ACCURACY OF THE MODIFIED DUE DATE RULE

James C. Bean
Daniel H. Hall

Department of Industrial & Operations Engineering
The University of Michigan
Ann Arbor, Michigan 48109

Technical Report 85-10

ACCURACY OF THE MODIFIED DUE DATE RULE

James C. Bean
Daniel N. Hall

Department of Industrial and Operations Engineering
The University of Michigan
Ann Arbor, Michigan 48109

ABSTRACT

This paper discusses the accuracy of two dispatching heuristics for the single machine total (or average) tardiness sequencing problem: the modified due date rule of Baker and Bertrand and an improvement. The Baker and Bertrand algorithm has one to two per cent error, on average, for 46 test problems. The algorithm logic is designed to fit as a module in a hierarchical system for larger, more complicated, due date driven scheduling systems. A variation is presented with errors less than one percent at the expense of the simple dispatching structure.

KEYWORDS: Scheduling, Total Tardiness

ACCURACY OF THE MODIFIED DUE DATE RULE

James C. Bean
Daniel N. Hall

Department of Industrial and Operations Engineering
The University of Michigan
Ann Arbor, Michigan 48109

1. Introduction

The single machine sequencing problem to minimize total tardiness (or equivalently, average tardiness) involves a set of tasks each having a known processing time, p_j, and due date, d_j. The tardiness of job j is defined as $T_j = \max(0, C_j - d_j)$, where C_j is the completion time assigned to job j. This problem is important in some large scheduling problems (Bean, Birge, Mittenthal and Noon [1984]). Though actual objectives in such problems are much more complicated, total tardiness is sometimes used as a surrogate.

It is not known whether or not the single machine problem to minimize total tardiness is NP-complete. A pseudo-polynomial algorithm has been presented in Lawler [1977]. Other effective approaches to optimally solving this problem include Schrage and Baker [1978], and Fisher [1976]. Heuristic approaches to the total tardiness problem include Wilkerson-Irwin [1971] and Montagne [1969].

The potential impact of the single machine total tardiness problem lies in its use as a module within large hierarchical systems. To be useful in such a hierarchical system, it is desirable for the single machine logic to schedule jobs singly and forward in time. For efficiency it is desirable to schedule in a greedy manner, using only local information. These so called dispatching rules are discussed in Baker and Kanet [1983] and Baker [1984].

Baker and Bertrand [1982] present a dispatching rule for the single machine total tardiness problem called the modified due date (MDD) rule. They show that it performs very well against other dispatching rules. In this note we test the MDD rule against known optimal solutions comparing total tardiness and secondary objectives such as maximum tardiness and mean tardiness among tardy jobs. A modification with better accuracy is presented and tested against the optimal solutions. Justification for the success of these dispatching rules is discussed. Section 2 presents the algorithms. Section 3 presents justification for the algorithms. Section 4 contains computational results.

2. The Algorithms

These algorithms contain much of the intelligence of both Emmons [1969] global dominance criteria and the local interchange from Wilkerson–Irwin [1971]. It is this combination that results in the efficiency and efficacy described in Section 4.

The MDD rule is a dispatching type algorithm which places individual jobs by minimizing the simple statistic $x_j = \max(p_j, d_j - \tau)$ where τ is the current time at dispatch. It is not a true sorting algorithm since the comparison of two jobs requires knowledge of the current time, τ. This algorithm does, however, meet the desired characteristics of scheduling forward in time with local information.

As discussed in Section 4, this algorithm performs surprisingly well on a large set of test problems taken from the literature. However, a slight modification improves its accuracy with little cost in computation. The algorithm with this modification is:

Augmented Algorithm

Step 0 : Set $\tau = 0$. Go to Step 1.

Step 1 : For each unscheduled job calculate $x_j = \max(p_j, d_j - \tau)$. Let $j^* = \arg\min_j x_j$. Go to Step 2.

Step 2 : Put j^* next in the sequence. Update τ to $\tau + p_{j^*}$. Go to Step 3.

Step 3 : For each job currently scheduled, construct a trial schedule with that job removed from its current position and placed last. If the best of such trial schedules has lower total tardiness than the current schedule, consider it the current schedule. If jobs remain unscheduled, then go to Step 1. Otherwise, stop.

The MDD rule is implemented by eliminating Step 3 and returning from Step 2 to Step 1 if jobs are left unscheduled. The augmentation of this algorithm with Step 3 allows an incorrectly placed job to be freed to a later position in the schedule. This algorithm is still very fast, but it loses the forward, local information characteristics of the MDD rule. The Augmented Algorithm could be used as a dispatching algorithm by rescheduling all unscheduled jobs at each dispatching time. Only the first job in the resulting sequence would be dispatched. The remaining would be rescheduled at the next dispatch time. Though this algorithm takes more time than a simple implementation of the MDD rule, it is till fast enough to be used in a control mode. This fact is evidenced by the timing in Table 2.

3. Justification of the Algorithms

As seen in Section 4, these algorithms deliver nearly optimal solutions on the problems tested. There is theoretical justification for this high degree of accuracy.

The MDD rule contains the local intelligence used by the Wilkerson–Irwin Algorithm. We know that, for two adjacent jobs j and k, we will locally lower tardiness if they are in earliest due date order unless $\tau + \max(p_j, p_k) > \max(d_j, d_k)$, in which case they should be in shortest processing time order. This pairwise interchange argument is not guaranteed to lead to an optimal sequence because the comparison of j and k is local.

Lemma 1: At any time, $\tau \geq 0$, $\max(p_j, d_j - \tau) \leq \max(p_k, d_k - \tau)$ if and only if j may precede k according to this interchange rule.

Proof: Omitted. (consists of several obvious cases)

The MDD rule goes further in that it performs global comparisons. Since the value x_j contains data pertaining only to job j, all unscheduled jobs can be considered simultaneously at each placement. As a corollary of Emmons [1969] classic set of dominance properties for the total tardiness problem we can see that these global comparisons are commonly optimal.

Theorem 2: (Emmons) Let β_j be the set of jobs known to precede job j in some optimal sequence. Let α_j be the set of jobs known to succeed job j in some optimal sequence. Finally, let α_j' be the set of jobs not represented in α_j. If any of the properties a), b) or c) below is satisfied for two jobs j and k, then j precedes k in some optimal sequence.

a) $p_j \leq p_k, \quad d_j \leq \max(d_k, p_k + \sum_{i \in \beta_k} p_i)$

b) $p_j > p_k, \quad d_j \leq d_k, \quad d_k + p_k \geq \sum_{i \in \alpha_j'} p_i$

c) $d_k \geq \sum_{i \in \alpha_j'} p_i$.

At time zero, $\tau = 0$, the comparison rule in Step 1 of the algorithm simplifies to

$$j^* = \arg\min_j \max(p_j, d_j).$$

In choosing j^* we hope that for all $k \neq j^*$ that j^* precedes k in some optimal sequence. By construction we know that $\max(p_{j^*}, d_{j^*}) \leq \max(p_k, d_k)$. If this fact were sufficient to prove that j^* precedes k then, by induction, we could prove that the MDD rule is in fact optimal (see Emmons [1969]). This is, of course, not the case. However, it is instructive to proceed in this manner. Pointing out the valid parts of such a proof and those that break down suggests a reason for the algorithms' accuracy and potential directions for bounding its error.

The fact that $\max(p_{j^*}, d_{j^*}) \leq \max(p_k, d_k)$ leads to five possible cases. In three of them we can prove that the decision is in fact optimal. The cases are

1) $d_{j^*} \leq p_{j^*} \leq p_k$

2) $d_{j^*} \leq p_{j^*} \leq d_k, \quad p_k < p_{j^*}$

3) $p_{j^*} < d_{j^*} \leq p_k$

4) $p_{j^*} < d_{j^*} \leq d_k, \quad p_{j^*} \leq p_k$

5) $p_{j^*} < d_{j^*} \leq d_k, \quad p_k < p_{j^*}$

Corollary 3: In cases 1, 3, and 4 above it is optimal to place j^* before k.

Proof: A corollary of Theorem 2 part a).∎

Equally interesting is the analysis of cases 2 and 5. In these cases we do not know that in fact the choice to place j^* before k is in error. We simply do not have enough information. From Theorem 2 parts b) or c), if $\sum_{i \in \alpha'_j} p_i$ is small enough, this choice will be optimal as well. Knowledge of this value requires knowledge of the future schedule and global information. The computational results show how frequently decisions were made in the known optimal cases and the unknown cases, 2 and 5.

4. Computational Results

Computational experiments were carried out on the University of Michigan Amdahl 5860 computer under the MTS operating system. All CPU times are given in seconds and exclude data input.

Test problems were taken from Baker [1974] and Fisher [1976]. In all, 46 problems were run ranging from 8 to 50 jobs. Table 1 shows the averages of runs in four problem sets using the MDD rule. The same information for the Augmented Algorithm is in Table 2. The first set of problems is comprised of the 16 problems in Baker [1974]. Sets two, three, and four are subsets of the 20, 30, and 50 job problems in Fisher [1976]. The column labeled "% 1,3,4" in Table 1 is the fraction of decisions made under the optimal cases in Corollary 3. The column labeled "MEAN ERROR" displays the arithmetic average of

$$\frac{\text{heuristic value} - \text{optimal value}}{\text{optimal value}}$$

for the problems in that set. In Table 2 the column "# OPT" presents how many of the problems in that set were solved optimally. These results are the same as for the MDD rule and, hence, are not included in Table 1.

6. Summary and Conclusions

This note discusses fast algorithms for the basic single machine sequencing problem with objective to minimize total tardiness. The MDD rule of Baker and Bertrand has the characteristics that jobs are scheduled forward in time using only local information. Hence, the logic of the algorithm may be useful as part of a hierarchical system to solve due date driven scheduling problems. The algorithms have negligible computation times for problems with 50 jobs. The error is on the order of one to two per cent.

Of the two algorithms discussed here, the Augmented Algorithm has better accuracy for the total tardiness problem. This does not indicate, however, that it should be the algorithm of choice in an application situation. As mentioned earlier, the objective to minimize total tardiness is usually a surrogate for the actual, more complex, objective. Hence, we should look at the general quality of solutions generated by the algorithms. In particular we will look at the distribution of tardiness in the solutions.

Table 3 compares the maximum tardiness and mean tardiness over tardy jobs for the solutions from the MDD rule, the Augmented Algorithm, and the optimal solution. For each problem the secondary values for the MDD rule were considered the reference points. The results from the Augmented Algorithm and the optimal solution were normalized by these values and the averages reported.

Keeping in mind that the MDD rule displays average error in total tardiness of under two per cent, it appears to have a distinct advantage in the other measures over both the Augmented Algorithm and any optimal algorithm. In many cases, going too far towards the total tardiness optimal produced schedules with particular jobs that were very tardy. Overall, our experience with practitioners indicates that the MDD rule delivers more usable solutions than either the Augmented Algorithm or any optimal algorithm.

The structure of this algorithm is simple enough that obvious changes could be made to handle ready times and sequence dependent set-ups. Some extensions are discussed in Baker and Bertrand [1982]. Job shop adaptations are in Baker and Kanet [1983] and Baker [1984]. These and other extensions are currently being investigated.

REFERENCES

Baker, K. [1974], **Introduction to Sequencing and Scheduling**, Wiley: New York.

Baker, K. [1984], "Sequencing Rules and Due-Date Assignments in a Job Shop," **Management Science**, Vol. 30, pp. 1093–1104.

Baker, K. and J. Bertrand [1982], "A Dynamic Priority Rule for Scheduling Against Due-Dates," **Journal of Operations Management**, Vol. 3, pp. 37–42.

Baker, K. and J. Kanet [1983], "Job Shop Scheduling with Modified Due Dates," **Journal of Operations Management**, Vol. 4, pp. 11–22.

Bean J., J. Birge, J. Mittenthal and C. Noon [1984], "An Adaptive Approach to Real-Time Scheduling," Technical Report 84-18, Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, Michigan 48109.

Emmons, H. [1969], "One-Machine Sequencing to Minimize Certain Functions of Job Tardiness," **Operations Research**, Vol. 17, pp. 701–715.

Fisher, M. [1976], "A Dual Algorithm for the One-Machine Scheduling Problem," **Mathematical Programming**, Vol. 11, pp. 229–251.

Lawler, E. [1977], "A Pseudopolynomial Algorithm for Sequencing Jobs to Minimize Total Tardiness," **Annals of Discrete Mathematics 1**, pp. 331–342.

Montagne, E. [1969], "Sequencing with Time Delay Costs," Arizona State University Industrial Engineering Research Bulletin No. 5, pp. 20–31.

Schrage, L. and K. Baker [1978], "Dynamic Programming Solution of Sequencing Problems with Precedence Constraints," **Operations Research**, Vol. 26, pp. 444–449.

Wilkerson, L. and J. Irwin [1971], "An Improved Method for Scheduling Independent Tasks," **AIIE Transactions**, Vol. 3, pp. 239–245.

This research was supported by a grant from General Motors Corporation.

TABLE 1: Computational Results for the MDD Rule

SOURCE	# JOBS	# PROBS	MEAN CPU	MAX CPU	MEAN ERROR	MAX ERROR	% 1,3,4
Baker	8	16	.002	.002	.004	.053	68
Fisher	20	13	.002	.002	.022	.118	73
Fisher	30	10	.004	.004	.012	.060	69
Fisher	50	7	.008	.009	.025	.055	78

TABLE 2: Computational Results for the Augmented Algorithm

SOURCE	# JOBS	# PROBS	# OPT	MEAN CPU	MAX CPU	MEAN ERROR	MAX ERROR
Baker	8	16	14	.002	.002	.004	.049
Fisher	20	13	7	.007	.008	.010	.066
Fisher	30	10	7	.016	.020	.006	.024
Fisher	50	7	1	.066	.072	.009	.017

TABLE 3: Secondary Objective Considerations

# JOBS	MDD T_{max}	MDD $\bar{T}_{T_j>0}$	AUG. T_{max}	AUG. $\bar{T}_{T_j>0}$	OPT. T_{max}	OPT. $\bar{T}_{T_j>0}$
8	1.000	1.000	1.012	1.000	1.041	1.058
20	1.000	1.000	1.162	1.142	1.195	1.140
30	1.000	1.000	1.000	1.075	1.021	1.065
50	1.000	1.000	1.057	1.178	1.040	1.204