Review Article/ Industrial Engineering http://doi.org/10.15446/ing.investig.v42n2.83394

An Efficient Algorithm Applied to Optimized Billing Sequencing

Un algoritmo eficiente aplicado a la secuencia de facturación optimizada

Anderson Rogério Faia Pinto[1] and Marcelo Seido Nagano[2]

ABSTRACT

This paper addresses the Optimized Billing Sequencing (OBS) problem to maximize billing of the order portfolio of a typical Distribution Center (DC). This is a new problem in the literature, and the search for the best billing mix has generated demands for better optimization methods for DCs. Therefore, the objective of this paper is to provide an effective algorithm that presents quick and optimized solutions for higher-complexity OBS levels. This algorithm is called Iterative Greedy Algorithm (IGA-OBS), and its performance is compared to the genetic algorithm (GA-OBS) by Pinto and Nagano. Performance evaluations were carried out after intense computational experiments for problems with different complexity levels. The results demonstrate that the GA-OBS is limited to medium-size instances, whereas the IGA-OBS is better adapted to reality, providing OBS with solutions with satisfactory time and quality. The IGA-OBS enables managers to make decisions in a more agile and consistent way in terms of the trade-off between the level of customer service and the maximization of the financial result of DCs. This paper fills a gap in the literature, makes innovative contributions, and provides suggestions for further research aimed at developing more suitable optimization methods for OBS.

Keywords: iterative greedy algorithm, genetic algorithm, maximized billing, distribution center

RESUMEN

Este documento aborda el problema de la Secuenciación Optimizada de Facturación (OBS) para maximizar la facturación de la cartera de pedidos de un centro de distribución (CD) típico. Este es un nuevo problema en la literatura, y la búsqueda de la mejor combinación de facturación ha exigido mejores métodos para optimizar los CD. Por lo tanto, el objetivo de este artículo es proporcionar un algoritmo eficaz que presente soluciones rápidas y optimizadas para niveles más altos de complejidad OBS. Este algoritmo se denomina Algoritmo Voraz Iterativo (IGA-OBS) y su rendimiento se compara con el del algoritmo genético (GA-OBS) de Pinto y Nagano. Se llevaron a cabo evaluaciones de desempeño después de intensos experimentos computacionales para problemas con diferentes niveles de complejidad. Los resultados demuestran que el GA-OBS se limita a instancias de tamaño medio, mientras que el IGA-OBS se adapta mejor a la realidad brindando soluciones en tiempo y calidad satisfactorios a OBS. El IGA-OBS permite a los gerentes tomar decisiones de una manera mas ágil y consistente frente al *trade-off* entre el nivel de servicio al cliente y la maximización del resultado financiero de los CD. Este artículo llena un vacío en la literatura, aporta contribuciones innovadoras y proporciona sugerencias para futuras investigaciones destinadas a desarrollar métodos de optimización más adecuados para OBS.

Palabras clave: algoritmo voraz iterativo, algoritmo genético, facturación maximizada, centro de distribución

Received: November 6th, 2019
Accepted: July 6th, 2021

Introduction

During the last decades, most companies have started to aim for large production and distribution volumes focusing on reducing lead times and inventory (van den Berg and Zijm, 1999; Richards, 2011; Haq and Boddu, 2017). The majority of customers, according to Pinto and Nagano (2020), have reduced the size of their orders and started to place them in shorter time intervals and minimum amounts of multiple Stock Keeping Units (SKUs) in their Distribution Centers (DCs). This tendency has resulted in shorter order fulfilment deadlines, and, consequently, it has started to demand greater process agility in DCs (Seyedrezaei *et al.*, 2012; Matthews and Visagie, 2013; Marchet *et al.*, 2015). The fact is that there are still no tools that can foresee the exact demand volume for dynamic stochastic environments in an unequivocal way (Seyedrezaei *et al.*, 2012; Sereshti and Bijari, 2013; Baud-Lavigne *et al.*, 2014). The option to maintain minimum levels in uncertain scenarios can cause, at a given billing moment, some SKU restrictions in the DC (Pinto *et al.*, 2018). Additionally, most customers do not accept billings or receiving partial purchases, for example,

[1] Ph.D. in Production Engineering, São Carlos School of Engineering, University of São Paulo, Brazil. Affiliation: Professor and Senior Researcher, Production Engineering Department, University of Araraquara, Brazil. E-mail: anderson@life.com.br

[2] Ph.D. in Mechanical Engineering, São Carlos School of Engineering, University of São Paulo, Brazil. Affiliation: Associate Professor and Senior Researcher, São Carlos School of Engineering, University of São Paulo, Brazil. E-mail: drnagano@usp.br

How to cite: Pinto, A. R. F., and Nagano, M. S. (2022). An Efficient Algorithm Applied to Optimized Billing Sequencing. *Ingeniería e Investigación, 42*(2), e83394. http://doi.org/10.15446/ing.investig.v42n2.83394

Attribution 4.0 International (CC BY 4.0) Share - Adapt

in the e-commerce sector (Rim and Park, 2008). This fault will result in conflicting orders and the need to determine billing and fulfilment rankings for these SKUs (Rim and Park, 2008; Slotnick, 2011; Seyedrezaei *et al.*, 2012; Huang and Ke, 2017; Ledari *et al.*, 2018; Leung *et al.*, 2018; Boysen *et al.*, 2019; Pinto and Nagano, 2020). In this sense, this paper adresses a specific decision-making problem to maximize billing, which is called Optimized Billing Sequencing (OBS). This problem was initially approached by Pinto *et al.* (2018) and refers to the optimization of the billing processes of the order portfolio in a typical DC. In practice, OBS problems are usually very complex, and the pressure to maximize results and meet delivery deadlines demands agility in finding the best solution. However, dealing with a set of rules, restrictions, and decision variables without the help of a suitable quantitative tool becomes a complex task when the objective is to optimize the OBS. Fast decision-making based solely on experience or feeling may lead to waste of time and financial losses in the DC (Pinto *et al.*, 2018). The literature, however, is insufficient and does not provide optimization methods (OMs) that can arrive at time and quality solutions that are satisfactory to all OBS instances. The available research neglects important practical aspects or offers lengthy solutions that constitute limitations for DCs. Therefore, there is a demand for OMs that are more robust and suited for the reality of DCs, which basically consist of achieving two OBS targets: OM robustness and practical application in DCs. Thus, this paper focusses on adapting to real world demands in order to deal with practical dilemmas not yet addressed by the OMs proposed for OBS. The entire OBS configuration under study is the same as the one considered by Pinto and Nagano (2020). In this sense, the amount of available inventory for billing is always controlled at minimum levels based on SKU demands. There are therefore some uncertainties regarding the management of the demand, which is stochastic, and billings occur based on Variable Time Windows (VTW). Most delivery deadlines are tighter, and there is a high frequency of small orders containing minimum amounts of multiple SKUs. Billing decisions prioritize order fulfilment and payment dates by the Earliest Due Date (EDD) rule. The proposed approach aims to enable managers to make decisions in a more agile and consistent way regarding the trade-off between the level of customer service and the financial result maximization of the aforementioned DCs. Thus, the objective of this paper is to provide an efficient billing maximization algorithm that, in an agile and consistent manner, produces optimized solutions for higher levels of OBS complexity. This algorithm is called Iterative Greedy Algorithm (IGA-OBS), and its performance is compared to the genetic algorithm (GA-OBS) by Pinto and Nagano (2020). In technical terms, the latter is an extension and improvement of the first GA-OBS that was proposed in the literature by Pinto *et al.* (2018), whereas, in methodological terms, this is a quantitative research based on mathematical modeling and computational simulation. Performance evaluations of the IGA-OBS in this work are carried out by means of intense computational experiments for a set of problems with different OBS complexity levels. We have focused our attention on the potential of IGA-OBS/GA-OBS for practical effectiveness and their capacity to adapt to the reality of current DCs. This paper is structured as follows: section 2 explains the OBS problem; section 3 presents the Literature review; section 4 expresses the model formulation of the OBS; section 5 presents the IGA-OBS; section 6 demonstrates the GA-OBS; section 7 brings the computational experiments and the performance assessments of the GA-OBS and the IGA-OBS; finally, the last section states the final considerations and the main suggestions for future studies and approaches to the OBS.

OBS problem

This section presents the OBS problem to maximize the billing of a typical DC. In this OBS, there are uncertainties regarding the management of the demand, which is stochastic, and SKU inventories available for billing are controlled at minimum levels in the DC. It is common for customers to place more than one order simultaneously, which constitutes a dynamic (online) input in the order portfolio regardless of SKU availability. These orders may have varied sizes and different quantities, or distinct unit selling prices for multiple combinations of different SKUs or of the same SKU. Most customers demand tighter delivery deadlines for a set of orders with multiple fulfilment and payment dates for a given VTW. Billing sequences are determined by analyzing the best combinations between fulfilment and payment dates, which are always prioritized by the EDD rule. All billings are generated after a certain number of orders accumulates in the order portfolio, which also occurs within time intervals pre-set by the VTW. Demands with partial inventory restrictions are billed according to costumer approval, and those referring to total restrictions are billed when the SKUs are available. Every order that is not billed due to SKU restrictions is transferred to the following VTW until the quantities of the mentioned SKUs are available in the DC. Therefore, the OBS problem is caused by restrictions or management failures resulting from the dynamics of changes, uncertainties, and disorders, in addition the pressing emergency in the reality of current DCs (Pinto *et al.*, 2018). Decisions are usually made based on fulfillment rankings pre-defined by internal policies, which include a set of rules, constraints, and decision variables inherent to the OBS. In the literature, the mechanism to determine which SKUs are billed for each order was classified by Pinto *et al.* (2018) as a variation of the Knapsack Problem (KP). Thus, the OBS may be reduced to a decision-making problem, for which the ideal solution is to maximize the total billing of the order portfolio (Pinto and Nagano, 2020).

Literature Review

The available literature shows the research by Pinto *et al.* (2018) to be the first to approach and propose an OM for a specific problem of the so-called OBS. This OM is formulated through a hybrid GA, whose structure is based on the canonic GA by Holland (1975) and programmed in Visual Basic for Applications (VBA) from Microsoft Office Excel 2013. The hybrid GA is called GA-OBS, and it is formulated by means of binary genetic structures that use an elitist selection and an aptitude function guided by penalties and repairs of individuals that are unfeasible to the OBS. This GA-OBS can deal with inventory restrictions and with customer acceptance criteria regarding billings of partial amounts of SKUs to attribute them in an optimized manner to the order portfolio demand in compliance with the First Come, First Served (FCFS) rule. Experiments demonstrated that the proposed GA-OBS provides solutions that optimize billing and expedite decision-making processes for the OBS.

More recently, an important innovation that aims to provide approaches to OBS that are better adapted to real-world needs was proposed by Pinto and Nagano (2020). This approach proposes an extension and enhancement of the OBS by Pinto *et al*. (2018), along with the Optimized Picking Sequence (OPS) by Pinto and Nagano (2019). These problems refer to the optimization of billing and manual picking processes, respectively, in a typical parts Warehouse (WA). The WA in question operates with a picking system that fits into the low-level picker-to-parts category with pick-and-sort process, and it has only one area known as Pick-up and Drop-off (P/D). The research objective was to provide a OM that integrated and offered optimized solutions for OBS/OPS in order to better deal with the trade-off between the level of customer service and the efficiency of said WA. The proposed OM was formulated by integrating two GAs called GA-OBS and GA-OPS. GA-OBS deals with inventory restrictions and possible partial billings, maximizes the total order portfolio billing by prioritizing the fulfilment and payment dates in compliance with the Earliest Due Date (EDD) rule, and generates a picking order for the GA-OPS. In the sequence, GA-OPS, which comprises the iteration of batch (GA_{BATCH}) and routing (GA_{TSP}) algorithms to satisfy all specificities of the problem and to minimize picking total time and cost for the OPS. Programming was done in Python, and both data inputs/outputs and results analyses were supported by Microsoft Office Excel 2016. Experiments with problems with different complexity levels showed that the proposed tool produces solutions of satisfactory quality and speeds up decision-making and operational processes so as to optimize financial results and productivity of the WAs.

Evidently, GA applications stand out for their robustness, implementation, and hybridization flexibility with other OMs (Gen *et al*., 2008; Bottani *et al*., 2012). However, Pinto *et al*. (2018) applied the GA-OBS to solve only one single-size problem, and they did not include large OBS instances. The objective of the authors' approach was to gain insights into the best population size configuration and number of generations linked to variations in the genetic operator parameters that can maximize the solution potential of the GA-OBS. Therefore, the authors themselves recognize the need for tests in problems of higher instances, as well as the implementation of other parameters, operators, and genetic representations or evolutionary designs that can improve performance. In Pinto and Nagano (2020), the proposal was to address the integration of problems by considering a series of practical dilemmas present in WAs. The solutions are of satisfactory quality for different instances and complexity levels configured for the OBS/OPS. However, solutions for problems in large instances demand considerable efforts in terms of checking and repairing chromosomes to be performed by GAs. These occurrences may result in an exponential increase of computational processing time and make the OM inefficient for some WAs.

Similar approaches to the OBS that presuppose inventory restrictions and uncertainties associated to the demand forecast were proposed by Rim and Park (2008) and Seyedrezaei *et al*. (2012). Rim and Park (2008) used the entire binary Linear Programming (LP) to deal with inventory restrictions in order to fulfil DC orders, aiming to maximize the Order Fill Rate (OFR). SKUs are only attributed to orders if there is available inventory in the DC; if there is not, orders are transferred to the next day and fulfilled according to SKU availability and priority rules to avoid excessive delays to the OFR. Experiments demonstrated that LP is better than the simple models in terms of order size and/or number of SKUs when compared to the FCFS rule. Seyedrezaei *et al.* (2012) applied the GA to a NP-complete inventory forecast and demand problem to plan and maximize the number of orders fulfilled according to Customer Importance, SKU Useful Life, and Back-Orders. This GA calculates the demand coefficient of each customer for a given time period and defines an inventory considering the DC's capacity and the useful life of the SKUs. Hence, orders that are not fulfilled due to SKU restrictions are transported to the next period (back-order). Compared to the Lingo software, the GA arrived at solutions with higher quality and satisfactory time to better manage DCs. In the search for more robust strategies and optimization methods, advanced technologies for intelligent decision making in manufacturing and logistics are presented by Chien *et al*. (2012). Other approaches focused on producing solutions that can maximize costs and/or maximize order fulfilment profit in an agile and flexible manner are found in high-impact journals (Ghiami *et al*., 2013; Mousavi *et al*., 2013; Bandyopadhyay and Bhattacharya, 2014; Diabat, 2014; Park and Kyung, 2014; Diabat and Deskoores, 2016; Kumar *et al*., 2016; Mousavi *et al*., 2017; İnkaya and Akansel, 2017).

Model formulation

In the OBS, the SKU notation refers to the registration number that distinguishes the n product types available in the DC stock. The quantity of each SKU in stock at a given t moment of the VTW is given by x_i, and it is represented by the set $X = \{x_1, x_2, ..., x_n\}$, in which the subscript $i = (1, 2 ..., n)$ denotes the i-th SKU. The Purchase Order Portfolio (POP) comprises n orders, represented by $POP = \{O_1, O_2, ..., O_n\}$, and the subscript j refers to the j-th order $\forall j = (1, 2, ..., n)$. These orders are from a set of m customers, represented by $CG = \{C_1, C_2, ..., C_m\}$, in which the subscript $a = (1, 2, ..., m)$ denotes the a-th customer (C_α) of the POP. The total demand of x_i in the POP is given by Q_i, whereas the demand of x_i in O_j is given by q_{ij}, being $O_j = \{q_{1j}, q_{2j}, ..., q_{nj}\}$ $\forall i = (1, 2 ..., n)$ attributes of C_α. If TV is the Total Value of the POP at the t instant of billing in the VTW, then, the TV will only be obtained if $x_i \geq Q_i$, in which y_i is the restriction of x_i and $w_i = (Q_i - y_i)$ is the availability of x_i in case $y_i > 0$. Thus, the notation $C_\alpha^{w_{ijYes}}$ determines that the customer accepts w_i billing to O_j, whereas $C_\alpha^{w_{ijNo}}$determines that the customer does not accept w_i billing to O_j. The insertion date of q_{ij} in the POP is denoted by id_{ij}, and, subsequently, the pre-defined priority criteria for the OBS are obtained: i) fd_{ij} – order fulfilment date of i in O_j; ii) pd_{ij} – order payment date of i in O_j and; iii) pr_{ij} – unit selling price of i in O_j. The entire billing process of q_{ij} is carried out by comparing Q_i to the offer of x_i, so that, every non-billed q_{ij} will be transferred to t_{+1} to be processed again by the next VTW. Up next are the indexes, parameters and restrictions, as well as the decision criteria and variables that configure the OBS optimization problem:

- **Indexes**

  i : denotes the i-th SKU of the n SKUs of the DC;
  j : denotes the j-th order of the n orders of the *POP*;
  α : denotes the a-th customer of the j-th order of *POP*.

- **Parameters and restrictions**

  VTW: Variable Time Windows;
  t : VTW billing moment;
  O_j : refers to the j-th order of the POP in t;
  C_a : refers to the a-th customer of the POP in t;
  q_{ij} : SKU demand in the O_j of the POP in t;
  Q_i : total demand of a SKU of the POP in t;
  x_i : total number of a SKU in the DC in t;
  y_i : restriction of x_i in the DC regarding the Q_i of the POP in t;
  w_i : partial availability of x_i $(Q_i - y_i)$ in the DC in case $y_i > 0$ in t.

- **Decision criteria and variables**

  pr_{ij} : SKU unit price in the O_j;
  fd_{ij} : SKU fulifilment date in the O_j;
  pd_{ij} : SKU payment date in the O_j;
  $C_\alpha^{w_{ijYes}}$: determines if C_a accepts billings of w_i to O_j;
  $C_\alpha^{w_{ijNo}}$: determines if C_a does not accept billings of w_i to O_j.

OBS optimization is subjected to the calculation of the possible Maximum Billing (MB) that can be obtained from the inventory of each SKU available in the DC versus the Q_i of the POP at a given t moment of the VTW. The calculation of the MB is then used to check the need for execution and of an IGA-OBS/GA-OBS search parameter to optimize the OBS. In cases where the MB < TV, *i.e.* if $x_i < Q_i$, so the MB will be the main parameter of the best possible solution for the OBS. The calculation criterion to obtain the MB, as demonstrated by Equation (1), prioritizes the highest pr_{ij} according to the following parameters: i) b_{wij} – billing value of w_{ij} and; ii) b_{qij} – billing value of q_{ij}.

$$\text{if } \; Q_i > x_i \rightarrow pr_{ij} \times w_{ij} = b_{w_{ij}} \; \text{ or if } \; Q_i \leq x_i \rightarrow pr_{ij} \times q_{ij} = b_{q_{ij}} \quad (1)$$

Then, *MB* can be obtained according to Equation (2). In the sequence, Table 1 demonstrates a calculation example of the MB for a given POP. In this example, we presuppose that the CD's inventory volume is represented by $X = \{3_a, 5_b, 3_c, 5_d, 0_e, 1_f, 0_g \text{ and } 5_h\}$.

$$MB = \sum_{j=1}^{O_j} \sum_{i=1}^{n} q_{ij} \times pr_{ij} + \sum_{j=1}^{O_j} \sum_{i=1}^{n} w_{ij} \times pr_{ij} \quad (2)$$

Table 1 shows that, given the availability of x_i, and prioritizing only the highest pr_{ij} according to b_{wij} and b_{qij}, the algorithm found the best billing mix, *i.e.*, MB = 1 840,00, then, MB < TV ($x_i < Q_i$). For example, for "d", despite $fd_{d200} < fd_{d250}$, the algorithm prioritized billing for O_{250}, given that, $pr_{d250} > pr_{d200}$ in POP. Note that the MB does not yet consider all the criteria and decision variables inherent to the OBS. Therefore, Total

Table 1. Maximum billing example

Number Order (O_j)	Code Customer (C_α)	Product Description (SKU)	Total SKU (Q_i)	Price Unit (pr_{ij})	Billing Order (b_{ij})	Order Date (id_{ij})	Fulfilment Date (fd_{ij})	Payment Date (pd_{ij})	Accepts partial q_i (w_{ij})	Stock Attribution (x_i)	Maximum Billing (*MB*)
100	10	*a*	3	50,00	150,00	10/05/2021	10/06/2021	10/07/2021	Yes	**2**	**100,00**
100	10	*c*	2	150,00	300,00	10/05/2021	10/06/2021	10/07/2021	Yes	**2**	**300,00**
150	15	*c*	1	155,00	155,00	15/05/2021	20/06/2021	20/07/2021	No	**1**	**155,00**
200	20	*b*	2	100,00	200,00	15/05/2021	10/06/2021	10/07/2021	No	**2**	**200,00**
200	30	*c*	2	150,00	300,00	15/05/2021	10/06/2021	10/07/2021	No	**-**	**-**
200	30	*d*	4	200,00	800,00	15/05/2021	10/06/2021	10/07/2021	No	**3**	**600,00**
250	10	*d*	2	212,50	425,00	18/05/2021	15/06/2021	10/07/2021	No	**2**	**425,00**
300	30	*a*	1	60,00	60,00	18/05/2021	10/06/2021	10/07/2021	No	**1**	**60,00**
300	30	*e*	2	25,00	50,00	18/05/2021	10/06/2021	10/07/2021	No	**-**	**-**
Total Value (*TV*)			**19**		**2 440,00**					**13**	**1 840,00**

Source: Authors

Billing (TB) maximization of the POP can be expressed as a mathematical programming model to maximize TB_{max}:

$$\text{Maximize } TB_{\max} = \sum_{j=1}^{O_j}\sum_{i=1}^{n} q_{ij} \times pr_{ij} + \sum_{j=1}^{O_j}\sum_{i=1}^{n} w_{ij} \times pr_{ij} \quad (3)$$

Subjected to:

$$x_i > 0 \qquad i = 1, 2, \ldots, n \quad (4)$$

$$x_i \geq q_{ij} \quad \forall C_{\alpha}^{w_{ijNo}} \qquad j = 1, 2, \ldots, n \quad (5)$$

$$w_{ij} > 0 \quad \forall C_{\alpha}^{w_{ijYes}} \qquad \alpha = 1, 2, \ldots, m \quad (6)$$

O_n: Fulfilment by priority fd_{ij}, in POP (7)
O_n: Fulfilment by precedent pd_{ij} in POP (8)
O_n: Fulfilment from highest to lowest pr_{ij} in POP (9)

The objective function in Equation (3) is to find the maximum possible billing of the POP. Restriction (4) ensures that q_i will only be attributed to O_j if, in a t given moment of the VTW, the variable $x_i > 0$ in the DC. Restriction (5) ensures that, $\forall C_{\alpha}^{w_{ijNo}}$, the total demand of q_{ij} can only be attributed to its corresponding O_j if, at a t given moment of VTW, the variable $x_i \geq q_{ij}$ in the DC. Restriction (6) will make sure that a given w_i can only be attributed to O_j if $w_i > 0$ in the POP and $C_{\alpha}^{w_{ijYes}}$. Variables (7), (8), and (9) determine that the rules for fd_{ij}, pd_{ij}, and pr_{ij} are satisfied in the POP.

Iterative Greedy Algorithm (IGA-OBS)

The logic of the IGA-OBS is based on the verification and attribution technique through the iteration of a set of interdependent elements which configure the OBS. The first phase of its formulation is the sorting mechanism of the POP by the fulfilment priority levels defined by $fd_{ij}/pd_{ij}/pr_{ij}$. It is assumed that fd_{ij} and pd_{ij} have respectively higher priority levels than the pr_{ij} of the POP. Hence, q_{ij} and w_{ij} are attributed by automatically comparing and updating inventory balances after each SKU attribution. Therefore, q_i or w_i is attributed to O_j by the variable s_{ij}. Thus, $s_{ij} = q_i$ or w_i; otherwise, $s_{ij} = 0$ to obtain the TB maximization according to Equation (10).

$$TB_{\max} = \sum_{j=1}^{O_j}\sum_{i=1}^{n} q_{ij} \times pr_{ij} + \sum_{j=1}^{O_j}\sum_{i=1}^{n} w_{ij} \times pr_{ij} \quad (10)$$

Thus, q_i or w_i may or may not be attributed to the j-th depending on parameters and restrictions, decision criteria, and variables inherent to the OBS. The attribution routine verifies whether there is a balance of x_i in the DC, and, after each execution, the IGA-OBS produces a solution that maximizes the TB of the POP. q_i or w_i to O_j is attributed according to the following conditions:

1. If $x_i = 0$, $s_{ij} = 0$ according to Equation (11).

$$\text{if } x_i = 0 \forall q_{ij} \rightarrow s_{ij} = 0 \quad (11)$$

2. If $x_i = w_{ij}$ in case $C_{\alpha}^{w_{ijNo}}$, $s_{ij} = 0$ according to Equation (12).

$$\text{if } x_i = w_{ij} \leftrightarrow C_a^{w_{ijNo}} \rightarrow s_{ij} = 0 \quad (12)$$

3. If $fd_{t+1} < fd_t$, fd_{t+1} is prioritized over fd_t according to Equation (13).

$$\text{if } x_i = w_{ij} \forall C_a^{w_{ijYes}} \leftrightarrow O_j^{fd_{t+1}} < O_j^{fd_t} \rightarrow pr_{ij}^{fd_{t+1}} \times w_{ij} = b_{q_{ij}} \quad (13)$$

4. If $pd_{t+1} < pd_t$, pd_{t+1} has preference over pd_t according to Equation (14).

$$\text{if } x_i = w_{ij} \forall C_a^{w_{ijYes}} \wedge O_j^{fd_t} = O_j^{fd_{t+1}} \leftrightarrow O_j^{pd_{t+1}} < O_j^{pd_t} \rightarrow pr_{ij}^{pd_{t+1}} \times w_{ij} = b_{q_{ij}} \quad (14)$$

5. If $pr_{ij+1} > pr_{ij}$, pr_{ij+1} is prioritized over pr_{ij} according to Equation (15).

$$\text{if } x_i = w_{ij} \forall C_a^{w_{ijYes}} \wedge O_j^{fd_t} = O_j^{pd_t} \leftrightarrow O_j^{pr_{ij+1}} > O_j^{pr_{ij}} \rightarrow pr_{ij}^{pr_{ij+1}} \times w_{ij} = b_{q_{ij}} \quad (15)$$

To exemplify the solution of the problem given by Table 1, Table 2 shows how the IGA-OBS performs both the POP ordering and the assignments of x_i to maximize TB. The following colors are used to demonstrate the attributions of q_{ij} and w_{ij} in order to exemplify the calculation that maximizes TB: i) *black*: it refers to the total attributions of q_{ij}; ii) *green*: it corresponds to the attributions of w_{ij}; and iii) *red*: it indicates the q_{ij} and w_{ij} that were not attributed due to the total restrictions of a given SKU.

On Table 2, it can be observed that, for "α", O_{100} and O_{300} have identical fd_{ij} and pd_{ij}. However, $pr_{a300} > pr_{a100}$ and, if O_{100} has $C_{\alpha}^{w_{ijYes}}$ as a criterion for billings of w_{ij}, $s_{i100} = 2\alpha$ and $s_{i300} = 1\alpha$. Note that for "c", O_{150}, even having the highest pr_c among the orders, is discarded because fd_{150} is higher than fd_{100} and fd_{200}. In case of "d", there is insufficient inventory of x_d to satisfy O_{200} and O_{250}, and both have $C_{\alpha}^{w_{ijNo}}$ as a criterion for billings of w_{ij}. Thus, if $fd_{200} < fd_{250}$, the option is to satisfy O_{200}. As for "e", there were no attributions as $x_d = 0$, that is, the entire IGA-OBS solution logic satisfies all demands inherent to the OBS to maximize the TB. Table 3 presents the list of q_{ij} to be billed, whereas the q_{ij} that will not be billed is represented by $y = \{1_a, 2_c, 2_d \text{ and } 2_e\}$ due to y_i restrictions. In sequence, Algorithm 1 shows the IGA-OBS pseudocode

Genetic algorithm (GA-OBS)

In the evolutive genetic structure of the GA-OBS, the representation of chromosome (C) is given by a binary string {0,1}, which attributes q_i or w_i to O_j by variable $s_{ij} = 1$; otherwise, $s_{ij} = 0$. Then, a C is divided into O_n genes, and q_{ij} or w_{ij} is an allele of the j-th gene according to Figure 1.

Table 2. IGA-OBS solution example

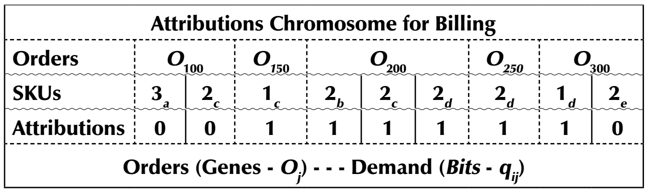
Number Order (O_j)	Code Customer (C_a)	Description Product (SKU)	Total SKU (Q_i)	Price Unit (pr_{ij})	Billing Order (b_{ij})	Order Date (id_{ij})	Fulfilment Date (fd_{ij})	Payment Date (pd_{ij})	Accepts partial q_i (w_{ij})	Stock Attribution (q_{ij} and w_{ij})	Total Billing (*TB*)
100	10	*a*	3	50,00	150,00	10/05/2021	10/06/2021	10/07/2021	Yes	**2**	**100,00**
300	30	*a*	1	60,00	60,00	18/05/2021	10/06/2021	10/07/2021	No	**1**	**60,00**
200	20	*b*	2	100,00	200,00	15/05/2021	10/06/2021	10/07/2021	No	**2**	**200,00**
100	10	*c*	2	150,00	300,00	10/05/2021	10/06/2021	10/07/2021	Yes	**1**	**150,00**
200	30	*c*	2	150,00	300,00	15/05/2021	10/06/2021	10/07/2021	No	**2**	**300,00**
150	15	*c*	1	155,00	155,00	15/05/2021	20/06/2021	20/07/2021	No	**0**	**-**
200	30	*d*	4	200,00	800,00	15/05/2021	10/06/2021	10/07/2021	No	**4**	**800,00**
250	10	*d*	2	212,50	425,00	18/05/2021	15/06/2021	10/07/2021	No	**0**	**-**
300	30	*e*	2	25,00	50,00	18/05/2021	10/06/2021	10/07/2021	No	**0**	**-**
Total Value (*TV*)			**19**		**2 440.00**					**12**	**1 610,00**

Source: Authors

Table 3. Billing list

Number Order (O_j)	Code Customer (C_a)	Product Description (SKU)	Total SKU (Q_i)	Price Unit (pr_{ij})	Billing Order (b_{ij})	Order Date (id_{ij})	Fulfilment Date (fd_{ij})	Payment Date (pd_{ij})	Accepts partial q_i (w_{ij})	Stock (x_i) Attribution (q_{ij} and w_{ij})	Total Billing (TB)
100	10	*a*	2	50,00	100,00	10/05/2019	10/06/2019	10/07/2019	Yes	**2**	**100,00**
100	10	*c*	1	150,00	150,00	10/05/2019	10/06/2019	10/07/2019	Yes	**1**	**150,00**
200	20	*b*	2	100,00	200,00	15/05/2019	10/06/2019	10/07/2019	No	**2**	**200,00**
200	30	*c*	2	150,00	300,00	15/05/2019	10/06/2019	10/07/2019	No	**2**	**300,00**
200	30	*d*	4	200,00	800,00	15/05/2019	10/06/2019	10/07/2019	No	**4**	**800,00**
300	30	*a*	1	60,00	60,00	18/05/2019	10/06/2019	10/07/2019	No	**1**	**60,00**
Total Value (*TV*)			**12**		**1 610,00**					**12**	**1 610,00**

Source: Authors



Attributions Chromosome for Billing									
Orders	O_{100}		O_{150}	O_{200}			O_{250}	O_{300}	
SKUs	3_a	2_c	1_c	2_b	2_c	2_d	2_d	1_d	2_e
Attributions	0	0	1	1	1	1	1	1	0
Orders (Genes - O_j) - - - Demand (*Bits* - q_{ij})									

Figure 1. Assignment chromosome for billing (GA-OBS)
Source: Authors

```
procedure: IGA-OBS // attribution iteration
input: problem data and IGA-OBS parameters
output: best solution
beginning
  sort out POP
     SKU: registration number of $i$ in POP
     $fd_j$ : fulfillment date of $i$ in $O_j$
     $pd_{ij}$ : payment date of $i$ in $O_j$
      sort out SKU:
    $pr_{ij}$ : unit selling price of $i$ in $O_j$
        for each $q_{ij}$ check:
           if $x_i > 0$ e $x_i \geq q_{ij}$ bill ← update inventory ($x_i$) and check next SKU;
if $x_i > 0$ e $q_{ij} > x_i$ and $C_\alpha^{w_{ijYes}}$ bill $w_i$ ← update $x_i$ and check next SKU;
if $x_i = 0$ ← do not bill and check next SKU
if $x_i > 0$ e $q_{ij} > x_i$ and $C_\alpha^{w_{ijNo}}$ ← do not bill and check next SKU;
if $pr_t < pr_{t+1}$ ← bill $pr_{t+1}$ ← update $x_i$ and check next SKU;
      output: the best solution;
end
```

Algorithm 1. General implementation structure (IGA-OBS)
Source: Authors

The population is a matrix denoted by N_{pop}, with N_{bits} being the number of bits of C, and N_{ger} the total of generations in the execution of the GA-OBS (Haupt, R. and Haupt, S., 2004). The initial N_{pop} is generated randomly (Man *et al.*, 1996), *i.e.*, if $s_{ij} < 0{,}5$, $s_{ij} = 0$; otherwise, $s_{ij} = 1$, and N_{bits} is equal to the number of POP lines. Therefore, the Billing Obtained (BO) by C is given by Equation (16).

$$BO = \sum_{j=1}^{O_j}\sum_{i=1}^{N_{bits}} s_{ij} \times q_{ij} \times pr_{ij} + \sum_{j=1}^{O_j}\sum_{i=1}^{N_{bits}} s_{ij} \times w_{ij} \times pr_{ij} \qquad s_{ij} = \{0,1\} \tag{16}$$

The fitness function ($F_{fitness}$) first penalizes every C that is not feasible to the OBS by assigning a negative value equal to the b_{qij} of the invalid bit, in which $\forall s_{ij} = 1$ according to the following conditions:If $s_{ij} = 1$ and $x_i = 0$: penalty Pe_x is imposed according to Equation (17).

$$Pe_{x_{ij}} = \sum_{j=i}^{N_{bits}} b_{q_{ij}} \text{ if } s_{ij} = 1 \forall x_i = 0 \rightarrow pr_{ij} \times q_{ij} = b_{q_{ij}} \tag{17}$$

1. If $s_{ij} = 1$ to w_{ij} in case $C_\alpha^{w_{ijNo}}$: penalty Pe_w is imposed according to Equation (18).

$$Pe_{w_{ij}} = \sum_{j=i}^{N_{bits}} b_{q_{ij}} \text{ if } s_{ij} = 1 \forall w_{ij} \leftrightarrow C_a^{w_{ij}No} \rightarrow pr_{ij} \times w_{ij} = b_{q_{ij}} \tag{18}$$

2. The s_{ij} = 1 must prioritize the first fd_{ij} in the POP: If x_i < Q_i, fd_t is preferred over fd_{t+1}; otherwise, penalty Pe_{fd} is imposed according to Equation (19).

$$Pe_{fd_{ij}} = \sum_{j=i}^{N_{bits}} b_{q_{ij}} \text{ if } x_i < Q_i \leftrightarrow s_{ij} = 1 \forall O_j^{fd_{t+1}} \wedge s_{ij} = 0 \forall O_j^{fd_t} \rightarrow pr_{ij}^{fd_{t+1}} \times q_{ij} = b_{q_{ij}} \quad (19)$$

3. The s_{ij} = 1 must prioritize the first pd_{ij} in the POP: after checking the fd_{ij}, if pd_{t+1} < pd_t, pd_{t+1} is prioritized; otherwise, penalty Pe_{pd} is imposed according to Equation (20).

$$Pe_{pd_{ij}} = \sum_{j=i}^{N_{bits}} b_{q_{ij}} \text{ if } O_j^{fd_t} = O_j^{fd_{t+1}} \wedge s_{ij} = 1 \forall O_j^{pd_t} \leftrightarrow O_j^{pd_{t+1}} < O_j^{pd_t} \rightarrow pr_{ij}^{pd_t} \times q_{ij} = b_{q_{ij}} \quad (20)$$

4. The s_{ij} = 1 must prioritize the highest pr_{ij} in the POP: after checking the fd_{ij} and the pd_{ij}, if pr_{ij+1} > pr_{ij}, pr_{ij+1} is prioritized; otherwise, penalty Pe_{pr} is imposed according to Equation (21).

$$Pe_{pr_{ij}} = \sum_{j=i}^{N_{bits}} b_{q_{ij}} \text{ if } O_j^{pd_t} = O_j^{pd_{t+1}} \wedge s_{ij} = 1 \forall O_j^{pr_t} \leftrightarrow O_j^{pr_{t+1}} > O_j^{pr_t} \rightarrow pr_{ij}^{pr_t} \times q_{ij} = b_{q_{ij}} \quad (21)$$

Next, $F_{fitness}$ makes repairs by swapping "l" for "0" in bits with incidence of Pe_x and Pe_w by means of Re_x and Re_w repairs according to Equations (22) and (23).

$$Re_{x_{ij}} = Pe_{x_{ij}} \leftrightarrow s_{ij} = 1 \forall x_i = 0 \rightarrow s_{ij} = 0 \quad (22)$$

$$\text{Re}_{w_{ij}} = Pe_{w_{ij}} \leftrightarrow s_{ij} = 1 \forall w_{ij} \leftrightarrow C_a^{w_{ij}No} \rightarrow s_{ij} = 0 \quad (23)$$

Therefore, b_{qij} and b_{wij} for N_{bits} repaired are \$0,00, and \$1,00 is attributed to $F_{fitness}$ in case $F_{fitness}$ ≤ 1, *i.e.*, $F_{fitness}$ may vary from \$0,00 to BO according to Equation (24).

$$F_{fitness} = \begin{cases} BO - \sum_{j=1}^{N_{bits}} (Pe_{fd_{ij}} + Pe_{pd_{ij}} + Pe_{pr_{ij}}) \\ \text{if } BO < \$1,00 \rightarrow F_{fitness} = \$1,00 \end{cases} \quad (24)$$

The roulette wheel by Holland (1975), linked to Elitism (E) by De Jong (1988), is used as a selection technique, in which only the best C (C_{Elite}) of each N_{ger} is transferred to become the first C of N_{ger+1}. The selection probability of each individual i is equivalent to a certain slice of the roulette wheel, as expressed by Equation (25).

$$p_{s_i} = \frac{F_{fitness\,i}}{\sum_{i=1}^{N_{pop}} F_{fitness\,i}} \quad (25)$$

To exemplify this, Table 4 shows the calculus of the selection probability for four individuals. Then, the select graphic by the roulette wheel with elitism is demonstrated by Figure 2.

Table 4. Evaluation calculus and selection percentage (GA-OBS)

Individual (*String*)	Fitness Function	Selection Percentage	Piece of Roulette
Chromosome (C_1)	1 360,00	39%	138,70
Chromosome (C_2)	1 250,00	35%	127,48
Chromosome (C_3)	460,00	13%	46,91
Chromosome (C_4)	460,00	13%	46,91
Total Population	**3 530,00**	**100%**	**360,00**

Source: Authors

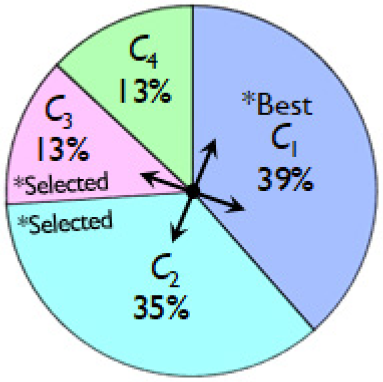
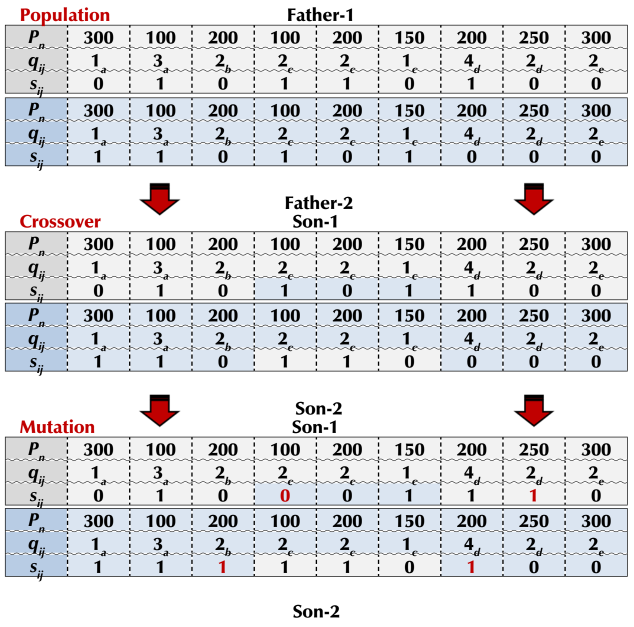


Figure 2. Graphic of the roulette wheel with elitism (GA-OBS)
Source: Authors

The implemented crossover operator is of the two-point kind, and the mutation is an adaptation of the flip type, both by Holland (1975). Figure 3 illustrates the crossover and mutation diagram implemented to GA-OBS.



P_n	300	100	200	100	200	150	200	250	300
q_{ij}	1_a	3_a	2_b	2_c	2_c	1_c	4_d	2_d	2_e
s_{ij}	0	1	0	1	1	0	1	0	0
P_n	300	100	200	100	200	150	200	250	300
q_{ij}	1_a	3_a	2_b	2_c	2_c	1_c	4_d	2_d	2_e
s_{ij}	1	1	0	1	0	1	0	0	0

P_n	300	100	200	100	200	150	200	250	300
q_{ij}	1_a	3_a	2_b	2_c	2_c	1_c	4_d	2_d	2_e
s_{ij}	0	1	0	1	0	1	1	0	0
P_n	300	100	200	100	200	150	200	250	300
q_{ij}	1_a	3_a	2_b	2_c	2_c	1_c	4_d	2_d	2_e
s_{ij}	1	1	0	1	1	0	0	0	0

P_n	300	100	200	100	200	150	200	250	300
q_{ij}	1_a	3_a	2_b	2_c	2_c	1_c	4_d	2_d	2_e
s_{ij}	0	1	0	0	0	1	1	1	0
P_n	300	100	200	100	200	150	200	250	300
q_{ij}	1_a	3_a	2_b	2_c	2_c	1_c	4_d	2_d	2_e
s_{ij}	1	1	1	1	1	0	1	0	0

Figure 3. Crossover and mutation diagram (GA-OBS)
Source: Authors

Figure 3 shows that the crossover is applied to the "Parents" according to the crossover probability (p_c) pre-defined for the GA-OBS. In the crossover, each O_j and q_{ij} remain static, while permutations occur only for s_{ij} in order to form the "Sons" of N_{ger+1}. Figure 3 also shows that the mutation affects the "Sons" resulting from the crossover, and it will be applied according to the mutation probability (p_m), whereas t_x is the rate for the exchange between the N_{bits} of C. Therefore, each bit to be mutated is randomly chosen and receives a value corresponding to the exchange between 0 and 1 for the s_{ij} of C. Note that the mutated bits (red) make changes in the genetic pattern of the "Sons".

Thereupon, p_c, p_m, and t_x allow a parameterization that can vary from 0 to 100%. However, after being pre-defined, they remain fixed during all the N_{ger} in each execution of the GA-OBS. To generate N_{pop+1}, the N_{pop} swapping technique with elitism by De Jong (1988) is used. The termination criterion used, prevailing the first one obtained, is that the MB or N_{ger} is used. Algorithm 2 shows the GA-OBS pseudocode.

```
procedure: GA-OBS // GA initial population generation
 input: problem data and GA parameters
 output: best solution ($C_{Elite}$)
 beginning
   random attributions $s_{ij} = 0$ or $s_{ij} = 1$ to $q_{ij}$ of $POP$
    aptitude $eval(N_{pop})$ by decoding routine;
      for each bit $s_{ij} = 1$:
        check if total ($q_{ij}$) or partial ($w_{ij}$) billing is possible:
          if $x_i > 0$ and $x_i \geq q_i$ bill $\leftarrow$ update inventory ($x_i$) and check next item;
 if $x_i > 0$ and $q_i > x_i$ and $C_\alpha^{w_{ijYes}}$ bill $w_i \leftarrow$ update $x_i$ and check next item;
 if $x_i = 0 \leftarrow$ penalize $b_{qij} \leftarrow$ repair bit and check next item;
 if $x_i > 0$ and $q_i > x_i$ and $C_\alpha^{w_{ijNo}} \leftarrow$ penalize $b_{wij}$ and repair item;
          if $fd_{t+1}$ billed before $fd_t \leftarrow$ penalize $b_{qij}$
          if $pd_{t+1}$ billed before $pd_t \leftarrow$ penalize $b_{qij}$
          if $pr_{t+1}$ billed and $pr_t > pr_{t+1} \leftarrow$ penalize $b_{qij}$
         $F_{fitness} \leftarrow BO$ – total of penalties and repairs;
             if $F_{fitness} < 0$
           $F_{fitness} \leftarrow \$ 1,00$;
     output: $F_{fitness}$ - individual's aptitude value ($C$);
    while (condition is not finished) of
       $N_{pop}$ crossing to generate $C(g)$;
       $N_{pop}$ mutation to modify $C(g)$;
       aptitude $eval(C)$ by codification routine;
       select $N_{pop}(g+1)$ for $N_{pop}(g)$ and $C(g)$;
       $g \leftarrow g+1$;
    end
    output: the best solution ($C_{Elite}$);
 end
```

Algorithm 2. General implementation structure (GA-OBS)
Source: Authors

Computational experiments and result analyses

This section and its subsections detail problem configurations, computational experiments, and analyses of the results obtained by the GA-OBS and the IGA-OBS. These analyses focused on assessing performance and on the conditions to adapt GA-OBS/IGA-OBS to the reality of the aforementioned DC. Algorithm implementations and computational experiments were carried out in a microcomputer featuring a 2.0GHz i7 processor and 8GB of RAM. Programming was conducted in Python, and both data inputs/outputs and analyses were supported by Microsoft Office Excel 2016. Experiments included a set of problems with different complexity levels based on the literature and the reality of DCs. The entire problem configurations and the instances used in the experiments are identical to those considered by Pinto and Nagano (2020). Therefore, the problems are classified into three categories: i) Small (SM); ii) Medium (ME) and; iii) Large (LG).

Test problems and parameter setting

This subsection details problem formulations and parameter configurations or each OBS category. For a better comparative analysis of the algorithms, all the problems are configured so that the MB is the optimized solution for the OBS. Thus, in the DC under study, there is a total of 1 250 types of SKUs to satisfy the POP at a given t moment of the VTW. Depending on the OBS category, SKU restrictions in the DC are the following: i) $x_i = 0$ from 1 to 2,5% and; ii) $x_i = 2$ from 2 to 3%. All billing processes take place immediately after VTW = 8 (hours), where 50% of the POP orders contain $C_\alpha^{w_{ijYes}}$ and fr_{Yes}. The number of SKUs that repeat among the n orders of each POP range from 50 to 80%. However: i) if $x_i = 0$, there will be no SKU repeated in another order; ii) if $x_i = 2$, there will be only one SKU repeated in another order. Date configurations are shown next, and they presuppose that t refers to the date of the last order input in the POP. For id_{ij}: i) 50% have $id_{ij} = t$; and ii) 50% of the remaining id_{ij} range from $id_{ij} = t_{-5}$ days to $id_{ij} = t_{-1}$ days. For fd_{ij}: i) 30% ($fd_{ij} = t$); ii) 30% ($fd_{ij} = t_{+10}$ days); iii) 20% ($fd_{ij} = t_{+20}$ days); and iv) 20% ($fd_{ij} = t_{+30}$ days). For pd_{ij}: i) 20% ($pd_{ij} = t$); ii) 20% ($pd_{ij} = t_{+10}$ days); iii) 20% ($pd_{ij} = t_{+15}$ days); iv) 20% ($pd_{ij} = t_{+30}$ days); and v) 20% ($pd_{ij} = t_{+45}$ days). pr_{ij} is randomly determined by an even distribution function [100, 1 000]. However: i) if $x_i = 0$, pr_{ij} = \$200,00; ii) if $x_i = 2$, pr_{ij} = \$100,00; and iii) if SKU repeated among orders is $C_\alpha^{w_{ijNo}}$, then pr_{ij} = \$ 120,00. Table 5 summarizes the problems formulated for the OBS.

Table 5. Summary of problem settings

OBS Problem	SM-1	SM-2	ME-3	ME-4	LG-6	LG-6
Number of orders of *POP*	10	15	20	30	40	50
Number of SKU in each O_j	2	2	4	4	6	6
Demand of each SKU in O_j	2	2	2	2	2	2
Different types of SKU in each O_j	20	30	80	120	240	300
Total demand of SKU in each O_j	40	60	160	240	480	600
Total Value (*TV*) of billing in each *POP*	15 000,00	20 000,00	54 908,00	9 836,00	209 408,00	250 274,00

Source: Authors

Computational results and analyses

This section describes the results of the computational experiments and the performance analyses of the GA-OBS/IGA-OBS proposed for the OBS. The analyses were carried out by means of Outcome Assessment Metrics (OAM), and the final assessment was consubstantiated by the set of average results obtained by the GA-OBS/IGA-OBS. GA-OBS computational experiments demonstrated that a critical factor in the generation of high-aptitude invididuals is the calibration of the genetic operators. There is no standard formula to indicate which ideal parameter configuration will produce the appearance of C_{Elite} in the GA-OBS. While considering that the new generations are not deterministic, we sought to better deal with the trade-off between the quality of the solutions and the computational efficacy of the GA-OBS. The option was to use average parameters that better adjusted to the real situations and the design of the evolutive genetic structure of the GA-OBS. After the execution series for the each OBS problem category, combinations and parameter minimum and maximum limits that provided the best solutions to the GA-OBS are summarized in Table 6.

Table 6. Parameterization for genetic algorithms (GA-OBS)

Parameters	SM-1	SM-2	ME-3	ME-4	LG-5	LG-6
Population size (N_{pop})	200	400	800	1 000	1 500	2 000
Generation number (N_{ger})	200	400	2.000	5 000	8 000	12 000
Crossover probability (p_c)	80%	80%	80%	50%	50%	50%
Mutation probability (p_m)	80%	50%	20%	20%	20%	30%
Mutation rate of bits (t_x)	2%	1%	1%	1%	1%	1%

Source: Authors

In Table 6, it is possible to verify that the initial N_{pop} ranges from 200 to 2 000 individuals, and that the N_{ger} ranges from 200 to 12 000 individuals. Notice that p_c ranges from 50 to 80%, while p_m ranges from 20 to 80% using a t_x of 1 or 2% depending on OBS complexity. Tables 7 and 8 sumarize the results obtained by the GA-OBS and IGA-OBS according to the following OAM: i) Tardiness in Customer Orders (TCO); ii) Number of Fulfilled Orders (NFO); iii) Total of Billed Products (TBP); iv) Computational Processing Time (CPT); v) Maximum Billing (MB) ; and vi) Total Billing (TB).

Tables 7 and 8 demonstrate that GA-OBS and IGA-OBS can satisfy all rules, restrictions, and decision criteria to maximize the TB for each OBS instance. The genetic structure implemented in the GA-OBS can conduct the search for C_{Elite} and meet *al*l conditions attributed to the OBS. In general, the GA-OBS converges up to C_{Elite} from the third until the tenth generation at most for each OBS category. This demonstrates that the size of both the N_{pop} and the N_{ger}, and that p_c, p_m, and t_x have proved to be sufficient to create a level of diversity capable of capturing all OBS specificities and allow the convergence of the GA-OBS.

However, obtaining the best results for instances of higher OBS levels is conditioned to significant increases in the size of the N_{pop} and the N_{ger}. Thus, there is a relative dependence on the size of the N_{pop} and the N_{ger}, which, if they are not large enough to expand the search space, will make the GA-OBS stagnate in a local solution that is distant from TB maximization. Note that, for a POP with more than 30 orders containing more than 80 lines and 200 SKUs, the GA-OBS obtained a much higher CPT than the IGA-OBS. This happens because the GA-OBS solutions require an exaustive search for a large number of possible solutions, thus demanding intensive effort in verifications and repairs, which exponentially expands the CPT and can make it unfeasible for OBS reality. Probabilistic properties and the configuration of the genetic representation linked to an aptitude function guided by penalties and repairs are critical

Table 7. Results obtained by genetic algorithms (GA-OBS)

OAMs	Measures	SM-1	SM-2	ME-3	ME-4	LG -5	LG -6
TCO	Unit	–	–	–	–	–	–
NFO	Unit	10	15	20	30	40	50
TBP	Unit	34	50	134	200	402	500
CPT	Minutes	0,522	1 124	70 857	316 952	1 367 555	3 492 102
MB	Dollar (US$)	14 400,00	19 000,00	52 908,00	94 336,00	201 608,00	240 274,00
TB	Dollar (US$)	14 400,00	19 000,00	52 908,00	94 336,00	201 608,00	240 274,00

Source: Authors

Table 8. Results obtained by the Iterative Greedy Algorithm (IGA-OBS)

OAMs	Measures	SM-1	SM-2	ME-3	ME-4	LG -5	LG -6
TCO	Unit	–	–	–	–	–	–
NFO	Unit	10	15	20	30	40	50
TBP	Unit	34	50	134	200	402	500
CPT	Minutes	0,063	0,067	0,083	0,100	0,133	0,200
MB	Dollar (US$)	14 400,00	19 000,00	52 908,00	94 336,00	201 608,00	240 274,00
TB	Dollar (US$)	14 400,00	19 000,00	52 908,00	94 336,00	201 608,00	240 274,00

Source: Authors

factors for the evolution and convergence of the GA-OBS. We also verified that the GA-OBS is extremely sensitive to the calibration of genetic operators and the parameterization of p_c, p_m, and t_x, which is the best possible to favor the diffusion of positive genetic features for each new generation of the GA-OBS. Improper parameterizations can destroy the aptitude of the individuals or force the evolution to occur more slowly, as well as leading to a premature convergence or demanding a CPT that makes the GA-OBS unfeasible. The fact is that, the greater the OBS instance, the greater the number of penalties and repairs, thus the longer the GA-OBS solution will be.

In practical terms, the GA-OBS is limited to medium-size problems and differs from the needs of managers when faced with the complexities present in the daily reality of DCs. On the other hand, the experiments evidenced that the performance of the IGA-OBS was much better than that of the GA-OBS, and that it produced optimized solutions with a CPT that is less than one minute for any OBS category. The use of the IGA-OBS enables managers to deal more quickly and consistently with higher levels of OBS complexity; the faster the flow of information, the higher is the degree of negotiation accuracy, and the faster is the OBS decision-making. These actions result in less waste of time and greater flexibility and precision to schedule billing and picking processes within the DC. IGA-OBS solutions optimize the quality of order protfolio fulfilment and cash flow management by reducing the DC's eventual financial losses. In general terms, the IGA-OBS provides a tool that enables managers to make decisions in a more agile and consistent way regarding the trade-off between the level of customer service and the maximization of the DC's financial result. In addition to that, the IGA-OBS does not use penalties or repairs, and it can be implemented without major difficulties to other OBS and DC configurations. The option to use Excel allows the main current management software programs to extract .xls files to make uploads to the IGA-OBS. Analysis of SKU inventory specificities and the best VTW adjustment regarding the POP size also contribute to formulating OMs with practical designs that are more robust and suitable for OBS.

Final considerations

This paper proposes an efficient algorithm to solve a specific billing maximization problem called Optimized Billing Sequencing (OBS). Initially approached by Pinto *et al*. (2018), OBS refers to the optimization of order portfolio billing processes in a typical Distribution Center (DC). In the OBS under study, Stock Keeping Unit (SKU) inventories are controlled at mimimum levels inside the DC. There are, however, uncertainties regarding the management of the demand, which is stochastic, and billings occur from Variable Time Windows (VTW). Most delivery deadlines are tight, and there is a high frequency of small orders containing minimum amounts of multiple SKUs. It is not uncommon that, when billing, determining fulfilment rankings may be necessary, as well as analyzing whether customers accept partial amounts due to SKU restrictions. Decision making about billing prioritizes fulfilment and payment dates in compliance with the Earliest Due Date (EDD) rule. Thus, the new algorithm proposed for the OBS was called Iterative Greedy Algorithm (IGA-OBS) and its performance was compared to the genetic algorithm (GA-OBS) by Pinto and Nagano (2020). Experiments with problems with different levels of complexity demonstrated that the algorithms satisfy all rules, restrictions, and decision variables, and they obtain solutions of satisfactory quality for all OBS instances. It was evidenced that the GA-OBS is limited to medium-size problems, as it demands a high computational processing time that differs from those required to the reality of current DCs. However, the GA-OBS is capable of producing optimized solutions with a computational processing time of less than one minute for any OBS problem. This research fills a gap in the literature and makes valuable contributions to further studies on the development of algorithms with practical designs that are more robust and suitable for OBS. The proposed IGA-OBS enables managers to make decisions in a more agile and consistent way in terms of the trade-off between the level of customer service and the maximization of the financial result of the aforementioned DC. There is still a vast field of inquiries and assumptions for new optimization methods for many other approaches and configurations for the so-called OBS. The main limitation is that the literature does not yet provide an available database with different OBS problems to better test the IGA-OBS/GA-OBS. Suggestions for further researches are: i) to conduct studies with actual applications, so a comparative analysis of the processes adopted by managers versus those resulting from the IGA-OBS can be made; ii) to implement more efficient designs to elements, parameters, and genetic operators, or formulate evolutionary genetic representations that improve the GA-OBS performance; iii) to carry out extensive computational experiments by means of comparative studies among other renowned metaheuristics versus the IGA-OBS; iv) to assess the IGA-OBS with dynamic variables in order to deal with payment deadlines, cashflow, demand forecast, and production lead times that replenish DC inventories.

Acknowledgements

The research of the authors is partially supported by grants numbers 306,075/2017-2 and 430,137/2018-4 from the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), as well as by grant 2700441 from the Fundação Nacional de Desenvolvimento do Ensino Superior Particular (FUNADESP), Brazil.

References

Bandyopadhyay, S. and Bhattacharya, R. (2014). Solving a tri-objective supply chain problem with modified NSGA-II algorithm. *Journal of Manufacturing Systems*, *33*(1), 41-50. https://doi.org/10.1016/j.jmsy.2013.12.001

Baud-Lavigne, B., Bassetto, S., and Agard, B. (2014). A method for a robust optimization of joint product and supply chain design. *Journal of Intelligent Manufacturing*, *27*(4), 741-749. https://doi.org/10.1007/s10845-014-0908-5

Bottani, E., Cecconi, M., Vignali, G., and Montanari, R. (2012). Optimisation of storage allocation in order picking operations through a genetic algorithm. *International Journal of Logistics Research and Applications*, *15*(2), 127-146. https://doi.org/10.1080/13675567.2012.694860

Boysen, N., De Koster, R., and Weidinger, F. (2019). Warehousing in the e-commerce era: A survey. *European Journal of Operational Research*. *277*(2), 396-411. https://doi.org/10.1016/j.ejor.2018.08.023

Chien, C., Kim, K. H., Liu, B., and Gen, M. (2012). Advanced decision and intelligence technologies for manufacturing and logistics. *Journal of Intelligent Manufacturing*, *23*(6), 2133-2135. https://doi.org/10.1007/s10845-011-0559-8

De Jong, K. (1988). Learning with genetic algorithms: An overview. *Machine Learning*, *3*(2-3), 121-138. https://doi.org/10.1007/BF00113894

Diabat, A. (2014). Hybrid algorithm for a vendor managed inventory system in a two-echelon supply chain. *European Journal of Operational Research*, *238*(1), 114-121. https://doi.org/10.1016/j.ejor.2014.02.061

Diabat, A. and Deskoores, R. M. (2016). A hybrid genetic algorithm based heuristic for an integrated supply chain problem. *Journal of Manufacturing Systems*, *38*, 172-180. https://doi.org/10.1016/j.jmsy.2015.04.011

Gen, M., Cheng, R., and Lin, L. (2008). *Network models and optimization: Multiobjective genetic algorithms approach*. Springer. https://doi.org/10.1007/978-1-84800-181-7

Ghiami, Y., Williams, T., and Wu, Y. (2013). A two-echelon inventory model for a deteriorating item with stock-dependent demand, partial backlogging and capacity constraints. *European Journal of Operational Research*, *231*(3), 587-597. https://doi.org/10.1016/j.ejor.2013.06.015

Haq, A. N. and Boddu, V. (2014). Analysis of enablers for the implementation of leagile supply chain management using an integrated fuzzy QFD approach. *Journal of Intelligent Manufacturing*, *28*(1), 1-12. https://doi.org/10.1007/s10845-014-0957-9

Haupt, R. L. and Haupt, S. E. (2004). *Practical genetic algorithms* (2nd ed.). Wiley.

Holland, J. H. (1975). *Adaptation in natural and artificial systems*. University of Michigan Press.

Huang, H. and Ke, H. (2017). Pricing decision problem for substitutable products based on uncertainty theory. *Journal of Intelligent Manufacturing*, *28*(3), 503-514. https://doi.org/10.1007/s10845-014-0991-7

İnkaya, T. and Akansel, M. (2017). Coordinated scheduling of the transfer lots in an assembly-type supply chain: a genetic algorithm approach. *Journal of Intelligent Manufacturing*, *28*(4), 1005-1015. https://doi.org/10.1007/s10845-015-1041-9

Kumar, R. S., Tiwari, M., and Goswami, A. (2016). Two-echelon fuzzy stochastic supply chain for the manufacturer-buyer integrated production-inventory system. *Journal of Intelligent Manufacturing*, *27*(4), 875-888. https://doi.org/10.1007/s10845-014-0921-8

Ledari, A. M., Pasandideh, S. H. R., and Koupaei, M. N. (2018). A new newsvendor policy model for dual-sourcing supply chains by considering disruption risk and special order. *Journal of Intelligent Manufacturing*, *25*(6), 1367-1376. https://doi.org/10.1007/s10845-015-1104-y

Leung, K., Choy, K., Siu, P. K., Ho, G., Lam, H., and Lee, C. K. (2018). A B2C e-commerce intelligent system for re-engineering the e-order fulfilment process. *Expert Systems with Applications*, *91*, 386-401. https://doi.org/10.1016/j.eswa.2017.09.026

Man, K. F., Tang, K. S., and Kwong, S. (1996). Genetic algorithms: Concepts and Applications. *IEEE Transactions on Industrial Electronics*, *43*(5), 519-534. https://doi.org/10.1109/41.538609

Matthews, J. and Visagie, S. (2013). Order sequencing on a unidirectional cyclical picking line. *European Journal of Operational Research*, *231*(1), 79-87. https://doi.org/10.1016/j.ejor.2013.05.011

Marchet, G., Melacini, M., and Perotti, S. (2015). Investigating order picking system adoption: A case-study-based approach. *International Journal of Logistics Research and Applications*, *18*(1), 82-98. https://doi.org/10.1080/13675567.2014.945400

Mousavi, S. M., Bahreininejad, A., Musa, S. N., and Yusof, F. (2017). A modified particle swarm optimization for solving the integrated location and inventory control problems in a two-echelon supply chain network. *Journal of Intelligent Manufacturing*, *28*(1), 191-206. https://doi.org/10.1007/s10845-014-0970-z

Mousavi, S. M., Hajipour, V., Niaki, S. T. A., and Alikar, N. (2013). Optimizing multi-item multi-period inventory control system with discounted cash flow and inflation: two calibrated meta-heuristic algorithms. *Applied Mathematical Modelling*, *37*(4), 2241-2256. https://doi.org/10.1016/j.apm.2012.05.019

Park, K. and Kyung, G. (2014). Optimization of total inventory cost and order fill rate in a supply chain using PSO. *The International Journal of Advanced Manufacturing Technology*, *70*(9-12), 1533-1541. https://doi.org/10.1007/s00170-013-5399-6

Pinto, A. R. F., Crepaldi, A. F., and Nagano, M. S. (2018). A Genetic algorithm applied to pick sequencing for billing. *Journal of Intelligent Manufacturing*, *29*(2), 405-422. https://doi.org/10.1007/s10845-015-1116-7

Pinto, A. R. F. and Nagano, M. S. (2019). An approach for the solution to order batching and sequencing in picking systems. *Production Engineering Research and Development*, *13*(3-4), 325-341. https://doi.org/10.1007/s11740-019-00904-4

Pinto, A. R. F., and Nagano, M. S. (2020). Genetic algorithms applied to integration and optimization of billing and picking processes. *Journal of Intelligent Manufacturing*, *31*(3), 641-659. https://doi.org/10.1007/s10845-019-01470-3

Richards, G. (2011). *Warehouse Management: A complete guide to improving efficiency and minimizing costs in the modern warehouse*. Kogan Page.

Rim, S. C. and Park, I. S. (2008). Order picking plan to maximize the order fill rate. *Computers and Industrial Engineering*, *55*(3), 557-566. https://doi.org/10.1016/j.cie.2008.01.012

Sereshti, N. and Bijari, M. (2013). Profit maximization in simultaneous lot-sizing and scheduling problem. *Applied Mathematical Modelling*, *37*(23), 9516-9523. https://doi.org/10.1016/j.apm.2013.05.004

Seyedrezaei, M., Najafi, S. E., Aghajani, A., and Valami, H. B. (2012). Designing a genetic algorithm to optimize fulfilled orders in order picking planning problem with probabilistic demand. *International Journal*, *1*(2), 40-57. http://www.riejournal.com/article_47673_0bc47688fe20f8368d8d6f2752146e3e.pdf

Slotnick, S. A. (2011). Order acceptance and scheduling: a taxonomy and review. *European Journal of Operational Research*, *212*(1), 1-11. https://doi.org/10.1016/j.ejor.2010.09.042

van den Berg, J. P., and Zijm, W. H. M. (1999). Models for warehouse management: Classification and examples. *International Journal of Production Economics*, *59*(1), 519-528. https://doi.org/10.1016/S0925-5273(98)00114-5