Integrated Workforce Scheduling and Flexible Flow Shop Problem in the Meat Industry

Beatrice Bolsi, Vinícius Lima, Thiago Queiroz, Manuel Iori

▶ **To cite this version:**

Beatrice Bolsi, Vinícius Lima, Thiago Queiroz, Manuel Iori. Integrated Workforce Scheduling and Flexible Flow Shop Problem in the Meat Industry. IFIP International Conference on Advances in Production Management Systems (APMS), Sep 2021, Nantes, France. pp.594-602, 10.1007/978-3-030-85902-2_63 . hal-04117674

HAL Id: hal-04117674
https://inria.hal.science/hal-04117674

Submitted on 5 Jun 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

This document is the original author manuscript of a paper submitted to an IFIP conference proceedings or other IFIP publication by Springer Nature. As such, there may be some differences in the official published version of the paper. Such differences, if any, are usually due to reformatting during preparation for publication or minor corrections made by the author(s) during final proofreading of the publication manuscript.

Integrated Workforce Scheduling and Flexible Flow Shop Problem in the Meat Industry

Beatrice Bolsi[1][0000−0002−6968−8689], Vinícius Loti de Lima[2][0000−0002−4805−4468], Thiago Alves de Queiroz[3][0000−0003−2674−3366], and Manuel Iori[1][0000−0003−2097−6572]

[1] Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia, 42122, Reggio Emilia, Italy, {beatrice.bolsi, manuel.iori}@unimore.it

[2] Institute of Computing, University of Campinas, 13083-852, Campinas-SP, Brazil, v.loti@ic.unicamp.br

[3] Institute of Mathematics and Technology, Federal University of Catalão, 75704-020, Catalão-GO, Brazil, taq@ufcat.edu.br

Abstract. We address a problem from a meat company, in which orders are produced in two stages, consisting of preparing meats on benches and allocating them to conveyors to be packed in disposable trays. In an environment where machines are unrelated, the company has to take daily decisions on the number and start time of working periods, the number of workers and their allocation to machines, and the scheduling of activities to satisfy the required orders. The objective of the problem is to minimize, in a lexicographic way, the number of unscheduled activities, the weighted tardiness, and the total production cost. To solve the problem, we propose a multi-start random constructive heuristic, which tests different combinations of number of workers in the machines and for each combination produces many different schedules of the orders. The results of our computational experiments over realistic instances show that the heuristic is effective and can support the company on its daily decisions.

Keywords: Food industry · Two-stage flexible flow shop problem · Workforce schedule · Multi-start random constructive heuristic.

1 Introduction

Production scheduling is one of the most common classes of problems faced by companies seeking to optimize their manufacturing system. In those problems, a set of jobs has to be scheduled in a set of machines while satisfying a set of practical constraints and optimizing a given objective. In this paper, we study a scheduling problem faced daily by a meat producing company. The company receives daily a set of orders to be produced in a single day. Each order is associated with a due date and is produced by following up to two stages: in the first stage, the meat is processed (cut) on a given bench, and in the second stage it is sent to a conveyor to be packed into disposable trays. Benches and conveyors

are seen as heterogeneous parallel machines, and their productivity depends on the number of workers operating each. Hence, the company needs to derive a new production plan every day, comprising the number of workers to operate each machine and the scheduling of operations composing the final orders. The scheduling problem faced by the company (which has a number of operational constraints that are formally discussed in Section 2) is a generalization of the well-known two-stage flexible (or hybrid) flow shop problem (see, e.g., [1]). The overall problem considers, in lexicographic way, the minimization of the number of unscheduled orders, the weighted tardiness, and the production costs.

The literature on flexible flow shop problems is broad. For surveys on general flexible flow shop variants, we refer the reader to [2], [3], and [4]. Concerning two-stage variants, in [5], the authors investigate the influence of repetitive scheduling in an environment with a single machine in the first stage and multiple lines in the second stage. In [6], we find an application related to a sterilization plant, aiming at reducing the number of tardy jobs and the makespan, while respecting sequence-independent setup times and jobs processed in parallel batches. The work of [7] handles a system with unrelated parallel machines on each stage and task tail group constraints, aiming at minimizing the total tardiness. The authors propose a new scheduling rule able to outperform twelve dispatching rules from the literature. In [8], the authors study environments composed of a single machine either in the first or in the second stage and propose several solution procedures for such problems. The work of [9] deals with a problem in the glass-ceramic industry, whose objective is to minimize the makespan and energy consumption. Besides an integer linear programming model, they propose constructive, tabu search, and ant colony optimization algorithms.

In this paper, we propose a multi-start random heuristic to solve the integrated workforce scheduling and two-stage flexible flow shop problem faced by the company. The heuristic iteratively tests different combinations for the number of workers, and, for each combination, it generates the production schedule by following a constructive heuristic. At each iteration, it assigns orders to benches and conveyors by following a list of priorities. By means of extensive computational experiments based on realistic instances, we can show that the heuristic is effective in finding good-quality solutions and can provide a quick support to the company on its daily decisions.

The remaining of the paper is organized as follows. Section 2 provides a formal description of the problem. Section 3 presents the proposed solution method. Section 4 provides the results of the computational experiments. Finally, Section 5 presents concluding remarks and some directions for future research.

2 Problem Definition

In this section, we present a detailed description of the problem. The company receives a set $\mathcal{O}$ of orders to be scheduled in a workday, by following several operational constraints. Each order is expected to be produced before its due date, which corresponds to the time in which it has to be shipped to its final

destination. Then, the problem is to determine the workforce and production schedule of a workday.

A workday. A workday is defined over a time horizon of 24 hours, and may have up to two disjoint working periods. The two periods are subject to time-related constraints, such as minimum and maximum start time, end time, and duration. Moreover, there is a minimum and a maximum number of workers that can be hired for each period. Thus, a first set of decisions involves the number of working periods, as well as their start time, end time, and hired workers.

A period in a workday. After deciding the duration and number of workers for each period, production-related decisions must be made. The production in each period is composed by two stages. Each stage has a set of heterogeneous parallel machines whose speed is variable. We assume workers have the same efficiency and the speed of each machine is proportional to the number of workers operating it. In addition, workers cannot be reallocated to different machines during a period. Thus, another decision is to determine the number of workers operating each machine during the period.

In our case study, first-stage machines are benches on which the meat is prepared, whereas second-stage machines are conveyors where the products are distributed to be packed into disposable trays that finally receive a stamp with the product information.

Details on the production in a period. The last set of decisions considers the scheduling of the orders in each period. Each order in $\mathcal{O}$ is associated with: a final product identification code; a due date; a type of disposable tray (referred to as the order family); a type of stamp; a net weight; a raw product type, which has a productivity measure on each machine of the first and second stage where it can be processed. In addition, some orders have to be processed in a single stage, whereas others have to be processed (sequentially) on both stages. For orders that have to be processed in both stages, the product quality is preserved by imposing a maximum waiting time (e.g., 60 minutes or so) for the buffer between the two stages. We assume the buffer has unlimited size. All machines have setup times: first-stage and second-stage machines must undergo setups whenever the raw product is changed and whenever different tray or stamp types are required, respectively. Another machine-related constraint imposes a fixed transportation time between the first and second stage.

Summarizing, the decisions of the overall problem involve the distribution of the working periods, the number of workers on each period, the number of workers operating each machine in each period, and the production scheduling. The quality of a solution is measured by three objectives to be minimized according to a lexicographic order: (1) the total number of unscheduled orders; (2) the total weighted tardiness; and, (3) the total production cost. Although machine setups are not formally included in the objective, they are highly disliked by the company and it is always preferable to keep their number as small as possible.

3 Proposed Heuristic

To solve the problem under consideration, we propose a multi-start random heuristic. First, let us describe function `schedule`$(\mathcal{O}, L)$, which schedules orders in $\mathcal{O}$ to a period that is already set (i.e., its time window is already fixed and workers are already allocated to machines). The list L has a priority coefficient l_o for each order $o \in \mathcal{O}$. The orders are iteratively scheduled by choosing the best order to be scheduled at each iteration based on the priority list in use, which considers, for instance, the creation of setups, the due dates, the coefficients, the size of the gaps created between productions. Whenever necessary, the sequence in $\mathcal{O}$ is used to break ties. Orders with early due dates that are processed in a single stage only are always chosen to be scheduled first. Whenever we are about to schedule orders that are processed in both stages, we first choose the second-stage machine with earliest available time, then we choose the best order to be scheduled in that machine following one of the priority lists, and, finally, we choose the best first-stage machine to schedule the chosen order, also based on a priority list.

The function `schedule` is used within a constructive heuristic for the integrated problem. The overall structure of this heuristic is presented in Algorithm 1. The algorithm creates several combinations of first- and second-period schedules by testing different combinations of number of workers and by iteratively updating the list of priority coefficients. The algorithm begins by testing different combinations of number of workers for the first period, and for each combination, the workers are allocated to machines by means of a function that consider a balance of the workload. After creating the schedule for the first working period, the heuristic follows by testing different possible schedules of the remaining orders in the second period. Again, for the second period all possible combinations of number of workers are tested and the workload based function is used again to allocate the workers to the machines. Then, different schedules for the second working period is created for each combination of workers. During the process of creating a schedule for a period, after fixing the number of workers for any of the two periods, we produce up to Π_{list} different schedules, which are iteratively obtained by updating the list L and calling the function `schedule`. Recall that L is used in the internal of `schedule` as an order-priority quantifier. In this way, L is updated in order to avoid tardiness, by increasing the priority coefficients of tardy orders in the current schedule.

Finally, our multi-start random heuristic is obtained by running Algorithm 1 for Π_{shuffle} iterations. In each of these iterations, we perform a reshuffle of the order sequence in $\mathcal{O}$. Recall that the sequence in $\mathcal{O}$ is used in a tie-breaking process in the internal of the `schedule` function, and, thus, it is expected to produce an impact in the final solution.

4 Computational Experiments

We implemented the multi-start random heuristic algorithm presented in Section 3 in C++. The algorithm was tested on 12 realistic instances, each representing

Algorithm 1 Heuristic for the integrated problem

```
L ← (0, 0, ..., 0) ∈ ℤ^|𝒪|. // list of priority coefficients associated with 𝒪
for all combinations of number of workers of stage one and two of the first working period do
    Assign workers to machines of the first working period by considering a workload balance
    for i ← 1, ..., Π_list do
        t1 ← schedule(𝒪, L)
        𝒪' ← 𝒪 minus the orders scheduled in the working period t1
        L' ← L updated for the orders in 𝒪'
        for all combinations of number of workers of stage one and two of the second working period do
            Assign workers to machines of the second working period by considering a workload balance
            for j ← 1, ..., Π_list do
                t2 ← schedule(𝒪', L')
                Update the best solution if its (lexicographic) objective function is worse than that of the current solution (t1, t2)
                if there is tardiness in the working period t2 then
                    Increase the coefficient of priority in L' of the orders with tardiness
                else
                    Break the loop
                end if
            end for
            if there is tardiness in the working period t1 then
                Increase the coefficient of priority in L of the orders with tardiness
            else
                Break the loop
            end if
        end for
    end for
end for
```

one day of production at the company. Some randomization was applied to all instances in order to meet the company's privacy requirements. The tests were executed in a computer with an Intel Core i7 1.2 GHz processor and 8 GB of RAM. The goal of the experiments is to study the impact of different parameters in the performance of the heuristic. The parameters under consideration are:

(i) the range of workers available for each working period;
(ii) the number Π_{shuffle} of random shuffles of the sequence $\mathcal{O}$;
(iii) the number Π_{list} of iterations related to the update of the priority coefficients of the orders to try to prevent tardiness.

For each parameter configuration, we evaluate the three objectives of the problem plus the number of setups in the second stage.

We present in Table 1 the results related to parameter (i), which is the number of workers hired in each working period. We consider three configurations: L,

meaning the Lowest fixed number of workers we may have; H, meaning the Highest fixed number of workers; and, R, meaning a Range/variable number of workers from L to H. These results were obtained with $\Pi_{\text{shuffle}} = 25$ and $\Pi_{\text{list}} = 5$, that are values defined after preliminary trial and error tests. By observing the results, we can note that the higher is the number of workers, the better is the solution. However, the best results are obtained when the heuristic has a range of workers (situation R) to decide on. For all instances, configurations R and H have provided the same number of unscheduled orders and weighted tardiness (except for $I11$, where R is better), but with R the production cost is always equal or smaller. The number of setups in stage two is small and comparable among all configurations. The average computational time in seconds for L, H and R is 9.2, 8.7 and 1666.3, respectively, so R requires a consistently larger amount of time to achieve the improved solutions.

As better solutions are obtained with configuration R, the following results consider only this configuration. In Table 2, we present the results we obtained when evaluating parameter (ii), the number of Π_{shuffle} iterations. We attempted three values, namely 1, 25, and 100. When Π_{shuffle} is set to 1, the heuristic looses its multi start characteristic. Hence, we expect to have better solutions as Π_{shuffle} increases, because we may change the sequence in which orders are scheduled. In these experiments, we have set $\Pi_{\text{list}} = 5$.

Table 1. Results for different (i) values of works in each working period.

Inst.	Unsch. Jobs			Weight. Tardiness			Prod. Cost			N. Setups		
	L	H	R	L	H	R	L	H	R	L	H	R
I01	2	0	0	68795	0	0	192	223	206	2	2	2
I02	2	0	0	1182	0	0	280	314	301	2	2	2
I03	0	0	0	7811	0	0	317	326	319	2	2	2
I04	0	0	0	209663	31738	31653	210	244	240	2	1	2
I05	3	0	0	25212	2414	2414	229	270	266	2	2	2
I06	13	1	1	156181	37174	37174	241	272	272	1	1	1
I07	3	0	0	26965	0	0	259	285	285	1	2	2
I08	0	0	0	0	0	0	91	95	92	1	2	2
I09	0	0	0	124397	46	46	291	344	344	2	2	2
I10	3	0	0	48977	0	0	318	332	328	1	2	2
I11	12	2	2	252213	255366	76081	329	380	373	1	1	1
I12	15	2	2	119295	197832	197832	340	395	395	1	1	1

Observing Table 2, the heuristic returns better solutions when more iterations are available. This means that restarting with possible different orders has a positive impact on the final solution even if the order sequence is the last criterion in the list of priorities. With 25 and 100 iterations, the heuristic is able to obtain the same number of unscheduled orders for all instances, although better weighted tardiness and production costs are achieved with 100 iterations for 3 instances (see $I06$, $I09$, and $I11$). The number of setups is relatively small and within the target value indicated the company (which is 3). The average computational time in seconds for Π_{shuffle}=1, 25, and 100 is 0.4, 7.6 and 30.8, respectively, so all three configurations are fast.

Table 2. Results for different (ii) numbers of Π_{shuffle} iterations.

Inst.	Unsch. Jobs			Weight. Tardiness			Prod. Cost			N. Setups		
	1	25	100	1	25	100	1	25	100	1	25	100
I01	0	0	0	525	0	0	213	206	206	1	2	2
I02	1	0	0	1118	0	0	304	301	301	1	2	2
I03	0	0	0	91	0	0	334	319	319	1	2	2
I04	0	0	0	51765	31653	31653	238	240	240	2	2	2
I05	0	0	0	18895	2414	2414	268	266	266	2	2	2
I06	4	1	1	117886	37174	16019	283	272	277	1	1	2
I07	0	0	0	244	0	0	295	285	285	2	2	2
I08	0	0	0	0	0	0	92	92	92	2	2	2
I09	0	0	0	1348	46	36	332	344	316	2	2	2
I10	0	0	0	0	0	0	347	328	328	1	2	2
I11	2	2	2	76081	76081	40502	373	373	371	1	1	1
I12	3	2	2	99445	197832	197832	399	395	395	1	1	1

Finally, we present in Table 3 the results for parameter (iii), the Π_{list} number of iterations to change the order coefficient of priority. For these results, we selected configuration R and $\Pi_{\text{shuffle}} = 100$, and tested Π_{list} equal to 1, 5, and 20. Overall, we have conclusions similar to the previous ones, meaning that higher values of Π_{list} can provide better solutions, especially if comparing the weighted tardiness and the production cost. The average computational time in seconds for Π_{list}=1, 5, 20 is 4.7, 30.8 and 177.1, respectively, which is again fast and compatible with a real-world use of the algorithm by a decision maker.

Table 3. Results for different (iii) numbers of Π_{list} iterations.

Inst.	Unsch. Jobs			Weight. Tardiness			Prod. Cost			N. Setups		
	1	5	20	1	5	20	1	5	20	1	5	20
I01	0	0	0	22069	0	0	211	206	206	1	2	2
I02	0	0	0	308	0	0	313	301	301	1	2	2
I03	0	0	0	100883	0	0	327	319	318	1	2	2
I04	0	0	0	117752	31653	31650	222	240	241	2	2	2
I05	0	0	0	23168	2414	2411	265	266	264	1	2	1
I06	2	1	1	83453	16019	9137	285	277	275	1	2	2
I07	0	0	0	162	0	0	287	286	286	2	2	2
I08	0	0	0	0	0	0	92	92	92	2	2	2
I09	0	0	0	99604	36	0	302	316	292	2	2	1
I10	0	0	0	13829	0	0	321	328	328	0	2	2
I11	2	2	2	90988	40502	11029	365	371	366	1	1	1
I12	3	2	2	99445	197832	197832	399	395	395	1	1	1

5 Concluding Remarks

We have proposed a multi-start random heuristic for a complex integrated workforce scheduling and two-stage flexible flow shop problem from a meat production system. The heuristic schedules orders by following a list of priorities and considering different workers on machines in order to reach solutions with all

orders scheduled, no tardiness, and the smallest possible production cost. The computational experiments have indicated that better solutions can be achieved when allowing more shuffles in the vector of orders (Π_{shuffle}), more iterations to handle the orders with tardiness (Π_{list}), and allowing the heuristic to decide on the number of works to set in each working period. In addition, even these more expensive settings have allowed practical computational time.

In future research, we are interested in studying how dynamic changes in assignments of workers to machines may affect productivity. Another direction is related to the extension of the proposed heuristic to include a local search step and possibly a tabu list in order to escape from local optima solutions. Finally, the proposed algorithm has been recently deployed to the company and, as attested by the company, it has been producing satisfactory results. We are currently working with the company to provide, as future research, an extensive comparison regarding the impact of the algorithm on their production system.

Acknowledgements

Vinícius Loti de Lima would like to thank the support of the São Paulo Research Foundation (process number 2017/11831-1). Thiago Alves de Queiroz acknowledges support by the National Council for Scientific and Technological Development (process number 311185/2020-7).

References

1. Emmons, H., Vairaktarakis, G.: The Hybrid Flow Shop, pp. 161–187. Springer US, Boston, MA (2013)
2. Linn, R., Zhang, W.: Hybrid flow shop scheduling: A survey. Computers & Industrial Engineering **37**(1), 57–61 (1999)
3. Ruiz, R., Vázquez-Rodríguez, J.A.: The hybrid flow shop scheduling problem. European Journal of Operational Research **205**(1), 1–18 (2010)
4. Tian-Soon Lee, Y.T.L.: A review of scheduling problem and resolution methods in flexible flow shop. International Journal of Industrial Engineering Computations **10**(1), 67–88 (2019)
5. Tsubone, H., Suzuki, M., Uetake, T., Ohba, M.: A comparison between basic cyclic scheduling and variable cyclic scheduling in a two-stage hybrid flow shop. Decision Sciences **31**(1), 197–222 (2000)
6. Rossi, A., Puppato, A., Lanzetta, M.: Heuristics for scheduling a two-stage hybrid flow shop with parallel batching machines: application at a hospital sterilisation plant. International Journal of Production Research **51**(8), 2363–2376 (2013)
7. Li, Z., Chen, Q., Mao, N., Wang, X., Liu, J.: Scheduling rules for two-stage flexible flow shop scheduling problem subject to tail group constraint. International Journal of Production Economics **146**(2), 667–678 (2013)
8. Hwang, F., Lin, B.: Survey and extensions of manufacturing models in two-stage flexible flow shops with dedicated machines. Computers & Operations Research **98**, 103–112 (2018)
9. Wang, S., Wang, X., Chu, F., Yu, J.: An energy-efficient two-stage hybrid flow shop scheduling problem in a glass production. International Journal of Production Research **58**(8), 2283–2314 (2020)