CONEM
CONGRESSO NACIONAL DE ENGENHARIA MECÂNICA
NATAL - RIO GRANDE DO NORTE - BRASIL

ABCM

XII CONGRESSO NACIONAL DE ENGENHARIA MECÂNICA
DE 29/07 A 02/08 DE 2024, NATAL-RN, BRASIL

CONEM2024-0014
AN EXACT METHOD FOR THE MULTI-PRODUCT P-BATCH PROCESSING TIME MAXIMIZATION PROBLEM

Tatiana Balbi Fraga, tatiana.balbi@ufpe.br[1]
Ítalo Ruan Barbosa de Aquino, italo_ruan_@hotmail.com[1]
Regilda da Costa e Silva Menêzes, regilda.smenezes@ufpe.br[1]

[1]Centro Acadêmico do Agreste, Universidade Federal de Pernambuco, Avenida Marielle Franco, Bairro Nova Caruaru, Caruaru - PE, CEP: 55014-900

Resumo: *Multi-product batch (MPB) problems arise in manufacturing environments where several batches of products are processed sequentially, sharing the same set of production units. Among the most well-known MPB problems stands out the multi-product batch scheduling (MPBS) problems, whose objective is to find one of the feasible batch processing schedules that best meets a set of objectives predefined by the company. Along with the most complex MPBS problems are the multi-product p-batch scheduling (MPpBS) problems, where the schedule occurs on parallel batch processing machines. When applying some local search heuristics to solve the MPpBS problem, it can be necessary to iteratively solve a single-stage multi-product p-batch sizing problem with different processing rates for each product, limited capacity units, and storage capacity defined both for each product and for all products (named in this article multi-product p-batch processing time maximization (MPpBPTM) problem). Then, the efficiency of the method developed for solving an MPpBPTM problem exerts a decisive influence on the computational cost of the local search algorithm applied to solve some MPpBS problems. In addition, the efficiency of solving an MPpBPTM problem can significantly impact storage costs and customer satisfaction. This article presents a mathematical model and an exact method with polynomial time complexity for the MPpBPTM problem. We developed a LINGO solver for solving the mathematical model and a C++ COPSolver library for applying the proposed exact method. We tested both solvers, comparing the results for a series of benchmarks with sizes varying between small-scale and very large-scale. As a result, we found that both solvers can obtain optimal solutions in just a few seconds, even when solving very large-scale instances. However, the exact method is much more efficient since it solves the problem in polynomial time. In addition, the modular structure of the library developed for COPSolver allows the easy application of the proposed solution method when developing new heuristics for solving MPpBS problems, making this tool a significant technological contribution to the production management of some industries as well as for researchers in the area.*

Palavras-chave: *multi-product p-batch problem, processing time maximization, integer programming model, analytical solution, COPSolver*

1. INTRODUCTION

The term multi-product batch production is usually applied to designate an intermittent production system in which batches of different products and sizes are processed one at a time, sharing the same production facilities. A multi-product batch plant basically consists of one or more production units, grouped into one or more stages, and possibly, intermediate storage locations. At each stage some distinct operations are performed and the batches are processed sequentially over the stages (serial batching). As stated by Petkov and Maranas (1998) the batch plant can operate under a single-product campaign (SPC) production mode, where all the batches of a given product are manufactured before production of the next product begins, or under mixed-product campaign (MPC) production modes where more batches are produced per unit time at the expense of increased changeover times and cleanup costs. Also the plant layout can take on different formats, such as: single unit, where all operations needed to process the batches are performed by a single processor; serial flowshop plant, in which all products follow the same linear flow through the production stages; parallel plants in which production units of the same type are arranged in parallel, forming each production stage (Shi *et al.*, 2017); and convergent and/or divergent networked processes, in which units are not arranged in lines, and the connection between the units may or may not exist so that the production paths in the flowshop network can be different for each product and the passing of previous batch can be possible so that the final sequence of the production at the last stage units can be

different from the initial sequence (Kim *et al.*, 1996).

The scientific literature address many different problems related to such industrial enviroment, including the design, planning, sizing, scheduling and balancing problems. The multi-product batch plant design problem is concerned to obtain the configuration of the plant and the equipment that minimize the capital cost of all the equipment items needed to fulfill the production requirements (Ravemark and Rippin, 1998). The configuration of the plant basically consists of defining the number of stages, the number of units and their capacity per stage, and the capacity for intermediate storages, as well as the best factory layout. The multi-product batch planning problem is basically determining the quantities to be produced, the number of batches and their sizes, the needs of materials and the total sales during each time period (Fumero *et al.*, 2016).

Since a single batch is processed at a time, it is necessary that the batches are dimensioned in such a way that the quantity produced of each product is sufficient to meet the demand during the period in which other batches are being processed (Eilon, 1985). This problem is named multi-product batch sizing. Eilon (1985) gives a detailed explanation of the inadequacy of the use of the well-know EBQ (economic batch quantity) square-root formula for multi-product batch production, and Liu *et al.* (2020) present a mixed-integer quadratic programming model and an exact optimization algorithm for a multi-stage multi-product serial batching sizing problem in a divergent networked plant of a steel industry.

In multi-product batch scheduling (MBS) problems it is necessary to determine the ideal batch size for each product and also the production sequence of the batches based on the known demand. Here, the solution must be determined in such a way that both setup and storage costs are minimized (Eilon, 1985; Arcade, 1993; Liu *et al.*, 2020). Some cientific works impose additional constraints on the MBS problem - such as release times and due dates for each batch, and sequencing constraints due to mismatched product colors sequences - and focus on other goals - such as optimize customer satisfaction and/or the plant performance (Méndez *et al.*, 2000; Shi *et al.*, 2017). Also, some cientific works consider MBS problems within different layouts as well as dinamic and multiperiod scenarios. In parallel MBS as well as in networked processes, it can also be necessary to define in which units each batch will be processed, and constraints that limit the units that can process each batch can also be imposed. Many cientifc works propose mixed-integer linear programming (MILP) formulations to solve MBS problems. As example, we can mention the work of Méndez *et al.* (2000) that presents a MILP continuous-time model and a two-step systematic methodology for a single stage multi-product batch scheduling problem with different due-dates. Later, Méndez and Cerdá (2003) also proposed another MILP formulation for reactive scheduling. As observerd by He and Hui (2008), with the problem size increasing, the computational effort of MILP increases greatly so it is very difficult for MILP to obtain acceptable solutions to large-size problems within reasonable time. Citing some works that use other approaches, Kim *et al.* (1996) propose a Gantt chart mapping method and a modified genetic algorithm for solving a networked flowshop multi-product batch scheduling problem as well as a rule-based reactive rescheduling scheme for dinamicaly overcoming in-process deviation from the initial optimized schedule and He and Hui (2008) present a rule-based genetic algorithm for the scheduling of single stage multi-product batch plants with parallel units.

It is importante to note that some production units are capable of processing multiple jobs at the same time. In such a case batches of different products and sizes can be grouped and processed by these units as a single batch. It is therefore necessary to determine the set of "batches" that each new compounded batch will contain. The multi-product balancing problem consists of determining these groupings, seeking to minimize the overall makespan. Husseinzadeh Kashan and Ozturk (2022) name this problem *p-batch* (parallel batching) in contradiction with the *s-batch* (serial batching) problem where jobs are processed one by one over production facilities, as described before. Fowler and Mönch (2022) present a taxonomy and an extensive literature review for the p-batch problems. According to the authors, a p-batch problem can be classified as compatible setting - where there is no restriction on which jobs can be used to form a batch - and incompatible family case - where only jobs belonging to the same family can be used to form a batch. Not mentioned by Fowler and Mönch (2022), Li *et al.* (2022) deal with a multi-product p-batch problem involving identical parallel units, capacity constraints, and incompatible families with different job sizes.

In a real production environment, some of these problems arise in a connected way, so many cientific works adress mixed multi-product batch problems. As example, Fumero *et al.* (2016) developed a multi-period mixed-integer linear programming model for MPCs multistage multi-product batch problems, which integrates design, production planning, and scheduling decisions. Also, the work of Méndez *et al.* (2000), previously mentioned, considers planning as an integrated part of solution in real-world multi-product batch plant scheduling and therefore includes the product batching problem in its proposed MILP continuous-time models and two-step systematic methodology.

1.1 Taxonomy and Notation

In short, based on the bibliographic survey conducted for the preparation of this paper, we identify that the multi-product batch problems can differentiate with the following aspects:

1) **number of stages** (σ): single stage (SS) or multistage (MS);
2) **layout**(λ): single unit (SU), flowshop (Fl), parallel (Pl), or networked (Nt) (convergent (CNt) and /or divergent (DNt));
3) **production mode** (μ): single-product campaing (SPC), mixed-product campaign (MPC), serial batching ($s\text{-}batch$) and/or parallel batching ($p\text{-}batch$);

4) **problem:**(π) : design (D), planning (P), sizing (Sz), scheduling (Sc) and/or balancing (B);

5) **time horizon** (η): single period (SP), multiperiod (MP) or cyclic (Cc);

6) **scenario** (γ): static (St) or dynamic (Dn);

7) **input** (ι): deterministic (Dt) and/or stochastic (St);

8) **performance measure** (δ): capital cost (C), setup cost (SC), holding cost (HC), work-in-process cost ($WIPC$), net profit (NP), makespan (C_{max}), total completion time (TC), weighted completion time (TWC), maximum lateness (L_{max}), maximal tardiness (T_{max}), total tardiness (TT), total weighted tardiness (TWT), the number of tardy jobs (NTJ), weighted number of tardy jobs ($WNTJ$), earliness/tardiness (E/T), weighted E/T (wE/T), customer satisfaction(CS), plant performance (PP) and/or processing time (PT);

9) **others** (ρ): limited capacity units (LCU), storage capacity by product (PSC), storage capacity (SC), work-in-process storage (WIP), release time (RT), due dates (DD), incompatible families (IF), different processing rates (DPR), sequence-dependent setup times ($SDST$), sequencing constraints (SqC).

So, to describe a specific multi-product batch problem, we propose the notation $\sigma|\lambda|\mu|\pi|\eta|\gamma|\iota|\delta|\rho$. Some of the multi-product batch problems and solution approaches discussed in the references cited in this paper are summarized in Tab. 1.

Table 1: **Some contributions on the multi-product batch problems.**

reference	problem	contributions
Eilon (1985)	$SS\|SU\|s\text{-}batch\|Sc\|SP$ $St\|Dt\|SC, HC\|-$	analysis of the effect of splitting batches
Ravemark and Rippin (1998)	$MS\|Pl\|SPC\|D\|Cc\|$ $St\|Dt\|C\|WIP$	convex MILP model solved with DICOPT++
Petkov and Maranas (1998)	$MS\|Pl\|SPC\|D\|Cc\|$ $St\|St\|NP\|-$	exact method
Méndez *et al.* (2000)	$SS\|Pl\|MPC\|P, Sc\|SP\|St\|$ $Dt\|WIPC\|RT, DD$	MILP continuous-time model and a two-step systematic methodology
Méndez and Cerdá (2003)	$SS\|Pl\|MPC\|Sc\|SP\|Dn\|Dt\|$ $wE/T\|RT, DD, SDST$	MIPL model and a rescheduling algorithm
He and Hui (2008)	$SS\|Pl\|MPC\|Sc\|SP\|$ $St\|Dt\|C_{max}, TT\|LCU, DD$	rule-based genetic algorithm
Fumero *et al.* (2016)	$MS\|Pl\|MPC\|D, P, Sc\|Cc\|$ $St\|Dt\|NP\|LCU, USC$	MILP model solved with CPLEX
Liu *et al.* (2020)	$MS\|DNt\|s\text{-}batch\|Sz\|MP\|St\|$ $Dt\|SC, WIPC\|LCU, WIP$	mixed-integer quadratic programming model and exact algorithm
Li *et al.* (2022)	$SS\|Pl\|p\text{-}batch\|Sc, B\|SP\|$ $St\|Dt\|C_{max}\|LCU, RT, IF$	MILP model and a constraint-guided artificial immune system based algorithm

1.2 Contributions of this Article

The work presented in this article is part of a project in which we propose the modeling and solution of a multiperiod multi-product batch scheduling problem that arises in a networked plant of a plastic bag industry. The industrial plant of this company is very similar to the multi-product s-batch networked plant of the steel industry presented by Liu *et al.* (2020) and illustrated in Fig. (**??**), however, the production process in this plastic bags industry begins in extruders, which are production units capable of processing different batches simultaneously. Therefore the problem we are interested is a multi-product *p-batch s-batch* problem in a networked plant close to the problem addressed by Li *et al.* (2022). Thus, part

of the problem solving process consists of solving a multi-product p-batch scheduling problem into extruders. Also, in the problem we study, in addition to demand, we must consider the factory and outlets' storage capacity, which is individually defined for products, but also for the set of products.

When we began to propose an algorithm to solve the studied problem through a local search method, we identified the need to determine, at each iteration, a maximum processing time for predefined multi-product p-batches, based on known restrictions, so that the new solution built was a feasible solution and at the same time with the best fitness. Using the notation presented before, this problem can be classified as a multi-product $SS|SU|p\text{-}batch|Sz|SP|St|Dt|PT|LCU, PSC, SC, DPR$ problem. For simplification purposes, we named this problem multi-product p-batch processing time maximization (MPBPTM) problem.

It is important to note that, since the MPBPTM problem need to be solved at each iteration, it has a direct impact on the overall efficiency of the local search method proposed for solving the multi-product p-batch s-batch problem in the plastic bag networked plant of the industry studied. In a future work, we intend to present the mathematical model and a Particle Collison based solution algorithm we are developing for this last problem. As scientific contributions of this paper, we present a mathematical model and a very efficient polynomial optmization method for solving the MPBPTM problem.

The rest of the paper is organized as follows. In the next sections, the MPBPTM problem is introduced. Section 2.1 describes an interger mathematical model for the problem and section 3 shows an analytical method for its solution. Section 4 displays the tests and results obtained by a solver in C++ applying the analytical solution and a solver developed in LINGO to new proposed benchmarks. Section 5 presents the paper's conclusions and suggestions for future works. Finally, in sections 6 and 7, the contributions to this work are informed and acknowledgments are given, respectively.

2. MULTI-PRODUCT P-BATCH PROCESSING TIME MAXIMIZATION PROBLEM

The multi-product p-batch processing time maximization problem arises when a set of different products are processed simultaneously in a same production batch. In this problem, it is considered that the quantity produced of each product is directly proportional to the processing time, however, with a different constant of proportionality (production rate) for each product. In addition, there is a maximum quantity allowed for the production of batch products, defined both individually and for the set. The production quantity of each product is mainly defined according to the demand for the product. However, it is still possible to stock the products and / or send them to the outlets. In both cases, there is a stocking / shipping limit for each product and a stocking / shipping limit for the set of products in the batch. Also, there is a time limit available for processing the batch. About the production, the industry manager has as priority to meet demand and he prefer to send the remaining production to outlets rather then stock it in the factory.

2.1 Mathematical Model

Given that:
UD_i is the demand for the product i;
I is the maximum quantity allowed for additional factory storage of all products in the batch;
UI_i is the maximum quantity allowed for stocking the product i in the factory;
O is the maximum quantity allowed for shipment of all products to outlets;
UO_i is the maximum amount of product i that can be shipped to outlets;
p_i is the production rate of product i;
Z is the timeout for batch processing;
P_i is the amount of product i produced;
D_i is the amount of product i delivered for the demand;
O_i amount of product i shipped to factory outlets;
I_i is the amount of product i that will be stored at the factory;
T is the batch processing time;

We have the problem:

$$max \quad T \tag{1}$$

$s.t.$

$$P_i - \text{p}_i * T = 0 \quad \forall i \tag{2}$$

$$P_i - D_i - O_i - I_i = 0 \quad \forall i \tag{3}$$

$$D_i \leq \text{UD}_i \quad \forall i \tag{4}$$

$$O_i \leq \text{UO}_i \quad \forall i \tag{5}$$

$$\sum_i O_i \leq \text{O} \tag{6}$$

$$I_i \leq \mathrm{UI}_i \quad \forall i \tag{7}$$

$$\sum_i I_i \leq \mathrm{I} \tag{8}$$

$$T \leq \mathrm{Z} \tag{9}$$

$$\min\{\mathrm{UD}_i - D_i, O_i + I_i\} = 0 \quad \forall i \tag{10}$$

$$\min\{\mathrm{O} - \sum_i \{O_i\}, \mathrm{UO}_i - O_i, I_i\} = 0 \quad \forall i \tag{11}$$

$$T, D_i, O_i, I_i \in \mathbb{Z}^+ \quad \forall i \tag{12}$$

where constraints in (2) relate the quantity produced, P_i, to batch processing time T. Constraints in (3) calculate the quantity produced, P_i, as a function of the primary variables, D_i, O_i and I_i. Constraints in (4), (5), and (7) state that the quantity delivered to demand, the quantity shipped to the outlets, and the factory-stocked quantity of each product must be less than their respective known limits. Constraints (6) and (8) state that both the sum of product quantities sent to the outlets and the sum of product quantities stocked in the factory must be less than their respective maximum allowed values. The restriction in (9) establishes that there is a batch processing time limit, Z, that must be respected. Equations in (10) and (11) define the production distribution priority. And finally, the constraints in (12) inform the nature of the decision variables.

3. ANALYTICAL SOLUTION

The method proposed in this paper for solution of the MPBPTM problem is composed of two distinct parts, which are: a) calculation of the maximum processing time of the predefined multi-product p-batch; and b) calculation of the part of the production used to meet demand, the part of the production sent to outlets and the part of the production stocked in factory. Next, we explain each of these parts in detail.

Calculation of maximum p-batch processing time:

It is possible to split the batch processing time into two time slots:

$$T = T' + T'' \tag{13}$$

where T' is the maximum processing time whose production will be used exclusively to meet the demand.

Then, being $D_i^{'}$ the quantity produced of the product i after T' units of processing time, we have:

$$D_i^{'} - \mathrm{p}_i * T' = 0 \quad \forall i \tag{14}$$

$$D_i^{'} \in Z \quad \forall i \tag{15}$$

and, as $D_i^{'} \leq D_i, \forall i$, we have:

$$D_i^{'} \leq \mathrm{UD}_i \quad \forall i \tag{16}$$

From the equations in (14) and the restrictions in (15) and (16), we have:

$$T' \leq \lfloor \mathrm{UD}_i / \mathrm{p}_i \rfloor \quad \forall i \tag{17}$$

And, as we wish maximum value to T', from (17) we have:

$$T' = \min_i \{\lfloor \mathrm{UD}_i / \mathrm{p}_i \rfloor\} \tag{18}$$

Observe that there may be a leftover corresponding to the demand not met by the production in the first part of the processing time, T'. This leftover, S_i, will be defined according to the equations in (19):

$$S_i = \mathrm{UD}_i - D_i^{'} \quad \forall i \tag{19}$$

It is also possible to consider that outlets are extensions of the factory stock. Therefore, it is possible to consider the factory stock of product i and the outlets stock i as single stock for the product i, E_i, so that:

$$E_i = O_i + I_i \tag{20}$$

$$E_i \leq \mathrm{UO}_i + \mathrm{UI}_i \tag{21}$$

$$\sum_i E_i \leq \mathrm{O} + \mathrm{I} \tag{22}$$

$$E_i \in Z \quad \forall i \tag{23}$$

As the time T' was used only for production that will meet the demand, T'' will be the remaining processing time, *i.e.* the maximum processing time used for production that will meet the leftover of demand, sent to the outlets and stored in factory. So we have:

$$E_i + S_i - \mathrm{p}_i * T'' = 0 \quad \forall i \tag{24}$$

Thus, after replacing (24) in (21) and applying some algebra, we have:

$$T'' \leq (\mathrm{UO}_i + \mathrm{UI}_i + S_i)/\mathrm{p}_i \quad \forall i \tag{25}$$

And, after replacing (24) in (22) and applying some algebra, we have:

$$T'' \leq (\mathrm{O} + \mathrm{I} + \sum_i S_i)/\sum_i \mathrm{p}_i \tag{26}$$

Finally, as T'' is an integer value and we want to maximize this value, then:

$$T'' = \lfloor \min\{\min_i\{(\mathrm{UO}_i + \mathrm{UI}_i + S_i)/\mathrm{p}_i\}, (\mathrm{O} + \mathrm{I} + \sum_i S_i)/\sum_i \mathrm{p}_i\}\rfloor \tag{27}$$

So, the analytical method for finding maximum T can be decomposed into 4 steps:
step 1: find T', where:

$$T' = \lfloor \min_i\{\mathrm{UD}_i/\mathrm{p}_i\}\rfloor \tag{28}$$

step 2: calculate S_i for all products.

$$S_i = \mathrm{UD}_i - \mathrm{p}_i * T' \quad \forall i \tag{29}$$

step 3: find T'', where:

$$T'' = \lfloor \min\{\min_i\{(\mathrm{UO}_i + \mathrm{UI}_i + S_i)/\mathrm{p}_i\}, (\mathrm{O} + \mathrm{I} + \sum_i S_i)/\sum_i \mathrm{p}_i\}\rfloor \tag{30}$$

step 4: calculate T, where:

$$T^* = T' + T'' \tag{31}$$

$$T = \min\{T^*, \mathrm{Z}\} \tag{32}$$

Note that, as we reduce the value of T', the consequent reduction of $D_i^{'}$ will be directly attributed to the leftover of demand S_i and, therefore, there will be no change in the optimal solution if it is considered to T', a value between 0 and the value found in step 1. Thus it is possible to use a simplification of the method presented, making $T' = 0$. In this case $S_i = \mathrm{UD}_i, \forall i$. The simplified version of the method is presented below:
step 1: find T^*, where:

$$T^* = \lfloor \min\{\min_i\{(\mathrm{UO}_i + \mathrm{UI}_i + \mathrm{UD}_i)/\mathrm{p}_i\}, (\mathrm{O} + \mathrm{I} + \sum_i \mathrm{UD}_i)/\sum_i \mathrm{p}_i\}\rfloor \tag{33}$$

step 2: calculate T, where:

$$T = \min\{T^*, \mathrm{Z}\} \tag{34}$$

This simplification of the method demonstrates that the value calculated to T does not depend on the way the production of the product i, P_i, is distributed between service to demand, D_i, outlets, O_i, and factory's stock, I_i.
Full solution:

We can determine now the values of D_i, O_i and I_i, $\forall i$, simply attending the priorities for distribution. The following three algorithms can be used to find a complete solution.

The Algorithm (1) is used for finding an optimimal value for T.

Then an initial production distribution can be done by the Algorithm (2). This distribution is made without considering the restrictions defined for the set of products. SO and SI represent, respectively, the total amount that can be sent to the outlets and the total amount that can be stocked in factory after the initial distribution. If this values are negative, this indicates that the proposed initial distribution is not a feasible distribution.

So, after applying Algorithm (2), if $SO \geq 0$ and $SI \geq 0$, solution found is an optimal solution. However, if $SO < 0$ or $SI < 0$, solution is not yet feasible. But as the solution found to T is feasible to the problem defined in (1) to (12), we have the following statements:
$if \quad SO < 0 \quad then \quad SI > 0 \quad and \quad |SO| < SI$
$if \quad SI < 0 \quad then \quad SO > 0 \quad and \quad |SI| < SO$

Therefore it is possible to meet the restrictions by applying the Agorithm (3). This algorithm redistributes the production of factory's stock to the outlets or vice versa, respectively, due to having a SI or SO negative.

Algorithm 1 Solving MBPTM problem | Part 01 - find an optimal T to the problem defined in (1) to (12).

Require: $\mathrm{UD}_i, \mathrm{UO}_i, \mathrm{UI}_i, \mathrm{p}_i, \forall i, \mathrm{O}, \mathrm{I}, \mathrm{Z}$
$T^* \leftarrow \lfloor \min\{\min_i\{(\mathrm{UO}_i + \mathrm{UI}_i + \mathrm{UD}_i)/\mathrm{p}_i\}, (\mathrm{O} + \mathrm{I} + \sum_i \mathrm{UD}_i)/\sum_i \mathrm{p}_i\} \rfloor$
$T \leftarrow \min\{T^*, \mathrm{Z}\}$
Return: T.

Algorithm 2 Solving MBPTM problem | Part 02 - calculate D_i, O_i and I_i, $\forall i$, ignoring the restrictions for the set of products of the batch.

Require: $\mathrm{UD}_i, \mathrm{UO}_i, \mathrm{UI}_i, \mathrm{p}_i, \forall i, \mathrm{O}, \mathrm{I}, \mathrm{Z}, T$.
for all i **do**
$E_i \leftarrow T * \mathrm{p}_i$
$D_i \leftarrow \min\{\mathrm{UD}_i, E_i\}$
$E_i \leftarrow E_i - D_i$
$O_i \leftarrow \min\{\mathrm{UO}_i, E_i\}$
$E_i = E_i - O_i$
$\mathrm{UO}_i = \mathrm{UO}_i - O_i$
$I_i = E_i$
$\mathrm{UI}_i = \mathrm{UI}_i - I_i$
end for
$SO = \mathrm{O} - \sum_i O_i$
$SI = \mathrm{I} - \sum_i I_i$
Return: $D_i, O_i, I_i, E_i, \mathrm{UO}_i, \mathrm{UI}_i, \forall i, SO, SI$.

Algorithm 3 Solving MBPTM problem | Part 03: redistribute production to comply with limitation restrictions for the batch set.

Require: $O_i, I_i, \mathrm{UO}_i, \mathrm{UI}_i, \forall i, SO, SI$.
if $SO < 0$ **then**
for all i **do**
$O_i \leftarrow O_i - \min\{O_i, UI_i, |SO|\}$
$I_i \leftarrow I_i + \min\{O_i, UI_i, |SO|\}$
$SO \leftarrow SO + \min\{O_i, UI_i, |SO|\}$
if $SO = 0$ **then** break looping for;
end if
end for
end if
if $SI < 0$ **then**
for all i **do**
$I_i \leftarrow I_i - \min\{I_i, UO_i, |SI|\}$
$O_i \leftarrow O_i + \min\{I_i, UO_i, |SI|\}$
$SI \leftarrow SI + \min\{I_i, UO_i, |SI|\}$
if $SI = 0$ **then** break looping for;
end if
end for
end if
Return: $O_i, I_i, \forall i$.

4. TESTS AND RESULTS

To test the developed model and analytical solution method, we created a solver in C++ named COPSolver: library for solving the multi-product p-batch processing time maximization problem and also a solver in LINGO named MBPTM.lng both available at codeocean and GitHub. A brief explanation of the software and links for downloads can be found in the paper of Fraga (2023). The tests were performed on a notebook with an Intel i7 processor. We tested the solvers developed for the benchmarks presented on Tables 2, 3 and 4, and for randon benchmarks generated by the following functions:

$$\mathrm{p}_i = \mathrm{rand}()\%30 + 10 \tag{35}$$

$$\mathrm{UD}_i = \mathrm{rand}()\%3000 + 800; \tag{36}$$

$$\mathrm{seed1} = \mathrm{rand}()\%3000 + 500; \tag{37}$$

$$\mathrm{seed2} = \mathrm{rand}()\%5000 + 1000; \tag{38}$$

$$\mathrm{O} = \mathrm{N}/2 * \mathrm{seed1}; \tag{39}$$

$$\mathrm{UO}_i = \mathrm{rand}()\%(\mathrm{seed1} - 500) + 500; \tag{40}$$

$$\mathrm{I} = \mathrm{N}/2 * \mathrm{seed2}; \tag{41}$$

$$\mathrm{UI}_i = \mathrm{rand}()\%(\mathrm{seed2} - 1000) + 1000; \tag{42}$$

$$\mathrm{Z} = 100; \tag{43}$$

Table 2: **Benchmark MPBPTM 2.**

i	1	2	total
p_i	60	40	
UD_i	1000	500	
UO_i	600	600	1000
UI_i	3000	2000	3000
		Z	100

Table 3: **Benchmark MPBPTM 3.**

i	1	2	3	total
p_i	60	40	50	
UD_i	1000	500	800	
UO_i	600	600	600	1500
UI_i	3000	2000	1000	3500
			Z	100

Table 4: **Benchmark MPBPTM 10.**

i	1	2	3	4	5	6	7	8	9	10	total
p_i	60	40	50	40	30	50	60	10	20	40	
UD_i	1000	500	800	500	400	500	2000	300	500	1000	
UO_i	600	600	600	1500	300	200	500	800	0	200	3000
UI_i	3000	2000	1000	800	3000	1000	400	300	200	0	5000
										Z	100

Randomly generated benchmarks were named RMPBPTM N, being N the number of products. Seeking to enable the reproduction of the results, in the computational construction of the benchmarks we used the function srand((unsigned)

source), where source is a defined value. To build the results presented in this work, we used source=0. The benchmarks used for the tests performed can be consulted at github.com/tbfraga/COPSolver.

Table 5 presents the results obtained by applying the analytical method (COPSolver) and the LINGO solver for the solution of the previously presented benchmarks.

Table 5: **Results obtained with the LINGO solver and the analytical method.**

problem	LINGO solver	time (s)	analytical method	time (s)
MPBPTM 2	55	0.03	55	< 0.01
MPBPTM 3	48	0.03	48	< 0.01
MPBPTM 10	30	0.05	30	< 0.01
RMPBPTM 20	100	0.06	100	< 0.01
RMPBPTM 50	98	0.08	98	< 0.01
RMPBPTM 100	98	0.11	98	< 0.01
RMPBPTM 1,000	78	1.20	78	< 0.01
RMPBPTM 2,000	70	3.25	70	< 0.01
RMPBPTM 5,000	70	15.12	70	< 0.02
RMPBPTM 10,000	70	55.19	70	< 0.04

Tables (6) and (7) shows the production distribution for MBPTM 3 and 10 found by both COPSolver and LINGO solver. Files containing all results can be found at github.com/tbfraga/COPSolver.

Table 6: **Results obtained with the LINGO solver and COPSolver for the MPBPTM 3.**

product	production	delivered (s)	send to outlets	stocked in factory
P1	2,880	1,000	300	1,580
P2	1,920	500	600	820
P3	2,400	800	600	1,000

Table 7: **Results obtained with the LINGO solver and COPSolver for the MPBPTM 10.**

product	production	delivered (s)	send to outlets	stocked in factory
P1	1,800	1,000	400	400
P2	1,200	500	600	100
P3	1,500	800	600	100
P4	1,200	500	700	0
P5	900	400	300	200
P6	1,500	500	200	800
P7	1,800	1800	0	0
P8	300	300	0	0
P9	600	500	0	100
P10	1,200	1000	200	0

It is important to point out that the LINGO solver was developed with the purpose of validating the results found by the proposed analytical method. As the analytical method is a polynomial time complexity method, we did not intend to compare computational costs, however it is possible to verify that both solvers are capable of finding optimal solutions for the proposed benchmarks very quickly, even for very large problems.

We have considered here a static problem, so in a future work we will study the complexities of solving the problem dynamically and try to propose new solution methods if needed.

5. CONCLUSIONS AND SUGGESTIONS FOR FUTURE WORKS

In this paper we presented the multi-product p-batch processing time maximization problem, as well as a mathematical model and an exact analytical solution method for this problem. The mathematical model and analytical method were tested, respectively, by a solver developed with the LINGO software, from LINDO Systems, and with a solver developed in C++ language. Optimum results were found for all proposed benchmarks, very quickly, even in the case of very large problems, which demonstrates the efficiency of the proposed method. As in this paper we considered a static problem, in a future work we will study the complexities that arise when considering the same problem in a dinamic scenario. We will also verify if there is the possibility of extending the proposed analytical method to solve a class of integer programming problems with similar characteristics to the studied problem.

6. CRediT AUTHORSHIP CONTRIBUTION STATEMENT

T.B. Fraga: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data Curation, Writing – original draft, Writing – review & editing, Supervision, Project administration. Í.R.B. Aquino: Investigation. R.C.S. Menêzes: Investigation.

7. ACKNOWLEDGMENTS

We are enormously grateful to Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) and to Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) for the financial support provided to our projects. We also thank LINDO systems team for the LINGO software license, without which this work would not have been possible and the to the owner of the company in the plastics sector, who allowed us to learn about his company's production process. Finally, we would like to thank Pró-reitoria de Extensão e Cultura da UFPE (PROExC) and the Research Director of Propesqi (Pró-reitoria de Pesquisa e Inovação da UFPE) for their support and recognition of our work, and dear Professors Antônio José da Silva Neto and João Flávio Vieira de Vasconcellos from IPRJ/UERJ, who contributed significantly to the formation of essential skills for the development of our projects. We also thank my co-worker Marcos Luiz Henrique, for having helped by evaluating the mathematical model and solution method proposed in this paper.

8. REFERENCES

Arcade, S., 1993. "Single machine multi-product batch scheduling: Testing several solution methods". *Omega*, Vol. 21, No. 6, pp. 709–711. ISSN 0305-0483. doi:https://doi.org/10.1016/0305-0483(93)90012-A.

Eilon, S., 1985. "Multi-product batch production on a single machine—a problem revisited". *Omega*, Vol. 13, No. 5, pp. 453–468. ISSN 0305-0483. doi:https://doi.org/10.1016/0305-0483(85)90073-8.

Fowler, J.W. and Mönch, L., 2022. "A survey of scheduling with parallel batch (p-batch) processing". *European Journal of Operational Research*, Vol. 298, No. 1, pp. 1–24. ISSN 0377-2217. doi:https://doi.org/10.1016/j.ejor.2021.06.012.

Fraga, T.B., 2023. "Copsolver: Open source software for solving combinatorial optimization and other decision problems — library for solving the multi-product p-batch processing time maximization problem". *Software Impacts*, Vol. 18, p. 100592. ISSN 2665-9638. doi:https://doi.org/10.1016/j.simpa.2023.100592.

Fumero, Y., Moreno, M.S., Corsano, G. and Montagna, J.M., 2016. "A multiproduct batch plant design model incorporating production planning and scheduling decisions under a multiperiod scenario". *Applied Mathematical Modelling*, Vol. 40, No. 5, pp. 3498–3515. ISSN 0307-904X. doi:https://doi.org/10.1016/j.apm.2015.09.046.

He, Y. and Hui, C.W., 2008. "A rule-based genetic algorithm for the scheduling of single-stage multi-product batch plants with parallel units". *Computers & Chemical Engineering*, Vol. 32, No. 12, pp. 3067–3083. ISSN 0098-1354. doi:https://doi.org/10.1016/j.compchemeng.2008.04.008.

Husseinzadeh Kashan, A. and Ozturk, O., 2022. "Improved milp formulation equipped with valid inequalities for scheduling a batch processing machine with non-identical job sizes". *Omega*, Vol. 112, p. 102673. ISSN 0305-0483. doi:https://doi.org/10.1016/j.omega.2022.102673.

Kim, M., Jung, J.H. and Lee, I.B., 1996. "Intelligent scheduling and monitoring for multi-product networked batch processes". *Computers & Chemical Engineering*, Vol. 20, pp. S1149–S1154. ISSN 0098-1354. doi:https://doi.org/10.1016/0098-1354(96)00199-8. European Symposium on Computer Aided Process Engineering-6.

Li, C., Wang, F., Gupta, J.N. and Chung, T., 2022. "Scheduling identical parallel batch processing machines involving incompatible families with different job sizes and capacity constraints". *Computers & Industrial Engineering*, Vol. 169, p. 108115. ISSN 0360-8352. doi:https://doi.org/10.1016/j.cie.2022.108115.

Liu, G., Li, F., Yang, X. and Qiu, S., 2020. "The multi-stage multi-product batch-sizing problem in the steel industry". *Applied Mathematics and Computation*, Vol. 369, p. 124830. ISSN 0096-3003. doi:https://doi.org/10.1016/j.amc.2019.124830.

Méndez, C., Henning, G. and Cerdá, J., 2000. "Optimal scheduling of batch plants satisfying multiple product orders with different due-dates". *Computers & Chemical Engineering*, Vol. 24, No. 9, pp. 2223–2245. ISSN 0098-1354. doi:https://doi.org/10.1016/S0098-1354(00)00584-6.

Méndez, C.A. and Cerdá, J., 2003. "Dynamic scheduling in multiproduct batch plants". *Computers & Chemical Engineering*, Vol. 27, No. 8, pp. 1247–1259. ISSN 0098-1354. doi:https://doi.org/10.1016/S0098-1354(03)00050-4. 2nd Pan American Workshop in Process Systems Engineering.

Petkov, S.B. and Maranas, C.D., 1998. "Design of single-product campaign batch plants under demand uncertainty". *AIChE Journal*, Vol. 44, No. 4, pp. 896–911. doi:https://doi.org/10.1002/aic.690440415.

Shi, B., Qian, X., Sun, S. and Yan, L., 2017. "Rule-based scheduling of multi-stage multi-product batch plants with parallel units". *Chinese Journal of Chemical Engineering*, Vol. 25, No. 8, pp. 1022–1036. ISSN 1004-9541. doi:https://doi.org/10.1016/j.cjche.2017.03.014.

9. COPYRIGHT LIABILITY