

Highlights

Multi-product Batch Processing Time Maximization Problem

Tatiana Balbi Fraga, Ítalo Ruan Barbosa de Aquino, Regilda da Costa e Silva Menêzes

- brief bibliographic review on multi-product batch problems;
- Multi-product Batch Processing Time Maximization (MBPTM) problem definition;
- linear integer programming model for the MBPTM problem;
- exact optimization method for solving the MBPTM problem.

Multi-product Batch Processing Time Maximization Problem

Tatiana Balbi Fraga^{a,*}, Ítalo Ruan Barbosa de Aquino^a, Regilda da Costa e Silva Menêzes^a

^aAgreste Academic Center - Federal University of Pernambuco, Avenida Marielle Franco, Nova Caruaru, Caruaru, 55014-900, PE, Brazil

Abstract

In the plastic bag extrusion process, it is necessary to determine the optimal processing time for batches formed by different products, which are processed simultaneously by the same extruder, but with different processing rates. The batch processing time must be determined in order to meet a series of known constraints, such as the limitation for the quantity produced for each product and for the quantity produced for the set of all products in the same batch. In this paper we present this problem as a new combinatorial optimization problem named Multi-product Batch Processing Time Maximization (MBPTM) problem. We also present a mathematical model for the MBPTM problem and an analytical solution method with polynomial time complexity, which proved to be able to obtain optimal solutions for several benchmarks in a very short time, even for very large instances.

*corresponding author

Email addresses: `tatiana.balbi@ufpe.br` (Tatiana Balbi Fraga),
`italo_ruan@hotmail.com` (Ítalo Ruan Barbosa de Aquino),
`regilda.smenezes@ufpe.br` (Regilda da Costa e Silva Menêzes)

Keywords: multiproduct batch, processing time maximization,
mathematical model, analytical solution, LINGO

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 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. However, the plant layout can take on different formats, such as: single unit; serial flowshop plant, in which all products follow a linear flow through the production stages; parallel plants in which production units of the same type are arranged in parallel, forming each production stage (Méndez *et al.*, 2000; Shi *et al.*, 2017); and networked processes, in which production 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, sizing, scheduling and balanc-

ing 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 how many stages, how many units at each stage, and what capacity for intermediate storages will be necessary, as well as the best factory layout.

Since a single products batch is processed at a time, it is also 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 products batches are being processed. This problem is named multi-product batch sizing.

We also have the multi-product batch scheduling (MPBS) problems, in which it is necessary to determine the ideal batch size for each product and also the production sequence of the products batches based on the known demand (Eilon, 1985). Here, the solution must be determined in such a way that both setup and storage costs are minimized (Eilon, 1985; Omega Journal, 1993; Liu *et al.*, 2020). Some scientific works impose additional constraints on the MPBS problem - such as release times and due dates for each products batch production, 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 scientific works consider MPBS problems within different

layouts and multiperiod scenario. In parallel MPBS 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.

It is important 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 grouped batch will contain. The multi-product balancing problem consists of determining these groupings, seeking to minimize the overall makespan. 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.

As normally, in a real production environment, some of these problems arise in a connected way, many authors work with mixed multi-product batch problems.

Li et al. (2022) deal with a PMPBS problem involving incompatible families with different job sizes and capacity constraints. In this problem, solutions are accepted in which two or more batches are processed simultaneously on the same machine. However, there are no production limitations for the production of each product group. The paper of Li et al. (2022) also presents an extensive literature review on production batch scheduling problems. Other important works deal with the problem of a multi-product batch plant

design (MPBPD). In this case the problem is 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).

About the solution method many papers propose mathematical models which are solved using IBM ILOG CPLEX or another constraint programming solver.

Another not very common definition for multi-product batch production is a type of process in which several products are processed simultaneously in the same production unit.

In this work, we are concerned with determining the ideal processing time for a single batch consisting of a set of products, each product being or not different from the other products of the same batch. That is, we consider a specific case where different products are processed simultaneously on the same processor. This case occurs, for example, during the processing of plastic bags in extruders, where the resins are melted, then pass through a cylinder forming a balloon and this balloon, after cooling, is stored in a coil which, in turn, is cut into several smaller coils with different models of plastic bags.

As scientific contributions, in this paper we define a new multi-product batch problem, named multi-product batch processing time maximization (MBPTM) problem, as well as a mathematical model and a polynomial time complexity analytical solution optimization method for solving this problem.

In the next sections, a presentation of problem is made along with an application example. Section 3 presents an interger linear mathematical model for the problem and section 4 presents an analytical method for its solution. Section 5 presents the tests and results obtained by a solver in C++ applying the analytical solution and a solver developed in LINGO to new proposed benchmarks. In sections 7 and 8, the contributions to this work are informed and acknowledgments are given, respectively. Finally, section 6 presents the paper's conclusions and suggestions for future works.

2. Multi-product batch processing time maximization problem

The multi-product 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 than stock it in the factory.

Example: A certain machine must process a batch containing 2 different products: A and B. The production rate of A is 60 g/min while the production rate of B is 40 g/min. The factory has free stock for a maximum of 3000 g of any product, and, according to the company's inventory policy, a maximum of 3000 g of product A and 2000 g of product B can be stocked at the factory. There is a demand for 1000 g of product A and 500 g of product B. The factory has an outlet that has free space in stock of 1000 g, which can receive a maximum of 600 g of each product. A maximum time of 100 minutes of this machine can be allocated for processing this batch. When distributing production the priority is to meet demand, then send to outlets and finally stock in factory. What is the maximum possible time for processing this batch and how production must be distributed ?

3. 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.

$$P_i - p_i * T = 0 \quad \forall i \tag{2}$$

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

$$D_i \leq UD_i \quad \forall i \tag{4}$$

$$O_i \leq UO_i \quad \forall i \tag{5}$$

$$\sum_i O_i \leq O \quad (6)$$

$$I_i \leq UI_i \quad \forall i \quad (7)$$

$$\sum_i I_i \leq I \quad (8)$$

$$T \leq Z \quad (9)$$

$$\min\{UD_i - D_i, O_i + I_i\} = 0 \quad \forall i \quad (10)$$

$$\min\{O - \sum_i \{O_i\}, UO_i - O_i, I_i\} = 0 \quad \forall i \quad (11)$$

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

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.

4. Analytical solution

O método proposto neste artigo para solução do MBPTM problem é composto por duas partes distintas, que são: a) cálculo do tempo máximo de processamento do lote multi-produtos; e b) cálculo e distribuição da produção entre a demanda, os outlets e o estoque em fábrica. A seguir, explicamos cada uma destas partes detalhadamente.

cálculo do tempo máximo de processamento do lote multi-produtos:

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

$$T = T' + T'' \quad (13)$$

where T' é o tempo máximo de processamento cuja produção será utilizada exclusivamente para atendimento da demanda.

Neste caso, sendo D'_i a quantidade produzida do produto i após T' unidades de tempo de processamento do lote, temos que:

$$D'_i - p_i * T' = 0 \quad \forall i \quad (14)$$

$$D'_i \in Z \quad \forall i \quad (15)$$

e, como $D'_i \leq D_i, \forall i$, temos que:

$$D'_i \leq UD_i \quad \forall i \quad (16)$$

Das equações em (14), (15) e (16), temos:

$$T' \leq \lfloor UD_i/p_i \rfloor \quad \forall i \quad (17)$$

E, como desejamos máximo valor para T' , de (17), temos:

$$T' = \min_i \{ \lfloor UD_i/p_i \rfloor \} \quad (18)$$

Observer que poderá haver uma sobra correspondente à demanda não atendida pela produção ocorrida na primeira parte do tempo de processamento do lote, T' . Esta sobra, S_i , será definida de acordo com as equações em (19):

$$S_i = UD_i - D'_i \quad \forall i \quad (19)$$

É possível também considerar que os outlets são uma extensão do estoque

da fábrica. Portanto it is possible to consider the factory stock of product i and the outlets stock i as single stock for the product i , de forma que:

$$E_i = O_i + I_i \quad (20)$$

$$E_i \leq \text{UO}_i + \text{UI}_i \quad (21)$$

$$\sum_i E_i \leq \text{O} + \text{I} \quad (22)$$

$$E_i \in Z \quad \forall i \quad (23)$$

where E_i is the sum of factory stock and outlets stock of the product i .

Como o tempo T' foi utilizado apenas para produção que irá atender à demanda, T'' será o restante do tempo de processamento, ou seja, o tempo máximo de processamento utilizado para produção que irá atender à sobra da demanda, será enviada para os outlets e estocada em fábrica. Assim temos que:

$$E_i + S_i - p_i * T'' = 0 \quad \forall i \quad (24)$$

Portanto, das restrições em (21) e equações em (24), temos:

$$T'' \leq (\text{UO}_i + \text{UI}_i + S_i)/p_i \quad \forall i \quad (25)$$

e, das restrições em (22) e equações em (24), temos:

$$T'' \leq (O + I + \sum_i S_i) / \sum_i p_i \quad (26)$$

Finalmente, como T'' é um valor inteiro e queremos maximizar esse valor, então:

$$T'' = \lfloor \min\{\min_i\{(UO_i + UI_i + S_i)/p_i\}, (O + I + \sum_i S_i) / \sum_i p_i\} \rfloor \quad (27)$$

So, the analytical method for finding maximum T can be decomposed into 4 steps:

step 1: find T' , where:

$$T' = \lfloor \min_{\forall i}\{UD_i/p_i\} \rfloor \quad (28)$$

step 2: calculate S_i for all products.

$$S_i = UD_i - p_i * T' \quad \forall i \quad (29)$$

step 3: find T'' , where:

$$T'' = \lfloor \min\{\min_{\forall i}\{(UO_i + UI_i + S_i)/p_i\}, (O + I + \sum_i S_i) / \sum_i p_i\} \rfloor \quad (30)$$

step 4: calculate T , where:

$$T^* = T' + T'' \quad (31)$$

$$T = \min\{T^*, Z\} \quad (32)$$

Note that, à medida em que reduzimos o valor de T' , a consequente redução do valor de D'_i será atribuída diretamente à sobra S_i e, portanto, 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 $S_i = UD_i, \forall i$. The simplified version of the method is presented below:

step 1: find T^* , where:

$$T^* = \lfloor \min\{\min_{\forall i}\{(UO_i + UI_i + UD_i)/p_i\}, (O + I + \sum_i UD_i)/\sum_i p_i\} \rfloor \quad (33)$$

step 2: calculate T , where:

$$T = \min\{T^*, Z\} \quad (34)$$

Essa simplificação do método demonstra que o valor ótimo calculado para T não depende da forma como a produção do produto i , P_i , é distribuída entre atendimento à demanda, D_i , outlets, O_i , e estoque de fábrica I_i .

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

find a complete solution.

The algorithm (1) is used for finding an optimal value for T .

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

Require: $UD_i, UO_i, UI_i, p_i, \forall i, O, I, Z$

$T^* \leftarrow \lfloor \min\{\min_{\forall i}\{(UO_i + UI_i + UD_i)/p_i\}, (O + I + \sum_i UD_i)/\sum_i p_i\} \rfloor$

$T \leftarrow \min\{T^*, Z\}$

Return: T .

Then an initial production distribution can be done by the algorithm (2). Esta distribuição é feita sem levar em consideração as restrições definidas para o conjunto de produtos do lote. SO e SI representam, respectivamente, o quantidade total disponível a ser enviada para os outlets e a quantidade total disponível para estoque em fábrica após a distribuição inicial. Caso esse valores sejam negativo, isso indica que a distribuição inicial proposta não é uma distribuição viável.

Assim, 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 $SO < 0$ *then* $SI > 0$ *and* $|SO| < SI$

if $SI < 0$ *then* $SO > 0$ *and* $|SI| < SO$

Therefore it is possible to meet the restrictions by applying the algorithm (3). Neste algoritmo redistribuímos a produção do estoque para os outlets ou vice versa, respectivamente, em função de termos um SI ou SO negativo.

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: $UD_i, UO_i, UI_i, p_i, \forall i, O, I, Z, T$.

for all i **do**

$E_i \leftarrow T * p_i$
 $D_i \leftarrow \min\{UD_i, E_i\}$
 $E_i \leftarrow E_i - D_i$
 $O_i \leftarrow \min\{UO_i, E_i\}$
 $E_i = E_i - O_i$
 $UO_i = UO_i - O_i$
 $I_i = E_i$
 $UI_i = UI_i - I_i$

end for

$SO = O - \sum_i O_i$

$SI = I - \sum_i I_i$

Return: $D_i, O_i, I_i, E_i, UO_i, UI_i, \forall i, SO, SI$.

5. Tests and results

To test the developed model and analytical solution method, we created a solver in C++ named COPSolver (V01_20230814, lib multiproduct-batch-processing-time-maximization-problem) and a solver in LINGO named MBPTM.lng both available at <https://github.com/tbfraga/COPSolver>. 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 random benchmarks generated by the following functions:

$$p_i = \text{rand}() \% 30 + 10 \quad (35)$$

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

Require: $O_i, I_i, UO_i, UI_i, \forall i, SO, SI$.

```

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:  $O_i, I_i, \forall i$ .

```

$$UD_i = \text{rand}() \% 3000 + 800; \quad (36)$$

$$\text{seed1} = \text{rand}() \% 3000 + 500; \quad (37)$$

$$\text{seed2} = \text{rand}() \% 5000 + 1000; \quad (38)$$

$$O = N/2 * \text{seed1}; \quad (39)$$

$$UO_i = \text{rand}() \% (\text{seed1} - 500) + 500; \quad (40)$$

$$I = N/2 * \text{seed2}; \quad (41)$$

$$UI_i = \text{rand}() \% (\text{seed2} - 1000) + 1000; \quad (42)$$

$$Z = 100; \quad (43)$$

Randomly generated benchmarks were named RMBPTM 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

rand((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/COPSSolver.

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

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

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 one-day period problem, so in a future work we will study the complexities of solving a multi-period problem and try to propose new solution methods if needed.

6. Conclusions and suggestions for future works

In this paper we presented the Multi-product 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 planning period of one day, in a future work we will study the complexities that arise when considering the same problem in a multi-period scenario. We will also verify if there is the possibility of extending the proposed analytical method to solve a class of linear integer programming problems with similar characteristics to the studied problem.

7. CRediT authorship contribution statement

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

8. 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.

References

- Eilon. (1985). Multi-product batch production on a single machine - A problem revisited. *OMEGA Int. J. of Mgmt Sci.*, Vol. 13 (5), pp. 453–468.
- Kashan, A. H., 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, pp. 102673.
- Kim, M., Jung, J. H. and Lee, I. (1996). Intelligent scheduling and monitoring for multi-product networked batch processes. *Computers chem. Engn*, Vol. 20 (Suppl.), pp. 1149–1154.
- Li, C., Wang, F., Gupta, J.N.D., Chung, T. (2022). Scheduling identical parallel batch processing machines involving incompatible families with

- different job sizes and capacity constraints. *Computers & Industrial Engineering*, Vol. 169, pp. 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, 124830.
- Méndez, C.A., Henning, G.P., Cerdá, J. (2000). Optimal scheduling of batch plants satisfying multiple product orders with different due-dates. *Computers and Chemical Engineering*, Vol. 24, pp. 2223–2245.
- OMEGA Journal. (1993). Single Machine Multi-product Batch Scheduling: Testing Several Solution Methods. *OMEGA Int. J. of Mgmt Sci.*, Vol. 21 (6), pp. 709–711.
- Ravemark, D. E., and Rippin, D. W. T. (1998). Optimal design of a multi-product batch plant. *Computers chem. Engng*, Vol. 22 (1-2), pp. 177–183.
- Shi, B., Qian, X., Sun, S., Yan, L. (2017). Rule-based scheduling of multi-stage multi-product batch plants with parallel units. *Chinese Journal of Chemical Engineering*, in press.

reference	problem	period	stage	units	int. stor.	constraints	objectives	method
Eilon (1985)	scheduling	single	single	single			minimize total setup; minimize daily holding cost;	
Shi <i>et al.</i> (2017)	scheduling	single	non-identical	parallel	release time	due date		

Table 1: Multi-product batch problems taxonomy

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 2: Benchmark MBPTM 2

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 3: Benchmark MBPTM 3

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

Table 4: Benchmark MBPTM 10

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

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

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 6: Results obtained with the LINGO solver and COPSolver for the MBPTM 3

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

Table 7: Results obtained with the LINGO solver and COPSolver for the MBPTM 10