Scheduling of a Real World Filter Production with Lot-Size 1

Frank Herrmann
Ostbayerische Technische Hochschule Regensburg - University of Applied Sciences Regensburg
Innovation and Competence Centre for Production Logistics and Factory Planning (IPF)
PO box 120327, 93025 Regensburg, Germany
E-Mail: Frank.Herrmann@OTH-Regensburg.de

***Abstract*—In industrial practice, a travelling crane on the ceiling of a factory hall transports products in process from one station to the next one in a production line. Due to space restrictions, there is no buffer between the stations. The production line at Fiedler Andritz, Regensburg in Germany, can be seen as an example of such a problem class. Such restrictions reduce the set of feasible schedules even more than the no-buffer restrictions discussed in the literature in the case of limited storage. Since this scheduling problem is integrated in the usual hierarchical planning, the tardiness is minimised. Due to the high number of jobs as well as the goal of a simple algorithm, scheduling is always done by priority rules at the company site. The standard approach of using the net processing time causes poor results. A simulation of the processing time is suggested. In addition, several very relevant priority rules from the literature are modified by this simulated processing and significantly better results are obtained.**

Keywords-Simulation of processing time; scheduling, flow-shop; no-buffer (blocking); no-wait; priority rules; real world application; filter production

I. INTRODUCTION

Specific products are produced by special machines, which are often grouped in a flow shop. They have to produce small batches with short response times, so scheduling algorithms are needed to ensure that under the constraint of a high average load of the flow shop, the due dates of the production orders are met. Nowadays, such special designed flow shops often have technological restrictions, which complicate the scheduling. For example, in cell manufacturing, buffer could be non-existent due to limited space and storage facilities. So, in recent years, a considerable amount of interest has arisen in no-buffer (blocking) scheduling problems and in no-wait scheduling problems, with makespan as objective criterion. Often, these production systems deliver products for other systems as well. Due to the hierarchical planning, which is implemented in enterprise resource planning systems (ERP system) (see e.g., [7]), the local completion times in one production system in many cases determine the earliest possible starting times in another production system. Thus, the delay of the operations in a production system has an impact on the effectivity of this coordination process. Therefore, tardiness is considered as objective criterion.

The paper is structured as follows. The real world application is described, followed by a literature review. Next, the algorithms are explained, then, the computational results are given and analysed and the papers ends with a conclusion.

II. A REAL WORLD APPLICATION

The problem is a modification of a partly automated production line at Fiedler Andritz in Regensburg, Germany, to produce filter (baskets) with a lot size of 1. All filters have unified constructions. They differ in varying heights of the baskets and there exist different designs.

The production line consists of 4 stations which are shown in Figure 1. Station 1 assembles 6 single batons (called consoles) on an assembly ground plate to a skeleton of a filter basket. Baton profiles are assembled into the provided slots of the filter basket skeletons. At the plunge station a wire coil is contrived in the device of a lining machine. The lining machine straightens the wire and inserts batons into the slots. To ensure stability, the span station installs span kernels in the case of outflow filter baskets and span belts in the case of inflow filter baskets. Then, the filter basket is lifted from the assembly ground plate and is transported to the welding station, at which the baton profiles are welded on the filter basket skeletons. The accomplished filter basket leaves the production line. Prior to this, the span medium is removed. An overhead travelling crane lifts a filter basket out of a station, transports it to the next station and inserts it directly in this station. This is just possible if this station is free. So, there is no buffer in the production line and each feasible schedule of jobs is a permutation of these jobs. Due to other operational issues, the crane can just be moved if all stations are inactive. Since an operation cannot be interrupted, the transport has to be performed after the completion of all operations on the stations in the flow shop. Due to further operational issues, this restriction has to be applied also to the first and the last station; note, that the crane loads S1 and unloads S4 as well. In summary, all stations are loaded and unloaded with filters during a common process and this process starts with the last station S4, followed by station S3, S2, until station S1 is reached. It is allowed that a station is empty; then this station is skipped (may be partially) in this process. In total, there are 10 part types whose processing times are listed in Table I.

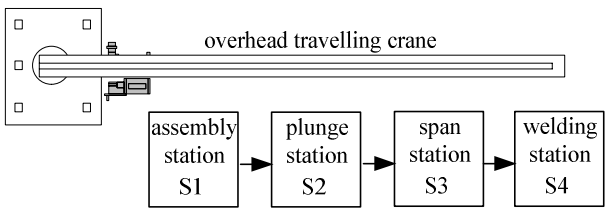

Figure 1. Structure of the production line.

 ISBN: 978-1-61208-831-0

At the company's production site, the jobs for filters are generated by an SAP system and produced filters are stored before they are assembled into other products or sold directly to customers. Therefore, all jobs with a release date after the beginning of a period are released at the beginning of this period. One period consists of one day with three 8 hour shifts. For this investigation, sequences of jobs of filter types with lot size 1 are randomly generated for each period t by an generating algorithm which has been designed in accordance with the proceeding in [22] and in [2]: An additional filter type F, released in period t, consumes capacity on each station during the time between t and its due date; the calculation for the capacity just uses the net processing time and does not regard the dependencies between the jobs (released so far). F is accepted as long as this consumed capacity on each station is below a maximal load level, otherwise it will be skipped to the next period. A maximal load level is an (intended) average load $\left(L_0(S)\right)$ plus 0, – 30% and +30% of $L_0(S)$. Over the first 5, 10, 15 etc. consecutive periods, the load level variations average to zero.

TABLE I. PROCESSING TIMES FOR ALL PART TYPES IN MINUTES.

Part type	Station				Sum of times
	S1	S2	S3	S4	
P1	100.5	50	53.5	9	213
P2	256.5	50	53.5	9	369
P3	122	135	90	75	422
P4	256.5	50	267	9	582.5
P5	182	200	135.5	140	657.5
P6	100.5	300	53.5	300	754
P7	223	250	196	220	889
P8	223	250	206.5	220	899.5
P9	100.5	300	267	300	967.5
P10	256.5	300	267	300	1123.5

In reality at the company, there are large numbers of periods with a low number of late jobs and large numbers of periods with a high (or even very high) number of late jobs. To achieve a comparable situation for this investigations, due dates are determined in a way so that scheduling with the FIFO rule (first-in-first-out) causes a specific percentage of late jobs. The company confirmed that job sequences with 30%, 50%, 70% and 85% of late jobs by scheduling with the FIFO rule (called time pressure) are comparable to the ones which occurred in the real operation. As a result of the generating algorithm's calculations the mean difference between the due dates and the release dates are between 2 and 3.5 days with a standard deviation of 0,5 days. Andritz Fiedler has confirmed that such timeframes for processing jobs are representative.

The time needed for loading and unloading a station is negligible compared to the duration of the operation itself. In addition, this task is independent from the allocation (or loading) of other stations and the required time is included in the duration of the operation.

The general scheduling problem consists of M stations and a pool of N jobs, which may change at any time, with known earliest possible starting times for release dates a_i $(1 \le i \le N)$ and due dates f_i $(1 \le i \le N)$ respectively. Also, there is the duration $t_{i,j}$ of operation $(o_{i,j})$ j $(1 \le j \le M)$ of job i $(1 \le i \le N)$, which is being worked on station j. As performance criteria average tardiness (T_{Mean}) and standard deviation of the tardiness (T_σ) are primarily analysed.

The time between two consecutive executions of the load process is determined by the maximum of the duration of the operations on the stations in the flow shop. This is called cycle time. This “load”-restriction, the no-buffer condition and the capacity of the stations are the main restrictions. Setup times are relatively small compared to operation times and are included in those.

The no-buffer condition means a relaxation of the scheduling problem with the (above) “load”-restriction. Scheduling problems with the no-buffer are proven to be NP-hard in the strong sense for more than two stations; see e.g., [5], which contains a good survey of such problems.

III. LITERATURE REVIEW

As mentioned earlier, the real application is close to the class of no-buffer (blocking) scheduling problems. Solutions for the no-buffer (blocking) scheduling problems are published in various papers. In [10], a schedule is extended by a (unscheduled) job that leads to the minimum sum of blocking times on machines which is called Profile Fitting (PF). Often, the starting point of an algorithm is the algorithm of Nawaz, Enscore and Ham, in short NEH algorithm, presented in [11], as it is the best constructive heuristic to minimize the makespan in the flow shop with blocking according to many papers, e.g., [3]. Therefore, [19] substituted the initial solution for the enumeration procedure of the NEH algorithm by a heuristic based on a makespan property proven in [18] as well as by the PF of [10]. [20] used an elaborated lower bound to realise a branch-and-bound algorithm which becomes a heuristic since the CPU time of a run is limited. Also, for minimising makespan, [4] realised and analysed a tabu search algorithm. As an alternative approach, [27] have developed a discrete particle swarm optimisation algorithm. In order to diversify the population, a random velocity perturbation for each particle is integrated according to a probability controlled by the diversity of the current population. Again, based on the NEH algorithm, [28] described a harmony search algorithm. First, the jobs (i.e., a harmony vector) are ordered by their non-increasing position value in the harmony vector, called largest position value, to obtain a job permutation. A new NEH heuristic is developed on the reasonable premise that the jobs with less total processing times should be given higher priorities for the blocking flow shop scheduling with makespan criterion. This leads to an initial solution with higher quality. With special settings as a result of the mechanism of a harmony search algorithm, better results are achieved. Also [17] presented an improved NEH-based heuristic and uses this as the initial solution procedure for their iterated greedy algorithm. A modified simulated annealing algorithm with a local search procedure is

 ISBN: 978-1-61208-831-0

proposed by [28]. For this, an approximate solution is generated using a priority rule specific to the nature of the blocking and a variant of the NEH-insert procedure. Buffering strategies are proposed by [30] to handle random machine breakdowns for minimizing makespan in a job shop. Again, based on the PF approach of [10][12] addressed two simple constructive heuristics. Then, both heuristics and the PF are combined with the NEH heuristic to three improved constructive heuristics. Their solutions are further improved by an insertion-based local search method. The resulting three composite heuristics are tested on the well-known flow shop benchmark of [24], which is widely used as benchmark in the literature. [9] achieve for flow shop scheduling with jobs arriving at different times with a simple and constructive heuristic method very good results for the average completion time.

To the best of my knowledge, only a few studies investigate algorithms for the total tardiness objective (for flow shops with blocking). [18] have developed a lower bound which reduces the number of nodes in a branch-and-bound algorithm significantly. [21] described several versions of a local search. First, with the NEH algorithm, they explore specific characteristics of the problem. A more comprehensive local search is developed by a GRASP based (greedy randomized adaptive search procedure) search heuristic.

Priority rules are still a first choice in the case of complex scheduling problems. Thus, in [23], for a complex scheduling problem the performance of priority rules are analysed. Another example is the application of priority rules for the dispatching of AGVs in flexible job shops in [6]. It might be that in the near future several such problems will be solved by more sophisticated heuristics as genetic algorithms for example.

IV. HEURISTIC SOLUTION BY PRIORITY RULES

The real application operates in dynamic environments where real time events like station failure, tool breakage, arrival of new jobs with high priority, changes of due dates etc. may turn a feasible schedule into an infeasible one. A feasible assignment of a job is achieved by a priority rule like earliest due date (EDD), because a priority rule orders a queue in front of a station quasi immediately. So, priority rules are still analysed in many studies on scheduling; one example of a recent one is [1].

Due to the "load"-restriction, the processing time of a job A on the flow shop depends on the other jobs processed on the flow shop at the same time. Therefore, its processing time can be significantly larger than the sum of the processing times of its single operations (t_A), called net processing time of job A. A realistic processing time for a job A is achieved, if the processing on the flow shop is simulated with respect to the jobs processed on this flow shop at the same time. After four cycles $(\tau_1,...,\tau_4)$, job A leaves the flow shop. So, these cycle times depend on the three jobs (X1, X2, and X3) on the flow shop as A is assigned to station 1, and the three jobs (B1, B2, and B3) following A in the sequence; this is illustrated in Figure 2. So, $\tau_1+\tau_2+\tau_3+\tau_4$ is the (total) processing time of A (tt_A). This processing time of job A is only correct, if the tail is (structurally) identical with the three jobs which follow job A in the final permutation. Since these are not known a deviation normally occurs between this calculated processing time and the processing time which will really occur if job A is chosen.

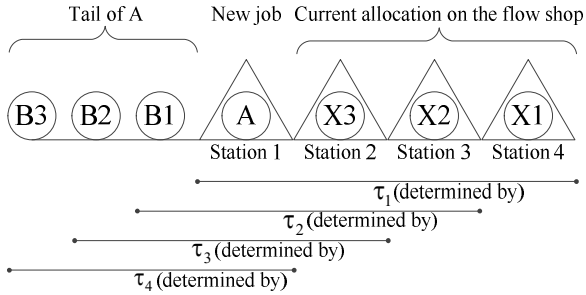

Figure 2. Calculation the processing time of a job A.

As known, tardiness is improved by assigning jobs with a small slack; the slack for job i is defined by $f_i - t - tt_i$, where t is the current time and f_i is the due date of job i. Investigations by the author (see also [2]) show, that for many job shop problems the rules

$$\text{CR+SPT} = \begin{cases} \frac{f_i - t}{tt_i}, & f_i - t - tt_i > 0 \\ tt_i, & f_i - t - tt_i \le 0 \end{cases}$$

(tt_i is the shortest processing time – the SPT rule; $\frac{f_i - t}{tt_i}$ is the critical ratio (CR)), ODD (EDD-rule – here) and $\text{SL}/\text{OPN} = \frac{f_i - t - tt_i}{M}$ (slack rule (SL) – here) are Pareto optimal to the average, the standard deviation and the maximum tardiness.

In addition, two more rules are applied, after adaption to this production line. One is the rule RR, proposed by [15], which seeks to minimize both mean flow time and mean tardiness of jobs. The priority index is $(f_i - t - tt_i)\cdot e^{-\eta} + e^{\eta}\cdot tt_i$, with utilisation level η of the entire flow shop defined by $\eta = \frac{b}{b+j}$ with b being the busy time and j being the idle time of the entire flow shop; the job with the lowest priority index will be processed next.

The other one is based on an optimal solution for the single-station weighted tardiness scheduling problem and was proposed by [14] as the weighted slack-based scheduling rule RM. The priority index is: $\frac{1}{\pi_i}\cdot e^{-\frac{k}{\bar{t}}\cdot\max\{f_i - t - tt_i, 0\}}$. As local processing time costing $\pi_i^l = tt_i$ is used, called RM local, and as global processing time costing $\pi_i^g = \sum_{i\in U_t} tt_i$, where U_t is the set of unfinished jobs in the pool of orders, excluding job i, is used, called RM global; [8] considered additional resource costs which do not fit to our problem or

have already been integrated (in other parameters); e.g., a (static) bottleneck resource cost is not applicable because it is assumed that each station is temporary a bottleneck.

V. COMPUTATIONAL RESULTS

The real world application is realised in the simulation tool "Plant Simulation" together with an implementation of the above mentioned hierarchical planning as realised in ERP systems, used in industrial practice. Average tardiness (T_{Mean}) and standard deviation of the tardiness (T_σ) reach a steady state by a simulation horizon of 5000 periods.

A large number of preliminary studies show that the parameter k has a significant impact on the performance of the local and the global RM rule and the best results are achieved with $k = 1$. In [26], k has a negligible influence, but they regard an RCPSP.

The impact of the simulated processing time is analysed first. Due to its implementation, its concrete values depend from the (3) jobs on the flow shop and the (3) jobs following. So, the values should be independent from the priority rules and the time pressure. But, it could be that individual rules prefer permutations of jobs in a cycle. This is observed in experiments with the (above explained) priority rules. In these experiments the mean, the standard deviation, the minimum and the maximum of the simulated processing times are measured. The measured values for all part types and all rules are between those for the SPT rule and the SL rule which are listed in Table II. These results demonstrate a huge deviation from the net processing time

TABLE II. NET PROCESSING TIME (NET) AND SIMULATED PROCESSING TIME IN MINUTES FOR THE RULES SPT AND SL.

Part type	SPT				
	Mean	Net	Standard deviation	Mini-mum	Maxi-mum
P1	1116.5	213	151.8	885	1349.5
P2	1159.8	369	100.45	1061	1349.5
P3	1088.6	422	138.1	929	1349.5
P4	1151.7	582.5	96.4	1061	1349.5
P5	1162.9	657.5	88.5	1063.5	1349.5
P6	1233.3	754	95.3	1098.5	1376
P7	1228.1	889	60.1	1164.5	1349.5
P8	1225.5	889.5	57.8	1164.5	1349.5
P9	1237.2	967.5	93.9	1098.5	1376
P10	1322.8	1123.5	27.9	1098.5	1376
	SL				
	Mean	Net	Standard deviation	Mini-mum	Maxi-mum
P1	1179	213	126.5	885	1349.5
P2	1192.6	369	112.3	885	1349.5
P3	1179.3	422	118.1	885	1349.5
P4	1181.7	582.5	111.5	885	1349.5
P5	1184.6	657.5	106.5	885	1349.5
P6	1198.3	754	106.1	885	1376
P7	1203.8	889	101.7	885	1376
P8	1207.9	889.5	98.2	885	1376
P9	1215.9	967.5	98.1	885	1376
P10	1225.9	1123.5	97.5	885	1376

Furthermore, the experiments show a significant impact by the tail. The values in Table II are representative for many tails. Exceptions occur if each tail only consists of jobs with a small net processing time, as with part type P1, or a high one, as with part type P10. In the first case, the simulated processing times have small mean values, e.g., for part type P1 901,1 minutes and for part type P10 1067,6 minutes, and in the second case the mean values are large, e.g., 1248,5 minutes for part type P1 and 1269,2 minutes for part type P10. The standard deviations are in the first case huge, e.g., for part type P1 173,1 minutes and for part type P10 187,6 minutes, and the second case low e.g., for part type P1 77,4 minutes and part type P10 61,3 minutes.

As indicated by these results, the performance of a priority rule is influenced by the concrete tail. Instead of regarding all 1000 possible tails, the study is limited on tails whose part types have similar net processing times and those whose part types have significantly different net processing times. The experiments for each priority rule show both significant and minor deviations. These significant deviations are not exceptions and the reduction is almost one-third (or even more). Results which are close to the best possible results are achieved, if each tail consists of jobs of part type P4 only (i.e., P4, P4, P4); this tail is used in the following. Experiments show that an accidental tail (i.e., the part types for a tail are randomly selected) is a very good alternative.

Table III contains the percental changes by using simulated processing times instead of net processing times. Due to the characteristic behaviour of the SPT rule and the SL rule to T_{Mean} and T_σ (see above), both are analysed first. The SPT rule benefits most from a more realistic processing time. In case of the SL rule, there are just small improvements but often significant deteriorations. As long as the CR+SPT rule uses CR – which is some kind of slack – there are deteriorations for T_{Mean}. Since the net processing time is much smaller than the simulated processing time, the CR+SPT rule with simulated processing time decides earlier according to SPT as the CR+SPT rule with net processing time which explains the improvement. Both rules RR and RM are a combination of slack and shortest processing time (SPT). The RR rule benefits from a more precise processing time. A smaller (percental) improvement compared to the SPT rule is caused by already better values if net processing is used and the impact of the slack; The impact of the slack is shown most clearly by a time pressure of 85%. Much better (percental) improvements for T_{Mean} in case of a small time pressure is caused because RR rule prefers critical jobs with positive slack much better than the SPT rule (which does not take due dates into account). The RM rule prefers small jobs if there is no slack and otherwise jobs with small slack. Depending on the degree of influence of the slack on the priority, it can be expected that their changes are between those of the rules SL and CR+SPT. The values in Table III confirm this conclusion primarily for the rule RM global. The concrete values depend on the time pressure. In the case of rule RM local, the percental improvements for T_{Mean} and for T_σ with small time pressure are comparable to the ones of the SPT rule. In the case of T_σ a better processing time

 ISBN: 978-1-61208-831-0

causes an increase of the (absolute) values for this rule on the level of the values of the SPT rule, except for a low time pressure.

TABLE III. CHANGE BY USING SIMULATED PROCESSING TIME INSTEAD OF NET PROCESSING TIME IN PERCENT, COMPARED TO THE RESULT WITH NET PROCESSING TIME.

Rule	Time pressure			
T_{Mean}	30%	50%	70%	85%
SPT	57.6	10.3	12.3	25.8
SL	-41.2	-2.3	0.23	-17
CR+SPT	-4	-10.3	-7.4	8.9
RR	75.3	4.2	3.9	11.9
RM local	48.9	10.4	11.7	21.5
RM global	3.3	-10.9	-4	-5.8
T_{σ}	30%	50%	70%	85%
SPT	68.65	31.3	27.2	48.11
SL	-20.1	1.2	3.4	-19
CR+SPT	51.4	34.1	37.1	48.6
RR	69.8	14.3	14.4	23.6
RM local	68.3	-21.2	-25.5	-242.8
RM global	50	-10.1	10.4	67.9

The absolute values, listed in Table IV (just the ones by using simulated processing times), differ partially from the results published in other papers. In order to judge this and the following more detailed results, it should be pointed out that the real world problem in this paper has a very special problem structure, compared to the representative problems usually regarded in the literature. The above mentioned expectation of the SPT and the SL rule – namely small T_{Mean} at the expense of large T_{σ} by the first rule and the opposite by the second one – is fulfilled. For many job shop and flow shop problems the CR+SPT rule outperforms the best results from the SPT and the SL rule. For this real world application CR+SPT delivers always, with one exception, results, which are much worse than the ones of all other rules. Since the rules RR and RM also combine slack and SPT, the worse results of CR+SPT are caused by a too late switch from preferring jobs with small slack to jobs with small SPT. Compared to other job shop problems this is more significant because just a misguided decision causes long cycles, which reduces the remaining slack for all other jobs much more than in the case of typical job shop problems. In addition, these long cycles could be very ineffective in terms of large times on some stations without any processing.

The performance of the RR rule for the real world application is primarily compared with the results in [16], because in [15] an open shop problem is regarded. In [16], the RR rule delivers better results than the other rules except for one case. In detail, in most of the cases, the improvements are less significant (partially much less) and the sequences of the regarded rules according to both performance criteria, are different. The results of the priority rules in [16] and in [15] are significantly impacted by the station utilisation levels, which are primarily 80% and 95%. In addition [16], uses an allowance factor and [15] uses the due date tightness, which both have a much smaller impact than the station utilisation level. In this investigation the time pressure is only increased by using tighter due dates while the load on the stations remains unchanged. Therefore, it can be expected that the time pressure here has a similar effect as the allowance factor in [16] or the due date tightness in [15]. A much larger impact is caused by a significant fluctuation of the load in the periods, so there are some periods where the load is either much higher than 95% and or much lower than 80% (as in [16] or in [15]). Thus a tighter due date has a more significant effect in periods with relative very high load than in periods with a relative very small load. In total, this seems to cause the different amount of improvement.

TABLE IV. ABSOLUTE PERFORMANCE MEASURES FOR PRIORITY RULES WITH SIMULATED PROCESSING TIME IN MINUTES.

Rule	Time pressure			
T_{Mean}	30%	50%	70%	85%
SPT	99.1	323.3	326.2	646.9
SL	161.6	321.5	344.7	1149.8
EDD	103.4	300.2	342.2	948.7
CR+SPT	581.97	575.7	574.6	1032.7
RR	40.7	279.5	313.5	823.1
RM local	55.3	267.1	278.6	626.2
RM global	134.9	346.7	359.2	1008.5
T_{σ}	30%	50%	70%	85%
SPT	314.5	449.2	473.4	826.7
SL	353.4	315.3	326.6	787.6
EDD	243.1	293.1	328.9	643.4
CR+SPT	2023.9	916.4	901.2	1464.1
RR	125.1	282.8	305.7	564.6
RM local	235.9	425.8	454.1	1187.4
RM global	395.5	456.2	464.6	890.04

The poor results of RM global compared with RM local (with one exception) contradicts the results in [8]. The same happens in the investigation of [26]. The local processing time costing prefers more often jobs with short processing times than the global processing time costing. This procedure explains the much better results in the case of those time pressures for which the SPT rule delivers a much better T_{Mean} than to the SL rule. In the other cases,i this procedure is beneficial if many tardy jobs are waiting in front of the flow job. Finally, in the flow shop in [26] with parallel resources and setup states the differences in the results of the rules are smaller than in this investigation and vice versa for the more general problem structure in [26].

Overall, the simulated processing times should be used in the priority rules. Then, the rules RR and RM local deliver the best mean tardiness. RR is beneficial with low and RM local with (very) high time pressure. The RR rule delivers the best standard deviation of the tardiness (for all time pressures).

VI. CONCLUSIONS

This paper presents a real world flow shop scheduling problem with more restrictive restrictions than the ones normally regarded in literature. To ensure online scheduling this investigation is restricted to those priority rules, which

 ISBN: 978-1-61208-831-0

are considered in literature as being very effective. The substitution of the net processing time, normally used in priority rules, by a simulated one delivers often significant better results. Some tests with optimisation solutions for a small test problem indicate that priority rules do not recognise when to prefer large variances of cycle times or small ones, respectively, and that the schedules of priority rules have outliers in the cycle times, which are usually avoided by optimal solutions. With this and other characteristics of optimal solutions an efficient metaheuristic like local search or genetic algorithm shall be developed next.

Up to now, the workers for the manual tasks are not scheduled. In addition, limited resources, like the available number of coils or assembly ground plates, occur at some company sites. Such requirements are also left to future investigations.

REFERENCES

[1] A. El-Bouri, “A cooperative dispatching approach for minimizing mean tardiness in a dynamic flowshop”, in Computers & Operations Research, Volume 39, Issue 7 (July), pp. 1305–1314, 2012.

[2] S. Engell, F. Herrmann, and M. Moser, “Priority rules and predictive control algorithms for on-line scheduling of FMS”, in Computer Control of Flexible Manufacturing Systems, S.B. Joshi and J.S. Smith (Eds.). Chapman & Hall, London, pp. 75–107, 1994.

[3] J. M. Framinan, R. Leisten, and C. Rajendran, “Different initial sequences for the heuristic of Nawaz, Enscore and Ham to minimize makespan, idletime or flowtime in the static permutation flowshop sequencing problem”, in International Journal of Production Research, 41, pp. 121–148, 2003.

[4] J. Grabowski and J. Pempera, “The permutation flow shop problem with blocking. A tabu search approach”, in Omega, 35 (3), 302–311, 2007.

[5] N. G. Hall and C. Sriskandarajah, “A survey of machine scheduling problems with blocking and no-wait in process”, in Operations Research 44 (3), pp. 510–525, 1996.

[6] J. Heger and T. Voß, “Dynamic priority based dispatching of AGVs in flexible job shops”, in Procedia CIRP, Volume 79, pp. 445–449, 2019.

[7] F. R. Jacobs, W. Berry, D. Whybark, and T. Vollmann, “Manufacturing Planning and Control for Supply Chain Management”, in McGraw-Hill/Irwin (New York), 6 edition, 2010.

[8] S. Lawrence and T. Morton, “Resource-constrained multi-project scheduling with tardy costs: Comparing myopic, bottleneck, and resource pricing heuristics.”, in European Journal of Operational Research 64, pp. 168–187, 1993.

[9] G. Li, N. Li, N. Sambandam, S. P. Sethi, and F. Zhang, “Flow shop scheduling with jobs arriving at different times” in International Journal of Production Economics, Volume 206, 2018, pp. 250–260, 2018.

[10] S. T. McCormick, M. L. Pinedo, S. Shenker, and B. Wolf. “Sequencing in an assembly line with blocking to minimize cycle time”. Operations Research, 37 (6), pp. 925–935, 1989.

[11] M. Nawaz, E. E. Enscore, and I. Ham, “A heuristic algorithm for the m-machine, n-job flow sequencing problem”, in Omega, 11(1), pp. 91–95, 1983.

[12] Q. Pan and L. Wang, “Effective heuristics for the blocking flowshop scheduling problem with makespan minimization”, in *Omega*, 40 (2), pp. 218–229, 2012

[13] R. M. V. Rachamadugu, “Technical Note –A Note on the Weighted Tardiness Problem”, in Operations Research, 35, pp. 450–452, 1987.

[14] R. M. V. Rachamadugu and T.E. Morton, “Myopic heuristics for the single machine weighted tardiness problem”, Working Paper No. 28-81-82, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, PA, 1982.

[15] T. S. Raghu and C. Rajendran, “An efficient dynamic dispatching rule for scheduling in a job shop” in International Journal of Production Economics 32, pp. 301–313, 1993.

[16] C. Rajendran and O. Holthaus, “A comparative study of dispatching rules in dynamic flowshops and job shops”, in European Journal of Operational Research, 116 (1), pp. 156–170, 1999.

[17] I. Ribas, R. Companys, and X. Tort-Martorell, “An iterated greedy algorithm for the flowshop scheduling problem with blocking”, in Omega, 39, pp. 293–301, 2011.

[18] D. P. Ronconi and V. A. Armentano, “Lower Bounding Schemes for Flowshops with Blocking In-Process”, in Journal of the Operational Research Society, 52 (11), pp. 1289–1297, 2001.

[19] D. P. Ronconi “A note on constructive heuristics for the flow-shop problem with blocking”. International Journal of Production Economics, pp. 39–48, 2004.

[20] D. P. Ronconi “A branch-and-bound algorithm to minimize the makespan in a flowshop with blocking”, Annals of Operations Research, (138), pp. 53–65, 2005.

[21] D. P. Ronconi and L. Henrique. “Some heuristic algorithms for total tardiness minimization in a flow shop with blocking”, Omega, 37 (2), pp. 272–281, 2009

[22] R. S. Russel, E. M. Dar-El, and B. W. Taylor, “A comparative analysis of the COVERT job sequencing rule using various shop performance measures”, in International Journal of Production Research, 25 (10), pp. 1523–1540, 1987.

[23] K. D. Sweeney, D. C. Sweeney, and J. F. Campbell, “The performance of priority dispatching rules in a complex job shop: A study on the Upper Mississippi River”, in International Journal of Production Economics, Volume 216, pp. 154–172, 2019.

[24] E. Taillard. “Benchmarks for basic scheduling problems”, in European Journal of Operational Research, 64 (2), pp. 278–285, 1993.

[25] A. P. Vepsatainen and T. E. Morton, “Priority rules for job shops with weighted tardiness costs”, in Management Science 33/8, pp. 95–103, 1987.

[26] S. Voß and A. Witt, “Hybrid Flow Shop Scheduling as a Multi-Mode Multi-Project Scheduling Problem with Batching Requirements: A real-world application”, in International Journal of Production Economics 105, pp. 445–458, 2007.

[27] X. Wang and L. Tang, “A discrete particle swarm optimization algorithm with self-adaptive diversity control for the permutation flow shop problem with blocking”, in Applied Soft Computing, (12, 2), pp. 652–662, 2012.

[28] L. Wang, Q.-K. Pan, and M.F. Tasgetiren, “A hybrid harmony search algorithm for the blocking permutation flow shop scheduling problem”, in Computers & Industrial Engineering, 61 (1), pp. 76–83, 2011.

[29] C. Wang S. Song, S. J. N. D. Gupta, and C. Wu, “A three-phase algorithm for flowshop scheduling with blocking to minimize makespan”, in Computers & Operations Research, 39 (11), pp. 2880–2887, 2012.

[30] Z. Wu, S. Sun, and S. Yu, “Optimizing makespan and stability risks in job shop scheduling”, in Computers & Operations Research, Volume 122, 2020.

 ISBN: 978-1-61208-831-0