23rd International Congress on Modelling and Simulation, Canberra, ACT, Australia, 1 to 6 December 2019
mssanz.org.au/modsim2019

# A heuristic for the generalized due-date min-max earliness-tardiness problem

**D. Oron**[a] **and A. Salehipour** [b]

[a]*The University of Sydney Business School, The University of Sydney, NSW 2006, Australia*
[b]*School of Mathematical and Physical Sciences, University of Technology, NSW 2007, Australia*
*Email: amir.salehipour@uts.edu.au*

**Abstract:** We study a single machine scheduling problem with generalized due-dates (GDD) with the goal of minimizing the maximum earliness-tardiness. Under the assumption of GDD the due-dates are not job-dependent but rather associated with the number of jobs previously processed. We develop an efficient heuristic for the problem. We compare the heuristic and the well-known shortest processing time (SPT) and longest processing time (LPT) dispatching rules, as well as a random rule. The extensive numerical tests indicate that the heuristic performs very well.

***Keywords:*** *Single machine, earliness, tardiness, generalized due-dates, heuristic*

## 1 INTRODUCTION

There exist many real life applications where the due-dates, i.e., the deadline for completing jobs are job-independent. Therefore, each job must be assigned to a due date. Under this environment the problem is referred to as scheduling with generalized due-dates (GDD). The GDD is typically used, for example, when the number of jobs completed by a due-date is important rather than the identity of the jobs that have been processed. Consider the following application from Sriskandarajah (1990). A transportation company requires a certain number of buses to be serviced by a set of due-dates. The service times, i.e., jobs' duration depends on the condition of the bus, e.g., mileage, age, depreciation, etc., but it is immaterial to the company which bus is available at which due-date.

The first study to consider scheduling with GDD is Hall (1986). The study provides several applications where one would consider GDD, such as public utility companies that are tasked with providing services to several cities, surveyors collecting customer information for market research, and airline companies that are required to plan flight routes according to licenses in their possession. The author shows that while minimizing the total tardiness and the number of tardy jobs can be performed in polynomial time, minimizing the weighted number of tardy jobs is NP-hard. On two parallel machines, however, minimizing both the total tardiness and the number of tardy jobs becomes NP-hard. It is shown that the single machine problem with GDD and the objective function of minimizing the weighted tardiness is NP-hard in the ordinary sense (Sriskandarajah (1990)). In Hall et al. (1991) the scheduling problems with GDD and their job-specific due-dates counterparts were compared. They note that one version is neither easier nor harder, from a computational perspective, than its counterpart. For example, minimizing the total tardiness with GDD is achieved by sorting the jobs in non-decreasing order of their processing times (see Hall (1986)), whereas the problem is NP-hard if one considers job-specific due-dates (Du and Leung (1990)). On the other hand, minimizing maximum lateness with precedence constraints is NP-hard when assuming GDD (Sriskandarajah (1990)), but is well known to be polynomially solvable with job-specific due-dates (Lawler (1973)).

The study of Mosheiov and Oron (2004) considers minimizing the total tardiness and the maximum tardiness on parallel machines. The problems are NP-hard and the authors develop lower bounds and show that the SPT first dispatching rule performs well. Gerstl and Mosheiov (2017) study a single machine scheduling problem with two objectives of total tardiness and total rejection costs. The rejection costs occur if the jobs are rejected, i.e., not processed. Minimizing the total tardiness subject to a constraint on the total rejection penalty is shown to be NP-hard, however, the sum of the two objectives is shown to be minimized in polynomial time.

In this paper, we study a single machine scheduling problem with GDD. The goal includes sequencing the jobs and determining the start time of each job such that the maximum earliness-tardiness across all jobs is minimized. A similar problem, however, under the assumption of job-specific due-dates (and not GDD) was studied by Sidney (1977), where it was shown that sequencing the jobs according to the earliest due date (EDD) first dispatching rule is optimal. The start time of jobs can then be determined by solving a linear program. Given the aforementioned studies it is highly likely that the problem of this study is strongly NP-hard.

The remainder of the paper is organized as follows. In Section 2, we provide a detailed problem description, and in Section 3, we develop a polynomial time heuristic to solve the problem. In Section 4, we provide an extensive numerical study to evaluate the efficiency and effectiveness of the heuristic and compare it with two well-know dispatching rules of SPT and LPT. Section 5 concludes the paper.

## 2 PROBLEM DESCRIPTION

Let $J = \{J_1, \ldots, J_n\}$ be the set of $n$ non-preemptive jobs available for processing at time zero on a single machine and $\delta = \{\delta_1, \ldots, \delta_m\}$ be the set of $m$ due dates, where $m < n$. Let $p_j$ be the processing time of job $J_j$, $j = 1, \ldots, n$. By definition, the due-dates are not associated with specific jobs, and therefore, the due-dates must be associated with jobs, implying that more than one job can be associated with a due-date. In this context, we let $\delta_j$ denote the due-date associated with job $J_j$. We denote by $E_j = \max\{\delta_j - c_j, 0\}$ and $T_j = \max\{c_j - \delta_j, 0\}$ the earliness and tardiness of the job in position $j$ of the job-sequence. The problem seeks a sequence of executing the jobs and determining their start times such that the maximum earliness and tardiness among all jobs is minimized, i.e., $\min \max_{J_j \in J}\{E_j, T_j\}$. Using the conventional three-field classification scheme, the problem of this study is then $1|\delta_j(GDD)| \max\{E_j, T_j\}$.

## 3 THE PROPOSED HEURISTIC

In this section we develop a heuristic algorithm for $1|\delta_j(GDD)|\max\{E_j, T_j\}$. Before detailing the operation of the heuristic, we provide some definitions and notations. Given $\delta$ the set of due-dates, we can easily calculate the length or duration of the processing period directly prior to a given due-date $\delta_i \in \delta$. This is $\tau_i = \delta_i - \delta_{i-1}$, where $\tau_1 = \delta_1$.

We note that several cases of the problem can be easily solved. For example, when the due-dates are very large, or when the number of due-dates is equal to the number of jobs. The latter is the main reason behind the assumption of $m < n$ in this study. Under general conditions, however, the problem is very difficult to solve, and we therefore propose a heuristic algorithm. The heuristic has two phases of "assignment" and "scheduling". The assignment phase ensures that jobs are assigned to the due-dates, and the scheduling phase determines the start time of jobs.

The assignment phase first assigns the largest $m$ jobs to the due-dates, as long as a job $J_j$ fits within a due date $\delta_i$, i.e., $p_j \leq \tau_i$. The assignment phase keeps assigning the jobs in a greedy way until all jobs are assigned or some jobs are left that cannot be assigned to any but the last due-date. The scheduling phase determines the start time of the jobs by pushing the jobs froward or backward as far as possible, relative to the associated due-dates, such that the deviations of the completion time of the jobs around the associated due-dates are minimized. It should be noted that the jobs that are assigned to the last due-date and has the processing time greater than the length of the due-dates will most likely incur delays.

## 4 NUMERICAL EXPERIMENTS

In order the evaluate the performance of the proposed heuristic we generate 1,000 instances with 50 and 100 jobs per combination of $m$ and $\nu$, which is the due-dates coefficient and controls the length of the due-dates. We randomly generate processing times from a discrete uniform distribution with parameters $(1, 20)$ and the first due-date with parameters $(0.5\tau, 2\tau)$, where $\tau = \max(p_1, \lceil(\sum_{j\in J} p_j/10)\rceil$. The remaining due dates are constructed from the first one with equal length of $\tau' = \lceil(\nu \times \sum_{j\in J} p_j/m)\rceil$ from each other. We set $m \in \{0.1n, 0.2n, 0.5n\}$ and $\nu \in \{1, 1.5, 2\}$. Each instance is only characterized by its $m$ and $\nu$. This leads to nine different combinations of $m$ and $\nu$, resulting in a total of 18,000 instances. Each instance has identical processing times and the first due-date.

Different values of parameters $m$ and $\nu$ over the relevant ranges ensure that we evaluate the performance of the heuristic over different problem settings, e.g., a small values of $m$ and $\nu$ leads to tight due-dates and a high density of jobs being assigned per due-date. On the contrary, large values of $m$ and $\nu$ result in further-apart due-dates and idle times are therefore expected to incur among due-dates. The running time of all instances required at most a few seconds on a standard desktop computer. This indicates that the heuristic can be used for large problem instances.

We also compared the proposed heuristic and the SPT and LPT dispatching rules, in which the schedules are developed by sorting the jobs in non-decreasing and non-increasing orders of the jobs' processing time, respectively, and then assigning the jobs to the due-dates as long as the start time of the jobs is before the first due-date relevant to the jobs. It is clear that in this way the SPT and LPT schedules do not include idle times, implying that they may perform well for small values of $m$ and $\nu$. The numerical experiments summarized in Table 1 confirm that. In the table, "HEUR" denotes the proposed heuristic algorithm.

In addition, we compared the proposed heuristic and a random algorithm, in which there is no specific order for sorting the jobs' processing time. We denote the random algorithm as "RND" in Table 1.

Table 1 further confirms the effectiveness of the proposed heuristic for solving problem $1|\delta_j\ (GDD)|\ \max\{E_j\ ,\ T_j\ \}$. From 18,000 solved instances, SPT, LPT and RND are able to obtain supe-rior solutions for almost 1.33% of instances (i.e., for 239 instances), and as expected when $m$ and $\nu$ take small values. It is not surprising to see that SPT and LPT performance deteriorates when $m$ and $\nu$ take larger values. Also, it is interesting to observe the significant amount of improvement (averaged over 1,000 instances) delivered by the proposed heuristic algorithm, over SPT, LPT and RND schedules, where the amount of improvement for an instances was calculated as $\frac{z_{SPT/LPT/RND} - z_{HEUR}}{z_{SPT/LPT/RND}} \times 100$. We note that we are not able

to evaluate the true optimality performance of the heuristic due to the unavailability of the optimal schedules. Overall, the heuristic appears to perform extremely well for all problem inputs.

**Table 1**. Summary of the numerical results.

| $n$ | $m$ | $\nu$ | # worse than SPT-LPT-RND | RND better than HEUR/SPT/LPT | Improving SPT (%) | Improving LPT (%) | Improving RND (%) | Time (sec.) |
|---|---|---|---|---|---|---|---|---|
| 50 | 5 | 1 | 144 | 33/418/150 | 8.42 | 3.71 | 8.84 | 0.0010 |
| 50 | 5 | 1.5 | 0 | 0/382/402 | 58.05 | 58.09 | 58.28 | 0.0011 |
| 50 | 5 | 2 | 0 | 0/459/653 | 74.31 | 74.70 | 74.27 | 0.0011 |
| 50 | 10 | 1 | 2 | 1/733/62 | 35.84 | 22.99 | 30.66 | 0.0012 |
| 50 | 10 | 1.5 | 0 | 0/519/309 | 70.35 | 68.95 | 69.90 | 0.0011 |
| 50 | 10 | 2 | 0 | 0/333/481 | 77.97 | 78.45 | 78.34 | 0.0011 |
| 50 | 25 | 1 | 85 | 58/953/13 | 55.23 | 32.15 | 44.52 | 0.0013 |
| 50 | 25 | 1.5 | 0 | 0/910/77 | 86.06 | 81.25 | 83.67 | 0.0015 |
| 50 | 25 | 2 | 0 | 0/831/134 | 90.95 | 89.09 | 89.99 | 0.0013 |
| 100 | 10 | 1 | 0 | 0/731/49 | 28.38 | 21.73 | 25.96 | 0.0022 |
| 100 | 10 | 1.5 | 0 | 0/481/289 | 68.37 | 67.74 | 68.21 | 0.0021 |
| 100 | 10 | 2 | 0 | 0/328/505 | 76.31 | 76.58 | 76.50 | 0.0026 |
| 100 | 20 | 1 | 0 | 0/961/11 | 59.61 | 50.89 | 55.23 | 0.0026 |
| 100 | 20 | 1.5 | 0 | 0/854/108 | 80.34 | 78.11 | 79.22 | 0.0026 |
| 100 | 20 | 2 | 0 | 0/738/202 | 83.79 | 82.63 | 83.20 | 0.0023 |
| 100 | 50 | 1 | 8 | 5/958/0 | 69.32 | 61.54 | 65.82 | 0.0025 |
| 100 | 50 | 1.5 | 0 | 0/954/0 | 92.30 | 90.22 | 91.35 | 0.0028 |
| 100 | 50 | 2 | 0 | 0/941/4 | 95.30 | 94.11 | 94.74 | 0.0037 |

## 5 CONCLUSION

We study a single machine scheduling problem with generalized due-dates. The objective consists of assigning the jobs to the due-dates and determining their respective start time in order to minimize the maximum earliness and tardiness with respect to the due-dates. We introduce a heuristic algorithm to solve the problem. We compare the heuristic and SPT and LPT rules, as well as a random rule. By conducting an extensive numerical tests we show that the heuristic delivers efficient solutions under different problem parameters.

## ACKNOWLEDGMENTS

Amir Salehipour is the recipient of an Australian Research Council Discovery Early Career Researcher Award (project number DE170100234) funded by the Australian Government.

## REFERENCES

Du, J. and J. Y.-T. Leung (1990). Minimizing total tardiness on one machine is np-hard. *Mathematics of Operations Research 15*(3), 483–495.

Gerstl, E. and G. Mosheiov (2017). Single machine scheduling problems with generalised due-dates and job-rejection. *International Journal of Production Research 55*(11), 3164–3172.

Hall, N. G. (1986). Scheduling problems with generalized due dates. *IIE Transactions 18*(2), 220–222.

Hall, N. G., S. P. Sethi, and C. Sriskandarajah (1991). On the complexity of generalized due date scheduling problems. *European Journal of Operational Research 51*(1), 100 – 109.

Lawler, E. L. (1973). Optimal sequencing of a single machine subject to precedence constraints. *Management Science 19*(5), 544–546.

Mosheiov, G. and D. Oron (2004). A note on the spt heuristic for solving scheduling problems with generalized due dates. *Computers & Operations Research 31*(5), 645 – 655.

Sidney, J. B. (1977). Optimal single-machine scheduling with earliness and tardiness penalties. *Operations Research 25*(1), 62–69.

Sriskandarajah, C. (1990). A note on the generalized due dates scheduling problems. *Naval Research Logistics (NRL) 37*(4), 587–597.