | Average| 745,303| 834,664 |    | Average| 203,149| 182,375|
| .8 | Minimum| 1,974,567| 1,975,371 | .8 | Minimum| 783,435| 807,010 | .8 | Minimum| 467,816| 472,181|
|    | Maximum| 2,432,395| 2,469,005 |    | Maximum| 1,354,473| 1,360,021|    | Maximum| 663,121| 733,618|
|    | Std Dev| 101,463| 103,019 |    | Std Dev| 144,007| 140,894 |    | Std Dev| 59,179| 74,217|
|    | Average| 2,141,245| 2,134,398 |    | Average| 1,055,998| 1,057,114|    | Average| 563,851| 587,846|
Neighborhood Job Search was computed. The improvement in the objective function above that of the Neighborhood Job Search is summarized in Figure 4.

In general, Figure 4 indicates improvement as the probability and number of SA stages increase. The results of the preliminary study indicate that the initial parameter settings should be .9 as the probability and the number of stages should be 45 for the 50, 150, and 250 job problem simulation.

Experimental results

Each simulation consisted of 30 different problem sets for the tardiness factors of .2, .6, and 1.0 and the range factors of .2, .4, .6, and .8 for a total of 12 job shop combinations. Separate simulations were conducted for all 12 combinations for 50 jobs, 150 jobs, and 250 jobs. The minimum value, maximum value, standard deviation, and average loss is documented for each simulation in Table 1 of 50 jobs, Table 2 for 150 jobs, and Table 3 for 250 jobs.

For the combinations given by the tardiness factors of .2, .4, .6, and .8 and the range of .2, the NM performs better than SA. For the combinations of tardiness factors .2, .4, and .6 and the range of .6, along with the combination of \( T = .2 \) and \( R = 1.0 \), SA is the best performer. This pattern holds true for 50, 150, and 250 jobs. The combinations of tardiness factors .4, .6, and .8 and the range of 1.0, along with the combination of \( T = .8 \) and \( R = .6 \), provide inconclusive results.

Table 4. Penalty summary for 50 Jobs, \( T = .2, .4, .6, \) and \( .8; \) and \( R = .2, .6, \) and 1.0.

| \( T \) | \( R \) |
|-----|-----|
| .2  | .2  | .6  | 1.0 |
| .2  | NM  | SA  | SA  |
| .4  | NM  | SA  | SA  |
| .6  | NM  | SA  | SA  |
| .8  | NM  | NM  | NM  |

Table 5. Penalty summary for 150 Jobs, \( T = .2, .4, .6, \) and \( .8; \) and \( R = .2, .6, \) and 1.0.

| \( T \) | \( R \) |
|-----|-----|
| .2  | .2  | .6  | 1.0 |
| .2  | NM  | SA  | SA  |
| .4  | NM  | SA  | NM  |
| .6  | NM  | SA  | NM  |
| .8  | NM  | NM  | SA  |

Table 6. Penalty summary for 250 Jobs, \( T = .2, .4, .6, \) and \( .8; \) and \( R = .2, .6, \) and 1.0.

| \( T \) | \( R \) |
|-----|-----|
| .2  | .2  | .6  | 1.0 |
| .2  | NM  | SA  | SA  |
| .4  | NM  | SA  | NM  |
| .6  | NM  | SA  | NM  |
| .8  | NM  | SA  | SA  |
Algorithm performance for each tardiness and range combinations outcomes are summarized in Tables 4–6. When the tardiness factor is larger and the range is smaller, the NM tends to provide a better solution. Conversely, SA tends to outperform the NM as the tardiness factor decreases and the range increases. This relationship remains constant for 50, 150, and 250 jobs. At the largest values of the tardiness factor and the range factor, the results are not conclusive.

Concluding remarks

In this research, we introduce a new loss function that is an extension of the deferral cost function (Lawler, 1964; McNaughton, 1959) and the late work function (Błażewicz, 1984; Błażewicz & Finke, 1987; Potts & Van Wassenhove, 1992a, 1992b). In the previous research, a due date and a relative due date was used that provided a single linear or nonlinear penalty. The MTLW function is different in that it allows the assignment of multiple due dates and results in a declining piecewise linear penalty. At each due date, the penalty structure changes until the last due date at which time the penalty cease.

In practice, instances may occur when delivery urgency leads to progress payments that are linked to multiple due dates. Placement of these due dates and the associated penalty can be part of the initial contract development. Over time, a loss of incentives or a price decrease can occur at each of the negotiated due dates and after the final due date, the product is valueless and the penalty ceases to accrue.

The MTLW function is used within this research to minimize a penalty in a single machine context but in other applications, it could be used to maximize utility. Some possible future areas for research include organizational learning, productivity improvement, quality and continuous improvement, depreciation of equipment, supplier payment schedules, and investment instruments or structured payments that are adjusted periodically.

Two metaheuristics that have been previously applied to various combinatorial applications, SA and the NM were empirically evaluated to determine their performance when measured by this new loss function. The metaheuristics were simulated using 12 problem sets with different degrees of difficulty for 50, 150, and 250 jobs. The results indicate both algorithms result in similar outcomes; however, when the tardiness factor is larger and the range is smaller, the NM tends to provide a better solution. SA tends to outperform the NM as the tardiness factor decreases and the range increases.

Funding

Bryan Kethley research was supported by a Non-Instructional Grant from Middle Tennessee State University.

References

Bean, J. C. (1994). Genetic algorithms and random keys for sequencing and optimization. *ORSA Journal on Computing*, 6, 154–160.

Błażewicz, J. (1984). Scheduling preemptible tasks on parallel processors with information loss. *Recherche Technique et. Science Informatiques*, 3, 415–420.

Błażewicz, J., & Finke, G. (1987). Minimizing mean weighted execution time loss on identical and uniform processors. *Information Processing Letters*, 24, 259–263. doi:10.1016/0020-0190(87)90145-1
Błażewicz, J., Pesch, E., Sterna, M., & Werner, F. (2004). Open shop scheduling problems with late work criteria. *Discrete Applied Mathematics, 134*, 1–24. doi:10.1016/S0166-218X(03)00339-1

Błażewicz, J., Pesch, E., Sterna, M., & Werner, F. (2005a). The two-machine flow-shop problem with weighted late work criterion and common due date. *European Journal of Operational Research, 165*, 408–415. doi:10.1016/j.ejor.2004.04.011

Błażewicz, J., Pesch, E., Sterna, M., & Werner, F. (2005b). A comparison of solution procedures for two-machine flow shop scheduling with late work criterion. *Computers & Industrial Engineering, 49*, 611–624. doi:10.1016/j.cie.2005.09.001

Błażewicz, J., Pesch, E., Sterna, M., & Werner, F. (2007). A note on the two machine job shop with the weighted late work criterion. *Journal of Scheduling, 10*, 87–95. doi:10.1007/s10951-006-0005-5

Błażewicz, J., Pesch, E., Sterna, M., & Werner, F. (2008). Metaheuristic approaches for the two-machine flow-shop problem with weighted late work criterion and common due dates. *Computers & Operations Research, 35*, 574–599. doi:10.1016/j.cor.2006.03.021

Carlson, J. G. H. (1973). Cubic learning curves: Precision tool for labor estimating. *Carlson, J. G. H.* (1973). Cubic learning curves: Precision tool for labor estimating. *Journal of Industrial and Engineering, 49*, 624. doi:10.1016/j.cie.2005.09.001

Černý, V. (1985). Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications, 45*, 41–51. doi:10.1007/BF00940812

Charon, I., & Hudry, O. (1993). The noising method: A new method for combinatorial optimization. *Operations Research Letters, 14*, 133–137. doi:10.1016/0167-6377(93)90023-A

Charon, I., & Hudry, O. (2001). The noising methods: A generalization of some metaheuristics. *European Journal of Operational Research, 135*, 86–101. doi:10.1016/S0377-2217(00)00305-2

Chen, Z., & Hall, N. G. (2008). Maximum profit scheduling. *Manufacturing & Service Operations Management, 10*, 84–107. doi:10.1287/msom.1060.0144

Codenotti, B., Manzini, G., Margara, L., & Resta, G. (1996). Perturbation: An efficient technique for the solution of very large instances of the Euclidean TSP. *INFORMS Journal on Computing, 8*, 125–133. doi:10.1287/ijoc.8.2.125

Golden, B. L., & Stewart, W. R. (1985). Empirical analysis of heuristics. In E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, & D. B. Shomys (Eds.), *The traveling salesman problem* (pp. 207–249). Wiley. Retrieved from Worldcat.org database.

Gu, J., & Huang, X. (1994). Efficient local search with space smoothing: A case study of the traveling salesman (TSP). *IEEE Transactions on Systems, Management and Cybernetics, 24*, 728–735. doi:10.1109/21.293486

Hariri, A. M. A., Potts, C. N., & Van Wassenhove, L. N. (1995). Single machine scheduling to minimize total weighted late work. *ORSA Journal on Computing, 7*, 232–242. doi:10.1287/ijoc.7.2.232

Kethley, R. B. (1997). Single and multiple machine scheduling to minimize total weighted late work: An empirical comparison of scheduling rules, algorithms and meta-heuristics using Taguchi loss functions (ProQuest Dissertations and Theses, 304360766). Order No. 9816973, The University of Mississippi, Oxford Mississippi.

Kethley, R. B., & Alidaee, B. (2002). Single machine scheduling to minimize total weighted late work: A comparison of scheduling rules and search algorithms. *Computers & Industrial Engineering, 43*, 509–528. doi:10.1016/S0360-8352(02)00123-7

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science, 220*, 671–680. doi:10.1126/science.220.4598.671

Lawler, E. L. (1964). On scheduling problems with deferral costs. *Management Science, 11*, 280–288. doi:10.1287/mnsc.11.2.280

Lawler, E. L. (1977). A “Pseudopolynomial” algorithm for sequencing jobs to minimize total tardiness. *Annals of Discrete Mathematics, 1*, 331–342. doi:10.1016/S0167-5060(08)70742-8

Levin, D. Z. (2000). Organizational learning and the transfer of knowledge: An investigation of quality improvement. *Organization Science, 11*, 630–647. doi:10.1287/orsc.11.6.630.12535

Lin, B. M. T., Lin, F. C., & Lee, R. C. T. (2006). Two-machine flow-shop scheduling to minimize total late work. *Engineering Optimization, 38*, 501–509. doi:10.1080/030521050500420439

Matsuo, H., Suh, C. J., & Sullivan, R. S. (1989). A controlled search simulated annealing method for the single machine weighted tardiness problem. *Annals of Operations Research, 21*, 85–108. doi:10.1007/BF02022094
McNaughton, R. (1959). Scheduling with deadlines and loss functions. *Management Science, 6* (1), 1–12. doi:10.1287/mnsc.6.1.1

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., & Teller, E. (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics, 21*, 1087–1090. doi:10.1063/1.1699114

Pinedo, M. (1995). *Scheduling: Theory, algorithms and systems*. Upper Saddle River, NJ: Prentice Hall. doi:10.1007/978-1-4614-2361-4

Potts, C. N., & Van Wassenhove, L. N. (1982). A decomposition algorithm for the single machine total tardiness problem. *Operations Research Letters, 5*, 177–181.

Potts, C. N., & Van Wassenhove, L. N. (1992a). Single machine scheduling to minimize total late work. *Operations Research, 40*, 586–595. doi:10.1287/opre.40.3.586

Potts, C. N., & Van Wassenhove, L. N. (1992b). Approximation algorithms for scheduling a single machine to minimize total late work. *Operations Research Letters, 11*, 261–266. doi:10.1016/0167-6377(92)90001-J

Schneiderman, A. M. (1988). Setting quality goals. *Quality Progress, 21*, 51–57. Retrieved from ABI/Inform database.

Smith, W. E. (1956). Various optimizers for single-stage production. *Naval Research Logistics Quarterly, 3*, 59–66. doi:10.1002/nav.3800030106

Sterna, M. (2007). Late work minimization in a small flexible manufacturing system. *Computers & Industrial Engineering, 52*, 210–228. doi:10.1016/j.cie.2006.12.004

Yang, W. (2009). Scheduling jobs on a single machine to maximize the total revenue. *Computers & Operations Research, 36*, 565–583. doi:10.1016/j.cor.2007.10.01.
