This is an Accepted Manuscript of an article published by Taylor & Francis in Engineering Management Journal on Jan 24, 2019

available online: https://doi.org/10.1080/10429247.2018.1522221

Lot Streaming Flow Shop with a Heterogeneous Machine

Augusto Ferraro
Departamento de Ingeniería, Universidad Nacional del Sur, Av. Alem 1253, Bahía Blanca (8000), Argentina
augusto.ferraro@uns.edu.ar
Daniel Rossit*
Departamento de Ingeniería, Universidad Nacional del Sur, Av. Alem 1253, Bahía Blanca (8000), Argentina
INMABB-UNS-CONICET, Av. Alem 1253, Bahía Blanca (8000), Argentina
daniel.rossit@uns.edu.ar
Adrián Toncovich
Departamento de Ingeniería, Universidad Nacional del Sur, Av. Alem 1253, Bahía Blanca (8000), Argentina
atoncovi@uns.edu.ar
Mariano Frutos
Departamento de Ingeniería, Universidad Nacional del Sur, Av. Alem 1253, Bahía Blanca (8000), Argentina
IIESS UNS-CONICET, San Andrés 800, Bahía Blanca (8000), Argentina
mfrutos@uns.edu.ar

* Corresponding author. Tel.: +54 (0291) 4595101 - Ext.: 3245. Fax: +54 (0291) 4595119
E-mail: daniel.rossit@uns.edu.ar (D. Rossit)

Abstract:

It is possible to obtain greater productivity of a production system by overlapping the operations required to process a manufacturing order. This methodology, known as lot streaming, requires to divide the production order (lot) into small sublots. In this article, we study production systems that include machines that operate in batch mode (processing a group of units at the same time) and single processing machines (processing one unit at a time) arranged in a flow shop configuration, i.e., all jobs must go through the same production stages in the same order. The

obtained results show that addressing the problem with consistent sublots (a common sublot size used for the whole process) is inefficient. On the other hand, addressing the problem considering the sizing of sublots for each machine (variable sublots), greatly improves the quality of the solution but is very costly computationally (limiting the size of the problem that can be solved). Therefore, a decomposition procedure is proposed on the decision of sublots sizing. This procedure greatly improves the solution obtained using consistent sublots and does so at a much lower computational cost than the variable sublots approach.

Keywords: Flow Shop, Lot Streaming, Makespan, Sequence Dependent Setups, Decomposition

Nowadays, it is imperative for manufacturing companies to run efficiently in order to remain competitive in the global market. The emergence of new technologies, continuous improvement in product quality, and changing customer requirements have all motivated the pursuit of shorter production cycles, which demand effective production plans that minimize work in process and cycle time and fulfill customer demands. Due to the nature of batch production, the use of appropriately sized lots in shop floor production planning is vital to achieving those objectives. Lot streaming is a technique that divides a given production lot into smaller sublots to allow them to be processed in an overlapping manner in a multi-stage production system (Biskup & Feldmann, 2006). This technique can effectively influence the flow of a batch (or lot) of jobs over the machines by determining the appropriate size of the sublots (also called transfer lots) (Sarin & Jaiprakash, 2007). Processing different sublots of the same job in an overlapping fashion during different production stages improves the job flow through the system. While traditional scheduling problems assume that jobs or lot sizes are fixed, lot streaming problems can be considered as sequencing problems where the sizes of processing sublots are

decision variables.

Lot streaming improves the power of local decision making at the shop floor control level, and production may be accelerated without requiring additional investment in new machines. Furthermore, this technique can be implemented and quickly adjusted to current job requirements. This technique has been implemented by Toyota and Dell, among others (Defersha & Chen, 2010; Cheng, Mukherjee, & Sarin, 2013), and has been widely studied and researched in various production environments.

While the benefits of using lot streaming techniques are well known, the literature frequently limits its use to consistent sublots over all the production process (Cheng et al., 2013). This means that the sublots keep a fixed size for all the machines of the production system. This additional constraint can limit the impacts of the lot streaming technique as the sublot sizing technique does not allow enough flexibility to appropriately size the sublot to each machine within the production system. To overcome this limitation, it is possible to use variables sublots where the sizing of the sublots is decided for each individual machine allowing for reconfiguration of the sublots. Biskup & Feldmann (2006) first acknowledged the negative impact of restricting the solution to consistent sublots and found that when the number of sublots is small, the impact could be higher. Subsequently, in Feldmann & Biskup (2008), the authors presented additional evidence of variable sublots improving the production makespan. However, this improvement in system's performance brings a considerable increase in the difficulty of solving the scheduling problem since the problem now requires the size of the sublots to be determined (Rossit, Tohmé, Frutos, Bard, & Broz, 2016). The variable sublot approach, which modifies the size of the sublot according to the production stage, is of vital importance for production systems that have both batch machines (which process several product units at the same time) and unit processing machines (which process one single

product at a time). In these cases, batch machines can process more than one unit at the same time while observing the maximum capacity of the machine, and the processing time is independent of the batch quantity processed. Examples of such production systems are millwork workshops where a press acts over hot plates with pressure to glue two or more parts together. This machine has a limited capacity, i.e., an upper bound on the sublot size that has to be processed, but the processing time is independent of its size. Another similar example can be found in the metallurgical industry where ovens used to perform heat treatments require a certain amount of time that is independent of the sublot size to be processed. Other industrial examples of heterogeneous (batch/single unit) processes are multi-layer ceramic capacitor manufacturing process, manufacturing of integrated circuits, and numerically controlled routers for cutting metal sheets or printed circuit boards (Koh, Koo, Kim, & Hur, 2005).

In heterogeneous systems, generating consistent sublots for the entire production process may decrease overall efficiency. Furthermore, such systems include machines that usually involve setup times, which increases the potential inefficiencies (Biskup & Feldmann, 2006). Operations management in heterogeneous production systems is usually not a straightforward task and generally requires the support of Information Technology (IT) tools for decision making such as those provided by the SIEMENS PREACTOR business unit or SAP's integrated management systems (Bożek & Wysocki, 2015). These tools generate production schedules for diverse production environments, handle the requirements of necessary materials and scheduling resources, and have a direct interface to link to Enterprise Resource Planning (ERP) systems. However, in production environments with batch production machines and other unit processing machines, the available commercial software does not solve the batch sizing problem and requires the scheduler to define the size of the sublots. This lack of support for the production management

function requires an adequate approach to address it. It is clear that incorrect sublot sizing would waste time generating bottlenecks by creating sublots that are too big for the batch machine, forcing previously scheduled jobs to wait for other jobs to arrive in order to be processed at the batch stage, or create small sublots that involve an excessive number of production runs in the batch machines such that jobs must wait until the batch machine is freed. The complexity of the sublot sizing decision increases as the production process involves batches of multiple products and the machine setups depend on the product previously processed. Sequence-dependent setup times impact production decisions as they involve non-productive operations that have to be performed on machines in addition to the job's processing time. These may include but are not limited to cleaning and fixing activities and loading of parts to machines (Ruiz & Maroto, 2006; Pan & Ruiz, 2012; Ciavotta, Minella, & Ruiz, 2013). Production in which setups depend on the sequence is more common than production in which they do not since the latter is a particular case of the sequence dependent one. This feature extends the scope of our work to many real-world industrial cases where the setup of the machines depends on the product scheduled previously. Ultimately, sequence independent setup times are a special case of sequence dependent setup times.

That is why it is necessary to generate tools that support this decision process without disregarding the scheduling problem, and this is studied in this article for flow shop manufacturing environments. Flow shop production systems consist of a set of jobs and machines in which all jobs must be processed on all machines in the same order, i.e., all jobs go through machine 1 first, then machine 2 and so on successively.

Specifically, the flow shop scheduling problem that combines unit and batch processing machines is studied in this article. To approach this problem, lot streaming strategies are proposed, which allow dividing production lots into sublots. As was previously explained, the configuration of these

sublots may significantly influence the performance of the production schedule.

To analyze the efficiency of lot streaming strategies, the solutions obtained by the ideal approach (variable sublots) are compared with the results of consistent sublots. This analyzes the case in which the scheduler selects "simplistic" solutions and to measure the impact that such a strategy can have on the performance of the system. However, the computational effort required by the variable sublots approach constrains the analysis. For dealing with this matter, a new procedure is proposed which allows decomposing the production system into three parts. This new approach reduces the computational effort without significantly worsening the objective function. The proposed decomposition method is extensively tested.

Literature Review

One of the original works on this topic was contributed by Trietsch & Baker (1993) who proposed a scheme for classifying the different features of the lot streaming problem, focusing mainly on flow shop configurations. Flow shop configurations are relevant because they are appropriate to improve manufacturing productivity and, for that reason, they account for more than a quarter of real-world production facilities (Pan et al., 2011; Toncovich, Rossit, Frutos, & Rossit, 2019). The work of Trietsch & Baker (1993) generated multiple research efforts to develop methods to solve the lot streaming problem. The first meta-heuristics to address the problem were presented by Kumar, Bagchi, & Sriskandarajah (2000) and Yoon & Ventura (2002). Kumar, Bagchi, & Sriskandarajah (2000) implemented the first meta-heuristics to address production systems with lot streaming techniques, and Yoon & Ventura (2002) applied genetic algorithms to solve a flow shop scheduling problem with lot streaming. Since then, several new approaches and algorithms have been applied to lot streaming scheduling problems. Tseng & Liao (2008) applied a particle swarm

optimization algorithm to solve lot streaming scheduling problems, and Pan, Tasgetiren, Suganthan, & Chua (2011) similarly implemented a discrete artificial bee colony algorithm. Pan & Ruiz (2012) used an estimation of distribution algorithm for flow shop with lot streaming, and more recently, Han, Gong, Sun, & Pan (2014) addressed a multi- objective lot streaming flow shop with a modified Non-Dominated Sorting Genetic Algorithm II (NSGA-II) algorithm. More recently, Han, Gong, Jin, & Pan (2016) contribute a stochastic blocking multi-objective approach to solve the problem, and in Rossit, Tohmé, Frutos, Bard, & Broz (2016), lot streaming techniques are extended to address non-permutation flow shop scheduling problems. Zhang et al. (2017) proposed a mathematical model and an effective modified migrating birds optimization (EMBO) algorithm to solve the problem of hybrid flowshop hybridizing with lot streaming with the objective of minimizing the total flow time. Han, Gong, Jin, & Pan (2017) proposed an evolutionary multiobjective robust scheduling algorithm to solve the blocking lot-streaming flow shop (BLSFS) scheduling problem with machine breakdowns. Gong, Han, & Sun (2018) presented a hybrid multi-objective discrete artificial bee colony (HDABC) algorithm for the blocking lot-streaming flow shop (BLSFS) scheduling problem with two conflicting criteria: the makespan and the earliness time. Meng, Pan, Li, & Sang (2018) considered an integrated lot-streaming flow shop scheduling problem in which lot-splitting and job scheduling are needed to be optimized simultaneously. They present an improved migrating birds optimization (IMMBO) to minimize the maximum completion time or makespan.

Considering specifically contributions that address lot streaming with variable sublots, Martin (2009) was among the first to make significant advances by proposing a novel hybrid approach to optimize this problem through a combination of a Mixed Integer Programming (MIP) solver, CPLEX, and evolutionary algorithms. A similar method was proposed by Defersha & Chen (2010)

where a flow shop with setups and variables sublots is solved with an algorithm based on a MIP solver (CPLEX) and a hybrid genetic algorithm. In Defersha & Chen (2012), a hybrid flexible flow shop problem is tackled, and an improved version of the previous algorithm is proposed that can be run on a parallel computing platform.

Recently, there have been efforts to address production systems that combine batch production machines and unit processing machines from the lot streaming approach. In this regard, Li, Meng, Li, & Tian (2016) addressed a similar problem in a job shop scheduling environment (a configuration where each job has its own work route and not all jobs go through the same machines in the same order) while considering transfer times between different manufacturing cells. Their production process consisted of several unit processing machines and a single batch processing machine, which belongs to an intermediate step of the production process for all the products. The problem was solved by an Ant Colony Optimization (ACO) algorithm. In order to cope with the effects of the batch processing machine positioned in the middle of a set of unit production machines, the authors propose to minimize idle time of the batch processing machine. Sequence dependent setup times were not considered, and transfer times between machines within the same cell were considered negligible. This assumption represents a reasonable simplification since the transfers from one cell to another have a greater impact than the movement within the cell. However, scheduling problems that exclusively consider operations within a single cell or a flow shop system may be significantly impacted by transfer times. In Li, Li, Meng, & Tian (2015), the same problem is solved by using a hyperheuristic based in an ACO algorithm and dispatching rules. These authors studied the implementation of several strategies to both schedule the jobs in the unit machines and form the batches to be processed in the batch processing machine.

To the best of our knowledge, Li et al. (2015) and Li et al. (2016) are the only published works

approach scheduling problems combining unit and batch processing machines. There are substantial dissimilarities between the problems studied by these authors and the scheduling problem analyzed in this article. Firstly, while Li et al. (2015) and Li et al. (2016) study job shop configurations, this article deals with flow shop configurations. Although the flow shop scheduling problem is a special case of the job shop scheduling problem (since considering individual operations routes for each job does not necessarily prevent the routes from being the same), there are additional aspects that should be considered. For instance, generally speaking, flow shop systems process higher production volumes than job shop systems (Pinedo, 2002). Therefore, complete lots belonging to the same product or job are processed rather than isolated product units and, consequently the implementation of lot sizing strategies throughout the production system, and not only in the batch processing machine, may have a considerable influence on scheduling performance. In addition, prior work does not consider sequence dependent setup times, which are studied in this article.

Problem Definition and Methodologies Description

The problem under study in this article consists of a flow shop system where one of the machines is a batch processing type and the remainder of the machines are single product processors. This type of production scenario is commonly found in the millwork and metallurgical industries and wherever production systems are arranged in the form of a production line where a batch processing operation is required. The flow shop system includes a group of n jobs that must be processed by a set of m machines. The flow shop scheduling problem is a combinatorial optimization scheduling problem in the NP-Complete class (Garey, Johnson, & Sethi, 1976). All jobs require processing by all machines and in the same technological order, i.e., the first operation on the first machine, the second operation on the second machine and so on. In this case study, one

of the machines is a batch workstation, where the processing times are independent of the size of the sublot that must be processed. In the remainder of the article, this machine is called as the "limiting machine." The sequence of the jobs and size of the sublots must be determined to minimize the production makespan (completion time) of the schedule. The setup times are sequence dependent, and there are fixed sublot transfer times between machines to represent the problem in a realistic way.

Initially, we show that if the problem is addressed using the classic approach (consistent sublots, here referred to as the "simple method"), the final obtained solution may be considerably worse in terms of the makespan objective than if this problem is addressed by means of the variable sublots approach (referred to as the "ideal method"). Nonetheless, the ideal method leads to a significant and fast rise in the computing time, considerably limiting the size of the problem that can be solved. Therefore, a new approach (the "partitioned method") is proposed by which the sizing of the sublots is decomposed into stages. The first stage is comprised of the first machine up to the machine before the limiting machine, the second stage consists of just the limiting machine, and the third stage includes the remaining machines after the limiting machine. This segmentation is represented in Exhibit 1. The sublots sizes are defined for each stage, where the sublots of the following stage can be formed only by units that belong to a sublot that has already finished its processing in the previous stage. This aids in the tracking of the work orders and the evolution of the production plan.

Insert Exhibit 1 Production system and its stages.

As discussed in the Introduction, considering lot streaming strategies implies that the sublot size to be processed is a decision variable which the scheduler can control. Thus, the decision of the size of sublots is added to the classic scheduling problem related to selecting job sequence. That

being said, deciding the size of a given sublot implies, in a certain way, determines the time this sublot will require on each machine. Considering that the metrics used to assess schedule performance are generally related to time, makespan is used as a performance measure and the dependency between the sublot sizes and the performance of a given schedule is clear. Therefore, it is not possible to separate these decisions in a hierarchical way, where the job sequence or the sublot sizes is set and then the other set of variables are determined considering the solution of the first problem. Furthermore, it is not clear which decision should be made first. Consequently, to solve the problem optimally, it is necessary to adopt a thorough approach by handling both decisions simultaneously. By proceeding in this manner, a software like CPLEX can find the optimal solution to the problem committing the optimal job sequence and sublot sizing. The decomposition procedure presented in this article can be implemented with any other mixed integer linear programming (MILP) solver including those based on metaheuristic optimization.

Formulation of the Lot Streaming Models

In this section, the mathematical formulations of the three employed methods are developed based on the guidelines of Sarin & Jaiprakash (2007) and Feldmann & Biskup (2008). It was necessary to formulate the mathematical models since the problem has not yet been addressed in the literature. First, the ideal method is described in detail, and then the partitioned and simple methods are presented as reformulations of the ideal method.

Model 1: Ideal Method - M1

This method (M1) implements the pure variable sublots approach, which allows complete flexibility to size the sublots. In each machine of the production process, the sublots can be rearranged, and therefore, this method can provides the optimal solution.

Parameters and sets:

i: index for the sublots, $i = 1, 2, \ldots, l$. j: index for the jobs, $j = 1, 2, \ldots, n$.

k: index for machines, $k = 1, 2, \ldots, m$.

m^*: machine that has processing times which are independent from the size of the sublot (limiting machine), $m^* \neq 1$, m.

p_{jk}: unit processing time of the job j on machine k if $k \neq m^*$ and the sublot processing time of the job j on machine k if $k = m^*$.

U_j: number of units of job j to be processed.

$t_{jj'k}$: set up time of job j' on machine k after having processed job j, $j \neq j'$. t'_{jk}: set up time of job j on machine k if job j is scheduled first.

f_{jk}: fixed sublot transfer time of any sublot of job j from machine k to machine $k + 1$, $k < m$ (including m*).

T: processing capacity of machine m^*.

G: an arbitrarily large number.

Variables:

s_{ijk}: size of sublot i from job j on machine k (integer variable).

c_{ijk}: completion time of sublot i of job j on machine k (continuous variable).

c_{max}: total completion time (makespan) (continuous variable).

w_{ijk}: binary variable which takes the value 1 if sublot i from job j is larger than 0 on machine k and 0 otherwise.

$x_{i'ijk}$: binary variable which takes the value 1 if sublot i from job j starts its processing on machine k not before than the completion time of sublot i' from the same job on machine $k-1$ and 0 otherwise, $k \neq 1$.

qjj': binary variable which takes the value 1 if job j is scheduled immediately before than job j', $j \neq j'$.

Objective function:

$$\min z = c_{max} \tag{1}$$

Constraints:

$$c_{max} \geq c_{ljm};\ \forall j \tag{2}$$

$$\sum_{u=1}^{n} s_{ujk} = U_j;\ \forall (j,k) \tag{3}$$

$$c_{1jk} - p_{jk}s_{1jk} - \left[\sum_{u=1,u\neq j}^{n} q_{uj}t_{ujk} + \left(1 - \sum_{u=1,u\neq j}^{n} q_{uj}\right)t'_{jk}\right] + G\left(1 - x_{i'1jk}\right) \geq c_{i'j(k-1)} + f_{j(k-1)}w_{i'j(k-1)};\ \forall (i',j,k): k \notin \{1, m^*\} \tag{4}$$

$$c_{1jm^*} - p_{jm^*}w_{1j} - \left[\sum_{u=1,u\neq j}^{n} q_{uj}t_{ujm^*} + \left(1 - \sum_{u=1,u\neq j}^{n} q_{uj}\right)t'_{jm^*}\right] + G\left(1 - x_{i'1jm^*}\right) \geq c_{i'j(m^*-1)} + f_{j(k-1)}w_{i'j(k-1)};\ \forall (i',j) \tag{5}$$

$$c_{ijk} - p_{jk}s_{ijk} + G\left(1 - x_{i'ijk}\right) \geq c_{i'j(k-1)} + f_{j(k-1)}w_{i'j(k-1)};\ \forall (i,i',j,k): i \neq 1, k \notin \{1, m^*\} \tag{6}$$

$$c_{ijm^*} - p_{jm^*}w_{ij} + G\left(1 - x_{i'ijm^*}\right) \geq c_{i'j(m^*-1)} + f_{j(k-1)}w_{i'j(k-1)};\ \forall (i,i',j): i \neq 1 \tag{7}$$

$$\sum_{u=1}^{i'-1} s_{uj(k-1)} - \sum_{u=1}^{i} s_{ujk} + G.x_{i'ijk} \geq 0;\ \forall (i,i',j,k): k \neq 1 \tag{8}$$

$$c_{ljk} \leq c_{1j'k} - p_{j'k}.s_{1j'k} - t_{jj'k} + G\left(1 - q_{jj'}\right);\ \forall (j,j',k): j \neq j', k \neq m^* \tag{9}$$

$$c_{ljk} \leq c_{1j'k} - p_{j'k}.s_{1j'k} - t_{jj'k} + G\left(1 - q_{jj'}\right);\ \forall (j,j',k): j \neq j', k \neq m^* \tag{10}$$

$$s_{ijm^*} \leq T;\ \forall (i,j) \tag{11}$$

$$s_{ijm^*} \leq w_{ij}G;\ \forall(i,j) \tag{12}$$

$$c_{1j1} - p_{j1}s_{1j1} \geq t'_{j1};\forall j \tag{13}$$

$$c_{(i+1)j1} - p_{j1}s_{(i+1)j1} = c_{ij1};\ \forall(i,j): i \neq l \tag{14}$$

$$c_{(i+1)jk} - p_{jk}s_{(i+1)jk} \geq c_{ijk};\ \forall(i,j,k): i \neq l, k \notin \{1, m^*\} \tag{15}$$

$$c_{(i+1)jm^*} - p_{jm^*}w_{(i+1)j} \geq c_{ijm^*};\ \forall(i,j): i \neq l \tag{16}$$

The objective function in Equation (1) is used to minimize the production schedule finishing time (or makespan). Constraint (2) expresses that the makespan cannot be less than the finishing time of the last sublot of each job processed on the last machine. With Constraint (3), it is established that the sum of the sublot sizes of a given job in a given machine must be equal to the lot size of that particular product. Constraints (4) and (5) determine whether the starting time of the first sublot of a particular job occurs after the finishing time of another sublot from the same job in two consecutive machines. Constraints (6) and (7) are similar to the two previous ones, but the comparison of sublots times is made with the sublot that is not the first sublot of the job. Constraint (8) indicates that, for a given job, no sublot can contain units that belong to those sublots of the precedent machine that have not been processed yet. With Constraints (9) and (10), the impossibility of the existence of intermingling of sublots is imposed. Constraints (11) and (12) limit the sublot size to be processed on the "limiting machine" and determine the values of the binary variables w_{ijk}. Constraints (13), (14), (15) and (16) limit the possibility of processing more than a single sublot at a time on each machine.

Model 2: Partitioned method - M2

The partitioned method (M2) separates the decision of sizing the sublots to three stages instead of sizing sublots in every machine as in the previously discussed ideal method (M1). Each stage contains a set of machines as described in Exhibit 1. Within each stage or group of machines, the sublots can be reconfigured and re-sized. The sublots are reconfigured with units of products that belong to sublots that have already been processed in the previous stage. Compared to the ideal M1 method, the partitioned method (M2) explores smaller feasible regions, and its solution approximates the ideal method's solution. This model represents a new approach to solve the problem and constitutes one of the main contributions of this article. The accuracy of the M2 approximation is assessed in the experimental section of this article.

The key differences of the M2 model with respect the M1 model resides in the integer variables *s* and binary variables *x*. A new set *h* is necessary to represent the stages for the sublot sizing decisions where the set *h* is defined as {1, 2, 3}. Thus, variables *s* and *x* must be rewritten.

s_{ijh}: size of sublot *i* from job *j* to be processed on the group of machines *h* (*h* = 1, 2, 3).

$x_{i'ijh}$: binary variable which takes value 1 if sublot *i* from job *j* starts its processing on machine $h + m^* - 1$ after (or at the same time) that sublot *i'* of the same job finishes its processing on machine $h + m^* - 2$ and 0 otherwise, $h < 3$.

$$c_{1jm^*} - p_{jm^*}w_{1j} - \left[\sum_{u=1,u\neq j}^{n} q_{uj}t_{ujm^*} + \left(1 - \sum_{u=1,u\neq j}^{n} q_{uj}\right)t'_{jm^*}\right] + G\left(1 - x_{i'1j1}\right) \geq c_{i'j(m^*-1)} + f_{j(m^*-1)}w_{i'j1};\ \forall(i', j) \quad (17)$$

$$c_{1jm^*} - p_{jm^*}w_{1j} - \left[\sum_{u=1,u\neq j}^{n} q_{uj}t_{ujm^*} + \left(1 - \sum_{u=1,u\neq j}^{n} q_{uj}\right)t'_{jm^*}\right] + G\left(1 - x_{i'1j1}\right) \geq c_{i'j(m^*-1)} + f_{j(m^*-1)}w_{i'j1};\ \forall(i', j) \quad (18)$$

$$c_{ij(m^*+1)} - p_{j(m^*+1)}s_{ij3} + G\left(1 - x_{i'ij2}\right) \geq c_{i'jm^*} + f_{jm^*}w_{i'j2};\ \forall(i, i', j) \quad (19)$$

$$c_{ijm^*} - p_{jm^*}w_{ij} + G\left(1 - x_{i'ij1}\right) \geq c_{i'j(m^*-1)} + f_{j(m^*-1)}w_{i'j1};\ \forall(i, i', j) \quad (20)$$

$$\sum_{u=1}^{i'-1} s_{ujg} - \sum_{u=1}^{i} s_{uj(g+1)} + G.x_{i'ijg} \geq 0;\ \forall(i, i', j, g) \quad (21)$$

Model 3: Simple method - M3

The simple method (M3) considers consistent sublots, i.e. the size of the sublot does not change at any stage or machine. As a result, the feasible region explored is smaller than that M1 and M2 methods. In this case, the integer variable *s* is indexed only on the sublots and jobs' sets, and the binary variable *x* is removed.

Computational Experiments

The experimental design aims to evaluate the performance of the M2 model as a method of solving the problem. To this end, M2 is compared with M3 (the simple solution of using a single sublot size throughout the whole process) to evaluate whether the usage of the proposed decomposition method M2 is advantageous or not. We will also compare M2 with M1 (variable sublots for all stages of the process) that provides the ideal solution to the problem but at the highest computational cost. The intention is to evaluate different problem sizes (combinations of number of machines and jobs) to be able to draw comprehensive conclusions from the application of the three models (M1, M2, and M3). As a scheme to generate these problems, we use the well-known instances in the flow shop literature (Taillard, 1993), which defines multiple cases using a maximum of twenty machines understanding that an industrial flow shop system usually has a smaller number of machines. In this article, we use twenty as the maximum number of machines. Exhibit 2 shows the multiple problems and the cases under which it was possible to find a solution for each model.

Insert Exhibit 2 Instances tested by each model.

Flow shop problems are NP-hard for m > 3 (Garey et al. 1976), and the size of the feasible solution space is proportional to (n!). Due to this property, an accurate MIP solver cannot solve problems with a very large number of jobs. In this work, we solve instances of up to nine jobs (as shown in Exhibit 2), where it can be seen that six different sizes of machines were used (3, 5, 7, 9, 15, and 20) and four jobs (3, 5, 7, and 9), and nineteen distinct instances were tested for the M2 and M3 models, while M1 was able to solve thirteen of the nineteen instances (M1 could not solve the 15 and 20 machine instances).

Simultaneously, for each instance, multiple maximum quantities of sublots were evaluated (i.e., different sizes of the set i), using a maximum of 2, 3, 4 and 5 sublots per job. This experimentation makes sense because it is necessary to know how many sublots are necessary to obtain the greatest benefit from the overlapping of operations proposed by the lot streaming approach. On the one hand, excessive lot streaming implies a multiplication of the fixed transfer times (f_{jk}) as well as an increase in the computation time by increasing the number of variables related to each sublot. However, on the other hand, insufficient lot steaming limits the ability of overlapping operations, thus reducing the associated benefits.

In addition, the influence of the batch machine within the production system was evaluated, thus analyzing if the proposed partitioning procedure is dependent on the relative position of the batch machine within the set of machines. If the proposed procedure provides good results only when the batch machine is in an intermediate position within all stages of the flow shop, then this result must be studied. Nevertheless, if the M2 method achieves good results regardless of where the batch machine is located, the potential of the M2 partitioning strategy increases. First, this analysis was performed considering the batch machine positioned at the "beginning" of the system, that is,

within the first third of the set of machines. Then it was located at an “intermediate” position, and finally at the "end" position (within the last third of the set of machines). Exhibit 3 provides the detail of where the batch machine was placed.

The computer used to perform the experimental work is a desktop with an Intel Core i7 processor with 16 GB RAM memory and a 64-bit operating system. The solver software is CPLEX 12.6.0, and the M1, M2, and M3 models were written in Pyomo (Hart, Watson, & Woodruff, 2011; Hart, Laird, Watson, & Woodruff, 2012). For the data and model management, Solver Studio for Excel was used (Mason, 2013).

Insert Exhibit 3 Detailed position of the limiting machine for each instance.

Parameter Specifications

Regarding the parameter generation, the following ranges were considered, based on an adaptation of the benchmark data of Taillard (1993) to the problem at hand:

i) The number of units for each lot of each job, U_j, was taken from a uniform distribution over the interval [20,50].

ii) The unit processing times, p_{jk} if $k \neq m^*$ (limiting machine), were considered uniformly distributed over [1,10]. Note that the combination of parameters U_j and p_{jk} resemble Taillard’s (1993) job processing time feature, where no job division was considered.

iii) The sublot processing time on the limiting machine, p_{jk} if $k = m^*$ (limiting machine), varies uniformly between 100 and 300.

iv) The limiting machine capacity, T, was taken from a uniform distribution on the interval [25,50]. (Notice that the capacity of the limiting machine can be smaller than the lot of units to process, then, the limiting machine would, also, require sublot sizing. This case is more

general than considering the limiting machine with capacity enough for the complete lot.)

v) The set up times, t'_{jk} and $t_{jj'k}$, vary from 45 to 85. (The highest setup time was selected in order to have an approximate 6:1 ratio between the highest possible processing time of an entire job and the highest set up time. The lowest setup time was selected so that the sum of two setup times could never be lower than another)

vi) The fixed sublot transfer times, f_{jk}, are drawn from a uniform distribution on the interval [10,20] for each experiment.

To avoid possible random correlation of parameters, five distinct parameter scenarios were analyzed for each instance (varying combinations of the number of machines, jobs, and sublots). The range of variation of the parameters introduced before was selected to represent a wide variety of production systems. For instance, parameter p_{jk} represents the possibility of having products with processing times up to ten times greater than the processing times of other products that are produced within the same system.

Results and discussion

In this section, the results obtained from the experiments that were carried out to test the proposed models are presented and divided in subsections.

In the first subsection, General results, the overview of the results of the proposed lot streaming procedure is shown. Then, in Variations of makespan values with the number of machines and jobs, it is shown whether the variations of the makespan obtained by the methods depend on the size of the problem. In Computing time, we analyze the computational effort required by each method, and in the Impact of the number of machines and jobs on lot streaming implementations subsection, the detail of the impact of those factors on the makespan is discussed. Then, if the

number of sublots is a significant factor is presented in the subsection Number of sublots available in the production system. Finally, it is studied whether the position of the limiting machine influences the results or not.

General results

The values shown in Exhibit 4 are the average of the five problems solved for each instance with a maximum of four sublots for each job with the limiting machine placed in the intermediate position. In Exhibit 4, the cells filled with a hyphen (-) correspond to the instances for which no valid solution could be found (no feasible solution was found during an hour of computing or the gap of the incumbent solution was too high). From Exhibit 4, the makespans for the instances of three machines are the same for the three methods (M1, M2, and M3). Comparing M1 and M2, this result is logical since the solutions explored by both methods are the same (the three stages of M2 overlap with the three machines). However, the fact that makespans of M1 and M3 are identical is not as logical, but it can be guessed that the system is too simple and the impact of the limiting machine in sublot sizing is preponderant enough to influence the problem solution. For larger instances, it can be seen that M2 performs quite similar to M1. For the cases of five machines, M2 yields M1's makespan and clearly outperforms M3. This tendency is intensified for instances with more machines. Nevertheless, for 7 and 9 machines, the comparison between M1 and M2 in some instances cannot be made because, in the computation time limit established, M1 does not converge to a good solution in some cases and the resulting gap is too large. This is the reason why in some instances of 7 and in all the instances of 9 machines, the average makespan of M2 outperforms the average makespan of M1.

Insert Exhibit 4 Average makespan values in the case where the limiting machine is in the middle

position and the maximum number of sublots per job is 4.

Another result shown in Exhibit 4, which are meaningful because the best results were produced with four sublots, is that the computational effort required by M1 is corroborated as it cannot solve the instances with the larger number of machines (15 and 20 machines). Moreover, for some of the instances, M1 reported no optimal solution after an hour of computing; in these cases, M2 found a better solution. For the larger instances, M2 provides much better solutions than M3. For example, in the cases of 20 machines, the difference goes from 3335 for M2 and 3705 for M3 (3 jobs) to 4807 and 5651 (7 jobs). These discrepancies between methods seem to show that there is a dependency on the size of the instance.

Variation of makespan values with the number of machines and jobs

As previously mentioned, the makespan gaps among the methods depend on the size of the instance. To provide additional insight into this result, Exhibit 5 presents the relation between the makespan and the number of jobs for the case of 9 machines and 4 sublots, and Exhibit 6 presents the relation between the makespan and the number of machines for the case of 5 jobs and 4 sublots. These images show the dependency of the results on the proposed models as the number of jobs or the number of machines vary respectively.

Insert Exhibit 5 Makespan vs the number of jobs for a production system consisting of 9 machines and 4 sublots.

Insert Exhibit 6 Makespan vs the number of machines in a production system consisting of 5 jobs and 4 sublots.

From Exhibit 5, it can be seen that M2 is a competent approximation of M1 and that M3 provides worse solutions than M1. Regarding M2 and M3, the solutions are quite dissimilar with M2 much improved. In addition, it is perceived from the slopes of the three curves that the three methods keep a similar gap among them, which means that for a given number of machines, the number of jobs will not increase the gaps among M3 and the other two methods. From Exhibit 6, it is shown again that M2 approximates M1 in a good manner. In regards to M3, the gap with respect to M1 and M2 is slight for small instances but increases as the number of machines increases. The shape of the three curves is similar; however, the slope for M3 is steeper, indicating that the number of machines apparently impacts the gap between M3 and M2.

Computing time

To study computational performance of the three methods, we analyzed the computational time required by each approach to perform the experiments as shown in Exhibits 7 and 8 (computational time is presented on a logarithmic scale to provide a better perspective of the relationships). Exhibit 7 illustrates the computing time as a function of the number of jobs for the case of 9 machines and 4 sublots, and Exhibit 8 illustrates the change of computing time against the number of machines considering 3 jobs and 4 sublots. The asterisk (*) indicates that the time limit was reached in more than one of the five problems.

Insert Exhibit 7 Computing time as a function of the number of jobs to be sequenced, for the case of 9 machines and 4 sublots.

Insert Exhibit 8 Computing time as a function of the number of machines in the production system for the case of 3 jobs and 4 sublots.

Exhibit 7 shows that the M3 method requires less computing time to solve the problem, as was expected. The slope of M1 was anticipated as a large increase in computational time occurs when the number of jobs increases. A notable aspect of M2 is that its computation time values are lower than those of M1. Moreover, the M2 slope is flatter than the M1 slope, in particular for instances with more jobs. This implies that the presented decomposition method is an effective method for reducing the required computational effort. Exhibit 8 shows the difficulty MI faces when solving problems with a larger number of machines. As M2 and M3 computing times remain lower than ten seconds even for twenty machines, M1 reaches its time limit with nine machines. The result that the computation times for M2 and M3 are more or less constant is explained by the fact that only additional continuous decision variables are added as the number of machines increases (those corresponding to the completion times), but no additional integer decision variables are needed, thus maintaining the problem with approximately the same degree of complexity.

Impact of the number of machines and jobs on lot streaming implementations

In this subsection, we analyzed the makespan obtained by the three methods and varying numbers of jobs and machines. For this, the dependency between the makespan improvement of M1 and M2 with respect to M3 is studied when considering four sublots. In Exhibit 9, the makespan improvement is related to the number of machines where the dashed lines represent M2 improvements and the solid lines show M1 improvements. Similarly, Exhibit 10 shows the improvements linked to the number of jobs.

Insert Exhibit 9 M1 and M2 makespan improvement over M3, as a function of the number of machines, with 4 sublots per job.

Insert Exhibit 10 M1 and M2 makespan improvement over M3, as a function of the number of

jobs, with 4 sublots per job.

Exhibit 9 suggests that the greater number of machines, the larger makespan improvements of M1 and M2 over M3, considering that the transfer time between stations is equal to zero and that there is no variability. In fact, this tendency between the improvement of the solution and the number of machines is observed in every curve (different number of jobs). In addition, the slope of the curves indicates that the impact of fewer jobs is greater than that of more jobs, and it flattens as the number of machines increases. Exhibit 10 shows the improvement of the makespan against the number of jobs, and we observe that for the greater number of machines, the incorporation of more jobs increases the difference between M2 and M3 makespans. From Exhibits 9 and 10, it is observed that M1 and M2 provide very similar results, which may suggest that M2 is a good approach to solve this type of problem. It is observed that as the number of machines increases, the beneficial effects of lot streaming also increase as the greater number of machines, the greater the percentage of reductions in makespan. This can be inferred from the fact that an increase in workflow due to the lot streaming strategy generates benefits, which are accumulated over a greater number of machines. However, these inferences cannot be considered conclusive and present an opportunity to future analysis.

Number of sublots available in the production system

The effect of the number of maximum sublots per job is analyzed and presented in Exhibit 11, which shows the evolution of the makespan against the number of maximum sublots in the cases of M2 and M3 with five machines and the limiting machine in the intermediate position.

Insert Exhibit 11 M2 and M3 makespan as a function of the number of maximum sublots per job in the case of 5 machines.

Exhibit 11 indicates that the makespan decreases as each lot is split into more sublots. However, this decrease is not consistent as it seems to be saturated with three sublots for M3 and with five for M2. Furthermore, M2 has a larger makespan decrease than M3 under the same problem conditions, resulting a better makespan. Exhibit 12 compares the improvement obtained with M1 and M2 against the number of sublots available when considering a production system with five machines.

Insert Exhibit 12 Improvement of M1 and M2 as a function of the number of sublots available in the case of 5 machines.

Exhibit 12 again shows how similar are the results of M1 and M2 are with regards to the improvement achieved. Moreover, with more sublots available, the improvement is higher but in a decreasing manner. The improvement limit seems to be reached more quickly in cases with more jobs. However, the number of sublots has a larger impact on M2 than on M1, and the makespan improvement is lower for the cases with more jobs. For M1, it seems that the makespan improvement is indifferent to the number of jobs as all the improvements are around 1 to 2%.

Influence of the limiting machine position

As a final analysis, the position of the limiting machine and its impact on the performance of the proposed method M2 is investigated. Exhibit 13 presents the case of nine machines and four sublots and captures when the best and worst makespans are generated regarding the position of the limiting machine for the same scenario (column “Problem”) with a “B” for the best value, “W” for the worst, and “0” otherwise. Both M2 and M3 methods are assessed in this manner.

Insert Exhibit 13 Makespan values varying the limiting machine position.

For M2, the best makespan value is obtained in approximately half of the scenarios when the limiting machine is at the ending position. In addition, the ending position results in the worst result in two of the scenarios. It is worth noting that in M3, unlike M2, more than half of the results have the highest makespan value with the limiting machine being at the ending position. For the intermediate position of the limiting machine, M2 returned the best makespan in two cases. An examination of the relative improvements of M2 average makespan values with respect to the M3 values are shown in Exhibit 14.

Insert Exhibit 14 Relative improvements of M2 for different positions of the limiting machine.

In twelve of the seventeen instances studied, M2 improves with respect to M3 to the greatest extent when the limiting machine is at the ending position. Not only that, but when it is placed in the intermediate position, the best improvement is never achieved. From these results, we can infer that M2 obtains the largest differences with respect to M3 when the limiting machine is at the ending position, thus indicating that M3 is inefficient for this kind of system. When the limiting machine is at the beginning position, M3 is still inefficient but not as bad as when in the ending position. The percentage values of Exhibit 14 indicate that M2 is a suitable method since its improvements over M3 are in a consistent range.

Implications for Engineering Managers

The proposed methods are tools that support decision making in short term production planning. These tools complement other management tools such as ERP systems and production scheduling systems by providing a lot streaming sizing tool. Commercial software packages do not provide a tool for guiding lot size in a quantitative or rigorous manner as they simply adopt the size indicated by the scheduler. Therefore, our research goes beyond these applications by providing support for job

sequencing and lot sizing.

Production and operations engineering managers examine planning data to make operational decisions on the assignment of orders to a system of machines and equipment. Our research provides value to them as it provides new knowledge and guidance on lot streaming, such as number of sublots and number of machines. Once the limiting machines are detected, number of sublots is defined, and workload of the current orders are considered, the methods described in this article can be selectively implemented to obtain a compromise between the quality of the sequence and the computational effort involved to obtain it.

This study is particularly important in a production system where its capacity and performance are tightly related such as those mentioned in the introduction where the presence of a batch machine impacts decisively in the production capacity. Performance measures that consider delivery dates can be highly influenced by the manner in which capacity is managed. In addition, the ideas presented in this work can show if further efforts on improving schedules generated by more sophisticated lot streaming techniques will result in comparable reductions in the total completion time of the orders or not.

Our work presents a set of results that have, by themselves, a special significance for production managers as an inefficient makespan implies an inefficient use of the production system. In this sense, the first insight that a production manager obtains from our results is that system efficiency is very responsive to the lot / sublot sizing decisions adopted. It is not enough to make that decision strictly considering only the limiting machine since our results show that high losses of efficiency are experienced when addressing this decision problem in a rudimentary way (M3) , resulting in efficiency losses of more than 10% in terms of the makespan objective (Exhibit 6). This inefficiency implies that production plans require more time in the manufacturing system to produce the same

quantity. On the other hand, it is also important to consider the computational resources that may be required to solve this type of problem as shown by the computation times of M1. In this sense, the M2 approach has ample advantages when solving the same instances in much shorter computation times without significant losses in solution quality. Furthermore, larger instances can be solved with M2 when M1 could not provide a practical solution. Our results show that the decomposition strategy provides very good results in reasonable computation times. Therefore, dividing decisions into stages yields better results than those obtained when considering only the limiting machine.

Conclusions

This work was motivated by the gap in flow shop lot streaming and scheduling literature on addressing production systems with heterogeneous machines. In this article, the authors fill this gap by presenting and comparing three comprehensive mathematical models for flow shop lot streaming scheduling in production systems where one of the machines has processing times that are independent of the sublot size that must be processed. The first model (M1) uses the variable sublots approach in the entire system and yielded the best results but required excessive computational effort to solve the problem. Thus, a decomposition method (M2) is developed and compared to the classic consistent sublots approach found in the literature (M3).

One important conclusion of this article is that the consistent sublot approach (M3) considerably limits the quality of the final solution, resulting between 5% and 15% of inferior quality than the ideal solution to the problem (variable sublots or M1) in terms of the makespan objective. Furthermore, the larger the system in terms of the number of machines and jobs, the poorer the solution quality. Nevertheless, it is shown that the variable sublots approach demands a considerable computational effort, thus becoming intractable to solve.

The developed decomposition procedure (M2) results in an efficient and accurate technique for managing the production systems under consideration. It considerably reduces the computational effort and provides very good solutions, similar to the variable sublots approach (M1). Regarding the performance of the decomposition procedure, the M2 method obtains fast computational processing times while the number of machines increases. With regards to the number of jobs increasing, the computational time tends to increase but at a slower rate than the variable sublot (M1) method. In addition, the decomposition M2 method performed properly under different production settings when the location of the limiting machine was varied.

Moreover, several managerial implications are detailed that are related to production planning decision support for engineering managers where current IT tools are lacking. Future research will develop meta-heuristic algorithms to solve the proposed problem for larger instances and apply the partitioning procedure to other production systems.

Acknowledgment

Our thanks are due to the anonymous referees and the editors for their helpful comments and critical analysis that helped us to improve this article.

References

Biskup, D., & Feldmann, M. (2006). Lot streaming with variable sublots: an integer programming formulation. *Journal of the Operational Research Society*, *57*(3), 296–303.

Bożek, A., & Wysocki, M. (2015). Flexible job shop with continuous material flow. *International Journal of Production Research, 53*(4), 1273-1290.

Ciavotta, M., Minella, G., & Ruiz, R. (2013). Multi-objective sequence dependent setup times

permutation flowshop: A new algorithm and a comprehensive study. *European Journal of Operational Research, 227*(2), 301–313.

Cheng, M., Mukherjee, N. J., & Sarin, S. C. (2013). A review of lot streaming. *International Journal of Production Research, 51*(23–24), 7023–7046.

Feldmann, M., & Biskup, D. (2008). Lot streaming in a multiple product permutation flow shop with intermingling. *International Journal of Production Research, 46*(1), 197–216.

Defersha, F. M., & Chen, M. (2010). A hybrid genetic algorithm for flowshop lot streaming with setups and variable sublots. *International Journal of Production Research, 48*(6), 1705– 1726.

Defersha, F. M., & Chen, M. (2012). Mathematical model and parallel genetic algorithm for hybrid flexible flowshop lot streaming problem. *The International Journal of Advanced Manufacturing Technology, 62*(1–4), 249–265.

Garey, M. R., Johnson, D. S., & Sethi, R. (1976). The complexity of flowshop and jobshop scheduling. *Mathematics of Operations Research, 1* (2), 117–129.

Gong, D., Han, Y., & Sun, J. (2018). A novel hybrid multi-objective artificial bee colony algorithm for blocking lot-streaming flow shop scheduling problems. *Knowledge-Based Systems, 148,* 115–130.

Han, Y. Y., Gong, D. W., Sun, X. Y., & Pan, Q. K. (2014). An improved NSGA-II algorithm for multi-objective lot-streaming flow shop scheduling problem. *International Journal of Production Research, 52*(8), 2211–2231.

Han, Y., Gong, D., Jin, Y., & Pan, Q. K. (2016). Evolutionary multi-objective blocking lot-streaming flow shop scheduling with interval processing time. *Applied Soft Computing, 42,* 229–245.

Han, Y., Gong, D., Jin, Y., & Pan, Q. (2017). Evolutionary Multiobjective Blocking Lot-Streaming

Flow Shop Scheduling With Machine Breakdowns. *IEEE Transactions on Cybernetics,* Article in press. doi: 10.1109/TCYB.2017.2771213

Hart, W. E., Laird, C., Watson, J. P., & Woodruff, D. L. (2012). *Pyomo–optimization modeling in python* (Vol. 67). Springer Science & Business Media.

Hart, W. E., Watson, J. P., & Woodruff, D. L. (2011). Pyomo: modeling and solving mathematical programs in Python. *Mathematical Programming Computation*, *3*(3), 219–260.

Koh, S. G., Koo, P. H., Kim, D. C., & Hur, W. S. (2005). Scheduling a single batch processing machine with arbitrary job sizes and incompatible job families. *International Journal of Production Economics*, *98*(1), 81–96.

Kumar, S., Bagchi, T. P., & Sriskandarajah, C. (2000). Lot streaming and scheduling heuristics for m-machine no-wait flowshops. *Computers & Industrial Engineering*, *38*(1), 149–172.

Li, D., Li, M., Meng, X., & Tian, Y. (2015). A hyperheuristic approach for intercell scheduling with single processing machines and batch processing machines. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *45*(2), 315–325.

Li, D., Meng, X., Li, M., & Tian, Y. (2016). An ACO-based intercell scheduling approach for job shop cells with multiple single processing machines and one batch processing machine. *Journal of Intelligent Manufacturing*, *27*(2), 283–296.

Martin, C. H. (2009). A hybrid genetic algorithm/mathematical programming approach to the multi-family flowshop scheduling problem with lot streaming. *Omega*, *37*(1), 126–137.

Mason, A. J. (2013). SolverStudio: A New tool for better optimisation and simulation modelling in Excel. *INFORMS Transactions on Education*, *14*(1), 45–52.

Meng, T., Pan, Q.-K., Li, J.-Q., & Sang, H.-Y. (2018). An improved migrating birds optimization for an integrated lot-streaming flow shop scheduling problem. *Swarm and Evolutionary*

Computation, 38, 64–78.

Pan, Q. K., Tasgetiren, M. F., Suganthan, P. N., & Chua, T. J. (2011). A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem. *Information Sciences, 181*(12), 2455–2468.

Pan, Q. K., & Ruiz, R. (2012). An estimation of distribution algorithm for lot-streaming flow shop problems with setup times. *Omega, 40*(2), 166–180.

Pinedo, M. (2015). Scheduling. Springer.

Rossit, D., Tohmé, F., Frutos, M., Bard, J., & Broz, D. (2016). A non-permutation flowshop scheduling problem with lot streaming: A Mathematical model. *International Journal of Industrial Engineering Computations, 7*(3), 507–516.

Ruiz, R., & Maroto, C. (2006). A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility. *European Journal of Operational Research, 169*(3), 781–800.

Sarin, S. C., & Jaiprakash, P. (2007). Flow shop lot streaming. Springer Science & Business Media.

Taillard, E. (1993). Benchmarks for basic scheduling problems. *European journal of operational research, 64*(2), 278–285.

Toncovich, A. A., Rossit, D. A. Frutos, M., & Rossit D. G. (2019). Solving a multi-objective manufacturing cell scheduling problem with the consideration of warehouses using a simulated annealing based procedure. *International Journal of Industrial Engineering Computations, 10*(1), 1–16.

Trietsch, D., & Baker, K. R. (1993). Basic techniques for lot streaming. *Operations Research, 41*(6), 1065–1076.

Tseng, C. T., & Liao, C. J. (2008). A discrete particle swarm optimization for lot-streaming flowshop scheduling problem. *European Journal of Operational Research, 191*(2), 360–373.

Yoon, S. H., & Ventura, J. A. (2002). An application of genetic algorithms to lot-streaming flow shop scheduling. *IIE Transactions, 34*(9), 779–787.

Zhang, B., Pan, Q.-K., Gao, L., Zhang, X.-L., Sang, H.-Y., & Li, J.-Q. (2017). An effective modified migrating birds optimization for hybrid flowshop scheduling problem with lot streaming. *Applied Soft Computing, 52*, 14–27.

About the Authors

Augusto Ferraro got his undergraduate degree in Industrial Engineering in 2016 from the Department of Engineering of Universidad Nacional del Sur, in Bahía Blanca, Argentina. He performed research tasks oriented to study scheduling and production planning problems as a scholar for one year before graduating. He currently works as supply-chain analyst for the Dow Chemical Company.

Daniel Rossit is a postdoctoral fellow of CONICET (National Research Council of Argentina) and Teaching Assitant at the Department of Engineering of the Universidad Nacional del Sur, in Bahía Blanca, Argentina. He got an undergraduate degree in Industrial Engineering as well as a Ph.D. in Engineering from his home university. His research is centered on operations research, optimization and production.

Adrián Toncovich received his B.S. degree in Industrial Engineering and his M.S. degree in Engineering from the Universidad Nacional del Sur (UNS), Argentina, in 2001, and 2006, respectively. He is currently pursuing a Ph. D. degree in Engineering at the Universidad de Zaragoza, Spain. He is currently an Associate Professor of Production Planning and Control in the Department of Engineering at the UNS. His research interests include production planning and scheduling, supply-chain management, and multicriteria decision-making.

Mariano Frutos is an Associate Researcher of CONICET (National Research Council of Argentina) and Assistant Professor at the Department of Engineering of the Universidad Nacional del Sur, in Bahía Blanca, Argentina. He got an undergraduate degree in Industrial Engineering as well as a Master's and a Ph.D. in Engineering from his home university. His research is centered on scheduling problems in production and its treatment through metaheuristic methods.

Contact: Daniel Rossit, Área de Organización Industrial, Departamento de Ingeniería, Universidad Nacional del Sur, Av. Alem 1253, Bahía Blanca, Argentina; daniel. rossit@uns.edu.ar; danielrossit@hotmail.com