Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Retail order picking scheduling with missing operations and limited buffer

Sawssen SOUIDEN [*,**] **Audrey CERQUEUS** [*] **Xavier DELORME** [*]
Jean-Lucien RASCLE [**]

[*] *Mines Saint-Etienne, Univ Clermont Auvergne, CNRS, UMR 6158 LIMOS, Institut Henri Fayol, F - 42023 Saint-Etienne France. (e-mail: sawssen.souiden@emse.fr; sawssen.souiden@boaconcept.com).*
[**] *Boa Concept SAS, 42000 Saint-Etienne, France.*

Abstract: Order picking is one of the most critical activities in the warehousing process. In order to obtain high profits, e-commerce retailers have focused on dedicated and accurate order picking methods. Zone picking is a picking approach commonly used for retail order picking where small-sized items have to be collected. Nevertheless, to become more efficient, it requires a scheduling policy for managing the bins flow which avoids blocking and mitigates order pickers unproductive times. This paper studies a specific variant of the Non-Permutation Flow-Shop (NPFS) scheduling problem with missing operations, transportation times and limited capacity constraints. A mathematical formulation of this new problem is proposed to minimize the total elapsed time for all machines. We study an illustrative example to analyze the differences between the optimal solution of the proposed model and a naive solution.

Keywords: Order picking, Scheduling, Non-Permutation Flow-Shop, Missing operations, Limited buffer capacity, Mixed integer linear programming.

1. INTRODUCTION

Order picking is one of the costly warehouse activities consisting of retrieving products or items that need to be retrieved from storage locations in order to fulfill a given customer's order. The most common order picking deployed in traditional warehousing systems is the manual picking, also denoted as "pick-by-order" or "picker-to-parts". Thereby, the order picker must travel between shelves to collect items until the completion of the order. Therefore, the efficiency of this process depends intrinsically on the order picker performance and may lead to several critical problems. The first problem is centered around routing issues in which an optimal route sequence is required to minimize the pickers travelling time (De Santis et al., 2018). The second problem, often identified as a batching problem, consists of regrouping clients' orders into batches before the collect activity (Hong et al., 2012). Hence, the aim is to find the best combination of orders to allocate to each batch. The third problem is assigning items to storage locations in the warehouse before the picking process in order to balance the workload between pickers (Scholz et al., 2017).

In the past decade, online ordering has emerged rapidly in the field of e-commerce. Based on the concept of retail order picking where items are small and delivery dates are tight, online retailers faced new challenges not only to satisfy final customers' requirements but also to ensure the efficiency of the order picking process (Boysen et al., 2019). Hence, one of the alternatives is to change the warehousing policies by switching to new order picking methods such as semi-automated or fully automated order picking. This method integrates technological systems to facilitate the order picking process and to reduce the amount of time a picker spends searching for the next pick.

Zone picking, which is the main focus of this paper, is one example of the semi-automated order picking methods. This method consists of dividing the warehouse into several zones that are interconnected with a conveyor system. One order picker is assigned to each zone. The picker's travel times are reduced thanks to the proximity of the items within his zone. This allows the pickers to focus more on the picking process rather than optimizing unnecessary travel.

Zone picking problems have been tackled from several perspectives: from strategic and tactical viewpoints such as storage location assignment and control decisions (Petersen, 2002), from an operational standpoint by integrating the zone picking problem with other order picking problems like routing (Ying-Chin and Jian-Wei, 2017) and batching (Le-Duc et al., 2005), or even focusing on the organisational issues, for example, (Huang et al., 2018) solved an order batch scheduling problem under zoning conditions.
Despite the efficiency of zoning configuration, the orders flow is rather complex to manage in practice since it may generate unproductive times (the order pickers may wait too long for the bins arrival) or cause a blocking of the conveyor system.
In this paper, we propose a new optimization problem corresponding to an emerging industrial issue related to the field of retail order picking, with the aim of handling the flow of a large number of small orders. Thus, the problem is modeled as a scheduling problem in which specific real conditions of the zone picking process are considered: zone skipping, buffers storage constraints and transportation times between the zones.

The remainder of this paper is organized as follows. The next section describes the considered problem. Section 3 will be dedicated to the literature reviews of related works. Section 4 presents the mathematical model and provides a linear formulation. An illustrative example is discussed in Section 5. Finally,

Section 6 presents the conclusions of this work and outlines future research directions.

2. PROBLEM DEFINITION

In this work, we consider a retail order picking line organized as a zone picking with sequential zoning. As shown in Fig. 1, the picking area is divided into several zones (also called stations), and one order picker is assigned to each station. The stations are connected by a conveyor allowing the bins transportation through the picking zones.

Reconfiguring a warehouse by reorganizing the picking zones or allocating the items to the stations according to the daily customers orders is not operationally feasible. Thus, we consider in this work that the items are already assigned to the stations. The items composing the orders are given, thus we can deduce the picking stations to visit. The bins have different stations to visit and may skip some zones which do not contain any ordered items. Hence, even if the bins are processed in the order of arrival in the stations, their order can differ from a station to another, since they can stop on different stations.
Once the order picker collects all the items needed for a bin on his station, he pushes back the bin on the conveyor in order to be transported to the next station. The picker can either deal with the bin immediately or let it wait in the buffer. The order picker prepares the bins in the order of their arrival in the machines buffer, thereby, the bins are actually handled in the first position in the buffers. We assume that virtually the first position of the buffer constitutes the picking station as depicted in Fig. 1. Nevertheless, the buffers' capacity is limited. If a bin arrives when the buffer is full, it remains on the conveyor belt while the machines buffer is saturated and hence hamper the bins flow. To avoid such congestion, overtaking the buffers' capacity is forbidden in our study. Moreover, we consider that the order picker must remain at his picking station from the moment he begins the preparation of the first bin until the completion of the last prepared bin. In other words, we aim to regroup as much as possible the working time periods in order to minimize the times spent by the order pickers at their picking stations. In such labor organisation, pickers can be assigned to other value-added activities in the warehouse, outside order picking.

From the problem definition, the factors that must be taken into account are the missing operations, the transportation times between the zones and the limited buffer capacities.
Since bins skip some machines, the bins processing sequence differs from one machine to another. This work employs the concept of job scheduling in the Non-Permutation Flow-Shop (NPFS) structure where the jobs order changes on different machines. We will henceforth consider the bin as a job and the picking station or the order-picker as a machine.

3. LITERATURE REVIEW

The Non-Permutation Flow-Shop (NPFS) scheduling problem (Tandon et al., 1991) is a generalization of the traditional Permutation Flow-Shop (PFS) scheduling problem (Johnson, 1954). In the PFS scheduling, the job sequence is identical for all machines whereas in a NPFS the sequence of jobs can be different on subsequent machines.

PFS and NPFS have similar constraints. Each machine can handle only one job at a time and each job has a fixed processing time on each machine. The change in the order of the jobs on a machine comes generally from the presence of buffers between machines (the order being changed in these buffers) or the presence of missing operations inducing different paths for the jobs and thus different permutations of jobs in the processing sequence.
In real-world case studies, NPFS gives more flexibility. For instance, jobs are not required to visit all machines or can be interchanged in a buffer. Rossit et al. (2018) reviewed and classified the NPFS problem in terms of objective functions for different variants and presented the optimization techniques used for solving NPFS.

Over the past decades, PFS and NPFS have received large attention, considering several optimization criteria and different types of constraints. In the literature, several surveys addressed the NPFS problem in a comparative context with the PFS approach in order to outline the performance of each configuration. The main objective functions considered in their studies are completion-time based objectives (Potts et al., 1991; Nagarajan and Sviridenko, 2009) and due dates based objectives (maximum tardiness and total tardiness in Liao and Huang (2010)). Liao et al. (2006) investigated all of those aforementioned criteria.

NPFS generally requires the existence of a buffer to generate feasible schedules. Nevertheless, to the best of our knowledge, most of the surveys studying flow-shop problems with capacity constraints consider a PFS configuration rather than the NPFS's one. Leisten (1990) studied PFS and NPFS scheduling problems by taking into account a finite buffer capacity and it also investigated the cases in which no buffer and unlimited capacity are considered. Brucker et al. (2003) tackled a PFS problem with the consideration of buffers with a limited capacity between every two consecutive machines, minimizing the makespan. The authors proved a relationship between stock size and the difficulty of solving the problem. Moslehi and Khorasanian (2014) investigated the limited buffer PFS scheduling problem optimizing makespan.

Only a few studies considered a limited buffer for NPFS and even fewer gave a nonlinear mathematical formulation. For example, Rossi and Lanzetta (2013) modeled the scheduling of a NPFS with buffers minimizing the makespan. The authors solved the problem by applying heuristic and meta-heuristic methods.

In recent decades, the presence of the missing operations in flow-shop scheduling problems gained interest. It allows considering situations in which some or all jobs skip some machines. Pugazhendhi et al. (2003) studied the NPFS problem with missing operations. They presented a simple heuristic procedure to optimize makespan and total time. Sadjadi et al. (2008) investigated three NPFS problems with missing operations by taking into account some specific constraints. They examined the constraints of time lags for the first problem and the sequence-depend setup time in the second one. In these first two problems, the authors considered the minimization of the makespan. In the third problem, they studied the minimization of the total weighted tardiness. Ramezanian et al. (2011) addressed the NPFS problem with missing operations, and solved it with a Genetic Algorithm and Tabu Search methods.

The consideration of the transportation times in the NPFS scheduling problem has received less attention. Most of the works considered that the transportation times between machines is negligible. However, when transportation times are

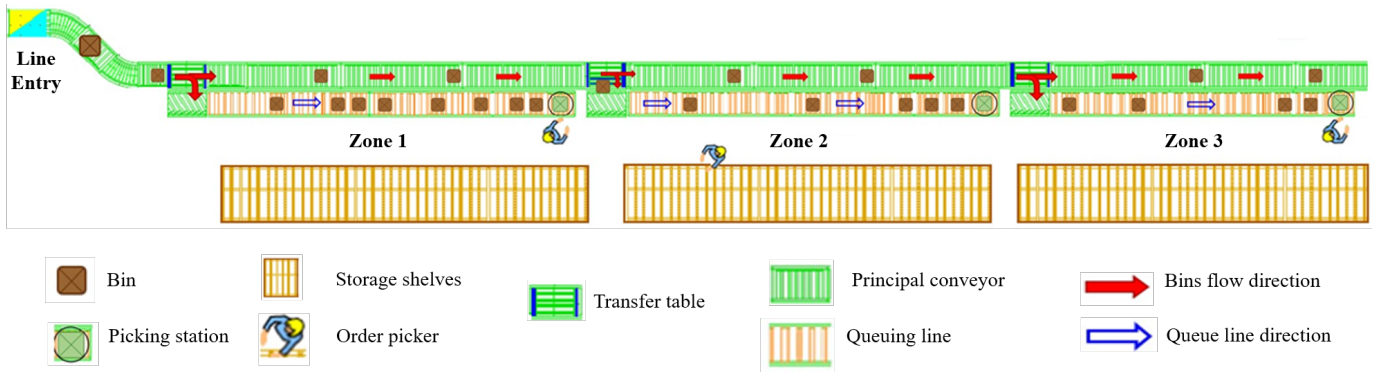

Fig. 1. A schematic illustration of a zone picking system

taken into consideration, generally the resources associated with transportation have limited capacity (Lamoudan et al., 2012; Chung-Yee and Zhi-Long, 2001).
Several studies in the literature address mathematical formulations for flow-shop scheduling. The authors consider several types of formulations based on different approaches. The assignment and positional approach consists of assigning jobs to positions in a permutation (Fang et al. (2013) for the PFS, Vahedi-Nouri et al. (2013); Rossi and Lanzetta (2013); Sadjadi et al. (2008) for the NPFS). The disjunctive approach uses decision variables on the precedence between each pair of jobs (Fang et al. (2013) for PFS). Finally, the time-indexed approach (Avella et al. (2017) for FPS). As far as we know, there is no formulation yet for NPFS scheduling problems based on the disjunctive approach and time indexed approach.

In summary, the problem studied in this paper is a new variant of the NPFS scheduling problem in which limited buffer capacity constraints, missing operations and transportation times are simultaneously taken into consideration. To the best of our knowledge, the integration of these features has not been tackled yet. Thus, we present a formalization of the problem and propose a linear mathematical optimization model.

4. MATHEMATICAL MODEL

We consider a set of n jobs $\mathscr{J} = \{1,2,\ldots,n\}$ to be processed on a set of m machines $\mathscr{M} = \{1,2,\ldots,m\}$. The set of machines visited by each job is given by a_{ij} (a_{ij} is equal to 1 if job j visits machine i, 0 otherwise). The machines are arranged in an ordered sequence. Thus, by considering a_{ij}, we can define for each job j the sequence of machines visited, which is presented by a set P_j of ordered pairs of successive machines, i.e. (i,i'). Note that i never takes the value m and similarly i' never takes the value 0. k_i defines for each machine i the number of jobs that visit it. Furthermore, m upstream buffers B_i are associated with each machine i with a capacity of b_i units. We decided to add an additional machine at the beginning of the line, seen as a launching point, in order to consider a modeling with intermediate buffers configuration. This will facilitate the modeling of the limited buffers constraints. Hence, from now on, a new set of machines will be considered in the problem, including this fictitious machine, denoted $\mathscr{M}^* = \mathscr{M} \cup \{0\}$. The processing time of the job $j \in \mathscr{J}$ in machine $i \in \mathscr{M}$ is given by p_{ij}. For machine 0, all jobs have a processing time equal to 0. Transportation times to carry jobs between two machines i and i' are taken into account ($T_{i,i'}$).

Inspired by the formulation of Fang et al. (2013), we propose a mixed integer program that uses binary variables to directly assign jobs to positions on machines. Thus, we assign an index k to determine the processing order of job j in machine i.

The decision variables are the following :

- $x_{i,j,k}$ = 1, if the job j is assigned to machine i in the k^{th} position of its sequence, 0 otherwise.
- $s_{i,j}$ is the starting time of job j on machine i.
- $S_{i,k}$ is the starting time of the job scheduled in the k^{th} position on machine i.

The aim of this paper is to minimize the total elapsed time for all machines which corresponds to the minimization of the order pickers cumulative presence time in our study context.

The mixed-integer linear programming model for this NPFS problem is:

$$\min \sum_{i=1}^{m} \left(S_{i,k_i} + \sum_{j=1}^{n} p_{i,j}\, x_{i,j,k_i} - S_{i,1} \right), \tag{1}$$

subject to:

$$\sum_{k=1}^{k_i} x_{i,j,k} = a_{i,j} \quad \forall i \in \mathscr{M}^*; \forall j \in \mathscr{J}; \tag{2}$$

$$\sum_{j=1}^{n} x_{i,j,k} = 1 \quad \forall i \in \mathscr{M}^*; \forall k = 1,\ldots,k_i; \tag{3}$$

$$s_{i,j} + p_{i,j} + T_{i,i'} \leq s_{i',j}, \forall j \in \mathscr{J}; \forall i, i' \in \mathscr{M}^*,\ (i,i') \in P_j; \tag{4}$$

$$S_{i,k} + \sum_{j=1}^{n} p_{i,j}\, x_{i,j,k} \leq S_{i,k+1}, \forall i \in \mathscr{M}^*; \forall k = 1,\ldots,k_i - 1; \tag{5}$$

$$S_{i,k} \leq s_{i,j} + \left(1 - x_{i,j,k}\right) H \ \forall i \in \mathscr{M}^*; \forall j \in \mathscr{J}; \forall k = 1,\ldots,k_i; \tag{6}$$

$$s_{i,j} \leq S_{i,k} + \left(1 - x_{i,j,k}\right) H \ \forall i \in \mathscr{M}^*; \forall j \in \mathscr{J}; \forall k = 1,\ldots,k_i; \tag{7}$$

$$\begin{aligned} S_{i',(k'-b_{i'})} + \sum_{j'=1}^{n} p_{i',j'}\, x_{i',j',(k'-b_{i'})} \\ \leq S_{i,k} + p_{i,j} + T_{i,i'} + \left(2 - x_{i,j,k} - x_{i',j,k'}\right) H \end{aligned} \tag{8}$$

$$\forall j \in \mathscr{J}; \forall i \in \mathscr{M}^*, i' \in \mathscr{M}, (i,i') \in P_j, \forall k = 1,\ldots,k_i; \forall k' = b_{i'}+1,\ldots,k_{i'}.$$

$$S_{0,k} + \delta \leq S_{0,k+1}, \forall k = 1,\ldots,n-1; \tag{9}$$

$$S_{i',k'}+p_{i',q}+T_{i',i''}+\delta \leq S_{i,k}+p_{i,j}+T_{i,i''} + \left(4-x_{i',q,k'}-x_{i'',q,k''-1}-x_{i,j,k}-x_{i'',j,k''}\right) H, \quad (10)$$
$$\forall j,q\in\mathscr{J}, j\neq q; \forall i,i'\in\mathscr{M}^*, i''\in\mathscr{M}, (i,i'')\in P_j, (i',i'')\in P_q;$$
$$\forall\, k=1,\ldots,k_i; \forall k'=1,\ldots,k_{i'}; \forall k''=2,\ldots,k_{i''}.$$

$$x_{i,j,k}\in\{0,1\} \quad \forall i\in\mathscr{M}^*; \forall j\in\mathscr{J}, \forall k=1,\ldots,k_i. \quad (11)$$

$$s_{i,j}\geq 0 \quad \forall i\in\mathscr{M}^*; \forall j\in\mathscr{J}; \quad (12)$$

$$S_{i,k}\geq 0 \quad \forall i\in\mathscr{M}^*; \forall k=1,\ldots,k_i; \quad (13)$$

The equation (1) presents the objective function (the cumulative presence time of operators), i.e. the sum on all machines of the difference between the end of the processing of the job scheduled last, and the starting time of the first job scheduled on the same machine. Constraints (2) ensure that the assignment of jobs in the processing sequence of a machine includes the jobs visiting it. Constraints (3) specify that exactly one job is scheduled to each position k of the processing sequence of machine i. Constraints (4) guarantee that each job is processed entirely on a machine and has the time to arrive to the next machine before its starting time on this next machine. Constraint set (5) ensures that a machine cannot start processing a job until the completion of the previous job in the sequence. Constraints (6) and (7) state linking constraints between the starting times $s_{i,j}$ and $S_{i,k}$, H is an arbitrarily large number. Constraints (8) insure that the capacity of the buffer is never exceeded. The left-hand side is the completion time of the job in the $(k'-b_{i'})^{th}$ position in the machine's processing sequence. The right-hand side is the arrival time of the job occupying the k'^{th} position in the same processing sequence. If the k'^{th} job and the $(k'-b_{i'})^{th}$ job are simultaneously in the buffer, then it means that the buffer is saturated. Hence, the constraints (8) insure that the job at position k'^{th} arrives in the machine's buffer after the job in the $(k'-b_{i'})^{th}$ position has left the machine. Constraints (9) ensure that jobs are not launched at the same moment on the machine 0, which implies that jobs visiting the same machine will not arrive at the same time. δ defines a minimum time between the arrival of two successive jobs at the same machine. Constraints (10) guarantee that jobs are processed in the order of their arrival at the buffer. Constraints (11) to (13) are the domain of the decision variables.

5. A NUMERICAL ILLUSTRATION

For a better understanding of the problem's specificities, an illustrative instance will be presented and analyzed. We compare a naive solution (NS), which could be applied in a company that does not integrate optimization techniques, to the optimal solution (OS) of the mathematical problem presented in the previous section and obtained using the commercial solver IBM ILOG Cplex Optimization Studio.

5.1 Description of the illustrative instance

We consider an industrial problem involving 4 machines and 10 jobs to be scheduled. Processing times are given in Table 1 in which each missing operation is represented by a dash (i.e. when a job skips a machine). According to the mathematical formulation in section 4, and in order to determine the jobs launching time in the flow (i.e. the launching time of each bin in the conveyor system), we add a fictitious machine denoted machine $M0$ which will be visited by all jobs with a processing time of 0. Hence, the sequence of machines to visit by each job can be deduced from the processing times given in Table 1.
Table 2 presents the transportation times between machines. We recall that machines are visited in the order of their index. We consider that the capacity of every buffer is equal to 2 (including the machine's capacity). The minimum time between the arrival of two jobs at the same machine is $\delta = 1$s.

Table 1. Processing times (in s)

Machine	Job									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
0	0	0	0	0	0	0	0	0	0	0
1	64	96	-	9	17	19	52	-	54	-
2	-	-	16	39	-	-	18	35	-	-
3	48	-	34	22	12	78	-	24	-	10
4	-	16	11	73	-	16	-	14	-	93

Table 2. Transportation times (in s)

Machine	1	2	3	4
0	94	96	98	99
1		53	64	83
2			50	52
3				46

5.2 Solutions comparison

To generate the naive solution, jobs are launched on the conveyor in increasing order of their index. Furthermore, their processing in a machine respects the order of their arrival in the buffer. The departure time of a bin from a machine is computed such that the buffer of the next machine to visit is not saturated at the time of the bin's arrival. This can induce a waiting time for a bin in a station even after the processing completion in this station. The bin is considered to be in the buffer during this waiting time. This solution is represented by a Gantt chart in Fig. 2. We represented the job processing in a machine by a box. The jobs stored in buffers are depicted by lines. Two parallel lines mean that two jobs are stored in a machine's buffer while another job is processed in this machine. In addition, when a job is processed in a machine, at most one job could be stored in a buffer, since the processing of the job is done in the buffer.

The job sequence is different for each machine even if we consider the jobs in their order of arrival in the machines which shows that the mere presence of missing operations implies an NPFS. We remind that we aim to minimize the total elapsed time for all machines (the cumulative presence time of operators), i.e. minimizing the unproductive times between the beginning of the first process in a machine and the end of the process of the last job in the same machine.

The Gantt chart of the NS shows that the adopted approach generated considerable non-activity times on machines $M2$, $M3$ and $M4$. Indeed, the NS makes sure that each job is processed without delay (when the job arrives at the buffer it is processed as soon as the machine is available). Nevertheless, this naive approach may generate a flow block. For instance, if job $J4$ had left directly the machine $M2$ after being processed to visit the next machine $M3$ then 3 jobs would be at its upstream buffer. Therefore, the system's flow will be blocked. To avoid this issue, when the buffer of the next machine to visit is full, the job will be held in the buffer of the current machine, thereby

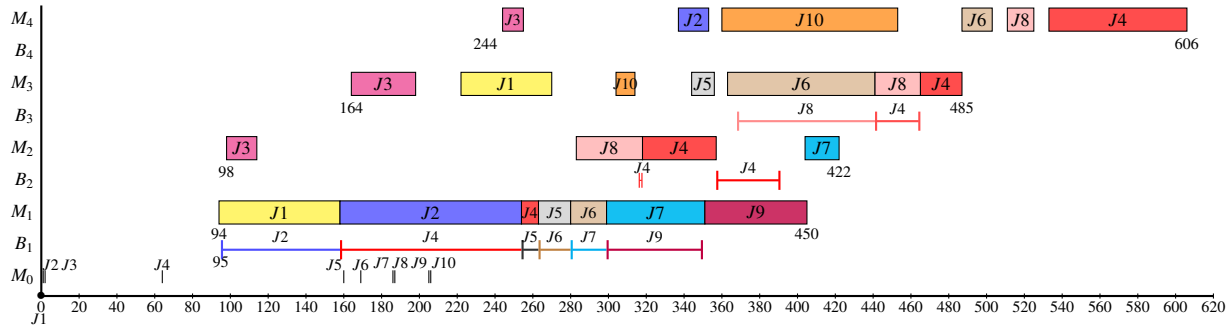

Fig. 2. Gantt chart of the naive solution

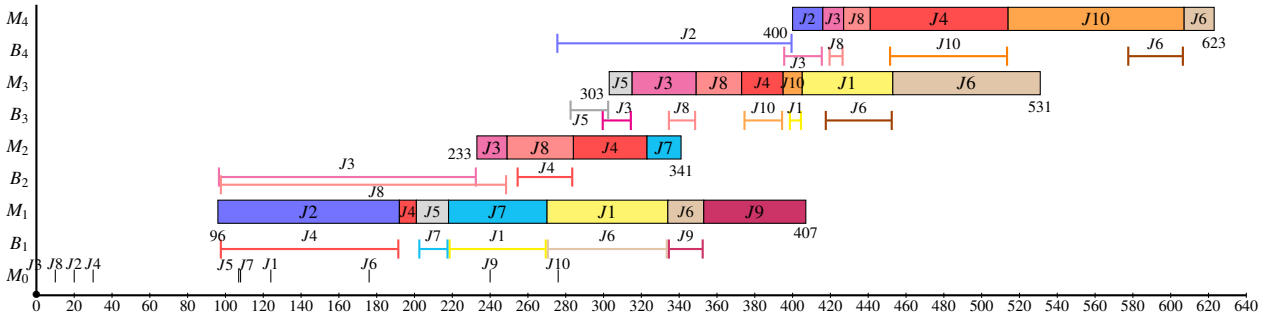

Fig. 3. Gantt chart of the optimal solution without makespan constraint

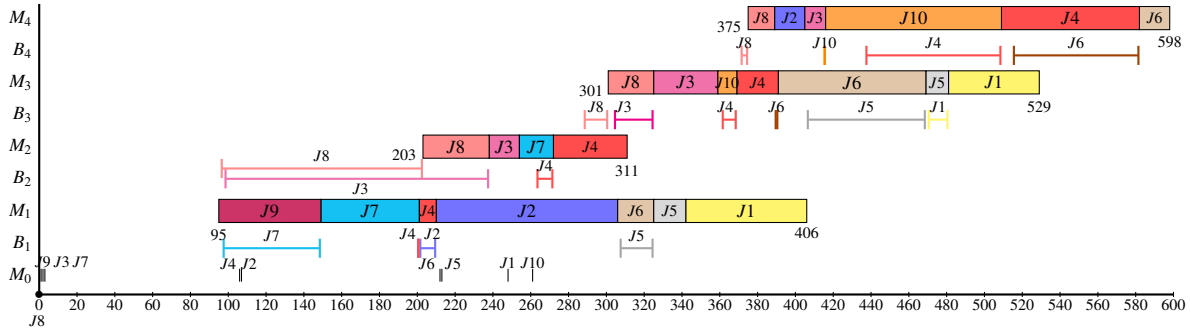

Fig. 4. Gantt chart of the optimal solution with makespan constraint

allowing the processing of the next job (see Fig. 2). We recall that the job is physically in the buffer area during its processing. During the waiting time, it stays in the same position in the buffer. In practice, the order picker will tend to send the bins as soon as the order picking is finished.

Fig. 3 illustrates the Gantt chart of the optimal solution obtained by minimizing the cumulative presence time of operators according to our model's constraints. In order to avoid unproductive times, the model delays the processing of the job even though the machine to visit is available. As shown in Fig. 3, the job $J3$ is held in the upstream buffers of machines B_2 despite the arrival of another job $J8$, the jobs $J5$ and $J2$ remained also in buffers B_3 and B_4 respectively. This highlights that the solution regroups the processing of jobs on the machines to avoid unproductive times.

The studied instance shows that with 40% of missing operations, the mathematical model provided an optimal solution reducing the operators cumulative presence time from 1346s for the NS to 870s for OS (yielding a 54.72% reduction in unproductive times). Although, it can be seen that the optimization of the unproductive times may seem like it comes at the expense of other economic performance criteria such as the makespan which is slightly larger (623s for the OS and 606s for the NS). The reason why OS has greater makespan than the NS is that for this illustrative example, the scheduling is not constrained by a deadline.

Neverthless, with minor changes by adding additional constraints in the mathematical model, (forcing the makespan to be less than the makespan of NS, i.e. 606s), we can generate schedules without degrading the cumulative presence time of operators. The OS obtained by this model has a cumulative presence time of operators of 870s (which is the same value obtained without considering makespan constraint) and a makespan value of 598s. This new solution is illustrated in Fig. 4.

6. CONCLUSION & PERSPECTIVES

This work defines a specific retail order picking scheduling problem. The problem is widely spread in the e-commerce retailing field in which retailers seek to minimize the order pickers unproductive times.

We have proposed a linear mathematical formulation of the problem handled as a non-permutation flow-shop scheduling

problem in which missing operations, transportation times and limited buffer capacities have been considered simultaneously. The optimization approach presented by our model is illustrated by a small instance. The optimal solution obtained with the model formulation is compared to a solution generated by a naive approach. The model presented a significantly better performance enabling efficient scheduling with respect to the aforementioned constraints.
Our mathematical formulation models a larger set of problems. Indeed it can be used for problems with intermediate buffers and a first machine that each job has to visit (for instance, this first machine could be a labeling machine and the rest of machines could be production ones). Comprehensive benchmark tests, composed of larger instances, are needed to evaluate the performance of the model.

For future research, it would be interesting to develop other mathematical formulations based on the disjunctive and indexed time approaches to evaluate and validate the performance of the models. We think that the model is not capable of solving large instances of an industrial problem. One approach is to consider heuristic and meta-heuristic methods to solve large-scale instances.
As mentioned in Section 1, order picking for e-commerce is a time-critical process, especially when facing tight delivery schedules. Hence, scheduling under due-dates constraints is an interesting and challenging problem since it takes explicitly into consideration one of the clients requirements. Moreover, in zone picking, the picking process is done by workers. Thus, the processing time of the bins depends on the picker performance. It would be interesting if we study the concept of stochastic processing by taking into consideration the variability of picking times rather than assuming deterministic processing conditions.

REFERENCES

Avella, P., Boccia, M., Mannino, C., and Vasilyev, I. (2017). Time-indexed formulations for the runway scheduling problem. *Transportation Science*, 51(4), 1196–1209.

Boysen, N., de Koster, R., and Weidinger, F. (2019). Warehousing in the e-commerce era: A survey. *Eur. J. Oper. Res.*, 277(2), 396–411.

Brucker, P., Heitmann, S., and Hurink, J. (2003). Flow-shop problems with intermediate buffers. *OR Spectrum*, 25(4), 549–574.

Chung-Yee, L. and Zhi-Long, C. (2001). Machine scheduling with transportation considerations. *J. Scheduling*, 4(1), 3–24.

De Santis, R., Montanari, R., Vignali, G., and Bottani, E. (2018). An adapted ant colony optimization algorithm for the minimization of the travel distance of pickers in manual warehouses. *Eur. J. Oper. Res.*, 267(1), 120–137.

Fang, K., Uhan, N.A., Fu Zhao, F., and Sutherland, J.W. (2013). Flow shop scheduling with peak power consumption constraints. *Ann. Oper. Res.*, 206(1), 115–145.

Hong, S., Johnson, A.L., and Peters, B.A. (2012). Batch picking in narrow-aisle order picking systems with consideration for picker blocking. *Eur. J. Oper. Res.*, 221(3), 557–570.

Huang, M., Guo, Q., Liu, J., and Huang, X. (2018). Mixed model assembly line scheduling approach to order picking problem in online supermarkets. *Sustainability*, 10(11), 1–16.

Johnson, S.M. (1954). Optimal two-and three-stage production schedules with setup times included. *Naval research logistics quarterly*, 1(1), 61–68.

Lamoudan, T., El Khoukhi, F., El Hilali Alaoui, A., and Boukachour, J. (2012). Flow shop scheduling problem with transportation times, two-robots and inputs/outputs with limited capacity. *IJICR*, 3(2), 221–230.

Le-Duc, Tho, and de Koster (2005). Determining number of zones in a pick-and-pack orderpicking system. *ERIM Report Series Reference No. ERS-2005-029-LIS*.

Leisten, R. (1990). Flowshop sequencing problems with limited buffer storage. *Int. J. Prod. Res.*, 28(11), 2085–2100.

Liao, C.J., Liao, L.M., and Tseng, C.T. (2006). A performance evaluation of permutation vs. non-permutation schedules in a flowshop. *Int. J. Prod. Res.*, 44(20), 4297–4309.

Liao, L. and Huang, C. (2010). Tabu search for non-permutation flowshop scheduling problem with minimizing total tardiness. *Appl Math Comput*, 217(2), 557–567.

Moslehi, G. and Khorasanian, D. (2014). A hybrid variable neighborhood search algorithm for solving the limited-buffer permutation flow shop scheduling problem with the makespan criterion. *Comput Oper Res*, 52, 260–268.

Nagarajan, V. and Sviridenko, M. (2009). Tight bounds for permutation flow shop scheduling. *Math Oper Res*, 34(2), 417–427.

Petersen, C.G. (2002). Considerations in order picking zone configuration. *IJOPM*, 22(7), 793–805.

Potts, C.N., Shmoys, D., and Williamson, D.P. (1991). Permutation vs. non-permutation flow shop schedules. *Oper. Res. Lett.*, 10(5), 281–284.

Pugazhendhi, S., Thiagarajan, S., Ajendran, C., and Anantharaman, N. (2003). Performance enhancement by using non-permutation schedules in flowline-based manufacturing systems. *Comput. Ind. Eng.*, 44(1), 133–157.

Ramezanian, R., Mehrabad, S.M., and Rahmani, D. (2011). Flow shop scheduling problem with missing operations: Genetic algorithm and tabu search. *IJAOR*, 1(2), 21–30.

Rossi, A. and Lanzetta, M. (2013). Scheduling flow lines with buffers by ant colony digraph. *Expert Syst Appl*, 40(9), 3328–3340.

Rossit, D.A., Tohmé, F., and Frutos, M. (2018). The non-permutation flow-shop scheduling problem: A literature review. *Omega*, 77, 143–153.

Sadjadi, S.J., Aryanezhad, M.B., and Ziaee, M. (2008). The general flowshop scheduling problem: Mathematical models. *Res. J. Appl. Sci.*, 8(17), 3032–3037.

Scholz, A., Schubert, D., and Wäscher, G. (2017). Order picking with multiple pickers and due dates-simultaneous solution of order batching, batch assignment and sequencing, and picker routing problems. *Eur. J. Oper. Res.*, 263(2), 461–478.

Tandon, M., Cummings, P., and LeVan, M. (1991). Flowshop sequencing with non-permutation schedules. *Comput. Chem. Eng.*, 15(8), 601–607.

Vahedi-Nouri, B., Fattahi, P., and Ramezanian, R. (2013). Minimizing total flow time for the non-permutation flow shop scheduling problem with learning effects and availability constraints. *J. Manuf. Syst.*, 32(1), 167–173.

Ying-Chin, H. and Jian-Wei, L. (2017). Improving order-picking performance by converting a sequential zone-picking line into a zone-picking network. *Comput. Ind. Eng.*, 113, 241–255.