Batch Assignment of Parallel Machines in an Automotive Safety Glass Manufacturing Facility*

Asignación de lotes de máquinas paralelas de procesamiento en una instalación de fabricación de vidrio para seguridad automotriz

Submitted on: 30 August, 2018 | Accepted on: 10 February, 2020 | Published on: 04 December, 2020

Juan Felipe Mora
Pontifica Universidad Javeriana, Colombia
ORCID: 0000-0003-4016-6905

Rabie Nait-Abdallah
Pontifica Universidad Javeriana, Colombia
ORCID: 0000-0002-3212-015X

Alvaro J. Lozano
American Glass Products (AGP), Colombia
ORCID: 0000-0002-3856-4739

Carlos Montoya
Pontifica Universidad Javeriana, Colombia
ORCID: 0000-0002-6472-8485

Ricardo Otero-Caicedo
Pontifica Universidad Javeriana, Colombia
ORCID: 0000-0002-0358-8538

* Research article
[a] Corresponding author. E-mail: rnait-abdallah@javeriana.edu.co

DOI: https://doi.org/10.11144/Javeriana.iued24.bapm

How to cite this article:
J. F. Mora, R. Nait-Abdallah, A. J. Lozano, C. Montoya, and R. Otero-Caicedo, "Batch assignment of parallel machines in an automotive safety glass manufacturing facility," *Ing. Univ.*, vol. 24, 2020. https://doi.org/10.11144/Javeriana.iued24.bapm

Abstract

In this paper, optimization algorithms are used to solve a batch assignment problem of parallel processing furnaces in American Glass Products (AGP), a world leader company in the design and manufacturing of curved armored glass for transportation purposes. The problem consists in optimizing the bending process, which is considered to be the bottleneck workstation in the armored glasses production line in AGP. The objective is to maximize the efficiency of the furnaces and minimize the tardiness delivery of orders. Due to the complexity and constraints of the problem, we developed a proper dispatch algorithm and a Tabu search technique. The results are encouraging: the indicators of furnace usage hours and tardiness delivery improved by 32 % and 7 %, respectively compared to the decisions made in the plant during an actual production week. This work was the winner of an operation research challenge between around 100 graduate students. The challenge was organized by Javeriana University and AGP.

Keywords: Optimization, batch processing machines problem (BPM), armored glass, Tabu search.

Resumen

En este artículo, fueron usados algoritmos de optimización para la solución del problema de asignación de lotes para programación de máquinas paralelas en American Glass Products (AGP), una compañía líder mundial en el diseño y la manufactura de vidrio blindado curvo para propósitos de transporte. El problema consistió en la optimización del proceso de curvado, el cual es considerado la estación cuello de botella en la producción de vidrio blindado en AGP. El objetivo fue maximizar la eficiencia de los hornos y minimizar la tardanza de las órdenes entregadas. Debido a la complejidad y las restricciones de este problema se desarrollaron un algoritmo propio de despacho y una técnica de búsqueda Tabú. Los resultados son alentadores: los indicadores del uso de hornos en horas y la tardanza en la entrega mejoraron en un 32 % y un 7 %, respectivamente en comparación con las decisiones tomadas en la planta durante una semana de producción real. Este trabajo fue uno de los ganadores de un reto de investigación de operaciones entre cien estudiantes. Este reto fue organizado por la Pontificia Universidad Javeriana y AGP.

Palabras clave: optimización, construcción de lotes, vidrio blindado, búsqueda Tabú.

Introduction

On an industrial level, the problem of batch assignment is critical. It directly impacts the delay to deliver the finished product, affecting the costs and therefore, the profits. To address this problem, we cover the case of the world's leading company in the design and manufacturing of curved armored glass for transportation purposes. AGP group was founded 50 years ago and currently has manufacturing operations in Peru, Colombia, and Brazil. It exports specialized armored glass to more than 1000 customers in 50 countries around the world. The AGP factory in Colombia is the second oldest in the group, founded in 1989. It specializes in safety glasses with high levels of ballistic protection for cars.

The manufacturing process of armored glass goes through several phases: cutting of the glass, vitrification, bending, polishing, assembly, and lamination. The problem addressed in this document focuses on the bending phase, which is critical for AGP because of its complexity, required time, constrained capacity, and the number of furnaces available for processing.

Problem Description

The bending phase in AGP consists in passing the glass pieces through furnaces. The effect of temperature gives them the required curved shape. In this phase, batches (groups of pieces with similar characteristics), are put into the furnaces. It is in this process where the complexity of the existing AGP planning problem lies, since pieces must be assigned to batches and batches to furnaces at a specific date (as we will show later, we only assign batches to a date without considering the actual sequence of batches within the furnaces). The objective is to optimize the planning of batches, maximizing the efficiency of the furnace usage and minimizing the tardiness in delivery of orders.

Every day AGP receives orders from its customers; an order contains one or several glass pieces.

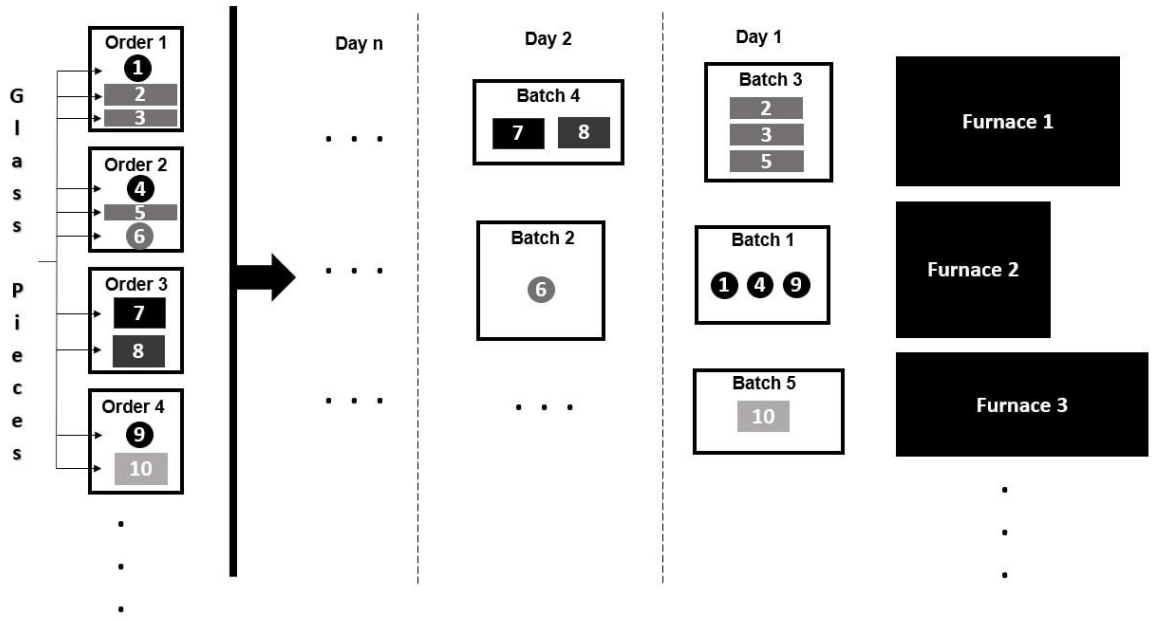
The planning process consists of:

1. Assigning the pieces to batches (a batch is a group of pieces that are going to be processed together in a furnace).
2. Assigning each batch created to a furnace and a processing date.

An order is a set of glass pieces that have to be delivered at the same time. Note that although the pieces of a given order have to be delivered together to the customer, they can be processed in different batches.

The figure 1 illustrates an example of the problem of programming AGP's daily production.

Figure 1. Graphic representation of the bending process in AGP



Source: Own elaboration

Figure 1 illustrates the batch assignment decision process. On the left side, a set of orders are received by AGP, each order consists of some number of pieces. In order to give a better explanation, we assume 10 pieces and 4 orders just as shown in figure 1. To assign pieces to batches, this case shows that pieces 1, 4, and 9 respect the constraints, and therefore are assigned to the same batch. Taking into account the features of the pieces in the batches, each piece is assigned to an appropriate furnace and to a processing day.

It is important to emphasize that in our problem we do not directly address the determination of batch sequences in the furnaces. Instead, we consider that each furnace is available during a certain amount of time per day (capacity expressed in hours). The problem is assumed to be feasible as long as the total processing time of the batches assigned to a furnace is less than its capacity. This assumption is structural as it simplifies the problem from a combination of a scheduling and a batch assignment problem to just a batch assignment problem.

This simplification is justified by the fact that the focus of the company was to address specifically the batch assignment part with all its features. Modeling all the scheduling constraints would be too complex and would have hindered the possibility of taking into account all the particularities that we wanted to address. However, in future works it would be interesting to expand the model to include more accurately the scheduling constraints.

Taking this into account, the general constraints that are considered for the development of the problem are:

- Each order has a delivery date to the customer.
- An order is ready when all its pieces have been processed.
- An order may be finished after the planned delivery date, but it would mean a tardiness that penalizes the objective function.

Specific characteristics of the glass pieces need to be considered (piece width, piece class, and piece ballistic level), as well as compatibility constraints between them for the creation of batches; that means, only pieces with similar processing characteristics could be assigned to the same batch.

On the other hand, the following are the constraints considered for arranging the batches:

- Pieces of different characteristics in a batch cannot be mixed. In particular, ballistic level and class are associated to each piece. Within a batch the maximum difference in the ballistic level of two pieces is 1 and the classes of the pieces must be the same.
- If a piece of the batch cannot enter into the furnace because of its characteristics, the batch cannot be assigned to that furnace.
- The width of a batch is equal to the sum of the widths of all pieces that compose it.
- A batch can be assigned to a furnace only if its width is less than or equal to, the width of the furnace. This is because the pieces of a batch are simultaneously processed in the furnace.
- There is no minimum or maximum limit on the number of batches that are created. However, no more batches can be created than the total quantity of pieces.

Regarding the furnaces, each one has specific characteristics, as well as different processing times and daily work capacity.

The optimization criteria identified for AGP are:

1. Minimizing usage hours of the furnaces (these hours are weighed according to an importance criterion defined by the company)
2. Minimizing tardiness delivery of orders

These criteria are strictly hierarchical. The company considers that criteria 1 is more important than criteria 2. As shown later in the document, this priority is translated in the objective function by setting a significative higher weight to criteria 1.

Based on this, we implemented a solution technique divided in two parts. The first one is a heuristic to find an initial solution. Secondly, the initial solution is complemented and improved with a Tabu Search metaheuristic. The detailed explanation of these methods will be further addressed in this document.

Literature Review

Production planning is one of the most important processes within a manufacturing company. The main goal is to determine how much to produce during a specific planning horizon in order to fulfill the demands for different time periods. Therefore, it is important to develop tools or methods that facilitate production planning decisions according to the specific features of the production process of a company.

The problem at hand can be classified as a Batch Processing Machines (BPMs) problem, with parallel non-identical machines and incompatible job families. Brucker, Mikhail and Shafransky [1] studied a two parallel, identical BPMs with unit processing times, unit set-up times, and a common deadline. Authors proved the NP-hard nature of such type of problems. Several studies address tardiness-based scheduling objectives on parallel batch processing machines. For example, Mönch, Balasubramanian and Fowler [2] proposed two different decomposition approaches based on the utilization of a genetic algorithm for minimizing total weighted tardiness on a parallel batch machines problem with incompatible job families and unequal ready times of the jobs. Chiang, Cheng and Fu [3] proposed a memetic algorithm for the identical parallel batch machines with incompatible job families and dynamic job arrival problem. Klemmt et al. [4] proposed a MIP model and a Variable Neighborhood Search (VNS) for minimizing total weighted tardiness on a parallel identical BPMs problem with different machine capacities, incompatible job families and unequal job ready times. Bilyk Mönch and Almeder [5] proposed a VNS and a Greedy Randomized Adaptive Search Procedure (GRASP) to minimize the total weighted tardiness on a parallel identical BPMs problem considering incompatible job families with precedence constraints. Gokhale and Mathirajan [6] developed certain heuristic algorithms for the parallel BPMs problem with unequal release times, incompatible job families, non-identical job sizes, heterogeneous batch processors, and allowance for job splitting with the objective of minimizing total weighted tardiness. Amouie [7] tackled the same problem with a heuristic based on column generation and a Differential Evolution (DE) metaheuristic. Moreover, Hulett, Damodaran and Amouie [8] addressed the non-identical parallel batch processing machines problem to minimize total weighted tardiness using particle swarm optimization. Finally, Lozano and Medaglia [9] proposed a two-phase hybridization of linear optimization methods and heuristics for a relaxed version of the problem at hand.

It is important to outline that, to the best of our knowledge there was no literature available that considers the BPMs problem with the non-identical and incompatible job families features together.

Mathematical Formulation

In this section we model our problem as a Mixed Integer Program. As far as we know and given the specific features of our problem, such an MIP formulation is not mentioned in the literature. Although the model will not be used to solve our problem (as shown later, the size of the problem prevents using an exact approach) an MIP formulation helps to understand in more details the decisions and constraints of the problem. The notation used in the formulation is presented below.

Sets

{i ϵ **P**}	Set of glass pieces
{j ϵ **O**}	Set of orders
{k ϵ F}	Set of furnaces
{l ϵ B}	Set of batches
{m ϵ L}	Set of ballistic levels
{n ϵ D}	Set of days, production processing horizon

Parameters

$p_{i,j}$	1, if piece *i* belongs to order *j* 0, otherwise
c_i	Class of piece *i*
$l_{i,m}$	1, if piece *i* has ballistic level *m* 0, otherwise
w_i	Width of piece *i*
$d_{j,n}$	1, if order *j* must be delivered on day *n*, due date 0, otherwise

$f_{i,k}$	1, if piece *i* can be processed in furnace *k* 0, otherwise
$t_{k,m}$	Processing time of furnace *k* to process a piece with ballistic level *m*
mw_k	Maximum available width of furnace *k*
$cf_{k,n}$	Capacity time of furnace *k* in the day *n*
$W_{tardiness}$	1
$W_{furnaces}$	50
V_k	Value assigned by the company to one hour of furnace *k*. The faster the furnace's processing time, the higher its value

$W_{tardiness}$ and $W_{furnaces}$ are the prioritization established by AGP for each of the parts that make up the objective function.

Decision Variables

$X_{i,l}$	1, if piece *i* is assigned to batch *l* 0, otherwise
$Y_{l,k,n}$	1, if batch *l* is assigned to furnace *k* the day *n* 0, otherwise
CP_i	Processing day of pieces *i*
CO_j	Delivery day of order *j*
T_j	Tardiness of order *j*
$Z_{i,r}$	1, if piece *i* and piece *r* are assigned in the same batch 0, otherwise
$PT_{l,k,n}$	Processing time of batch *l* in the furnace *k* the day *n*
U_k	Usage hours of the furnaces *k*

Objective Function

$$minimize\ Z = W_{furnaces} \sum_{\forall k \in F} U_k * V_k + W_{tardiness} \sum_{\forall j \in O} T_j \quad (1)$$

Subject to

$$\sum_{\forall l \in B} X_{i,j} = 1;\ \forall i \in P \quad (2)$$

$$\sum_{\forall j \in O} \sum_{\forall n \in D} Y_{l,k,n} \leq 1;\ \forall l \in B \quad (3)$$

$$\sum_{\forall i \in P} X_{i,j} \leq M \sum_{\forall k \in F} \sum_{\forall n \in D} Y_{l,k,n};\ \forall l \in B \quad (4)$$

$$\sum_{\forall i \in P} X_{i,j} \geq \sum_{\forall k \in F} \sum_{\forall n \in D} Y_{l,k,n};\ \forall l \in B \quad (5)$$

$$CP_i \geq \sum_{\forall k \in F} \sum_{\forall n \in D} n * Y_{l,k,n} - M(1 - X_{i,j});\ \forall i \in P, \forall l \in B \quad (6)$$

$$CP_i \leq \sum_{\forall k \in F} \sum_{\forall n \in D} n * Y_{l,k,n} + M(1 - X_{i,j});\ \forall i \in P, \forall l \in B \quad (7)$$

$$CO_j \geq CP_i * p_{i,j};\ \forall i \in P, \forall j \in O \quad (8)$$

$$T_j \geq CO_j - \sum_{\forall n \in D} n * d_{j,n};\ \forall j \in O \quad (9)$$

$$X_{i,l} + X_{r,l} \leq Z_{i,r} + 1; \forall i, r\ \in P, i \neq r, \forall l \in B \quad (10)$$

$$X_{i,l} + X_{r,l} \geq 2 * Z_{i,r} + 1; \forall i, r\ \in P, i \neq r, \forall l \in B \quad (11)$$

$$Z_{i,r}(C_i - C_r) = 0, \forall i, r\ \in P, i \neq r \quad (12)$$

$$Z_{i,r}\left(\left(\sum_{\forall m \in L} b_{i,m} * m\right) - \left(\sum_{\forall m \in L} b_{r,m} * m\right)\right) \leq 1, \forall i, r\ \in P, i \neq r \quad (13)$$

$$Z_{i,r}\left(\left(\sum_{\forall m\in L} b_{i,m} * m\right)-\left(\sum_{\forall m\in L} b_{r,m} * m\right)\right)\geq -1, \forall i,r\ \in P, i\neq r \tag{14}$$

$$\sum_{\forall i\in P} X_{i,j} * w_i \geq \sum_{\forall k\in F}\sum_{\forall n\in D} Y_{l,k,n} * mw_k\ ;\ \forall l\in B \tag{15}$$

$$PT_{l,k,q} \geq \sum_{\forall m\in L} t_{k,m} * b_{i,m} - \left(1-X_{i,j}\right) - \left(1-Y_{l,k,n}\right)M; \forall i\in P, \forall l\in B, \forall k\in F, \forall n\in D \tag{16}$$

$$\sum_{\forall l\in B} PT_{l,k,n} \leq cf_{k,n};\ \forall k\in F, \forall n\in D \tag{17}$$

$$\sum_{\forall l\in B} PT_{l,k,n} = U_k;\ \forall k\in F \tag{18}$$

$$X_{i,l}\ \in \{0,1\};\ \forall i\in P, \forall l\in B \tag{19}$$

$$Y_{l,k,n}\ \in \{0,1\};\ \forall l\in B, \forall k\in F, \forall n\in D \tag{20}$$

$$CP_i \geq 0;\ \forall i\in P \tag{21}$$

$$CO_j \geq 0;\ \forall j\in O \tag{22}$$

$$T_j \geq 0;\ \forall j\in O \tag{23}$$

$$Z_{i,r}\ \in \{0,1\};\ \forall i,r\in P, i\neq r \tag{24}$$

$$PT_{l,k,n} \geq 0;\ \forall l\in B, \forall k\in F, \forall n\in D \tag{25}$$

$$U_k \geq 0;\ \forall k\in F \tag{26}$$

The objective function (1) minimizes the usage hours of the furnaces and tardiness. Constraint set (2) ensures that each piece is assigned to one batch. Constraint set (3) ensures that a batch cannot be assigned to more than one furnace and more than one day. Constraint

sets (4) and (5) ensure that if a piece is assigned to a batch *l*, this batch must be assigned to a furnace *k* and a day *n*. If no pieces are assigned to a batch, this batch must not be assigned to either a furnace or a day. Constraint sets (6) and (7) ensures that the day on which the piece is processed is the same assigned day to the batch that it belongs to. *M* is a constant that is big enough to activate binary variables in the constraints (4), (6), and (7). Constraint set (8) determines the delivery day of order *j* which is the day when the last piece that belongs to the order is processed. Constraint set (9) determines the tardiness of order *j* based on its delivery day and due date. Constraint sets (10) and (11) ensure that if pieces *i* and *r* are assigned to the same batch $Z_{i,r} = 1$, otherwise $Z_{i,r} = 0$. Constraint set (12) ensures that the pieces of a batch must have the same class. Constraint sets (13) and (14) ensure that the maximum ballistic level difference between the pieces of a batch cannot be greater than 1. Constraint set (15) ensures that the batch width must be less or equal to the assigned furnace width. Constraint sets (16) determines the processing time of the batch l: $PT_{l,k,n}$. Which is the processing time of the piece of the batch that requires the longest time to process. Note that the variable $PT_{l,k,n}$ depends on n index because each furnace has different capacity times per day. If the batch *l* is assigned to a furnace *k* and a day *n*, $PT_{l,k,n} > 0$. If it is not, $PT_{l,k,n} = 0$. Constraint set (17) ensures that furnaces capacities are respected. Constraint set (18) determines the furnace use adding up all processing times of all the batches. Finally, the set of constraints from (19) to (26) specify the nature of the decision variables.

This formulation leads to a very large model with more than 450,000 variable and 100 million constraints (considering our case of 591 pieces, 152 orders, 4 furnaces, 160 batches in average, 7 ballistic levels, and 7 days). Even if we had reduced the size of the problem to only consider 100 pieces, 25 orders and batches, a planning horizon of just one day, and only one furnace, we would still end up with 12,000 variable and over 1 million constraints. Hence, exact methods are not an option to resolve the problem and heuristic approaches must be considered instead.

Methods

We developed two methods. The first one was based on dispatch rules, which find a feasible solution and improve current AGP results. The second method was a metaheuristic one that improved the distribution of pieces in the batches and found a better objective function.

Method No.1: Dispatch Algorithm

The first method consists of a fast heuristic based on dispatch rules to provide an initial solution to the Tabu Search algorithm.

The first heuristic is based on three criteria

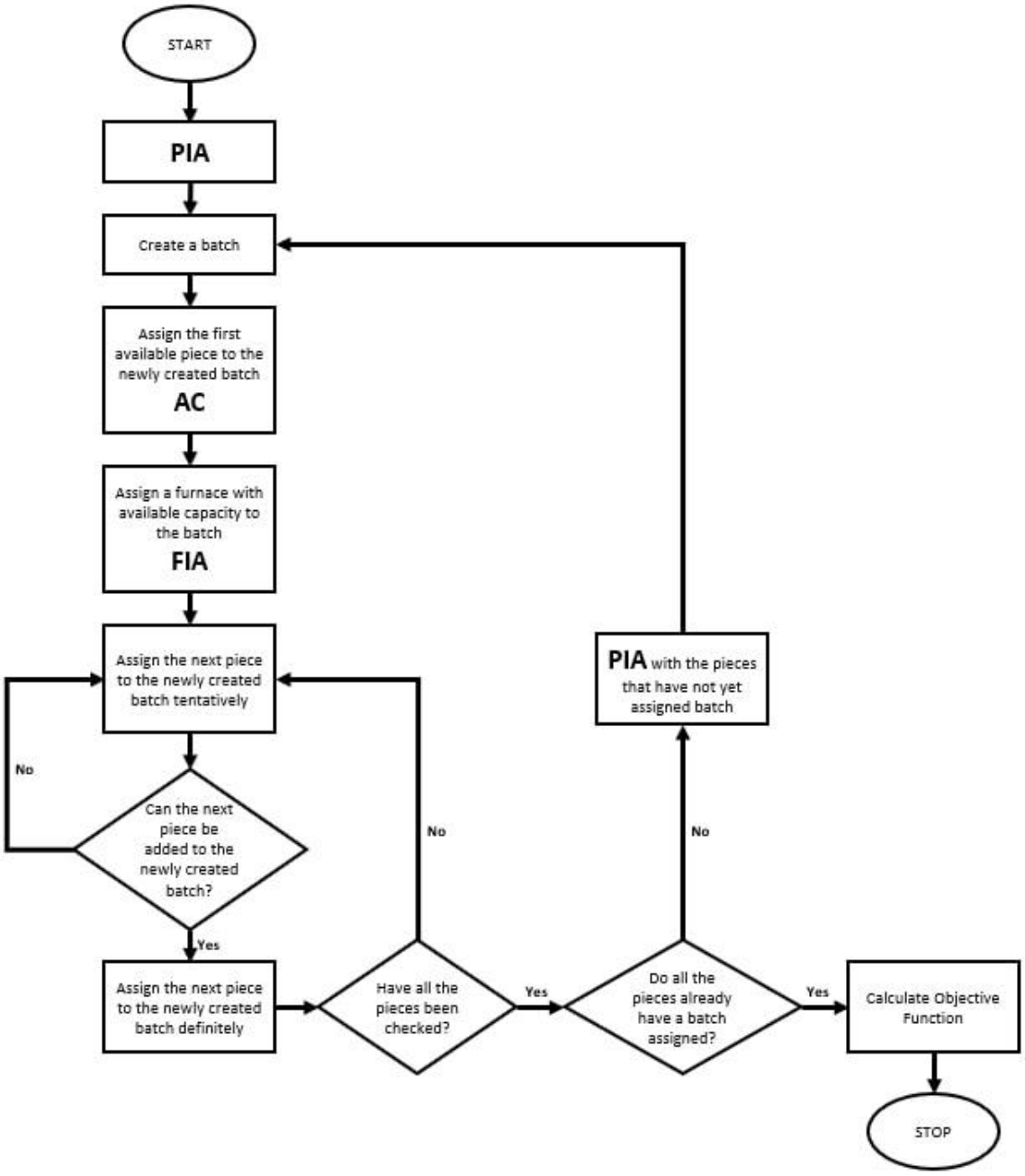
1. A criterion to sort the pieces before creating the batches (PIA).
2. A criterion to check the possibility of assigning pieces to a batch (AC).
3. A criterion to sort the furnaces in order to assign the batches (FIA).

Figures 2 and 3 show the pseudocode and flow diagram of the heuristic.

First, the pieces are sorted according to a criterion that we called PIA (Piece Information Arrangement which is actually an LPT [longest processing time] dispatching rule). In our implementation we used decreasing processing times for this criterion (as we will describe later, we also tested other criteria such as due date or width but with worse results). Once pieces are sorted according to the PIA criterion, batches are created and assigned to a furnace (the furnace to which the batch is assigned later is determined by the criterion FIA). The first piece is assigned to a newly created batch. The following pieces are added to this batch as long as the AC (Allocation Criterion) is met. AC criterion basically makes sure that pieces within a batch are all compatible and that the batch can be processed in the furnace. When no additional piece can be added to the current batch, a new batch is created, and the same process is carried on again.

The batches are assigned to the furnaces according to the third criterion FIA (Furnace Information Arrangement). It reflects the fact that furnaces have different processing times and different capacities. This criterion sets a priority to the fastest furnaces on average (based on the company estimation) and with the highest capacity.

Figure 2. Flow diagram of the heuristic



Source: Own elaboration

Figure 3. Pseudocode of the heuristic

Algorithm 1: Assignment method Pseudo-code

```
1:  assignment_pieces_to_batches ():
2:  Sort (arrangement of pieces according to PIA)
    Sort (arrangement of furnaces according to FIA)
3:  for l = 1 to number_batches
4:      for i = 1 to number_pieces
5:          if batch(l) is not assigned
6:              if piece(i) is not assigned to a batch
7:                  piece(i) is assigned to the batch(l)
8:                  call subroutine assignment_batches_to_furnaces (batch(l))
9:              end if
10:         else
11:             if piece(i) accomplishes the requirements of the batch(l)
12:                 piece(i) is assigned to the batch(l)
13:             end if
14:         end if
15:     end for
16:     If all pieces are assigned exit for
17: end for
18: end assignment_pieces_to_batches
19:     Subroutine assignment_batches_to_furnaces (batch(l)):
20: for k=1 to number_furnances
21:     for d=1 to number_days
22:         if batch(l) can be assigned to the furnace(k)
23:             if furnace(k) on the day(q) is enough capacity to process all the pieces of the
                batch(l)
24:                 batch(l) is assigned to the furnace(k) on the day(q)
25:                 end subroutine assignment_batches_to_furnaces
26:             end if
27:         end if
28:     end for
29: end assignment_batches_to_furnaces
```

Source: Own elaboration

Algorithm 1 shows a pseudo-code of the assignment algorithm. Line 2 organizes the pieces according to PIA and the furnaces according to FIA. Line 3 and 4 define two loops. The first one goes through the batches and the other one through the pieces. Lines 5 through 14 verify if batch *l* was not assigned to a furnace yet. If batch *l* is not assigned, line 6 verifies if piece *i* was not assigned to any batch, and if it is not, it is assigned to batch *l* (line 7). The subroutine is called in line 8. It will be explained below. If batch *l* was already assigned (line 10) then it is necessary to check if piece *i* could be assigned, according to all the explained constraints above, to batch *l* (lines 11 and 12).

The assignment of a batch is described in lines 19 through 29. The input parameter is batch *l*. Line 20 and 21 define two loops for the number_furnaces and number_days iterations,

respectively. The first one goes through the furnaces and the other one through the production planning horizon. Line 22 verifies if batch *l* could be assigned to furnace *k*. This depends on the parameter ($f_{i,k}$) of the piece that has already been assigned to batch *l*. Line 23 verifies if furnace k on day n has enough capacity to process the entire batch *l*. If it is possible, line 24 assigns batch *l* to furnace *k* on day *n*.

For a better understanding, an example is given with numerical parameters based on the one described in figure 1. Table 1 shows the sets that were defined in the mathematical model, tables 2 and 3 show the parameters of the pieces and furnaces, respectively. These data have been modified and do not correspond to the real data of AGP.

Table 1. Sets Initialization

P	{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
O	{1, 2, 3, 4}
F	{1, 2, 3}
B	{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
L	{1, 4, 7}
D	{7/9/17, 8/9/17}

Source: Own elaboration

Table 2. Piece parameters

Piece	Order	Class	Ballistic Level	Width	Due Date	Furnace 1	Furnace 2	Furnace 3
1	1	1	7	0.5	8/9/17	1	1	1
2	1	2	4	0.5	8/9/17	1	0	1
3	1	2	4	0.8	8/9/17	1	0	1
4	2	1	7	0.7	6/9/17	1	1	1
5	2	2	4	0.7	6/9/17	1	0	1
6	2	1	7	1.5	6/9/17	1	1	1
7	3	2	1	1	7/9/17	1	0	1
8	3	2	1	1	7/9/17	1	0	1
9	4	1	7	1	4/9/17	1	1	1
10	4	2	1	1	4/9/17	1	0	1

Source: Own elaboration

Table 3. (a), (b), (c): Furnace parameters

(a) Processing time of furnace *k* to process a piece with ballistic level *m*

Ballistic Level	Furnace 1	Furnace 2	Furnace 3
1	8	0	9
4	3	0	5
7	5	7	5

(b) Maximum available width of furnace *k*

Furnaces	Max Width
1	2.3
2	2.2
3	1.2

(c) Capacity time of furnace *k* on day *n*

Day	Furnace 1	Furnace 2	Furnace 3
1	5	12	9
2	8	10	0

Source: Own elaboration

First (line 2) the algorithm organizes the pieces taking into account PIA criterion. In this case, all the pieces were organized in descending order according to piece processing time (which is also proportional to the ballistic level). The obtained arrangement was {1, 4, 6, 9, 5, 3, 2, 8, 7, 10}. Line 3 organizes the furnaces according to FIA criterion. In this example, all the furnaces were organized by the capacity of each one. The obtained arrangement was {2, 1, 3}.

In the first iteration, the piece that could be assigned to batch 1 was piece 1. As this piece could be processed in any furnace and there was still capacity available in all of them, batch 1 had to be assigned to furnace 2 on day 1, according to FIA. The next iteration showed that piece 4 could be assigned to batch 1 because it fulfilled all constraints (width, ballistic level, and class). On the other hand, the next piece, number 6, could not be assigned to this batch because the sum of all the widths of pieces 1, 4, and 6, was higher than the maximum width in furnace 2. Checking the rest of the pieces, piece 9 was the last one that could be added to batch 1.

In the second iteration of the *l* index loop the first piece available to be assigned to batch 2 was piece 6. Even though there was enough space to add other pieces, none of the other pieces (number 2, 3, 5, 7, 8, 10) fulfilled the assignment constraints. As this piece 6 could be processed in any furnace and there was still capacity available in the furnaces, batch 2 had to be assigned to the furnace 2 on day 2, because there was not enough capacity on day 1 to process the batch. If furnace 2 had not had sufficient capacity to process, the batch would have been assigned to furnace 1 (furnace 1 is the second-best choice according to the explained organization above). The same logic is applied until all pieces are assigned.

Table 4 shows the results of this example.

Table 4. Results example

Batch	Pieces	Class	Ballistic Level	Total Width	Furnace	Day	Proc. Time	Orders	Tardiness
1	1, 4, 9	1	7	2.2	2	7/9/17	7	1	0
2	6	1	7	1.5	2	8/9/17	7	2	2
3	2, 3, 5	2	4	2	1	7/9/17	3	3	1
4	7, 8	2	1	2	1	8/9/17	8	4	3
5	10	2	1	1	3	7/9/17	9		

Source: Own elaboration

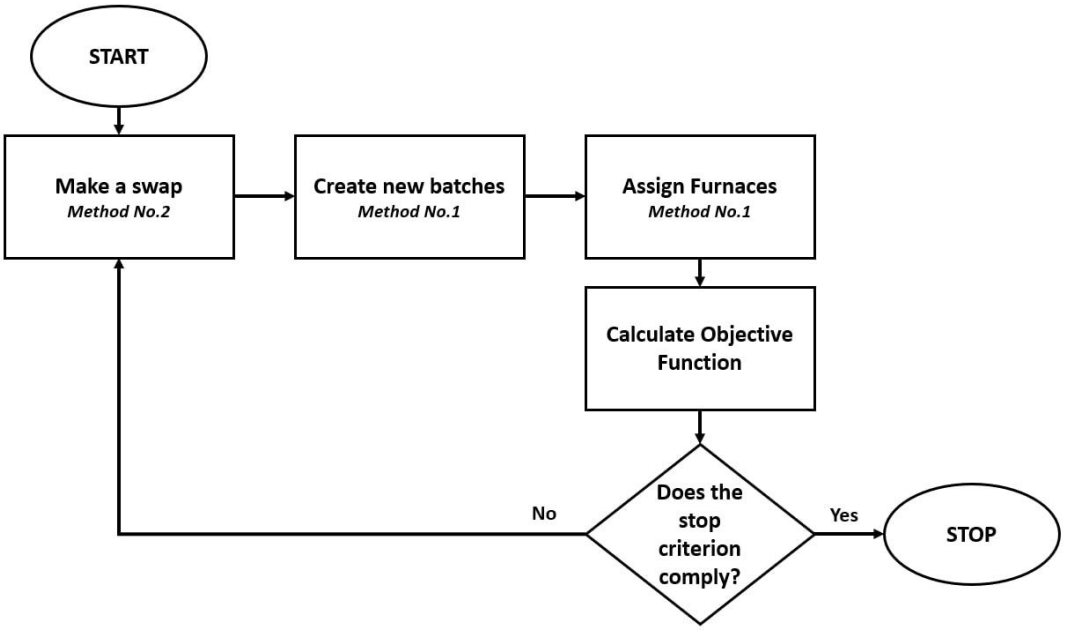
In this example the objective function is 1706. Tardiness of orders corresponds to 6 and usage hours of the furnaces corresponds to 34. In this case we are assuming the same values of the constants $W_{furnaces}$ and $W_{tardiness}$ explained above.

Method No. 2: Tabu Search

Tabu Search is a metaheuristic that guides a local heuristic search procedure to explore the solution space beyond local optimality [7].

Our method is a mix between a classical Tabu Search and our heuristic algorithm presented above. In each iteration, Tabu decides the order of pieces that is then used as an input to Method 1 (it overrides the order given by the PIA criterion). For each order given by Tabu, Method 1 recreates the new batches, assigns the furnaces, and calculates the new objective function (figure 4).

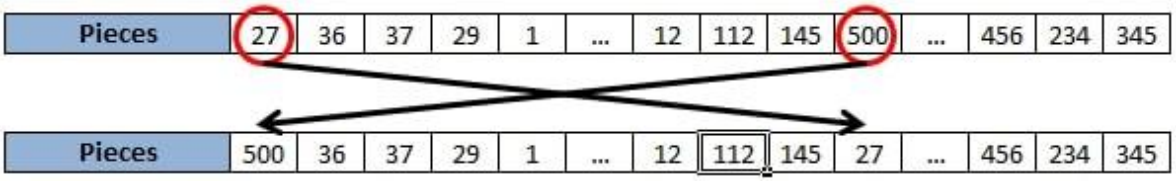
Figure 4. Hybridization between method no. 1 and method no. 2



Source: Own elaboration

The neighborhood explored in each iteration consists in all the swaps that can be done between two pieces as shown in figure 5.

Figure 5. Change the arrangement of the piece, method no. 2



Source: Own elaboration

No aspiration criterion was used in the Tabu. For the stopping criterion we chose the number of iterations without improving the objective function. We performed several tests and beyond 10 iterations the algorithm did not improve any further. We also adjusted the size of the Tabu list to 50 by performing several tests. The computational time depends on the number of pieces we consider, 1 minute for 99 pieces and 7 minutes for 201 pieces.

Numerical Results

To solve the problem, we developed a software that processes the initial data of the problem and finds the programming of the production in a planning horizon of 1 week in AGP.

Before comparing our results to the AGP solution, we ran some instances of our metaheuristic and compared them to the exact mathematical model presented above.

We ran 12 random instances. We had to select small instances for which CPLEX could find an optimal solution in less than an hour. We decided to set this limit as the objective set by the company is to be able to find a good solution in a less than a few minutes.

Table 5. Mathematical model and Tabu search results

Instances	Pieces	Mathematical Model		Tabu Search		Gap (%)
		Obj. Func.	Computational Time (s)	Obj. Func.	Computational Time (s)	
1	**5**	250.75	0	253.75	0	1.20 %
2	**5**	248.74	0	248.75	0	0.00 %
3	**5**	516.83	0	520.83	0	0.77 %
4	**5**	320.30	0	325.31	0	1.56 %
5	**5**	414.26	0	418.27	0	0.97 %
6	**10**	710.25	22	716.25	0	0.84 %
7	**10**	370.99	3	378.00	0	1.89 %
8	**10**	584.88	32	584.90	0	0.00 %
9	**10**	520.47	91	1023.73	0	97 %
10	**10**	326.83	240	883.98	0	170 %
11	**15**	369.49	56	381.50	1	3.25 %
12	**15**	561.80	20	572.81	1	1.96 %

Source: Own elaboration

We can observe in table 5 that in all instances our heuristic runs in less than 1 second. In most cases our gap compared to the optimal solution is less than 5 %. We notice however that in two cases we have solutions that are pretty far from the optimum. Keep in mind that

our objective was to find a solution that is good enough to improve the current solution of the company in a very short time. That is why we will focus the rest of the analysis on the company's case study. However, we leave for further studies the research on the specific characteristics of the instances that make our algorithm unstable.

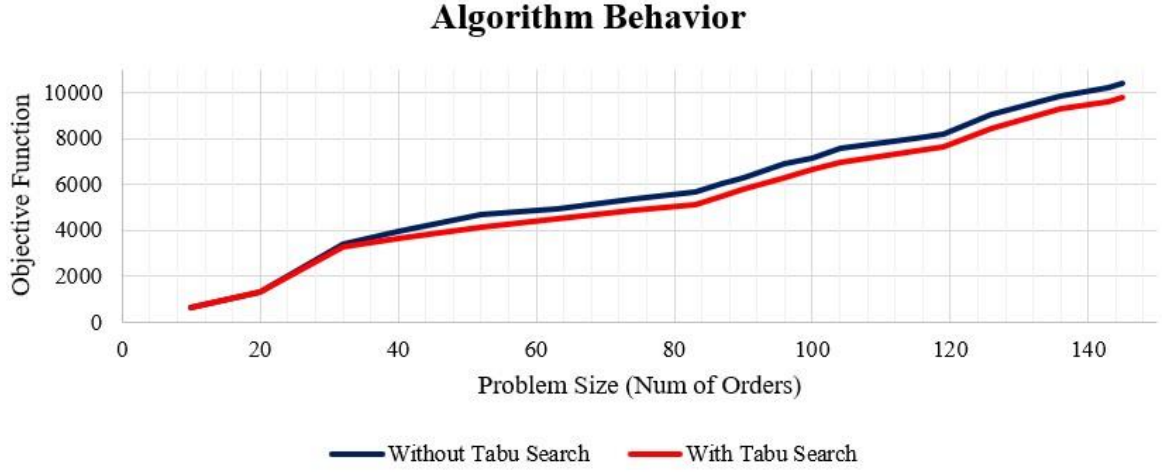
In the following section we present first the results obtained from the case study. This instance has 152 orders that represent 591 pieces. The number of furnaces considered for this case were 4. Once the application is executed according to the input parameters established for this case, we proceeded to analyze the results obtained from the designed methods. We compared our results with the current results of the plant in the selected week that was used as case study.

Case Study

A significant improvement in the objective function was achieved of 31.65 %, represented mostly by a reduction of 32.03 % in hours of furnace use. This is a valuable result for the company as it means they can fulfill the same demand with less resources. We also reduced by 7.42 % the tardiness delivery of orders.

We also analyzed the performance of the algorithm developed in order to illustrate how the Tabu Search improves the objective function obtained by the first method. In this instance, we considered the same 4 furnaces and the same input parameters, but we changed the number of orders to analyze the behavior of the algorithm as the size of the problem changes. We can see that for small instances the Tabu does not improve the objective function in a significant way. For larger instances, the Tabu search improves the initial solution in a range between 5 and 10 % (figure 6)

Figure 6. Behavior of the developed algorithm with and without Tabu Search



Source: Own elaboration

Finally, we performed statistical tests to evaluate the sensitivity to the PIA criteria of both of our methods. We considered sorting the pieces according to an ascending and descending order of the processing time (which is determined by the pieces' ballistic level), width, and delivery dates. Therefore, we can list six different criteria for PIA which could lead to different initial solution values. These statistical tests were build considering samples of ten instances for two scenarios that varied depending on the number of pieces. In the first scenario, each sample is related to an instance with 201 pieces, randomly selected from the 591 pieces available from the information given by AGP. In the second scenario, each sample is related to an instance with 99 pieces, also randomly selected. We then analyzed the difference of the mean objective function obtained with the "descending processing time" rule (which was the one we used in this research as it gave better results) and each one of the 5 remaining ordering criteria. A paired t-test was performed considering a 95 % significance level. The null hypothesis is defined as "the means of the two objective functions are equal." In the case with 201 pieces, the results show that the null hypothesis is rejected for both the heuristic and Tabu search. This means that both methods are sensitive to the PIA criteria.

However, in the case of 99 pieces, the null hypothesis is always accepted for the Tabu, while for the Heuristic, it is only accepted in one comparison and rejected for the 4 remaining ones. This suggests that with smaller problems, Tabu is robust enough to reach the same objective function regardless of the value of the initial solution. Table 6 shows the results.

Table 6. Results example

PIA pairwise comparison	p-value			
	201 pieces		99 pieces	
	With Tabu Search	***Without Tabu Search (Initial solution)***	**With Tabu Search**	***Without Tabu Search (Initial solution)***
Descending processing time ***vs*** Ascending processing time	0.001	0.014	*0.119*	0.027
Descending processing time ***vs*** Ascending width	0.000	0.002	*0.165*	0.017
Ascending processing time ***vs*** Descending width	0.030	0.027	*0.122*	0.030
Descending processing time ***vs*** Ascending due date	0.000	0.002	*0.118*	*0.104*
Descending processing time ***vs*** Descending due date	0.017	0.008	*0.051*	0.021

Source: Own elaboration

Conclusions

This paper considered a real-world scheduling problem commonly observed in AGP. As the problem under study was NP-hard, a hybridization of two methods was developed. The algorithm's results were compared with AGP's previous solution. We showed that optimization techniques can improve the overall production process reducing the objective function by 30 %. Being able to reduce furnace capacity not only means that more orders can be processed in less time, but it also helps AGP to rethink its established procedures.

This project will serve AGP as a first step to define new criteria for the operation in the bending process. A good process optimization allows companies to offer a higher quality service and customer satisfaction, which is a fundamental factor for organizations.

References

[1] P. Brucker, K. Mikhail, and Y. Shafransky, "Batch scheduling with deadlines on parallel machines," *Ann. Oper. Res.*, vol. 83, pp. 23–40, 1998. doi: 10.1023/A:1018912114491

[2] L. Mönch, H. Balasubramanian, and J. Fowler, "Minimizing total weighted tardiness on parallel batch process machines using genetic algorithms," in *Operations Research Proceedings 2002*. Berlin: Springer, 2003, pp. 229–234. doi: 10.1007/978-3-642-55537-4_37

[3] T.-C. Chiang, H.-C. Cheng, and L.-C. Fu, "A memetic algorithm for minimizing totalweighted tardiness on parallel batch machines with incompatible job families and dynamic job arrival," *Comput. Oper. Res.*, vol 37, no. 12, pp. 2257–2269, 2010. doi: 10.1016/j.cor.2010.03.017

[4] A. Klemmt, G. Weigert, C. Almeder, and L. Mönch, "A comparison of mip-based de-composition techniques and vns approaches for batch scheduling problems," *Proc. 2009 Winter Sim. Conf. (WSC)*, pp. 1686–1694, 2009. doi: 10.1109/WSC.2009.5429173

[5] A. Bilyk, L. Mönch, and C. Almeder, "Scheduling jobs with ready times and pre-cedence constraints on parallel batch machines using metaheuristics," *Comput. Ind. Eng.*, vol. 78, 175–185, Oct. 2014. doi: 10.1016/j.cie.2014.10.008

[6] R. Gokhale and M. Mathirajan, "Minimizing total weighted tardiness on heterogeneous batch processors with incompatible job families," *Int. J. Adv. Manuf. Tech.*, vol. 70, pp. 1563–1578, 2014.

[7] M. Amouie, *Minimizing Total Weighted Tardiness for Non-Identical Parallel Batch Processing Machines*. DeKalb, IL: Northern Illinois University, 2014.

[8] M. Hulett, P. Damodaran, and M. Amouie, "Scheduling non-identical parallel batch processing machines to minimize total weighted tardiness using particle swarm optimization," *Comput. Ind. Eng.*, vol. 113, pp. 425–436, 2017. doi: 10.1016/j.cie.2017.09.037

[9] A. Lozano and A. Medaglia, "Scheduling of parallel machines with sequence dependent batches and product incompatibilities in an automotive glass facility," *J. Scheduling*, vol. 17, no. 6, pp. 521–540, 2014. doi:10.1007/s10951-012-0308-7