Graphical Abstract

Multi-period and Multi-product Batch Processing Time Maximization Problem

Tatiana Balbi Fraga

Highlights

Multi-period and Multi-product Batch Processing Time Maximization Problem

Tatiana Balbi Fraga

- ...;
- ...;
- ...;
- ....

Multi-period and Multi-product Batch Processing Time Maximization Problem

Tatiana Balbi Fraga[a,*]

[a]*Agreste Academic Center - Federal University of Pernambuco, Avenida Marielle Franco, Nova Caruaru, Caruaru, 55014-900, PE, Brazil*

Abstract

...

Keywords: multiproduct batch, processing time maximization, C++, LINGO

1. Introduction

T....

As scientific contributions,....

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-period and multi-product batch processing time maximization problem

In MPMPBPTM problem consideramos um horizonte de planejamento com ND dias (ND $\geq$ 1) e uma demanda diária ($\mathrm{UD}_{id} \geq 0$) durante este período para cada produto do lote considerado. Adicionalmente, já existe

*corresponding author

Email address: `tatiana.balbi@ufpe.br` (Tatiana Balbi Fraga)

um plano de produção prévio, de forma que é possível que já exista uma programação anterior de produção para cada produto do lote. Assim a disponibilidade para estocagem e para envios para as lojas de fábrica ficará sujeita a alteração de acordo com o planejamento prévio de produção.

O problema consiste em determinar a tempo máximo de processamento para o lote considerado, levando em conta as restrições de limite consideradas.

A seção a seguir apresenta um modelo matemático para este problema.

3. Mathematical model

Given that:

UD_{id} is the demand for the product i on day d;
PO_{id} is the planned production of the product i on day d;
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;

UI_{id} is the maximum quantity allowed for stocking in the factory the product i on day d;

being UI_{i0} the maximum quantity allowed for stocking in the factory the product i on the (*first*) planning day ($d = 1$);

I_d is the free factory storage of all products on the end of the day d;

being I_0 the initial free inventory for all products on factory on the start of the (*first*) planning day ($d = 1$);

P_i is the amount of product i to be produced on batch;
D_{id} is the amount of product i delivered for the demand on day d;
O_{id} amount of product i shipped to factory outlets;
T is the batch processing time.

We have the problem:

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

s.t.

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

$$P_i + UI_{i1} - D_{i1} - O_{i1} = \mathrm{UI}_{i0} - \mathrm{PO}_{i1} \tag{3}$$

$$UI_{id} - UI_{i(d-1)} - D_{id} - O_{id} = -\mathrm{PO}_{id} \quad \forall i, \forall (d > 1) \tag{4}$$

$$I_1 + \sum_i \{P_i - D_{i1} - O_{i1}\} = \mathrm{I}_0 \tag{5}$$

$$I_d - I_{d-1} + \sum_i \{P_i - D_{id} - O_{id}\} = 0 \quad \forall (d > 1) \tag{6}$$

$$\sum_d O_{id} \leq \mathrm{UO}_i \quad \forall i \tag{7}$$

$$\sum_i \sum_d O_{id} \leq \mathrm{O} \tag{8}$$

$$D_{i1} \leq \mathrm{UD}_{i1} \quad \forall i \tag{9}$$

$$D_{id} \leq \sum_{k=1}^{d} \mathrm{UD}_{ik} - \sum_{k=1}^{d-1} D_{ik} \quad \forall i, \forall d > 1 \tag{10}$$

$$T \leq \mathrm{Z} \tag{11}$$

$$UI_{id}, I_d, O_{id}, D_{id} \in \mathbb{Z}^+ \quad \forall i, d \tag{12}$$

$$I_d \in \mathbb{Z}^+ \quad \forall d \tag{13}$$

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 autlets, 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 autlets 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. And finally, the constraints in (10) inform the nature of the decision variables.

4. Analytical solution

It is possible to consider the factory stock and the outlets stock as single stock, so we have:

$$E_i = O_i + I_i \tag{14}$$

where E_i is the sum of the quantity stored at the factory and the quantity sent to the outlets of the product i.

So that:

$$E_i \leq \mathrm{UO}_i + \mathrm{UI}_i \tag{15}$$

and

$$\sum_i E_i \leq \mathrm{O} + \mathrm{I} \tag{16}$$

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

$$T = T' + T'' \tag{17}$$

and consider that T' is the maximum processing time used only for production that will meet the demand. Thus, we can find T', solving the reduced problem:

$$max \quad T' \tag{18}$$

s.t.

$$D_i - \mathrm{p}_i * T' = 0 \quad \forall i \tag{19}$$

$$D_i \leq \mathrm{UD}_i \quad \forall i \tag{20}$$

$$D_i, T' \in \mathbb{Z}^+ \quad \forall i \tag{21}$$

This reduced problem can be rewritten in the form:

$$max \quad T' \tag{22}$$

s.t.

$$T' \leq \lfloor \mathrm{UD}_i/\mathrm{p}_i \rfloor \quad \forall i \tag{23}$$

So we have that T' will be the smallest of the ratios $\lfloor \mathrm{UD}_i/\mathrm{p}_i \rfloor$ of all products.

Once we find the value of T', we can calculate the value of unmet demand for each product after T', S_i, through the equations in (24).

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

Now we consider that the time interval T'' will be used for the production of the quantities that will be stored (in the factory and in the outlets), as well as of the demand not met by the production in the first time interval, S_i.

In this case, we can find T''' by solving the second reduced problem:

$$max \quad T'' \tag{25}$$

s.t.

$$E_i - \mathrm{p}_i * T'' = 0 \quad \forall i \tag{26}$$

$$E_i - S_i \leq \mathrm{UO}_i + \mathrm{UI}_i \quad \forall i \tag{27}$$

$$\sum_i E_i - S_i \leq \mathrm{O} + \mathrm{I} \tag{28}$$

$$E_i, T'' \in \mathbb{Z}^+ \quad \forall i \tag{29}$$

Again, using a little algebra, we can rewrite this reduced problem in the form:

$$max \quad T'' \tag{30}$$

$s.t.$

$$T'' \leq \lfloor (\mathrm{UO}_i + \mathrm{UI}_i + S_i)/\mathrm{p}_i \rfloor \quad \forall i \tag{31}$$

$$T'' \leq \lfloor (\mathrm{O} + \mathrm{I} + \sum_i S_i)/\sum_i \mathrm{p}_i \rfloor \tag{32}$$

So, being N the number of products in the batch, we will have N + 1 inequalities that limit the value of T'' by constants and again T'' will be defined by the smallest value.

So, the analytical method can be decomposed into 4 steps:

step 1: find T', where:

$$T' = \lfloor \min_{\forall i} \{\mathrm{UD}_i/\mathrm{p}_i\} \rfloor \tag{33}$$

step 2: calculate S_i for all products.

$$S_i = \mathrm{UD}_i - \mathrm{p}_i * T' \quad \forall i \tag{34}$$

step 3: find T'', where:

$$T'' = \lfloor \min\{\min_{\forall i}\{(\mathrm{UO}_i + \mathrm{UI}_i + S_i)/\mathrm{p}_i\}, (\mathrm{O} + \mathrm{I} + \sum_i S_i)/\sum_i \mathrm{p}_i\} \rfloor \tag{35}$$

step 4: calculate T, where:

$$T = T' + T'' \tag{36}$$

$$T = \min\{T, \mathrm{Z}\} \tag{37}$$

It is probably possible to extend this thinking to the solution of a class of linear optimization problems. However, we will work on this concept in an upcoming paper.

Solution for the example presented before

Applying the method for solving the example presented in section 2, we have:

$T' \leq \lfloor 1000 \text{ g}/(60 \text{ g/min}) \rfloor$

$T' \leq 16 \text{ min} \quad \text{for A}$

$T' \leq \lfloor 500 \text{ g}/(40 \text{ g/min}) \rfloor$

$T' \leq 12 \text{ min} \quad \text{for B}$

so:

$T' = 12 \text{ min}$

Then:

$S_A = 1000 \text{ g} - 12 \text{ min} * 60 \text{ g/min} = 280 \text{ g}$

$S_B = 500 \text{ g} - 12 \text{ min} * 40 \text{ g/min} = 20 \text{ g}$

And so we have:

$T'' \leq \lfloor (3000 \text{ g} + 600 \text{ g} + 280 \text{ g})/(60 \text{ g/min}) \rfloor$

$T'' \leq 64 \text{ min} \quad \text{for A}$

$T'' \leq \lfloor (2000 \text{ g} + 600 \text{ g} + 20 \text{ g})/(40 \text{ g/min}) \rfloor$

$T'' \leq 65 \text{ min} \quad \text{for B}$

$T'' \leq \lfloor (1000 \text{ g} + 3000 \text{ g} + 300 \text{ g})/(100 \text{ g/min}) \rfloor$

$T'' \leq 43 \text{ min} \quad \text{for A and B}$

So that:

$T'' = 43$ min

So we have:

$T = 55$ min

As $T <$ Z, then $T = 55$ min will certainly be the optimal solution.

In this case we will have:

$P_A = 3300$ g e $P_B = 2200$ g

$D_A = 1000$ g e $D_B = 500$ g

$E_A = 2300$ g e $E_B = 1700$ g

Note that it is not important to know the values of O_i and I_i, $\forall i$, however, after determining a solution, these values can be found by taking a solution from the undetermined system of inequalities:

$$O_i + I_i = E_i \quad \forall i \tag{38}$$

$$O_i \leq \mathrm{UO}_i \quad \forall i \tag{39}$$

$$\sum_i O_i \leq \mathrm{O} \tag{40}$$

$$I_i \leq \mathrm{UI}_i \quad \forall i \tag{41}$$

$$\sum_i I_i \leq \mathrm{I} \tag{42}$$

$$O_i, I_i \in \mathbb{Z}^+ \quad \forall i \tag{43}$$

5. Tests and results

To test the developed model and analytical solution method, we created a solver in C++ and a solver in LINGO (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 1, 2 and 3, and for randon benchmarks generated by the following functions:

$$\mathrm{p}_i = \mathrm{rand}()\%30 + 10 \tag{44}$$

$$\mathrm{UD}_i = \mathrm{rand}()\%3000 + 800; \tag{45}$$

$$\mathrm{seed1} = \mathrm{rand}()\%3000 + 500; \tag{46}$$

$$\mathrm{seed2} = \mathrm{rand}()\%5000 + 1000; \tag{47}$$

$$\mathrm{O} = \mathrm{N}/2 * \mathrm{seed1}; \tag{48}$$

$$\mathrm{UO}_i = \mathrm{rand}()\%(\mathrm{seed1} - 500) + 500; \tag{49}$$

$$\mathrm{I} = \mathrm{N}/2 * \mathrm{seed2}; \tag{50}$$

$$\mathrm{UI}_i = \mathrm{rand}()\%(\mathrm{seed2} - 1000) + 1000; \tag{51}$$

$$\mathrm{Z} = 100; \tag{52}$$

Randomly generated benchmarks were named RMPBPTMP 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/blinded.

Table 4 presents the results obtained by applying the analytical method (C++ solver) and the LINGO solver for the solution of the previously presented benchmarks.

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 1: Benchmark MPBPTMP 001

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 2: Benchmark MPBPTMP 002

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 analytical solution method for this problem. The mathematical model and analytical

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 3: Benchmark MPBPTMP 003

problem	LINGO solver	time (s)	analytical method	time (s)
MPBPTMP 1	55	0.04	55	0.001
MPBPTMP 2	48	0.06	48	0.001
MPBPTMP 3	30	0.09	30	0.001
RMPBPTMP 20	100	0.08	100	0.001
RMPBPTMP 50	98	0.12	98	0.001
RMPBPTMP 100	98	0.12	98	0.002
RMPBPTMP 1,000	78	3.10	78	0.002
RMPBPTMP 10,000	70	168.87	70	0.006

Table 4: Results obtained with the LINGO solver and the analytical method

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.

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.

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.