symmetry

MDPI

Article

Fuzzy Order Acceptance and Scheduling on Identical Parallel Machines

Menşure Zühal Erişgin Barak and Melik Koyuncu *

Department of Industrial Engineering, Cukurova University, Sarıçam 01330, Turkey; zuhalebarak@gmail.com
* Correspondence: mkoyuncu@cu.edu.tr

Abstract: In this study, we focus on the fuzzy order acceptance and scheduling problem in identical parallel machines (FOASIPM), which is a scheduling and optimization problem to decide whether the firm should accept or outsource the order. In general, symmetry is a fundamental property of optimization models used to represent binary relations such as the FOASIPM problem. Symmetry in optimization problems can be considered as an engineering tool to support decision-making. We develop a fuzzy mathematical model (FMM) and a Genetic Algorithm (GA) with two crossover operators. The FOASIPM is formulated as an FMM where the objective is to maximize the total net profit, which includes the revenue, the penalty of tardiness, and the outsourcing. The performance of the proposed methods is tested on the sets of data with orders that are defined by fuzzy durations. We use the signed distance method to handle the fuzzy parameters. While FMM reaches the optimal solution in a reasonable time for datasets with a small number of orders, it cannot find a solution for datasets with a large number of orders due to the NP-hard nature of the problem. Genetic algorithms provide fast solutions for datasets with a medium and large number of orders.

Keywords: parallel machine scheduling; genetic algorithm; fuzzy order acceptance and scheduling; fuzzy ranking; fuzzy mathematical model

check for updates

Citation: Erişgin Barak, M.Z.; Koyuncu, M. Fuzzy Order Acceptance and Scheduling on Identical Parallel Machines. *Symmetry* **2021**, *13*, 1236. https://doi.org/10.3390/sym13071236

Academic Editor: José Carlos R. Alcantud

Received: 29 May 2021
Accepted: 8 July 2021
Published: 9 July 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The problem of Order Acceptance and Scheduling (OAS) has become increasingly important in the last two decades [1]. The order acceptance and scheduling problem occurs in production/service systems where not all orders can be accepted due to capacity constraints and the aim is to sequence orders to have the best schedule of accepted orders in a make-to-order system (MTO) [2]. The fuzzy order acceptance and scheduling problems in identical parallel machines (FOASIPM) is a scheduling and optimization problem to decide on whether the firm should accept the order or outsource.

In general, symmetry is a fundamental property of optimization models used to represent binary relations such as the FOASIPM problem [3]. In addition, there are many classifications of Fuzzy Linear Programming (FLP). According to Zimmermann's classification, FLPs are divided into two categories which are symmetrical and nonsymmetrical models. The objectives and constraints have the same importance in the fuzzy decisions in the symmetrical models, while they have different importance in the non-symmetrical models [4]. Hence, the FOASIPM problem falls into the symmetrical model category.

The FOASIPM problem is not only important for the production department but also the marketing department in a manufacturing company. It aims at increasing the satisfaction of the customer by delivering to the customer on time by making a proper and visionary production schedule. Late delivery orders due to improper planning can lead to deterioration of customer relationships, loss of customers, or delay penalties. On the other hand, poor planning and inefficient scheduling can lead to making decisions incorrectly, such as delivering late or rejecting some orders.

Timely delivery of orders is crucial for both customers and firms. When there are problems in delivering orders on time due to capacity constraints, outsourcing can be used,

which allows orders that cannot be included in the schedule to be delivered to the customer on time, so that timely delivery leads to higher customer satisfaction, strengthens customer relationships and increases the reliability of the company [5,6].

Due to increasing global competition, companies specialize in systems to produce custom orders to meet customer needs. For this reason, today's companies' production systems are mostly evolving into a "make-to-order" system [7]. As a result of this change, the variety of orders received from customers has increased. Most of the manufacturing organization encounters the situation of surplus demand than the capacity to produce [6].

The increasing variety and quantity of orders make it difficult for companies to schedule the orders in the production line. If the orders exceed the production capacity too much, the companies can meet these orders by making new investments, but if this situation is temporary, the costs for the companies will be very high and redundant. Therefore, companies usually have to decide whether to accept or reject orders due to capacity constraints. However, to reject orders is to give a negative response to the customer. This situation may harm the relationship with the customer and may reduce customer loyalty and even lead to the loss of the customer. To increase customer satisfaction and keep good relationships with customers, companies avoid rejecting orders and companies try to find solutions for them. Thus, outsourcing becomes a solution method for the companies that want to accept all orders instead of rejecting them. Outsourcing also enables firms to follow innovations, review business process policies that lead to improving their operating decisions [6].

One of the main advantages of outsourcing is that companies have the option to transfer their orders to outsourcing firms that specialize in this area. Outsourcing becomes more and more popular because of cost reduction, minimizing the number of rejected orders, flexibility in staff utilization, increasing quality, and competitiveness. However, out-sourcing may result in additional costs for the company due to differences in pricing policies between firms. If it is decided to outsource to another company, either no profit can be made on that product, or in some cases, it causes a small loss to the company. Mostly, the companies accept this cost so as not to worsen customer relations and lose the customer.

Briefly, the company wants to accept all incoming orders and make a decision for producing the orders or outsource them. Timely delivery of orders to the customer is important both for the reliability of the company and for non-payment of delay penalties. Therefore, orders that are not profitable due to late delivery on time should be outsourced. In this paper, we develop a fuzzy mathematical model that decides whether the orders produce in the company or outsource. The objective of the model is to maximize the total net revenue while scheduling the orders that will be produced in the company.

This presented paper is organized as follows. Section 2 defines the literature review and briefly presents the recent studies with their methods used. In Section 3 the result found by FMM (Fuzzy Mathematical Model), GADOC (Genetic Algorithm Davis Order Crossover), and GATPC (Genetic Algorithm Two Point Crossover) are presented. Section 4 is related to the discussion of the results. Section 5 is about the data generation and the details of methods used in this study and finally, Section 6 is the conclusion part of this paper.

2. Literature Review

The order acceptance and scheduling (OAS) problem has long been studied in the literature. However, the number of studies on this topic has increased significantly in the last decade. In this chapter, deterministic single-machine OAS studies and then multi-machine OAS problems are explained. Afterward, stochastic and fuzzy studies are explained. The first and largest study of the OAS literature was conducted with a comprehensive review, detailing the origins, development, related and similar areas by Slotnick in 2011 [1]. She categorized the OAS problem into single or multiple machines in stochastic or deterministic systems. OAS studies conducted after this comprehensive review can be classified as single

or multiple-machine systems. OAS studies on single machines are separated by different parameters as well as objective functions and solution methods.

In 2010, one of the pioneering studies in the field of OAS, for maximizing net profit which is the sum of revenue per order minus the total weighted delay penalty on a single machine solved with a developed mathematical model. They classified the orders by release date, revenue, due dates, production time, sequence-dependent setup times, and weighted delay penalty. If the order's delay penalty was equal to or greater than its revenue, the order was not included in the list of accepted orders. They developed a Mix Integer Linear Programming (MILP) to solve this problem. Due to the NP-hardness of the model for larger problems, they proposed three heuristic procedures [2]. There are many studies that have tried to solve the same problem using different solution methods and comparing the models in terms of their efficiency. A tabu search algorithm, an exact branch-and-bound with genetic programming, an artificial bee colony based hyper-heuristic algorithm, a diversity controlling genetic algorithm, a dispatching-rule-based genetic algorithm, a hybrid particle swarm optimization method with tabu search using dispatching rules learned by genetic programming, and a more recent study with a memetic algorithm includes hybridization of biased random key genetic algorithm with an adaptive large neighborhood search was developed to handle this problem [8–10]. A fuzzy mathematical model of the same problem was examined using fuzzy times [11].

A simultaneous OAS problem study to maximize the total net revenue in a single machine was solved using a mathematical programming model and two population-based heuristic algorithms called the biogeography-based optimization algorithm and the genetic algorithm [12]. The OAS problem with machine availability constraints was conducted with earliness and lateness penalties. The objective of their study was to maximize the total net revenue [13].

A green OAS problem with maximizing the total revenue of the accepted orders after subtracting the carbon emission cost and the electricity cost under different time intervals on a single machine was solved using mixed-integer linear programming [14]. Another study on the OAS problem with a single machine environment was solved with two mix-integer programmings and a complicated instance branch and bound algorithm [15]. The order acceptance with two-stage assembly scheduling was presented to maximize the profit, which was calculated as the sum of revenues of the accepted orders minus total weighted tardiness with developing a mixed-integer linear programming and non-semi and semi permutation genetic algorithms to solve the problem [16].

There are several types of multi-machine system order acceptance and scheduling studies in the literature. First, a general survey of deterministic system scheduling with rejection has also dealt with order acceptance and scheduling in multi-machine and sing-le-machine systems [17]. The OAS problem on a flow shop system with two identical machines was conducted with one exact and two heuristic methods to deal with heterogeneous customer orders [18]. A simultaneous order acceptance and scheduling problem on a non-identical six-machine system to maximize revenue was solved using a Lagrangian relaxation algorithm and monolithic mixed-integer linear programming. In that study, the orders have a due date, a revenue, a delayed penalty, and different processing times at the machines, and sequence-dependent setup times [19]. Another study examined the OAS problem on two unrelated parallel machines with minimizing the makespan and total penalty of rejected jobs and solved the problem using integer linear programming, deterministic and randomized approximation algorithms [20].

On two identical machines, the OAS problem was solved with one exact and two heuristic algorithms maximizing the total revenue of orders characterized by their revenue, processing time, due date, and tardiness weight [21]. In a flow shop, the OAS problem was presented with two objective functions simultaneously minimizing makespan and maximizing the total net revenue. Mixed-integer linear programming and an effective parallel neighborhood search algorithm were formulated to deal with this problem [22].

The OAS without concurrent machine availability time on unrelated parallel machines was presented, where the orders were ready at the beginning of the schedule while the machines are unavailable at the beginning. The objective is to minimize the makespan and the total cost of rejecting orders. Due to its NP-hard property, the problem was first studied with a polynomial-time algorithm to obtain a lower bound on the optimal objective value of the problem and then examined a heuristic by using some relaxation techniques [23].

The OAS problem without the due date on identical parallel machines environment with weighted completion time was worked out with a mixed-integer linear programming model to maximize the revenue minus the scheduling cost and a branch and bound algorithm [24]. On unrelated parallel machines, a bi-objective OAS problem with machine-dependent processing times and machine eligibility constraints was presented. The first objective is to maximize the total net profit concerning the revenue, tardiness cost, and production cost and the second is to minimize the makespan, which is a classic scheduling criterion concerning productivity. Due to the conflicting objective to solve the problem, the mathematical model was formulated as multi-objective mixed-integer linear programming and a list scheduling-based multi-objective parthenogenesis algorithm was developed [25]. Another OAS problem on unrelated parallel machines with the objective of the maximization of total net revenue after determining accepting orders and how to arrange the accepted orders in a processing sequence. Two water flow algorithms were developed and compared with, particle swarm optimization embedded variable neighborhood search with variable neighborhood search, to find out their solution qualities [26].

One of the recent studies on OAS with pricing policies in a parallel machine environment was defined as a linear demand function with primary market volume and demand price sensitivity of order was used while defining the revenue. A mixed-integer linear program was formulated to maximize the net profit. Compression amount and expansion amount were also considered as lost from revenue [27]. Another newly addressed OAS problem on identical parallel machines was with the maximization of total net revenue. To solve the problem, first, a mixed-integer program was formulated and then, a Branch-Relax-and-Check exact method was developed which performs better [28].

In a green manufacturing system, the OAS problem was examined with maximization of revenue with constraints of energy consumption costs and machine launch costs. One of the dynamic programming algorithms based on an encoding and decoding procedure with a random-key to value rule and two unique neighborhood structures, crossover neighborhood structure, and a mutation neighborhood structure was developed to handle the problem [29]. A simultaneous OAS study addressing both single-machine and three parallel machine systems was examined to minimize the completion time of all accepted jobs and the total penalties of all rejected orders which are characterized by different release dates, processing times, and penalties. A total of three different algorithms are proposed to handle the problem according to the release date of orders [30].

In the comprehensive review of the OAS literature, some studies deal with OAS problems stochastically. Stochastic studies were carried out in which the arrival of orders, production times, or due dates in the system is expressed with a distribution [1]. In some of these studies, random processing time and random common due dates are included to solve the OAS problem with simulation methods. The random machine breakdowns or preemptions in the OAS problem are also studied [31]. Unlike stochastic studies when the data obtained from the system were not sufficient to find the distributions, the OAS problem was tried to be solved in a single machine system using fuzzy logic to express the uncertainties [11].

In this paper, apart from the identical parallel machine system order acceptance and scheduling (OAS) problem studies in the literature [26], the production time, set up times, and due dates are modeled as fuzzy numbers. Fuzzy order acceptance and scheduling problems are handled in a single machine environment in the previous study [11]. In this presented study, uncertainties that exist in production systems and durations are considered as fuzzy parameters, and a more real-life production system order acceptance

and scheduling problem is modeled in two or more identical parallel machines. In this paper, the objective of the fuzzy order acceptance and scheduling problem in an identical parallel machine system is to maximize the net revenue, which is the sum of revenues minus total weighted tardiness and outsourcing cost. When the objective function has formalized the maximization of total net revenue, as in real life, an additional outsourcing penalty is taken into account rather than rejecting the order and losing the customer. To the best of our knowledge, there is no research on the order acceptance and scheduling problem both considering the fuzzy duration and outsourcing penalty on identical parallel machines. One of the main contributions of this study is integrating outsourcing decisions on order acceptance and scheduling on identical parallel machine systems with fuzzy durations. The other contribution of this study is the fuzzy mathematical model (FMM) and two genetic algorithms GADOC (Genetic Algorithm with Davis Order Crossover) and GATPC (Genetic Algorithm with Two Point Crossover) are developed to solve the problem. The genetic algorithm was applied to the problem with the elitism strategy and the DOC (Davis Order Crossover) and TPC (Two Point Crossover) operators for the first time on the OAS problem in the identical parallel machine system. The effect of the crossover method in the genetic algorithm on FOASIPM is also shown in this paper. Additionally, the experiments show that the proposed heuristic methods are effective and rapid in solving the FOASIPM problem.

3. Materials and Methods

In the presented study, we examine order acceptance and scheduling problems on identical parallel machines with fuzzy parameters. The objective of the study is to obtain the maximum revenue with scheduling orders that are known at the beginning of the planning horizon. Each order is characterized by revenue, fuzzy production time, fuzzy delivery date, fuzzy sequence-dependent set-up time, outsourcing, and weighted delay penalty cost. The completion time of orders depends on the sequence of the order in the schedule. Some orders may not be completed before the delivery date due to scheduling constraints. It is decided to gain profit, pay a delayed discount or pay an outsourcing penalty according to the difference between the completion time and delivery date. For orders that cannot be completed by the delivery date, firms should choose one of the options of paying a delayed penalty or outsourcing the order.

The order acceptance and scheduling problem is an optimization problem. As mentioned in the literature review, the exact and heuristic methods are used for solving this optimization problem. Therefore, in this paper, a fuzzy mathematical model and two heuristic methods are developed.

In manufacturing systems, Scheduling generally faced with uncertainties resource-related and/or job-related [32]. Additionally, issues such as machine hold-ups and man-made factors may lead to uncertainties in the processing and set up times. Therefore, crisp processing and crisp sequence-dependent setup times may not be enough to model the real system problem appropriately [31]. In the previous works on OAS, some stochastic OAS problems handle uncertainties with stochastic modeling, and the processing times and due dates are defined as random variables with a known probability distribution. In production systems, having enough information or data to define the probability distribution of random processing or set up time may not be possible. Additionally, in the MTO manufacturing system with more customized orders, it may be hard to collect the data and it can be hard and novel for the firm. Therefore, defining a probability distribution of processing and sequence-dependent setup times may be very challenging under conditions where finding exact historical data is extremely difficult. However, uncertainties can be also modeled with fuzzy logic. Therefore, in this presented paper, the order acceptance and scheduling problem is handled by taking this uncertainty condition into account using fuzzy logic and numbers [33].

Fuzzy logic emerged in the context of the theory of fuzzy sets introduced in 1965 [34]. The fuzzy logic is widely used to model production processes where uncertainties exist.

When measurements are made in a production process, production times and machine setup times may not be the same, even for the same type of product. In addition to these, delivery times of orders may be uncertain in production systems. To handle these types of uncertainties in order acceptance and scheduling problems fuzzy mathematical programming is developed. In the FOASIPM problem considered in this paper, the fuzzy processing times, fuzzy sequence-dependent setup times, and fuzzy due dates are defined by triangular fuzzy numbers (TFN). Which type of fuzzy number should be used is decided according to the properties of the data used in the problem. Since the fuzzy processing time, sequence-dependent setup time, and fuzzy due date of each order are suitable to represent as triangular fuzzy numbers.

As mentioned in the introduction, this developed model is an example of fuzzy linear programming. The existing FOASIPM is a linear programming problem in which some of the coefficients are represented by non-negative triangular fuzzy numbers. All the fuzzy coefficients should be converted or approximated into their nearest symmetric fuzzy numbers in linear programming problems with fuzzy coefficients. However, the results after this conversion become approximate and the constraints cannot be exactly described. The type of fuzziness and its specification are the key points while defining fuzzy linear programming. One classification of FLP is made on which of its components is fuzzy or crisp. The basic components of Linear Programming (LP) are decision variables, objective functions, constraints and their coefficients and inequalities. For a model of LP to be an FLP, at least one of these components must be fuzzy [35]. In this presented FOASIPM problem can be defined as the FLP problem with fuzzy coefficients of constraints.

To operate with fuzzy parameters, it is necessary to convert fuzzy parameters to crisp parameters. The ranking methods are used to convert fuzzy parameters to crisp parameters to be able to operate with fuzzy parameters. In the year 2000, the signed distance method was developed by Yao for ranking triangular fuzzy numbers. This ranking method was chosen because it is more successful than the centroid method, which is another ranking method [36]. The signed distance method is used in ordering the triangle fuzzy numbers, just as it is applied in real numbers. After the operation of two triangular fuzzy numbers such as addition or subtraction, the obtained result fuzzy number will be a triangular fuzzy number. For ranking the fuzzy triangle numbers, the signed distance formulation is used. Calculation with this Equation (1) defined the signed distance of a triangular number $\widetilde{F}$. Let $\widetilde{F}$= (u_1, u_2, u_3) is a triangular fuzzy number with u_1 and u_2 which are respectively the lower and upper bounds of $\widetilde{F}$ [37].

$$d\left(\widetilde{F}, 0\right) = \frac{1}{4}(2\,u_2 + u_1 + u_3) \tag{1}$$

Let $d(\widetilde{M})$ denote the signed distance of a fuzzy number $\widetilde{M}$ = (x_1, x_2, x_3), the ranking relation between $\widetilde{M}$ and $\widetilde{F}$ is defined as $\widetilde{M} > \widetilde{F} \Leftrightarrow d(\widetilde{M}) > d(\widetilde{F})$.

3.1. Mathematical Model

The fuzzy order acceptance and scheduling in the identical parallel machine is an optimization problem. The objective of this problem is to have the maximum net revenue. The net revenue can be maximized with the minimum amount of delay and outsourcing cost. For this reason, one of the methods that can find the optimum scheduling with the maximum profit despite the due date constraints is the mathematical model. Therefore, we develop a fuzzy mathematical model for Fuzzy Order Acceptance and Scheduling on Identical Parallel Machines (FOASIPM) and the model assumptions are given below and model parameters and variables are given in Table 1.

Table 1. The fuzzy mathematical model parameters and variables.

Parameter	Definition
O	Set of Order
M	Set of Machine
i	Order index (i = 1, . . . , o)
j	Order index (j = 1, . . . , o)
m	Machine index (m = 1, . . . , m)
w_i	The weighted delay penalty cost of order i, i∈O
Co_i	The outsourced penalty cost of order i,
e_i	The revenue of order i, i∈O
$\widetilde{p}$	The fuzzy parameter of production times of order i, i∈O
$\widetilde{d}_{ii}$	The fuzzy parameter of the due date of order i, i∈O
$\widetilde{S}_{0i}$	The fuzzy parameter of beginning setup times of order i, i∈O
$\widetilde{S}_{ij}$	The fuzzy parameter of sequence-dependent setup times of order i, i∈O
B	Big positive number
A	Average of assigned order to each machine
C_i	The completion time of order i, i∈O
T_i	The delay time of order i, i∈O
R_i	The net profit if or not delay occurs for order i, i∈O
K_i	The net profit if or not outsourcing occurs for order i, i∈O
a_i	Acceptance of order i, if accepted 1, otherwise 0. i∈O
b_{im}	Assignment of order i on machine m as 1, if not 0. i∈O
x_{ijm}	Assignment of order j after order i on machine m as 1, if not 0. i∈O

Assumptions must be made while developing mathematical models. For example, an assumption in the model is that orders have no priority. If there is a priority among the orders, it must be included in the model. In the single-time assumption, it is assumed that the machine is not broken and the product is produced uninterruptedly. If these assumptions are not accepted or taken into account, a completely different new mathematical model must be established. Including one of these assumptions in the problem completely changes the structure and the model of the problem. For example, if orders are prioritized against each other, the model has to be changed. Priority over orders will affect the total net revenue and scheduling. For this reason, the superiority of the orders over each other is considered equal in this study. Another example can be the idle times of machines. Machine idle times in the system are also not considered. If idle times were included in the model, the order completion times would change. As a result, penalties for delay and outsourcing may increase and net profit may decrease. Idle time is another factor that affects both the nature of the problem and its result. Therefore, the following assumptions given above are critical in constructing the model of the problem:

i. All orders are received before the planning and all the characteristics of receiving orders are known.
ii. Orders have no priority over each other.
iii. Production times of orders and sequence-dependent setup times do not change according to the assigned machine.
iv. There is more than one identical parallel machine, which does not remain idle in the system.
v. The orders are produced in a single time.
vi. The orders can be produced on any machine. Machine breakdown times in the system are omitted in the model.
vii. All machines had the same properties, and it is assumed that the machines are not broken while the system is working.

In the following parts, the mixed-integer linear model with objective function and constraints.

Objective Function:

$$\mathbf{Max} \sum_{\mathbf{i} \in \mathbf{O}} (\mathbf{K_i}) \tag{2}$$

Subject to

$$\mathbf{R_i} = (\mathbf{e_i} \times \mathbf{a_i}) - (\mathbf{w_i} \times \mathbf{T_i}) \; \forall \mathrm{i} \in \mathrm{O} \tag{3}$$

$$\mathbf{K_i} = (\mathbf{R_i} - ((1 - \mathbf{a_i}) \times \mathbf{Co_i}) \; \forall \mathrm{i} \in \mathrm{O} \tag{4}$$

$$\mathbf{a_i} = \sum_{\mathbf{m} \in \mathbf{M}} \mathbf{b_{im}} \; \forall \mathrm{i} \in \mathrm{O} \tag{5}$$

$$\sum_{\mathbf{m} \in \mathbf{M}} \mathbf{b_{im}} \leq \mathbf{1} \; \forall \mathrm{i} \in \mathrm{O} \tag{6}$$

$$\mathbf{B}.\mathbf{a_i} \geq \mathbf{C_i} \; \forall \mathrm{i} \in \mathrm{O} \tag{7}$$

$$\mathbf{C_i} \geq \tilde{\mathbf{s}}_{\mathbf{0i}} \mathbf{b_{im}} + \tilde{\mathbf{p}}_{\mathbf{i}} \mathbf{b_{im}} \; \forall \mathrm{i} \in \mathrm{O}, \; \forall \mathrm{m} \in \mathrm{M} \tag{8}$$

$$\mathbf{C_i} \geq \tilde{\mathbf{s}}_{\mathbf{ij}} \mathbf{x_{ijm}} + \tilde{\mathbf{p}}_{\mathbf{i}} \mathbf{b_{im}} \; \forall \mathrm{i} \in \mathrm{O}, \; \forall \mathrm{m} \in \mathrm{M} \tag{9}$$

$$\mathbf{C_j} - \mathbf{C_i} + \mathbf{B}\left(\mathbf{3} - \mathbf{x_{ijm}} - \mathbf{b_{im}} - \mathbf{b_{jm}}\right) \geq \tilde{\mathbf{s}}_{\mathbf{ij}} + \tilde{\mathbf{p}}_{\mathbf{j}} \; \forall \mathrm{i} \in \mathrm{O}, \forall \mathrm{j} \in \mathrm{O}, \; \mathrm{i} \neq \mathrm{j}, \; \forall \mathrm{m} \in \mathrm{M} \tag{10}$$

$$\mathbf{C_i} - \mathbf{C_j} + \mathbf{B}\left(\mathbf{2} + \mathbf{x_{ijm}} - \mathbf{b_{im}} - \mathbf{b_{jm}}\right) \geq \tilde{\mathbf{s}}_{\mathbf{ji}} + \tilde{\mathbf{p}}_{\mathbf{i}} \; \forall \mathrm{i} \in \mathrm{O}, \forall \mathrm{j} \in \mathrm{O}, \; \mathrm{i} \neq \mathrm{j}, \; \forall \mathrm{m} \in \mathrm{M} \tag{11}$$

$$\mathbf{T_i} \geq \mathbf{C_i} - \tilde{\mathbf{d}}_{\mathbf{i}} \; \forall \mathrm{i} \in \mathrm{O} \tag{12}$$

$$\mathbf{x_{ijm}} \geq \mathbf{b_{jm}} \; \forall \mathrm{i} \in \mathrm{O}, \forall \mathrm{j} \in \mathrm{O}, \; \mathrm{i} \neq \mathrm{j}, \; \forall \mathrm{m} \in \mathrm{M} \tag{13}$$

$$\mathbf{x_{ijm}} \geq \mathbf{b_{im}} \; \forall \mathrm{i} \in \mathrm{O}, \forall \mathrm{j} \in \mathrm{O}, \; \mathrm{i} \neq \mathrm{j}, \; \forall \mathrm{m} \in \mathrm{M} \tag{14}$$

$$\mathbf{A} \geq \sum_{\mathbf{i} \in \mathbf{O}} \mathbf{b_{im}} \; \forall \mathrm{m} \in \mathrm{M} \tag{15}$$

$$\mathbf{C_i}, \mathbf{T_i} \geq \mathbf{0} \; \mathbf{C_0} = \mathbf{0} \; \forall \mathrm{i} \in \mathrm{O} \tag{16}$$

$$\mathbf{a_i}, \mathbf{b_{im}}, \mathbf{x_{ijm}} \in \{\mathbf{0}, \mathbf{1}\} \; \forall \mathrm{i} \in \mathrm{O}, \; \forall \mathrm{j} \in \mathrm{O}, \; \mathrm{i} \neq \mathrm{j}, \forall \mathrm{m} \in \mathrm{M} \tag{17}$$

In the following part of constraints, crisp constraints Equations (18)–(22) are converted from fuzzy constraints, respectively, Equations (8)–(12) with the signed distance method for ranking triangular fuzzy numbers.

$$\mathbf{C_i} \geq \frac{\mathbf{1}}{\mathbf{4}} \mathbf{b_{im}}[(\mathbf{s1_{0i}} + \mathbf{2s2_{0i}} + \mathbf{s3_{0i}}) + (\mathbf{p1_i} + \mathbf{2p2_i} + \mathbf{p3_i})] \tag{18}$$

$$\mathbf{C_i} = \frac{\mathbf{1}}{\mathbf{4}} \left[\left((\mathbf{s1_{ij}} + \mathbf{2s2_{ij}} + \mathbf{s3_{ij}})\mathbf{x_{ijm}}\right) + ((\mathbf{p1_i} + \mathbf{2p2_i} + \mathbf{p3_i})\mathbf{b_{im}})\right] \tag{19}$$

$$\mathbf{C_j} - \mathbf{C_i} + \mathbf{B}\left(\mathbf{3} - \mathbf{x_{ijm}} - \mathbf{b_{im}} - \mathbf{b_{jm}}\right) \geq \frac{\mathbf{1}}{\mathbf{4}} \left[\left(\mathbf{s1_{ij}} + \mathbf{2s2_{ij}} + \mathbf{s3_{ij}}\right) + \left(\mathbf{p1_j} + \mathbf{2p2_j} + \mathbf{p3_j}\right)\right] \tag{20}$$

$$\mathbf{C_i} - \mathbf{C_j} + \mathbf{B}\left(\mathbf{2} + \mathbf{x_{ijm}} - \mathbf{b_{im}} - \mathbf{b_{jm}}\right) \geq \frac{\mathbf{1}}{\mathbf{4}} \left[\left(\mathbf{s1_{ji}} + \mathbf{2s2_{ji}} + \mathbf{s3_{ji}}\right) + (\mathbf{p1_i} + \mathbf{2p2_i} + \mathbf{p3_i})\right] \tag{21}$$

$$\mathbf{T_i} \geq \mathbf{C_i} - \frac{\mathbf{1}}{\mathbf{4}} \left(\mathbf{d1_i} + \mathbf{2d2_i} + \mathbf{d3_i}\right) \tag{22}$$

The objective function is given in Equation (2) of the FOASIPM problem maximizes the total net revenue. The total net revenue formula is given in Equation (3), which is equal to the difference between the revenue of orders and the delay penalty. Equation (4) calculates the outsourcing costs in case no profit is obtained due to delay from the order. Equation (5) prevents outsourced orders from being assigned to machines and Equation (6) ensures that orders are assigned to only one machine. Equation (7) ensures that the completion time is 0 if an order is not accepted. If the product is accepted, then the constraint becomes redundant. Equation (8) ensures that the completion time of the first order produced on a machine is at least, must be equal to the sum of set up time and production time. Equation (9) ensures that the completion time of an order produced on a machine is at least, must be equal to the sum of set up times and its production time. Equation (10) calculates the differences between the completion times of orders. If two orders are two consecutive orders on the same machine, the difference between completion times is at least the sum

of the set-up time and the production time. This constraint does not apply if two orders are not two consecutive orders on the same machine or different machines. Equation (11) calculates the differences between the completion times of orders. If two orders are not two consecutive orders on the same machine, the difference between completion times is at least the sum of the set-up time and the production time. This constraint does not apply if two orders are not two consecutive orders on the same machine or different machines. Equation (12) calculates the delay time of order. Equations (13) and (14) ensure that if two orders are consecutive orders on a machine, these two orders must have been assigned to that machine. Equation (15) ensures that the number of assigned orders to each machine has to be equal or as close as each. Equation (16) defines that completion time and delay time must be at least zero or greater. Equation (17) is defining the binary variables of accepting the order, assigning it to the machine, and sequencing in the machine which can take the value of 1 or 0. Equations (18)–(22) are the equations that were developed to calculate the parameters defined by fuzzy numbers of expressed Equations (8)–(12), respectively. To obtain these fuzzy equations the signed distance method is used.

3.2. Genetic Algorithm for FOASIPM

Order acceptance and scheduling problem is an NP-Hard problem [2]. The solution space of the FOASIPM problem is huge due to both including scheduling and optimization. Therefore, a heuristic algorithm such as GA is necessary to quickly find a solution that is close to optimum in huge search spaces. Manufacturing firms are seeking acceptable and fast solutions for their production planning problem due to today's competitive business environment. GA also has an advantage with handling continuous and discrete combinatorial problems such as order acceptance and scheduling problems. Genetic Algorithms focus on optimum values for the objective function instead of suboptimal values, especially in struggling with big size problems such as scheduling and optimization problems, the solutions of GA can find the successful solution very close or sometimes slightly less efficient to optimum instead of ensuring an optimal solution each time. This situation also provides an advantage over the time to find a solution in a short time [38].

Due to the NP-hard nature of the FOASIPM problem, it can only be solved for small datasets using the exact solution method. We develop two analogous Genetic Algorithms with different crossover methods used in GADOC and GATPC to solve the FOASIPM problem to obtain a near-optimal solution in a short computational time frame. In the following section, we explain the steps of the genetic algorithm whose flow is shown in Figure 1. The steps of GA are initialization, selection, crossover, mutation, replacement, and stopping criteria.

3.2.1. Initialization

Initialization is one of the most important steps in the algorithm because it affects the execution time and the possibility of finding a feasible solution. In the initialization phase, genes, chromosomes, and fitness functions are created. In the FOASIPM problem, the gene represents an order received from the customer and the chromosomes represent a solution as a sequence of orders for all machines. The fitness function is the total net revenue of each chromosome.

3.2.2. Selection

There are many types of selection processes such as steady-state, tournament, elitism, etc. in GA. In this algorithm, we use elitism to ensure that the best chromosomes from the current population pass into the new population and that these elite chromosomes are not lost when the new population is created [39]. Chromosomes are ranked according to their fitness function values and the first 50% are determined as elite chromosomes and the remaining non-elite chromosomes.

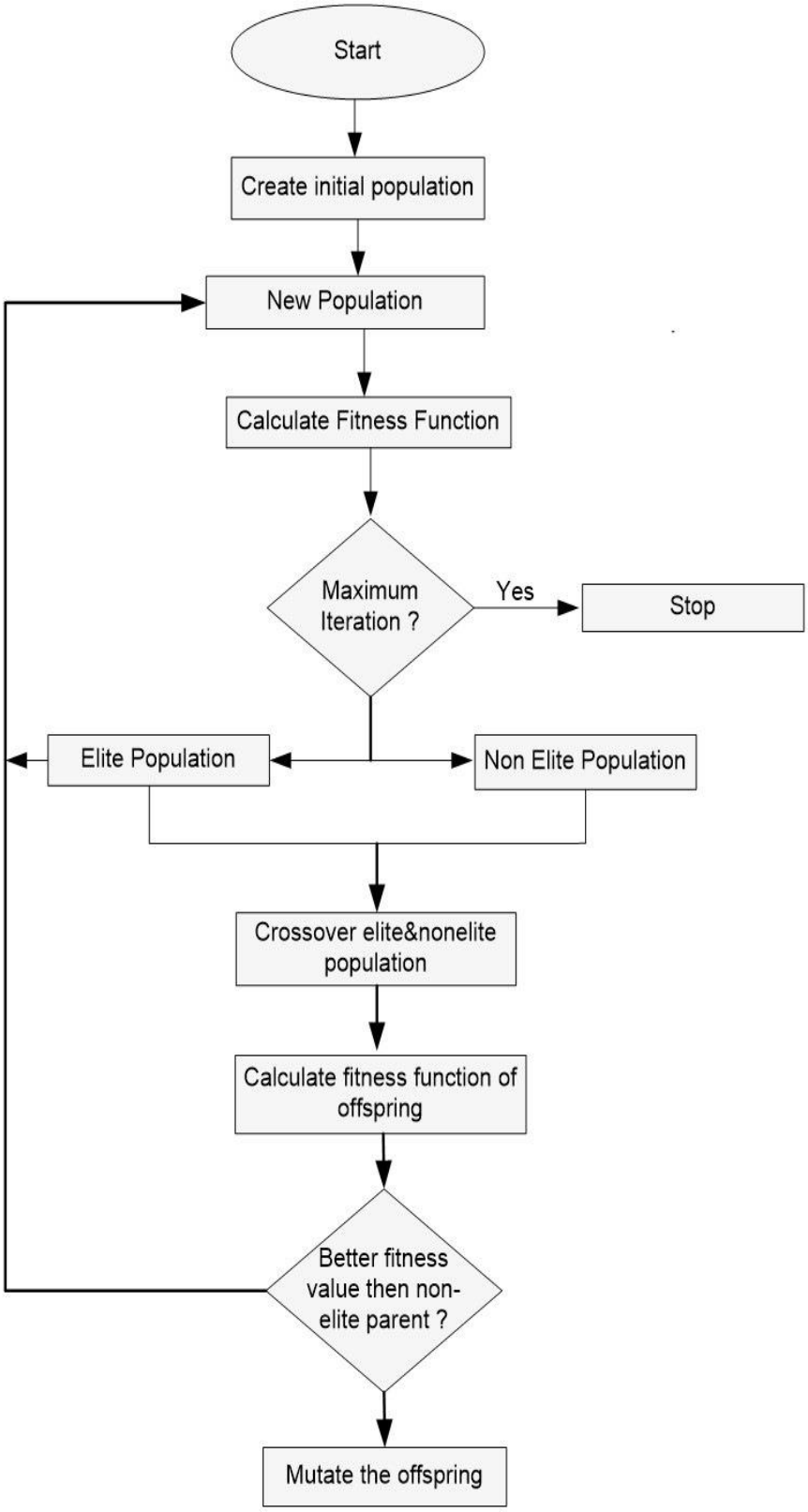


Figure 1. The flow of GADOC and GATPC.

3.2.3. Crossover

The crossover in the GA is the process of generating offspring by selecting the chromosomes from the existing population to create a new population. In this GA, the best elite chromosome and the worst non-elite chromosomes are first selected for crossover. Then

the second-best elite chromosome and the second-worst non-elite chromosome are selected for crossover. This process is continued between all elite chromosomes and non-elite chromosomes, respectively. We use two crossover methods which are the Davis order and two-point crossover methods for generating offspring. The Davis order is used for permutation-based crossovers to transmit information about relative ordering to the offsprings crossover method works with choosing two random crossover points from the elite chromosome and this segment is copied to the offspring. Then, starting with the crossover point on the non-elite chromosome, the remaining unused numbers are copied from the non-elite chromosome to the first offspring by wrapping them around the list. In GADOC, for increasing the randomization of the algorithm, the size of the segment between two random crossover points is chosen randomly from one to four orders. In Figure 2, the size of two random points is selected between four orders. In this crossover method, the sequence of at least one and at most four orders on the chromosome of the offspring are the same as the elite chromosome.

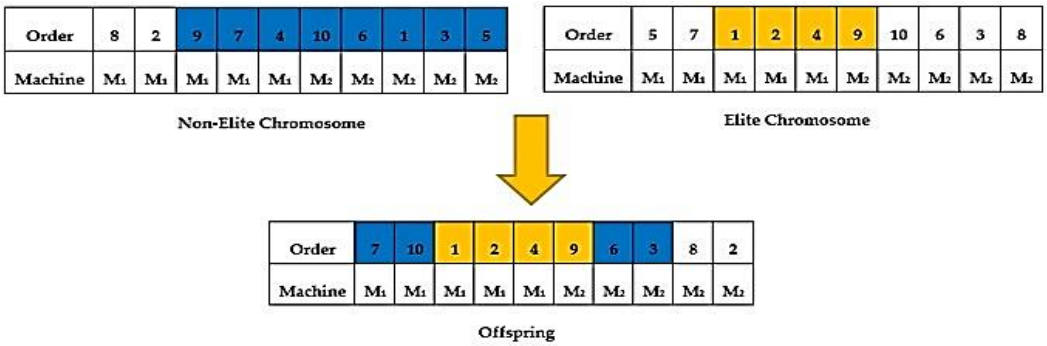


Figure 2. The Davis order crossover process of the GADOC with n = 10 and m = 2.

The crossover in the GATPC is the process of generating new springs by changing selecting two points on chosen chromosomes. In the two-point crossover, in addition to the two randomly chosen points, the width of the region to change between the two chromosomes is also randomly selected as one to four orders. In a two-point crossover, chromosome pairs are cut from two different places and each chromosome is divided into three parts. The crossing is made by exchanging the parts mutually. If a single piece of crossing is exchanged between two chromosomes, two new chromosomes are obtained as shown in Figure 3. However, due to elitism strategy selection, only one new offspring can survive. With the two-point crossover method, two offspring are breeding from one elite and nonelite chromosome. However, since half of the population consists of the elite chromosome, one of the two offspring must be selected and join the new population. The one with a better fitness function value survives among two offspring. Those who have a bad fitness function value are eliminated such as a non-elite chromosome.

3.2.4. Mutation

The mutation in both Genetic Algorithms is used when the offspring does not offer a better fitness function value than their non-elite chromosome after the crossover process. The offspring's fitness functions must be better than their non-elite chromosome. After this comparison process, it is decided whether the new individual joins the new population or mutated. The swap mutation operator is used for the mutation process. The process is carried out by changing the places of the two randomly chosen orders on two random spots on the offspring chromosome as shown in Figure 4. After this mutation process, a new offspring chromosome with a new different sequence is added to the new population.

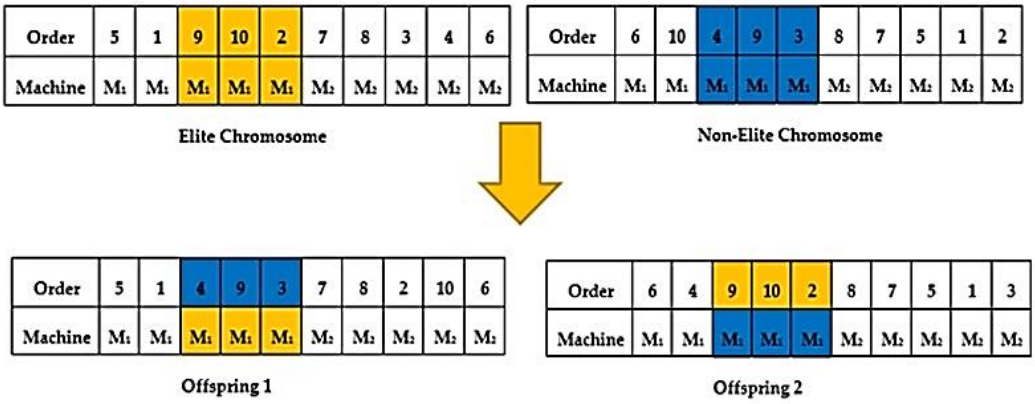


Figure 3. The two-point crossover process of the GATPC with n = 10 and m = 2.

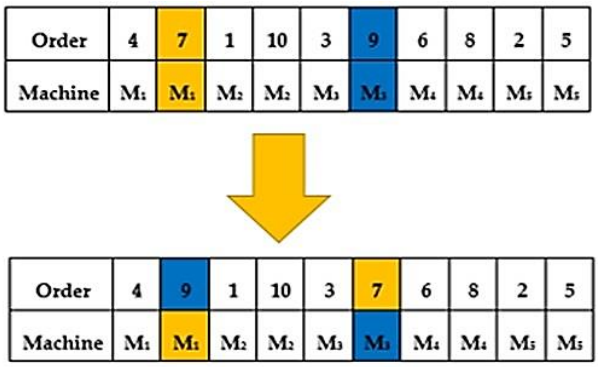


Figure 4. The swap mutation on a chromosome in GADOC and GATPC for n = 10 and m = 5.

3.2.5. Replacement

The replacement is the final process of the GADOC. The elite chromosomes pass directly to the new population without any change. However, non-elite chromosomes must be a crossover with the elite chromosomes. As a result of the crossover process, if the offspring have better fitness function value than non-elite individuals, they move into the new population. Offspring that are not better than their non-elite chromosome can replace the new population after the mutation process. Offspring with worse fitness function than their non-elite chromosome can join the new population after the mutation process. Thus, by comparing the fitness function and increasing the diversity in the population with crossover and mutation, the probability of coming closer to the optimal solution for the new population is increasing.

3.2.6. Stopping Criterion

In GA, there are several types of termination conditions such as not finding more improvement in the population after several iterations or reaching several generations. In this paper, the algorithm stops when it reaches the desired number of iterations. In Algorithm 1, the pseudo-code steps of the GADOC and GATPC are placed.

Algorithm 1 GADOC and GATPC pseudo codes

1: Create a new chromosome
2: Create a chromosome random numbers between [0, 1] as many as order number n
3: Label random numbers between 1 and n. Assigned orders to the machines.
4: Order in descending sequence and get a random sequence of orders.
5: **Repeat** until reaching the population size.
6: Calculate the fitf. of a chromosome
7: Order these chromosomes in descending order according to their fitf.
8: Label the best half of the population as elite and add to the new population.
9: Crossover the elite with the non-elite chromosomes.
10: Compare the Fitf. of offspring and non-elite chromosomes.
11: **If** Fitf.(non-elite) $\leq$ Fitf.(offspring), Add offspring to the new population.
12: **Else** do swap mutation to the offspring chromosome.
13: Add offspring chromosomes to the new population.
14: **Stop** when the offspring chromosomes are finished.
15: **Return** to 6.
16: **Stop** when the maximum iteration is reached.

4. Results

In this section, the effectiveness of mathematical models and developed heuristic algorithms are shown and evaluated. In the presented paper, the order acceptance and scheduling problem turned into a different problem when fuzzy duration and outsourcing cost was included. Since this problem has not been solved before with these parameters in the literature, the datasets were reconstructed. Thus, there is no dataset that we can directly use to show the efficiency of the FMM and genetic algorithms, we adapted datasets to use in our study [2].

The experiments are carried out for order numbers n = 10, 15, 20, 25, 50, 100 and machine numbers m = 2, 3, 4, 5 and for each order. Each order has revenue, fuzzy production time, fuzzy due dates, fuzzy sequence-dependent setup time, weighted delay penalty cost, and outsourcing penalty cost. Datasets have different τ, tardiness factor, and R, the due date range for each order number because the datasets are chosen randomly to use in our study [2]. While the datasets with the same order number were converted to datasets with a different number of machines, due date ranges decreased as the number of machines increased. The number of data generated and used in the study is shown in Table 2.

Table 2. Generated number of data.

Order Number	Machine Number	Total Instances	Total Dataset
10	2,3,4,5	200	20
15	2,3,4,5	300	20
20	2,3,4,5	400	20
25	2,3,4,5	500	20
50	2,3,4,5	1000	20
100	2,3,4,5	2000	20
Total		4400	120

To solve the FOASIPM model, we used the CPLEX solver of the software GAMS. Genetic Algorithms are coded in the MATLAB 2018b program. The experiments of datasets belong to the FOASIPM problem are carried out on a computer 64 bit with Intel Core i5 7200 CPU 2.71 GHz and 8 RAM.

Computational Results

We create small (10–15 orders), medium (20–25 orders), and large-sized (50–100 orders) samples to analyze the performance of the methods developed for the FOASIPM problem. The properties and problems of the created datasets are explained in the previous sections.

Since FMM only provides solutions for small datasets with 2, 3, 4, and 5 identical parallel machines, we solve small-size problems with the software GAMS. (n = 10 and m = 2, 3, 4 and 5)

As we mentioned in the previous works, order acceptance and scheduling problems are NP-Hard [1,2]. Heuristic optimization algorithms are more advantageous and more preferred in solving large problems. Therefore, to solve the larger size FOASIPM problem, we develop two genetic algorithms. First, we compare the results of FMM for the small data set with the results of GADOC and GATPC for the same dataset. However, datasets with medium and large order numbers are solved with only the GADOC and GATPC algorithms.

In genetic algorithms, determining the initial population and iteration number are also crucial decisions on the quality of algorithm performance while solving the problem. In machine scheduling problems, the solution space is large. As the number of orders increases, the solution space becomes larger. The genetic algorithm finds a solution with a randomized mechanism. Therefore, replication must also be performed to find a better solution. To reduce the number of replications required, the initial population is increased to 10,000 chromosomes. Moreover, the number of iterations for small instances is 50, while for medium and large instances, the number of iterations for GADOC and GATPC is set to 100.

To better understand the solution of datasets an example of an optimum solution and schedule belongs to a 10 order dataset on a two-machine system is given in Figure 5. According to the solution found by FMM four orders, 3, 2, 9 and 10 are assigned to machine 1. The other four orders 6, 8, 4, and 5 are assigned to the 2nd machine and the remaining two orders 1 and 7 are outsourced. The net income generated by this schedule is 91.1% of the total income. Outsource and late penalties reduced profits by 8.9%.

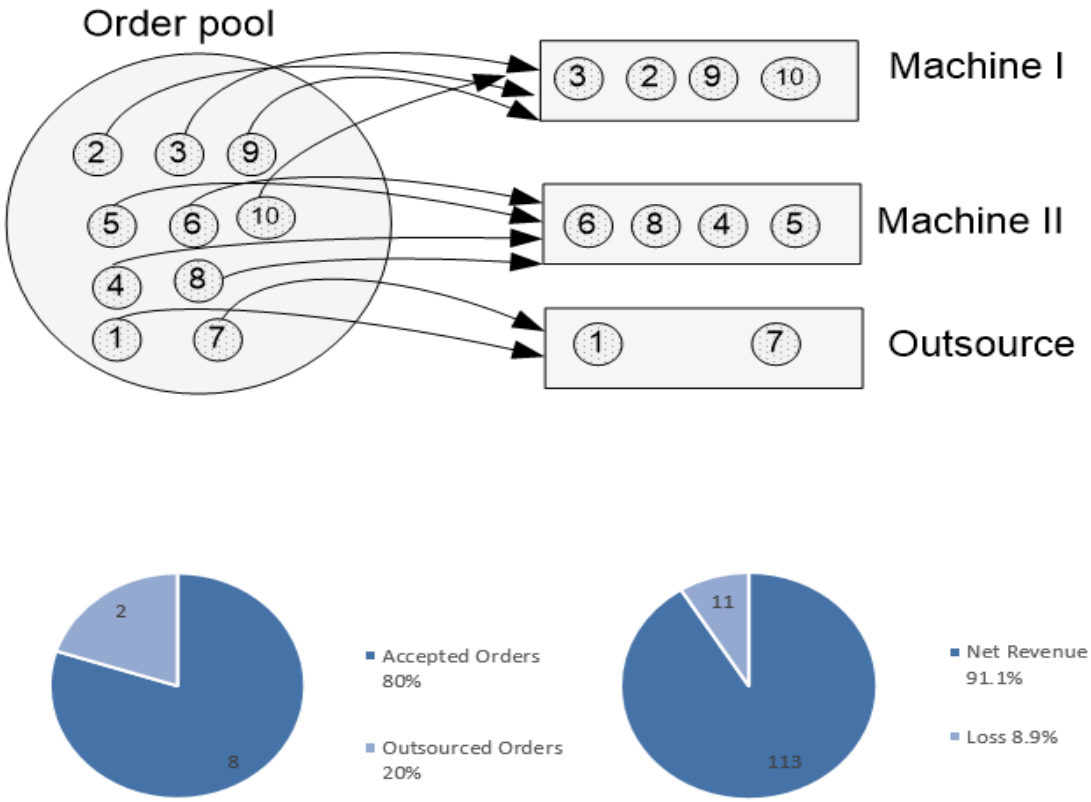


Figure 5. An example solution of a data set with n = 10 and m = 2.

The performance of the algorithm is determined as the deviation from the total revenue or the upper bound. The deviation from the upper bound is calculated as the difference

between the total revenue of all orders in the dataset and the results found by the algorithm divided by the total revenue of all orders of the dataset, as shown in Equation (23).

$$\textbf{Upper Bound Deviation } (\textbf{UBD}) = \textbf{100} \times \frac{\textbf{Total Revenue} - \textbf{Solution}}{\textbf{Total Revenue}} \quad (23)$$

The UBD value is a net profit percentage value that indicates how much the solution found by the algorithm deviates from the total revenue. In the schedule created by the algorithm, there is a decrease in the net total revenue as a result of the decision to outsource some orders and to deliver some orders late. Therefore, a small UBD value is expected in performance measurement. An algorithm with a small UBD value is considered more successful because the closer the algorithm finds the total revenue, the closer it will be to the desired result. If the UBD value is high, the worse and undesirable results will be obtained.

The maximum, minimum, and average upper bound deviations (UBD) of FMM, GADOC, and GATPC are shown in Table 3 for small orders of 2, 3, 4, and 5 machines. For each order number on each machine number, five datasets are generated. For example, the maximum value in column 1 of Table 3 shows the highest deviation among the solution of five datasets. Similarly, the minimum value indicates the lowest deviation, and the average deviation indicates the mean deviation of the solution of five datasets.

Table 3. Upper bound deviation for n = 10, n = 15, and m = 2, 3, 4, 5 of the FMM, GADOC, and GATPC algorithms.

Order Number	Machine Number	FMM Max	FMM Avg	FMM Min	GADOC Max	GADOC Avg	GADOC Min	GATPC Max	GATPC Avg	GATPC Min
10	2	19.3	11.5	4.3	19.3	11.5	4.3	19.3	12.4	0.3
10	3	22.1	13.7	5.2	22.1	13.7	5.2	22.2	13.8	5.2
10	4	30.5	21.7	3.4	30.5	21.7	3.4	40.3	23.7	3.4
10	5	36.4	24.1	9.8	36.4	25.2	9.8	36.4	26.4	9.8
Average		27.0	17.8	5.63	27.0	18.0	5.7	29.55	19.1	5.7
15	2		NA		13.4	8.7	5.7	17.9	13	12.1
15	3		NA		14.7	11.2	8.4	17.3	14.6	11.3
15	4		NA		19.8	14.9	9.8	21.8	16.9	9.9
15	5		NA		23.4	17.3	11.6	25.6	18.9	15
Average			NA		17.8	13.0	8.9	20.7	15.9	12.1

NA Not Available.

The maximum value in column 1 of Table 3 shows the highest deviation among the solution of five datasets. Similarly, the minimum value indicates the lowest deviation, and the average deviation indicates the mean deviation of the solution of five datasets.

The FMM can solve all machine combinations of 10 orders, but it cannot find an optimal solution when the number of orders is increased to 15. Reaching the optimal solution is difficult even for small problems at FMM. Therefore, the performance comparison of FMM, GADOC, and GATPC can be performed for only 10 orders.

According to Table 3, with 10 order instances, the FMM and GADOC methods are successful to find optimal scheduling for two machines. However, GATPC cannot perform in finding the optimum solution as the other two methods. The average UBD for 10 orders on two machines is only 11.5% lost from the total revenue, and 88.5% of total revenue can be gained as net profit with the schedule found by FMM and GADOC methods. Even in 10 orders and on two machines, the GATPC can find a schedule with an average UBD of 12.4%. On three identical parallel machines, FMM the GADOC, and the GATPC perform close to each other with an average loss of 13.7%. On four identical parallel machines, the average profit deviates from the total revenue is 21.7% for the FMM and GADOC. For the GATPC this UBD value is 23.7%. On five parallel machine systems, FMM performed better than other algorithms with an average deviation of 24.1%. While GADOC deviates from 25.2% from total profit, GATPC shows a deviation of 26.4%.

Comparing the average deviations on all machines for 10 order sizes, FMM performed best with net profit which is 82.2% of the total revenue. The second-best performance is GADOC with 82% and the third-best performance is GATPC with average net revenue of 81.9%. According to the average result of solving 10 order datasets, the highest average maximum deviation is 29.55% is for the GATPC and the minimum average deviation is 5.63% for the FMM.

When the solution of the datasets on all machines with 10 order sizes is examined, the highest deviations from the total revenue are calculated as 40.3% for 10 orders on the four machine systems for the GATPC algorithm. Likewise, the minimum UBD is observed as 3.4% for all methods in the system with four parallel machines. With 10 orders when the number of machines increases, the average loss percentage gets larger because as the number of machines increases in the datasets belonging to the same number of orders, due date ranges decrease. Due to the shrinkage of due dates, as the number of machines increases, the UBD value also increases as shown in Table 3 for the same number of orders.

In Table 3, there are also the solutions of 15 orders on all machines. FMM could not provide a solution for datasets with 15 orders as seen in Table 3. When the average performance of GADOC and GATPC algorithms in two-machine datasets is examined, GADOC achieved an average of 8.7% and GATPC found 13% less net profit from total revenue. On three machines the GADOC average UBD value is 11.2% and 14.6% for GATPC. On four machines the gap for net revenue is 14.9% for the GADOC and 16.9% for GATPC. On five machines the average UBD is 17.3% for the GADOC and 18.9% for the GATPC.

The highest maximum UBD value with 25.6% for GATPC is in five machine systems with 15 orders. The minimum UBD value of 5.7% for the GADOC is in two machine system with 15 orders. It is seen that UBD values are lower for GADOC in the solutions of 10 and 15 order sets on a different number of machines. Therefore, in small datasets, the GADOC algorithm performs better in solving the problem by finding higher profitable schedules than GATPC. The GADOC has shown a very good performance compared to the GATPC in systems with 2, 3, 4, and 5 parallel machines.

With 15 orders when the number of machines increases, the average loss percentage gets larger as occurs with 10 orders. According to the average UPD value of the FMM, GADOC, and GATPC methods for 10 orders on all machine numbers, FMM is slightly better than GADOC. Additionally, GADOC is better than the GATPC in terms of UBD. For the same order number 10 and 15, the overall UBD value gets larger, as the number of machines increased due to the shrinkage in the due date range.

While FMM solved only half of the small datasets, GADOC and GATPC provided the solution for all datasets. The GADOC has achieved a very small UBD value of 13% in the average of the whole machine number in 15 datasets. The GADOC succeed to reduce this deviation to an average of 5.7% for 15 orders on two machines. In this problem dataset, for 10 and 15 orders datasets are generated with randomized τ, tardiness factor, and R, due date range. Therefore, the average loss found by GADOC is 13% for 15 orders and 18% for 10 orders for all machines is due to randomized chosen the tardiness factor and due date range of the datasets.

Similar results are showed by GATPC are because the characteristics of the datasets affect the result. However, the τ, tardiness factor, and R, due date range values are the same for the datasets with the same number of orders with different machine numbers. Therefore, the increase in UBD value is clearly can be seen in Table 3. For example, for the GADOC, with 15 orders on two machines the average UBD value is 8.7%, on three machines is 11.2%, on four machines is 14.9% and finally 17.3% for five machines. Therefore, as the number of machines increases, it was observed that the UBD value increases as a result of the narrowing in the due date range in the solution of the datasets where the number of orders is the same.

In medium-size problems, GADOC and GATPC offer solutions for order sets with 20 and 25 for all machine numbers. However, the FMM method solves with the GAMS

software cannot succeed in finding the optimal solution to any problem instances as shown in Table 4. For problem instances with 20 orders on five machines, the average UBD is 16.9% for the GADOC algorithm. Additionally, for problem instances with 25 orders and four machines the average UBD is 20.5% for the GADOC algorithm. For GATPC, the worst average UBD value on two machines with 20 orders is 22.3%, and on four machines with 25 orders average UBD value is 22.2%. In these instances, GADOC and GATPC showed weaker performance. However, the GADOC and the GATPC, with instances with 20 orders and two machines have the best performance with a minimum UBD value of 3.5% and 8.8%, respectively. Additionally, in instances with 25 orders and two machines the GADOC 10.4% reached the highest profit with the schedule among all machine numbers whereas for GATPC for 25 orders 11.2% on three machines.

Table 4. Upper bound deviation for n = 20, n = 25, and m = 2, 3, 4, 5 5 of the FMM, GADOC, and GATPC algorithms.

Order Number	Machine Number	Max	FMM Avg	Min	Max	GADOC Avg	Min	Max	GATPC Avg	Min
20	2		NA		43.7	15.6	3.5	49.1	22.3	8.8
20	3		NA		21.6	13.2	7.0	24.8	17.4	10.8
20	4		NA		23.9	16.1	9.4	24.3	17.0	9.8
20	5		NA		24.3	16.9	8.1	27.0	18.7	8.9
Average			NA		28.4	15.5	7.0	31.3	18.9	9.6
25	2		NA		23.8	15.9	10.4	29.3	20.5	14.1
25	3		NA		21.8	18.3	14.0	23.2	16.6	11.2
25	4		NA		23.4	20.5	17.5	24	22.2	20
25	5		NA		25.5	19.9	13.6	27.4	22	18.1
Average			NA		23.6	18.7	13.9	26	20.3	15.9

NA Not Available.

It is observed that as the number of machines increases with the same number of orders UBD value generally increases because of the due dates of orders in datasets. For both algorithms, the average maximum UBD value is higher for an order of 25 compared to instances of 20 orders. However, comparing the average minimum mean deviations in medium-sized datasets, it is seen that it is lower in the dataset of 20 orders than 25 orders. The GADOC provided a better solution than the GATPC algorithm in solving any sample set except samples in 25 orders and 3 machine systems. The best profit that can be gained for the GADOC and GATPC algorithm is 96.5% and 91.2%, respectively, on two machines with 20 orders in the medium dataset. The worst profit gained with the GADOC and GATPC algorithm is 56.3% and 50.9%, respectively, on the same group of datasets on two machines with 20 orders. This situation shows that the profit that can be obtained from a group of orders varies depending on the characteristics of the orders. Additionally, it shows that the profit that can be obtained from a group of orders varies depending on the characteristics of the orders. The fact that the highest and lowest profits are found in the same group dataset for both genetic algorithms proves this situation. Additionally, when we look at the average UBD values of the two genetic algorithms from the total income, GADOC performed better than GATPC for each number of machines in the medium size dataset. The GADOC algorithm has achieved an average profit of 83.3% and 74.3%, respectively, in datasets with order numbers of 20 and 25. These average profit values for the GATPC algorithm are 79.9% and 71.7%, respectively.

In large-size problems, GADOC and GATPC offered a solution for order sets with 50 and 100 for all machine numbers as shown in Table 5. However, as in the medium size instances, the FMM method did not succeed in finding the optimal solution to any problem instances in a large dataset. According to Table 5, the average of all machines UBD for 50 orders is 16.7% and for 100 orders is 25.7% for the GADOC algorithm. In other words, the GADOC achieved to solve the average total net revenue of 50 orders is 83.3% of total income and for 100 orders is 74.3%. As in small and medium datasets, in solving

large datasets, GATPC performed worse than GADOC, achieving an average UBD value of 20.1% and 28.3%, respectively.

Table 5. Upper bound deviation for n = 50, n = 100, and m = 2, 3, 4, 5 of the FMM, GADOC, and GATPC algorithms.

Order Number	Machine Number	FMM Max	FMM Avg	FMM Min	GADOC Max	GADOC Avg	GADOC Min	GATPC Max	GATPC Avg	GATPC Min
50	2		NA		38.2	21.3	3	40.4	24.3	5.8
50	3		NA		38.7	21.4	2	37.7	25.6	6.3
50	4		NA		14.6	7.9	2.1	17.6	10.9	5.8
50	5		NA		20	16.3	13.4	22.6	19.6	12.8
Average			NA		27.9	16.7	7.0	22.08	20.1	7.7
100	2		NA		28.1	24.1	22.2	40.5	24.2	5.8
100	3		NA		27.6	23.3	17.1	29.5	27.6	25.1
100	4		NA		30	25.9	24.5	32.8	28.7	24.6
100	5		NA		34	28.8	25.6	34.7	32.5	31.1
Average			NA		29.9	25.7	22.4	34.4	28.3	21.7

NA Not Available.

According to Table 5, for 100 orders the average UBD value increased from 24.7% to 25.7% of total income as the number of machines increased due to shrinking in the due date range of orders. The best net profit is obtained with GADOC from the 50 orders on three machine instances among large-size instances with 98% profit of total revenue. For GATPC, this profit is 94.2% of total revenue. In the same table, in the solution of large datasets, the maximum deviation from the profit is in 50 orders and two machine dataset with a value of 38.2%. In other words, approximately 38% less profit of the total income can be obtained with the scheduling by the GADOC algorithm. This value is 40.4% for GATPC. For both Algorithms, the average maximum and minimum, and average deviation of 100 orders of all machine numbers are higher than 50 orders. Especially the average minimum deviation of 100 orders is higher than 50 orders.

When all datasets are considered, the GADOC algorithm finds smaller UBD values than the GATPC algorithm. Even though FMM performs better in small datasets, it cannot solve larger datasets. As the number of machines increases, the UBD value is increased due to the decrease in the due date range because as the number of machines increases, the due date range decreases. If the due date range is small, the possibility of paying outsources or delay penalty for the order increases. Therefore, total net revenue decreases. Additionally, the reason why the UBD value does not increase directly as the number of orders in the datasets increases due to the random selection of τ, tardiness factor, and R, due date range.

By using the solution results of the datasets found in previous tables for each dataset and each number of machines, the deviation results were compared to more clearly reveal the performance difference between the two genetic algorithms. The improvement was calculated and the results are given in Table 6. The improvement of percentage is calculated with Equation (24).

$$\mathbf{100} \times \frac{\mathbf{Solution\ of\ GADOC} - \mathbf{Solution\ of\ GATPC}}{\mathbf{Solution\ of\ GATPC}} \tag{24}$$

In Table 6, the success of two different genetic algorithms which have the same operators except crossover is compared in solving the order acceptance and scheduling problem. The datasets that GADOC shows the best improvement compared to GATPC are in the dataset developed for 15 orders and 2 machines. The datasets that GADOC shows the worst improvement compared to GATPC are in the data set developed for 25 orders and 3 machines. In datasets with the same number of orders, as the number of machines increases, an overall decrease is observed in the improvement of GADOC compared to GATPC. The reason why the improvement does not increase as the number of orders increases may be also due to the randomly selected datasets in the previous study for generating datasets for this study.

Table 6. The improvement of GADOC over GATPC.

Order	Machine	GADOC % Improvement Over GATPC Solution
10	2	7.26
10	3	0.72
10	4	8.44
10	5	4.55
15	2	33.08
15	3	23.29
15	4	11.83
15	5	8.47
20	2	30.04
20	3	24.14
20	4	5.29
20	5	9.63
25	2	22.44
25	3	−10.24
25	4	7.66
25	5	9.55
50	2	12.35
50	3	16.41
50	4	27.52
50	5	16.84
100	2	0.41
100	3	15.58
100	4	9.76
100	5	11.38
Average		12.77

An increase is observed in the solution of datasets with the same number of orders and the different number of machines. In addition, the fact that the genetic algorithm finds the results randomly is another factor that affects this improvement. Therefore, the other reason for the differences in the improvement rate can be due to the randomized nature of the genetic algorithm while solving the datasets. The same conditions are applied to be able to compare the effect of the crossover operator. For example, the same number of initial populations and the same number of iterations are used while solving all datasets and one recurrence is made for each algorithm.

When the data in Table 6 is examined, the GADOC algorithm gives better results than GATPC in almost all of the datasets with an average of 12.77%. Therefore, the DOC operator solved the problem more effectively than the TPC operator. This result is because of the genetic sequencing difference rate between generations. The gene sequences in offspring's chromosomes are less similar to elite and non-elite chromosomes in the DOC operator. There is much more variation in the DOC method than in the TPC method in terms of the sequence of genes on the chromosome. Having individuals with very different genetic sequences with the DOC operator helps to scan wider and different points in the solution space. This increases the probability of the algorithm finding different and better results. Therefore, the DOC operator was able to solve the fuzzy order acceptance and scheduling problem more successfully than the TPC operator.

Another criterion that shows the solution performance of algorithms is the execution time. For this reason, the solution times of the GADOC and GATPC are also evaluated. The Execution Time of GADOC in Figure 6 and GATPC in Figure 7 on each order set is also examined.

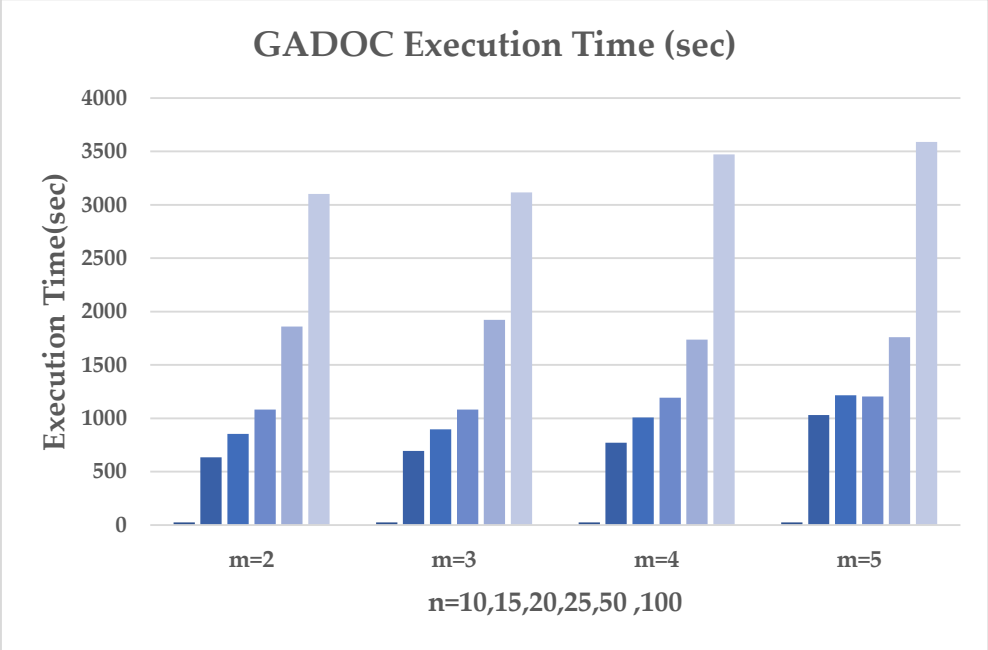


Figure 6. The average execution time (sec) of the GADOC for each machine m = 2, 3, 4, 5 and order number n = 10, 15, 20, 25, 50, 100.

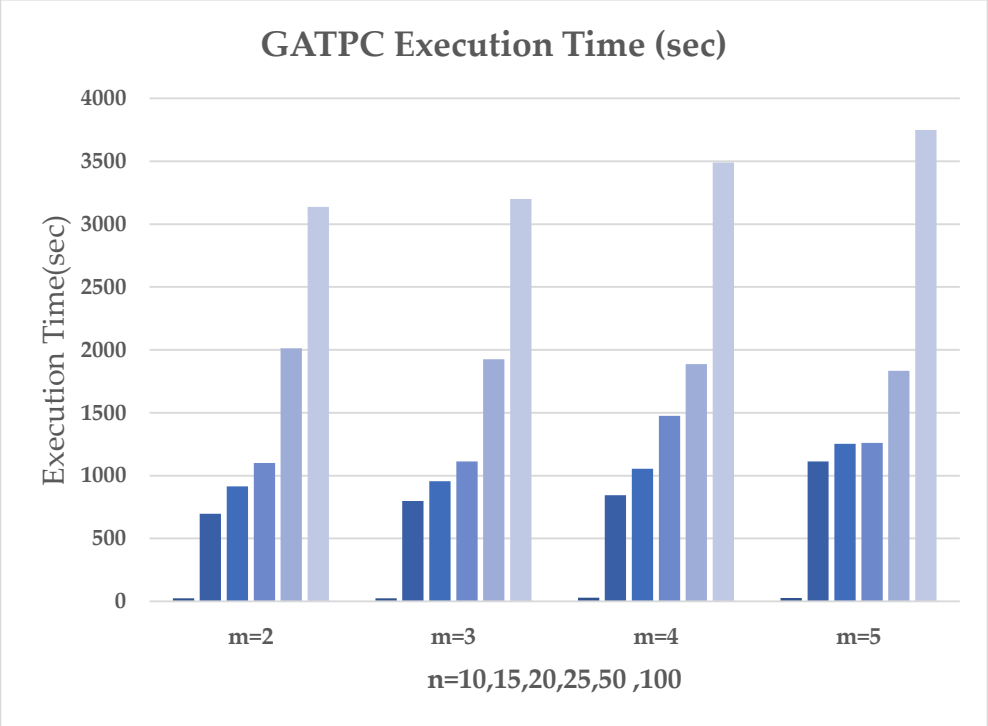


Figure 7. The average execution time (sec) of the GATPC for each machine m = 2, 3, 4, 5 and order number n = 10, 15, 20, 25, 50, 100.

The average solution times for each dataset and each machine of GADOC and GATPC algorithms are given in. The solution time of the GADOC is generally slightly shorter compared to the GATPC. Algorithms solve orders of 10 in as little as 20 s on average. The solution time changes in a few seconds as the number of machines increases. However, as the number of orders increases, the solution time of the algorithm also increases. For an order of 15, 1000 initial population and 100 iterations scheduling problem can be solved

in an average of 10 min for two machines in both algorithms, while it can solve the same problem in up to 16 min for four machines. When the number of orders increases to 20, it solves the problems in the two-machine system in an average of 18 min, and when the number of machines increases to 5, the solution time reaches 20 min for both algorithms. When the number of orders is 25, the solution time for two machines is 18 min on average, while the solution time for five machines increases to 24 min on average. In the case where the number of orders is 50, the GADOC algorithm solves the problem with two machines on average in 31 min for two machines. GATPC solves in an average of 33 min. With the increase in the number of machines, there is not much change in the solution time. In the solution of datasets where the number of orders is 100, on average, GADOC and GATPC are 51 min in a two-machine system, while the solution time increases up to 62 min with the number of machines increasing up to 5. Genetic algorithms can solve all numbers of the dataset according to FMM. A genetic algorithm, which finds a very good solution in 1 h even for 100 orders, is a very effective method, instead of the FMM method, which cannot find a solution for 15 orders in 10 h.

5. Discussion

This study seeks to answer the question of how maximum profit can be achieved by making proper schedules for the order acceptance and scheduling problem with including the outsourcing policy with the uncertainties in the identical parallel machine production systems. Since this problem is an optimization problem involving uncertainty, first a fuzzy mathematical model is developed to solve the problem with this exact method. As a result of solving the problem with a mathematical model, very successful schedules are obtained that can achieve a net high profit. However, the exact method can only succeed to solve small-scale problems. For this reason, a heuristic algorithm genetic algorithm was developed to handle larger-scale problems. To analyze the performance of different crossover operators in the genetic algorithm in the FOASIPM problem and the changes with the size of the problem, two different crossover methods were developed. When three methods are compared, the success of the FMM is quite better in finding scheduling with higher profit.

In solving small datasets, GADOC has shown a successful solution performance very close to FMM. GATPC showed the worst performance in solving small datasets. In larger datasets, FMM could not solve the problem. In larger datasets, when GADOC and GATPC solution performances were compared, GADOC provided better solutions to the problem by far. In addition, when the solution times were compared, GADOC found a faster solution to the problem than the GATPC.

When the deviations from total revenue are analyzed for FMM, the characteristics of the datasets are important when the deviation rates from the total income are evaluated. Datasets with randomized t and R values were used while creating the datasets. As the number of orders increases, the UBD values show a less increase instead of the number of orders increasing directly.

When regenerating the datasets for different machine numbers in the same order number, only due dates were changed and due dates were reduced accordingly as the number of machines increased. As the number of machines increases, a clearer increase in the UBD value is observed. When the number of machines increases in the same order number datasets, the decrease in profit is due to the increase in outsourcing or lateness penalties as a result of the narrowing in the due date range in datasets.

The same shift in results is observed with the heuristic algorithms GADOC and GATPC. In datasets with the same order number, the decrease in profit was observed as the number of machines increased. The reason why the same increase is not directly observed as the number of orders increases is that the t and R values are randomized when generating datasets with different order numbers. Nevertheless, as the number of orders increases, a decrease in profit is also observed. The reason is the expansion in solution space as the number of orders and machines increases. Algorithms must be able to quickly

scan the entire solution space to find the best of all possible solution methods. However, this is quite difficult for the algorithm in an NP-Hard problem. As the number of orders increases, the solution space expands too much. This situation is more clearly understood in the difference between the solution time of heuristic algorithms for small and large datasets. Heuristic algorithms can find solutions for small datasets in seconds, but finding solutions for large datasets takes a much longer time. In addition, as the number of orders in the datasets increases, the net profit rate in the solutions found in general decreases. This is because of many scheduling possibilities. As the solution space expands, it becomes challenging for the algorithm to find the best result.

In small datasets, FMM showed the best performance, while the second-best GADOC and GATPC were the worst performing algorithms. However, the success of the GADOC algorithm close to FMM also shows how accurate and successful the operators used in the genetic algorithm are. Using elitism and Davis order crossover methods in the genetic algorithm is quite successful in solving the problem of FOASIPM.

On the other hand, when both GA algorithms are compared, even all the steps are the same and only the crossover methods are different. The GADOC performs better than the GATPC algorithm. The crossover method difference affects the solution's performance and the algorithm's success. Therefore, this reflects the importance of the crossover method. The newly formed individuals in the DOC operator have a more different sequence from the parent's chromosomes compared to the TPC operator. Therefore, genetic variation between generations is greater in the DOC operator. Thus, the genetic algorithm with DOC operator can scan the space larger than the TPC operator. This ability of GADOC makes it more successful in solving the FOASIPM problem. Similar situations were observed in previous order acceptance and scheduling problems. Increasing the gene variation between generations in optimization problems with a large solution set increases the success of the heuristic algorithm in machine scheduling problems [40].

This study is a guide for companies with capacity constraints that want to meet the needs of production and marketing departments at the same time by making a schedule in a system with identical parallel machines. On the other hand, this paper sheds light on how outsourcing can be used with order acceptance and scheduling problems. In this paper with scheduling, the profit can be maximized and customer satisfaction can be increased by making on-time delivery. It can also be analyzed how the increasing number of orders and the number of machines in the system affect the difficulty of the problem. The success of the developed heuristic algorithm in solving the problem provides a guide solution method for solving similar problems. The success of the operators used in the genetic algorithm and the importance of operator selection are also clearly seen in this study. This paper is an important scheduling study that can shed light on future studies. The FOASIPM is a very important scheduling study because it deals with both time and profit in a manufacturing system. The uncertainties that exist in real life are included in this problem as fuzzy durations. On the other hand, in cases where the order cannot be accepted, the order can be delivered to the customer on time by outsourcing. The system is tried to be modeled close to a real production system. While establishing the model in this study, some assumptions have been made and some system parameters could not be included. However, of course, more realistic system parameters can be added to this problem in future studies such as machine breakdown times, and idle times. On the other hand, the revenue parameter of the orders can be modeled dynamically. It can also be added to the objective function as other factors that reduce profit such as inventory or holding cost.

6. Conclusions

The paper provides optimization and scheduling on the fuzzy order acceptance problem on identical parallel machines. Firstly, an order acceptance and scheduling problem is formalized as a mathematical model with the objective of the maximum net profit. In this problem, a fuzzy mathematical model enables optimization in terms of both profit and

time in the production schedules developed for real-life production planning. On the other hand, it also includes outsourcing penalty costs of orders. Fuzzy times that exist in real-life systems are handled in this model as fuzzy numbers. Thus, modeling order acceptance and scheduling problems including uncertainties and outsourcing options makes the problem more up-to-date and closer to real-life systems.

The FMM offers optimum scheduling successfully. However, this is possible for FMM in solving small datasets. To cope with this situation, the genetic algorithm, which is a heuristic algorithm, is used. Two different genetic algorithms using two different crossover methods are developed. Three solution methods are tested with small datasets and they are compared according to their solving performance and execution time. The upper bound deviation is used to compare the solving performance. The upper bound deviation is the deviation of the result from the total revenue of each order set. The FMM has the best solution for datasets with a slight difference from the GADOC algorithm. GATPC is an algorithm that solves the instances the worst. In large datasets, GADOC and GATPC are compared and the GADOC algorithm performs better. In this study, how the crossover methods affect the solution of FOASIPM is also demonstrated with two different crossover methods in genetic algorithms.

Author Contributions: Conceptualization, M.Z.E.B. and M.K.; methodology, M.Z.E.B. and M.K.; software, M.Z.E.B.; validation, M.Z.E.B. and M.K.; formal analysis, M.Z.E.B. and M.K.; investigation, M.Z.E.B.; resources, M.Z.E.B.; data curation, M.Z.E.B.; writing—original draft preparation, M.Z.E.B. and M.K.; writing—review and editing, M.Z.E.B. and M.K.; visualization, M.Z.E.B. and M.K.; supervision, M.K.; project administration, M.K.; funding acquisition, M.K. Both authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are openly available in [FigShare] at [https://doi.org/10.6084/m9.figshare.14614866.v2] (accessed on 29 May 2021).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Slotnick, S.A. Order acceptance and scheduling: A taxonomy and review. *Eur. J. Oper. Res.* **2011**, *212*, 1–11. [CrossRef]
2. Og, C.; Salman, F.S.; Yalçın, Z.B. Order acceptance and scheduling decisions in make-to-order systems. *Int. J. Prod. Econ.* **2010**, *125*, 200–211. [CrossRef]
3. Wu, S.-C.; Lu, T.; Liu, S.-T. A Fuzzy Approach to Support Evaluation of Fuzzy Cross Efficiency. *Symmetry* **2021**, *13*, 882. [CrossRef]
4. Zimmermann, H.-J. *Fuzzy Sets, Decision Making, and Expert Systems*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 1987; pp. 15–44.
5. Coşar, C. Try not to be late!—The importance of delivery service in online shopping. *Organ. Mark. Emerg. Econ.* **2017**, *8*, 177–192.
6. Chakraborty, P.S.; Majumder, G.; Sarkar, B. An Approach To Manage Constraint Resource And Outsourcing Decision. *J. Sci. Ind. Res.* **2006**, *65*, 625–631.
7. Gupta, D.; Benjaafar, S. Make-to-order, make-to-stock, or delay product differentiation? A common framework for modeling and analysis. *IIE Trans.* **2004**, 529–546. [CrossRef]
8. Cesaret, B.; Oğuz, C.; Salman, F.S. A tabu search algorithm for order acceptance and scheduling. *Comput. Oper. Res.* **2012**, *39*, 1197–1205. [CrossRef]
9. Chaurasia, S.N.; Kim, J.H. An Artificial Bee Colony Based Hyper-heuristic for the Single Machine Order Acceptance and Scheduling Problem. In *Decision Science in Action*; Asset Analytics; Springer: Singapore, 2019; pp. 51–63.
10. He, L.; Guijt, A.; de Weerdt, M.; Xing, L.; Yorke-Smith, N. Order acceptance and scheduling with sequence-dependent setup times: A new memetic algorithm and benchmark of the state of the art. *Comput. Ind. Eng.* **2019**, *138*. [CrossRef]
11. Koyuncu, E. A Fuzzy Mathematical Model for Order Acceptance and Scheduling Problem. *Int. J. Math. Comput. Sci.* **2017**, *11*, 145–148.
12. Zandieh, M.; Roumani, M. A biogeography-based optimization algorithm for order acceptance and scheduling. *J. Ind. Prod. Eng.* **2017**, *34*, 312–321. [CrossRef]
13. Thevenin, S.; Zufferey, N.; Widmer, M. Order acceptance and scheduling with earliness and tardiness penalties. *J. Heuristics* **2016**, *22*, 849–890. [CrossRef]
14. Chen, S.-H.; Liou, Y.-C.; Chen, Y.-H.; Wang, K.-C. Order Acceptance and Scheduling Problem with Carbon Emission Reduction and Electricity Tariffs on a Single Machine. *Sustainability* **2019**, *11*, 5432. [CrossRef]

15. Wang, S.; Ye, B. Exact methods for order acceptance and scheduling on unrelated parallel machines. *Comput. Oper. Res.* **2018**, *104*, 159–173. [CrossRef]
16. Yavari, M.; Marvi, M.; Akbari, A.H. Semi-permutation-based genetic algorithm for order acceptance and scheduling in two-stage assembly problem. *Neural Comput. Appl.* **2019**, *32*, 2989–3003. [CrossRef]
17. Shabtay, D.; Gaspar, N.; Kaspi, M. A survey on offline scheduling with rejection. *J. Sched.* **2012**, *16*, 3–28. [CrossRef]
18. Wang, X.; Xie, X.; Cheng, T.C.E. Order acceptance and scheduling in a two-machine flowshop. *Int. J. Prod. Econ.* **2013**, *141*, 366–376. [CrossRef]
19. Emami, S.; Sabbagh, M.; Moslehi, G. A Lagrangian relaxation algorithm for order acceptance and scheduling problem: A globalised robust optimisation approach. *Int. J. Comput. Integr. Manuf.* **2015**, *29*, 535–560. [CrossRef]
20. Lin, F.; Zhang, X.; Cai, Z. Approximation Algorithms for Scheduling with Rejection on Two Unrelated Parallel Machines. *Int. J. Adv. Comput. Sci. Appl.* **2015**, *6*, 260–264. [CrossRef]
21. Wang, X.; Huang, G.; Hu, X.; Edwin Cheng, T.C. Order acceptance and scheduling on two identical parallel machines. *J. Oper. Res. Soc.* **2015**, *66*, 1755–1767. [CrossRef]
22. Lei, D.; Guo, X. A parallel neighborhood search for order acceptance and scheduling in flow shop environment. *Int. J. Prod. Econ.* **2015**, *165*, 12–18. [CrossRef]
23. Jiang, D.; Tan, J. Scheduling with job rejection and nonsimultaneous machine available time on unrelated parallel machines. *Theor. Comput. Sci.* **2016**, *616*, 94–99. [CrossRef]
24. Palakiti, V.P.; Mohan, U.; Ganesan, V.K. Order Acceptance and scheduling parallel machine environment with weighted completion Time. *Eur. J. Ind. Eng.* **2018**, *12*, 535–537. [CrossRef]
25. Wang, B.; Wang, H. Multiobjective Order Acceptance and Scheduling on Unrelated Parallel Machines with Machine Eligibility Constraints. *Math. Probl. Eng.* **2018**, *2018*, 1–12. [CrossRef]
26. Wu, G.-H.; Cheng, C.-Y.; Yang, H.-I.; Chena, C.-T. An improved water flow-like algorithm for order acceptance and scheduling with identical parallel machines. *Appl. Soft Comput.* **2018**, *71*, 1072–1084. [CrossRef]
27. Esfandiari, S.; Mashreghi, H.; Emami, S. Coordination of Order Acceptance, Scheduling, and Pricing Decisions in Unrelated Parallel Machine Scheduling. *Int. J. Ind. Eng. Prod. Res.* **2019**, *30*. [CrossRef]
28. Naderi, B.; Roshanaei, V. Branch-Relax-and-Check: A tractable decomposition method for order acceptance and identical parallel machine scheduling. *Eur. J. Oper. Res.* **2020**, *286*, 811–827. [CrossRef]
29. Kong, M.; Pei, J.; Liu, X.; Lai, P.-C.; Pardalos, P.M. Green manufacturing: Order acceptance and scheduling subject to the budgets of energy consumption and machine launch. *J. Clean. Prod.* **2020**, *248*. [CrossRef]
30. Liu, P.; Lu, X. New approximation algorithms for machine scheduling with rejection on single and parallel machine. *J. Comb. Optim.* **2020**, *40*, 929–952. [CrossRef]
31. Karakaş, E.; Özpalamutçu, H. A genetic algorithm for fuzzy order acceptance and scheduling problem. *Int. J. Optim. Control. Theor. Appl. (IJOCTA)* **2019**, *9*. [CrossRef]
32. Ouelhadj, D.; Petrovic, S. A survey of dynamic scheduling in manufacturing systems. *J. Sched.* **2008**, *12*, 417. [CrossRef]
33. Stadje, W. Selecting jobs for scheduling on a machine subject to failure. *Discret. Appl. Math.* **1995**, *63*, 257–265. [CrossRef]
34. Zadeh, L.A. Fuzzy sets. *Inf. Control* **1965**, *8*, 338–353. [CrossRef]
35. Bector, C.; Chandra, S. *Fuzzy Mathematical Programming and Fuzzy Matrix Games*; Springer; Berlin/Heidelberg, Germany, 2005; Volume 169.
36. Syed, J.K.; Aziz, L.A. Fuzzy Inventory Model without Shortages Using Signed Distance Method. *Appl. Math. Inf. Sci.* **2007**, *1*, 203–209.
37. Yao, J.-S.; Wu, K. Ranking fuzzy numbers based on decomposition principle and signed distance. *Fuzzy Sets Syst.* **2000**, *116*, 275–288. [CrossRef]
38. Rivera, G.; Cisneros, L.; Sánchez-Solís, P.; Rangel-Valdez, N.; Rodas-Osollo, J. Genetic Algorithm for Scheduling Optimization Considering Heterogeneous Containers: A Real-World Case Study. *Axiom* **2020**, *9*, 27. [CrossRef]
39. Rani, S.; Suri, B.; Goyal, R. On the Effectiveness of Using Elitist Genetic Algorithm in Mutation Testing. *Symmetry* **2019**, *11*, 1145. [CrossRef]
40. Rom, W.O.; Slotnick, S.A. Order acceptance using genetic algorithms. *Comput. Oper. Res.* **2009**, *36*, 1758–1767. [CrossRef]