Scheduling loads injection during flows merging in a collector

B. Vacher[1,2], A. Jouglet[1], D. Nace[1], S. Pietrowicz[2] and M. Bouznif[2]

[1] Sorbonne Universités, Université de Technologie de Compiègne, CNRS, Heudiasyc UMR 7253, CS 60319, Compiègne cedex 60203, France
blandine.vacher,antoine.jouglet,dritan.nace@hds.utc.fr
[2] SAVOYE, Dijon 21000, France
blandine.vacher,stephane.pietrowicz@savoye.com; marwane.bouznif@a-sis.com

Keywords: scheduling, job shop, logistics.

1 Industrial context

These last years, the world of supply chain has been significantly impacted by the consumer society and by the extensive use of new technologies. The constant increase of orders to deal with is forcing logistics actors to be more reactive and competitive. SAVOYE is a company specialized in the automation of logistics warehouses, and is a manufacturer of equipments for order preparation. In this context, the company focus on optimizing its solutions. To complete an order, many operations are required (such as erecting a box, moving stored items on bins, weighting a box). Optimizing loads (boxes, bins, containers, etc.) travel time could significantly improve the warehouse performance.

In this paper, we consider the optimization of the injection of loads coming from different sources onto a single collector in aim to maximize the throughput and ensure the highest production rate possible. We define a flow as an ordered list of loads following the same path. We are interested in merging several flows into a single one. The system is made of several aisles materialized by conveyors arranged side by side which join the same collector. Each flow is carried by a conveyor and its loads are waiting in line to be injected onto the collector (see Figure 1).

The goal is to maximize the throughput of the final flow carried by the collector, conveying loads of the different flows coming from different aisles. To do that, we should be able to choose, for each aisle, the date when each load has to be injected onto the collector. In practice, a load can be injected at a given date t from a given conveyor i if there is no load in the junction between the conveyor i and the collector (see Figure 1) at time t. While the conveyor from the firt aisle can freely inject a load (since there is always no load in front of its exit), it is not the case at any time t for the conveyors downstream for which loads injected from conveyors upstream may occupy the place and do not allow these conveyors to inject loads at this precise time t.

Maximizing the throughput of the collector is equivalent to reducing the number of empty space on the collector while it runs at its highest mechanical speed capacity. Thus, we propose an approach computing the optimal dates of loads injection, based on which is built a final flow with as less as possible empty spaces. These injection dates are defined with respect to a given feasible final sequence in which the loads are waiting at the exit of the collector.

2 Problem definition

Let L be the set of the n loads to be injected onto the collector, each of them being identified by a unique identifier which corresponds to its position in the wished exit sequence

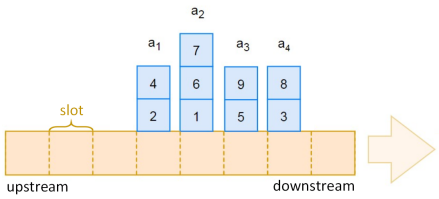

Fig. 1. Exemple of the studied systeme : instance A

σ. Let $A = \{a_1, \ldots, a_k\}$ be the set of the k aisles numbered from the most upstream to the most downstream. We assume that the system has full knowledge of the loads present in each aisles. Then, we denote by h_i the number of loads waiting in the aisle a_i. Next, for the purposes of notation, we define function $a_i : \mathbb{N} \to \mathbb{N}$ such that $a_i(j) = l \in L$ where l is the identifier of j^{th} load waiting in aisle a_i, i.e. its position in sequence σ. One can remark that we have $L = \{a_i(j), \forall i = 1 \ldots k, \forall j = 1 \ldots h_i\}$.

An instance is shown in Figure 1 with $k = 4$, $n = 9$, $A = \{a_1, a_2, a_3, a_4\}$, $L = \{1, 2, \ldots, 9\}$, $\sigma = (1, 2, 3, ..., 9)$. On this instance we notice that $h_1 = 2$ and $a_2(1) = 1$.

Finally, the collector can be divided into slots corresponding to the space occupied by a load (often larger than the physical space used, in order to take into account a safety space). To simplify, we will consider in this paper that aisles are equally distributed on consecutive slots along the collector, as shown in Figure 1. Moreover, the duration needed to run through the distance of one slot is taken as unit of time.

3 Optimally injecting loads onto the collector

We are looking to inject loads by respecting the given σ sequence so that the collector has the maximal throughput, fluid and continuous, compared to its mechanical capabilities. We recall that the maximal throughput is reached if loads can be injected onto the collector with no empty space between loads and without slowing down its speed. We will show below that this problem can be modeled as a job shop scheduling problem and can be solved using the algorithm describd below.

3.1 A job shop scheduling problem

Definition 1. *The sequence σ is said to be* feasible *if it verifies the precedence constraints induced by the mechanic configuration of the aisles: $\forall a_i \in A$, $\forall j, p \in \{1, \ldots, h_i\}$, $j < p$ we have $a_i(j) < a_i(p)$.*

For each feasible sequence σ, we are able to compute the optimal injection date for each load in such a way that there is no empty space between loads on the collector running at its maximal speed. For that, we model the problem as a *job shop* scheduling problem as follows.

There are n jobs $\{J_u, u \in L\}$ being associated with each load $\{u \in L\}$. Each job has an ordered list of operations to follow. An operation is a task to be processed on a special machine. There are k machines $\{M_1, ..., M_k\}$, each one being associated with an aisle. Reminder that aisles are numbered from upstream to downstream and are equally distributed on consecutive slots. Then it takes one unit of time for a load on the collector to pass from one aisle to the following one. Thus, a load injected onto the collector from aisle a_i will pass in front of each aisle a_p with $p \in \{i, i+1, \ldots, k\}$. This mechanism is represented by

the fact that each job J_u associated with a load u from aisle a_i consists of $k-i+1$ unitary operations $\{o_{u,i}, o_{u,i+1}, \ldots, o_{u,k}\}$. Operations on this ordered list have to be processed consecutively, without waiting, respectively by machines $\{M_i, M_{i+1}, \ldots, M_k\}$.

By construction, the injection date onto the collector of the load u from aisle a_i corresponds to the start-time of the first operation $o_{u,i}$ of the job J_u, while the start-times of the following operations (i.e. $\{o_{u,i+1}, \ldots, o_{u,k}\}$) represent the dates at which the same load is in front of each following aisles.

Note that each job is made of at least one operation on the last machine M_k (since each load passes in front of at least the last aisle a_k). The sequence of scheduled operations on machine M_k corresponds exactly to the order in which the associated loads will pass in front of the last aisle and their starting time to the time at which the loads pass in front of the last aisle. Therefore, if load u is before load v in the given final sequence, operation $o_{u,k}$ has to be scheduled before operation $o_{v,k}$. Moreover, enforcing the constraints that machine M_k has to process operations without idle time guarantees the fact that there is no empty space between loads on the collector, maximizing its throughput.

To conclude, maximizing the throughput of the collector is equivalent to schedule all operations on M_k without idle time in the order given by the associated loads on the final sequence.

3.2 A scheduling algorithm

Several methods to solve the Job Shop problem already exist, as in the review proposed by Jacek Blazevitcz *et. al.* (1995) and new approaches shown by J. F. Gonçalves *et. al.* (2005) or by P. Pongchairerks (2016). However, this scheduling problem can be solved by the following algorithm thanks to the specificities of our model.

To begin with, supposing the first load of the sequence comes from aisle a_i, the operations $\{o_{1,i}, o_{1,i+1}, \ldots, o_{1,k}\}$ associated with the job J_1 start at date t', $t'+1$, $\ldots$, $t'+k-i$.

Then, we schedule the operations of the job J_u corresponding to the next load u in the final sequence. This job will be scheduled first by dealing with the operation $o_{u,k}$ to be processed on machine M_k, just after (without idle time) the previous operation $o_{u-1,k}$ scheduled on it. Supposing that the operation $o_{u,k}$ starts at date t, the operation $o_{u,k-1}$ (if it exists) is scheduled on machine M_{k-1} at date $t-1$, and so on, until having all operations scheduled. This procedure is iteratively applied on the $n-1$ jobs.

Real injecting dates on the collector are deduced directly from the start time of the first operation of each associated job according to the real duration of a time slot (which depends on the speed of the collector).

Table 1. Scheduling of the instance A

Time	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
M_1	**2**		**4**								
M_2	**1**	2		4		**6**	**7**				
M_3		1	2		4	**5**	6	7		**9**	
M_4			1	2	**3**	4	5	6	7	**8**	9

The Table 1 shows the application of this algorithm on the instance given in Figure 1. The load identifier written in a box means that this load is passing in front of the aisle pointed out by the line index, on the time defined by the column index. The wished

final sequence is actually scheduled without empty space on the final machine M_4 and the injecting dates are deduced from their first occurrence (bolded in the table).

3.3 A formula for the injecting dates

Let i_1 be the aisle index containing the load 1, first load of σ. Moreover, we suppose that we can inject loads from date 0.

Proposition 1. *The first operation on M_k will start at the earliest date*
$t_0 = \max_{a_i(1), i=1...i_1} \{k - i - a_i(1) + 1\}$. *Then, the injecting date of load $u \in L$ is given by $T(u)$ such that: $\forall i = 1...k, \forall j = 1...h_i,\ T(a_i(j)) = t_0 + a_i(j) - 1 - (k - i)$.*

Proof. Thanks to the previous job shop scheduling problem, we can calculate t_0 the earliest date we can schedule the first operation on machine M_k (proof not given in this paper). given the identifier (so the wished place on σ) of the j^{th} load waiting on the aisle a_i

Reminder that $a_i(j)$ gives the position of j^{th} load on aisle a_i on the final sequence σ. We deduce that the last operation of the job associated with the load $a_i(j)$ starts at $t_0 + a_i(j) - 1$. Thus, we deduce that this load injecting date is $(k - i)$ slots of time before according to the aisles layout hypothesis.

4 Results and perspectives

We succeed in maximizing the collector throughput while running at its maximal mechanical speed capacity, totally occuping and sorting loads according to a wished final sequence. To do that, we found an algorithm which always give an uninterrupted flow whatever feasible sequence for one instant of the system. Thanks to this method, we were able to extract a formula to directly calculate the loads injection dates. Iterating the process at strategic dates, linked correctly, allows for an uninterrupted dynamic flow on the collector.

This method has been adapted for aisles dispatched randomly across the collector and we also have thought about the construction of a good final sequence (if not given). Those solutions led to a patent deposit.

References

J. Blazevitcz, W. Domschke, E. Pesch, 1995, "The job shop scheduling problem: Conventional and new solution techniques", *European Journal of Operational Research* 93 (1996) 1-33

J. F. Gonçalves, J. J. Mendes, and M. G. C. Resende, 2005, "A hybrid genetic algorithm for the job shop scheduling problem", *European Journal of Operational Research*, vol. 167, no. 1, pp. 77–95.

P. Pongchairerks, 2016, "Efficient local search algorithms for job-shop scheduling problems", *International Journal of Mathematics in Operational Research*, vol. 9, no. 2, pp. 258–277.