processes

MDPI

Article

Application of Non-Dominated Sorting Genetic Algorithm (NSGA-II) to Increase the Efficiency of Bakery Production: A Case Study

Majharulislam Babor [1,*], Line Pedersen [2], Ulla Kidmose [2], Olivier Paquet-Durand [1] and Bernd Hitzmann [1]

[1] Institute of Food Science and Biotechnology, Department of Process Analytics and Cereal Science, University of Hohenheim, 70599 Stuttgart, Germany
[2] Department of Food Science, Aarhus University, 8200 Aarhus N, Denmark
* Correspondence: majhar@uni-hohenheim.de

Abstract: Minimizing the makespan is an important research topic in manufacturing engineering because it accounts for significant production expenses. In bakery manufacturing, ovens are high-energy-consuming machines that run throughout the production time. Finding an optimal combination of makespan and oven idle time in the decisive objective space can result in substantial financial savings. This paper investigates the hybrid no-wait flow shop problems from bakeries. Production scheduling problems from multiple bakery goods manufacturing lines are optimized using Pareto-based multi-objective optimization algorithms, non-dominated sorting genetic algorithm (NSGA-II), and a random search algorithm. NSGA-II improved NSGA, leading to better convergence and spread of the solutions in the objective space, by removing computational complexity and adding elitism and diversity strategies. Instead of a single solution, a set of optimal solutions represents the trade-offs between objectives, makespan and oven idle time to improve cost-effectiveness. Computational results from actual instances show that the solutions from the algorithms significantly outperform existing schedules. The NSGA-II finds a complete set of optimal solutions for the cases, whereas the random search procedure only delivers a subset. The study shows that the application of multi-objective optimization in bakery production scheduling can reduce oven idle time from 1.7% to 26% while minimizing the makespan by up to 12%. Furthermore, by penalizing the best makespan a marginal amount, alternative optimal solutions minimize oven idle time by up to 61% compared to the actual schedule. The proposed strategy can be effective for small and medium-sized bakeries to lower production costs and reduce CO_2 emissions.

Keywords: bakery manufacturing; efficiency; multi-objective optimization; NSGA-II; no-wait flow shop

Citation: Babor, M.; Pedersen, L.; Kidmose, U.; Paquet-Durand, O.; Hitzmann, B. Application of Non-Dominated Sorting Genetic Algorithm (NSGA-II) to Increase the Efficiency of Bakery Production: A Case Study. *Processes* **2022**, *10*, 1623. https://doi.org/10.3390/pr10081623

Academic Editor: Dariusz Dziki

Received: 25 July 2022
Accepted: 13 August 2022
Published: 16 August 2022

1. Introduction

With the recent rapid growth of industrial automation, manufacturing processes have increased efficiency in resource usage while minimizing economic costs. Concurrently, carbon dioxide (CO_2), a significant byproduct of many manufacturing processes, strongly affects climate change and global warming. As a result, environmental policies, such as carbon emission reduction, have been extensively studied, in addition to cost reduction. The distribution and scheduling of processing tasks among energy-consuming machines directly affect overall energy consumption and, as a result, CO_2 emissions [1]. Much attention has been paid to finding optimized schedules for a production facility with the lowest possible makespan and energy consumption in this context.

The flow shop scheduling problem (FSSP) has been studied as one of the most effective decision-making tools in manufacturing processes. It was shown to be a non-deterministic polynomial-time (NP)-hard problem with a few exceptions [2]. The problem becomes more complicated as the number of machines in the various processing phases increases [3]. However, to achieve efficient production with an optimal schedule, FSSP has been widely

applied, irrespective of the type of manufacturing facility. In the simplified model, m machines operate a set of n products that is characterized as a combinatorial optimization problem in the literature. The permutation FSSP is a version of FSSP in which all the machines process each product in the same order.

A variant of FSSP, known as no-wait FSSP, refers to an arrangement where the waiting time between two consecutive processing stages of a product is unacceptable. The production environment for many products in the food, chemical, and pharmaceutical sectors is commonly considered a no-wait FSSP, such as bakery products. In such cases, the start time of the first operation of a product can be scheduled independently; however, the following stages are performed without interruption and delay. In practice, many flow shop problems are more complicated than just enforcing the no-wait rule. As a result, adjustments to the simplified flow shop model have been extensively studied to meet the needs of particular manufacturing processes, such as bakery [4–8], ice cream [9] and chemical processes [10] and pharmaceutical processes [11]. In bakeries, the processing route of products is identical in terms of the number and order of processing tasks, which is not the case for the permutation flow shop. As a result, the scheduling problem in bakery manufacturing can be categorized as a hybrid no-wait FSSP. However, Hecker et al. [5] simplified this into a permutation flow shop and increased the production efficiency of a German bakery with 40 products by minimizing makespan by 8.6%. Huber and Stuckenschmidt [8] only optimized the schedule for baking to serve customers in a retail store with freshly baked goods. In their investigation of a small-scale Spanish bakery production line, Babor et al. [4] observed that the existing production schedule is substantially inefficient. In addition to makespan, the authors took oven idle time into account and simplified the two objectives into a single objective using a linear weighting approach. Using this method, the authors optimized the investigated production to a satisfactory level while minimizing the makespan and oven idle time by 28% and 8%, respectively [4]. However, this method has drawbacks because the weighting factors are entirely determined by personal preference, which may result in a suboptimal solution. Additionally, the optimizer produces poor results when attempting to solve multi-objective problems with non-convex Pareto fronts that are unknown beforehand.

The goals of optimized manufacturing, which have been extensively studied in the literature, are the minimization of the makespan, total tardiness, and total flow times [12–17]. A variant of FSSP, known as the green flow shop scheduling, has an environmental criterion, such as carbon emissions. Qu et al. [18] explored the trade-offs between makespan and energy usage. Lu et al. [19] studied a permutation flow shop problem to minimize the total energy consumption using a hybrid multi-objective algorithm. Li et al. [20] proposed rescheduling an ongoing production using machine learning and optimization to deal with real-time exceptions. Lu and Qiao [21] used an adaptive evolutionary algorithm to address a hybrid flow shop scheduling problem and showed that the proposed technique effectively lowers unnecessary energy usage. Similarly, numerous studies had reducing energy consumption in manufacturing plants as an objective of scheduling optimization [22–24].

1.1. Motivation

Although many studies have been conducted on production scheduling problems, only a few authors have contributed to bakery manufacturing [4–6,8,25,26]. According to reports, small and medium-sized bakeries in EU countries still lack optimal production, and bakers primarily plan production schedules based on previous experiences [4,5]. Due to the poorly optimized production planning in small and medium-sized bakeries, large bakeries with modern manufacturing technologies substantially dominate the market. This provides a vast opportunity to explore production efficiency and make small and medium-sized bakeries competitive in the market.

1.2. Novelty

Minimizing makespan is the main focus in bakery manufacturing because it accounts for the majority of the production costs. The baking stage consumes the largest portion of the energy, ranging from 26% to 78% depending on the product category [27]. Saving production costs by means of minimizing both makespan and energy waste due to oven idle time increases the profit margin. A few studies considered the machine idle time when aiming to optimize the production cost [5,28]. In practice, the manufacturing facility and production infrastructure determine the constraints of the flow shop model. To the best of our knowledge, no previous research has explained the mathematical construction of the flow shop model for bakeries with constraints that bakers routinely follow.

1.3. Contributions

This study proposes a hybrid no-wait flow shop scheduling model (NWFSSM) for bakery manufacturing with a detailed description of the constraints of the mathematical model. Small and medium-sized bakeries have limitations when measuring and recording energy consumption data for machinery. Instead, we use the idle time of the machines that consume energy while performing no tasks. To the best of our knowledge, no previous research has implemented multi-objective optimization to reduce makespan and oven idle time (OIDT) in real bakery manufacturing scheduling problems. By performing the Pareto-based multi-objective optimization algorithm NSGA-II and random search procedure, the bakery production is optimized, resulting in a set of improved schedules. Additionally, the opportunity to reduce total manufacturing expenditure is explored by contrasting the trade-offs between makespan and OIDT.

2. Introduction to a Bakery Manufacturing Process

In bakeries, for most products, flour, water, yeast, and salt are mixed and kneaded to produce dough. The yeast is employed to initiate fermentation in the dough. In this process, in addition to aroma precursors, CO_2 is produced as a desired leavening agent to enhance the product texture and volume. Temperature and humidity significantly impact CO_2 production because they influence the yeast fermentation of sugars. Therefore, bakers determine the duration, temperature, and humidity of the fermentation chamber where the mixed and kneaded dough will be proved. The fermentation process is one of the most important quality determinants, and changes in process parameters affect the end product's quality. As a result, it is critical to stick to the time limits set for operations, including kneading, dough rest, and proofing [29,30]. The processing tasks follow a sequence, i.e., each task is a prerequisite to the following. Furthermore, permitting a task to continue for an additional duration implies overtreatment and a delay in the next; both are undesirable outcomes. Therefore, the next task begins without delay as soon as the previous one finishes.

Figure 1 shows the most straightforward processing route for bakery products. The manual transfer of unfinished items from one machine to another is called the transfer phase. Since the duration and machine settings for a task are predetermined and cannot be changed, we assume the energy consumption during its operation time to be constant, regardless of the schedule's level of optimization. However, the idle time changes based on the schedule, as does the energy consumption owing to this idle time.

There are two types of machines and equipment in a bakery: those that require no preparation time and those that need preparation time prior to an operation. The idle time of machines in the first group is unimportant because energy consumption during that phase is prevented by turning them off; this applies to most machines, such as kneaders. However, machines in the latter group run even when they perform no task. As the preparation time for these machines causes a delay in processing, doing so is convenient for bakers. A baking oven is an excellent example of the group that needs time to achieve the setpoint temperature before baking begins and is, therefore, left running throughout.

Reducing the idle time of such machines is of interest, as it directly influences the cost of production.

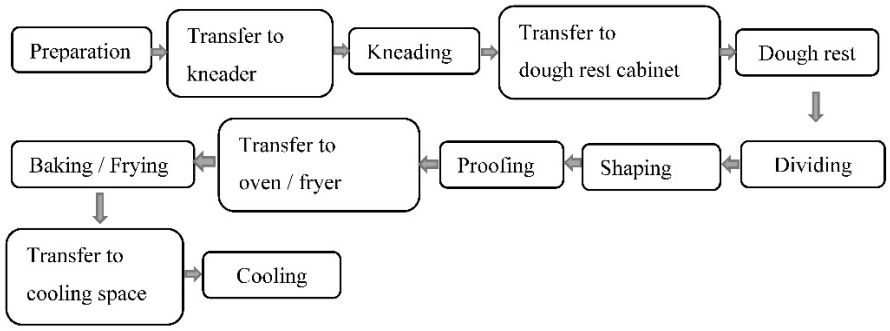

Figure 1. A simplified processing route for bakery products.

In practice, bakers aim to keep the makespan as short as possible because, like many other manufacturing processes, this accounts for a more significant part of the costs. It has been demonstrated that a production schedule with the shortest makespan does not mean that machines have the shortest idle time. Instead, many solutions produce the same makespan, but the idle time is unevenly distributed [4,6].

3. Materials and Methods

This paper investigates six bakery manufacturing problems (BMP) from a small-scale bakery in Denmark. Most of the bakery products in the problems are similar, with a few differences, such as adding and removing one or more products, which distinguishes them from each other. The computer language Python 3.7 [31] was used to implement the NWFSSM, as well as to perform the simulation and optimization on a computer running Windows 10 as the operating system with a configuration of an Intel Core i5 at 4 × 3.20 GHz, 8.00 GB ram.

3.1. Problem Definition and Modeling

The schematic diagram for the bakery production scheduling optimization procedure is shown in Figure 2. Information about products, machines, and personnel work schedules is required for bakery production scheduling optimization. For a product, the duration of each processing task and the machines that can carry out the task are fundamental to optimization. Furthermore, manual operations must be explicitly recorded to maintain a continuous process flow. Since the employees are limited in number, their working plan is necessary to allocate the task among them accordingly. Based on the collected information and any given production sequence, a schedule is simulated using proposed NWFSSM. To begin with, the bakery's actual production sequence is used to determine the current makespan and OIDT. The next stage involves running an optimization algorithm to find optimal solutions with a minimized makespan and OIDT. NWFSSM is used to simulate the production schedule each time the algorithm delivers a new production sequence. During the optimization process, the calculated makespan and OIDT are employed as quality criteria to contrast generated production sequences. The optimization is carried out until a stopping criterion is met.

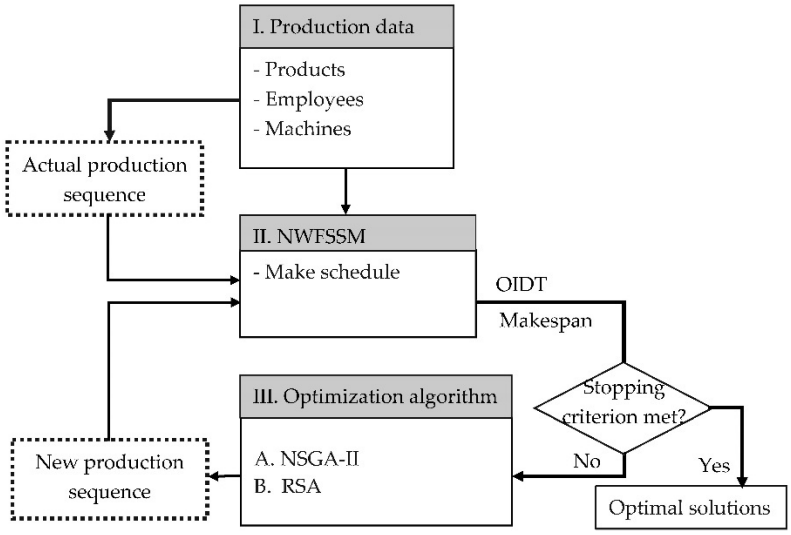

Figure 2. Schematic flow diagram of bakery production optimization.

In bakeries, many machines are often available to complete a task, such as a stone oven, oven chamber A, and oven chamber B for baking. The duties assigned to the machines are determined by their functionality and the product requirement. As a result, a product that requires a baking chamber can be assigned to either oven chamber A or oven chamber B rather than the stone oven. Sometimes the dough of more than one product is prepared and kneaded to take advantage of the same ingredients and resource capacity. However, after completing a few tasks, the dough separates to conduct additional treatments, which are distinct from one another. There is no general procedure for separating dough, as it depends on the product recipe, machine functionality, and machine capacity in the following stages. Since the dough for the products is common, none of them can be scheduled independently at any time. Therefore, it is convenient to organize the bakery products into different groups (G) based on predecessor constraints.

Table 1 shows an example of production data for one group with one pre-product and three products (P). Each product in a group has a bowl processing time, indicating how long it must wait after the processing of the group has started. One product in a group has no waiting time, meaning it has no predecessor and, thus, can start at any time.

Table 1. A simplified structure of data for one product group.

Group (*G*)	Product (*P*)	Bowl Time (*W*) [min]	Product Name	Stage (*s*)	Stage Name	Processing Time [min]	Machine (*M*)/ Employee (*E*)
1	1	0	Pre-product	1	Preparation	8	Employee
				2	Kneading	17	Kneader
	2	25	Product A	1	Dividing	10	Divider
				2	Shaping	17	Employee
				3	Proofing	50	Proving chamber
				4	Baking	35	Oven A
	3	35	Product B	1	Dividing	7	Divider
				2	Shaping	5	Employee
				3	Dough rest	33	Dough rest cabinet
				4	Rolling	32	Rolling machine
				5	Baking	18	Oven B
	4	47	Product C	1	Refining	6	Employee
				2	Dough rest	12	Dough rest cabinet
				3	Proofing	67	Proving chamber
				4	Baking	17	Oven B

The rest of the products in a group must wait for a certain period to initialize the processing. This waiting time specifies the time difference between two products within a group; however, there is no time difference between the two stages of a product. In the data shown in Table 1, the pre-product has no predecessor. Products A, B, and C, in contrast, can start after the pre-product with bowl process periods of 25 min, 35 min, and 47 min, respectively. Here, the bowl process period is the sum of the duration of preparation, kneading, and bowl resting period. Product A has no bowl resting period, as it begins right after the kneading is completed. In contrast, product B and product C have bowl resting periods of 10 min and 22 min, respectively. In practice, a manufacturing line has several product groups like this, which must be produced as efficiently as possible. The efficiency of production is determined by the order in which these groups are produced.

Figure 3 shows the schedule for a group of products using the data presented in Table 1. The dough is prepared and kneaded together before being split into several items, signifying that the pre-product is the predecessor for other items. Similarly, after the dividing for product A is completed, the dividing for product B begins, indicating the presence of a machine block. Product C has a recipe-based predecessor and starts after shaping for product A is completed. Meanwhile, the processing of the products runs in parallel, without waiting until the final stage is completed.

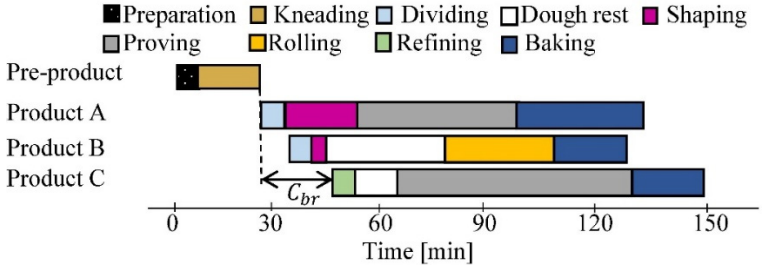

Figure 3. A schedule for a simplified group of products that share the same dough. C_{br} shows the bowl resting time for product C.

In another scheme, despite the initial treatments being wholly separate and different, two or more unfinished products are later merged. In practice, it is inefficient to run an energy-consuming device when part of its capacity is unused, which wastes energy. Therefore, bakers combine goods from various processing routes, but have similar machine requirements to perform the following task: baking in the same oven, at the same temperature, for the same amount of time. In such instances, hybrid NWFSSM should arrange the schedule so that the start time for the combined stage is the same. Figure 4 illustrates the schedule of three products that are to bake together. Ideally, in the flow shop model, one machine can perform one task of one product at a time. However, to meet the demands of an actual situation, we consider that an oven can simultaneously perform the same task for several products in the same group. The hybrid NWFSSM is explained in Appendix A.

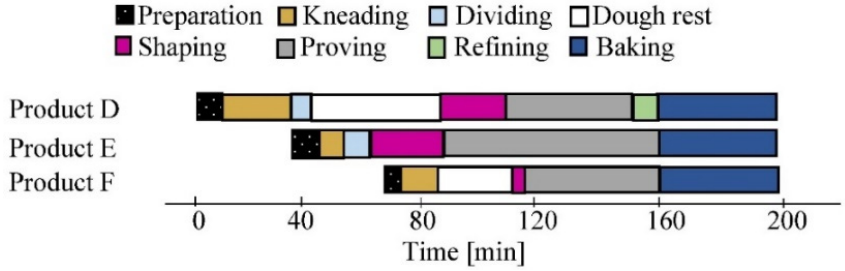

Figure 4. A schedule for one simplified group of products baked in one oven compartment at the same time.

Multi-objective optimization is applied in many real-world problems when a process must fulfill multiple criteria or objectives that conflict with one another. The advantage is

that several optimal solutions show distinct tradeoffs between the objectives that facilitate decision-making. Given a decision space χ, mapped into $\mathbb{R}$ for q objective functions $f_1 : \chi \to \mathbb{R}, \ldots, f_q : \chi \to \mathbb{R}$, a multi-objective optimization problem can be stated as follows.

$$min\ f_1(x), \ldots,\ min\ f_q(x);\ x \in \chi\ and\ q > 1 \tag{1}$$

where $f_1(x), \ldots, f_q(x)$ are conflicting objective functions such that satisfying one function can result in other functions being unsatisfied.

Pareto dominance is a fundamental idea in multi-objective optimization that precisely describes objective values. The benefit of using Pareto dominance is that it eliminates the need for additional information from the problem set when comparing objective vectors. To define the Pareto dominance, given two vectors in the decision space, $\vec{\mathbf{a}} = \{a_1, \ldots, a_q\}$ and $\vec{\mathbf{b}} = \{b_1, \ldots, b_q\}$ and $\vec{\mathbf{a}}$ is said to dominate $\vec{\mathbf{b}}$ ($\vec{\mathbf{a}} \preccurlyeq \vec{\mathbf{b}}$) if and only if $\vec{\mathbf{a}}_{\,\mathrm{d}} \leq \vec{\mathbf{b}}_{\,\mathrm{d}}$ for every $\mathrm{d} \in \{1, \ldots, \mathrm{q}\}$ and $\vec{\mathbf{a}}_{\,\mathrm{d}} < \vec{\mathbf{b}}_{\,\mathrm{d}}$ for at least one of $\mathrm{d} \in \{1, \ldots, \mathrm{q}\}$. In words, $\vec{\mathbf{a}}$ dominates $\vec{\mathbf{b}}$, if $\vec{\mathbf{a}}$ is not worse in any objective and better in at least one objective than $\vec{\mathbf{b}}$ [32,33].

The Pareto efficiency, also known as Pareto optimality, refers to the solutions in a decision space, where improving one of the objectives causes at least one of the others to deteriorate. The collection of Pareto optimal solutions expressing the best trade-offs between the objectives are known as non-dominated solutions, forming the Pareto frontier (PF). Figure 5 describes the concept of the Pareto frontier where solutions A, B, and C are Pareto optimal solutions and represents PF for two objective functions, f_1 and f_2. Solution D is dominated by solution B because it improves f_1 while not worsening f_2, and thus is not a Pareto optimal. The Pareto dominance operator can be used to separate weak solutions like D from the solutions in PF.

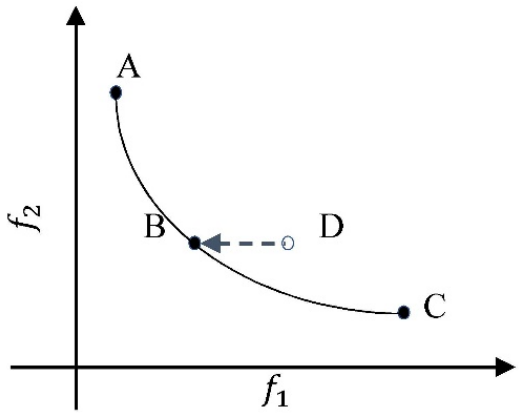

Figure 5. Pareto frontier for two objective functions of the minimization problem.

3.2. Optimization Algorithms

A product group sequence is a set of discrete numbers that expresses a solution to a problem, such as $\{1, 2, \ldots, N\}$ for N groups. Each number indicates a product group, and the order in which the numbers appear in the sequence indicates when the product's process begins. From a mathematical standpoint, for N product groups, there are $N!$ potential solutions, with each consisting of N unique integers.

In Appendix B, Algorithm A1 shows the structure of NSGA-II that finds a set of Pareto optimal solutions [34]. Many studies have performed this algorithm as a classical approach to solve multi-objective optimization problems [35–38]. Figure 6 illustrates the general procedure for NSGA-II. Initially, it has a random population $(PO_{t=0})$ of size I where members of the population are called individuals $\{1, 2, \ldots, i, i+1, \ldots, I\}$. One individual is a sequence of product groups. Individuals are evaluated using NSFSSM to

find their fitness vectors, which are made up of objective values. The population is sorted into different ranks using the fast non-dominated sorting approach, which is explained later.

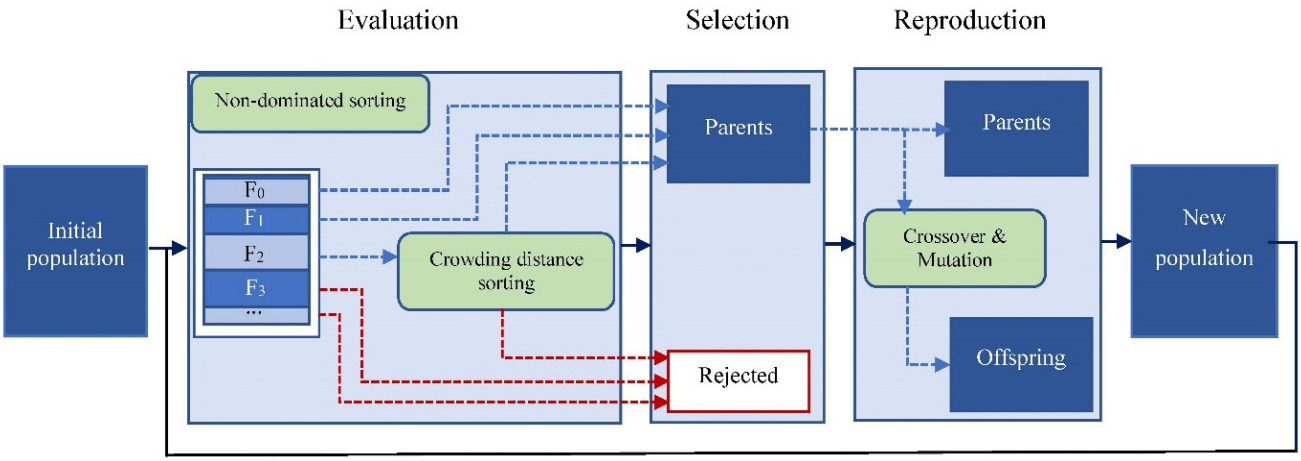

Figure 6. NSGA-II procedure. Non-dominated sorting divides the population into different ranks. F_0, F_1, F_2, ... are different ranks where a rank contains the individuals of the same front of optimality.

The offspring $(Q_{t=0})$ of the same size I is created using crossing over and mutation operators. A binary tournament selection routine is employed to find parents from the current population to construct an offspring individual. In this process, four random individuals from the current population $(PO_{t=0})$ are compared and the individual with the best rank is chosen as one of the parents. The same procedure is followed to select another parent to perform crossing-over and mutation. The combined population of size $2I$ $(R_{t=0} = PO_{t=0} + Q_{t=0})$ is formed. After evaluation, the population is sorted into different ranks. The best parent individuals for PO_t are chosen from the combined population. Since the combined population is double in size $(2I)$ compared to the actual population size (I), a screening procedure is employed. The selection criterion is the individuals' fitness, with the best rank taking priority.

If individuals at the best rank, such as F_0, are insufficient to fill the population slot, the nearest ranks are sequentially utilized until the population size is I. When a rank contains more individuals than the empty slots, it is difficult to establish the precise I size of the best population. Therefore, a crowding distance operator (Figure 7) that computes the distance between the individuals in one front is performed. The crowding distance for border solutions is set to infinity as an exception. The individuals are sorted in descending order of crowding distance to choose the solutions of higher crowding distance first and fulfill the remaining slot in the parent population PO_t. Figure 6 illustrates that the F_2 front, which carries more individuals than the empty slot for PO_t, falls under the crowding distance sorting.

The description of the non-dominated sorting operator is as follows. The population is sorted into different ranks, F_r where $r = \{0, 1, 2, \ldots\}$, based on the Pareto dominance operator (Figure 8). In other words, the fitness of individual i is compared with that of other individuals, $i + 1$, $i + 2$, ..., I. There are three conceivable outcomes when comparing two individuals; say i and $i + 1$: i dominates $i + 1$, $i + 1$ dominates i, or no one dominates none. Finally, the number of other individuals that dominate individual i is recorded as the sum of domination. The sum of domination determines the rank of i. The rank starts from F_0 and i can be a member of this rank if no other individuals dominate it (sum of domination is 0). The rank F_0 contains the individuals that provide Pareto optimal solutions to the problem. An increase in the value of r in F_r signifies a decline in Pareto strength. The individuals in F_1 forms an independent front; however, all of them are suboptimal compared to the individuals in rank F_0 (Figure 8). Similarly, based on the individual's dominance level, the entire population is ordered into distinct ranks. The larger an individual's sum of domination, the higher the r value for its front F_r, implying a

weaker solution to the problem. This approach is known as fast non-dominated sorting, which divides the population into separate fronts [34].

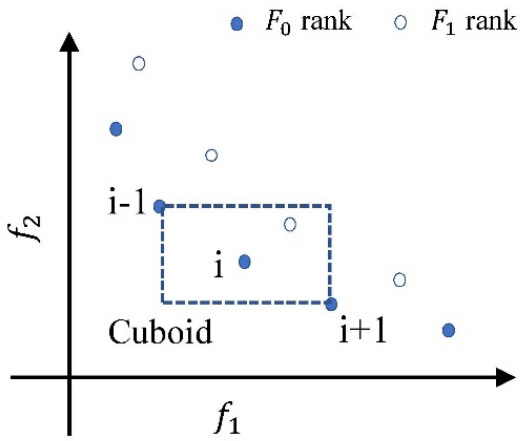

Figure 7. Crowding distance calculation for individual i. f_1 and f_2 represent two objectives. Filled circles show the individuals' fitness of the same non-dominated front F_0.

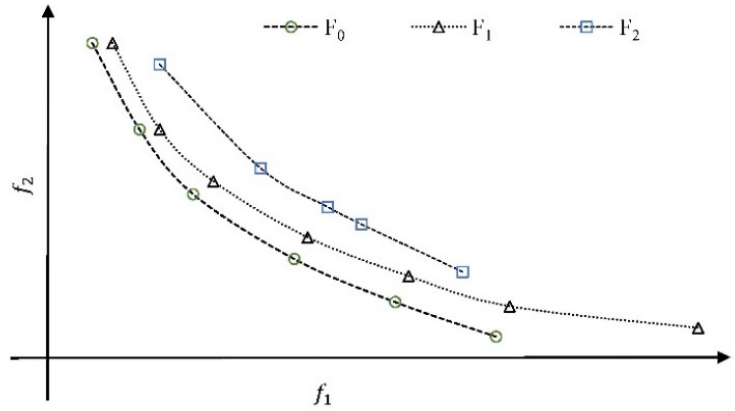

Figure 8. Concept of ranks using the fast non-dominated sorting approach. F_0, F_1, and F_2 are different ranks. Solutions in F_0 rank are Pareto optimal. Solutions in F_1 and F_2 are suboptimal and dominated by at least one of the other solutions.

The crossover operator in NSGA-II creates a new production sequence (offspring) from two separate parent sequences. Combining segments from distinct sequences allows for product groups to be repeated and eliminated, effectively breaking the solution technique. Figure 9 shows an example of the single-point crossover between parent solutions, X_1 and X_2 $(N = 7)$ using the proposed Algorithm A2, shown in Appendix B.

The offspring o_1 is taken from the parent X_1; only X_{1,SC_1} is randomly shuffled (Figure 9a). To form offspring o_2 (Figure 9b), X_{1,SC_1}, and X_{2,SC_2} should be taken. However, it is clear that $\{2, 3\}$ will be repeated and $\{5, 6\}$ will be removed from the offspring. Therefore, the elements in X_{2,SC_1} which are not present in X_{1,SC_1} with this order, are considered to avoid repeating the same numbers in O_2.

Furthermore, we apply swap and reversion mutation operators to two randomly selected regions to maintain a higher level of population variety (Figure 10). In swap mutation, two numbers in one individual are randomly chosen and swapped so that they exchange their locations. In contrast, all the numbers between two random points are reversely ordered in reversion mutation. The parameter settings of NSGA-II are shown in Table 2.

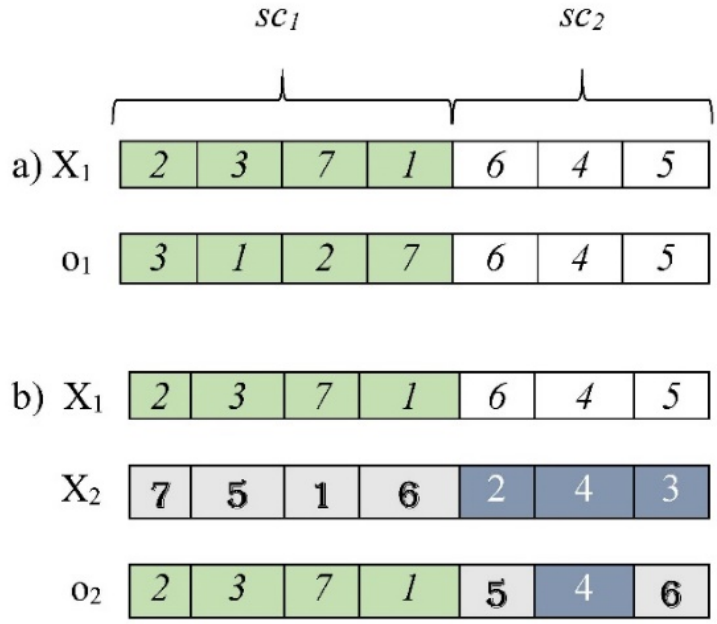

Figure 9. Concept of single-point crossover. X_1, X_2 are parent, and o_1, o_2 are offspring, as shown in (**a**,**b**) respectively. A random point 4 divides each parent into two sections, sc_1 and sc_2. X_{1,SC_1} is Section 1 of the parent X_1.

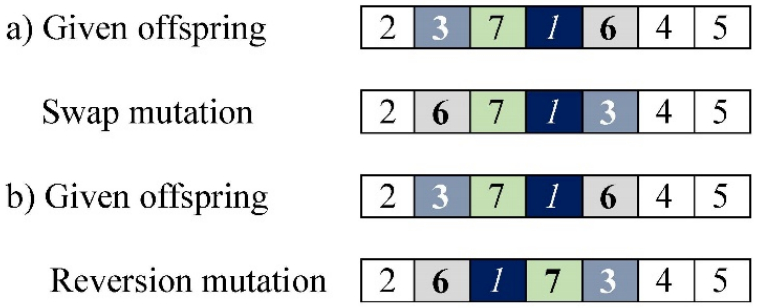

Figure 10. (**a**) Swap mutation and (**b**) reversion mutation for the same solution with two random points, 2 and 5.

Table 2. Parameter settings for NSGA-II.

Parameters	NSGA-II
Termination criteria	100 iterations
Population size	50
Selection	Binary tournament selection
Crossover	Single-point crossover
Crossover rate	1
Mutations	Swap and reversion mutation
Mutation rate	$\frac{1}{\text{Product groups}}$

The Pareto-based multi-objective random search algorithm (RSA) procedure is as follows.

- Step 1. Initialize the number of iterations and evaluate a random solution using hybrid NWFSSM. Save the fitness vector with makespan and OIDT in a repository of non-dominated solutions. The main loop of the random search algorithm starts from the next stage.
- Step 2. Generate a new random product group sequence and calculate its fitness using hybrid NWFSSM.

- Step 3. Using the Pareto dominance operator, compare the fitness of the new solution to the repository solutions. The new solution should only be added to the repository if none dominates it. Otherwise, discard the new solution.
- Step 4. Erase an old solution from the repository if the newly added solution dominates it.
- Step 5. Repeat the procedure from Step 2 to Step 4 to complete the number of iterations. The members of the repository are the set of Pareto optimal solutions for a given problem.

In 20 trials, employing each of the problems separately, it was observed that the Pareto fronts for the NSGA-II exhibited no improvement from 100 to an increase in the iteration for up to 500 iterations. In practice, production scheduling is routinely carried out, and finding optimized schedules within the shortest computational time is preferable. Therefore, in this study, NSGA-II was performed with a population size of 50 for 100 iterations, while RSA was performed with 5200 iterations to achieve approximately the same computational time. Depending on the problems, the computational time varied from 16 min to 20 min. The Pareto front (PF) for a problem is not initially known. Therefore, we take collective non-dominated solutions from the algorithms and refer to them as the PF to compare the performance of the individual algorithm.

The algorithms' performance was assessed using Equations (2) and (3), which calculate the closest proximity to the PF (CPF) [39] and the maximum spread of the solutions in the front (MSF) [40], respectively. A smaller CPF value indicates that the front of an algorithm is near the PF and thus performs better. In contrast, a higher MSF value indicates the better performance of an optimization algorithm.

$$CPF = \frac{\sqrt{\sum_{l=1}^{L} D_l^2}}{L} \tag{2}$$

where L is the number of solutions in the Pareto front and D_l is the Euclidean distance between the l^{th} solution and its nearest solution in the front found by the Algorithm that is under evaluation.

$$MSF = \left[\frac{1}{q} \sum_{i=1}^{q} \left[\frac{min\left(f_i^{max}, F_i^{max}\right) - max\left(f_i^{min}, F_i^{min}\right)}{F_i^{max} - F_i^{min}} \right]^2 \right]^{1/2} \tag{3}$$

where q is the number of objectives, f_i^{max} and f_i^{min} are the maximum and minimum values of the i^{th} objective in PF, respectively, F_i^{max} and F_i^{min} are the maximum and minimum values of the i^{th} objective in the front, provided by the algorithm under evaluation.

The conversion rate (CR) from best makespan to $OIDT$ reduction in alternative Pareto solutions is calculated using Equation (4).

$$CR = \frac{\frac{OIDT_{bms} - OIDT_{aps}}{OIDT_{bms}}}{\frac{MSP_{aps} - MSP_{bms}}{MSP_{bms}}} \tag{4}$$

where $OIDT_{bms}$ and MSP_{bms} are oven idle time (OIDT) and makespan at the best makespan solution, respectively, and $OIDT_{aps}$ and MSP_{aps} are OIDT and makespan at an alternative Pareto solution point, respectively.

4. Results and Discussion

This section presents the actual state of efficiency and optimization results obtained from the bakery. Table 3 shows the current state of efficiency of the schedules. The makespan and OIDT explain the similarities across the cases: the average makespan is 443 min, with a standard deviation of 13 min, while the average OIDT is 338 min, with a standard deviation of 31 min.

Table 3. The actual state of the efficiency of the production schedules, collected from the bakery.

Instance	Number of Products	Makespan [min]	OIDT [min]
BMP-I	36	419	333
BMP-II	42	442	328
BMP-III	39	446	341
BMP-IV	42	439	352
BMP-V	44	450	391
BMP-VI	48	461	285

Figure 11 shows the fitness of the individuals generated in one NSGA-II runtime. The color gradient here represents the distinct rank of individuals from 100 generations; a value of 0 in the color bar is the best rank. The population density near the best ranks is the lowest. The color gradient nearly shows the ideal shape for the Pareto frontier. However, the shape of the Pareto front, and, ultimately, the rank F_0, may differ in real problems.

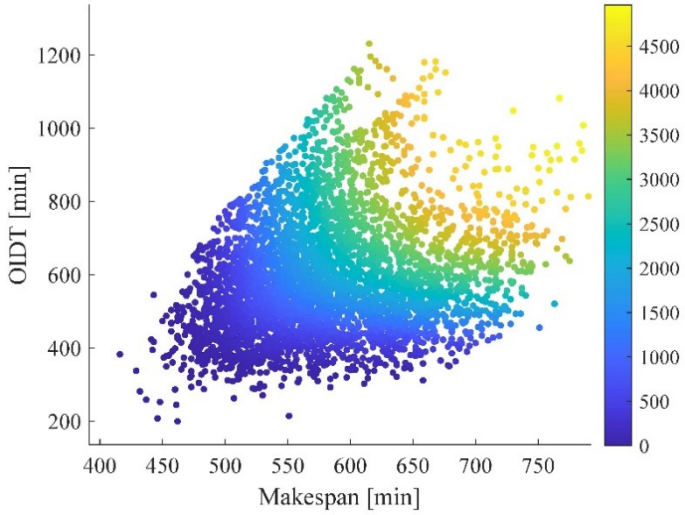

Figure 11. The fitness of the solutions generated during one runtime of NSGA-II for BMP V. The color gradient shows the order of the ranks.

Figure 12 displays the Pareto solutions obtained for BMP-I. The RSA only delivered two of the seven Pareto solutions, whereas the NSGA-II provided the entire Pareto front with seven solutions. Despite a 69 min rise in makespan, the drop in OIDT over the front was just 65 min. The solutions G2 from NSGA-II and R1 from RSA had the same makespan, but R1 showed a 14% higher OIDT (>120 min), demonstrating how obtaining a solution at a minimum makespan can contribute to energy waste. The optimal solution G1, which was closest to the existing solution and offered a reduction in makespan and OIDT of 12% and 27%, respectively.

The RSA generated more solutions in objective space for BMP-II than the NSGA-II, but none was optimal (Figure 13). In contrast, the NSGA-II presented all three PF-contributing solutions. Against a range of two times larger makespan, the Pareto line produced a difference of 46 min on the OIDT axis, indicating that the gain in OIDT across the front is minimal. However, the existing solution was largely dominated by two solutions (G1, and G2), with G1 being the closest, reducing makespan by 13% and OIDT by 27%.

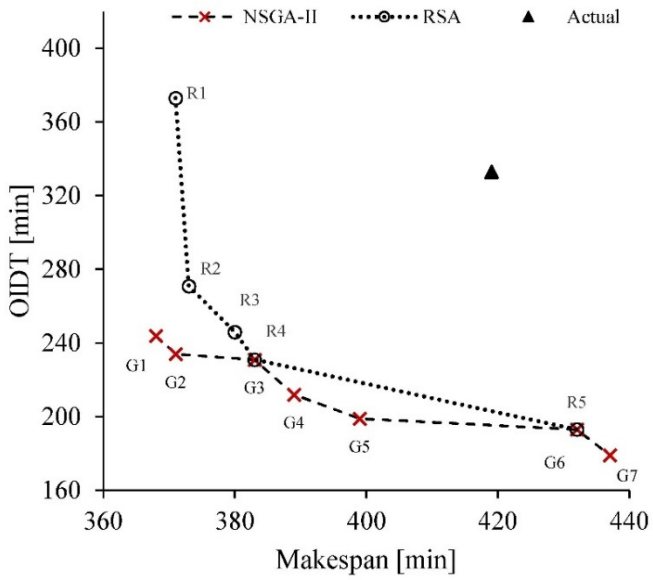

Figure 12. The solutions for BMP-I obtained by NSGA-II and RSA.

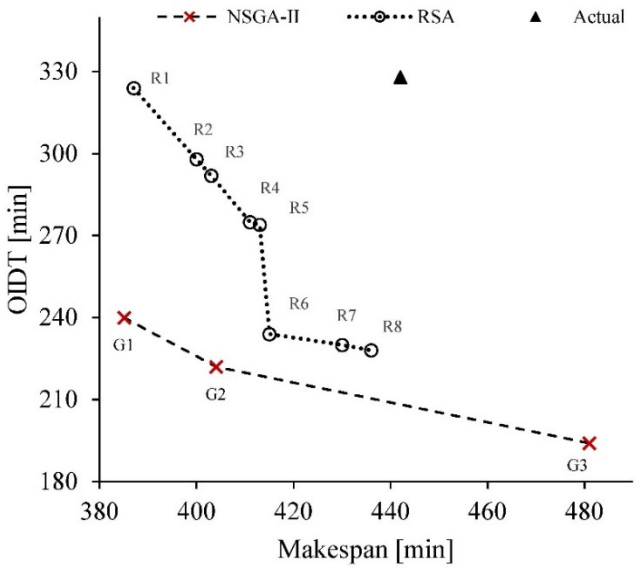

Figure 13. The solutions for BMP-II obtained by NSGA-II and RSA.

The actual BMP-II schedule was inefficient; compared to the closest Pareto solution G1, it had a 12% longer makespan and a 1.7% longer OIDT. The PF revealed that, compared to solution G1, solution G5 extends the makespan by 11 min while reducing OIDT by 52% and minimizing the makespan by 9.6% from the actual state.

Figure 15 shows BMP-IV solutions, illustrating the shortest front with only two Pareto alternatives within a range of 1 min makespan. The RSA failed to find any Pareto solution to this problem. However, results from NSGA-II showed that the OIDT achieved a significant 140 min reduction over the front, meaning that a vast gain in OIDT is possible. The results showed that, instead of a minimum makespan solution at G1, the solution with a minimum OIDT (G2) could be a better option, since it offers a 50% reduction in OIDT by losing only 1 min of makespan. Compared to the actual schedule, the solution at G2 improved this by 9.3% and 61%, respectively, in terms of makespan and OIDT.

The BMP-III solutions showed a sharp drop in OIDT (194 min) over the front (Figure 14). The NSGA-II provided six optimal solutions, with the RSA sharing only one.

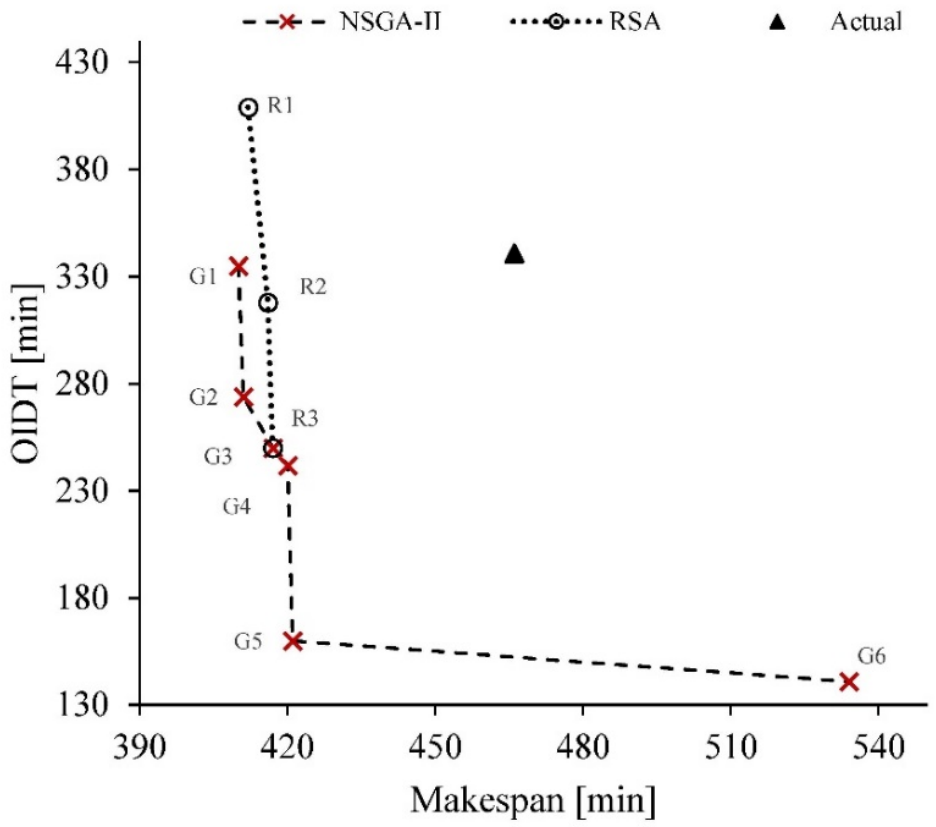

Figure 14. The solutions found by NSGA-II and RSA for BMP-III.

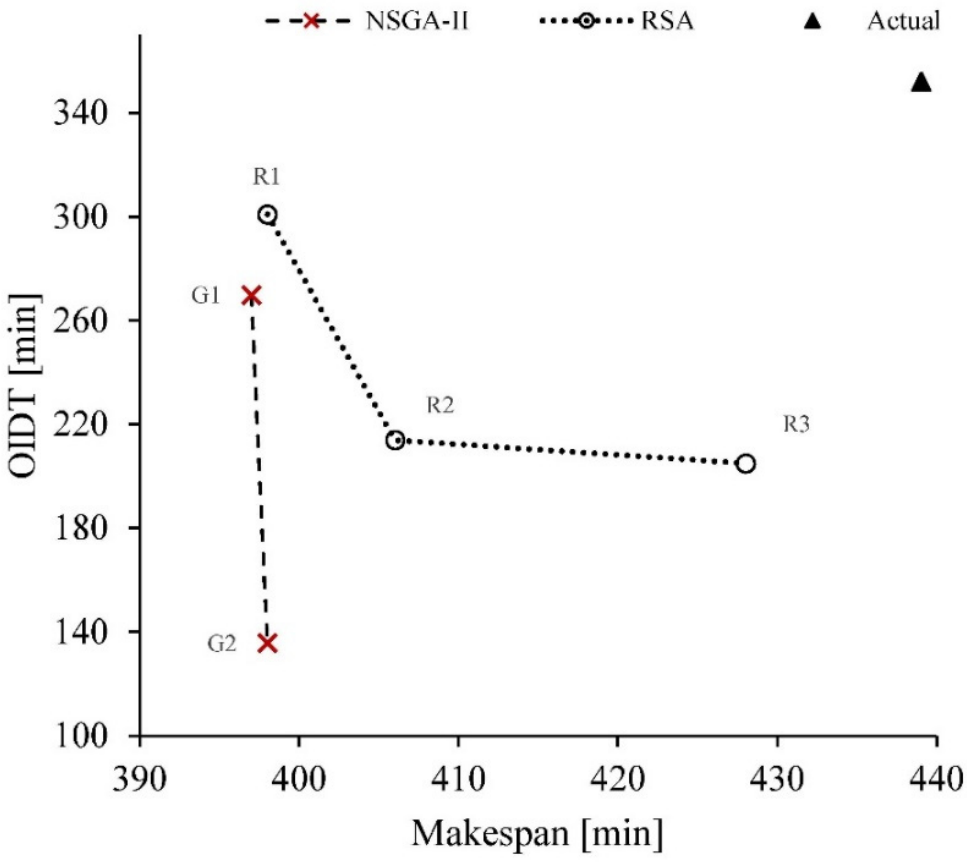

Figure 15. The solutions found by NSGA-II and RSA for BMP-IV.

With a Euclidean distance of 35 min, the real solution to the problem of BMP-V was close to the best makespan solution (G1) (Figure 16). The optimal point G1 improved efficiency by minimizing the makespan by 7.5% and OIDT by 2%. However, within a makespan range of 46 min, other front-end options reduced OIDT by up to 50%. Instead of finding an optimal solution with the smallest OIDT, the RSA generated a number of suboptimal options.

The NSGA-II presented six Pareto solutions for BMP-VI, represented in Figure 17. Despite being suboptimal, the existing schedule was the closest to one optimal solution (G4) with a slightly longer makespan (0.9%) and OIDT (9.8%). Compared to other production plans, the actual schedule for this problem shows the lowest OIDT. However, the obtained optimized solutions show that there are still alternative solutions that can improve the production efficiency. In comparison to the actual level of efficiency, solutions G2 and G5 both exhibited a substantial reduction in makespan and OIDT, respectively. Additionally, G2 demonstrated a considerable improvement in OIDT, while only slightly increased compared to the best makespan for this problem.

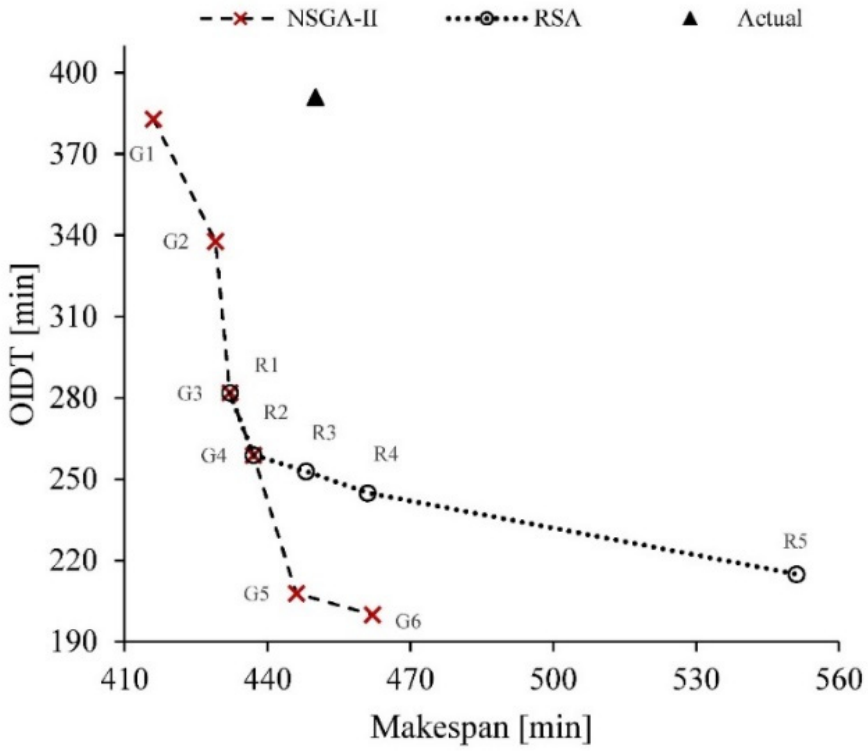

Figure 16. The solutions offered by NSGA-II and RSA for BMP-V.

RSA had the highest CPF, and lowest MSF for all the instances, meaning that NSGA-II outperformed this (Table 4). The NSGA-II had a CPF of 0 and an MSF of 1 for the problems, which indicates that it carried all the optimal solutions in its front, while RSA found a subset of them. For BMP-I and BMP-II, the random front was closest to NSGA-II; however, it failed to obtain even a single optimal solution for BMP-II.

The trade-offs between the decision variables, makespan, and OIDT, as determined by Equation (4), are shown in Table 5. To calculate the conversion rate from makespan to OIDT, the fitness of several solution points is compared to the best makespan solution (G1). In other words, by penalizing 1% of the best makespan, the data indicate the amount of gain in OIDT at each optimal point. The results reveal that solution G2 for BMP-I reduced OIDT by 5% for every percentage increase in makespan from the lowest makespan. The best conversion rate for BMP-II, BMP-III, BMP-IV, BMP-V, and BMP-VI was 1.5, 74.6, 197.0, 6.8, and 24.9, respectively. This demonstrates that the solution with the shortest makespan had

a larger OIDT for the instances. However, for most cases, the gain in OIDT was substantially more significant at the solution next to the best makespan.

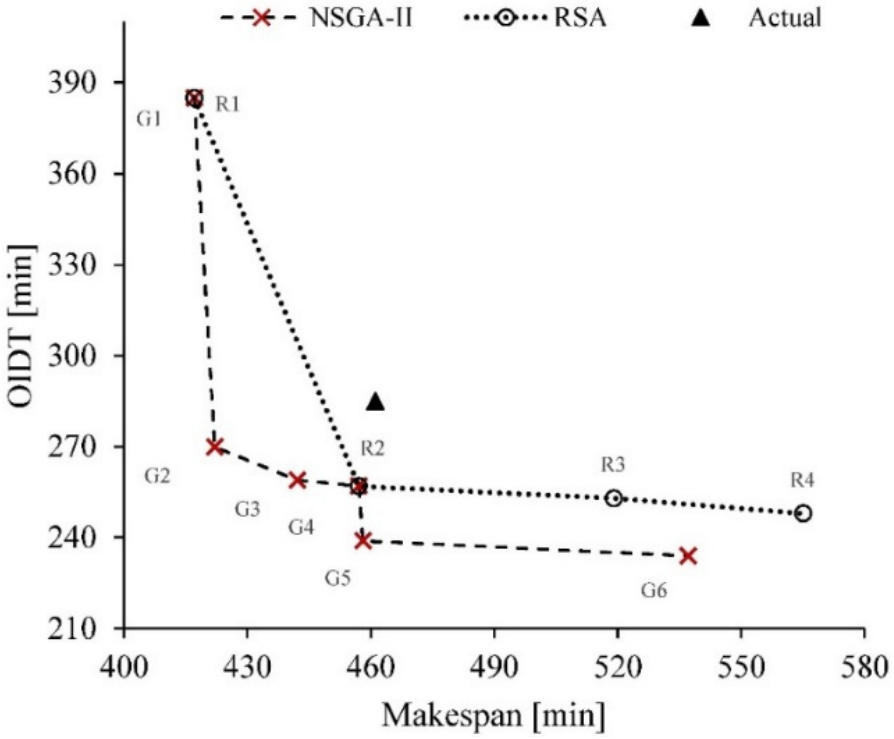

Figure 17. The solutions obtained by NSGA-II and RSA for BMP-VI.

Table 4. Performance comparison of algorithms.

Instances	CPF		MSF	
	NSGA-II	**RSA**	**NSGA-II**	**RSA**
BMP-I	0	18	1	0.73
BMP-II	0	8	1	0.71
BMP-III	0	100	1	0.80
BMP-IV	0	125	1	0.47
BMP-V	0	48	1	0.72
BMP-VI	0	24	1	0.90

Table 5. Tradeoffs between makespan and OIDT; gain in OIDT [%] by losing 1% of makespan relative to the best makespan solution (G1) obtained by NSGA-II.

Solution	BMP-I	BMP-II	BMP-III	BMP-IV	BMP-V	BMP-VI
G1	-	-	-	-	-	-
G2	**5.0**	**1.5**	**74.6**	**197.0**	3.7	**24.9**
G3	1.3	0.7	14.9	-	**6.8**	5.4
G4	2.3	-	11.4	-	6.4	3.4
G5	2.2	-	19.5	-	6.3	3.8
G6	1.2	-	1.9	-	4.3	1.3
G7	1.4	-	-	-	-	-

The studied problems were taken from the same bakery and had only a few changes in the range of products. However, the Pareto front and optimal solutions for them are substantially different. As a result, the tradeoffs between makespan and OIDT were found to be completely different from each other.

5. Conclusions

This study investigated six hybrid flow shop schedules from bakeries to obtain the shortest makespan and options for reducing energy waste. A hybrid flow shop model is proposed, which simulates and evaluates a scheduling solution while considering the real constraints. The multi-objective optimization methods, non-dominated sorting genetic algorithm (NSGA-II), and random search procedure were performed to find efficient production schedules.

NSGA-II found the optimal solutions with the best trade-off between makespan and OIDT for the instances. In contrast, RSA performed the worst, delivering a partially optimal set of solutions. The findings revealed that the investigated production schedule could be made more efficient by reducing the makespan by up to 12%. In most cases, the OIDT was drastically under-optimized, resulting in energy waste—a single-objective optimization with only makespan reduction may overlook this. By taking both makespan and OIDT into account, the optimizers can provide even more options and effective solutions for lowering manufacturing costs and CO_2 emissions. The trade-offs between objectives show that, by raising the best makespan by 1%, the gain in OIDT can be as high as 197. As a result, rather than focusing only on makespan reduction, a combination of makespan and OIDT reduction expands the opportunity to lower overall production costs.

The combined efficiency of multiple production lines in the same bakery can be evaluated using a distributed flow shop scheduling model as a reference for future investigations. The impact of exchanging a set of products between the production lines on overall production efficiency could be an interesting research topic.

Author Contributions: M.B. performed conceptualization, methodology, software, formal analysis, investigation, visualization, writing—original draft preparation. L.P. contributed with the resources and data curation. U.K. provided the resources and played the role of project administration. O.P.-D. contributed to conceptualization and methodology. B.H. supervised the study and played the role of project administration. All authors have read and agreed to the published version of the manuscript.

Funding: This study was funded by EIT Food of the European Institute of Innovation and Technology (EIT), a body of the European Union, the E.U. Framework Program for Research and Innovation for the project entitled "Optimization of bakery processes by a computational tool together with consumer feedback to minimize ecological footprint and food waste: Making baking more efficient".

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: This study was conducted using production data from a bakery in Denmark. Due to privacy and ethical concerns, the production data are not available. However, similar bakery production data from a different bakery have been published by Babor and Hitzmann [41], and are publicly accessible online.

Conflicts of Interest: The authors confirm having no known involvement in any organization with any financial interest in the subject and materials presented in this manuscript.

Appendix A

This section describes the formulation of a hybrid no-wait flow shop scheduling model (NWFSSM) for bakery manufacturing. Table A1 shows the notions that were used to explain NWFSSM.

Table A1. Nomenclature.

Notations	
Constants	
N	Number of group of products
n	Number of products in a group
m	Number of machines
e	Number of employees
Sets	
G	Set of product groups; $G = \{1, 2, \ldots, N\}$
P	Set of products in group g; $P = \{1, 2, \ldots, n\}$
M	Set of machines; $M = \{1, 2, \ldots, m\}$
V	Set of oven compartments; $V \subset M$
U	Set of machines with unlimited capacity; $U \subset M$
E	Set of employees; $E = \{1, 2, \ldots, e\}$
Indexes	
g	Index of groups; $g = 1, 2, \ldots, N$
p	Index of products in group g; $p = 1, 2, \ldots, n$
k	Index of machines; $k = 1, 2, \ldots, m$
l	Index of employees; $l = 1, 2, \ldots, e$
s	Index of the processing stage
t	Index of production runtime in minute.
Variables	
$W_{g,\,p}$	Time difference between product p and its predecessor in group g
$PT_{g,p,s}$	Processing time at stage s of product p in group g
$PT_{g,p,k}$	Processing time of product p in group g at machine k
$ST_{g,p,s}$	Start time for the operation at stage s of product p in group g
$CT_{g,p,s}$	Completion time of stage s of product p in group g
$start_k$	The time when machine k starts its first operation
end_k	The time when machine k finishes its last operation
$start_l$	The time when employee l starts the work.
end_l	The time when employee l finishes the work.
$O_{g,\,p,s,k}$	$\begin{cases} 1, & \text{if the product } p \text{ in group } g \text{ is processed on machine } k \text{ at stage } s \\ 0, & \text{if otherwise} \end{cases}$
$O_{g,\,p,s,l}$	$\begin{cases} 1, & \text{if the product } p \text{ in group } g \text{ is processed by employee } l \text{ at stage } s \\ 0, & \text{if otherwise} \end{cases}$

The makespan and oven idle time (*OIDT*) are calculated from the hybrid NWFSSM using Equations (A1) and (A2), respectively. In bakeries, one oven may have several compartments, which bakers use independently for different products. The sum of the idle time of all compartments is used to calculate *OIDT*. Hence, *OIDT* can be bigger than the makespan.

$$Makespan = max\left(CT_{g,p,s}\right) \quad \forall g \in G,\ \forall p \in P \tag{A1}$$

$$OIDT = \sum_{k=1}^{m}(end_k - start_k - \sum_{g=1}^{N}\sum_{p=1}^{n} pt_{g,p,k}\ ) \quad \forall g \in G,\ \forall p \in P, \forall k \in V \tag{A2}$$

The hybrid NWFSSM is described as follows.

$$Min\ (Makespan) \tag{A3}$$

$$Min\ (OIDT) \tag{A4}$$

which is subject to

$$ST_{g,1,1}\ \geq\ 0 \quad \forall g \in G \tag{A5}$$

$$ST_{g,p,1} = ST_{g,1,1} + W_{g,\,p} \quad \forall g \in G,\ \forall p \in P \setminus \{1\} \tag{A6}$$

$$PT_{g,\,p,s} > 0 \quad \forall g \in G, \quad \forall p \in P, \quad \forall k \in M \tag{A7}$$

$$CT_{g,p,s} = ST_{g,p,s} + PT_{g,p,s} \quad \forall g \in G, \quad \forall p \in P \tag{A8}$$

$$ST_{g,p,s+1} = CT_{g,p,s} \quad \forall g \in G, \forall p \in P \tag{A9}$$

$$\sum_{k=1}^{m} O_{g,\,p,s,k} \leq 1 \quad \forall g \in G, \ \forall p \in P, \ \forall k \in M \tag{A10}$$

$$\sum_{g=1}^{N} \sum_{p=1}^{n} \sum_{t=ST_{g,p,s}}^{CT_{g,p,s}} O_{g,p,s,k} \leq \ (CT_{g,p,s} - ST_{g,p,s}) \quad \forall g \in G, \ \forall p \in P, \ \forall k \in M \setminus (U \cup V) \tag{A11}$$

$$\sum_{g=1}^{N} \sum_{p=1}^{n} \sum_{t=ST_{g,p,s}}^{CT_{g,p,s}} O_{g,p,s,k} \leq \ n(CT_{g,p,s} - ST_{g,p,s}) \quad \forall g \in G, \ \forall p \in P, \ \forall k \in V \tag{A12}$$

$$\sum_{g=1}^{N} \sum_{p=1}^{n} \sum_{t=ST_{g,p,s}}^{CT_{g,p,s}} O_{g,p,s,k} \leq \ \sum_{g=1}^{N} n \quad \forall g \in G, \forall p \in P, \forall k \in U \tag{A13}$$

$$start_l < end_l \quad \forall l \in E \tag{A14}$$

$$\sum_{l=1}^{e} O_{g,\,p,s,l} \leq 1 \quad \forall g \in G, \ \forall p \in P, \ \forall l \in E \tag{A15}$$

$$\sum_{g=1}^{N} \sum_{p=1}^{n} \sum_{t=ST_{g,p,s}}^{CT_{g,p,s}} O_{g,p,s,l} \leq \ (CT_{g,p,s} - ST_{g,p,s}) \, \forall g \in G, \forall p \in P, \forall l \in E \tag{A16}$$

$$ST_{g,p,s} \geq \ start_l \quad \forall O_{g,p,s,l} = 1, \ \forall g \in G, \ \forall p \in P, \ \forall l \in E \tag{A17}$$

$$CT_{g,p,s} \leq \ end_l \quad \forall O_{g,p,s,l} = 1, \ \forall g \in G, \ \forall p \in P, \ \forall l \in E \tag{A18}$$

$$\sum_{k=1}^{m} O_{g,\,p,s,k} + \sum_{l=1}^{e} O_{g,p,s,l} \leq 1 \quad \forall g \in G, \ \forall p \in P, \ \forall k \in M, \ \ \forall l \in E \tag{A19}$$

The objective functions, minimization of makespan, and oven idle time (OIDT) are shown in Equations (A3) and (A4), respectively. Constraint (A5) implies that the start time for the predecessor product of any group can be ≥ 0. Constraint (A6) describes this for successor products in the group. Constraint (A7) ensures that the processing time for any stage is greater than 0 min. Conditions (A8) and (A9) guarantee the no-wait state between two consecutive product stages. Constraint (A10) indicates that an operation from a product can occupy only one machine. Except for the ovens $(k \in V)$ and the machine with unlimited capacity $(k \in U)$, any machine can only perform one task at a time, as defined by constraint (A11). In contrast, constraint (A12) relaxes ovens for multiple items from the same group. Constraint (A13) states that machines with unlimited capacity can perform any number of tasks at a time. Condition (A14) ensures that the working shift of employees is valid for task allocation. Constraints (A15) and (A16) limit the number of employees assigned to a single task and the number of tasks assigned to a single employee at any given time, respectively. Constraints (A17) and (A18) confine the task allocation among employees within their working time. Condition (A19) defines the limitation to occupying either an employee or a machine for a task. It is worth mentioning that a few tasks in the bakery process require no machine and employee, such as dough rest.

Appendix B

Algorithm A1. Non-dominated sorting genetic algorithm II (NSGA-II)

1: Initialize population PO_0, iteration $t = 0$
2: Fast non-dominating sorting
3: **while** $t \leq$ total iteration **do**
4: $Q_t = \varnothing$ //Initialize offspring population
5: **for** all $i \in \{1, 2, \ldots, I\}$ **do**//create offspring until the size of Q_t is I
6: Select parents (x_1, x_2) $x_1 \in PO_t$ *and* $x_2 \in PO_t$
7: $rc_t^i =$ recombine (x_1, x_2)//crossover
8: $mu_t^i =$ mutate $\left(rc_t^i\right)$//mutation
9: $Q_t = Q_t \cup mu_t^i$//insert new offspring in Q_t
10: $R_t = PO_t \cup Q_t$//combine parent and offspring population
11: F = fast non-dominated sorting (R_t)//F = $(F_1, F_2, \ldots)$ all non-dominated fronts of R_t
12: $PO_{t+1} = \varnothing$ *and* $r = 0$//Initializing new parent solutions and the best rank
13: **while** $|PO_{t+1}| + |F_r| \leq I$ **do**//until the parent solution is filled
14: Crowding distance operator (F_r)
15: $PO_{t+1} = PO_{t+1} \cup Fr$//insert individuals from ith non-dominated front in PO_{t+1}
16: $r = r + 1$
17: sort individuals in descending order
18: $PO_{t+1} = PO_{t+1} \cup Fr\,[1 : (I - |PO_{t+1}|)]$//add first $(I - |PO_{t+1}|)$ individuals of F_r
19: $t = t + 1$//increase the generation

Algorithm A2. Single-point crossover for product sequence

1: Select parents (x_1, x_2) $x_1 \in PO_t$ *and* $x_2 \in PO_t$

2: b = random number between 1 and N//random crossover point
3: $o_1 = o_2 = x_1[1 : b]$ //initializing offspring o_1 and o_2
4: $o_1 =$ random shuffle $(o_1) \cup x_1[b + 1 : N]$
5: $x_{2,SC_{1\_unique}} = \{x : x \in x_2[1 : b]$ and $x \notin o_2\}$//ignore repeating same element
6: $x_{2,SC_2} = x_2[b + 1 : N]$
7: t = 1
8: for all $e \in x_{2,SC_2}$ do
9: if $e \notin o_2$//if the element is not in o_2
10: $o_2 = o_2 \cup e$//include the element e in o_2
11: else
12: $o_2 = o_2 \cup x_{2,\,SC_{1\_unique}}[t]$//include the t-th element from $x_{2,SC_{1\_unique}}$ in o_2
13: $t = t + 1$

References

1. Ghorbani Saber, R.; Ranjbar, M. Minimizing the Total Tardiness and the Total Carbon Emissions in the Permutation Flow Shop Scheduling Problem. *Comput. Oper. Res.* **2022**, *138*, 105604. [CrossRef]
2. Gonzalez, T.; Sahni, S. Flowshop and Jobshop Schedules: Complexity and Approximation. *Oper. Res.* **1978**, *26*, 36–52. [CrossRef]
3. Vidal, G.H.; Hernández, J.R.C.; Minnaard, C. Modeling and Statistical Analysis of Complexity in Manufacturing Systems under Flow Shop and Hybrid Environments. *Int. J. Adv. Manuf. Technol.* **2021**, *118*, 3049–3058. [CrossRef]
4. Babor, M.; Senge, J.; Rosell, C.M.; Rodrigo, D.; Hitzmann, B. Optimization of No-Wait Flowshop Scheduling Problem in Bakery Production with Modified PSO, NEH and SA. *Processes* **2021**, *9*, 2044. [CrossRef]
5. Hecker, F.T.; Stanke, M.; Becker, T.; Hitzmann, B. Application of a Modified GA, ACO and a Random Search Procedure to Solve the Production Scheduling of a Case Study Bakery. *Expert Syst. Appl.* **2014**, *41*, 5882–5891. [CrossRef]
6. Babor, M.; Hitzmann, B. Application of Nature-Inspired Multi-Objective Optimization Algorithms to Improve the Bakery Production Efficiency. *Eng. Proc.* **2022**, *19*, 31.
7. Swangnop, S.; Duangdee, T.; Duangdee, J. Design of Production Planning Process for Bakery Manufacturer. In Proceedings of the 2019 IEEE 6th International Conference on Industrial Engineering and Applications (ICIEA), Tokyo, Japan, 12–15 April 2019; pp. 178–182.
8. Huber, J.; Stuckenschmidt, H. Intraday Shelf Replenishment Decision Support for Perishable Goods. *Int. J. Prod. Econ.* **2021**, *231*, 107828. [CrossRef]

9. Wari, E.; Zhu, W. A Constraint Programming Model for Food Processing Industry: A Case for an Ice Cream Processing Facility. *Int. J. Prod. Res.* **2019**, *57*, 6648–6664. [CrossRef]
10. Ahmed, H.; Ricardez-Sandoval, L.A.; Vilkko, M. Centralized and Hierarchical Scheduling Frameworks for Copper Smelting Process. *Comput. Chem. Eng.* **2022**, *164*, 107864. [CrossRef]
11. Ge, C.; Yuan, Z. Production Scheduling for the Reconfigurable Modular Pharmaceutical Manufacturing Processes. *Comput. Chem. Eng.* **2021**, *151*, 107346. [CrossRef]
12. Brum, A.; Ruiz, R.; Ritt, M. Automatic Generation of Iterated Greedy Algorithms for the Non-Permutation Flow Shop Scheduling Problem with Total Completion Time Minimization. *Comput. Ind. Eng.* **2022**, *163*, 107843. [CrossRef]
13. Gao, K.; Pan, Q.; Suganthan, P.N.; Li, J. Effective Heuristics for the No-Wait Flow Shop Scheduling Problem with Total Flow Time Minimization. *Int. J. Adv. Manuf. Technol.* **2013**, *66*, 1563–1572. [CrossRef]
14. Ravindran, D.; Selvakumar, S.J.; Sivaraman, R.; Haq, A.N. Flow Shop Scheduling with Multiple Objective of Minimizing Makespan and Total Flow Time. *Int. J. Adv. Manuf. Technol.* **2005**, *25*, 1007–1012. [CrossRef]
15. Samarghandi, H. Minimizing the Makespan in a Flow Shop Environment under Minimum and Maximum Time-Lag Constraints. *Comput. Ind. Eng.* **2019**, *136*, 614–634. [CrossRef]
16. Yu, A.J.; Seif, J. Minimizing Tardiness and Maintenance Costs in Flow Shop Scheduling by a Lower-Bound-Based GA. *Comput. Ind. Eng.* **2016**, *97*, 26–40. [CrossRef]
17. Han, Z.; Zhang, Q.; Shi, H.; Zhang, J. An Improved Compact Genetic Algorithm for Scheduling Problems in a Flexible Flow Shop with a Multi-Queue Buffer. *Processes* **2019**, *7*, 302. [CrossRef]
18. Qu, M.; Zuo, Y.; Xiang, F.; Tao, F. An Improved Electromagnetism-like Mechanism Algorithm for Energy-Aware Many-Objective Flexible Job Shop Scheduling. *Int. J. Adv. Manuf. Technol.* **2022**, *119*, 4265–4275. [CrossRef]
19. Lu, C.; Gao, L.; Li, X.; Pan, Q.; Wang, Q. Energy-Efficient Permutation Flow Shop Scheduling Problem Using a Hybrid Multi-Objective Backtracking Search Algorithm. *J. Clean. Prod.* **2017**, *144*, 228–238. [CrossRef]
20. Li, Y.; Carabelli, S.; Fadda, E.; Manerba, D.; Tadei, R.; Terzo, O. Machine Learning and Optimization for Production Rescheduling in Industry 4.0. *Int. J. Adv. Manuf. Technol.* **2020**, *110*, 2445–2463. [CrossRef]
21. Lu, H.; Qiao, F. An Efficient Adaptive Genetic Algorithm for Energy Saving in the Hybrid Flow Shop Scheduling with Batch Production at Last Stage. *Expert Syst.* **2022**, *39*, e12678. [CrossRef]
22. Busse, J.; Rieck, J. Mid-Term Energy Cost-Oriented Flow Shop Scheduling: Integration of Electricity Price Forecasts, Modeling, and Solution Procedures. *Comput. Ind. Eng.* **2022**, *163*, 107810. [CrossRef]
23. Cui, W.; Lu, B. Energy-Aware Operations Management for Flow Shops under TOU Electricity Tariff. *Comput. Ind. Eng.* **2021**, *151*, 106942. [CrossRef]
24. Lian, X.; Zheng, Z.; Wang, C.; Gao, X. An Energy-Efficient Hybrid Flow Shop Scheduling Problem in Steelmaking Plants. *Comput. Ind. Eng.* **2021**, *162*, 107683. [CrossRef]
25. Duarte, B.P.M.; Gonçalves, A.M.M.; Santos, L.O. Optimal Production and Inventory Policy in a Multiproduct Bakery Unit. *Processes* **2021**, *9*, 101. [CrossRef]
26. Huber, J.; Gossmann, A.; Stuckenschmidt, H. Cluster-Based Hierarchical Demand Forecasting for Perishable Goods. *Expert Syst. Appl.* **2017**, *76*, 140–151. [CrossRef]
27. Therkelsen, P.; Masanet, E.; Worrell, E. Energy Efficiency Opportunities in the U.S. Commercial Baking Industry. *J. Food Eng.* **2014**, *130*, 14–22. [CrossRef]
28. Sha, D.Y.; Lin, H.-H. A Multi-Objective PSO for Job-Shop Scheduling Problems. *Expert Syst. Appl.* **2010**, *37*, 1065–1070. [CrossRef]
29. Paquet-Durand, O.; Zettel, V.; Yousefi-Darani, A.; Hitzmann, B. The Supervision of Dough Fermentation Using Image Analysis Complemented by a Continuous Discrete Extended Kalman Filter. *Processes* **2020**, *8*, 1669. [CrossRef]
30. Yousefi-Darani, A.; Paquet-Durand, O.; Zettel, V.; Hitzmann, B. Closed Loop Control System for Dough Fermentation Based on Image Processing. *J. Food Process Eng.* **2018**, *41*, e12801. [CrossRef]
31. Van Rossum, G.; Drake, F., Jr. *Python Tutorial*; Technical Report CS-R9526; Centrum voor Wiskunde en Informatica (CWI): Amsterdam, The Netherlands, 1995.
32. Elhossini, A.; Areibi, S.; Dony, R. Strength Pareto Particle Swarm Optimization and Hybrid EA-PSO for Multi-Objective Optimization. *Evol. Comput.* **2010**, *18*, 127–156. [CrossRef]
33. Emmerich, M.T.M.; Deutz, A.H. A Tutorial on Multiobjective Optimization: Fundamentals and Evolutionary Methods. *Nat. Comput.* **2018**, *17*, 585–609. [CrossRef] [PubMed]
34. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [CrossRef]
35. Kang, P.; Deng, H.; Wang, X. Research on Multi-Equipment Collaborative Scheduling Algorithm under Composite Constraints. *Processes* **2022**, *10*, 1171. [CrossRef]
36. Asefi, H.; Jolai, F.; Rabiee, M.; Tayebi Araghi, M.E. A Hybrid NSGA-II and VNS for Solving a Bi-Objective No-Wait Flexible Flowshop Scheduling Problem. *Int. J. Adv. Manuf. Technol.* **2014**, *75*, 1017–1033. [CrossRef]
37. Amelian, S.S.; Sajadi, S.M.; Navabakhsh, M.; Esmaelian, M. Multi-Objective Optimization for Stochastic Failure-Prone Job Shop Scheduling Problem via Hybrid of NSGA-II and Simulation Method. *Expert Syst.* **2022**, *39*, e12455. [CrossRef]
38. Zhan, X.; Xu, L.; Ling, X. Task Scheduling Problem of Double-Deep Multi-Tier Shuttle Warehousing Systems. *Processes* **2020**, *9*, 41. [CrossRef]

39. Veldhuizen, D.A.V.; Lamont, G.B. *Multiobjective Evolutionary Algorithm Research: A History and Analysis*; Department of Electrical and Computer Engineering, Graduate School of Engineering, Air Force Institute of Technology: Wright-Patterson, OH, USA, 1998.
40. Zitzler, E.; Deb, K.; Thiele, L. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evol. Comput.* **2000**, *8*, 173–195. [CrossRef]
41. Babor, M.; Hitzmann, B. Small and Medium-Sized Bakery Production Data for Scheduling. *Mendeley Data* **2022**, *2*. [CrossRef]