Application of Nature-Inspired Optimization Algorithms to Improve the Production Efficiency of Small and Medium-Sized Bakeries

Dissertation to obtain the doctoral degree of Natural Science

(Dr. rer. nat.)

Faculty of Natural Science

University of Hohenheim

Institute of Food Science and Biotechnology

Submitted by

Md Majharul Islam Babor

Born on 10 July 1990 in Sylhet, Bangladesh

Submitted in 2023

Dean:	Prof. Dr. Uwe Beifuss
First reviewer:	Prof. Dr. Bernd Hitzmann
Second reviewer:	Prof. Dr.-Ing. Reinhard Kohlus
Submitted in:	Janualry 2023
Oral examination on:	. .2023

Chapter 1

Preface

1.1. Acknowledgements

This thesis was conducted at the Department of Process Analytics and Cereal Science, Institute of Food Science and Biotechnology, University of Hohenheim from July 2020 to September 2022 with the goal of achieving Doctorate in Natural Science (Dr. rer. nat.) degree.

First and foremost, I would like to express my gratitude to my supervisor, Prof. Dr. Bernd Hitzmann, for his excellent guidance and support throughout this study. Discussions with him have always been interesting and have provided new ideas for improving the methods used in this study. I am grateful to him for providing me with the opportunity to work on the EU project "PrO4Bake," which allowed me to gain analytical skills. I would also like to thank my co-supervisor, Prof. Dr. Reinhard Kohlus, for his supervision and suggestions.

Anyone reading this thesis who was born before 2019 is a survivor of the Covid-19 pandemic, and I believe the majority of us would not be alive if the vaccine had not been developed. I would like to thank all of the scientists and collaborators who worked thousands of sleepless hours to keep us alive. A large part of this study was conducted during this pandemic when infection rates were high worldwide. Nonetheless, PrO4Bake project partners collected data from bakeries, which was obviously for the project but created the foundation for this study. I would like to thank all my project partners for it.

I would like to thank one of the co-authors of this study and my colleague Dr. Oliver Paquet-Durand for sharing his outstanding research ideas. I would also like to thank Dr. Viktoria Zettel, Dr. Abdolrahim Yousefi-Darani, Dr. Pegah Sadeghi Vasafi, Dr. Sendeku Takele Alemneh, Leah Munyendo, Ann Mary Kollemparembil, Shubhangi Srivastava, Kim Brettschneider and my other colleagues in the department of Process Analytics and Cereal Science for their cordial support. Aside from academic tasks, enjoying flammkuchen baked in Hohenheim's wood oven a few times would be saved in my memory for a long time. I want to thank Herbert Götz, and Dr. Viktoria Zettel for making delicious flammkuchen and Dr. Oliver Paquet-Durand for baking in the wood oven with their expert hands.

I want to thank my family because without their support, I would not have been able to come to Germany and eventually conduct this research.

1.2. Co-authors

The scientific work presented in this thesis was conducted in cooperation with co-authors from the University of Hohenheim, University of Aarhus and Spanish Council for Scientific Research (CSIC).

Optimization of No-Wait Flowshop Scheduling Problem in Bakery Production with Modified PSO, NEH and SA

By Majharulislam Babor, Julia Senge, Cristina M. Rosell, Dolores Rodrigo and Bernd Hitzmann

Author contributions: M. Babor performed conceptualization, methodology, software, formal analysis, investigation, visualization, writing—original draft preparation. J. Senge and D. Rodrigo contributed with the resources and data curation. C. M. Rosell provided the resources and played the role of project administration. B. Hitzmann supervised the study and played the role of project administration.

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

By Majharulislam Babor, Line Pedersen, Ulla Kidmose, Olivier Paquet-Durand and Bernd Hitzmann

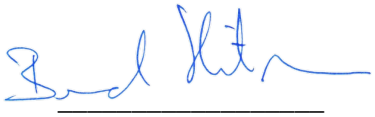
Author contributions: M. Babor performed conceptualization, methodology, software, formal analysis, investigation, visualization, writing—original draft preparation. L. Pedersen contributed with the resources and data curation. U. Kidmose provided the resources and played the role of project administration. O. Paquet-Durand contributed to conceptualization and methodology. B. Hitzmann supervised the study and played the role of project administration.

Modeling and Optimization of Bakery Production Scheduling to Minimize Makespan and Oven Idle Time

By Majharulislam Babor, Olivier Paquet-Durand, Reinhard Kohlus, and Bernd Hitzmann

Author contributions: M. Babor performed conceptualization, methodology, software, formal analysis, investigation, visualization, writing—original draft preparation. O. Paquet-Durand contributed to conceptualization and methodology. R. Kohlus contributed to methodology and supervised the study. B. Hitzmann supervised the study and played the role of project administration.

Stuttgart, 20/01/2023



Place, Date **Signature of supervisor**

1.3. Publication List

Peer-reviewed publications

- M. Babor, J. Senge, C. M. Rosell, D. Rodrigo and B. Hitzmann. Optimization of No-Wait Flowshop Scheduling Problem in Bakery Production with Modified PSO, NEH and SA. Processes, Volume 9, Issue 11, 2044, November 2021. https://doi.org/10.3390/pr9112044

- M. Babor, L. Pedersen, U. Kidmose, O. Paquet-Durand and B. Hitzmann. Application of Non-Dominated Sorting Genetic Algorithm (NSGA-II) to Increase the Efficiency of Bakery Production: A Case Study. Processes, Volume 10, Issue 8, 1623, August 2022. https://doi.org/10.3390/pr10081623

- M. Babor, O. Paquet-Durand, R. Kohlus and B. Hitzmann. Modeling and Optimization of Bakery Production Scheduling to Minimize Makespan and Oven Idle Time. Scientific reports, volume 13, article 235 (2023), January 2023. https://doi.org/10.1038/s41598-022-26866-9

Conference

- M. Babor, B. Hitzmann. Application of Nature-Inspired Multi-Objective Optimization Algorithms to Improve the Bakery Production Efficiency. Presented at the 1st International Electronic Conference on Processes: Processes System Innovation, 17–31 May 2022, published in Eng. Proc. Volume 19, Issue 1, 31, May 2022. https://doi.org/10.3390/ECP2022-12630

Datasets

- M. Babor, B. Hitzmann. Small and medium-sized bakery production data for scheduling. Two datasets: BK15, BK50, Mendeley Data, V2, June 2022. http://doi.org/10.17632/dhgbssb8ns.2

- M. Babor, B. Hitzmann. Production data from a small bakery manufacturing line. One dataset: BK20, Mendeley Data, V1, June 2022. https://doi.org/10.17632/7x5t3rxx5f.1

1.4. Summary

Increasing production efficiency through schedule optimization is one of the most influential topics in operations research that contributes to decision-making process. It is the concept of allocating tasks among available resources within the constraints of any manufacturing facility in order to minimize costs. It is carried out by a model that resembles real-world task distribution with variables and relevant constraints in order to complete a planned production. In addition to a model, an optimizer is required to assist in evaluating and improving the task allocation procedure in order to maximize overall production efficiency. The entire procedure is usually carried out on a computer, where these two distinct segments combine to form a solution framework for production planning and support decision-making in various manufacturing industries. Small and medium-sized bakeries lack access to cutting-edge tools, and most of their production schedules are based on personal experience. This makes a significant difference in production costs when compared to the large bakeries, as evidenced by their market dominance.

In this study, a hybrid no-wait flow shop model is proposed to produce a production schedule based on actual data, featuring the constraints of the production environment in small and medium-sized bakeries. Several single-objective and multi-objective nature-inspired optimization algorithms were implemented to find efficient production schedules. While makespan is the most widely used quality criterion of production efficiency because it dominates production costs, high oven idle time in bakeries also wastes energy. Combining these quality criteria allows for additional cost reduction due to energy savings as well as shorter production time. Therefore, to obtain the efficient production plan, makespan and oven idle time were included in the objectives of optimization.

To find the optimal production planning for an existing production line, particle swarm optimization, simulated annealing, and the Nawaz-Enscore-Ham algorithms were used. The weighting factor method was used to combine two objectives into a single objective. The classical optimization algorithms were found to be good enough at finding optimal schedules in a reasonable amount of time, reducing makespan by 29 % and oven idle time by 8 % of one of the analyzed production datasets. Nonetheless, the algorithms' convergence was found to be poor, with a lower probability of obtaining the best or nearly the best result. In contrast, a modified particle swarm optimization (MPSO) proposed in this study demonstrated significant improvement in convergence with a higher probability of obtaining better results.

To obtain trade-offs between two objectives, state-of-the-art multi-objective optimization algorithms, non-dominated sorting genetic algorithm (NSGA-II), strength Pareto evolutionary algorithm, generalized differential evolution, improved multi-objective particle swarm optimization (OMOPSO) and speed-constrained multi-objective particle swarm optimization (SMPSO) were implemented. Optimization algorithms provided efficient production planning with up to a 12 % reduction in makespan and a 26 % reduction in oven idle time based on data from different production days. The performance

comparison revealed a significant difference between these multi-objective optimization algorithms, with NSGA-II performing best and OMOPSO and SMPSO performing worst.

Proofing is a key processing stage that contributes to the quality of the final product by developing flavor and fluffiness texture in bread. However, the duration of proofing is uncertain due to the complex interaction of multiple parameters: yeast condition, temperature in the proofing chamber, and chemical composition of flour. Due to the uncertainty of proofing time, a production plan optimized with the shortest makespan can be significantly inefficient. The computational results show that the schedules with the shortest and nearly shortest makespan have a significant (up to 18 %) increase in makespan due to proofing time deviation from expected duration. In this thesis, a method for developing resilient production planning that takes into account uncertain proofing time is proposed, so that even if the deviation in proofing time is extreme, the fluctuation in makespan is minimal. The experimental results with a production dataset revealed a proactive production plan, with only 5 minutes longer than the shortest makespan, but only 21 min fluctuating in makespan due to varying the proofing time from -10 % to +10 % of actual proofing time.

This study proposed a common framework for small and medium-sized bakeries to improve their production efficiency in three steps: collecting production data, simulating production planning with the hybrid no-wait flow shop model, and running the optimization algorithm. The study suggests to use MPSO for solving single objective optimization problem and NSGA-II for multi-objective optimization problem. Based on real bakery production data, the results revealed that existing plans were significantly inefficient and could be optimized in a reasonable computational time using a robust optimization algorithm. Implementing such a framework in small and medium-sized bakery manufacturing operations could help to achieve an efficient and resilient production system.

1.5. Zusammenfassung

Die Steigerung der Produktionseffizienz durch die Optimierung von Arbeitsplänen ist eines der am meisten erforschten Themen im Bereich der Unternehmensplanung, die zur Entscheidungsfindung beiträgt. Es handelt sich dabei um die Aufteilung von Aufgaben auf die verfügbaren Ressourcen innerhalb der Beschränkungen einer Produktionsanlage mit dem Ziel der Kostenminimierung. Diese Optimierung von Arbeitsplänen wird mit Hilfe eines Modells durchgeführt, das die Aufgabenverteilung in der realen Welt mit Variablen und relevanten Einschränkungen nachbildet, um die Produktion zu simulieren. Zusätzlich zu einem Modell sind Optimierungsverfahren erforderlich, die bei der Bewertung und Verbesserung der Aufgabenverteilung helfen, um eine effiziente Gesamtproduktion zu erzielen. Das gesamte Verfahren wird in der Regel auf einem Computer durchgeführt, wobei diese beiden unterschiedlichen Komponenten (Modell und Optimierungsverfahren) zusammen einen Lösungsrahmen für die Produktionsplanung bilden und die Entscheidungsfindung in verschiedenen Fertigungsindustrien unterstützen. Kleine und mittelgroße Bäckereien haben zumeist keinen Zugang zu den modernsten Werkzeugen und die meisten ihrer Produktionspläne beruhen auf persönlichen Erfahrungen. Dies macht einen erheblichen Unterschied bei den Produktionskosten im Vergleich zu den großen Bäckereien aus, was sich in deren Marktdominanz widerspiegelt.

In dieser Studie wird ein hybrides No-Wait-Flow-Shop-Modell vorgeschlagen, um einen Produktionsplan auf der Grundlage tatsächlicher Daten zu erstellen, der die Beschränkungen der Produktionsumgebung in kleinen und mittleren Bäckereien berücksichtigt. Mehrere einzel- und mehrzielorientierte, von der Natur inspirierte Optimierungsalgorithmen wurden implementiert, um effiziente Produktionspläne zu berechnen. Die Minimierung der Produktionsdauer ist das am häufigsten verwendete Qualitätskriterium für die Produktionseffizienz, da sie die Produktionskosten dominiert. Jedoch wird in Bäckereien durch hohe Leerlaufzeiten der Öfen Energie verschwendet was wiederum die Produktionskosten erhöht. Die Kombination beider Qualitätskriterien (minimale Produktionskosten, minimale Leerlaufzeiten der Öfen) ermöglicht eine zusätzliche Kostenreduzierung durch Energieeinsparungen und kurze Produktionszeiten. Um einen effizienten Produktionsplan zu erhalten, wurden daher die Minimierung der Produktionsdauer und der Ofenleerlaufzeit in die Optimierungsziele einbezogen.

Um optimale Produktionspläne für bestehende Produktionsprozesse von Bäckereien zu ermitteln, wurden folgende Algorithmen untersucht: *Particle Swarm Optimization, Simulated Annealing* und *Nawaz-Enscore-Ham*. Die Methode der Gewichtung wurde verwendet, um zwei Ziele zu einem einzigen Ziel zu kombinieren. Die Optimierungsalgorithmen erwiesen sich als gut genug, um in angemessener Zeit optimale Pläne zu berechnen, wobei bei einem untersuchten Datensatz die Produktionsdauer um 29 % und die Leerlaufzeit des Ofens um 8 % reduziert wurde. Allerdings erwies sich die Konvergenz der Algorithmen als unzureichend, da nur mit einer geringen Wahrscheinlichkeit das beste oder nahezu beste Ergebnis berechnet wurde. Im Gegensatz dazu zeigte der in dieser Studie ebenfalls untersuchte *modifizierte Particle-swarm-Optimierungsalgorithmus* (mPSO) eine deutliche Verbesserung

der Konvergenz mit einer höheren Wahrscheinlichkeit, bessere Ergebnisse zu erzielen im Vergleich zu den anderen Algorithmen.

Um Kompromisse zwischen zwei Zielen zu erzielen, wurden moderne Algorithmen zur Mehrzieloptimierung implementiert: *Non-dominated Sorting Genetic Algorithm* (NSGA-II), *Strength Pareto Evolutionary Algorithm, Generalized Differential Evolution, Improved Multi-objective Particle Swarm Optimization* (OMOPSO), and *Speed-constrained Multi-objective Particle Swarm Optimization* (SMPSO). Die Optimierungsalgorithmen ermöglichten eine effiziente Produktionsplanung mit einer Verringerung der Produktionsdauer um bis zu 12 % und einer Verringerung der Leerlaufzeit der Öfen um 26 % auf der Grundlage von Daten aus unterschiedlichen Produktionsprozessen. Der Leistungsvergleich zeigte signifikante Unterschiede zwischen diesen Mehrziel-Optimierungsalgorithmen, wobei NSGA-II am besten und OMOPSO und SMPSO am schlechtesten abschnitten.

Die Gärung ist ein wichtiger Verarbeitungsschritt, der zur Qualität des Endprodukts beiträgt, indem der Geschmack und die Textur des Brotes positiv beeinflusst werden kann. Die Dauer der Gärung ist jedoch aufgrund der komplexen Interaktion von mehreren Größen abhängig wie der Hefezustand, der Temperatur in der Gärkammer und der chemischen Zusammensetzung des Mehls. Aufgrund der Variabilität der Gärzeit kann jedoch ein Produktionsplan, der auf die kürzeste Produktionszeit optimiert ist, sehr ineffizient sein. Die Berechnungsergebnisse zeigen, dass die Pläne mit der kürzesten und nahezu kürzesten Produktionsdauer eine erhebliche (bis zu 18 %) Erhöhung der Produktionsdauer aufgrund der Abweichung der Gärzeit von der erwarteten Dauer aufweisen. In dieser Arbeit wird eine Methode zur Entwicklung einer robusten Produktionsplanung vorgeschlagen, die Veränderungen in den Gärzeiten berücksichtigt, so dass selbst bei einer extremen Abweichung der Gärzeit die Schwankung der Produktionsdauer minimal ist. Die experimentellen Ergebnisse für einen Produktionsprozess ergaben einen robusten Produktionsplan, der nur 5 Minuten länger ist als die kürzeste Produktionsdauer, aber nur 21 Minuten in der Produktionsdauer schwankt, wenn die Gärzeit von -10 % bis +10 % der ermittelten Gärzeit variiert.

In dieser Studie wird ein Vorgehen für kleine und mittlere Bäckereien vorgeschlagen, um ihre Produktionseffizienz in drei Schritten zu verbessern: Erfassung von Produktionsdaten, Simulation von Produktionsplänen mit dem hybrid No-Wait Flow Shop Modell und Ausführung der Optimierung. Für die Einzieloptimierung wird der mPSO-Algorithmus und für die Mehrzieloptimierung NSGA-II-Algorithmus empfohlen. Auf der Grundlage realer Bäckereiproduktionsdaten zeigten die Ergebnisse, dass die in den Bäckereien verwendeten Pläne ineffizient waren und mit Hilfe eines effizienten Optimierungsalgorithmus in einer angemessenen Rechenzeit optimiert werden konnten. Die Umsetzung eines solchen Vorgehens in kleinen und mittelgroßen Bäckereibetrieben trägt dazu bei effiziente und robuste Produktionspläne zu erstellen und somit die Wettbewerbsfähigkeit dieser Bäckereien zu erhöhen.

Contents

Introduction and Outline

Chapter 2

2.1. Introduction

With the aid of numerous cutting-edge technologies, such as the Internet of Things (IoT), process analytics, machine learning, and artificial intelligence (AI), manufacturing industries are revolutionizing operation facilities under the frame of Industry 4.0. The application of these digital technologies in industrial domains offers the potential to lean towards automation by improving production efficiency, lowering production loss and resource wastage, and getting access to predictive maintenance of the operation facilities [1]. Decisions are made based on data gathered from operation facilities themselves in order to establish robust approaches; nevertheless, the challenge is to identify insights and meaningful aspects that can contribute to improving the operation facilities. Although the global market is becoming more competitive, this data-driven decision-making methodology offers an opportunity to reduce the disparity between large, medium, and small-scale industries. The process is straightforward: gather data, uncover insights, and incorporate AI to improve current processes.

The automation in the food sector is being greatly motivated by the fourth industrial revolution [2], from the land to the consumer's hand, and food processing is the centre of this chain. The consumption of bakery products within this segment has expanded significantly over time, with a recent deviation in many countries during the Covid-19 pandemic due to the closure of food outlets. Due to variances in the region, culture, consumer preferences, and seasonality, the varieties of bakery items are extraordinarily wide. Study shows that there are more than 3200 registered varieties of bakery goods only in Germany, each of which has a unique recipe [3]. In 2021, the German market had an estimated 15 billion Euros in total turnover (1.5 million Euros per company), employing 240,000 people [4]. Despite the steady expansion of the market, the number of artisanal bakeries has fallen dramatically in the last 60 years, from 55,000 to 9,965. Furthermore, as of 2021, 60 % of bakeries contributed 7 % of yearly sales, while only 6 % of bakeries produced a significant 69 % of annual sales [5].

There might be many reasons behind such sales distribution, but clearly, a few companies are dominating the bakery market. Additionally, it conveys the impression that the number of artisanal bakeries is lowering as a result of the market dominance of a few large bakeries. If we limit our attention to the operation facilities, the volume of flour they process annually is higher, which has a significant impact on the cost of production. Small and medium-sized bakeries, in contrast, are unable to generate comparable revenue due to high operational costs caused by the lower amount of flour usage [6]. Furthermore, whereas large bakeries use cutting-edge technologies to make resilient operational decisions, small and medium-sized bakeries continue to rely on personal experience [6–8]. It makes a definite difference in how well they use resources like energy, manpower, raw materials, and time, which altogether influence production costs.

The motivation of this study is the gap between large bakeries and small and medium-sized bakeries in applying cutting-edge technologies, particularly for production optimization, without altering

the plant's infrastructure. This study investigates the impact of implementing a contemporary decision-making strategy in small and medium-sized bakeries with the primary objective of creating a digital tool to minimize production time and energy waste. The novelty of this study compared to other similar studies is that it explores cases from several bakeries located in various European nations with vast disparities in operation facilities, product range, and recipes to offer a common framework.

2.1.1. Bakery Production Scheduling Problem

Imagine a baker produces different types of bread where each bread has individual characteristics in ingredients and dough processing. Ingredients are prepared initially based on bread recipes. The further dough processing stages including kneading, dough rest, dividing, shaping, proofing, and baking are performed accordingly that involve both machines and workers [8, 9]. The variation in dough treatments can be explained in many ways, including the order, duration, and machine that is used for each stage of processing, which distinguish each type of bakery product [10–12]. To perform one of these processing tasks baker has limited options in either machines or workers [13, 14] and therefore cannot start producing many products at the same time. A baker may prepare ingredients for many products, such as Product A, Product B, etc., at the same time. However, if the bakery features one kneader, kneading of only Product A can be performed right away. To perform the kneading of Product B, the baker must wait at least until the kneading of Product A is completed. This waiting time for Product B equals the duration of the kneading of Product A. The duration of kneading and other processing tasks for Product A and Product B might be different. Therefore, similar to the blockage in the kneader, the baker needs to plan the schedules of the machines for the following processing tasks. The planning for Product B is influenced by Product A and so does the following products. Therefore, the baker produces one after one so that dough processing tasks for every product are performed under the limitation of the machines.

To define this problem simply, a baker has n products which have to be processed through m machines in the same order, but with various durations of tasks. It could be simple, if the durations for the corresponding processing stages are the same regardless of the product recipes. If Product A and Product B have the same durations for kneading and following stages, doesn't matter which product a baker produces first, the ultimate makespan is the same because they occupy the machines in the same manner, i.e., duration and order. However, due to the variations in the durations, the order in which a baker produces them influences the makespan significantly. Additionally, it determines the schedule for a machine when it is active and when it is idle. Some machines consume energy when they are idle and cause energy waste [8]. Therefore, besides makespan, reduction of energy waste is important.

Many processing operations are carried out manually, particularly in small and medium-sized bakeries because they are considered to be "works of art" and cannot be automated. Throughout the production period, many individuals skilled in various processes work to do these manual tasks. Additionally, every bakery includes alternatives for operations that require a machine, such as a stone oven and an oven chamber for baking. The machines are chosen depending on the requirements of the product

recipe, for example, a product that requires baking in a stone oven, cannot be baked in an oven chamber. Even so, a stone oven includes multiple compartments, each of which may accommodate a batch of product, however, can follow independent baking duration and other specifications. Considering these, the scheduling model for bakeries is identical due to additional constraints, which are covered in greater detail in the following sections.

2.1.2. Production Optimization

Makespan and energy waste are two major goals in manufacturing engineering to achieve efficient production systems [15–20]. A baker must create a production plan that minimizes makespan and energy waste in order to make bakery production more efficient. In other words, it is the product order in which all processing tasks are distributed among machines in such a way that the manufacturing time and energy waste are kept to a minimum. In a simple way, one can experiment with all possible orders of the products and simulate production planning using production data to calculate makespan and energy waste. The possible orders for n products are $n!$, with the only difference being the combination of product positions [6, 8]. With smaller n values or fewer products, this approach may work with modern computers to find the best production order in a reasonable computational time. However, in small and medium-sized bakeries, bakers use a production line to produce 20 or more products in a batch. The number of orders for 20 products is 2.43×10^{18} and simulating all of them is time-consuming, and impractical. Such problems are classified as NP-hard (non-deterministic polynomial-time hard) in the literature and are challenging to solve mathematically [21–23].

Many studies offered effective approaches to finding a solution, if not the best, nearly the best within a reasonable computational time to the production planning problem [7, 24–30]. The approaches can be discussed under two segments: flow shop models and optimization algorithms. A production schedule is simulated using the flow shop model, which represents when the processing tasks are performed by machines. To do so, production data such as processing tasks, duration, and machines to perform each product task is required. Depending on the manufacturing environment, the flow shop model can be extremely complex. The flow shop model is featured by the constraints of the target manufacturing environment, and therefore no defined model will work for all manufacturing environments. Optimization algorithms, the second segment, have been widely used to improve the production efficiency of various types of products. These two segments are linked to finding the optimized solution to the production planning problem. Optimization algorithms propose production orders, which the flow shop model uses to simulate the entire production planning. The optimization objective, such as makespan, is calculated from the simulated schedule and sent back to the optimization algorithm for evaluation. Optimization algorithms generate new production orders in order to improve the objective values until a specific termination criterion is met [7, 8, 24, 30].

This approach finds an optimized schedule for a given production, which may not be the best one, but is expected to be near to the best one. Many optimization algorithms were used in the area of

production scheduling. Among them, nature-inspired optimization algorithms are widely used, the common characteristic of which is that the idea behind these algorithms is borrowed from nature. Despite these algorithms being powerful in solving many engineering optimization problems, some limitations have to be explained. The results obtained by nature-inspired algorithms can be different in different runs for the same problem [7, 8]. Whether the performance of a nature-inspired algorithm in solving problems is effective is one of the challenges that scientists and engineers deal with commonly. There is numerous research on optimization that demonstrate the challenge of being trapped in local optima, which makes algorithm deliver poor solutions. It is a matter of concern when the dimension of the problem is higher and there are many local optimum regions in the search space. Since the exact solution to most real-world problems is unknown, there is no way to directly compare the performance of an algorithm. Many scientists proposed effective approaches to improve the performance of the algorithms, mostly by modifying classical nature-inspired algorithms. Many performance metrics have been proposed to compare the performance of the optimization algorithms.

2.1.3. Single Objective Optimization

In many fields, scientists and engineers routinely tackle enormous optimization problems. Let us consider two simple scenarios. In the first case, to determine the value of one independent variable, let us assume x_1, from a range of continuous values that lead to the best value for the dependent variable $f(x_1)$. The best value of $f(x_1)$ is defined by system requirements, which can be either minimum or maximum. In the minimization problem, the objective is to find the value of x_1 for which $f(x_1)$ is as minimal as possible. In the second case, if the number of independent variables rises from 1 to n (≥ 2), such as $x_1, x_2, x_3, \dots, x_n$, the objective remains to find the minimum value for $f(x_1, x_2, x_3, \dots, x_n)$ [31].

Even though the only difference between the two cases is the number of independent variables, the complexity of the problem increases significantly in the latter case due to the n-dimensional search space. Finding one optimum value for x_1 in the former case is easier than finding the optimum combination of n values (for n independent variables) in the latter case. However, in both cases, the goal is to minimize one value, which keeps the objective space one-dimensional. Many nature-inspired optimization algorithms have been proposed to solve the optimization problem with higher dimension in search space but a single objective, which solve the real problems satisfactorily. Among them, particle swarm optimization [32] and simulated annealing [33] are widely applied algorithms. However, the effectiveness of algorithms in solving problems is an area of research where scientists are still contributing with more robust approaches.

2.1.4. Multi-Objective Optimization

In many engineering optimization problems, in addition to the complexity caused by the high dimensional search space, the number of objectives increases, with satisfying one causing dissatisfying another [34, 35]. There are a few other approaches to dealing with such issues, such as combining multiple objectives into one using the weighted sum method. The advantage of this approach is that, despite having a multi-objective problem, it solves similar to a single objective problem and provides satisfactory optimization results. However, the weighting factor is provided based on personal preference, and it is unknown whether combining the given weighting factors will result in an optimized solution if the problem is completely unknown. Furthermore, depending on the shape of the optimum region in the multi-dimensional objective space, this approach has limitations and may result in a sub-optimal solution [34]. There are a few optimization algorithms that address multi-objective optimization problems by providing a set of Pareto solutions rather than a single solution. These solutions represent trade-offs between objectives, from which a decision maker can select the best solution to meet the system's demand [36, 37]. In this study, non-dominated sorting genetic algorithm (NSGA-II) [38], multi-objective particle swarm optimization (MOPSO) [39], improved multi-objective particle swarm optimization (OMOPSO) [40], speed-constrained multi-objective particle swarm optimization (SMPSO) [41], generalized differential evolution (GDE3) [42] and strength Pareto evolutionary algorithm (SPEA2) [43] are used to solve multi-objective production optimization problems.

2.2. Outline

I. Processing of bakery production data

Real production data were used to understand the fundamentals and constraints of bakery manufacturing in small and medium-sized bakeries. Data were collected as part of the EU project, "PrO4Bake", which was funded by EIT Food and coordinated by the University of Hohenheim with partners from seven different EU nations [44]. Project partners collected production data from approximately 20 small and medium-sized bakeries in seven different European countries. Each dataset depicts the resource allocation for the processing of bakery items in an existing production planning. The collected data were cleaned and formatted for further analysis. The structure of production data is simplified and explained in publications (included in section 3). Three complete processed and cleaned bakery production datasets are published with the labels BK15, BK50 [14], and BK20 [13]. Annex A shows the production dataset BK15.

II. Modeling and nature-inspired optimization algorithms

A hybrid no-wait flow shop model was developed to address the constraints of common bakery practice. The production data from all collaborated bakeries were simulated using this model to assess the efficiency of the existing production planning. The optimization algorithms are used to determine whether the current schedule is under-optimized and whether overall production efficiency can be improved. A few single objective optimization algorithms were used at this stage. PSO was modified to improve its efficiency in finding the optimal schedule in order to design a robust optimization algorithm. Section 3.1 includes a link to a research article on this subject.

III. Application of multi-objective optimization algorithms

The goal of this phase of research was to find a solution that provides the least amount of makespan and oven idle time at the same time. Because ovens cannot be turned off during production, keeping idle time to a minimum reduces energy waste and, as a result, CO_2 emissions. Therefore, multi-objective optimization algorithms were implemented. A published research article presented in section 3.2 describes the mathematical modeling of the hybrid flow shop model for bakeries and a multi-objective optimization algorithm.

IV. Performance comparison of multi-objective optimization algorithms

The performance of the state-of-art multi-objective optimization algorithms was investigated. Three production data sets with varying numbers of products were used to find the most efficient optimization method. The implemented algorithms are distinguished by performance metrics that indicate the quality of obtained solutions, as presented in section 3.3.

V. A resilient bakery production schedule under uncertain proofing time

Yeast is used as a leavening agent in many bakery products to improve the texture and flavour of breads and other bakery products. The yeast fermentation process in bread making is known as "proofing." Proofing time is uncertain because it is affected by flour quality, yeast cell metabolism, temperature, and humidity. Because the optimized schedules are based on the assumption of deterministic duration for all stages, the entire production flow can be disrupted due to uncertainties in proofing time. Section 3.4 presents a resilient production planning that takes uncertain proofing time into account.

VI. Discussion, conclusion, and final remarks

A brief discussion demonstrating the consistency between the articles and findings is addressed in section 3.5. The related studies from the literatures are also presented in order to compare the findings and novelty of the methods used in this thesis. The study's concluding observations, and final remarks are presented in Section 4.

Chapter 3

Publications and Findings

3.1. Optimization of No-Wait Flowshop Scheduling Problem in Bakery Production with Modified PSO, NEH and SA

Majharulislam Babor, Julia Senge, Cristina M. Rosell, Dolores Rodrigo and Bernd Hitzmann

Citation

M. Babor, J. Senge, C. M. Rosell, D. Rodrigo and B. Hitzmann. Optimization of No-Wait Flowshop Scheduling Problem in Bakery Production with Modified PSO, NEH and SA. Processes, Volume 9, Issue 11, 2044, November 2021. https://doi.org/10.3390/pr9112044

Note: The references cited in the following research article are listed at the end of this section and are not included in the bibliography.

processes

Publications and Findings

MDPI

Article

Optimization of No-Wait Flowshop Scheduling Problem in Bakery Production with Modified PSO, NEH and SA

Majharulislam Babor [1,*], Julia Senge [1], Cristina M. Rosell [2], Dolores Rodrigo [2] and Bernd Hitzmann [1]

[1] Department of Process Analytics and Cereal Science, University of Hohenheim, 70599 Stuttgart, Germany; Julia.Senge@uni-hohenheim.de (J.S.); bernd.hitzmann@uni-hohenheim.de (B.H.)
[2] Instituto de Agroquímica y Tecnología de Alimentos (IATA-CSIC), Catedrático Agustín Escardino Benlloch 7, 46980 Paterna, Valencia, Spain; crosell@iata.csic.es (C.M.R.); lolesra@iata.csic.es (D.R.)
* Correspondence: majhar@uni-hohenheim.de

Abstract: In bakery production, to perform a processing task there might be multiple alternative machines that have the same functionalities. Finding an efficient production schedule is challenging due to the significant nondeterministic polynomial time (NP)-hardness of the problem when the number of products, processing tasks, and alternative machines are higher. In addition, many tasks are performed manually as small and medium-size bakeries are not fully automated. Therefore, along with machines, the integration of employees in production planning is essential. This paper presents a hybrid no-wait flowshop scheduling model (NWFSSM) comprising the constraints of common practice in bakeries. The schedule of an existing production line is simulated to examine the model and is optimized by performing particle swarm optimization (PSO), modified particle swarm optimization (MPSO), simulated annealing (SA), and Nawaz-Enscore-Ham (NEH) algorithms. The computational results reveal that the performance of PSO is significantly influenced by the weight distribution of exploration and exploitation in a run time. Due to the modification to the acceleration parameter, MPSO outperforms PSO, SA, and NEH in respect to effectively finding an optimized schedule. The best solution to the real case problem obtained by MPSO shows a reduction of the total idle time (TIDT) of the machines by 12% and makespan by 30%. The result of the optimized schedule indicates that for small- and medium-sized bakery industries, the application of the hybrid NWFSSM along with nature-inspired optimization algorithms can be a powerful tool to make the production system efficient.

Keywords: no-wait flowshop; bakery industry; optimization; production efficiency

Citation: 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. https://doi.org/10.3390/pr9112044

Academic Editors: Simant Upreti and Carl Schaschke

Received: 9 September 2021
Accepted: 11 November 2021
Published: 15 November 2021

1. Introduction

In an industrial production system, scheduling jobs is an important integral part. It refers to the arrangement of jobs to be processed on machines under some constraints. A production system can fail to be efficient due to the lack of planning and proper allocation of tasks among machines. In the literature, such problems are known as flowshop scheduling problems (FSSP) or a job shop scheduling problem and have been observed as NP (nondeterministic polynomial time)-complete combinatorial optimization problems when the number of machines exceeds two [1].

In FSSP, jobs are n products, each of which consists of a set of tasks. These tasks need to be performed sequentially by a set of m machines. In some flowshop environments, these tasks need to proceed continuously without any interruption in between, known as no-wait FSSP. In such cases, the production start time of a product can be independent. However, once it starts, the waiting time between two consecutive tasks is unacceptable. In a typical FSSP environment, all the products (n) undergo all the machines (m) in the same order, which is widely known as permutation FSSP.

In the bakery production system, the dynamic process involving physical and biochemical reactions, such as yeast fermentation, causes change in the behavior of the material.

Due to the addition of living yeast cells, the dough is strictly sensitive to time and temperature as they affect CO_2 production significantly and thus influence the texture and aroma of the final product. Even though some processing steps, for example, dough rest and cooling after baking, may not require any energy-consuming machine, they have a certain duration for these tasks. Exceeding the predefined duration for a task causes overtreatment and delay for the next treatment. In a bakery production environment, both are unacceptable as they directly affect the product quality. Therefore, the processing tasks need to be performed accordingly in time. From this perspective, the bakery production schedule falls in the no-wait FSSP.

Abdollapour et al. presented a mathematical model of a flexible no-wait flow shop problem with a capacity restriction on the machines [2]. Bewoor et al. described a mathematical model of a no-wait flow shop scheduling model and proposed a hybrid particle swarm optimization algorithm to solve the scheduling problem [3]. Application of no-wait FSSP was provided by many other studies [4,5]. However, there are some exceptional characteristics in the bakery production system and very few authors have focused on this branch [6–8]. Therefore, there are opportunities to improve production modeling and optimization. Medium- and small-scale bakeries, even in many developed countries, are reportedly conducting production operations inefficiently [7], merely based on personal experience. A recent study reported that there are enough opportunities to improve production efficiency in the bakery industry by forecasting sales and optimizing the production schedule [9]. In this scheme, sales forecasting tackles the waste of food by predicting the market demand for each type of product and the optimized production schedule concerns minimizing the production cost of the predicted products in terms of time and energy consumption. However, this article focuses on the optimization of the production schedule and addresses a hybrid NWFSSM with an application in a bakery industry that involves several constraints and objectives that were not included in previous studies. By using the proposed model, a production schedule can be simulated and the cost of production, such as makespan, and the TIDT can be calculated. However, to find an optimized schedule with minimum cost, an optimizer must be integrated.

In the literature, many heuristic and meta-heuristic optimizations were proposed with their modification and showed a comparison of their performance by solving scheduling problems. In the early stage, the constructive approach, widely known as NEH [10], is probably the one mostly refined and modified by other authors [11]. On the other hand, many meta-heuristics and their modified approaches have been applied to solve FSSP, such as particle swarm optimization [12,13], simulated annealing [14], genetic algorithm [13,15,16], ant colony optimization [7], and firefly algorithm [17].

Ever since the PSO algorithm was introduced by Eberhart and Kennedy [18], it was demonstrated that despite being a powerful algorithm, it easily falls into a local minima when solving complex and high dimensional problems [19]. Moreover, a common disadvantage of nature-inspired evolutionary optimization algorithms, like PSO, is that there is no certainty of always achieving a better result. Studies have found that a simple change in the PSO algorithm reportedly increases the performance effectiveness in terms of finding a quality solution to FSSP within a reasonable time. By adjusting the random value in the acceleration of a particle using a prescribed probability, PSO reportedly can avoid being trapped in local minima [20]. The addition of local search functionality in the PSO algorithm was presented to have improved performance [21,22]. A robust hybrid PSO algorithm was proposed by Ji et al. [23] where PSO-Based exploration and SA-based exploitation were integrated. Similarly, Issa et al. proposed a scheme of merging PSO with a recently proposed sine cosine optimization algorithm to impose optimum exploration and exploitation in searching the solution [24].

In this research, we propose a modified PSO (MSPO) to tackle the issues associated with providing premature optimization results by implementing the subtraction operator. The behavior of PSO concerning exploration and exploitation due to the change in acceleration parameters was investigated to solve the "no-wait" flowshop scheduling problem

effectively. An existing production line of a bakery was taken as a case with the objectives of minimizing the makespan and TIDT. The performance of MPSO was compared with standard PSO with two different configurations (PSO-A, PSO-B), SA, and NEH. The robustness and frequency to surpass a certain level of optimization result were investigated. The main contribution of the paper can be summarized as follows:

I. Develop an application of the optimization algorithms to improve the efficiency of the existing bakery production line by delivering a realistic schedule with reduced makespan and machine idle time.
II. Thoroughly present the formulation of a hybrid "no-wait" flowshop model (NWFSSM) in the bakery industry where a product follows a set of sequential tasks. However, the order, duration, and required machine for the tasks might be different for different products.
III. Carry out modification to standard PSO to avoid premature convergence to the local optimum solution.
IV. The proposed strategy requires no significant additional computational time, and implementation is as simple as standard PSO yet highly effective.
V. Demonstrate a performance comparison between MPSO and standard PSO, SA, and NEH to give an impression of how reliable the proposed method is.

The rest of the paper is organized as follows. Section 2 describes the formulation of the bakery scheduling problem and modifications in the PSO algorithm. Section 3 shows the comparative performance analysis of modified and standard PSO, SA, and NEH in solving the bakery scheduling problems.

2. Materials and Methods

The formulation of the bakery scheduling problem, simulation, and all optimization algorithms were performed 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. The simulation of the hybrid NWFSSM and optimization algorithms were programmed and performed using the programming language Python 3.7 [25].

In this study, two problem sets of bakery production data were used, which were collected from two European small- and medium-sized enterprise bakeries. In problem I, the bakery had 21 different product batches and was used to simulate the current production schedule. In the next step, the optimization algorithms were performed to find an optimized schedule and compared with current production efficiency.

The second case, problem II was taken from Hecker et al. [6] which had 40 product batches. The authors described a no-wait flowshop model for that bakery production and performed standard PSO to optimize it. Therefore, problem II was chosen to compare the performance of the proposed MPSO with that study.

2.1. Bakery Production System

Each product in the bakery has a recipe, from a scheduling perspective, which is the combination of the number of processing tasks and the sequence of these tasks. Figure 1 shows a typical production line in a bakery. However, the recipes can vary in many aspects, such as the total number and sequence of the tasks, use of a specific machine, machine set up (e.g., temperature), and duration for the tasks. Furthermore, many tasks are performed manually e.g., loading flour, yeast, salt, and water into a mixer, transferring dough into a fermentation chamber, or baking oven. This considers that only machines and machine capacity in scheduling may overlook the limitation of the manpower. Additionally, to conduct one task, there are multiple alternative machines in bakeries. The number of alternative machines for different tasks is also different (Figure 1). As a result, the task allocation procedure in bakery production is different than most of the production systems. Therefore, a real bakery production scheduling problem can not be considered as permutation FSSP, rather such a model can be categorized as a hybrid-NWFSSM [26].

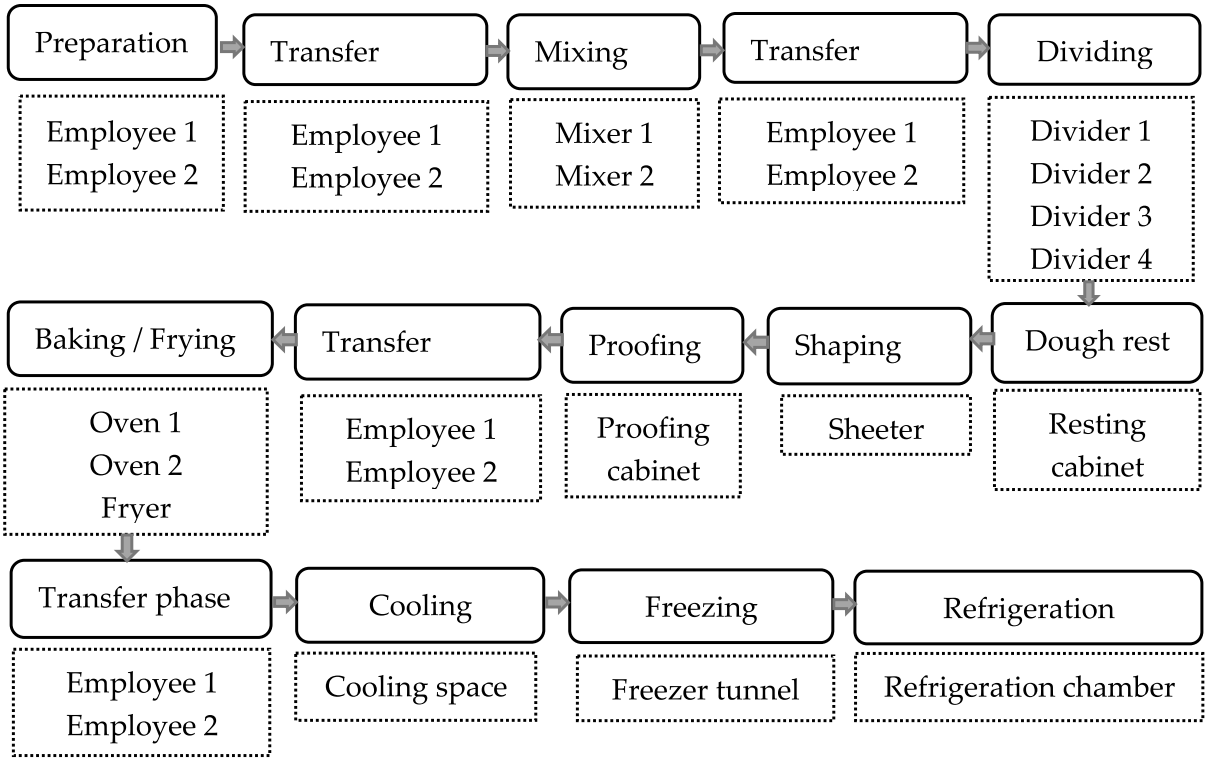


Figure 1. Production line of a bakery with processing tasks (-) and resource alternatives (. . .) to perform the tasks.

2.2. Problem Description

The problem dealt with in this paper can be discussed in two parts. Firstly, a hybrid NWFSSM was required to simulate the bakery production schedule. The main difficulty here was to integrate a set of constraints to reflect the actual production. It was expected that hybrid NWFSSM provides some information to evaluate one simulated production schedule. In bakery production efficiency, the main focus lies on the makespan and TIDT of the machines as they consume energy. Therefore, the output of the hybrid NWFSSM was the makespan and TIDT calculated from the simulated production schedule.

Secondly, an optimized schedule had to be found that has a minimum makespan and TIDT. The possible ways to schedule the production of n products can be found from a permutation of n items without repetition but in a different order of items. Mathematically, it can be expressed by $P(n) = n!$. In other words, the only difference between $n!$ schedules is the order of n products i.e., which product should be produced when. However, this can influence production efficiency significantly. If the order of n products in which they will be produced is changed, makespan and TIDT will also be changed. Moreover, if n is higher, the possible ways to schedule are also increased by $n!$. For 21 product batches in problem I, there are 5.1×10^{19} possible production schedules. The main difficulty is that there is no exact method to find the best schedule in a shorter period of time. Using mathematical programming, all possible solutions can be calculated and evaluated to find the optimal schedule. However, it might take a long computation time. Therefore, many heuristic and metaheuristic optimization algorithms have been used to solve such problems. In such cases, the optimization algorithms may not find the best possible solution, but an almost best solution, which is usually better compared to what actually followed by the bakeries.

In this study, PSO, MPSO, SA, and NEH were performed to search for an optimized schedule from the massive number of options.

2.3. Objective Function

In this study, the main objectives were to reduce the makespan and TIDT of the machines in bakery production. It can be described as follows:

$$Cost = C_{max} + WTIDT \tag{1}$$

where $Cost$ is the cost of optimized schedule, C_{max} is the makespan, and $WTIDT$ is the weighted total idle time of the machines. In the objective function, C_{max} is the exact makespan of an optimized schedule and WTIDT is calculated from the idle time of the machines of the same schedule using weighting factors (explained in Section 2.4). The weighting factor for a machine should reflect the cost due to the energy consumption contributed in total production cost. The highly energy-consuming machines should have a high weighting factor, hence a smaller increase in idle time would have a bigger impact on the total cost compared to the other machines. In this study, the weighting factor for the machines were chosen, such that WTIDT for the existing schedule was lower than the makespan. However, it completely depends on the goal. If any bakery wants to focus more on the idle time reduction of the machines, the weighting factors for individual machines can be tuned such that WTIDT is bigger than the existing schedule's makespan. The cost reduction due to the optimization algorithm was calculated using Equation (2).

$$Cost \text{ reduction} = \frac{Cost_{opt}}{Cost_0} \times 100 \tag{2}$$

where $Cost_{opt}$ and $Cost_0$ are the cost from optimized and existing production schedules, respectively calculated using Equation (1).

2.4. Hybrid NWFSSM for the Bakery Production

The 'no-wait' flowshop scheduling model for bakery production proposed by Hecker et al. [6] was implemented to simulate new production data. However, since the proposed model was strictly for one bakery, some limitations had to be overcome to make a new model to reflect real production scenarios, which will be discussed in this section.

Table 1 shows an example of production data. It shows that the number and order of processing stages are different for the products that are in the common practice of the bakery industry. In this example, square bread has dough rest as a processing task, while pan bread requires no dough rest. Moreover, the number of stages for sourdough, square bread, and pan bread are 4, 8, and 11, respectively. The number and order of the stages may vary depending on the recipe of the products, which also varies from bakery to bakery. Therefore, it is difficult to make a common order of processing stages with a higher number of products that will work for every bakery industry. Moreover, the addition of a new product may require modifying the order of processing stages.

In their study, Hecker et al. [6] proposed an idea to make an order by mixing machines and processing stages. In the proposed order, 26 stages were shown despite the maximum stage for any product being 12. The idea was to make a common route for all the products. If a certain stage or machine in the route is not relevant to a product, it can be ignored by setting a 0 min processing duration. The machine for a task was kept fixed, although it is more flexible in practice. For instance, baking a product can be done either in Oven 1 or Oven 2 as there are mostly multiple alternatives for the same processing task. Moreover, in the bakery, many processing tasks are conducted manually by employees. It is practical to consider the limited capacity of the employees, which was ignored in the model proposed by Hecker et al. [6]. Moreover, the evaluation criteria of a schedule, either makespan or TIDT was used by the authors.

Table 1. Example production data for three products.

Product Name	Processing Stage	Duration	Resources
Sourdough	1. Preparation	3	Employee
	2. Mixing	13	Mixer
	3. Transfer phase	1	Employee
	4. Proofing	360	Proofing cabinet
Square bread	1. Preparation	3	Employee
	2. Mixing	16	Mixer
	3. Transfer phase	1	Employee
	4. Dough rest	15	Resting cabinet
	5. Dividing	3	Divider
	6. Molding	10	Employee
	7. Proofing	75	Proofing cabinet
	5. Baking	18	Oven
	6. Cooling	360	Cooling cabinet
	7. Freezing	70	Freezer
	8. Packaging	10	Employee
Pan bread	1. Preparation	3	Employee
	2. Mixing	12	Mixer
	3. Transfer phase	1	Employee
	4. Dividing	4	Divider
	5. Shaping	5	Shaper
	6. Proofing	85	Proofing cabinet
	7. Baking	14	Oven
	9. Cooling	60	Cooling cabinet
	10. Freezing	70	Freezer
	11. Packaging	10	Employee

To eliminate these limitations in previous studies, a new flowshop model was created, the main features and constraints of which can be explained as follow.

- The name, number, and order of the stages for the products are independent. We propose no common route, rather it will be according to the original recipe. Therefore, due to the addition or deletion of a product, the complexity regarding the order of processing tasks can be ignored.
- Instead of fixing the machines, alternatives for a task are proposed. As a result, during task allocation, the algorithm will try to allocate the task for all products among these alternatives in a way that the overall makespan and WTIDT will be the lowest for a production sequence.
- The shift plan of employees is integrated, thus tasks are allocated within their working plan. Moreover, the conflict in manual tasks is avoided. Since delay between tasks is unacceptable, and interruption in one stage may cause delay to other stages and products, the task allocation for employees was found to be vital.
- The addition of new products and machines may require searching for a new optimized schedule as workload and capacity, respectively rise. Change in the shift plan or number of employees can also influence the overall production efficiency. Similarly, a machine, for instance, an oven can be added in the machine alternatives for baking to observe its influence on the production efficiency. The proposed model allows simulating the production by adapting such changes and monitor the efficiency to help the administration in decision making.
- Instead of a single objective, both makespan and WTIDT were taken into consideration to find a cost-effective solution to the production schedule. The WTIDT reflects the contribution of each machine in the final production cost.

Figure 2 illustrates the procedure of solving a given bakery production scheduling problem. The working principle can be described in four sections, namely, production data, hybrid NWFSSM, optimization algorithms, and visualization of results. In the production

data, the name of the products, processing stages, start time, finish time, used machine for each stage, shift plan for employees, and current production sequence were recorded. The extracted information was used to simulate the schedule using hybrid NWFSSM. The output of the model is makespan and WTIDT for any production sequence. Afterward, the optimization algorithm executes and searches for an improved solution set, which is delivered back to the hybrid NWFSSM. The updated solution is converted into a sequence of production to make a new schedule and evaluation. This procedure is repeated until a certain termination criterion for one run is met. At the final stage, the schedule is visualized and cost reduction in terms of makespan and WTIDT is presented.

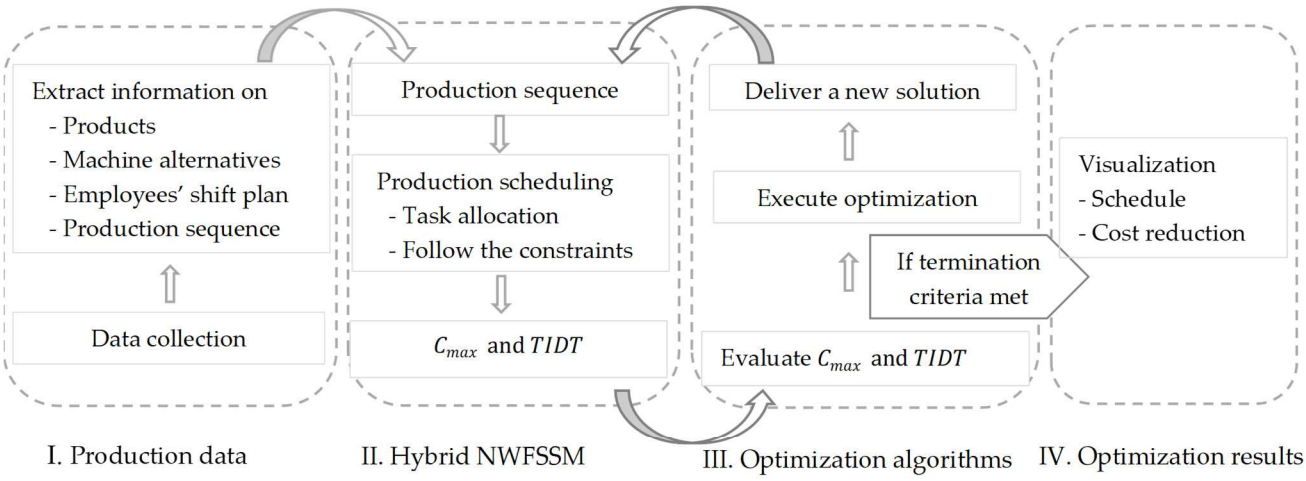


Figure 2. Working principle of bakery production optimization using hybrid no-wait flowshop scheduling model (NWFSSM).

The progress of hybrid NWFSSM is presented in Figure 3. The assumptions and constraints are as following:

- Let us assume a production line has n products $P \in \{P_i \mid i = 1, 2, 3, \ldots, n\}$, where P indicates the set of products.
- To complete the processing of these products, there are m machines $M \in \{M_k \mid k = 1, 2, 3, \ldots, m\}$ and e employees $E \in \{E_l \mid l = 1, 2, 3, \ldots, e\}$.
- $P_{i,s}$ refers to one processing time of product P_i at stage s where $s = \{1, 2, 3, \ldots, S_i\}$ and S_i is the final stage for the product P_i.
- A set of alternatives, either machines or employees that can perform $P_{i,s}$ processing task is $A_{i,s}$ where $A_{i,s} \subset M$ or $A_{i,s} \subset E$.
- None of the machines and employees can process more than one task at a time.
- It is assumed that machines are available from the beginning to the end of the production time.
- The task allocation among employees must follow their working schedule such that none of the allocated tasks is scheduled out of their availability.

The allocation of tasks among machines and employees was performed by using Algorithm 1. The makespan (C_{max}) and weighted total idle time (WTIDT) of machines were calculated by Equations (3) and (4) respectively.

$$C_{max} = maximum(\{f_i \mid i = 1, 2, \ldots, n\}) \tag{3}$$

where f_i is the finish time of the product P_i.

$$WTIDT = \sum_{k=1}^{m} \left(end_k - start_k - \sum_{i=1}^{n} pt_{i,k} \right) \times W_k \tag{4}$$

where k denotes one machine from a total number of m machines, $start_k$ and end_k refer to the production time when machine k started and ended respectively, $pt_{i,k}$ is the processing time of the product P_i that was allocated to the machine k and W_k refers weighting factor for idle time of machine k.

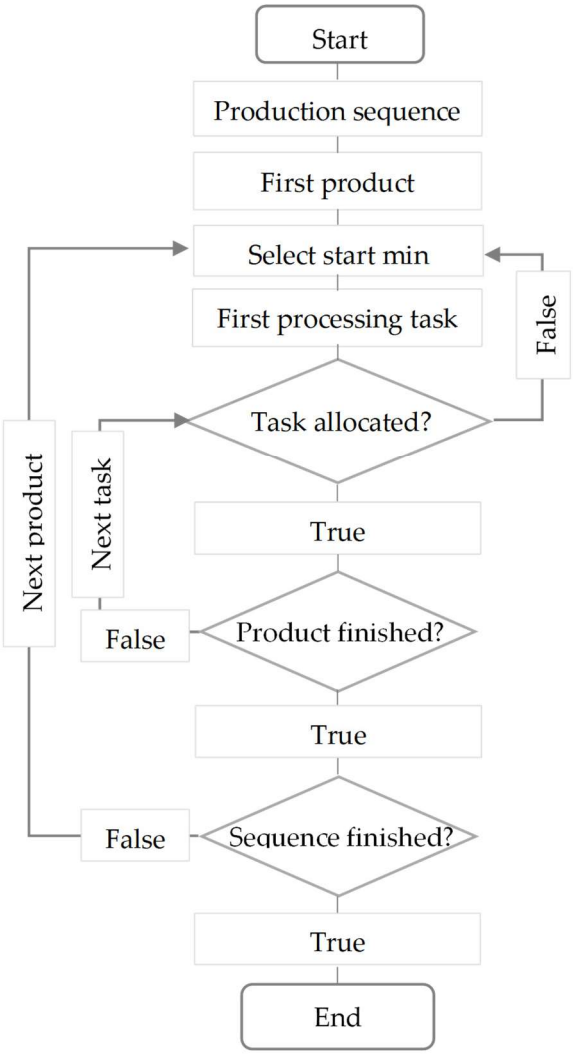


Figure 3. Scheduling procedure of hybrid NWFSSM.

In a bakery production environment, the idle time of all the machines is not equally important for the cost of production. These machines can be used immediately after switching on, and hence no energy is consumed during idle time. In contrast, several machines e.g., ovens require to run a certain time to fulfill the settings required to perform the processing task. In such cases, to avoid the unexpected interruption in the production, the machines are mostly kept running even when no products are processed. The idle time of these machines must be pronounced with a higher weighting factor as it increases the overall production cost. Therefore, W_k was used to discriminate the machines in contributing to the cost function through WTIDT in Equation (1). WTIDT was used only for optimization purposes to represent the cost due to the TIDT. In contrast, TIDT refers to the sum of the idle time of all the machines in a schedule. TIDT was calculated by using Equation (4), just as with the WTIDT, however with $W_k = 1$ for all the machines.

Algorithm 1. Tasks allocation procedure.

```
1   begin
2       generate a solution // e.g., from PSO
3       sort jobs (J) according to the given solution // e.g., SPV rule
4       for P_i = 1 to n do // for all the products
5           for sm = t_0 to t_f do // start minutes for P_i e.g., t_0, t_1, ..., t_f
6               s = 1, st_s = sm // start time for first stage
7               uninterrupted = True // no-wait constraint
8               while uninterrupted is True and s ≤ S_i // for all the stages
9                   task minutes_{i,s} = {st_s, st_s + 1, ..., st_s + pt_{i,s}}
10                  resource available = False, alternative = 1
11                  while resource available is False and alternative ≤ A_{i,s}
12                      if alternative is free during task minutes_{i,s}
13                          Allocate task to this alternative resource
14                          resource available = True // break loop (line 11)
15                          s = s + 1, st_s = st_s + pt_{i,s} // start time for next stage
16                      else try with next alternative
17                          alternative = alternative + 1
18                  if resource available = False // no alternative is free
19                          uninterrupted = False // no-wait constraint is interrupted
20                          // break loop (line 8) and try with a new sm (line 5)
```

To illustrate the proposed NWFSSM, there are 3! = 6 possible production sequences to produce three products e.g., sourdough, square bread, and pan bread (Table 1). For simplicity, no alternative machines are shown. Proofing, dough rest, cooling cabinets, and freezer are considered to have the capacity to operate three products at a time. The rest of the resources can operate only one product at a time. For instance, if the mixer is occupied for square bread, the mixing of pan bread cannot be executed during this time as there is only one mixer and it can operate one product at a time. If a production sequence of {sourdough, square bread, pan bread} is followed, the schedule is presented in Figure 4.

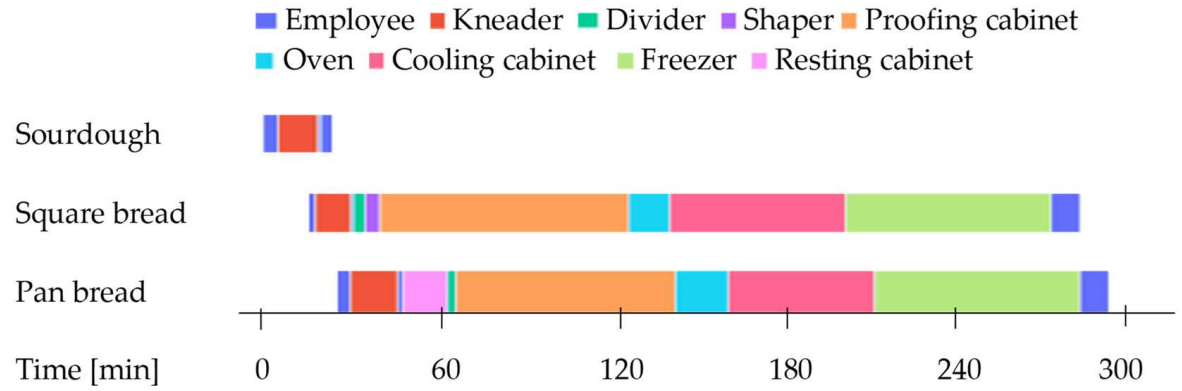


Figure 4. Example schedule of three products with a production sequence of {sourdough, square bread, pan bread}.

According to the schedule, the makespan is 289 min as this is the maximum time any product is finished. The idle time for the oven is 2 min. However, if the production sequence is changed to {pan bread, sourdough, square bread}, the values are changed to 295 min, and 7 min respectively.

2.5. Optimization Algorithms

2.5.1. Modification to Standard PSO

PSO is a metaheuristic optimization algorithm inspired by the behavior of animals like birds on the food search. The position of one particle or individual carries one solution to the given problem. The particles search the solution in a space in the form of multi-

paths obliged by the movement in a swarm following a quasi-likelihood function. The instant movement and direction to which a particle moves are controlled by two dominant positions in the space i.e., current global best position and best personal position. The former offers the best solution to the problem and the latter is best in the solution history of the corresponding individual particle.

Each movement results in a change of the position of the particle, thus a new solution is proposed. The motion of particles is controlled by velocity parameters. In standard PSO, the velocity control parameters are proposed to be constant over the runtime. However, in this study, a certain modification to accelerate the movement of particles is proposed which consequently would come across with a complementary solution from the updated particle position. The integration of parameters α allows PSO to revise the motion of the particles shown in Equations (5)–(7).

$$v_i^{t+1} = v_i^t \times W + C_1^{t+1} \times r_1 \odot \left[g^* - x_i^t\right] + C_2^{t+1} \times r_2 \odot \left[x_i^* - x_i^t\right] \quad (5)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (6)$$

$$C_1^{t+1} = C_1^t + \alpha \quad (7)$$

where C_1^{t+1} and C_2^{t+1} indicate the velocity parameters, v_i^{t+1} is the velocity, x_i^{t+1} is the updated position vector of particle i at iteration $t+1$, g^* is the global best position, x_i^* is the personal best position in the history of particle i, W is a parameter that controls the influence of the previous velocity of the particle, r_1, r_2 refer to the random numbers between 0 and 1, $\odot$ indicates the multiplication of a scalar with a vector, and α is the acceleration parameter.

Equation (6) is the addition of three velocities, where the first part refers to the velocity in the previous movement (t), like inertia, the second part is called social velocity, and the third part is cognitive velocity. In standard PSO, parameters, W, C_1, and C_2 are kept constant over a runtime, which means a constant influence of three velocities on the final movement. In this study, Equation (5) was added to modify the parameter C_1. A negative α indicates a decrease of C_1 with iterations, and vice-versa. Therefore, the ratio of C_1 and C_2 is changed over a runtime, and thus the relative influence of social and cognitive velocity in the final movement. Table 2 shows the values of acceleration parameters for standard PSO (PSO-A, PSO-B) and modified PSO (MPSO). The value of α depends on the number of iterations, and the initial value of C_1 and C_2, which can be described by the following equation.

$$\alpha = \frac{C_2 - C_1}{d \times total\ iteration} \quad (8)$$

where C_1 and C_2 are velocity parameters and $C_1 > C_2$, d determines how many iterations the value of C_1 would be higher than C_2. For this study, d was set to 0.80 which indicates that from the beginning to 80% of total iterations, the value of C_1 was higher than C_2.

Table 2. Configuration of acceleration parameters in particle swarm optimization (PSO) and modified PSO (MPSO) for 50 iterations.

Configuration	Version	W	C_1			C_2
			Iteration = 0	α	Iteration = 50	
PSO	PSO-A	0.50	1.00	-	1.00	1.80
	PSO-B	0.50	1.00	-	1.00	1.00
Modified PSO	MPSO	0.50	1.80	−0.02	0.80	1.00

Figure 5 shows the change in the ratio of C_1 and C_2 over a runtime. It explains how the ration of social and cognitive velocity is also indirectly controlled where the social velocity is more prominent at the initial phase and cognitive velocity is given more strength at the later phase. Therefore, particles in MPSO tends to move to the individual best

position at the later phase which may lead to a new global best position instead of being stuck in the current best position. For a new optimization problem, the global optimum is unknown, hence, metaheuristic algorithms like PSO cannot differentiate local optima from global optimum. Since PSO is based on the food-searching behavior of animals, it never stops in one location. The efficiency of searching for an improved and better location for food depends on how good the agents, e.g., birds, are directed. From an algorithmic point, this can be represented by tuning the parameters. The final solution returned by algorithm may not be the global optimum for a problem, however, it can be compared with the best solution from many runs of multiple algorithms. In this study, the trend of finding an improved solution is observed to explain whether an algorithm tends to stuck in local minima.

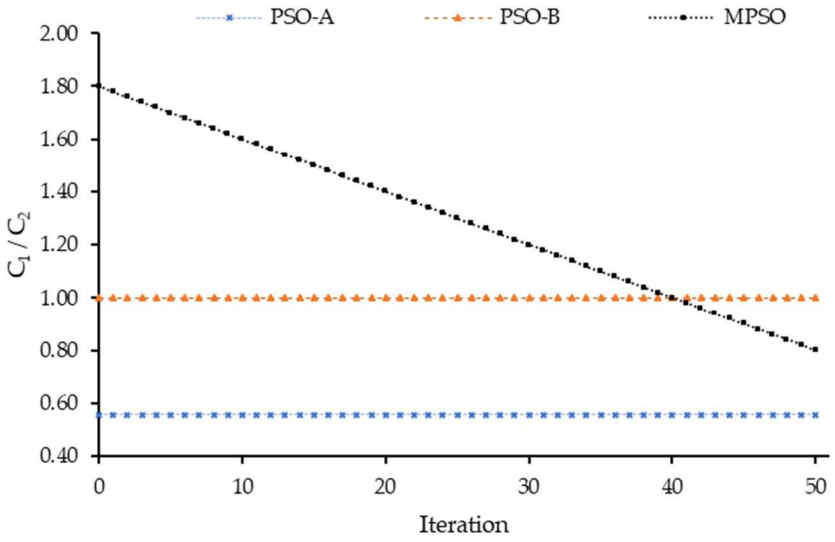


Figure 5. Ratio of acceleration parameters C_1 and C_2 in a runtime.

The construction of the MPSO algorithm proceeds as follows.

- Step 1: The parameters for MPSO are initialized. For this study, the number of particles (i) and number of iterations (t) are 10 and 50, respectively. The cost function of Equation (1) was used as an objective to minimize the cost. The initial position for the particles x_i^0 were allocated randomly and kept the same for all the runs. The initial velocity for the particles v_i^0 were set to 0. The initial velocity parameters, C_1, C_1, and W were 1.80, 1.00, and 0.05 respectively. The acceleration parameter (α) was set to -0.02.
- Step 2: The swarm of particles is used as the solutions to the problem in hybrid NWFSSM to calculate the objective function. The fitness of particles is compared to determine the global best position (g^*) in the swarm and individual best position (x_i^*) in the history of a particle.
- Step 3: The position of the particles is changed using Equations (5) and (6).
- Step 4: The modification of the velocity parameter C_1 is performed using Equation (7).
- Step 5: Steps 2 to 4 repeated until the maximum iteration (t) is met.

To study the influence of modification in searching behavior, the relative Euclidean distance between the current position of particles and the global best position was measured using Equation (9).

$$D_{i,g*}^t = \sqrt{\sum_{d=1}^{n} \left(X_{d,i} - X_{d,g*} \right)^2} \tag{9}$$

where $D_{i,g*}^t$ is the Euclidean distance between particle i and global best position g^* at iteration t, n is the dimension of the position vector, and $X_{d,i}$ and $X_{d,g*}$ indicate the value at index d in the position vector of particle i and global best g^* respectively.

To smooth the Euclidean distance data from 50 iterations, the Savitzky–Golay filter [27] was used with a window size of 5001 (1 iteration had 1000 distance values) and 3rd degree of polynomials. To evaluate the convergence of the mean of the best cost (MBC_t) of one iteration, t was calculated using Equation (10).

$$MBC_t = \frac{\sum_{r=1}^{R} BC_{t,r}}{R} \tag{10}$$

where r and R indicate an instance of run and total runs of the algorithm respectively, and t is the iteration.

2.5.2. PSO Solution Approaches

In PSO, the position vector of particle i consists of continuous values $\left[x_{i,1}^t,\ x_{i,2}^t,\ x_{i,3}^t, \ldots, x_{i,n}^t\right]$, which is considered as one solution to the problem. The dimension of the position vector is the same as the number of variables to optimize i.e., n-dimension for n products. However, a solution in hybrid NWFSSM (Figure 2) is expected to be a sequence of products consisting of discrete values as $\{1,\ 2,\ 3, \ldots, n\}$. Each value in this sequence represents a product. Therefore, the n-dimensional position vector must be converted into a sequence of discrete numbers to reflect the n corresponding products. The smallest position value (SPV) rule has been introduced and widely practiced to meet this requirement. In this arrangement, each of the continuous values in a position vector is conjugated with a product with respect to their indexes. For instance, one position vector $[x_{i,1}^t,\ x_{i,2}^t,\ x_{i,3}^t]$ is conjugated with {1, 2, 3} where 1, 2, and 3 indicate sourdough, square bread, and pan bread, respectively. After an iteration, the index of continuous values in the position vector is changed and sorted by the rule of smallest to the largest value (Table 3). The conjugated products are rearranged according to the sorted index to get a new product sequence. To illustrate, if $x_{i,3}^t$ is the smallest value in the updated position vector, pan bread will be placed first in the output product sequence.

Table 3. Numerical explanation of converting a position vector into a production sequence using the smallest position value (SPV) rule.

Iteration (t)	Position Vector (x_i^t)	Sorted Index *	SPV	
			Input Product Sequence **	Output Product Sequence **
1	[1.24, −0.80, −1.60]	[3, 2, 1]	{1, 2, 3}	{3, 2, 1}
2	[1.04, −0.90, −0.70,]	[2, 3, 1]	{1, 2, 3}	{2, 3, 1}
3	[0.44, −0.10, 0.60]	[2, 1, 3]	{1, 2, 3}	{2, 1, 3}

* Index of values in x_i^t sorted by the smallest to the largest value i.e., index of smallest value placed first; ** each number in the sequence reflects one product and at which position to produce.

2.5.3. Simulated Annealing (SA) and Nawaz-Enscore-Ham (NEH) Algorithms

Simulated annealing is one of the most widely-used metaheuristic optimization algorithms inspired by the annealing procedure of the metal by merging the idea of a metropolis Monte Carlo criterion to avoid being trapped to local minima [28].

The implementation of the SA algorithm proceeds as follows.

- Step 1: The parameters for SA are initialized. The temperature (T), final temperature (T_f), cooling factor (cf), and an initial random solution to the problem are introduced. The proposed solution is a sequence of products that is evaluated using hybrid NWFSSM and assigned as an accepted cost $(Cost_A)$.
- Step 2: Generate a random neighboring solution to the problem. The position of two random products in the accepted sequence is swapped and evaluated to get the cost of the new solution $(Cost_N)$.

- Step 3: The cost of the accepted solution and neighboring solution is compared and the relative difference (Δ) is calculated by using Equation (11).

$$\Delta = \frac{Cost_N - Cost_A}{Cost_A} \tag{11}$$

- Step 4: The relative difference (Δ) is evaluated. If $\Delta \leq 0$, the neighboring solution and its cost are updated as the accepted solution and accepted cost ($Cost_A$), respectively. Otherwise, the Boltzmann distribution function (Equation (12)) is introduced to calculate the probability (p) as the solution acceptance criterion:

$$p = e^{-\frac{\Delta}{T}}. \tag{12}$$

  If $p >$ a random number between 0.6 and 1.0, the neighboring solution and its cost are updated as the accepted solution and accepted cost ($Cost_A$). In other cases, the neighboring solution is discarded.
- Step 5: The temperature (T) is updated by using Equation (13):

$$T = T \times cf. \tag{13}$$

- Step 6: Step 2 to 5 repeated until the final temperature (T_f) is reached. The accepted solution at T_f is counted as the final solution to the problem.

The construction of the NEH algorithm is described below.

- Step 1: The products are sorted in the descending order of the respective total processing time to get a product sequence (PS) e.g., $PS = \{P_1, P_2, \ldots, P_n\}$ where n is the total number of products and product P_n requires the lowest processing time.
- Step 2: The first two products of the PS are taken to construct the subsequences $\{P_1, P_2\}$ and $\{P_2, P_1\}$. The subsequences are evaluated by using hybrid NWFSSM, such that no other products are required to process. The best subsequence of the two products is accepted for further evaluation.
- Step 3: The next product in the PS is taken into account such that an additional product has to be produced. It is inserted at every possible position of the accepted product sequence to construct subsequences.
- Step 4: The subsequences with an additional product are evaluated by using a hybrid NWFSSM and are compared. The subsequence with the lowest cost is assigned as a new accepted product sequence.
- Step 5: Step 3 and 4 are repeated until all the products in PS are added in the accepted product sequence. The final accepted sequence is used as the solution to the problem.

2.6. Approach to Evaluate the Performance of Optimization Algorithms

The optimization algorithms, except NEH, provide different results in different runs. Therefore, there is no certainty about delivering a best or nearly best solution always. Additionally, it is challenging to complete one optimization run within a reasonable time when the number of products, stages, and alternative resources are higher. Therefore, it is crucial to know the performance efficiency of optimization algorithms. To evaluate performance, the optimization was repeated 100 times for each of the algorithms, except NEH, to solve problem I.

In this study, the cumulative frequency was used to evaluate the performance i.e., how frequently an optimizer delivers best, better, good, and worst results. Cumulative frequency of 50 in a 10.0% cost reduction means an optimizer reduced cost by at least 10.0% in 50 runs.

3. Results and Discussion

3.1. Effect of Modification on the Performance of PSO

Figure 6a shows an example of the relative Euclidean distance between the current position of particles and the global best position (g^*) from 100 test runs of MPSO, and Figure 6b represents the distribution of distance values at the 10th iteration, describing almost a normal distribution. However, from the raw Euclidean distance (Figure 6a), no clear pathway was observed. Therefore, the Savitzky Golay method was used to filter the data.

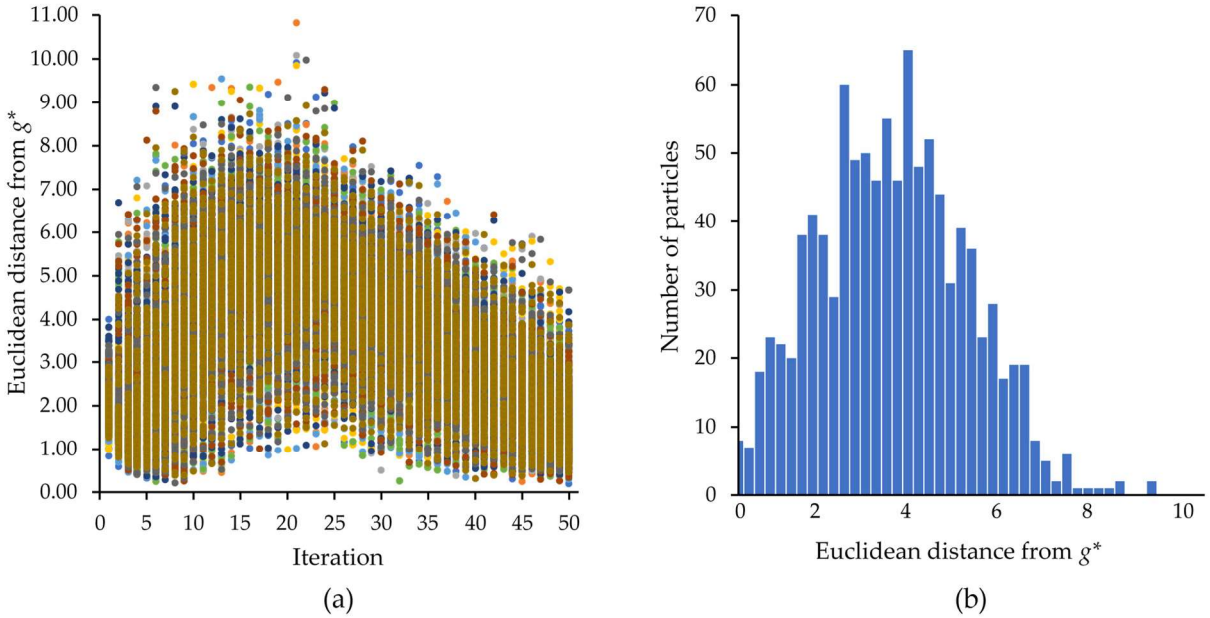


Figure 6. Euclidean distance between the current position of particles and global best position (g^*) recorded from modified particle swarm optimization (MPSO) with 10 particles in 100 test runs: (**a**) From 50 iterations; (**b**) distribution of distance values at 10th iteration.

Figure 7 illustrates the Euclidean distance of particles from the global best position with Savitzky Golay filtering for PSO-A, PSO-B, and MPSO. It shows that at the initial phase, the distance of particles from the global best position (g^*) was nearly the same for all the versions of PSO. In PSO-A, with the constant and lowest ration C_1 / C_2, 0.56, the particles searched solutions closer to the global best position.

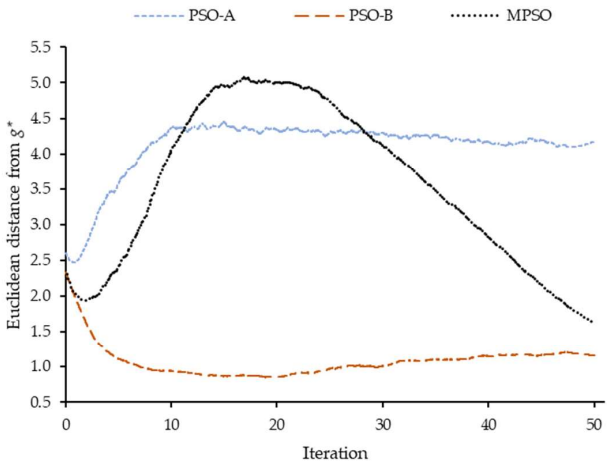


Figure 7. Influence of modification in the pathway of particles.

PSO-B, in contrast, had constant ration C_1 / C_2 of 1.00, and particles were searching solutions always far from the best position. Despite PSO-A and PSO-B showing a rapid change in the first 10 iterations, a constant distance was observed till the end of the runtime. However, due to the change in the ratio of C_1 / C_2 over run time, MPSO had a good combination of searching solutions moderately far away at the initial phase, followed by an inclination toward the best position.

Figure 8 shows the cumulative frequency of standard PSO and MPSO to optimize the cost of different levels. In the worst cases of respective algorithms, MPSO provided a cost reduction of 10.0%, while for PSO-A and PSO-B, it was 7.5% and 6.5%, respectively. The figure shows that the possibility of giving the worst optimization results (<10.0%) by PSO-A is 3%, followed by PSO-B with 9%. However, MPSO showed no such cases in 100 test runs. MPSO outperformed PSO-A and PSO-B in the probability to obtain a better level of optimization results (at least 12.0% cost reduction). In 94 out of 100 runs, MPSO was able to reach this level while for PSO-A and PSO-B, it was 92 and 77, respectively. However, cost reductions by 14.0% or higher were found to be the most difficult level of optimization. At that level, PSO was outperformed by MPSO significantly, provided in 87 runs, which is 25 higher than PSO-A, and 33 higher than PSO-B.

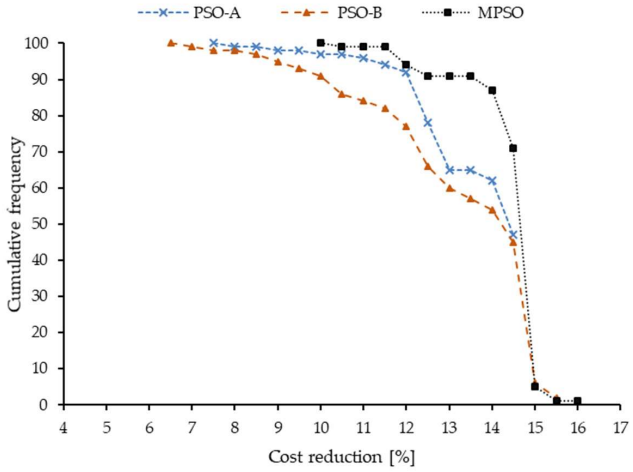


Figure 8. Performance comparison of standard particle swarm optimization (PSO) and modified PSO (MPSO).

Figure 9 shows the trend of searching for an improved solution during a run from PSO-A, PSO-B, and MPSO that turned into the corresponding best optimization results. It demonstrates that MPSO had the best optimization result with a reduction of cost by 16.3%.

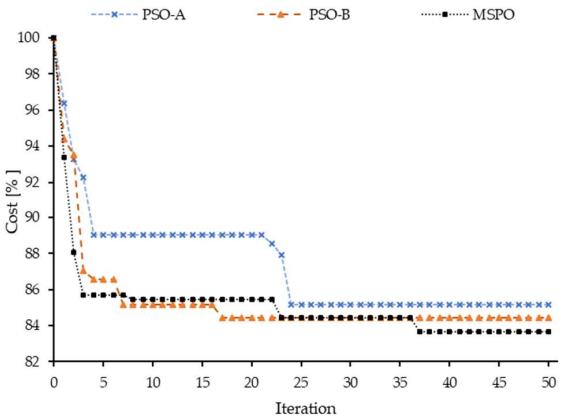


Figure 9. Optimization trend during a run time.

Figure 10 depicts the convergence of the PSO variants calculated from the mean of the best optimization cost at different iterations. PSO-A had the worst mean cost reduction in the first 10 iterations of particles, while PSO-B and MPSO had a similar mean convergence.

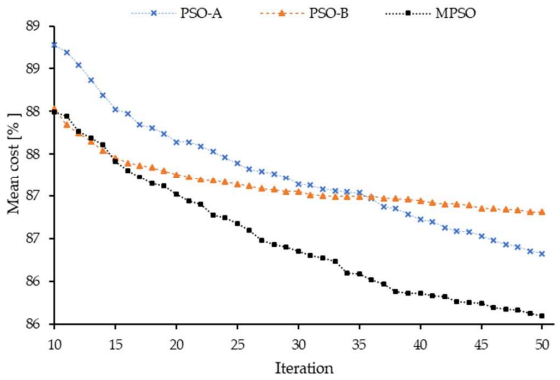


Figure 10. Mean convergence of optimization from 100 test runs.

However, the scenario changed over the next iterations as PSO-A showed a sharp improvement and outperformed PSO-B. MPSO showed the most consistency in improving the solution from the beginning to the end, which can be explained by the convergence rate. The convergence rate describes how the solutions were upgraded in terms of cost reduction by every iteration. The rate of the mean convergence of PSO-A, PSO-B, and MPSO is 0.27%, 0.26%, and 0.28%, respectively. In PSO-B, both cognitive and social velocities of particles in the swarm were equally weighted. According to this study, such configuration showed no visible improvements after the 25th iteration. In contrast, in PSO-A, cognitive velocity was constantly 1.8 times higher compared to the social velocity and outperformed PSO-B. MPSO was configured in a way that initially, the weight to social velocity was relatively higher and gradually decreased to give more strength to cognitive velocity. Such modifications in PSO were found to be efficient to obtain the best optimization results with a higher possibility.

Figure 11 represents the performance comparison of MPSO, SA, and NEH. It shows that NEH optimized the cost of production by 14.0%. However, NEH was outperformed by MPSO 87 times and by SA 72 times in 100 test runs. Moreover, SA provided the very worst quality optimization results with the lowest cost reduction of 4.5%, where MPSO had 10.0% at the worst level of performance. Therefore, it can be said that MPSO outperformed SA and NEH by solving the optimization problem most efficiently.

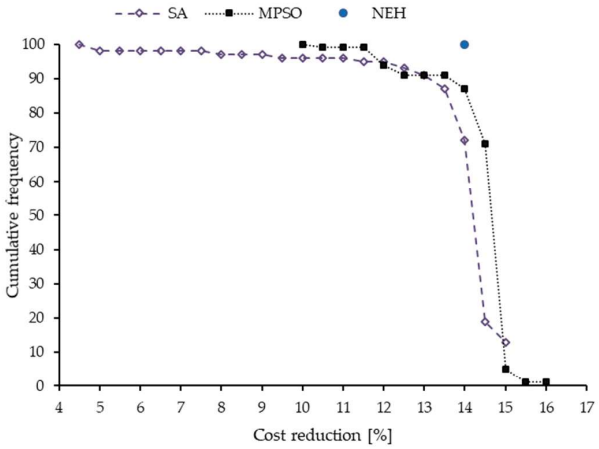


Figure 11. Performance comparison of modified particle swarm optimization (MPSO), simulated annealing (SA), and Nawaz-Enscore-Ham (NEH) to solve problem I.

The best results from the optimized production schedules obtained by MPSO are described here. Figure 12 shows the comparison of idle time of only energy-consuming machines used for the production. It was observed that proper allocation of tasks among machines can reduce the overall idle time significantly. The results show that idle time for mixers, ovens, a fryer, and a slicer were reduced by 25%, 45%, 8%, 76%, and 38%, respectively, whereas for dividers it increased by 54%. The TIDT in the optimized schedule was reduced by 12%, and makespan by 30%. Table 4 shows the cost calculation for the best schedule obtained by MPSO. The optimized cost ($Cost_{opt}$) and initial cost (C_0) were calculated from the optimized and existing schedule respectively using Equation (1). WTIDT represents the cost due to the idle time of the machines calculated by using Equation (4).

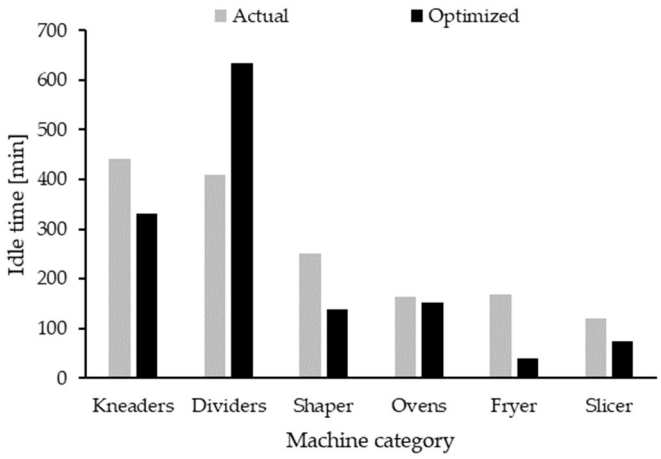


Figure 12. Comparison of machines' idle time in the actual and optimized schedule.

Table 4. Cost calculation for the existing and optimized schedules.

	Makespan [min]	WTIDT	Total Cost	Cost Reduction [%]
Existing schedule	968	3110	4078	16.3%
Optimized schedule	675	2736	3411	

3.2. Performance Comparison between MPSO and PSO from Literature

As a benchmark, problem II and constraints described by Hecker et al. [6] were taken into consideration. The authors found that the standard PSO performed slightly better than ant colony optimization in optimizing makespan. For better comparisons, the objective function was changed to a single objective to optimize makespan as the authors described. Moreover, instead of WTIDT, the comparison was on TIDT.

Figure 13 shows that the best makespan from the study was 8.4% [6], where for the proposed MPSO it was 8.7%. In 10 out of 21 runs, MPSO provided better optimization than the best performance of PSO. MPSO showed a possibility of 90% to optimize the makespan at least by 8.0%, while PSO showed only 47% possibility. The worst performance from MPSO was 7.6% of makespan reduction. The figure describes that there is a possibility (52%) that PSO cannot reach the worst performance of MPSO. In 11 out of 21 runs, PSO showed a relatively lower performance than the worst case of MPSO. However, optimizing one from makespan and TIDT may not be always cost-effective. It was observed that the optimized schedule with the objective of reducing makespan may have a higher TIDT, which causes high energy consumption. Contrarily, a schedule with optimized TIDT may have a higher makespan, meaning the production run time will be longer.

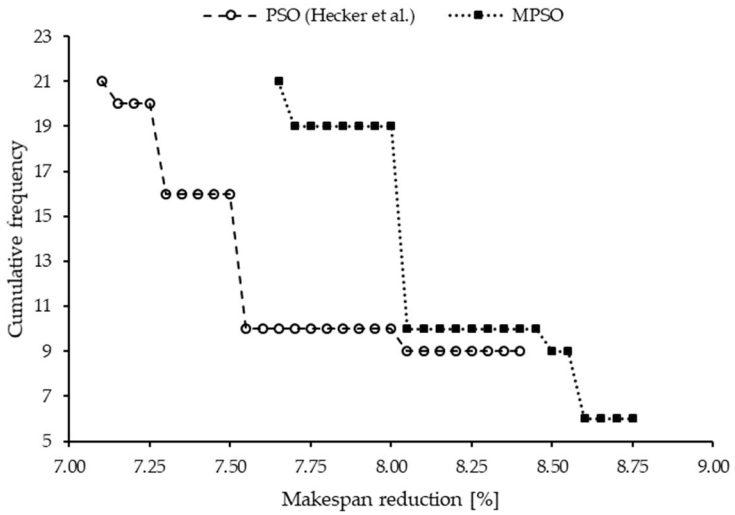


Figure 13. Performance comparison of particle swarm optimization (PSO) implemented by Hecker et al. [6] with modified PSO (MPSO) to optimize the makespan of problem II.

Figure 14 shows the effectiveness of the single objective optimized results. As makespan reduction was the only objective, TIDT was ignored by the optimization algorithm. The result shows that despite a higher makespan reduction, in some cases, the TIDT was increased, indicated by a negative TIDT [%]. In the worst scenario, the TIDT was increased by 10%, while makespan was reduced by 8%. Similar results were also obtained by Hecker et al. [6].

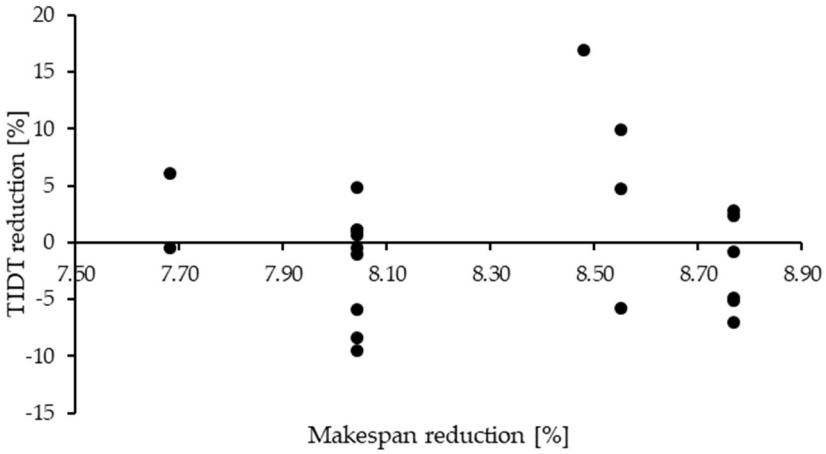


Figure 14. The effect of makespan optimization on total idle time (TIDT) in problem II.

From the industrial perspective, it depends on how the makespan and TIDT influence the overall production cost which may vary from region to region and bakery to bakery. Therefore, the addition of a weighting factor for each machine may allow modifying the cost function to reflect the real production cost.

4. Conclusions and Outlook

The motivation of this study was to develop a task allocation model for production in bakery industries consisting of complex and mixed constraints, along with searching for an optimized schedule. The integration of many constraints, mainly to make feasible, increases the difficulty to optimize the production. However, the results demonstrated that an optimized schedule could still reduce the makespan and machine idle time of an

existing production line significantly by performing heuristic and metaheuristic optimization algorithm.

Furthermore, the robustness and reliability of the optimization algorithms were examined. The study revealed that the convergence rate of PSO could be highly increased by adjusting the acceleration ratio to the best configuration using supporting parameters. Such modification in PSO could outperform standard PSO, SA, and NEH.

This study might be valuable for bakery industries and decision-making authorities to design an efficient bakery production system. A change in the arrangement e.g., addition of machines, employees, or changing the work plan of employees can be conformed to monitor the production efficiency before implementation in a real production line.

Several ideas deserve further investigation, such as the integration of more constraints that might occur in specific bakeries to provide customization opportunities to the bakery industry. Moreover, the application of other heuristic and meta-heuristic algorithms and a combination of their best features can be explored to find a robust optimizer.

Author Contributions: M.B. performed conceptualization, methodology, software, formal analysis, investigation, visualization, writing—original draft preparation. J.S. and D.R. contributed with the resources and data curation. C.M.R. provided the resources and played the role of project administration. 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 research has been funded by the EIT Food of the European Institute of Innovation and Technology (EIT), a body of the European Union, the EU Framework Programme 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: Two datasets were used for this study. Data used for problem I is not publicly available due to restrictions, e.g., privacy and ethical reason. However, data used for problem II was taken from Hecker et al. [4] and are available at the corresponding journal site.

Acknowledgments: The data used for this study (problem I) was collected from Panishop Bakeries (Zaragoza, Spain) as a part of the EIT Food project. Apart from that, the study was not influenced by any other factors.

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

References

1. Garey, M.R.; Johnson, D.S.; Sethi, R. The Complexity of Flowshop and Jobshop Scheduling. *Math. Oper. Res.* **1976**, *1*, 117–129. [CrossRef]
2. Bewoor, L.; Prakash, V.C.; Sapkal, S.U. Evolutionary Hybrid Particle Swarm Optimization Algorithm for Solving NP-Hard No-Wait Flow Shop Scheduling Problems. *Algorithms* **2017**, *10*, 121. [CrossRef]
3. Abdollahpour, S.; Rezaian, J. Two new meta-heuristics for no-wait flexible flow shop scheduling problem with capacitated machines, mixed make-to-order and make-to-stock policy. *Soft Comput.* **2016**, *21*, 3147–3165. [CrossRef]
4. Chen, R.-X.; Li, S.-S.; Li, W.-J. Multi-agent scheduling in a no-wait flow shop system to maximize the weighted number of just-in-time jobs. *Eng. Optim.* **2018**, *51*, 217–230. [CrossRef]
5. Sun, H.; Jiang, A.; Ge, D.; Zheng, X.; Gao, F. A Chance Constrained Programming Approach for No-Wait Flow Shop Scheduling Problem under the Interval-Valued Fuzzy Processing Time. *Processes* **2021**, *9*, 789. [CrossRef]
6. Hecker, F.T.; Hussein, W.B.; Paquet-Durand, O.; Hussein, M.A.; Becker, T. A case study on using evolutionary algorithms to optimize bakery production planning. *Expert Syst. Appl.* **2013**, *40*, 6837–6847. [CrossRef]
7. 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]
8. Duarte, B.P.M.; Gonçalves, A.M.; Santos, L.O. Optimal Production and Inventory Policy in a Multiproduct Bakery Unit. *Processes* **2021**, *9*, 101. [CrossRef]

9. Huber, J.; Stuckenschmidt, H. Intraday shelf replenishment decision support for perishable goods. *Int. J. Prod. Econ.* **2020**, *231*, 107828. [CrossRef]
10. Nawaz, M.; Enscore, E.E.; Ham, I. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega* **1983**, *11*, 91–95. [CrossRef]
11. Pan, Q.-K.; Gao, L.; Wang, L.; Liang, J.; Li, X.-Y. Effective heuristics and metaheuristics to minimize total flowtime for the distributed permutation flowshop problem. *Expert Syst. Appl.* **2019**, *124*, 309–324. [CrossRef]
12. Sha, D.; Lin, H.-H. A multi-objective PSO for job-shop scheduling problems. *Expert Syst. Appl.* **2010**, *37*, 1065–1070. [CrossRef]
13. Tian, X.; Liu, X. Improved Hybrid Heuristic Algorithm Inspired by Tissue-Like Membrane System to Solve Job Shop Scheduling Problem. *Processes* **2021**, *9*, 219. [CrossRef]
14. Lin, S.-W.; Cheng, C.-Y.; Pourhejazy, P.; Ying, K.-C. Multi-temperature simulated annealing for optimizing mixed-blocking permutation flowshop scheduling problems. *Expert Syst. Appl.* **2020**, *165*, 113837. [CrossRef]
15. Zhan, X.; Xu, L.; Ling, X. Task Scheduling Problem of Double-Deep Multi-Tier Shuttle Warehousing Systems. *Processes* **2020**, *9*, 41. [CrossRef]
16. Sun, X.; Wang, Y.; Kang, H.; Shen, Y.; Chen, Q.; Wang, D. Modified Multi-Crossover Operator NSGA-III for Solving Low Carbon Flexible Job Shop Scheduling Problem. *Processes* **2020**, *9*, 62. [CrossRef]
17. Fong, S.; Lou, H.-L.; Zhuang, Y.; Deb, S.; Hanne, T. Solving the permutation flow shop problem with firefly algorithm. In Proceedings of the 2014 2nd International Symposium on Computational and Business Intelligence, New Delhi, India, 7–8 December 2014; pp. 25–29.
18. Eberhart, R.; Kennedy, J. A new optimizer using particle swarm theory. In Proceedings of the MHS'95 Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 4–6 October 1995.
19. Sun, T.; Xu, M.-H. A Swarm Optimization Genetic Algorithm Based on Quantum-Behaved Particle Swarm Optimization. *Comput. Intell. Neurosci.* **2017**, *2017*, 1–15. [CrossRef]
20. Chan, C.-L.; Chen, C.-L. A cautious PSO with conditional random. *Expert Syst. Appl.* **2015**, *42*, 4120–4125. [CrossRef]
21. Zhang, L.; Wu, J. A PSO-Based Hybrid Metaheuristic for Permutation Flowshop Scheduling Problems. *Sci. World J.* **2014**, *2014*, 902950. [CrossRef]
22. Jiang, Q.; Liao, X.; Zhang, R.; Lin, Q. Energy-Saving Production Scheduling in a Single-Machine Manufacturing System by Improved Particle Swarm Optimization. *Math. Probl. Eng.* **2020**, *2020*, 8870917. [CrossRef]
23. Ji, M.; Yang, Y.; Duan, W.; Wang, S.; Liu, B. Scheduling of no-wait stochastic distributed assembly flowshop by hybrid PSO. In Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC), Vancouver, BC, Canada, 24–29 July 2016; pp. 2649–2654.
24. Issa, M.; Hassanien, A.E.; Oliva, D.; Helmi, A.; Ziedan, I.; Alzohairy, A. ASCA-PSO: Adaptive sine cosine optimization algorithm integrated with particle swarm for pairwise local sequence alignment. *Expert Syst. Appl.* **2018**, *99*, 56–70. [CrossRef]
25. Van Rossum, G. *Python Tutorial*; Technical Report CS-R9526; Centrum voor Wiskunde en Informatica (CWI): Amsterdam, The Netherlands, 1995.
26. Pinedo, M.L. *Scheduling: Theory, Algorithms, and Systems*, 5th ed.; Springer: Berlin/Heidelberg, Germany, 2016; ISBN 978-3-319-26578-0.
27. Savitzky, A.; Golay, M.J.E. Smoothing and Differentiation of Data by Simplified Least Squares Procedures. *Anal. Chem.* **1964**, *36*, 1627–1639. [CrossRef]
28. Kirkpatrick, S.; Gelatt, C.D.; Vecchi, M.P. Optimization by Simulated Annealing. *Science* **1983**, *220*, 671–680. [CrossRef] [PubMed]

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

Majharulislam Babor, Line Pedersen, Ulla Kidmose, Olivier Paquet-Durand and Bernd Hitzmann

Citation

M. Babor, L. Pedersen, U. Kidmose, O. Paquet-Durand and B. Hitzmann. Application of Non-Dominated Sorting Genetic Algorithm (NSGA-II) to Increase the Efficiency of Bakery Production: A Case Study. Processes, Volume 10, Issue 8, 1623, August 2022. https://doi.org/10.3390/pr10081623

Note: The references cited in the following research article are listed at the end of this section and are not included in the bibliography.

Publications and Findings

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

Processes **2022**, *10*, 1623

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.

Processes **2022**, *10*, 1623

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.

Processes **2022**, *10*, 1623

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

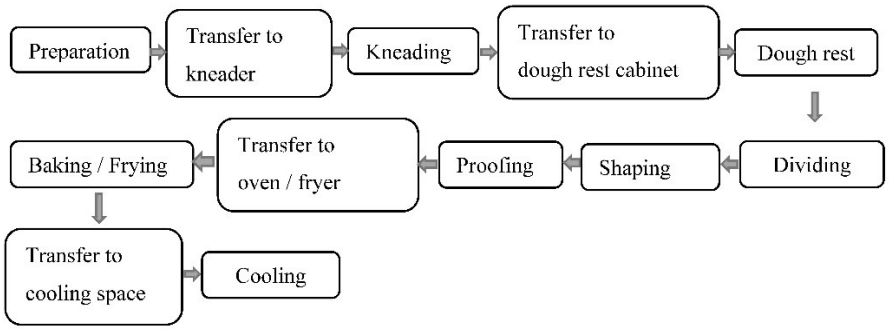


Figure 1. A simplified processing route for bakery products.

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.

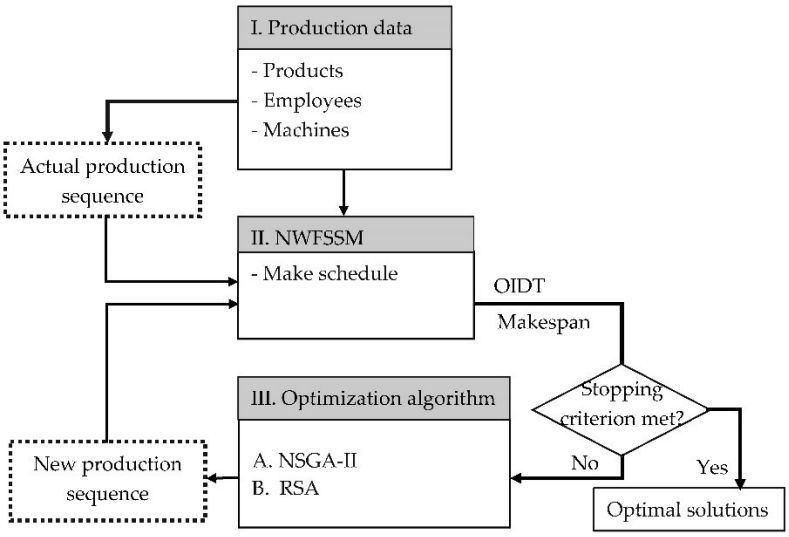


Figure 2. Schematic flow diagram of bakery production optimization.

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

Processes **2022**, *10*, 1623

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.

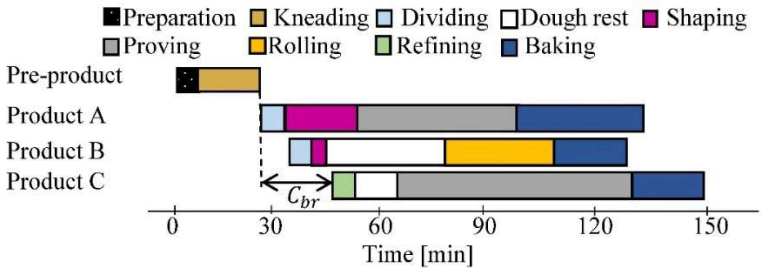


Figure 3. A schedule for a simplified group of products that share the same dough. C_{br} shows the bowl resting time for product C.

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.

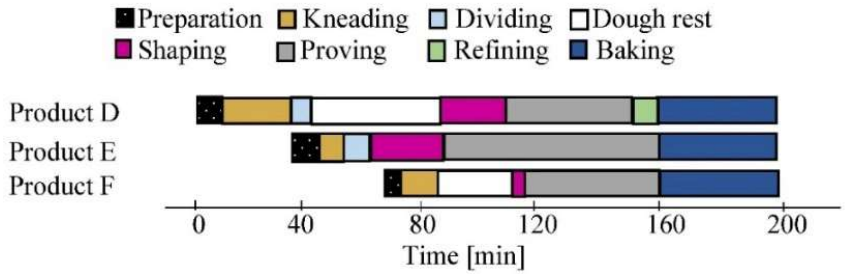


Figure 4. A schedule for one simplified group of products baked in one oven compartment at the same time.

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.

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

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.

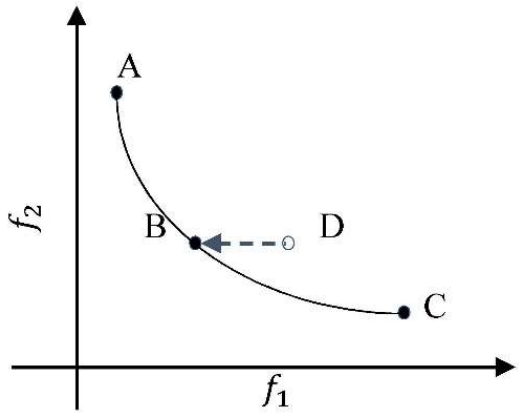


Figure 5. Pareto frontier for two objective functions of the minimization problem.

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.

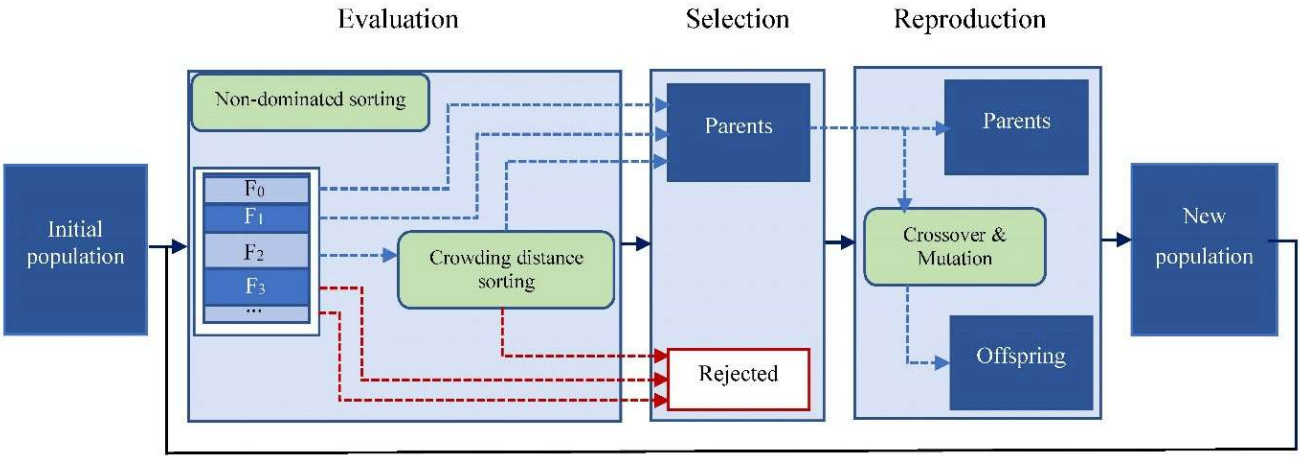


Figure 6. NSGA-II procedure. Non-dominated sorting divides the population into different ranks. F_0, F_1, F_2, ... are different ranks where a rank contains the individuals of the same front of optimality.

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].

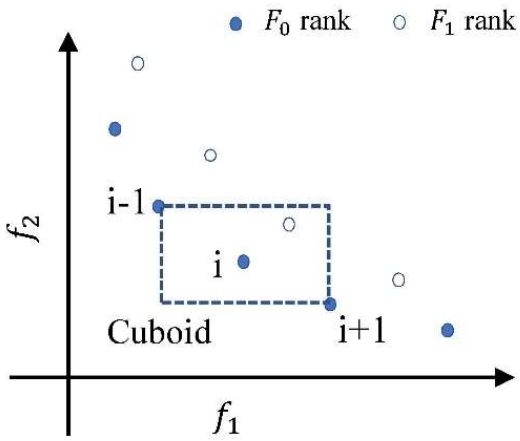


Figure 7. Crowding distance calculation for individual *i*. f_1 and f_2 represent two objectives. Filled circles show the individuals' fitness of the same non-dominated front F_0.

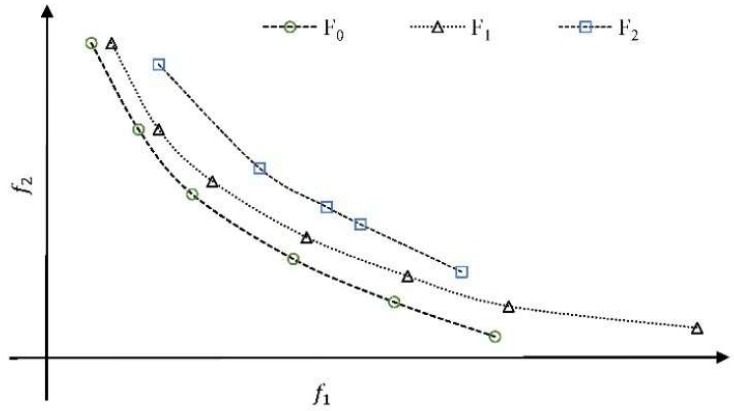


Figure 8. Concept of ranks using the fast non-dominated sorting approach. F_0, F_1, and F_2 are different ranks. Solutions in F_0 rank are Pareto optimal. Solutions in F_1 and F_2 are suboptimal and dominated by at least one of the other solutions.

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.

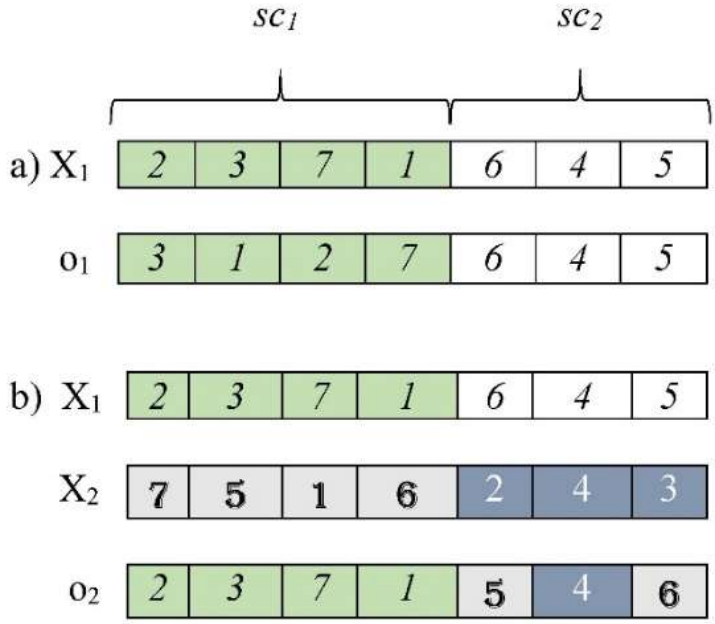


Figure 9. Concept of single-point crossover. X_1, X_2 are parent, and o_1, o_2 are offspring, as shown in (**a**,**b**) respectively. A random point 4 divides each parent into two sections, sc_1 and sc_2. X_{1,SC_1} is Section 1 of the parent X_1.

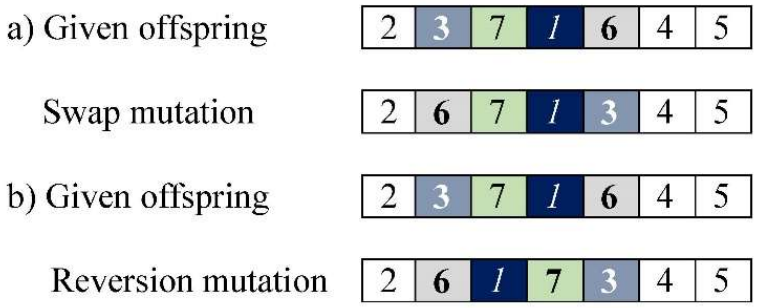


Figure 10. (**a**) Swap mutation and (**b**) reversion mutation for the same solution with two random points, 2 and 5.

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.

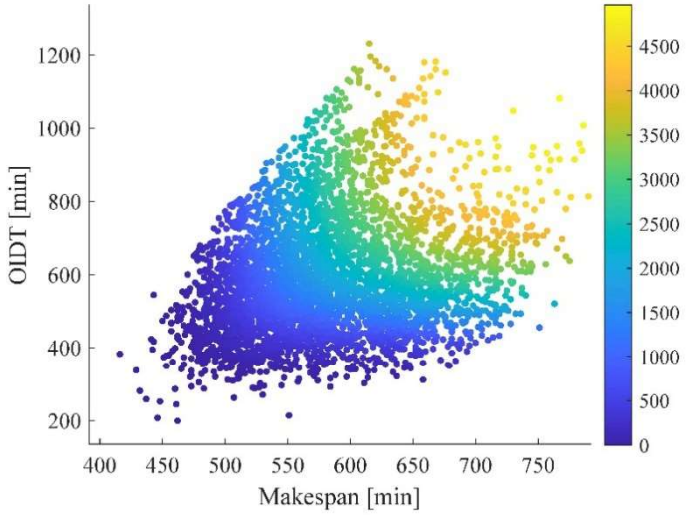


Figure 11. The fitness of the solutions generated during one runtime of NSGA-II for BMP V. The color gradient shows the order of the ranks.

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%.

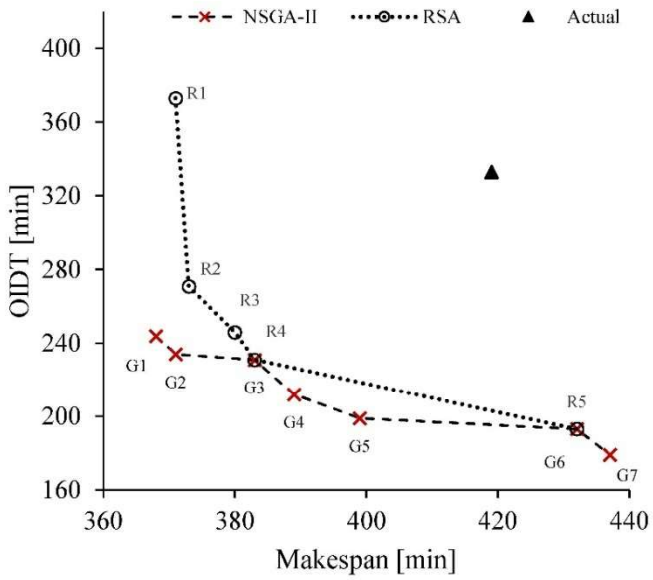


Figure 12. The solutions for BMP-I obtained by NSGA-II and RSA.

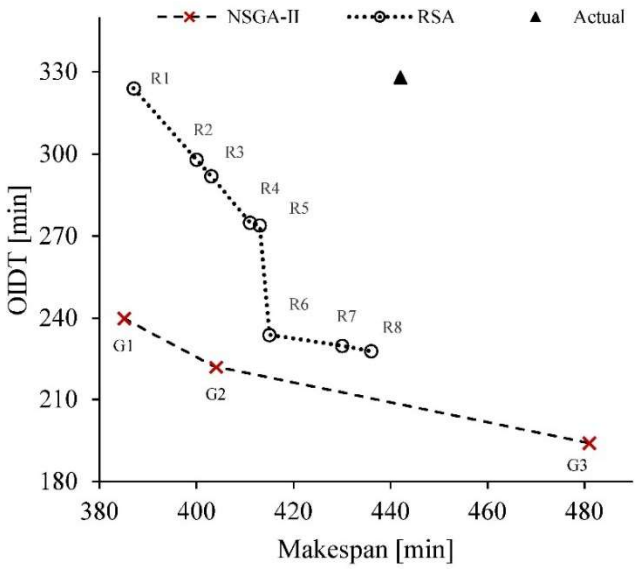


Figure 13. The solutions for BMP-II obtained by NSGA-II and RSA.

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.

Processes **2022**, *10*, 1623

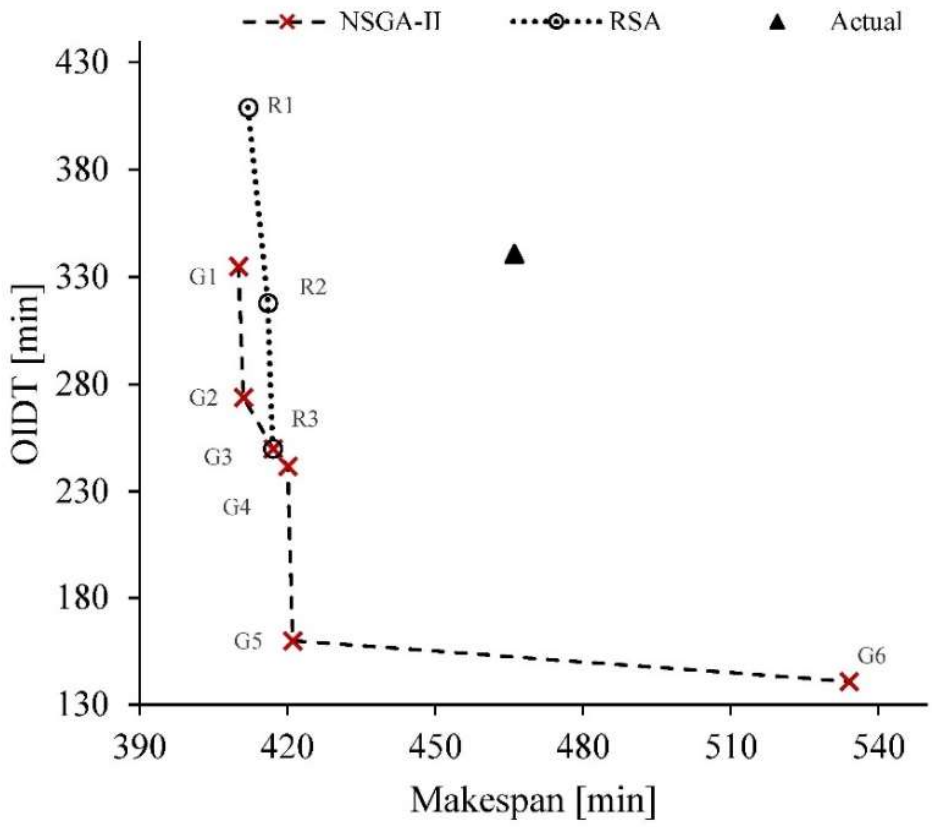


Figure 14. The solutions found by NSGA-II and RSA for BMP-III.

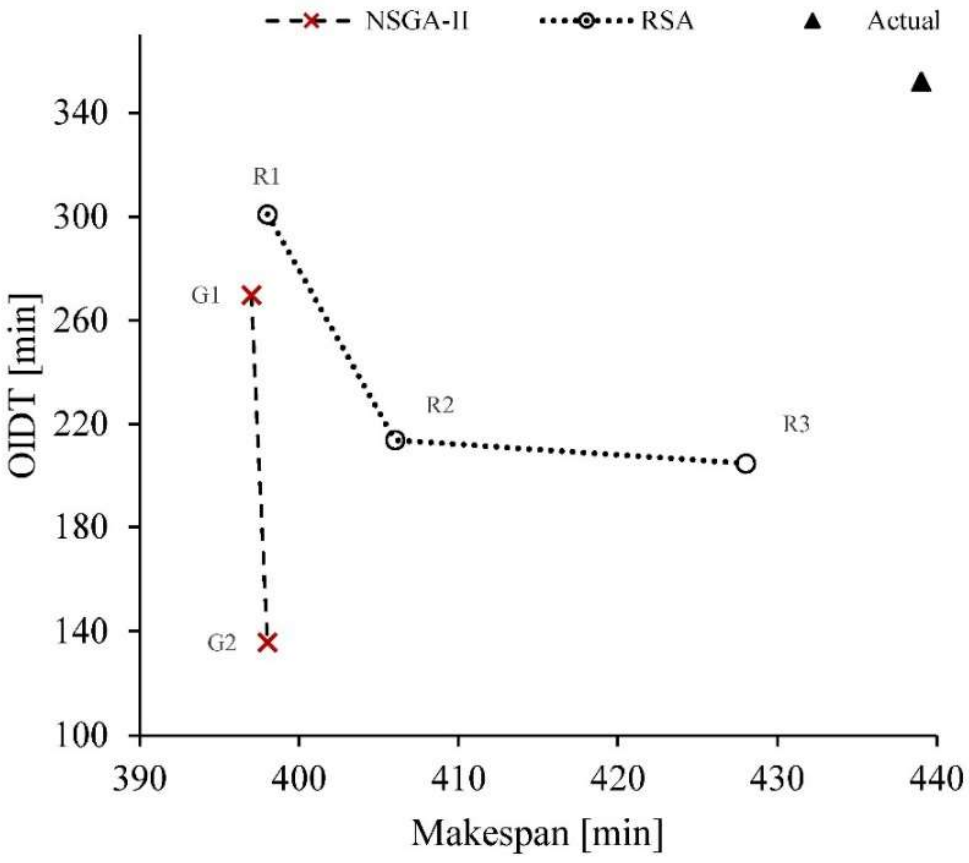


Figure 15. The solutions found by NSGA-II and RSA for BMP-IV.

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.

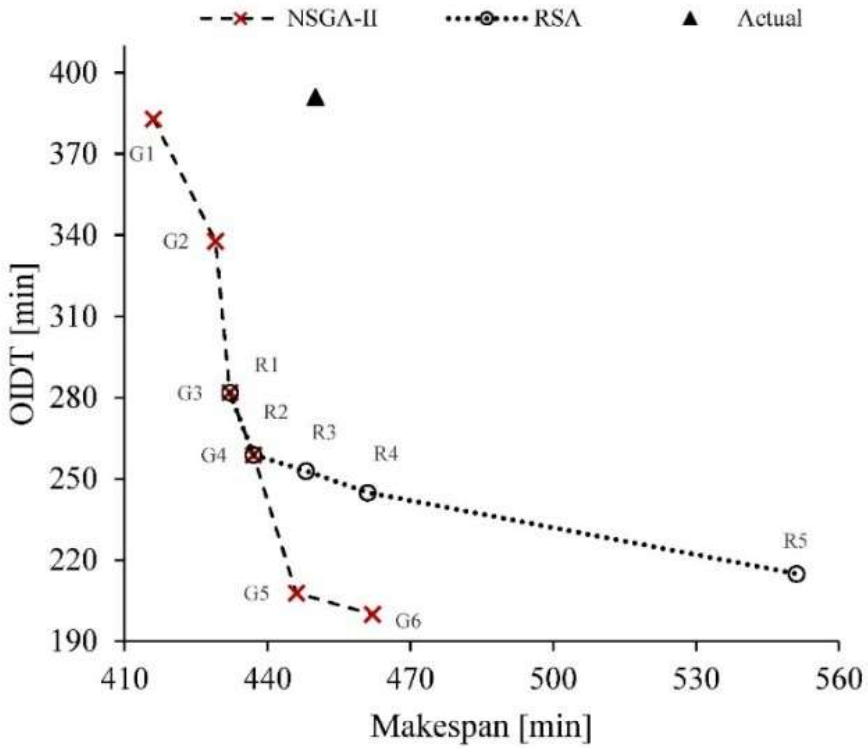


Figure 16. The solutions offered by NSGA-II and RSA for BMP-V.

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.

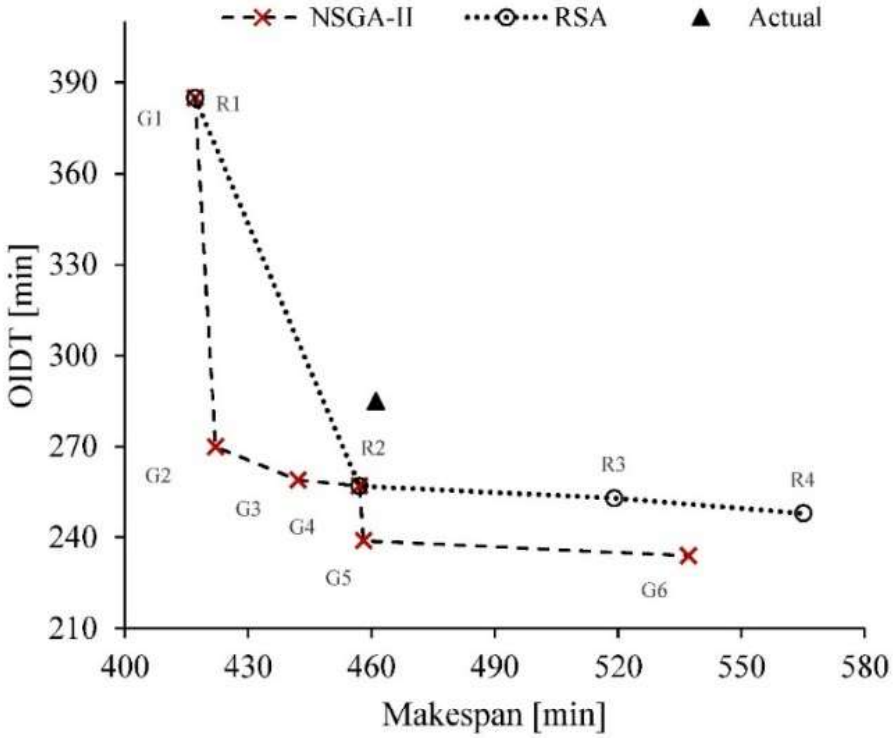


Figure 17. The solutions obtained by NSGA-II and RSA for BMP-VI.

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)

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

Algorithm A2. Single-point crossover for product sequence

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

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

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]

3.3. Modeling and Optimization of Bakery Production Scheduling to Minimize Makespan and Oven Idle Time

Majharulislam Babor, Olivier Paquet-Durand, Reinhard Kohlus and Bernd Hitzmann,

Citation

M. Babor, O. Paquet-Durand, R. Kohlus and B. Hitzmann. Modeling and Optimization of Bakery Production Scheduling to Minimize Makespan and Oven Idle Time. Scientific reports, volume 13, article 235 (2023), January 2023. https://doi.org/10.1038/s41598-022-26866-9

Note: The references cited in the following research article are listed at the end of this section and are not included in the bibliography.

scientific reports

Check for updates

OPEN

Modeling and optimization of bakery production scheduling to minimize makespan and oven idle time

Majharulislam Babor[1]✉, Olivier Paquet-Durand[1], Reinhard Kohlus[2] & Bernd Hitzmann[1]

Makespan dominates the manufacturing expenses in bakery production. The high energy consumption of ovens also has a substantial impact, which bakers may overlook. Bakers leave ovens running until the final product is baked, allowing them to consume energy even when not in use. It results in energy waste, increased manufacturing costs, and CO_2 emissions. This paper investigates three manufacturing lines from small and medium-sized bakeries to find optimum makespan and ovens' idle time (OIDT). A hybrid no-wait flow shop scheduling model considering the constraints that are most common in bakeries is proposed. To find optimal solutions, non-dominated sorting genetic algorithm (NSGA-II), strength Pareto evolutionary algorithm (SPEA2), generalized differential evolution (GDE3), improved multi-objective particle swarm optimization (OMOPSO), and speed-constrained multi-objective particle swarm optimization (SMPSO) were used. The experimental results show that the shortest makespan does not always imply the lowest OIDT. Even the optimized solutions have up to 231 min of excess OIDT, while the makespan is the shortest. Pareto solutions provide promising trade-offs between makespan and OIDT, with the best-case scenario reducing OIDT by 1348 min while increasing makespan only by 61 min from the minimum possible makespan. NSGA-II outperforms all other algorithms in obtaining a high number of good-quality solutions and a small number of poor-quality solutions, followed by SPEA2 and GDE3. In contrast, OMOPSO and SMPSO deliver the worst solutions, which become pronounced as the problem complexity grows.

Bakery is one of the major food manufacturing sectors, with steady increases in market share and per capita consumption. Craft bakery sales in Germany in 2021 were 14.9 billion Euros (exclusive of VAT), with an increase of 0.18 billion Euros per year. To meet market demand, the amount of flour consumed, the variety of products developed, and the number of personnel employed have all expanded over the past decade. According to reports, each bakery uses on average of 372 MWh of energy annually, resulting in 101 tons of CO_2 emissions[1,2]. As the business environment has become more competitive, the objectives for improving the efficiency of a manufacturing system have widened. In order to satisfy customers, meet market demand, and turn a profit, an optimum cost-time profile is crucial. It includes cost savings via the efficient use of assets and materials. Makespan, tardiness, earliness, and energy consumption are some of the most commonly employed cost-cutting objectives in various production environments. However, bakery manufacturing, particularly in small and medium-sized bakeries, is prone to inefficiencies because employees perform many tasks manually for operations that cannot be automated. Furthermore, employee salaries are said to account for a significant amount of the cost[1]. As a result, when planning the production schedule, bakers focus primarily on lowering the makespan.

Production scheduling with more than two machines is a non-deterministic polynomial-time (NP)-hard problem[3,4]. The difficulty of finding the best schedule increases as the number of products, processing stages, and alternative machines for each stage grows. Therefore, the flow shop scheduling problem has been extensively studied to improve the efficiency of several production and service environments, such as bakery[5–7], glass[8], steel[9], wood[10], chemical process[11], energy system[12], healthcare system[13–15]. To put it simply, it is the process of allocating tasks of varying durations from n products to m machines. It also provides supplementary information for assessing a schedule, such as makespan and energy use, that change based on how the tasks are allocated. The most

[1]Institute of Food Science and Biotechnology, Department of Process Analytics and Cereal Science, University of Hohenheim, 70599 Stuttgart, Germany. [2]Institute of Food Science and Biotechnology, Department of Process Engineering and Food Powders, University of Hohenheim, 70599 Stuttgart, Germany. ✉email: majhar@uni-hohenheim.de

common type of flow shop scheduling problem is the permutation flow shop, in which each product must pass through all the *m* machines independently in the same order[4]. In complex cases, a processing task of a product may depend on another product. It is known as "no-wait" flow shop scheduling problem when there is no delay allowed between two successive tasks of a product. Many hybrid flow shop models have been developed to reflect the reality, which is mostly specific to a production system[16]. To make production systems energy-efficient and environmentally friendly, many flow shop models have been proposed, which are widely known as "green flow shop model". Here, in order to establish an efficient resource allocation, total energy consumption is taken into account in addition to makespan[17].

Although modern industries have been applying many powerful decision-making tools, such as intelligent manufacturing systems, to address complex challenges[18,19], small and medium-sized bakeries continue to rely on personal experience[6,7]. Furthermore, bakeries' product range and amount change frequently due to market demand and seasonality, demanding continuous monitoring of production efficiency. However, only a few studies have focused on improving bakery manufacturing. In a recent study, Huber and Stuckenschmidt[20] implemented machine learning approaches to predict hourly sales of bakery items in a retailer store and optimize the baking schedule so that bakers serve customers with fresh products. Nonetheless, the production of a vast number of products, from flour to finished or unfinished goods before delivery to retailers, is a separate segment. In a case study with a medium-sized German bakery, Hecker et al.[6] observed that the makespan of an existing manufacturing line with 40 products can be lowered by 8.6%. Swangnop et al.[7] developed a scheduling model for a bakery in Thailand and demonstrated that the existing production, planned based on experience, is inefficient. An additional factor that most bakers overlook is the ovens' energy consumption, which has a vital influence on manufacturing costs and CO_2 emissions. It has been reported that only baking consumes up to 78% of the total energy depending on the product category[21]. Bakeries typically feature multiple ovens with varied functionalities that are employed according to product specifications[5,6]. As a result, by minimizing the idle time of the ovens, a large quantity of energy can be saved, lowering manufacturing costs and CO_2 emissions. Babor et al.[5] investigated a small Spanish bakery and observed that actual production is poorly optimized. The authors weighed the machines' idle time and makespan to the objective function.

The motivation for this study is twofold. First, to solve production optimization problems for bakeries with two objectives: minimizing makespan and energy waste due to oven idle time. It is a hybrid no-wait flow shop scheduling problem because of the following exceptions in bakery manufacturing. Many tasks are carried out manually, and there are numerous substitute machines that can carry out the remaining tasks. Additionally, there are production constraints for a variety of products recipes. Therefore, a mixed-integer linear programming approach for hybrid no-wait flow shop scheduling model (HNFSM) is proposed to simulate bakery production scheduling. The Pareto optimal solutions obtained by multi-objective optimization algorithms are used to analyze the trade-offs between the objectives. Secondly, to compare the performance of five multi-objective optimization algorithms to solve the instances. Because production optimization is time-consuming and performed frequently, attaining optimal solutions in the lowest computation time is essential. We used multi-objective optimization algorithms of two types: evolutionary algorithms and particle swarm optimization-based metaheuristics. Non-dominated sorting genetic algorithm (NSGA-II), strength Pareto evolutionary algorithm (SPEA2), and generalized differential evolution (GDE3) are taken from the former category, while improved multi-objective particle swarm optimization (OMOPSO), and speed-constrained multi-objective particle swarm optimization (SMPSO) are from the latter. To assess their effectiveness, four quality indicators are used: cardinality, convergence, distribution and spread, and convergence and distribution of the obtained solutions. To cluster the solutions into distinct qualities, a Gaussian mixture model[22] is used.

The following are the contributions of the current study. State-of-the-art multi-objective optimization methods are used to optimize the production efficiency of small and medium-sized bakeries employing a hybrid no-wait flow shop model. By combining various performance metrics, the Gaussian mixture model is used to assess how effectively algorithms solve problems of three complexity levels while varying the number of products and predecessor constraints.

The remainder of the paper is structured as follows. An introduction to bakery production is given in the next section. "Materials and methods" section describes mathematical formulation of a hybrid no-wait flow shop scheduling model to simulate bakery manufacturing. Besides, multi-objective optimization algorithms and their performance indicators are presented. In "Results and discussion" section, the effectiveness of algorithms in solving scheduling problems for bakeries is analysed. "Conclusions and future works" section of this paper provides a summary of findings.

Bakery production

A bakery product undergoes a series of processing steps. Each product has a recipe that specifies the order, duration, and machines that will be utilized to complete the tasks. Figure 1 shows a simplified processing route. Making the dough starts by mixing and kneading the ingredients, such as flour, water, and salt. In most cases, yeast is added to induce fermentation, which produces the leavening agent CO_2 and other aroma precursors. If small and medium-sized bakeries are considered, the transfer of unfinished products from one machine to another is performed manually by employees almost after every processing task. The fermentation process has a vital impact on the final product's quality. Because temperature and humidity have a considerable impact on yeast sugar fermentation, the duration of the processing stages under various conditions is strictly controlled. Performing one task longer than the predefined duration results in overtreatment and consequently, delay for the following tasks, and a loss of product quality, both of which are undesirable. Therefore, as soon as ingredients are mixed and kneaded, the next stages are carried out with no delay.

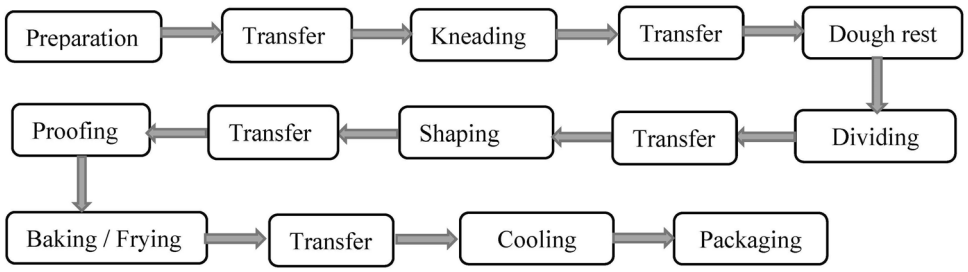


Figure 1. A simplified processing route for bakery products.

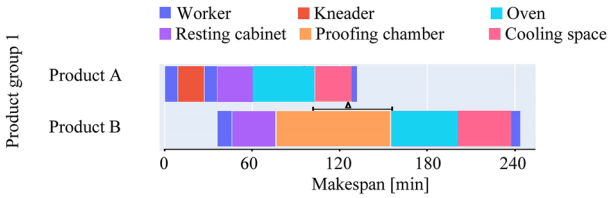


Figure 2. Gantt chart showing schedule of one product group with unified initial stages. The triangle (Δ) shows the oven idle time.

To avoid wasting energy, bakers turn off machines during the idle time—the time between two scheduled operations. However, ovens need preparation time to reach the set temperature before performing baking. Turning them off after an operation requires a restart well ahead of the next operation. When they are turned off, no energy is consumed, but the temperature steadily declines. If the idle time between two tasks is short, the temperature drops less, and the time required to re-heat is reduced. However, if the temperature drops sharply, such as due to chilly weather, the assumed time may not be long enough to reach the set point. Again, with the prolonged idle period, the right time to restart the ovens must be considered to avoid wasting energy and have them ready at the proper time. When the number of products is large and there are many manual tasks to perform, it is difficult for bakers to keep track. The following tasks must be postponed accordingly if the oven's temperature is not up to the set point in time. It may affect the product quality and lead to inefficient production. To avoid these consequences, bakers keep the ovens running throughout the production time.

The duration and machine set up for processing steps are predetermined. Hence, the energy consumption during operational time is constant regardless of how optimized a schedule is. In contrast, the effectiveness of the schedule influences oven idle time, which has a direct impact on the quantity of wasted energy. Moreover, small, and medium-sized manufacturers have limitations in recording energy data for each device. In this case, the idle time of the ovens can be an ideal indicator of energy waste. Machine idle time has been investigated in various studies as one of the objectives for optimizing production schedule[5,6,23,24].

Materials and methods

In this paper, three bakery production optimization problems, labelled with BK15, Bk40 and BK50 were solved. The number in labels specifies how many products were produced in each manufacturing line, for example, the dataset BK15 contains production information for 15 bakery products. BK15 and BK50 were taken from Babor and Hitzmann[25] and BK40 was taken from Hecker et al.[6]. BK15 had three employees, eight machines, and two ovens with four compartments; BK40 had eleven machines, three ovens, and nine compartments; and BK50 had ten employees, fourteen machines, and three ovens, and ten compartments. In bakeries, many ovens have separate compartments, each of which can be used independently to bake a batch of products. In the following discussion, the problems are labelled according to the approximate number of total products. The implementation and simulation of HNFSM and multi-objective optimization algorithms were performed using the computer language Python (version 3.7)[26] on a computer running Microsoft Windows 10 as the operating system with a configuration of an Intel Core i5 at 4 × 3.20 GHz, 8 GB ram.

Problem definition and scheduling model. In small, and medium-sized bakeries, using the same dough for various products made from the same ingredients is a frequent practice. This practice takes advantage of the machines' capabilities in the initial stages to reduce preparation time. Bakers split the dough after completing a few processing tasks into various parts. It enables the products to be treated differently in subsequent phases to meet recipe requirements. There is no common procedure for separating dough as it completely depends on the type of products and recipes. Figure 2 illustrates a schedule of two products that are produced from the same dough and shared the same processing machines at the initial phase.

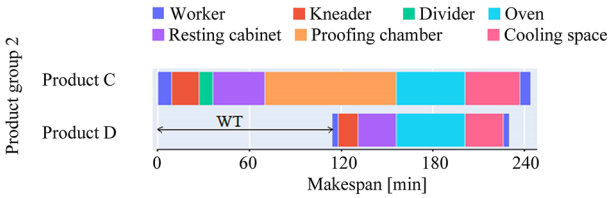


Figure 3. Simplified schedule for one product group where baking is performed together. Here WT is the difference between the start time of Product C and Product D.

Group	Product	Bowl time [min]	Name	Processing stage	Duration [min]	Machine/Employee
1	1	0	Product A	Preparation	9	Employee
				Kneading	18	Kneader
				Dividing	9	Employee
				Dough rest	25	Resting cabinet
				Baking	42	Oven
				Cooling	25	Cooling space
				Packaging	4	Employee
	2	36	Product B	Dividing	10	Employee
				Dough rest	30	Resting cabinet
				Proofing	80	Proofing chamber
				Baking	45	Oven
				Cooling	36	Cooling space
				Packaging	7	Employee

Table 1. Simplified production data for one product group.

In another scenario, multiple products that came across different processing routes are baked in the same oven. Because baking consumes high energy, running an oven while it is only partially occupied causes energy waste. Figure 3 shows a schedule for two products where baking is performed together. In both cases (Figs. 2, 3), the products are internally dependent such that their common tasks must be performed at the same time. This preceding rule is used to arrange products into groups in the flow shop model. Only one product in a group has no predecessor, which means it can be scheduled at any time throughout the production runtime. However, the schedules for the rest of the products in that group depend on it. Table 1 represents simplified production data for one product group which is visualized in Fig. 2.

In reality, many product groups are organized based on their internal dependence. Within a group, each product has an individual bowl time. It indicates the start time difference between a predecessor product and any other product in a group. For one processing stage, there might be multiple alternative machines and employees, from which one should be selected based on availability. In general, for dough rest and cooling, no energy-consuming machines are required and therefore are considered to have the capacity to operate as many products as possible at a time. Similarly, due to having enough space in proofing chambers, it is assumed that the proofing stage has no blockage. Considering the bakers' practice, ovens can operate multiple products from the same product group at a time (Fig. 3). The rest of the machines can perform a task only from one product.

Figure 4 shows the procedure of optimizing the bakery production schedule. It can be discussed in three distinct segments: data collection, HNFSM, and optimization algorithm. Information about bakery products, machines, and employees is recorded during data collection. Depending on the internal dependence the products are sorted into distinct groups. An initial product sequence, an order of product groups in which they are produced, is transferred to HNFSM. In HNFSM, the processing tasks are allocated among the machines and employees. Here, the actual scheduling, i.e., exact start and end time, machine, or employee to conduct a task is determined. Like other flow shop models, the products that are placed first in the order will get priority in occupying the machines and employees. The following products are scheduled according to machine availability. The makespan and OIDT are calculated as a quality indicator for a schedule, which is considered as a baseline to start improving. The optimization algorithm proposes a new candidate solution vector, which requires to convert into a product group sequence to use for HNFSM. The candidate solution conversion approach is explained later. This procedure is repeated until a certain termination criterion is met.

This section describes the mathematical formulation of the proposed hybrid no-wait flow shop scheduling model. Table 2 defines the notions that are used to describe HNFSM. Ovens at bakeries typically contain multiple compartments that may each be used separately to bake different products. OIDT may exceed the makespan since it calculates the total idle time of all oven compartments. The following equations are used to calculate makespan and OIDT.

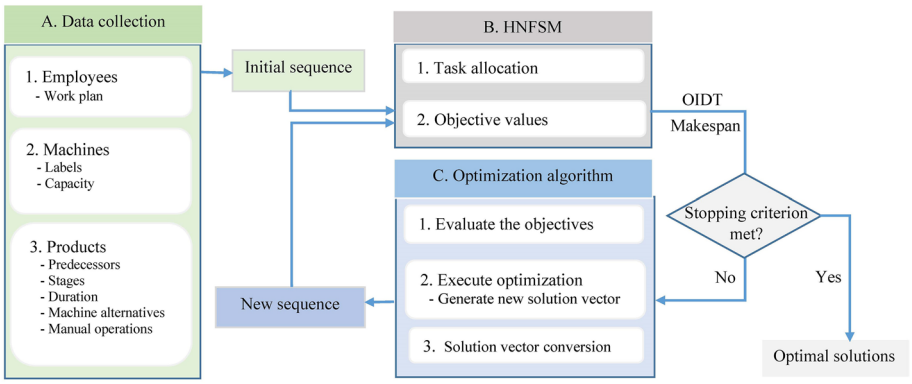


Figure 4. Schematic diagram of bakery production optimization using hybrid no-wait flow shop scheduling model (HNFSM).

Notation	Description
NG	Number of groups of products
n	Number of products in a group
m	Number of machines
e	Number of employees
g	Index of groups;$g = 1, 2, \ldots, NG$
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 minutes
PG	Set of product groups;$G = \{1, 2, \ldots, NG\}$
P	Set of products in a 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;$E = \{1, 2, \ldots, e\}$
$WT_{g,p}$	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 processed by 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
$O_{g,p,s,k}$	$\begin{cases} 1, & \text{if the product } p \text{ in group } g \text{ is processed on machine } k \text{ at stage } s \\ 0, & \text{if otherwise} \end{cases}$
$O_{g,p,s,l}$	$\begin{cases} 1, & \text{if the product } p \text{ in group } g \text{ is processed by machine } l \text{ at stage } s \\ 0, & \text{if otherwise} \end{cases}$

Table 2. Notations used in hybrid no-wait flow shop scheduling model (HNFSM).

$$Makespan = max\left(CT_{g,p,s}\right) \quad \forall g \in PG, \forall p \in P \tag{M1}$$

$$OIDT = \sum_{k=1}^{m} \left(end_k - start_k - \sum_{g=1}^{NG} \sum_{p=1}^{n} PT_{g,p,k} \right) \quad \forall g \in PG, \forall p \in P, \forall k \in V \tag{M2}$$

The HNFSM is described as follows.

$$Min(Makespan) \tag{M3}$$

$$Min(OIDT) \tag{M4}$$

Subject to

$$ST_{g,1,1} \geq 0 \quad \forall g \in PG \tag{M5}$$

$$ST_{g,p,1} = ST_{g,1,1} + WT_{g,p} \quad \forall g \in PG, \forall p \in P \backslash \{1\} \tag{M6}$$

$$PT_{g,p,s} > 0 \quad \forall g \in PG, \forall p \in P, \forall k \in M \tag{M7}$$

$$CT_{g,p,s} = ST_{g,p,s} + PT_{g,p,s} \quad \forall g \in PG, \forall p \in P \tag{M8}$$

$$ST_{g,p,s+1} = CT_{g,p,s} \quad \forall g \in PG, \forall p \in P \tag{M9}$$

$$\sum_{k=1}^{m} O_{g,p,s,k} \leq 1 \quad \forall g \in PG, \forall p \in P, \forall k \in M \tag{M10}$$

$$\sum_{g=1}^{NG} \sum_{p=1}^{n} \sum_{t=ST_{g,p,s}}^{CT_{g,p,s}} O_{g,p,s,k} \leq \left(CT_{g,p,s} - ST_{g,p,s}\right) \quad \forall g \in PG, \forall p \in P, \forall k \in M \backslash (U \cup V) \tag{M11}$$

$$\sum_{g=1}^{NG} \sum_{p=1}^{n} \sum_{t=ST_{g,p,s}}^{CT_{g,p,s}} O_{g,p,s,k} \leq n\left(CT_{g,p,s} - ST_{g,p,s}\right) \quad \forall g \in PG, \forall p \in P, \forall k \in V \tag{M12}$$

$$\sum_{g=1}^{NG} \sum_{p=1}^{n} \sum_{t=ST_{g,p,s}}^{CT_{g,p,s}} O_{g,p,s,k} \leq \sum_{g=1}^{N} n \quad \forall g \in PG, \forall p \in P, \forall k \in U \tag{M13}$$

$$start_l < end_l \quad \forall l \in E \tag{M14}$$

$$\sum_{l=1}^{e} O_{g,p,s,l} \leq 1 \quad \forall g \in PG, \forall p \in P, \forall l \in E \tag{M15}$$

$$\sum_{g=1}^{NG} \sum_{p=1}^{n} \sum_{t=ST_{g,p,s}}^{CT_{g,p,s}} O_{g,p,s,l} \leq \left(CT_{g,p,s} - ST_{g,p,s}\right) \quad \forall g \in PG, \forall p \in P, \forall l \in E \tag{M16}$$

$$ST_{g,p,s} \geq start_l \quad \forall O_{g,p,s,l} = 1, \forall g \in PG, \forall p \in P, \forall l \in E \tag{M17}$$

$$CT_{g,p,s} \leq end_l \quad \forall O_{g,p,s,l} = 1, \forall g \in PG, \forall p \in P, \forall l \in E \tag{M18}$$

$$\sum_{k=1}^{m} O_{g,p,s,k} + \sum_{l=1}^{e} O_{g,p,s,l} \leq 1 \quad \forall g \in PG, \forall p \in P, \forall k \in M, \forall l \in E \tag{M19}$$

The objective functions are shown in Eqs. (M3), (M4). Constraint (M5) states that the start time for the predecessor product of any group can be ≥ 0. Constraint (M6) defines it for successor products in the group. It includes a time difference between the start time of predecessor and successor products. Constraint (M7) declares that the processing time for any stage must be greater than 0 min. The no-wait condition between two consecutive stages of a product is defined by conditions (M8) and (M9). Constraint (M10) ensures that an operation from a product can occupy only one machine. A machine can only perform one task at a time except for the ovens ($k \in V$) and the machines with unlimited capacity ($k \in U$), as defined by constraint (M11). Constraint (M12) allows ovens to bake multiple products from the same group. According to constraint (M13), machines with unlimited capacity can perform any number of tasks at a time. Condition (M14) validates the shift plan of employees. Constraint (M15) limits the number of employees assigned to a single task. Constraint (M16) restricts the number of tasks assigned to a single employee at any given time. Employee job allocation is limited by constraints (M17) and (M18) to be within their working hours. Condition (M19) states that either an employee or a machine limitation can be occupied for a task. However, in the bakery process, some tasks require no machine and employee, such as dough rest.

Multi-objective optimization. Most real-world optimization problems that scientists and engineers handle routinely are multi-objective problems, where systems demand satisfying more than one parameter. Conventionally, such problems are simplified in two diverse ways: after converting multiple objectives into one by using the linear weighting method and featuring objectives as constraints. These approaches provide an optimized solution to a satisfactory level without handling the complex interrelations between multi-objectives.

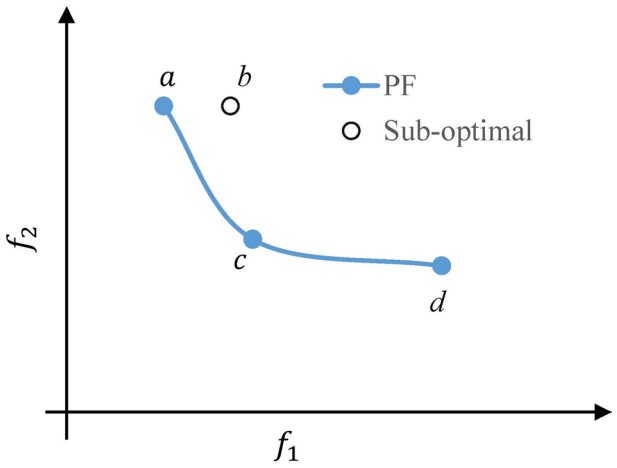


Figure 5. Concept of Pareto dominance for bi-objective functions (f_1 and f_2) optimization minimization problem.

Nonetheless, depending on the type of problems, these approaches have limitations. The former method relies on personal preference when determining the importance of objectives, which has a major impact on the solution. Furthermore, weighting factors might lead the optimizer to a poor solution when solving a problem with a non-convex Pareto front that is unknown beforehand. The latter approach struggles to deal with the high-dimensional, multi-objective optimization problems and is prone to producing suboptimal solutions. In reality, many multi-objective optimization problems do not show continuous solutions in the objective space. Therefore, if objectives are restricted to different ranges, for an optimizer it is challenging to find an optimal solution that meets these constraints. There are many optimal solutions to multi-objective problems with many local minima in a multi-objective space. Following traditional methods, the entire procedure must be repeated many times, each time adjusting the weighting factors or constraints for the objectives to make sure that the obtained solution is not one of these local minima. However, there is no guarantee that a complete set of optimum solutions will be obtained. To address this problem, several multi-objective optimization algorithms have been proposed[27–33]. 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 minimization problem can be stated as follows (Eq. 1).

$$minf_1(x), \ldots, minf_q(x); x \in \chi \text{ and } q > 1 \tag{1}$$

where $f_1(x), \ldots, f_q(x)$ are objective functions such that minimizing one function leads to an increase in others.

Multi-objective optimization, unlike single-objective optimization, generates a collection of optimal solutions by displaying tradeoffs between the objectives in the objective space[24]. Therefore, an objective vector has q values, such as $[f_1(x), ..., f_q(x)]$, each of which reflects the extent of the corresponding objective. Figure 5 outlines an objective space for a bi-objective ($q = 2$) problem. The multiple optimal solutions in this space are selected such that they show the best tradeoffs between the objectives, which is defined by Pareto dominance. Pareto dominance is the fundamental of multi-objective optimization algorithms, extensively used to distinguish optimal solutions from suboptimal solutions. To define the Pareto dominance, given two objective vectors, $\vec{a} = [a_1, ..., a_q]$ and $\vec{b} = [b_1, ..., b_q]$ and $\vec{a}$ is said to dominate $\vec{b}$ ($\vec{a} \preccurlyeq \vec{b}$) if and only if $\vec{a}_d \leq \vec{b}_d$ for every $d \in \{1, \ldots, q\}$ and $\vec{a}_d < \vec{b}_d$ for at least one of $d \in \{1, \ldots, q\}$. In words, $\vec{a}$ dominates $\vec{b}$, if $\vec{a}$ is not worse in any objective and better in at least one objective than $\vec{b}$[34,35]. Figure 5 shows that the objective vector $\vec{a}$ dominates $\vec{b}$ as it improves f_1 while not worsening f_2. However, considering $\vec{a}$, $\vec{c}$ and $\vec{d}$, no one dominates none and thus together they form the Pareto front (PF).

Optimization algorithms. The concept of Pareto dominance has been used fundamentally in these multi-objective optimization algorithms to find a collection of optimal solutions from a population that progresses over the generations. The strength Pareto evolutionary algorithm (SPEA) is one that was later improved to SPEA2 by eliminating a few weaknesses[32]. Similarly, the non-dominated sorting genetic algorithm (NSGA) was improvised for NSGA-II by reducing the computation complexity using a fast non-dominated sorting approach[27]. The inclusion of elitism, a feature that preserves the good solutions over generations, in NSGA-II makes it comparable with SPEA2. The third version of generalized differential evolution (GDE3), which originated from the differential evolution algorithm, is relatively a new member of this group[36].

Based on the simulation of the social behavior of birds, the particle swarm optimization algorithm was first proposed by Eberhart and Kennedy[37]**.** This concept has been used in several studies to develop multi-objective optimization algorithms[34,34,38–45]. In an improved particle swarm optimization-based algorithm, known as OMOPSO, Pareto dominance, crowding distance, and mutation operators are included, resulting in highly competitive performance[44]. Later, an extended version, speed-constrained multi-objective particle swarm

Generation (G)	Sequence vector ($\vec{x}_{i,G}$)	Sorted index	Product group sequence
1	[1.22, − 1.08, 1.90]	[1–3]	{2, 1, 3}
2	[− 0.29, 0.25, 0.80]	[1–3]	{1, 2, 3}
3	[1.06, 0.30, -0.20]	[1–3]	{3, 2, 1}

Table 3. Conversion of a sequence vector to a product group sequence using smallest position value rule.

optimization (SMPSO), was introduced. It is reportedly aimed to adapt particle velocity when it gets higher to generate an effective position in the search space[42].

However, there is no guarantee that all MOA solutions are truly optimal for an unknown problem. In most previous studies, the effectiveness of multi-objective optimization methods has been demonstrated by solving different mathematical test functions. Solving real-world high-dimensional production scheduling problems are computationally expensive and rarely used as benchmarks to test algorithms. A few studies applied these state-of-art multi-objective algorithms to solve scheduling problems[46–51]. In Annex A, the algorithms are briefly described.

Solution vector conversion. The optimization candidate solution is a vector consisting of continuous values of size equal to the number of product groups in a problem. In contrast, a solution to HNFSM is a set of discrete numbers, where each number represents one product group. The order of these discrete numbers makes the difference in the final schedule as it implies when each product should be produced. Therefore, a conversion of the algorithmic solution is required. To convert the solution vector into a set of discrete numbers, the smallest position value rule is employed. Table 3 explains the candidate solution conversion procedure for 3 product groups using the smallest position value rule. In this approach, the index of each value in the solution vector is conjugated with a product group. The indexes are sorted by the rule of smallest to the largest value in the vector. The sorted index is used as a solution to the problem. The order of the numbers in the product sequence specifies the order of conjugated product groups in which they should be produced.

Performance indicators. Ye et al.[52,53] have described the difficulty in achieving effectiveness and efficiency while finding optimized solution to no-wait flow shop scheduling problems. In this study, the performance of an algorithm is evaluated based on obtained Pareto front, called candidate PF, with the true Pareto front (PF*) to a problem. Initially, PF* for a problem is unknown. Once the candidate fronts (PF) for a problem are obtained by algorithms, the PF* is calculated by taking only Pareto optimal solutions from them.

Cardinality. To measure the cardinal quality of optimal solutions, Pareto domination strength is used, which considers the number of optimal and non-optimal solutions in a candidate PF obtained by any algorithm. Pareto domination strength was calculated by using Eq. (2). A higher Pareto domination strength indicates the worst performance of an algorithm.

$$PDS = \frac{|\{a : a \in PF \text{ and } a \notin PF^*\}| - |\{a : a \in PF \text{ and } a \in PF^*\}|}{|PF| \times |PF^*|} \tag{2}$$

where PDS is Pareto domination strength, $|.|$ indicates cardinality of a set, a is an objective vector.

Distribution and spread. The maximum spread of the solutions in a front captures the spread of the solutions in a front using Eq. (3). A higher value for this indicator represents that an algorithm performed better.

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

where MSF is maximum spread of the solutions in a front, Q is the number of objectives, f_q^{max} and f_q^{min} are the maximum and minimum values of the qth objective in PF^*, respectively, F_q^{max} and F_q^{min} are the maximum and minimum values of the qth objective in the PF provided by the algorithm that is under evaluation.

Convergence. Convergence measures the degree of proximity between PF* and its approximation, e.g., candidate PF obtained by an algorithm. As a convergence indicator, distance to the Pareto front represents how close the solutions of two fronts are. A higher distance to the Pareto front, calculated by using Eq. (4), indicates an algorithm performed worst.

$$DPF = \frac{1}{|PF^*|} \left(\sum_{a \in PF^*} \min_{b \in PF} Ed(a, b) \right) \tag{4}$$

where DPF is distance to the Pareto front, $Ed(a, b)$ is the Euclidean distance between objective vectors a and b, PF^* is the true Pareto front, and PF is the front obtained by an algorithm that is under evaluation.

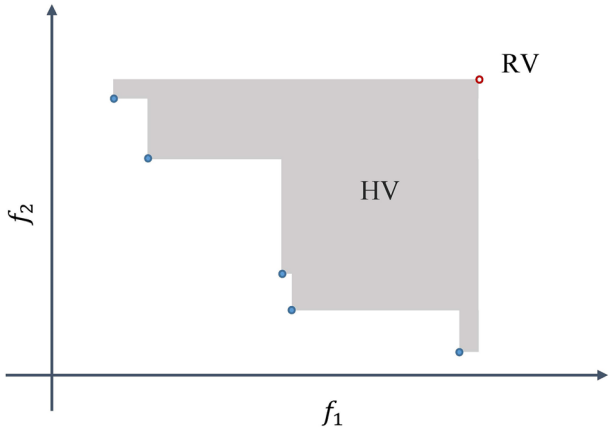


Figure 6. Hypervolume (HV) of one front for two objective functions (f_1 and f_2). Filled circles are solutions in front and empty circle is a reference objective vector (RV).

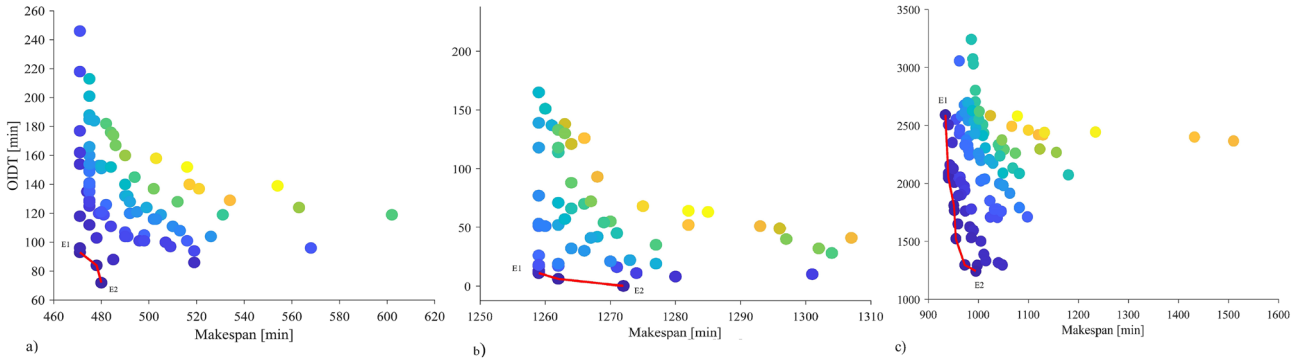


Figure 7. Candidate solutions represented by their quality indicators (circles) obtained by five optimization algorithms and true Pareto front (line) for problem (**a**) BK15, (**b**) BK40 and (**c**) BK50. The color gradient of the circles represents the quality of the solutions, with the darkest blue being the best and pale yellow being the worst.

Convergence and distribution. The hypervolume of the front in objective space describes the convergence and distribution of the solutions obtained by an algorithm. It calculates the volume of the space covered by the solutions of a front and delimited from above by a reference objective vector. It defines the upper limit for each objective in the objective space to consider for calculating hypervolume (Fig. 6) by using Eq. (5). A higher relative hyper volume indicates that an algorithm performed better.

$$HP_{PF} = \lambda_q\left(\bigcup_{a\in PF}[a, RV]\right) \tag{5}$$

where λ_q is the q-dimensional Lebesgue measure, PF is a front obtained by an algorithm that is under test, a is an objective vector and RV is a reference objective vector. In this study, for the two objectives, relative hyper area (Eq. 6) is used to compare the performance of different algorithms.

$$RHA[\%] = \frac{HP_{PF} \times 100}{HP_{PF^*}} \tag{6}$$

where RHA is relative hyper area, HP_{PF} and HP_{PF^*} are hyper volumes for a front obtained by an algorithm that is under evaluation and true Pareto front for a problem, respectively.

Results and discussion

True Pareto front and candidate solutions. Figure 7 shows quality indicators representing candidate solutions in objective space obtained by algorithms. True Pareto front (PF*) is the line that connects only a set of optimal solutions. PF* for the instances has only a few solutions, even though many sub-optimal solutions exist nearby. BK15 (Fig. 7a) and BK40 (Fig. 7b) have three Pareto solutions each, but BK50 has seven (Fig. 7c). For BK15 and BK40, the difference between boundary solutions (E1 and E2), which represent extreme tradeoffs between objectives inside PF*, is insignificant. Boundary solutions for BK15 reveal tradeoffs within a 9 min difference in makespan and 21 min difference in OIDT, while they are 13 min and 11 min differences for BK40,

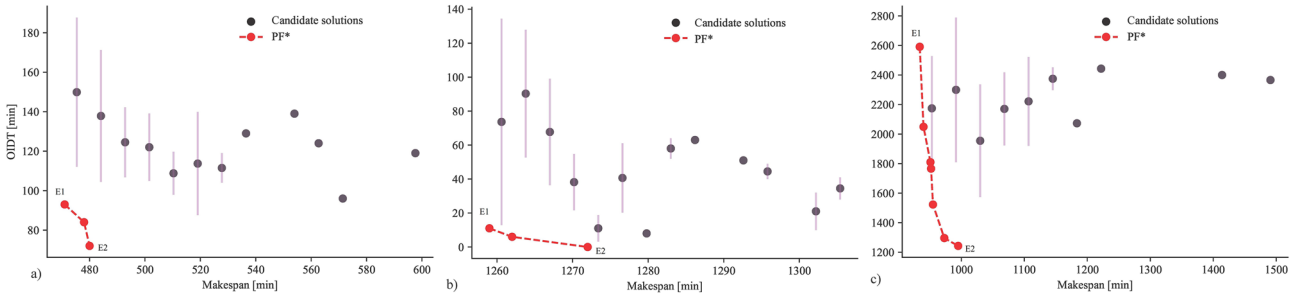


Figure 8. Binned scatter plot of quality indicators representing candidate solutions obtained by five algorithms for instances: (**a**) BK15, (**b**) BK40 and (**c**) BK50. Error bars represent the standard deviation of OIDT.

respectively. In contrast, due to a 61 min rise in makespan, PF* for BK50 conveys a large 1348 min decline in OIDT. Unlike many mathematical function optimizations described in previous studies[31,36,42], the solutions in PF* are not continuous in objective space. Furthermore, PF* may have a convex or concave form, which adds to the complexity of solving higher-dimensional problems for optimizers.

Despite a dramatic drop in OIDT over the PF*, BK50 has the highest OIDT (1243 min) at E2. The reason could be that many products in BK50 have predecessors. If multiple products within a group require different specific ovens and the number of products in the group is higher, it is most likely to have higher oven idle time. Candidate solutions that are more densely dispersed in the higher OIDT area support it, with only a few solutions found around E2 (Fig. 7c). In contrast, BK15 has only 3 groups with more than one product and BK40 has no group with multiple products. The PF* of these cases shows a minimum OIDT of 72 min and 0 min, respectively at the E2 point. In addition, their candidate solutions are distributed throughout the objective axis (Fig. 7a,b). BK50 has many groups where the initial stages are processed combinedly to take advantage of machine capacity and save preparation time. Since these products eventually require different baking ovens, finding these ovens available at different time spans leads to a higher OIDT.

Multiple solutions were achieved at the shortest makespan (E1) in all cases, but they were dispersed unevenly, with a few having significantly larger OIDT than the respective Pareto solution (E1). The binned scatter plot (Fig. 8) gives two indications. Firstly, a schedule with minimum makespan does not guarantee to have minimum OIDT. A similar result was observed in previous studies[5,23]. Therefore, a schedule optimized with a goal to minimize the makespan might be highly inefficient in energy usage. For example, the candidate solutions for BK15 above E1 shows up to 150 min higher OIDT compared to E1 despite having the same makespan, which is even higher for BK40 (231 min). Figure 8 demonstrates that the error bars are pronounced in the region of the lower makespan. It implies that there is a high possibility an optimizer will produce poor solutions around the shortest makespan with a higher energy waste due to OIDT.

Secondly, solutions with a marginal increase in the shortest makespan could result in an acute reduction in OIDT. As a result, a substantial amount of energy can be saved, lowering operational expenses and CO_2 emissions. Because the makespan dominates manufacturing cost, the gain in OIDT is compared to the loss in makespan from the shortest makespan at E1. If any Pareto solution other than E1 offers an intense reduction in OIDT while losing a marginal amount of makespan, the entire manufacturing cost can be reduced even more. For example, for BK50, E2 offers OIDT drop by 8% for each percentage increase in makespan from E1 (Fig. 8c). In other words, E2 is more efficient than E1 since it lowers OIDT by 1348 min while increasing makespan by only 61 min.

Candidate Pareto front. Figure 9 shows the candidate PF for BK15 attained by algorithms. For 50 and 100 iterations, NSGA-II and SPEA2 showed better performance. However, with increasing the iteration size, OMOPSO and GDE3 obtained improved solutions too. In contrast, despite offering a high number of solutions, SMPSO displayed comparatively poor performance. A similar performance was observed for BK50 (not shown). In contrast, for BK40 (not shown), the NSGA-II performed worst compared to OMOPSO and SMPSO. SPEA2 always found only one solution, though it was close to being the optimal solution.

Figure 10 shows the improvement of candidate fronts obtained by algorithms over different iteration sizes for BK50. NSGA-II, SPEA2, and GDE3 improved solution quality remarkably over different iteration sizes (Fig. 10a–c, respectively). However, fronts from OMOPSO and SMPSO were similar, and both displayed poor improvement. All the solutions in PF* for BK50 were obtained by NSGA-II and SPEA2 combinedly, while GDE3 featured a few solutions near PF*. In contrast, no contribution in PF* was observed from MOPSO and SMPSO. Comparable results were obtained for BK15 (not shown). For BK40, GDE3, OMOPSO, and SMPSO obtained Pareto solutions to form PF* and no contribution from NSGA-II and SPEA2 was observed (not shown).

Hecker et al.[6] used single objective optimization methods to reduce the makespan of BK40, and the results showed that a modified genetic algorithm obtained a minimum makespan of 1261 min 4 times out of 21 runs. Solutions with the best makespan were compared even though multi-objective solutions were found in this study. The best makespan of 1259 min was attained by NSGA-II and SMPSO four times out of four separate runs with varying iteration sizes, while SPEA2, GDE3, and OMOPSO achieved this three times each.

Pareto domination strength and maximum spread of the solutions. Table 4 shows the pareto domination strength and maximum spread of the solution for the algorithms. According to Pareto domination

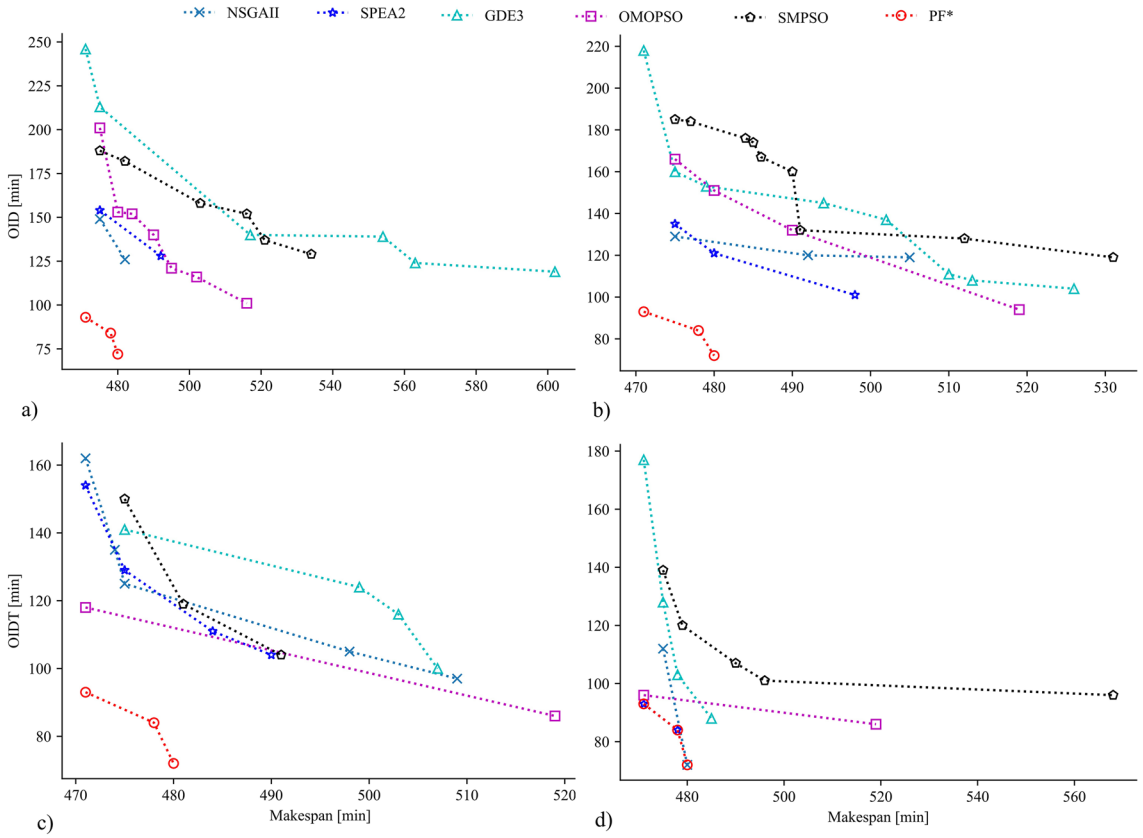


Figure 9. Candidate Pareto front (PF) for BK15 from optimization algorithms with—(**a**) 50 iterations, (**b**) 100 iterations, (**c**) 200 iterations and (**d**) 300 iterations.

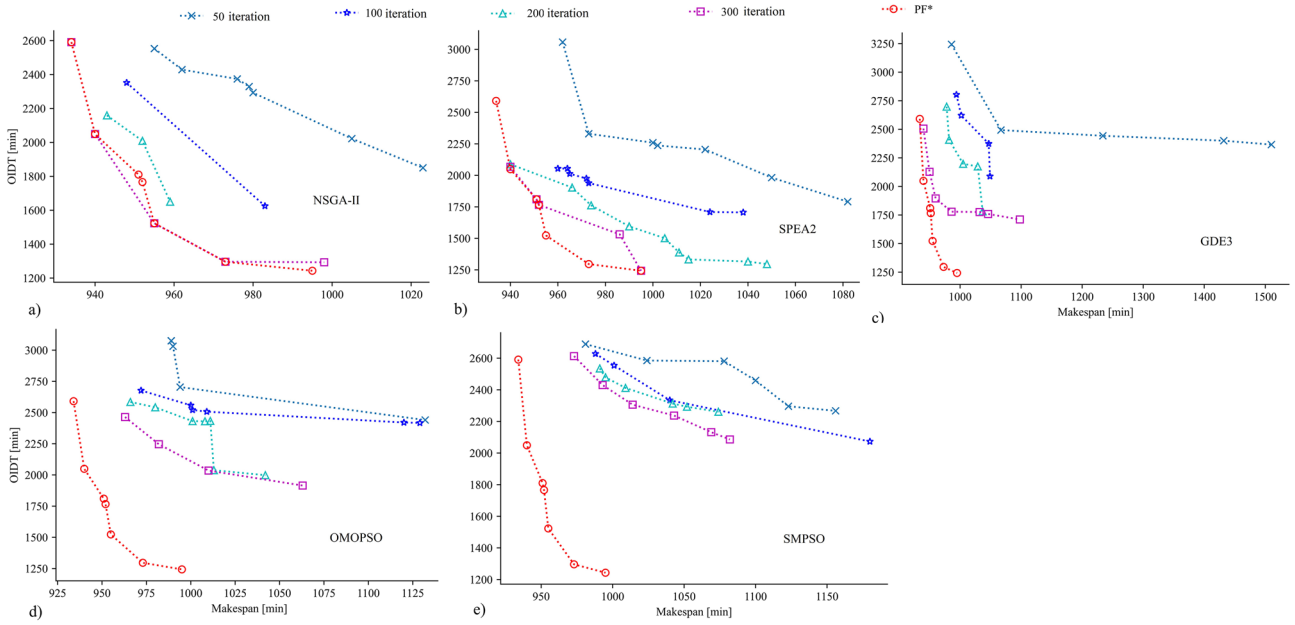


Figure 10. Improvement of candidate Pareto fronts (PF) for BK50 over different iteration size obtained by (**a**) NSGA-II, (**b**) SPEA2, (**c**) GDE3, (**d**) OMOPSO and (**e**) SMPSO. A label of 50 iteration indicates the front is obtained by and algorithm with 50 iterations.

Problem	Algorithm	Pareto domination strength				Maximum spread of solutions			
		Iteration				Iteration			
		50	100	200	300	50	100	200	300
BK15	NSGA-II	0.33	0.33	0.20	0.00	1.13	1.84	0.17	0.80
	SPEA2	0.33	0.11	0.17	−0.33	0.97	0.23	0.37	1.00
	GDE3	0.22	0.25	0.33	0.00	0.15	0.13	0.16	0.46
	OMOPSO	0.24	0.17	0.00	0.00	0.10	0.08	0.20	0.51
	SMPSO	0.33	0.33	0.33	0.33	0.44	0.29	0.28	0.06
BK40	NSGA-II	0.17	0.22	0.11	0.11	0.81	0.25	0.44	0.71
	SPEA2	−0.33	−0.33	−0.33	−0.33	0.00	0.00	0.00	0.00
	GDE3	−0.33	−0.11	−0.33	−0.33	0.00	0.61	0.61	0.83
	OMOPSO	0.20	0.14	0.17	0.00	0.26	0.66	0.72	0.42
	SMPSO	0.17	−0.11	0.00	−0.33	0.32	0.76	0.96	1.00
BK50	NSGA-II	0.10	0.14	0.05	−0.09	0.82	1.00	1.00	0.98
	SPEA2	0.10	0.10	0.08	−0.09	0.49	0.78	0.79	1.00
	GDE3	0.14	0.14	0.14	0.10	0.18	0.50	0.66	0.75
	OMOPSO	0.14	0.14	0.14	0.14	0.17	0.48	0.76	0.74
	SMPSO	0.14	0.14	0.10	0.10	0.55	0.66	0.71	0.69

Table 4. Calculated Pareto domination strength and maximum spread of solutions in front for algorithms.

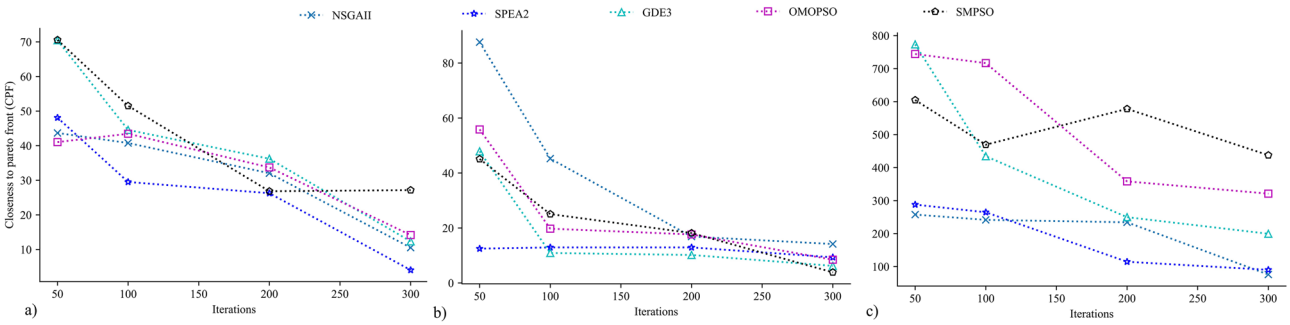


Figure 11. Distance to Pareto front of candidate fronts obtained by algorithms for (**a**) BK15, (**b**) BK40 and (**c**) BK50.

strength, SPEA2, GDE3, and OMOPSO were observed to perform better for solving BK15, with the worst being SMPSO. For BK40, SPEA2 and GDE3 improved solution quality consistently over different iteration sizes. In contrast, NSGA-II performed worst. The Pareto domination strength of NSGA-II and SPEA2 to solve BK50 was better and found no significant difference between them. Similarly, the difference between the Pareto domination strength of GDE3, OMOPSO, and SMPSO is minor and performed worst. According to this performance metric, only SPEA2 showed better performance in all instances.

The maximum spread of the solutions in front measures the distribution and spread of candidate solutions over the PF offered by an algorithm, with a greater number indicating better performance. NSGA-II had higher maximum spread of the solutions in all instances and was found to outperform all other algorithms in this performance metric. The maximum spread of the solutions of GDE3 was equivalent to NSGA-II in most circumstances in terms of problems and iteration sizes. But it had the lowest value for BK40 where only one solution was obtained every time. In contrast, GDE3 had better maximum spread of the solutions for BK40. In most scenarios, GDE3 and OMOPSO was remarkably comparable to each other. With the increasing iteration size to solve BK15, the maximum spread of the solutions of SMPSO decreased. It means that with a short iteration size, it was able to find solutions that had better distribution, but they were mostly suboptimal. In contrast, a large iteration size obtained comparatively better solutions, however, their dispersion was poor. For BK40 and BK50, SMPSO had a modest maximum spread of the solutions.

Distance to Pareto front. The distance to Pareto front for the algorithms is shown in Fig. 11. This performance metric represents how close a candidate's front to PF* is, where a low value indicates better performance. For BK15, SPEA2 exhibited promising improvement over different iteration sizes, while SMPSO was observed to perform worst (Fig. 11a). There is no substantial difference between the distance to Pareto front of NSGA-II, GDE3, and OMOPSO. For BK40, the PF obtained by SPEA2 had the lowest value with minimum

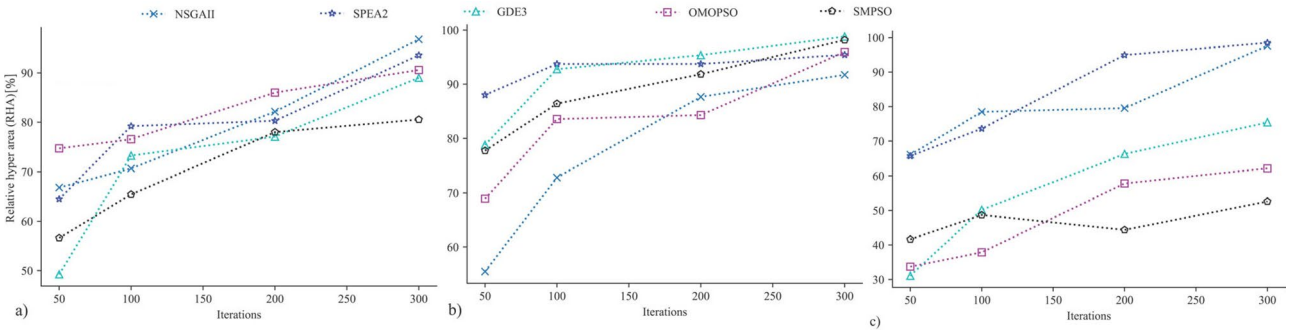


Figure 12. Relative hyper area (RHA) of candidate Pareto fronts (PF) obtained by algorithms for (**a**) BK15, (**b**) BK40 and (**c**) BK50.

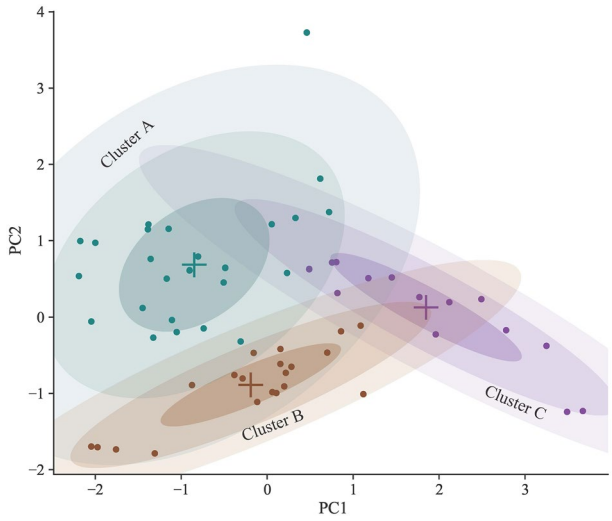


Figure 13. Clusters of algorithms' solutions found by Gaussian mixture model.

iteration (Fig. 11b). However, with increasing iteration sizes, the distance to Pareto front of GDE3, OMOPSO, and SMPSO was comparable to that of SPEA2. According to this performance indicator, NSGA-II had the worst distance to Pareto front with 50 iterations, which sharply improved with increased iteration sizes, yet could not outperform any algorithms. Figure 11c shows the distance to Pareto front of algorithms for BK50, where NSGA-II and SPEA2 outperformed GDE3, OMOPSO, and SMPSO. The values of BK50 distinguished algorithms' performance at every iteration which was not prominent for BK15 (Fig. 11a) and BK40 (Fig. 11b).

Relative hyper area. Figure 12 presents the relative hyper area of algorithms. It measures convergence and distribution of algorithms with a higher value indicating better performance. NSGA-II had higher value for BK15 and BK50, while for BK40 it performed worst, and SPEA2 had higher relative hyper area for all the cases. GDE3 and OMOPSO had moderate relative hyper area for all instances. In contrast, SMPSO showed the lowest relative hyper area for BK15 and BK50, and higher for BK40.

The distance to Pareto front and the relative hyper area for BK15 and BK40 illustrate the significant improvement in solutions' quality over different iteration sizes for algorithms (Figs. 11, 12). With the higher iteration size, the performance difference between algorithms was found to be minimum. Only NSGA-II, SPEA2, and GDE3 were able to follow this trend in BK50, while OMOPSO and SMPSO fell behind. One reason could be that there are more local minima in the solution space of BK50 compared to that of BK15 and BK40. Many suboptimal solutions exist for BK50 with higher OIDT with a small difference in makespan (Fig. 8c). Additionally, BK50 has a higher dimension—maximum product groups—to optimize.

Performance evaluation of algorithms. Performance metrics explain a specific feature of solution quality. The solutions, however, can be categorized into different quality levels using a clustering approach. The frequency with which an algorithm produces good or poor-quality solutions is a measure of its efficiency. Initially, the performance metrics for all instances are used to perform principal component analysis (PCA)[54]. Two principal components (PC1 and PC2) with higher variances were taken to perform a Gaussian mixture model for clustering[22]. Figure 13 shows three clusters. The clusters' solutions were identified using the associated labels, which refer to instances, algorithms, and iteration sizes such as BK15, NSGA-II, and 50, respectively. The quality

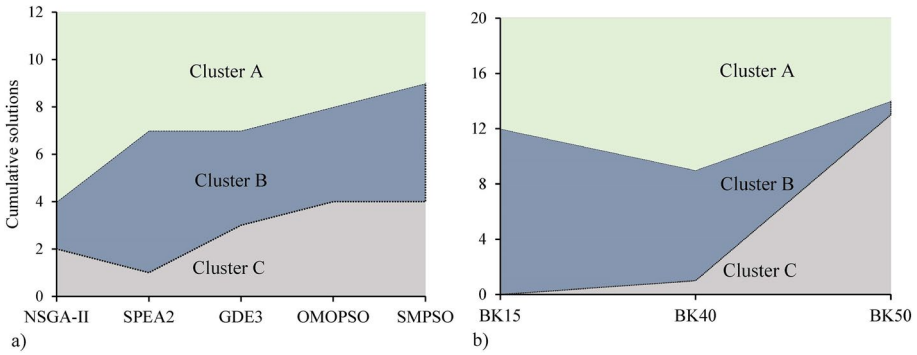


Figure 14. Distribution of obtained solutions among clusters with respect to: (**a**) algorithms (**b**) problems.

of different clusters were determined based on corresponding performance metrics where Cluster A represents better performances, and Cluster B and Cluster C show moderate and worst performances, respectively.

Figure 14 represents the distribution of obtained solutions into three clusters. Cluster A has 25 high-quality solutions, Cluster B has 21, and Cluster C has 14 low-quality solutions for the instances. NSGA-II had the highest number of solutions in Cluster A, followed by SPEA2 and GDE3 (Fig. 14a). In contrast, OMOPSO and SMPSO have the lowest number of solutions in this cluster. NSGA-II, with only four moderate and worst solutions, outperformed all other algorithms. SPEA2, with the highest number of moderate and lowest worst solutions, followed NSGA-II. In terms of distribution of obtained solutions among clusters, GDE3 performed slightly better than OMOPSO. In comparison to NSGA-II and SPEA2, SMPSO has the lowest solutions in Cluster A and the highest in Cluster C, indicating worse performance. For BK15, a large number of obtained solutions were moderate, with no worst solution (Fig. 14b). In contrast, for BK40, the majority of solutions fell into cluster A, emphasizing a problem that is comparatively easy to solve. BK50 revealed a considerable rise in the difficulty of obtaining moderate and better solutions, with Cluster C accounting for 65% of all solutions. Three of the six solutions in Cluster A were achieved by NSGA-II, two by SPEA2, and one by GDE3. In Cluster B, there is only one solution for BK50, which was obtained by SPEA2. In contrast, all the solutions from OMOPSO and SMPSO are in Cluster C. With only one Cluster C solution for BK50, NSGA-II and SPEA2 displayed consistently better performance. According to the cluster analysis, NSGA-II outperformed all other algorithms, followed by SPEA2. GDE3 performed better than OMOPSO and SMPSO, but OMOPSO and SMPSO showed no notable difference in performance.

The comparison of computation time of algorithms is performed with 50 iterations for the instances. Although OMOPSO needed the shortest calculation time (12 min, 58 min, 324 min for BK15, BK40 and BK50, respectively), the difference between it and other methods is insignificant. It took roughly the same amount of time for NSGA-II and SPEA2 in each case—13 min, 62 min, and 342 min, respectively. GDE3 showed slightly lower computation time with instances taking 12 min, 61 min, and 337 min, respectively. SMPSO, in contrast, had the longest computing time for every instance (13 min, 67 min, and 360 min, respectively). In comparison to OMOPSO, the extension to SMPSO appears to have triggered slightly high computing time as velocity constraints are applied to each iteration and dimension of the problem.

The current study shows that production planning using a flow shop model is feasible in practice when considering the actual resource limitations in bakeries. Along with makespan, minimizing the oven idle time also offers the potential to substantially lower manufacturing costs. To improve the current state of production efficiency in real cases from bakeries, multi-objective optimization algorithms were integrated with hybrid no-wait flow shop model. Among them, NSGA-II performed better in solving problems of various dimensions. Moreover, when multiple products share a predecessor, the increased oven idle time results in energy loss. Therefore, wherever possible, it is suggested to keep the processing route for a product separate from other products. Six bakery production datasets from Denmark were used by Babor et al.[55] to increase the production efficiency. The results revealed that NSGA-II performed efficiently to reduce makespan by up to 12% and oven idle time by up to 61%. Particle swarm optimization was used in a study[5] to obtain the best planning for a bakery's production in Spain. The optimum solution, according to the results, minimized the makespan by 29% and the oven idle time by 8%.

Conclusions and future works

In this paper, three production optimization problems from small and medium-sized bakeries were investigated. The objectives of optimization were to minimize simultaneously makespan and oven idle time (OIDT). A hybrid no-wait flow shop scheduling model with all constraints encountered in practice was implemented to simulate the bakery schedule. The optimum schedules were found using five multi-objective optimization algorithms: non-dominated sorting genetic algorithm (NSGA-II), strength Pareto evolutionary algorithm (SPEA2), generalized differential evolution (GDE3), improved multi-objective particle swarm optimization (OMOPSO), and speed-constrained multi-objective particle swarm optimization (SMPSO). To compare the efficiency of the algorithms, each problem was solved with different iteration sizes.

The computational results revealed that the shape of a true Pareto front is determined by the characteristics of the problems, such as the number of items, product interdependency, and alternative machinery. Although makespan has the most influence on production expenditure, it was observed that a substantial reduction in

OIDT is possible. Many solutions with the shortest makespan had higher OIDT (up to 231 min) that showed significant energy waste and CO_2 emissions. Therefore, with the same makespan, multi-objective algorithms can provide solutions with reduced energy waste. Furthermore, many Pareto solutions, aside from the one with the shortest makespan, provide better tradeoffs between makespan and OIDT. It means that by losing a very marginal amount in makespan, some solutions offer a substantial reduction in OIDT. BK50 showed an additional 1348 min of oven idle time can be reduced if the makespan is increased by 61 min. Therefore, the overall production expenditure can be significantly minimized. Product group formulation may influence OIDT. In the best-case scenario, for BK40 with no predecessor in any group, a schedule with 0 min OIDT is possible. However, because many products have a few combined initial processing stages, for BK50 the lowest possible OIDT is 1243 min, resulting in significant energy loss.

NSGA-II outperformed other algorithms by obtaining a smaller number of poor solutions and a high number of better solutions. SPEA2 followed NSGA-II by delivering promising solutions. GDE3 performed slightly better than OMOPSO and SMPSO. The performance of OMOPSO and SMPSO was poor to solve the instances and no significant difference between them was observed. However, OMOPSO had the lowest computation time while SMPSO had approximately 11% higher computation time due to the addition of velocity constraints.

The deterministic duration of the processing tasks and the absence of machine maintenance or failure assumed in this study may not reflect many realistic production problems. Based on prior relevant studies[56,57], the effects of non-deterministic processing duration and machine disturbances on the production efficiency of bakeries could be an interesting subject for future research.

Data availability

For this study, production data from bakeries in Europe were used. BK40 was collected and analyzed by Hecker et al.[6], whereas BK15 and BK50 are publicly accessible[25]. The production data are available from the corresponding author on reasonable request.

Appendix

Multi-objective optimization algorithms

Non-dominated sorting genetic algorithm (NSGA-II). The following is the description of the NSGA-II (Fig. 15) proposed by Deb et al.[27].

Create a random population of size *NP*. Each individual in the population is a candidate solution vector to a problem. Assess the individuals in the population using HNFSM by calculating objective values.

1. The objective values are used to build fitness vectors. The population is sorted into distinct rankings using the fast non-dominated sorting strategy. Using the Pareto dominance operator, each individual's fitness is compared to that of others. The rank of an individual is determined by the number of other individuals who dominate it, which is known as the sum of domination. If the sum of domination for an individual is 0, it is called a non-dominated solution or Pareto optimal solution. Individuals with a higher sum of domination have a suboptimal solution to the problem.
2. A binary tournament selection process is used to choose two parents from the existing population for creating two offspring. In the binary tournament selection process, four individuals from the current population are picked and the one with the best rank is chosen as one of the parents. The same procedure is followed to complete the parents' poll to perform crossover and mutation. It is repeated to create new offspring of *NP* size.
3. Evaluate offspring to get the corresponding fitness vectors. The offspring and parent population are combined. During this phase, the population doubles in size.
4. The combined population is sorted into different ranks using the fast non-dominated sorting approach.
5. To select the best population of *NP* size, the individuals with the best rank are chosen first. If the best rank does not have enough individuals to fill all the empty slots of the best population, the individuals of subsequent ranks are chosen. If a rank has more individuals than empty slots, the crowding distance operator is used to select individuals from the less crowded part of the objective space. The crowding distance is set to infinity for border solutions of a rank to give preference over others. All the solutions of the rank are sorted in descending order of crowding distance. Individuals with a higher crowding distance fill the empty slot first until the best population size reaches *n*.
6. Employ crossover and mutation operators to produce *NP* offspring from the best population.
7. Repeat steps 4–7 until the termination criterion is met.

A.2: Strength Pareto evolutionary algorithm (SPEA2). The process flowchart for SPEA2 is shown in Fig. 16. The brief of SPEA2[32] is as follows:

1. Generate an initial population of size *N* where each individual is a candidate solution vector for a problem. Create an empty external archive. Set the size limit for external archives to *EA*. In this archive, the best individuals are stored.
2. Evaluate the individuals to get fitness values. The fitness of the individuals in the population is calculated. Initially, a strength value for each individual is calculated using Eq. (A1).

$$S(i) = \left|\left\{j | j \in PO_G + AP_G \wedge i \succ j\right\}\right| \tag{A1}$$

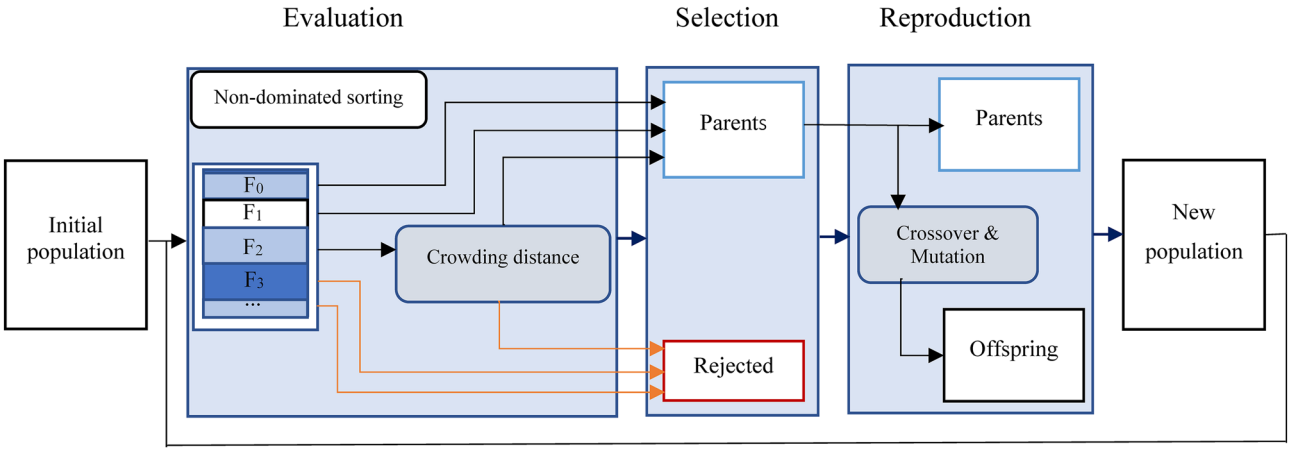


Figure 15. An evaluation procedure of the NSGA-II[55]. Non-dominated sorting divides the population into different ranks (F_0, F_1, F_2, …). Individuals from the same optimal front are maintained in one rank.

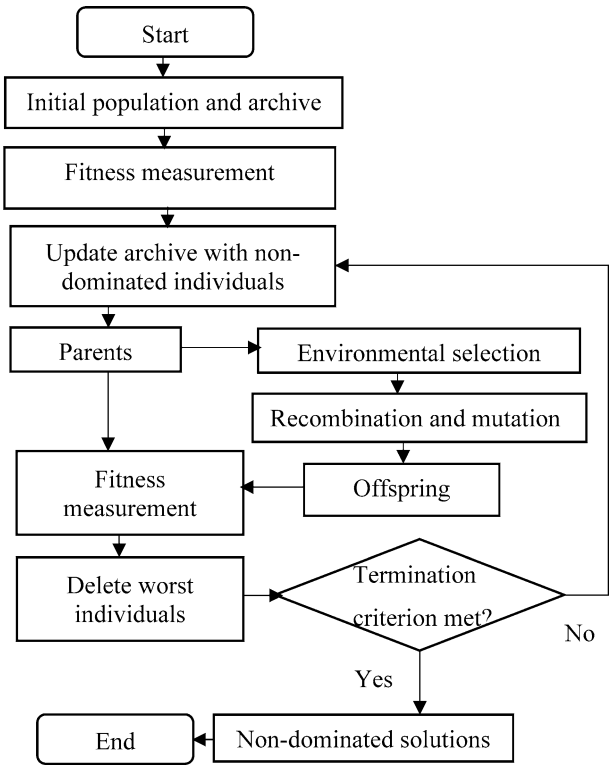


Figure 16. Flowchart of strength Pareto evolutionary algorithm (SPEA2).

where $|.|$ indicates the cardinality of a set, $+$ sign is for multiset union, $\succ$ symbolizes the Pareto dominance relation, PO_G is the population and AP_G is archive size at generation G. Therefore, $S(i)$ represents the number of other individuals (j) in the population and archive that are dominated by an individual i.

The raw fitness of individual i is defined by Eq. (A2).

$$R(i) = \sum_{j \in PO_G + AP_G, j \succ i} S(j) \tag{A2}$$

Equation (3) implies that the raw fitness of an individual is the sum of the strength of its dominators. In the next stage, a density estimation approach is employed. For that purpose, kth nearest neighbor method is adapted. The distances between i and all other individuals (j) in objective space are calculated. The distance list is sorted in increasing order. The kth element gives the distance sought σ_i^k. Equation (A3) defines the density calculation for an individual.

$$D(i) = \frac{1}{\sigma_i^k + 2} \tag{A3}$$

where $k = \sqrt{NP + EA}$, NP and EA are the size of the total population and archive, respectively.
Finally, the fitness of an individual i can be stated as follows:

$$F(i) = R(i) + D(i) \tag{A4}$$

3. Take all non-dominated individuals to archive. If the size of the archive exceeds the limit, reduce it by using a truncation operator that prevents boundary solutions from being removed. If the size is less than the limit, fill it with dominated individuals.
4. Employ binary tournament selection with replacement to obtain offspring. Initially, all individuals are compared based on Pareto dominance. Rank the individuals depending on their domination level. Afterward, estimate their density information within the corresponding rank. It represents the sum of distances between the two closest individuals along with each objective. Based on these two sorting approaches, the parent selection is performed. Apply recombination and mutation operators to keep high diversity among the population.
5. Combine offspring, parent population, and archived individuals. Delete the worst 50% of combined population based on their fitness values (as shown in step 2).
6. Continue with steps 3–5 until a stopping criterion is met.

A.3: Generalized differential evolution (GDE3). The differential evolution algorithm was first introduced by Storn and Price[58]. Like all other evolutionary algorithms, it has a random initial population, which is improved over the generations. It features cross-over, mutation, and selection operators to improve the solution. The selection rule is one of the key differences compared to other evolutionary algorithms. It is the process to decide whether a new individual should replace one from the population to generate efficient individuals for the next generation. The decision is taken based on some constraints that are regulated by crossover constant and differential variation between two individuals. Later, differential evolution algorithm was extended to generalized differential evolution (GDE) for multi-objective optimization problems by modifying the selection rule[59]. The optimization procedure of GDE3 is presented in Fig. 17. GDE3, an improved version, can be described as follows[36].

1. Initialize population of size NP ($NP \geq 4$), amplification constant for differential variation, $F \in (0, 1+]$, crossover constant, $CR \in [0, 1]$, dimensions or parameters of the problem, D, the maximum generation G_{max}, and a constant $A = 0$. The solution vector is $\overrightarrow{x}_{i,G}$ for an individual i at G generation where $i = \{1, 2, \ldots, NP\}$, and $G = \{1, 2, \ldots, G_{max}\}$. The value of d dimension in i individual at G generation is indicated by $x_{d,i,G}$, where $d = \{1, 2, \ldots, D\}$.
2. Mutate and recombine each individual in the population. For an individual i, chose three different individuals randomly R_1, R_2, R_3 where $R_1 \neq R_2 \neq R_3 \neq i$ and $R_1, R_2, R_3, i \in \{1, 2, \ldots, NP\}$. Chose random parameter d_{rand} where $d_{rand} \in \{1, 2, \ldots, D\}$. For each dimension ($d$), the following procedure is applied to $\overrightarrow{x}_{i,G}$ to get a new individual, which is known as a trial vector $\left(\overrightarrow{u}_{i,G}\right)$.

$$u_{d,i,G} = \begin{cases} x_{d,R_3,G} + F \times \left(x_{d,R_1,G} - x_{d,R_2,G}\right) & ifr < CRord = d_{rand} \\ x_{d,i,G} & otherwise \end{cases} \tag{A5}$$

where r is a random value between 0 and 1 and $d_{rand} \in \{1, 2, .., D\}$
3. Decide whether the trial vector should become a member of generation $G + 1$.

$$\vec{x}_{i,G+1} = \begin{cases} \vec{u}_{i,G} & if\ \overrightarrow{u}_{i,G} \preceq_c \vec{x}_{i,G} \\ \vec{x}_{i,G} & otherwise \end{cases} \tag{A6}$$

where the symbol $\preceq_c$ is constraint domination. To define it, $\overrightarrow{u}_{i,G}$ constraint dominates $\overrightarrow{x}_{i,G}$ if any of the following conditions is true[60]:

$\overrightarrow{u}_{i,G}$ is feasible and $\overrightarrow{x}_{i,G}$ is not.
$\overrightarrow{u}_{i,G}$ and $\overrightarrow{x}_{i,G}$ are infeasible and $\overrightarrow{u}_{i,G}$ dominates $\overrightarrow{x}_{i,G}$ in constraint function space.
$\overrightarrow{u}_{i,G}$ and $\overrightarrow{x}_{i,G}$ are feasible and $\overrightarrow{u}_{i,G}$ dominates $\overrightarrow{x}_{i,G}$ in objective space.
Set the following conditions:

$$\begin{matrix} A = A+1 \\ \vec{x}_{NP+A,G+1} = \vec{u}_{i,G} \end{matrix} \text{ if } \begin{cases} \forall d\colon C_d\left(\vec{u}_{i,G}\right) \leq 0 \\ \text{and} \\ \vec{x}_{i,G+1} == \vec{x}_{i,G} \\ \text{and} \\ \vec{x}_{i,G} \prec \vec{u}_{i,G} \end{cases} \tag{A7}$$

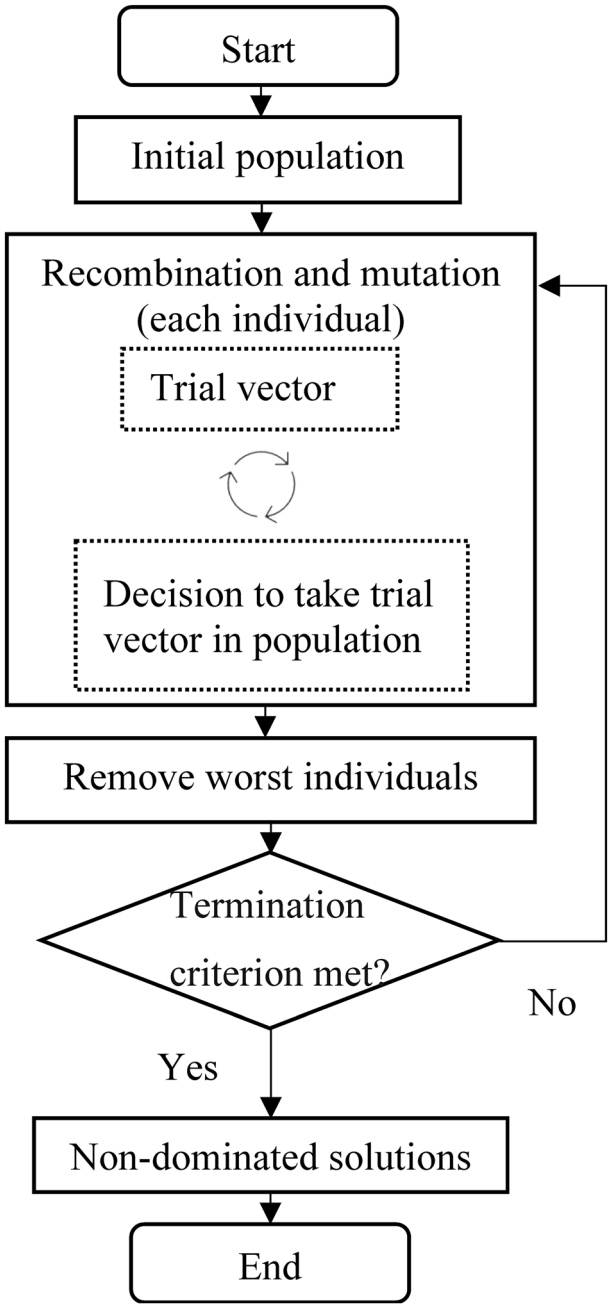


Figure 17. Optimization procedure for generalized differential evolution (GDE3).

where $C_d\left(\overrightarrow{u}_{i,G}\right)$ indicates constraint associated with dth dimension.

4. Repeat steps 2–3 *NP* times to complete mutation and recombination of the population of the generation *G*.
5. Select individuals $\left(\overrightarrow{x}\right)$ that meet the following condition:

$$\vec{x} \in \left\{\vec{x}_{1,G+1}, \vec{x}_{2,G+1}, \ldots, \vec{x}_{NP+A,G+1}\right\}: \begin{cases} \forall i \;\; \vec{x} \prec_c \vec{x}_{i,G+1} \\ \text{and} \\ \forall\left(\vec{x}_{i,G+1}: \vec{x}_{1,G+1} \prec_c \vec{x}\right) \end{cases} \quad CD(\vec{x}) \leq CD\left(\vec{x}_{i,G+1}\right) \tag{A8}$$

where *CD* is crowding distance that measures the crowdedness of a vector in a non-dominated set.

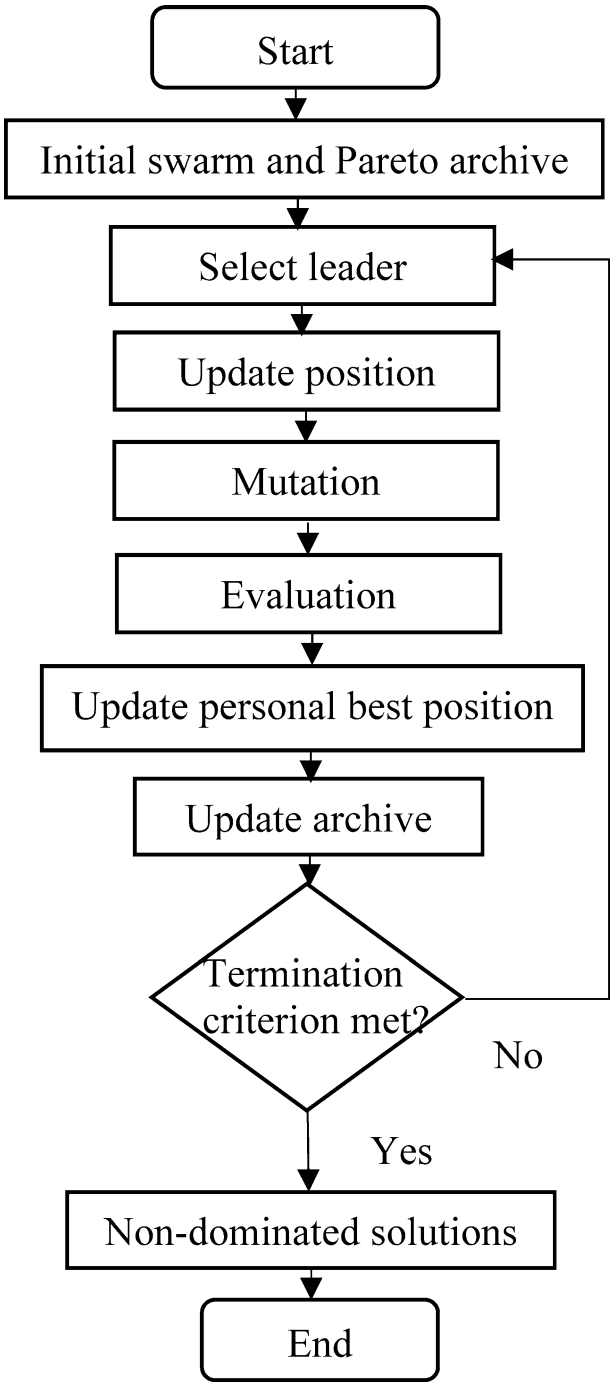


Figure 18. Flowchart of improved multi-objective particle swarm optimization algorithm (OMOPSO) to obtain non-dominated solutions.

6. Remove the individuals in $\overrightarrow{x}$ from population. Set $A = A - 1$ and repeat step 5 while $A > 0$.
7. Increase the generation from G to $G + 1$.
8. Repeat the steps 2 – 7 while $G \leq G_{max}$.

A.4: Improved multi-objective particle swarm optimization (OMOPSO). Figure 18 shows the flowchart for OMOPSO. The following is the description of OMOPSO proposed by Sierra and Coello[44].

1. Initialize a swarm where each particle in the swarm is a candidate solution vector to solve a problem. Evaluate them and initialize the best position and velocity of the particles. Set the size for leaders and initialize leaders

Parameters	NSGA-II, SPEA2	GDE3	OMOPSO	SMPSO
Population size	50	50	50	50
Mutations	Polynomial	–	Uniform, non-uniform	Polynomial mutation
Mutation rate	$\frac{1}{\text{Productgroups}}$	–	$\frac{1}{\text{Productgroups}}$	$\frac{1}{\text{Productgroups}}$
Crossover	Simulated binary	Differential evolution	–	–
Crossover rate	1	1	–	–
Selection	Binary tournament	Differential evolution	–	–
Other parameters		CR = 1 F = 0.4	C1 = random (1.5, 2) C2 = random (1.5, 2) W = random (0.1, 0.5)	C1 = random (1, 2.5) C2 = random (1, 2.5) W = 0.2

Table 5. Parameters setting for the algorithms.

from the existing swarm. Save the leaders in the Pareto archive. Initialize generations, $G = \{1, 2, \ldots, G_{max}\}$. The crowdedness of the leaders is calculated.

2. For each particle select a leader through a binary tournament. The selection criterion is the crowding distances where a leader with a higher crowding distance is chosen.
3. Update the position of the particle to a new position using Eqs. (A9), (A10).

$$\vec{v}_{i,G} = W \times \vec{v}_{i,G-1} + C_1 \times r_1 \times \left(\vec{x}_{i,best} - \vec{x}_{i,G-1}\right) + C_2 \times r_1 \times \left(\vec{L}_h - \vec{x}_{i,G-1}\right) \tag{A9}$$

$$\vec{x}_{i,G} = \vec{x}_{i,G-1} + \vec{v}_{i,G} \tag{A10}$$

where i indicates one particle, $\overrightarrow{v}$ is the velocity, W is the inertia weight, C_1 and C_2 are velocity control parameters, r_1 and r_2 are random numbers between 0 and 1, $\overrightarrow{x}_{i,best}$ and $\overrightarrow{x}_{i,G}$ are particle's best position and current position at generation G, respectively, $\overrightarrow{L}_h$ is a position vector of the selected leader of the h index from the Pareto archive.

4. Divide swarm into three parts to employ distinct mutation treatments: no mutation, uniform mutation, and non-uniform mutation.
5. The particles are evaluated. The personal best position (x_{best}) for each particle is updated by comparing the current and personal best fitness.
6. Update leader set in archive by including non-dominated Pareto solutions and removing dominated solutions. Calculate the crowding distance of the leaders. Eliminate leaders based on crowding distance if the size of the archive exceeds the limit. Increase the generation from G to $G + 1$.
7. Repeat the process (steps 2–6) until G reaches G_{max}. Save the Pareto archive as the set of optimal solutions to the problem.

A.5: Speed-constrained multi-objective particle swarm optimization (SMPSO). SMPSO, an extended version of OMOPSO, was proposed by Nebro et al.[42]. In this proposal, the values for velocity parameters in Eq. (10), C_1 and C_2 are controlled by using following constriction coefficient (χ) calculated by Eq. (A11).

$$\chi = \frac{2}{2 - \varphi - \sqrt{\varphi^2 - 4\varphi}} \tag{A11}$$

where $\varphi = \begin{cases} C_1 + C_2 & if C_1 + C_2 > 4 \\ 1 & otherwise \end{cases}$

The velocity of the particle in each parameter d is bounded using the following velocity constriction equations (Eqs. A12, A13):

$$v_{i,d,G} = \begin{cases} delta_d & if v_{i,d,G} > delta_d \\ -delta_d & if v_{i,d,G} \leq delta_d \\ v_{i,d,G} & otherwise \end{cases} \tag{A12}$$

$$delta_d = \frac{UB_d - LB_d}{2} \tag{A13}$$

where UB_d, and LB_d are upper bound and lower bound of the parameter d.

To summarize the modifications in SMPSO from OMOPSO, for each particle, the velocity is calculated by Eq. (A10), which is then multiplied by the constriction coefficient (χ) (Eq. A11). The resulting value for each parameter is constrained by Eqs. (A12), (A13). The rest of the procedure is the same as OMOPSO. Table 5 shows the parameters setting of the algorithms.

Received: 9 August 2022; Accepted: 21 December 2022
Published online: 05 January 2023

References

1. Babor, M. *et al. Automation Science and Technology* (Food2Multimedia GmbH, Radbruch, 2021).
2. Zentralverband des Deutschen Bäckerhandwerks e. V. Wirtschaftsfaktor Bäckerhandwerk. (2022).
3. Gonzalez, T. & Sahni, S. Flowshop and jobshop schedules: Complexity and approximation. *Oper. Res.* **26**, 36–52 (1978).
4. Liang, Z., Zhong, P., Liu, M., Zhang, C. & Zhang, Z. A computational efficient optimization of flow shop scheduling problems. *Sci. Rep.* **12**, 845 (2022).
5. 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 SA. Process.* **9**, 2044 (2021).
6. 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.* **41**, 5882–5891 (2014).
7. Swangnop, S., Duangdee, T. & Duangdee, J. Design of production planning process for bakery manufacturer. In *2019 IEEE 6th International Conference on Industrial Engineering and Applications (ICIEA)* 178–182 (2019). https://doi.org/10.1109/IEA.2019.8714851.
8. Wang, S., Wang, X., Chu, F. & Yu, J. An energy-efficient two-stage hybrid flow shop scheduling problem in a glass production. *Int. J. Prod. Res.* **58**, 2283–2314 (2020).
9. Liu, L., Chang, Z. & Song, S. Optimization of a molten iron scheduling problem with uncertain processing time using variable neighborhood search algorithm. *Sci. Rep.* **12**, 7303 (2022).
10. Fathollahi-Fard, A. M., Woodward, L. & Akhrif, O. Sustainable distributed permutation flow-shop scheduling model based on a triple bottom line concept. *J. Ind. Inf. Integr.* **24**, 100233 (2021).
11. Dziurzanski, P. *et al.* Solving the multi-objective flexible job-shop scheduling problem with alternative recipes for a chemical production process. In *Applications of evolutionary computation* (eds Kaufmann, P. & Castillo, P. A.) 33–48 (Springer, 2019). https://doi.org/10.1007/978-3-030-16692-2_3.
12. Du, J., Zhang, Z., Li, M., Guo, J. & Zhu, K. Optimal scheduling of integrated energy system based on improved grey wolf optimization algorithm. *Sci. Rep.* **12**, 7095 (2022).
13. Ala, A., Alsaadi, F. E., Ahmadi, M. & Mirjalili, S. Optimization of an appointment scheduling problem for healthcare systems based on the quality of fairness service using whale optimization algorithm and NSGA-II. *Sci. Rep.* **11**, 19816 (2021).
14. Ikeda, K., Nakamura, Y. & Humble, T. S. Application of quantum annealing to nurse scheduling problem. *Sci. Rep.* **9**, 12837 (2019).
15. Valdano, E., Poletto, C., Boëlle, P.-Y. & Colizza, V. Reorganization of nurse scheduling reduces the risk of healthcare associated infections. *Sci. Rep.* **11**, 7393 (2021).
16. Wang, S., Wang, X. & Yu, L. Two-stage no-wait hybrid flow-shop scheduling with sequence-dependent setup times. *Int. J. Syst. Sci. Oper. Logist.* **7**, 291–307 (2020).
17. Yuksel, D., Tasgetiren, M. F., Kandiller, L. & Pan, Q. -K. Metaheuristics for energy-efficient no-wait flowshops: A trade-off between makespan and total energy consumption. In *2020 IEEE Congress on Evolutionary Computation (CEC)* 1–8 (IEEE, 2020). doi:https://doi.org/10.1109/CEC48606.2020.9185554.
18. Zhang, F., Bai, J., Yang, D. & Wang, Q. Digital twin data-driven proactive job-shop scheduling strategy towards asymmetric manufacturing execution decision. *Sci. Rep.* **12**, 1546 (2022).
19. Zhou, L. *et al.* Production and operations management for intelligent manufacturing: A systematic literature review. *Int. J. Prod. Res.* **60**, 808–846 (2022).
20. Huber, J. & Stuckenschmidt, H. Intraday shelf replenishment decision support for perishable goods. *Int. J. Prod. Econ.* **231**, 107828 (2021).
21. Therkelsen, P., Masanet, E. & Worrell, E. Energy efficiency opportunities in the U.S. commercial baking industry. *J. Food Eng.* **130**, 14–22 (2014).
22. Bouman, C. A. *et al.* CLUSTER: An unsupervised algorithm for modeling gaussian mixtures. 20 (1997).
23. Babor, M. & Hitzmann, B. Application of nature-inspired multi-objective optimization algorithms to improve the bakery production efficiency. In *ECP 2022* 31 (MDPI, 2022). doi:https://doi.org/10.3390/ECP2022-12630.
24. Ye, H., Li, W. & Nault, B. R. Trade-off balancing between maximum and total completion times for no-wait flow shop production. *Int. J. Prod. Res.* **58**, 3235–3251 (2020).
25. Babor, M. & Hitzmann, B. Small and medium-sized bakery production data for scheduling. (2022) https://doi.org/10.17632/DHGBSSB8NS.2.
26. Van Rossum, G. & Drake Jr, F. *Python Tutorial; Technical Report CS-R9526*. (Centrum voor Wiskunde en Informatica (CWI): Amsterdam, The Netherlands, 1995).
27. Deb, K., Pratap, A., Agarwal, S. & Meyarivan, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Computat.* **6**, 182–197 (2002).
28. Fonseca, C. M. & Fleming, P. J. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization (1993).
29. Horn, J. D., Nafpliotis, N. & Goldberg, D. A niched Pareto genetic algorithm for multiobjective optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence* (1994). https://doi.org/10.1109/ICEC.1994.350037.
30. Srinivas, N. & Deb, K. Multiobjective function optimization using nondominated sorting genetic algorithms. *Evol. Comput.* **2**, 1301–1308 (1994).
31. Zitzler, E., Deb, K. & Thiele, L. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evol. Comput.* **8**, 173–195 (2000).
32. Zitzler, E., Laumanns, M. & Thiele, L. SPEA2: Improving the strength pareto evolutionary algorithm. In: *Computer Engineering and Networks Laboratory (TIK) Department of Electrical Engineering Swiss Federal Institute of Technology (ETH) Zurich ETH Zentrum* TIK-Report 103, (2001).
33. Zitzler, E. & Thiele, L. An evolutionary algorithm for multiobjective optimization: The strength pareto approach. *Computer Engineering and Communication Networks Lab TIK, Swiss Federal Institute of Technology ETH* 43 (1998).
34. Elhossini, A., Areibi, S. & Dony, R. Strength pareto particle swarm optimization and hybrid EA-PSO for multi-objective optimization. *Evol. Comput.* **18**, 127–156 (2010).
35. Emmerich, M. T. M. & Deutz, A. H. A tutorial on multiobjective optimization: Fundamentals and evolutionary methods. *Nat. Comput.* **17**, 585–609 (2018).
36. Kukkonen, S. & Lampinen, J. GDE3: The third evolution step of generalized differential evolution. In: *2005 IEEE Congress on Evolutionary Computation*, vol. 1, 443–450 (2005).
37. Eberhart, R. & Kennedy, J. A new optimizer using particle swarm theory. In *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science* 39–43 (1995). https://doi.org/10.1109/MHS.1995.494215.
38. Coello, C. A. C., Pulido, G. T. & Lechuga, M. S. Handling multiple objectives with particle swarm optimization. *IEEE Trans. Evol. Comput.* **8**, 256–279 (2004).
39. Coello, C. C. & Lechuga, M. S. MOPSO: A proposal for multiple objective particle swarm optimization. In *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No. 02TH8600)*, Vol. 2, 1051–1056 (IEEE, 2002).
40. Durillo, J. J. *et al.* Multi-objective particle swarm optimizers: An experimental comparison. In *Evolutionary Multi-criterion Optimization* (eds Ehrgott, M. *et al.*) 495–509 (Springer, 2009). https://doi.org/10.1007/978-3-642-01020-0_39.

41. Hu, X. & Eberhart, R. Multiobjective optimization using dynamic neighborhood particle swarm optimization. I:n *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*, Vol. 2, 1677–1681 (2002).
42. Nebro, A. J. *et al.* SMPSO: A new PSO-based metaheuristic for multi-objective optimization. In: *2009 IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making(MCDM)* 66–73 (2009). doi:https://doi.org/10.1109/MCDM.2009.4938830.
43. Ray, T. & Liew, K. M. A swarm metaphor for multiobjective design optimization. *Eng. Optim.* **34**, 141–153 (2002).
44. Sierra, M. R. & Coello Coello, C. A. Improving PSO-based multi-objective optimization using crowding, mutation and ∈-dominance. In *Evolutionary Multi-Criterion Optimization* (eds Coello Coello, C. A. *et al.*) 505–519 (Springer, 2005). https://doi.org/10.1007/978-3-540-31880-4_35.
45. Sun, Y. & Gao, Y. A multi-objective particle swarm optimization algorithm based on Gaussian mutation and an improved learning strategy. *Mathematics* **7**, 148 (2019).
46. Ahmadi, E., Zandieh, M., Farrokh, M. & Emami, S. M. A multi objective optimization approach for flexible job shop scheduling problem under random machine breakdown by evolutionary algorithms. *Comput. Oper. Res.* **73**, 56–66 (2016).
47. Deliktaş, D., Özcan, E., Ustun, O. & Torkul, O. Evolutionary algorithms for multi-objective flexible job shop cell scheduling. *Appl. Soft Comput.* **113**, 107890 (2021).
48. He, L., Cao, Y., Li, W., Cao, J. & Zhong, L. Optimization of energy-efficient open shop scheduling with an adaptive multi-objective differential evolution algorithm. *Appl. Soft Comput.* **118**, 108459 (2022).
49. Li, X. & Ma, S. Multi-objective memetic search algorithm for multi-objective permutation flow shop scheduling problem. *IEEE Access* **4**, 2154–2165 (2016).
50. Santos, V. L. A., Carvalho, T. F. M., de Assis, L. P., Weiss-Cohen, M. & Guimarães, F. G. Multi-objective iterated local search based on decomposition for job scheduling problems with machine deterioration effect. *Eng. Appl. Artif. Intell.* **112**, 104826 (2022).
51. Abido, M. A. & Elazouni, A. Modified multi-objective evolutionary programming algorithm for solving project scheduling problems. *Expert Syst. Appl.* **183**, 115338 (2021).
52. Ye, H., Li, W., Abedini, A. & Nault, B. An effective and efficient heuristic for no-wait flow shop production to minimize total completion time. *Comput. Ind. Eng.* **108**, 57–69 (2017).
53. Ye, H., Li, W. & Abedini, A. An improved heuristic for no-wait flow shop to minimize makespan. *J. Manuf. Syst.* **44**, 273–279 (2017).
54. Jolliffe, I. T. *Principal Component Analysis* (Springer-Verlag, 2002).
55. 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* **10**, 1623 (2022).
56. Ye, H., Wang, X. & Liu, K. Adaptive preventive maintenance for flow shop scheduling with resumable processing. *IEEE Trans. Automat. Sci. Eng.* **18**, 106–113 (2021).
57. Miyata, H. H., Nagano, M. S. & Gupta, J. N. D. Integrating preventive maintenance activities to the no-wait flow shop scheduling problem with dependent-sequence setup times and makespan minimization. *Comput. Ind. Eng.* **135**, 79–104 (2019).
58. Storn, R. & Price, K. Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces. *J. Global Optim.* **11**, 341–359 (1997).
59. Kukkonen, S. & Lampinen, J. An Extension of Generalized Differential Evolution for Multi-objective Optimization with Constraints. in *Parallel Problem Solving from Nature - PPSN VIII* (eds. Yao, X. et al.) 752–761 (Springer, 2004). doi:https://doi.org/10.1007/978-3-540-30217-9_76.
60. Lampinen, J. A constraint handling approach for the differential evolution algorithm. in *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)* vol. 2 1468–1473 (2002).

Author contributions

M.B. performed conceptualization, methodology, software, formal analysis, investigation, visualization, writing—original draft preparation. O.P. contributed to conceptualization and methodology. R.K. contributed to methodology and supervised the study. 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

Open Access funding enabled and organized by Projekt DEAL. 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".

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to M.B.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

3.4. A resilient bakery production schedule under uncertain proofing time

In the optimization results presented in publications (sections 3.1, 3.2, and 3.3), the durations for processing stages are assumed to be deterministic. A change in the duration of a single stage has an impact on the optimized schedule, and in the worst-case scenario, it may cause a significant increase in makespan. The impact of uncertain stage duration on an optimized schedule is discussed in this section.

3.4.1. Background

In theory, practitioners limit themselves to reactive production planning that is obtained based on deterministic data about resources such as processing duration, machinery, and materials. Many industrial processes, in contrast, are subject to uncertainties and disturbances due to duration, the availability of resources, and the nature of the materials, which are impossible to eliminate. In many bakery products, yeast is added as a leavening agent that improves the texture and taste of the bread. Due to the fermentation of yeast, CO_2 is produced, which makes bread fluffy, along with other aroma precursors that improve the taste. However, this fermentation is a long process and is called "proofing". The duration of proofing is uncertain as it depends on the quality of flour, metabolism of yeast cells, temperature, and humidity. Bakers decide the completion of proofing based on their personal experience. Such uncertainty in proofing time significantly influences the stability of the planned schedule. As a result, despite optimizing the planning, the entire schedule of machines and workers must be changed because of the uncertain proofing time, resulting in inefficient production planning with a longer makespan.

3.4.2. Contribution

The main contribution of this section is to propose a proactive production planning approach for small and medium-sized bakeries. The uncertainty in the proofing time is encountered in resilient and proactive production planning so that the impact on the makespan and oven idle time is minimal even if it changes by ±10 % of the expected proofing time.

3.4.3. Materials and methods

To conduct this study, data from one small bakery manufacturing line (BK20) with about 20 products, and 8 machines, including 2 ovens, is used [1]. The three compartments in one oven allow for the independent baking of three different products, while the other one has one baking chamber. The hybrid no-wait flow shop model (HNFSM, the best performing single objective optimization algorithm modified particle swarm optimization (MPSO) and multi-objective optimization algorithm non-dominated sorting genetic algorithm (NSGA-II) are used in this section. For details on MPSO and NSGA-II, see sections 3.2 and 3.3, respectively. The implementation and simulation of HNFSM and NSGA II were performed using the computer language Python 3.7 on a computer running Microsoft Windows 10 as the operating system with a configuration of an Intel Core i5 at 4 × 3.20 GHz, 8 GB ram.

3.4.4. Resilient production planning

Figure 1 depicts the approach used to obtain proactive bakery production scheduling when proofing time is uncertain. To begin the solution process, an initial production sequence is used in addition to employee, machine, and product data (module A). Initially, the expected proofing duration, which was the actual proofing duration on the day of data collection, is used. Module B only modifies the duration of the proofing stages. The proofing time deviation (PTD) is the rate of change in proofing time that ranges from -10 % to 10 % of the expected proofing time. As a result, the expected proofing time is changed 21 times with the same production sequence, and each time HNFSM produces a schedule and calculates the objectives in module C. The robustness of one production sequence is determined using the stored objective values to start module D, where the optimization algorithm generates a new production sequence to repeat the process until a stopping criterion is met.

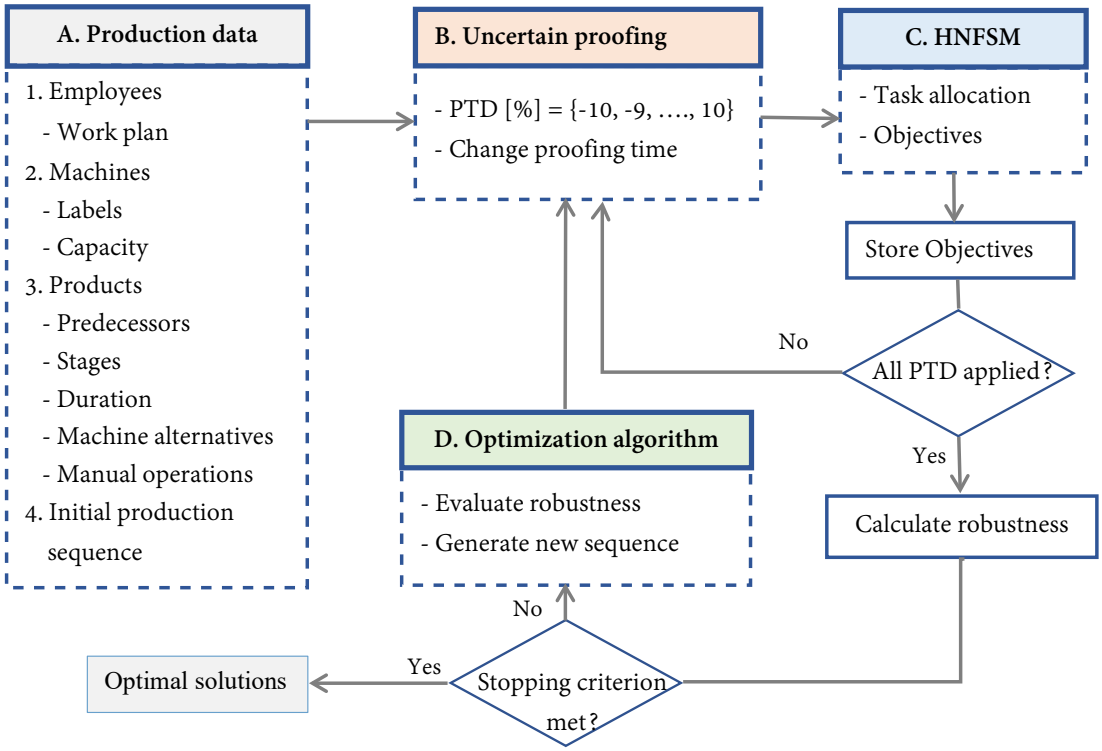


Fig. 1 The solution approach for determining a resilient production schedule with uncertain proofing time. The proofing time deviation (PTD) ranges from -10 % to 10 % of the expected proofing time.

Due to PTD, 21 makespan values ($makespan_{PTD}$) are calculated from a single production sequence during the solution process. The robustness of a production sequence is defined by the maximum makespan (Eq. 1) and minimum makespan (Eq. 2), which are to be minimized.

$$Min\ (max\ (makespan_{PTD})) \quad (1)$$

$$Min\ (min\ (makespan_{PTD})) \quad (2)$$

3.4.5. Results and discussion

When the proofing time is increased, optimized schedules with the shortest makespan are more likely to be extended. The single optimization algorithm MPSO was run multiple times to find the production planning with the single objective of minimizing makespan under the assumption that proofing time is as deterministic as expected. The optimized production sequences were then used to test the robustness with PTD. Table 1 shows that there are multiple solutions with lower makespan within a small range of makespan, with the minimum being 424 min for this production line. Interestingly, despite applying PTD, where it is assumed that the proofing time can be as low as 10 %, there is almost no possibility of reducing makespan, as demonstrated by the minimum makespan after applying PTD (Table 1). The machine and worker limitations prevent from finishing production early, despite lowering the proofing time. The maximum makespan and range of makespan, in contrast, show a significant increase in makespan due to PTD. MPSO obtained the production sequences, but the uncertain proofing may result in a significant increase of up to 18 %.

Table 1 Influence of uncertain proofing time on optimized makespan.

Optimized makespan	**Makespan after applying PTD**		
	Minimum	**Maximum**	**Increased [% optimized]**
424	424	470	11
427	426	467	10
429	429	499	16
429	429	506	18
430	430	495	15
430	430	474	10

NSGA-II was used to generate optimized schedules with the shortest maximum makespan and minimum makespan due to uncertain proofing time. Fig. 2 represents four Pareto solutions obtained by NSGA-II. The objective space shows a significant reduction in maximum makespan within only 5 min range of minimum makespan. It indicates with a small increase in minimum makespan, the production planning can be more resilient. The minimum makespan (PTD) is the lowest possible makespan due to proofing time deviation. Solution G1 shows 422 min of minimum makespan, which is only 2 min smaller than the shortest makespan obtained by MPSO with no change in proofing time. However, G1 shows makespan can be as high as 476 min due to uncertain proofing time. Solution G2 and G3 show a consistent decrease in the maximum makespan of PTD. Solution G4 shows only 21 min difference between the maximum makespan and minimum makespan of PTD while the minimum makespan is only 5 min higher than the shortest possible makespan. As a result, the production sequence that corresponds to the G4 solution in objective space could provide resilient production planning, with a makespan that can range from 427 to 448 min due to uncertain proofing time.

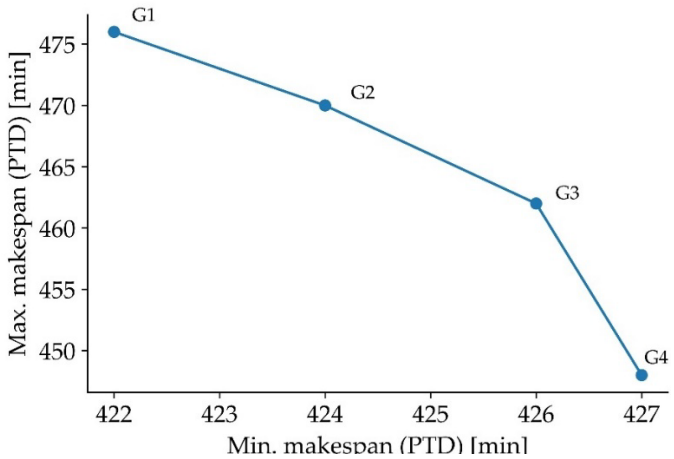


Fig. 2 Trade-offs between two objectives between Pareto solutions obtained by NSGA-II

3.4.6. Conclusion

In this section, a small bakery production line was investigated to find a resilient production planning solution when dealing with proofing time uncertainty using a hybrid no-wait flow shop model and the non-dominated sorting genetic algorithm (NSGA-II). The deviation in proofing time is considered to be between -10 % and +10 % of the actual proofing time.

The computational results showed that uncertainty in proofing time has a significant impact on makespan. Production planning with the shortest makespan may result in inefficient production with an increase in makespan of up to 18 %. In contrast, NSGA-II obtained multiple solutions that show minimum makespan deviation due to uncertain proofing time and only 5 min higher than the shortest possible makespan. The most resilient production planning had a makespan deviation of only 21 min, with a minimum of 427 min.

Because makespan dominates the cost of bakery production, a substantial increase in makespan due to uncertain proofing time may increase the cost. An optimized schedule can also fail to be efficient, and it can be more prominent for a larger number of products. As a result, this approach of finding resilient and proactive production planning with a multi-objective optimization algorithm can contribute to a sustainable production system in small and medium-sized bakeries.

Availability of Data and Materials

The bakery production data used for this analysis is published by Babor and Hitzmann [1].

Reference

[1] Babor, M., Hitzmann, B. "Production data from a small bakery manufacturing line", Mendeley Data, V1, 2022, https://doi.org/10.17632/7x5t3rxx5f.1

Chapter 4

Discussions, Conclusions and Final Remarks

4.1. Discussions

In the bakery manufacturing, the dough's behavior changes as a result of biological and physical processes, like yeast fermentation. Due to the inclusion of living yeast cells, time and temperature have a considerable impact on CO_2 production, which in turn affects the final product's texture and aroma. Therefore, every processing step must be finished accordingly from the beginning without any delay in between. The model used to schedule products in this manner is known as the no-wait flow shop model. Several studies have described the mathematical modeling of the no-wait flow shop model [24, 25, 45, 46]. In contrast, a few studies [7, 47–49] focused on bakery production schedules, but none of them described the constraints of small and medium-sized bakeries in production scheduling model. The flow shop model's constraints always determine the model's complexity; however, without incorporating these constraints into the flow shop model, the schedule cannot be realistic and applicable in any real production line. In large scale bakeries, the scenarios can be different, and thus most of these constraints may not be applicable.

Hecker et al. [7] used data from a single bakery production line to implement a permutation flow shop model for solving bakery production optimization problem. In the permutation flow shop model, all products must pass through all machines in the same order [29, 50]. Bakeries, on the other hand, have alternative machines for a processing task, such as kneader 1 and kneader 2 for kneading, and if any product is kneaded by kneader 1, it skips kneader 2 and vice versa. The existence of such parallel machine formation was simplified in their study by using zero minute processing time for machines that are irrelevant and skipped by a product. The method worked for the investigated production line, but it leaves many unnecessary stages with zero minute duration for a product in the schedule. Despite the maximum stage for any product being 12, the production schedule shows 26 stages for each product. It occurred as a result of creating a common order of machines and stages to work for each product, which differs greatly between bakeries. Although that there were many manual tasks in the production, the employee's work plan was not included. There were no predecessors in products, which means that any product can be scheduled at any time and is not dependent on other products.

Huber and Stuckenschmidt [48] investigated bakery retail store operations in Germany. The authors concentrated on creating a decision support system by forecasting hourly demand and scheduling only baking to serve customers freshly baked goods. The scheduling here remains limited by ovens, baking duration, and baking trays, which are dependent on the number of bakery goods predicted by the forecast model. To simulate the production planning of a bakery with ten bakery goods and identify inefficiency and bottlenecks, Hussein et al. [51] used Arena software [52]. The authors demonstrated that the actual production line was run with suboptimal planning, and pointed out some of the reasons behind it. The authors added that because machines were left idle for extended periods of time, the current planning wasted a substantial amount of energy. However, as the major goal was to use production simulation to

determine the scope of improvement, no additional optimization methods were applied to determine the best production scheduling.

The flow shop model with constraints in small and medium-sized bakeries is explored in this thesis since it has not been used in any pertinent investigations that have been undertaken thus far. In small and medium-sized bakeries, dough of various products is prepared together before being processed through a few common stages. This is done in order to benefit from the machines' capacity, which is used to process some initial stages, including kneading. In another instance, bakers use the same oven chamber for multiple goods that came across different processing routes. Even though the early processing steps are all carried out in various machines in this instance, the baking is done simultaneously in a single oven compartment. Additionally, a lot of processing tasks are carried out manually by workers. Due to the limited number of employees, their work schedule has a big impact on the production schedule. Section 3.1 provided a description of these constraints featured in the hybrid no-wait flow shop model proposed in this thesis. The constraints are shown graphically and mathematically in sections 3.2 and 3.3.

The minimization of makespan [16–19], tardiness [16, 17, 26, 53], earliness [26, 53], flow time [19], carbon emission [16, 17] and energy consumption [17, 18, 20] are among the objectives of optimal manufacturing that have been extensively explored in the literature. Ovens are kept running in bakeries to prevent production from disruptions because it takes time for them to reach the desired temperature if they are turned off in between baking two batches of goods. Furthermore, compared to other machines used in bakeries, ovens consume a significant amount of energy. Therefore, the energy used by ovens when they are not in use is wasted. In a perfect scenario, the energy consumption data from ovens in particular would be used to determine the least energy wasting production planning. Nevertheless, small and medium-sized bakeries do not have the facility to track the energy used by a single machine. The ideal alternative, which was employed in this study to keep the input requirement low for optimizing the actual manufacturing line, is the idle time of the ovens.

Using single objective optimization approach, Hecker et al. [22] separately used makespan and total machine idle time. The disadvantage of this strategy is that, as this thesis clearly demonstrates, planning with the shortest makespan invariably results in increased overall machine idle time. Utilizing weighting factors and treating each target as a single goal is another strategy for dealing with multiple objectives. This strategy was used in this thesis at the initial phase (see section 3.1). The idea of Pareto dominance was later applied to help decision-makers visualize the trade-offs between objectives (see section 3.2 and section 3.3). This method is well-established in many domains, including operations research, to address multi-objective optimization problems.

With more than two machines, production scheduling is a non-deterministic polynomial-time (NP) hard problem, and as the number of products, processing stages, and alternative machines for stages rises, it becomes more difficult to find optimized schedules. The entire production schedule can be simulated using a flow shop model, and a graphical representation of the simulated scheduling can reveal

potential inefficiencies. The question is, however, what production planning actually eliminates these inefficiencies. With the available computer resources, it is not feasible to complete the simulation of production schedules using every possible combination of product orders and comparisons in a reasonable amount of time in order to obtain the exact optimized production planning. To address this issue, the application of nature-inspired optimization algorithms is established method that can provide optimized schedule to a satisfactory level.

The behavior, natural evolution, and development of animals serve as the basis for nature-inspired optimization algorithms. They are widely called as swarm intelligence and evolutionary algorithms [54]. These algorithms are employed to solve several problems, such as feature selection [55, 56], parameter estimation [57, 58], classification [59–61], mathematical functions [43, 62], supply chain optimization [63–65], travelling salesman problem [66–68], networking [69, 70], inverse problem [71, 72], producing planning problem [7–9, 73], and image processing [74, 75]. The advantage of employing these algorithms is that they are efficient in solving high dimensional optimization problems with or without the constraints and regardless of the complexity of the system. However, because multiple runs provide varied outcomes, it remains uncertain whether the obtained result is optimized. Multiple repetitions of the process are required, which is highly time consuming, and yet the final outcome might not be optimum. Therefore, nature-inspired algorithms are widely criticized for their inability to solve real-world problems efficiently [76]. In order to overcome the shortcomings many researchers have modified them to make the searching procedure efficient, such that the probability of obtaining optimized result is high [76–80]. The advantage of these proposed methods is the accelerated searching speed with which the optimum combination of variables is found in variables space during optimization effectively.

The concept of accelerated searching process can be described by exploration, which steers search agents toward a global search, and exploitation, which points to a local search. Despite having the advantages of these approaches independently in solving many problems, solving many high dimensional real optimization problem requires both of them in single run [81]. In this thesis, one of the most popular nature-inspired optimization methods, particle swarm optimization, was modified and found to have increased performance with a higher likelihood of obtaining optimum outcomes [8]. The proposed modified particle swarm optimization algorithm established a combination of both exploration and exploitation with a particular proportion in a run as a means of finding solutions (see section 3.1). This work shows that such a modification enhances algorithm convergence, identifies the optimal region from a high-dimensional search space, and increases the likelihood of discovering a better solution through further exploration close to the optimal area. It outperformed standard particle swarm optimization, simulated annealing, and Nawaz-Enscore-Ham optimization methods in solving real bakery production optimization problem. The optimization algorithm discovered a solution that reduced the makespan by 30 % and oven idle time by 8 %, according to the results based on a Spanish bakery production dataset [8].

Several multi-objective optimization methods have been employed to resolve production scheduling problems. According to the literature, production efficiency has significantly increased in many manufacturing environments. Studies have used a variety of cutting-edge multi-objective optimization algorithms to determine the Pareto front for multiple objectives to aid in decision-making. Among them, non-dominated sorting genetic algorithm (NSGA-II) has been widely used in many production environment [82], such as metal [83], electricity tariff-based scheduling [20, 30, 84], packaging [16], glass [20] . In this thesis, the production schedules for a Danish bakery were optimized using NSGA-II and multi-objective random search technique (see section 3.2). The results demonstrated that the schedules with the smallest makespan do not necessarily have the smallest oven idle time. The trade-offs in objective space demonstrated that the shortest makespan schedules have a large oven idle time, resulting in energy waste. In contrast, the schedule with a little longer makespan significantly reduced oven idle time, which lowers overall production costs. This suggests the benefit of having several solutions rather than one that addresses multiple objectives, such as by using weighting factors for objectives to make single objective space. A single solution leaves open the question of whether one or both objectives could still be improved, as well as which objective was compromised or gained, and by how much. According to the results of the research, NSGA-II improved the production schedules by reducing oven idle time by up to 26 % and makespan by up to 8 %. Furthermore, alternative optimal solutions reduced oven idle time by up to 61 % relative to the existing schedule by marginally penalizing the best makespan in objective space [9].

The performance of the multi-objective method is vital because the dimension of the production scheduling problem is bigger, for example, the search space contains 20 dimensions for 20 products. Several research have examined the effectiveness of these algorithms, mostly in solving mathematical functions, and have proposed better versions of these algorithms. In this thesis five state-of-the-art multi-objective optimization algorithms — non-dominated sorting genetic algorithm (NSGA-II), strength Pareto evolutionary algorithm (SPEA2), and generalized differential evolution (GDE3), multi-objective particle swarm optimization (OMOPSO), and speed-constrained multi-objective particle swarm optimization (SMPSO) — were applied (see section 3.4). Four quality indicators — cardinality, convergence, distribution and spread, and convergence and distribution of the derived solutions — are employed to evaluate their efficacy. The optimization results from three different bakery production datasets revealed that NSGA-II performed better compared to other algorithms.

In practice, a lot of production systems have process variable uncertainties, such duration and resource availability, which have a big impact on the ideal schedule. Several research examined the risk, reliability, resilience, and sensitivity in planning due to process variable uncertainty. Many studies provided frameworks for risk assessment for project scheduling to predict the scenarios, the influence on project completion time and cost owing to duration uncertainty, and resource availability [85–88]. Himmiche et al. [27] used stochastic discrete event system to evaluate the robustness of the schedules

under machine failures and uncertain reparation duration. A stochastic mixed integer programming for underground mine production scheduling optimization was proposed by Huang et al. [89] by taking into account the grade uncertainty of the mines. Moradi and Shadrokh [90] investigated the resilience of schedules for maintenance activities during planned shutdown of a gas refinery plant. For parallel machine rescheduling, a fuzzy logic based decision support system was proposed by Petrovic and Duenas [91] in the presence of uncertain disruption. Liu et al. [28] studied on robust optimization of a molten iron scheduling problem under uncertain processing time and demonstrated the significant improvement in production efficiency.

Similarly, in bakery production the proofing time shows uncertainty and therefore, it affects the production schedule. However, there is no study that evaluated the impact of uncertain proofing time in bakery production planning. To address the uncertainty in proofing time in bread making process, a robust production planning is proposed in this thesis (see section 4.3). A few optimal production plans were obtained utilizing a production dataset and the proposed modified particle swarm optimization algorithm with the only goal of makespan minimization. The findings indicated that the uncertain proofing could significantly lengthen the optimal makespan by up to 18 %. By applying multi-objective optimization algorithm NSGA-II, the study found a resilient production planning for the studied problem that has minimum makespan deviation. The most robust bakery production planning showed a makespan deviation of only 21 min, with a minimum of 427 min.

4.2. Conclusions

This study presented a framework for improving the production efficiency of small and medium-sized bakeries by using nature-inspired optimization algorithms. Production data from real bakeries were used to integrate the underlying production floor constraints in a hybrid no-wait flow shop model (HNFSM), which is used to simulate production planning based on actual data. The objectives were to minimize the makespan and oven idle time at the same time. In the first phase, two objectives were unified using the weighting factors method, and the problem was solved by applying classical single objective optimization algorithms. Particle swarm optimization (PSO), simulated annealing, and Nawaz-Enscore-Ham algorithms were used to optimize an existing production line. In the second phase, multi-objective algorithms were used to find Pareto optimal solutions for problems that show trade-offs between objectives to assist in decision-making. State-of-the-art evolutionary and particle swarm optimization-based metaheuristics algorithms, the non-dominated sorting genetic algorithm (NSGA-II), the strength Pareto evolutionary algorithm, generalized differential evolution, improved multi-objective particle swarm optimization, and speed-constrained multi-objective particle swarm optimization were used. To solve production planning problems efficiently, a computational experiment on determining the performance of multi-objective algorithms was conducted. Finally, an approach is proposed to address the negative impact of uncertain proofing time on optimized schedules with the shortest makespan in order to find resilient production planning.

The HNFSM incorporates the constraints of multiple small and medium-sized bakeries from seven different EU nations into a single framework, which is further explained with the mathematical formulation. The computational results revealed that almost all the studied production lines are under-optimized, and there are several alternative production plans that could significantly increase efficiency without requiring any infrastructure changes. The makespan and oven idle time were significantly reduced by using nature-inspired optimization algorithms. The study clearly shows that the shortest makespan does not necessarily provide the shortest oven idle time, and thus it should be considered in the objectives. When compared to existing production schedules, the makespan can be reduced by up to 29 %, while oven idle time can be reduced by up to 26 %. Furthermore, schedules with the shortest or nearly shortest makespan are prone to a significant increase in makespan due to the uncertain proofing time. Makespan of an optimized schedule of 20 products increased by as high as 18 % due to uncertain proofing time. The study found that by including uncertain proofing time in the optimization process, resilient producing planning can be obtained with a minimal impact on the final makespan, deviation of no more than 21 minutes, despite an extreme deviation in proofing time (-10 % to 10 % of actual proofing time).

This study investigated the effectiveness of optimization algorithms in addition to solving real-world production planning problems. PSO was modified to address one of the drawbacks of using nature-inspired optimization algorithms: being trapped in local minima when solving high-dimensional problems. The modified PSO algorithm outperformed classical single objective optimization algorithms

in terms of finding optimal solutions and was found to be less prone to being trapped in local minima. Within a reasonable computational time, modified PSO can find nearly the best, if not the best, production planning. According to the study's findings, NSGA-II performed better than all other multi-objective algorithms in terms of effectively finding Pareto solutions.

A common framework featuring a hybrid no-wait flow shop model and various nature-inspired optimization algorithms was used to solve multiple real-world bakeries' production planning problems. The findings of this study directly contribute to improving the production efficiency of small and medium-sized bakeries and lowering production costs by minimizing manufacturing and oven idle time. The impact of additional machines on total production efficiency can also be determined using this framework. Therefore, before making plans to purchase new machines, a baker can estimate advantages. As part of Industry 4.0, this approach can assist small and medium-sized bakeries in becoming more dynamic, sustainable, and resilient in production planning and operations.

4.3. Final Remarks

The crucial aspect of optimizing a bakery production line is gathering production data on product recipes, machines, and workers. Information such as the predecessor, duration, and alternative machines and workers to perform the task must be collected for each processing stage. Similarly, a list of machines with their capacities and workers with their shift schedules are required to construct a realistic production plan. The precision of the data determines whether optimized planning can be implemented in a real production line.

The computational time for this approach is intensive. Furthermore, because the product range in bakeries changes frequently, the optimization procedure must be repeated on a regular basis. A production line with 20 products, on the other hand, can be optimized to a satisfactory level in about 20 min of computational time if the optimizer is well-tuned and effective enough. Because computing power is expected to increase in the near future, such a framework can be made available online for bakers.

Bibliography

[1] M. Ghobakhloo, "Industry 4.0, digitization, and opportunities for sustainability," Journal of Cleaner Production, vol. 252, p. 119869, 2020, https://doi.org/10.1016/j.jclepro.2019.119869

[2] A. Hassoun et al., "The fourth industrial revolution in the food industry-Part I: Industry 4.0 technologies," Critical reviews in food science and nutrition, pp. 1–17, 2022, https://doi.org/10.1080/10408398.2022.2034735

[3] S. Meyerding, A. Kürzdörfer, and B. Gassler, "Consumer Preferences for Superfood Ingredients—the Case of Bread in Germany," Sustainability, vol. 10, no. 12, p. 4667, 2018, https://doi.org/10.3390/su10124667

[4] Zentralverband des Deutschen Bäckerhandwerks e., V., "Wirtschaftsfaktor Bäckerhandwerk," 2022, 2022, [online] https://www.baeckerhandwerk.de/baeckerhandwerk/zahlen-fakten (accessed: Sep. 1 2022).

[5] Zentralverband des Deutschen Bäckerhandwerks e., V., "Umsatzentwicklung und -verteilung," 2022, 2022. [online] https://www.baeckerhandwerk.de/baeckerhandwerk/zahlen-fakten/umsatzentwicklung-und-verteilung (accessed: Sep. 1 2022).

[6] M. Babor, J. Senge, B. Hitzmann, D. Yee, Y. Huang, J. Fischer, R. Sollacher, "Production and selling optimizations for bakeries, Automation Science and Technology," Food2Multimedia GmbH, p. 60-69, 2021.

[7] F. T. Hecker, M. Stanke, T. Becker, and B. Hitzmann, "Application of a modified GA, ACO and a random search procedure to solve the production scheduling of a case study bakery," Expert Systems with Applications, vol. 41, no. 13, pp. 5882–5891, 2014, https://doi.org/10.1016/j.eswa.2014.03.047

[8] M. Babor, J. Senge, C. M. Rosell, D. Rodrigo, and B. Hitzmann, "Optimization of No-Wait Flowshop Scheduling Problem in Bakery Production with Modified PSO, NEH and SA," Processes, vol. 9, no. 11, p. 2044, 2021, https://doi.org/10.3390/pr9112044

[9] M. Babor, L. Pedersen, U. Kidmose, O. Paquet-Durand, and B. Hitzmann, "Application of Non-Dominated Sorting Genetic Algorithm (NSGA-II) to Increase the Efficiency of Bakery Production: A Case Study," Processes, vol. 10, no. 8, p. 1623, 2022, https://doi.org/10.3390/pr10081623

[10] P. Poinot et al., "Influence of formulation and process on the aromatic profile and physical characteristics of bread," Journal of Cereal Science, vol. 48, no. 3, pp. 686–697, 2008, https://doi.org/10.1016/j.jcs.2008.03.002

[11] M. C. Zghal, M. G. Scanlon, and H. D. Sapirstein, "Effects of Flour Strength, Baking Absorption, and Processing Conditions on the Structure and Mechanical Properties of Bread Crumb," Cereal Chemistry Journal, vol. 78, no. 1, pp. 1–7, 2001, https://doi.org/10.1094/CCHEM.2001.78.1.1

[12] A. Cappelli, L. Bettaccini, and E. Cini, "The kneading process: A systematic review of the effects on dough rheology and resulting bread characteristics, including improvement strategies," Trends in Food Science & Technology, vol. 104, pp. 91–101, 2020, https://doi.org/10.1016/j.tifs.2020.08.008

[13] M. Babor and B. Hitzmann, "Small and medium-sized bakery production data for scheduling," 2022, https://doi.org/10.17632/dhgbssb8ns.2

[14] M. Babor and B. Hitzmann, "Production data from a small bakery manufacturing line," 2022, https://doi.org/10.17632/7x5t3rxx5f.1

[15] S. A. Mansouri, E. Aktas, and U. Besikci, "Green scheduling of a two-machine flowshop: Trade-off between makespan and energy consumption," European Journal of Operational Research, vol. 248, no. 3, pp. 772–788, 2016, https://doi.org/10.1016/j.ejor.2015.08.064

[16] X. Wen, K. Wang, H. Li, H. Sun, H. Wang, and L. Jin, "A two-stage solution method based on NSGA-II for Green Multi-Objective integrated process planning and scheduling in a battery packaging machinery

workshop," Swarm and Evolutionary Computation, vol. 61, p. 100820, 2021, https://doi.org/10.1016/j.swevo.2020.100820

[17] D. X. Huo, X. J. Xiao, and Y. J. Pan, "Multi-Objective Energy-Saving Job-Shop Scheduling Based on Improved NSGA-II," Int. j. simul. model., vol. 19, no. 3, pp. 494–504, 2020, https://doi.org/10.2507/IJSIMM19-3-CO12

[18] Q. Zeng, J. Li, R. Li, T. Huang, Y. Han, and H. Sang, "Improved NSGA-II for energy-efficient distributed no-wait flow-shop with sequence-dependent setup time," Complex Intell. Syst., 2022, https://doi.org/10.1007/s40747-022-00830-6

[19] V. Fernandez-Viagas, P. Perez-Gonzalez, and J. M. Framinan, "The distributed permutation flow shop to minimise the total flowtime," Computers & Industrial Engineering, vol. 118, pp. 464–477, 2018, https://doi.org/10.1016/j.cie.2018.03.014

[20] M. Liu, X. Yang, F. Chu, J. Zhang, and C. Chu, "Energy-oriented bi-objective optimization for the tempered glass scheduling," Omega, vol. 90, p. 101995, 2020, https://doi.org/10.1016/j.omega.2018.11.004

[21] M. R. Garey, D. S. Johnson, and R. Sethi, "The Complexity of Flowshop and Jobshop Scheduling," Mathematics of OR, vol. 1, no. 2, pp. 117–129, 1976, https://doi.org/10.1287/moor.1.2.117

[22] H. Stefansson, S. Sigmarsdottir, P. Jensson, and N. Shah, "Discrete and continuous time representations and mathematical models for large production scheduling problems: A case study from the pharmaceutical industry," European Journal of Operational Research, vol. 215, no. 2, pp. 383–392, 2011, https://doi.org/10.1016/j.ejor.2011.06.021

[23] M. A. Abdeljaouad, Z. Bahroun, A. Omrane, and J. Fondrevelle, "Job-shop production scheduling with reverse flows," European Journal of Operational Research, vol. 244, no. 1, pp. 117–128, 2015, https://doi.org/10.1016/j.ejor.2015.01.013

[24] S. Abdollahpour and J. Rezaian, "Two new meta-heuristics for no-wait flexible flow shop scheduling problem with capacitated machines, mixed make-to-order and make-to-stock policy," Soft Comput, vol. 21, no. 12, pp. 3147–3165, 2017, https://doi.org/10.1007/s00500-016-2185-z

[25] R.-X. Chen, S.-S. Li, and W.-J. Li, "Multi-agent scheduling in a no-wait flow shop system to maximize the weighted number of just-in-time jobs," Engineering Optimization, vol. 51, no. 2, pp. 217–230, 2019, https://doi.org/10.1080/0305215X.2018.1458844

[26] O. Engin and B. Engin, "Hybrid flow shop with multiprocessor task scheduling based on earliness and tardiness penalties," JEIM, vol. 31, no. 6, pp. 925–936, 2018, https://doi.org/10.1108/JEIM-04-2017-0051

[27] S. Himmiche, P. Marangé, A. Aubry, and J.-F. Pétin, "Robust production scheduling under machine failures - A DES based evaluation approach," IFAC-PapersOnLine, vol. 51, no. 7, pp. 271–276, 2018, https://doi.org/10.1016/j.ifacol.2018.06.312

[28] L. Liu, Z. Chang, and S. Song, "Optimization of a molten iron scheduling problem with uncertain processing time using variable neighborhood search algorithm," Scientific reports, vol. 12, no. 1, p. 7303, 2022, https://doi.org/10.1038/s41598-022-10891-9

[29] I. H. Osman and C. N. Potts, "Simulated annealing for permutation flow-shop scheduling," Omega, vol. 17, no. 6, pp. 551–557, 1989 https://doi.org/10.1016/0305-0483(89)90059-5

[30] M. F. Rego, J. C. E. M. Pinto, L. P. Cota, and M. J. F. Souza, "A mathematical formulation and an NSGA-II algorithm for minimizing the makespan and energy cost under time-of-use electricity price in an unrelated parallel machine scheduling," PeerJ. Computer science, vol. 8, e844, 2022, https://doi.org/10.7717/peerj-cs.844

[31] E. M. Zakharova and I. K. Minashina, "Review of multidimensional optimization methods," J. Commun. Technol. Electron., vol. 60, no. 6, pp. 625–636, 2015, https://doi.org/10.1134/S1064226915060194

[32] J. Kennedy and R. Eberhart, "Particle swarm optimization: Proceedings of ICNN'95 - International Conference on Neural Networks, Perth, WA, Australia," 1995, https://doi.org/10.1109/ICNN.1995.488968

[33] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," Science (New York, N.Y.), vol. 220, no. 4598, pp. 671–680, 1983, https://doi.org/10.1126/science.220.4598.671

[34] N. Gunantara, "A review of multi-objective optimization: Methods and its applications," Cogent Engineering, vol. 5, no. 1, p. 1502242, 2018, https://doi.org/10.1080/23311916.2018.1502242

[35] K. Deb, "Multi-objective Optimization," in Search Methodologies, E. K. Burke and G. Kendall, Eds., Boston, MA: Springer US, 2014, pp. 403–449, https://doi.org/10.1007/978-1-4614-6940-7_15

[36] M. Ehrgott, Multicriteria optimization, 2nd ed. Springer Berlin, Heidelberg, 2005.

[37] N. Gunantara and N. P. Sastra, "Cooperative diversity selection protocol using Pareto method with multi objective criterion in wireless ad hoc networks," IJMUE, vol. 11, no. 5, pp. 43–54, 2016, https://doi.org/10.14257/ijmue

[38] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE Trans. Evol. Computat., vol. 6, no. 2, pp. 182–197, 2002, https://doi.org/10.1109/4235.996017

[39] C. Coello, G. T. Pulido, and M. S. Lechuga, "Handling multiple objectives with particle swarm optimization," IEEE Trans. Evol. Computat., vol. 8, no. 3, pp. 256–279, 2004, https://doi.org/10.1109/TEVC.2004.826067

[40] M. R. Sierra and C. A. Coello Coello, "Improving PSO-Based Multi-objective Optimization Using Crowding, Mutation and ∈-Dominance," in Lecture Notes in Computer Science, Evolutionary Multi-Criterion Optimization, D. Hutchison et al., Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 505–519, https://doi.org/10.1007/978-3-540-31880-4_35

[41] A. J. Nebro, J. J. Durillo, J. Garcia-Nieto, C. A. Coello Coello, F. Luna, and E. Alba, "SMPSO: A new PSO-based metaheuristic for multi-objective optimization: IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making(MCDM)," pp. 66–73, https://doi.org/10.1109/MCDM.2009.4938830

[42] S. Kukkonen and J. Lampinen, "GDE3: The third Evolution Step of Generalized Differential Evolution: IEEE Congress on Evolutionary Computation," vol. 1, pp. 443–450, https://doi.org/10.1109/CEC.2005.1554717

[43] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength pareto evolutionary algorithm," TIK Report, vol. 103, ETH Zurich, Computer Engineering and Networks Laboratory, 2001, https://doi.org/10.3929/ethz-a-004284029

[44] PrO4Bake, EU project funded by EIT Food, "Optimisation of bakery processes by a computational tool together with consumer feedback to minimise ecological footprint and food waste: Making baking more efficient," Runtime: 2020 - 2021. [Online] https://www.eitfood.eu/projects/optimization-of-bakery-processes-by-a-computational-tool-together-with-consumer-feedback-to-minimize-ecological-footprint-and-food-waste-2020 (accessed: Sep. 1 2022)

[45] L. Bewoor, V. Chandra Prakash, and S. Sapkal, "Evolutionary Hybrid Particle Swarm Optimization Algorithm for Solving NP-Hard No-Wait Flow Shop Scheduling Problems," Algorithms, vol. 10, no. 4, p. 121, 2017, https://doi.org/10.3390/a10040121

[46] H. Sun, A. Jiang, D. Ge, X. Zheng, and F. Gao, "A Chance Constrained Programming Approach for No-Wait Flow Shop Scheduling Problem under the Interval-Valued Fuzzy Processing Time," Processes, vol. 9, no. 5, p. 789, 2021, https://doi.org/10.3390/pr9050789

[47] S. Swangnop, T. Duangdee, and J. Duangdee, "Design of Production Planning Process for Bakery Manufacturer," in 2019 IEEE 6th International Conference on Industrial Engineering and Applications:

ICIEA 2019 : April 12-15, 2019, Tokyo, Japan, Tokyo, Japan, 2019, pp. 178–182, https://doi.org/10.1109/IEA.2019.8714851

[48] J. Huber and H. Stuckenschmidt, "Intraday shelf replenishment decision support for perishable goods," International Journal of Production Economics, vol. 231, p. 107828, 2021, https://doi.org/10.1016/j.ijpe.2020.107828

[49] F. T. Hecker, W. B. Hussein, O. Paquet-Durand, M. A. Hussein, and T. Becker, "A case study on using evolutionary algorithms to optimize bakery production planning," Expert Systems with Applications, vol. 40, no. 17, pp. 6837–6847, 2013, https://doi.org/10.1016/j.eswa.2013.06.038

[50] C. N. Potts, D. B. Shmoys, and D. P. Williamson, "Permutation vs. non-permutation flow shop schedules," Operations Research Letters, vol. 10, no. 5, pp. 281–284, 1991, https://doi.org/10.1016/0167-6377(91)90014-G

[51] W. B. Hussein, F. Hecker, M. Mitzscherling, and T. Becker, "Computer Modelling and Simulation of Bakeries' Production Planning," International Journal of Food Engineering, vol. 5, no. 2, 2009, https://doi.org/10.2202/1556-3758.1565

[52] Rockwell Automation, "Arena Simulation Software," [Online] https://www.rockwellautomation.com/en-us/products/software/arena-simulation.html (accessed: Sep. 1 2022)

[53] J. Schaller and J. M. Valente, "Heuristics for scheduling jobs in a permutation flow shop to minimize total earliness and tardiness with unforced idle time allowed," Expert Systems with Applications, vol. 119, pp. 376–386, 2019, https://doi.org/10.1016/j.eswa.2018.11.007

[54] A. P. Piotrowski, M. J. Napiorkowski, J. J. Napiorkowski, and P. M. Rowinski, "Swarm Intelligence and Evolutionary Algorithms: Performance versus speed," Information Sciences, vol. 384, pp. 34–85, 2017, https://doi.org/10.1016/j.ins.2016.12.028

[55] S. Aalaei, H. Shahraki, A. Rowhanimanesh, and S. Eslami, "Feature selection using genetic algorithm for breast cancer diagnosis: experiment on three different datasets," Iranian Journal of Basic Medical Sciences, vol. 19, no. 5, pp. 476–482, 2016.

[56] S. M. Vieira, L. F. Mendonça, G. J. Farinha, and J. M. Sousa, "Modified binary PSO for feature selection using SVM applied to mortality prediction of septic patients," Applied Soft Computing, vol. 13, no. 8, pp. 3494–3504, 2013, https://doi.org/10.1016/j.asoc.2013.03.021

[57] L. Yi-jian, Z. Jian-ming, and W. Shu-qing, "Parameter estimation of cutting tool temperature nonlinear model using PSO algorithm," J. Zheijang Univ.-Sci. A, vol. 6, no. 10, pp. 1026–1029, 2005, https://doi.org/10.1631/jzus.2005.A1026

[58] H. Zeinoddini-Meymand, B. Vahidi, R. A. Naghizadeh, and M. Moghimi-Haji, "Optimal Surge Arrester Parameter Estimation Using a PSO-Based Multiobjective Approach," IEEE Trans. Power Delivery, vol. 28, no. 3, pp. 1758–1769, 2013, https://doi.org/10.1109/TPWRD.2013.2257880

[59] S. K. Satapathy, S. Dehuri, and A. K. Jagadev, "EEG signal classification using PSO trained RBF neural network for epilepsy identification," Informatics in Medicine Unlocked, vol. 6, pp. 1–11, 2017, https://doi.org/10.1016/j.imu.2016.12.001

[60] P. Melin, F. Olivas, O. Castillo, F. Valdez, J. Soria, and M. Valdez, "Optimal design of fuzzy classification systems using PSO with dynamic parameter adaptation through fuzzy logic," Expert Systems with Applications, vol. 40, no. 8, pp. 3196–3206, 2013, https://doi.org/10.1016/j.eswa.2012.12.033

[61] E. Alba, J. Garcia-Nieto, L. Jourdan, and E.-G. Talbi, "Gene selection in cancer classification using PSO/SVM and GA/SVM hybrid algorithms," in IEEE Congress on Evolutionary Computation, 2007. CEC 2007, Singapore, 2007, pp. 284–290. http://doi.org/10.1109/CEC.2007.4424483

[62] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: empirical results," Evolutionary computation, vol. 8, no. 2, pp. 173–195, 2000, http://doi.org/10.1162/106365600568202

[63] A. M. Fathollahi-Fard, K. Govindan, M. Hajiaghaei-Keshteli, and A. Ahmadi, "A green home health care supply chain: New modified simulated annealing algorithms," Journal of Cleaner Production, vol. 240, p. 118200, 2019, http://doi.org/10.1016/j.jclepro.2019.118200

[64] A. Singh Yadav, M. Abid, S. Bansal, S. L. Tyagi, and T. Kumar, "FIFO & LIFO in green supply chain inventory model of hazardous substance components industry with storage using simulated annealing," Adv. Math., Sci. J., vol. 9, no. 7, pp. 5127–5132, 2020, http://doi.org/10.37418/amsj.9.7.79

[65] K. Park and G. Kyung, "Optimization of total inventory cost and order fill rate in a supply chain using PSO," Int J Adv Manuf Technol, vol. 70, 9-12, pp. 1533–1541, 2014, http://doi.org/10.1007/s00170-013-5399-6

[66] K.-P. Wang, L. Huang, C.-G. Zhou, and W. Pang, "Particle swarm optimization for traveling salesman problem," in 2003 International Conference on Machine Learning and Cybernetics, Xi'an, China, 2003, pp. 1583–1585.

[67] H. Zhou, M. Song, and W. Pedrycz, "A comparative study of improved GA and PSO in solving multiple traveling salesmen problem," Applied Soft Computing, vol. 64, pp. 564–580, 2018, http://doi.org/10.1016/j.asoc.2017.12.031

[68] Y. Shuai, S. Yunfeng, and Z. Kai, "An effective method for solving multiple travelling salesman problem based on NSGA-II," Systems Science & Control Engineering, vol. 7, no. 2, pp. 108–116, 2019, http://doi.org/10.1080/21642583.2019.1674220

[69] M. Elbes, S. Alzubi, T. Kanan, A. Al-Fuqaha, and B. Hawashin, "A survey on particle swarm optimization with emphasis on engineering and network applications," Evol. Intel., vol. 12, no. 2, pp. 113–129, 2019, http://doi.org/10.1007/s12065-019-00210-z

[70] J. Wang, Y. Gao, W. Liu, A. K. Sangaiah, and H.-J. Kim, "An Improved Routing Schema with Special Clustering Using PSO Algorithm for Heterogeneous Wireless Sensor Network," Sensors (Basel, Switzerland), vol. 19, no. 3, 2019, http://doi.org/10.3390/s19030671

[71] F. M. Barboza, W. E. Medeiros, and J. M. Santana, "Customizing constraint incorporation in direct current resistivity inverse problems: A comparison among three global optimization methods," GEOPHYSICS, vol. 83, no. 6, E409-E422, 2018, http://doi.org/10.1190/geo2017-0188.1

[72] A. Jamasb, S.-H. Motavalli-Anbaran, and K. Ghasemi, "A Novel Hybrid Algorithm of Particle Swarm Optimization and Evolution Strategies for Geophysical Non-linear Inverse Problems," Pure Appl. Geophys., vol. 176, no. 4, pp. 1601–1613, 2019, http://doi.org/10.1007/s00024-018-2059-7

[73] A. A. K. Danish, A. Khan, K. Muhammad, W. Ahmad, and S. Salman, "A simulated annealing based approach for open pit mine production scheduling with stockpiling option," Resources Policy, vol. 71, p. 102016, 2021, http://doi.org/10.1016/j.resourpol.2021.102016

[74] N. Setyawan, N. Mardiyah, K. Hidayat, Nurhadi, and Z. Has, "Object Detection of Omnidirectional Vision Using PSO-Neural Network for Soccer Robot," in EECSI 2018: Proceedings : 2018 5th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI 2018) : 16-18 October 2018, Malang, Indonesia, Malang, Indonesia, 2018, pp. 117–121.

[75] A. S. Ansari et al., "Detection of Pancreatic Cancer in CT Scan Images Using PSO SVM and Image Processing," BioMed research international, vol. 2022, p. 8544337, 2022, http://doi.org/10.1155/2022/8544337

[76] S. Pasupuleti and R. Battiti, "The gregarious particle swarm optimizer (G-PSO)," in Proceedings of the 8th annual conference on Genetic and evolutionary computation - GECCO '06, New York, New York, USA, 2006.

[77] Z. Lian, X. Gu, and B. Jiao, "A novel particle swarm optimization algorithm for permutation flow-shop scheduling to minimize makespan☆," Chaos, Solitons & Fractals, vol. 35, no. 5, pp. 851–861, 2008, http://doi.org/10.1016/j.chaos.2006.05.082

[78] H. Fan, "A modification to particle swarm optimization algorithm," Engineering Computations, vol. 19, no. 8, pp. 970–989, 2002, http://doi.org/10.1108/02644400210450378

[79] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600), http://doi.org/10.1109/CEC.2002.1004493

[80] X. H. Shi, Y. C. Liang, H. P. Lee, C. Lu, and L. M. Wang, "An improved GA and a novel PSO-GA-based hybrid algorithm," Information Processing Letters, vol. 93, no. 5, pp. 255–261, 2005, http://doi.org/10.1016/j.ipl.2004.11.003

[81] F. Rezaei and H. R. Safavi, "GuASPSO: a new approach to hold a better exploration–exploitation balance in PSO algorithm," Soft Comput, vol. 24, no. 7, pp. 4855–4875, 2020, http://doi.org/10.1007/s00500-019-04240-8

[82] S. Verma, M. Pant, and V. Snasel, "A Comprehensive Review on NSGA-II for Multi-Objective Combinatorial Optimization Problems," IEEE Access, vol. 9, pp. 57757–57791, 2021, http://doi.org/10.1109/ACCESS.2021.3070634

[83] W. Tan, X. Yuan, Y. Yang, and L. Wu, "Multi-objective casting production scheduling problem by a neighborhood structure enhanced discrete NSGA-II: an application from real-world workshop," Soft Comput, vol. 26, no. 17, pp. 8911–8928, 2022, http://doi.org/10.1007/s00500-021-06697-y

[84] M. Hemmati Far, H. Haleh, and A. Saghaei, "A fuzzy bi-objective flexible cell scheduling optimization model under green and energy-efficient strategy using Pareto-based algorithms: SATPSPGA, SANRGA, and NSGA-II," Int J Adv Manuf Technol, vol. 105, no. 9, pp. 3853–3879, 2019, http://doi.org/10.1007/s00170-019-03797-w

[85] R. K. Chakrabortty, A. Abbasi, and M. J. Ryan, "A Risk Assessment Framework for Scheduling Projects With Resource and Duration Uncertainties," IEEE Trans. Eng. Manage., vol. 69, no. 5, pp. 1917–1931, 2022, http://doi.org/10.1109/TEM.2019.2943161

[86] M. A. Isah and B.-S. Kim, "Integrating Schedule Risk Analysis with Multi-Skilled Resource Scheduling to Improve Resource-Constrained Project Scheduling Problems," Applied Sciences, vol. 11, no. 2, p. 650, 2021, http://doi.org/10.3390/app11020650

[87] A. Maravas and J.-P. Pantouvakis, "Project cash flow analysis in the presence of uncertainty in activity duration and cost," International Journal of Project Management, vol. 30, no. 3, pp. 374–384, 2012, http://doi.org/10.1016/j.ijproman.2011.08.005

[88] A. Öztaş and Ö. Ökmen, "Judgmental risk analysis process development in construction projects," Building and Environment, vol. 40, no. 9, pp. 1244–1254, 2005, http://doi.org/10.1016/j.buildenv.2004.10.013

[89] S. Huang, G. Li, E. Ben-Awuah, B. O. Afum, and N. Hu, "A Stochastic Mixed Integer Programming Framework for Underground Mining Production Scheduling Optimization Considering Grade Uncertainty," IEEE Access, vol. 8, pp. 24495–24505, 2020, http://doi.org/10.1109/ACCESS.2020.2970480

[90] H. Moradi and S. Shadrokh, "A robust reliability-based scheduling for the maintenance activities during planned shutdown under uncertainty of activity duration," Computers & Chemical Engineering, vol. 130, p. 106562, 2019, http://doi.org/10.1016/j.compchemeng.2019.106562

[91] D. Petrovic and A. Duenas, "A fuzzy logic based production scheduling/rescheduling in the presence of uncertain disruptions," Fuzzy Sets and Systems, vol. 157, no. 16, pp. 2273–2285, 2006, http://doi.org/10.1016/j.fss.2006.04.009

Appendices

Annex A. Bakery production data (BK15)

Table A1. List of machines

ID	Name	Category
1	Kneader 1	Rotating kneader
2	Kneader 2	Lifting kneader
4	Divider	Divider
7	Cutter	Cutter
8	Shaper	Shaper
9	Proving chamber	Proving chamber
10	Oven A1	Stone oven
11	Oven A2	Stone oven
12	Oven A3	Stone oven
13	Oven B	Baking chamber

Table A2. List of employees who directly operate manual tasks and their working schedule. Shift time is presented corresponding to production run time in minutes.

Name	Shift start [min]	Shift end [min]
Employee 1	0	540
Employee 2	0	540
Employee 3	260	700

Table A3. List of products and their process details

Product group	Product ID	Bowl time	Product name	Batches	Stage name	Duration (PT)	Machines / Employees (M/E)
1	1	0	Product 1	1	Preparation	5	Manual
1	1	0	Product 1A	1	Kneading	21	Lifting kneader
1	1	0	Product 1A	1	Dividing	16	Divider
1	1	0	Product 1A	1	Proofing	105	
1	1	0	Product 1A	1	Load Baking	14	Manual
1	1	0	Product 1A	1	Baking	56	Stone oven
1	1	0	Product 1A	1	Unload baking	17	Manual
1	1	0	Product 1A	1	Cooling	1	Manual
1	2	82	Product 1A-2	1	Baking	56	Stone oven
1	3	82	Product 1A-3	1	Baking	56	Stone oven
2	1	0	Product 2A	1	Preparation	9	Manual
2	1	0	Product 2A	1	Load kneader	2	Manual
2	1	0	Product 2A	1	Kneading	18	Rotating kneader
2	1	0	Product 2A	1	Kneading disch.	2	Manual
2	1	0	Product 2A	1	Dividing	9	Manual
2	1	0	Product 2A	1	Dough rest	25	
2	1	0	Product 2A	1	Load Baking	3	Manual
2	1	0	Product 2A	1	Baking	50	Baking chamber
2	1	0	Product 2A	1	Discharge baking	3	Manual
2	1	0	Product 2A	1	Cooling	25	
2	1	0	Product 2A	1	Unmolding	4	Manual
2	2	40	Product 2B	1	Dividing	10	Manual
2	2	0	Product 2B	1	Dough rest	30	
2	2	0	Product 2B	1	Proofing	80	
2	2	0	Product 2B	1	Load Baking	2	Manual
2	2	0	Product 2B	1	Baking	60	Baking chamber
2	2	0	Product 2B	1	Discharge baking	2	Manual
2	2	0	Product 2B	1	Cooling	36	
2	2	0	Product 2B	1	Unmolding	7	Manual
2	3	50	Product 2C	1	Dividing	15	Manual
2	3	0	Product 2C	1	Dough rest	45	
2	3	0	Product 2C	1	Proofing	115	
2	3	0	Product 2C	1	Load Baking	3	Manual
2	3	0	Product 2C	1	Baking	58	Baking chamber
2	3	0	Product 2C	1	Discharge baking	2	Manual
2	3	0	Product 2C	1	Cooling	30	
2	3	0	Product 2C	1	Unmolding	3	Manual
3	1	0	Product 3A	1	Preparation	4	Manual
3	1	0	Product 3A	1	Kneading	7	Lifting kneader
3	1	0	Product 3A	1	Discharge kneading	2	Manual
3	1	0	Product 3A	1	Load dividing	18	Divider
3	1	0	Product 3A	1	Dividing	13	Divider
3	1	0	Product 3A	1	Dough rest	32	

Table A3. List of products and their process details (continues)

Product group	Product ID	Bowl time	Product name	Batches	Stage name	Duration (PT)	Machines / Employees (M/E)
3	1	0	Product 3A	1	Load Baking	11	Manual
3	1	0	Product 3A	1	Baking	54	Stone oven
4	1	0	Product 4	1	Preparation	7	Manual
4	1	0	Product 4	1	Kneading	18	Rotating kneader
4	1	0	Product 4	1	Dividing	11	Divider
4	1	0	Product 4	1	Cutting-forming	35	Cutting Machine
4	1	0	Product 4	1	Transfer to Proofing	1	Manual
4	1	0	Product 4	1	Proofing	120	
4	1	0	Product 4	1	Load baking	3	Manual
4	1	0	Product 4	1	Baking	70	Baking chamber
4	1	0	Product 4	1	Unload baking	2	Manual
4	1	0	Product 4	1	Cooling	31	
4	1	0	Product 4	1	packaging	10	Manual
5	1	0	Product 5	1	Preparation	11	Manual
5	1	0	Product 5	1	Kneading	10	Rotating kneader
5	1	0	Product 5	1	Dough rest	35	
5	1	0	Product 5	1	Dividing	2	Manual
5	1	0	Product 5	1	Shaping	2	Shaper
5	1	0	Product 5	1	Transfer to Proofing	1	
5	1	0	Product 5	1	Proofing	100	
5	1	0	Product 5	1	Load Baking	6	
5	1	0	Product 5	1	Baking	42	Stone oven
5	1	0	Product 5	1	Unload baking	3	Manual
5	1	0	Product 5	1	Cooling	32	
5	1	0	Product 5	1	packaging	3	Manual
6	1	0	Product 6	2	Preparation	5	Manual
6	1	0	Product 6	2	Kneading	9	Rotating kneader
6	1	0	Product 6	2	Dough rest	26	
6	1	0	Product 6	2	Dividing	10	Divider
6	1	0	Product 6	2	Freezing	1	
7	1	0	Product 7	1	Preparation	5	Manual
7	1	0	Product 7	1	Kneading	10	Rotating kneader
7	1	0	Product 7	1	Dough rest	21	
7	1	0	Product 7	1	Cutting-forming	48	Cutting Machine
7	1	0	Product 7	1	Proofing	105	
7	1	0	Product 7	1	Load Baking	4	Manual
7	1	0	Product 7	1	Baking	25	Stone oven
7	1	0	Product 7	1	Unload baking	3	Manual
7	1	0	Product 7	1	Cooling	20	
8	1	0	Product 8	1	Preparation	6	Manual
8	1	0	Product 8	1	Kneading	23	Rotating kneader
8	1	0	Product 8	1	Cutting-forming	30	Cutting Machine
8	1	0	Product 8	1	Dough rest	36	
8	1	0	Product 8	1	Load baking	10	Manual

Table A3. List of products and their process details (continues)

Product group	Product ID	Bowl time	Product name	Batches	Stage name	Duration (PT)	Machines / Employees (M/E)
8	1	0	Product 8	1	Baking	15	Stone oven
8	1	0	Product 8	1	Unload baking	10	Manual
9	1	0	Product 9	1	Preparation	4	Manual
9	1	0	Product 9	1	Kneading	11	Rotating kneader
9	1	0	Product 9	1	Dividing	12	Divider
9	1	0	Product 9	1	Dough rest	19	
9	1	0	Product 9	1	Shaping	5	Manual
9	1	0	Product 9	1	Proofing	120	
9	1	0	Product 9	1	Load Baking	3	Manual
9	1	0	Product 9	1	Baking	42	Stone oven
9	1	0	Product 9	1	Unload baking	4	Manual
9	1	0	Product 9	1	Cold-storage	2	Manual
10	1	0	Product 10	1	Preparation	4	Manual
10	1	0	Product 10	1	Kneading	16	Rotating kneader
10	1	0	Product 10	1	Cutting-forming	56	Cutting Machine
10	1	0	Product 10	1	Dough rest	24	
10	1	0	Product 10	1	Shaping	12	Manual
10	1	0	Product 10	1	Proofing	80	
10	1	0	Product 10	1	Load baking	5	Manual
10	1	0	Product 10	1	Baking	42	Stone oven
10	1	0	Product 10	1	Unload baking	5	Manual
10	1	0	Product 10	1	Cooling	80	
10	1	0	Product 10	1	packaging	5	Mnaual
11	1	0	Product 11	1	Preparation	6	Manual
11	1	0	Product 11	1	Kneading	23	Rotating kneader
11	1	0	Product 11	1	Cutting-forming	50	Cutting Machine
11	1	0	Product 11	1	Tr. To Cold-storage	1	Manual
12	1	0	Product 12A	1	Defrost	80	
12	1	0	Product 12A	1	Load baking	3	Manual
12	1	0	Product 12A	1	Baking	18	Stone oven
12	1	0	Product 12A	1	Unload baking	3	Manual
12	2	83	Product 12B	1	Load baking	3	Manual
12	2	0	Product 12B	1	Baking	18	Stone oven
12	2	0	Product 12B	1	Unload baking	3	Manual
12	3	86	Product 12C	1	Load baking	3	Manual
12	3	0	Product 12C	1	Baking	18	Stone oven
12	3	0	Product 12C	1	Unload baking	3	Manual
13	1	0	Product 13	1	Preparation	5	Manual
13	1	0	Product 13	1	Load kneader	2	Manual
13	1	0	Product 13	1	Kneading	14	Rotating kneader
13	1	0	Product 13	1	Kneading disch.	2	Manual
13	1	0	Product 13	1	Dividing	4	Manual
13	1	0	Product 13	1	Dough rest	30	
13	1	0	Product 13	1	Load Baking	1	Manual

Table A3. List of products and their process details (continues)

Product group	Product ID	Bowl time	Product name	Batches	Stage name	Duration (PT)	Machines / Employees (M/E)
13	1	0	Product 13	1	Baking	45	Baking chamber
13	1	0	Product 13	1	Discharge baking	2	Manual
13	1	0	Product 13	1	Cooling	20	
13	1	0	Product 13	1	Unmolding	4	Manual
14	1	0	Product 14	1	Preparation	12	Manual
14	1	0	Product 14	1	Load kneader	2	Manual
14	1	0	Product 14	1	Kneading	10	Rotating kneader
14	1	0	Product 14	1	Kneading disch.	2	Manual
14	1	0	Product 14	1	Dividing	7	Manual
14	1	0	Product 14	1	Dough rest	46	
14	1	0	Product 14	1	Load Baking	2	Manual
14	1	0	Product 14	1	Baking	48	Baking chamber
14	1	0	Product 14	1	Discharge baking	2	Manual
14	1	0	Product 14	1	Cooling	28	
14	1	0	Product 14	1	Unmolding	2	Manual
15	1	0	Product 15	1	Preparation	3	Manual
15	1	0	Product 15	1	Load kneader	2	Manual
15	1	0	Product 15	1	Kneading	15	Rotating kneader
15	1	0	Product 15	1	Kneading disch.	2	Manual
15	1	0	Product 15	1	Dividing	5	Manual
15	1	0	Product 15	1	Dough rest	20	
15	1	0	Product 15	1	Shaping	15	Shaper
15	1	0	Product 15	1	Proofing	90	
15	1	0	Product 15	1	Load Baking	2	Manual
15	1	0	Product 15	1	Baking	38	Stone oven
15	1	0	Product 15	1	Discharge baking	2	Manual
15	1	0	Product 15	1	Cooling	28	
15	1	0	Product 15	1	Unmolding	2	Manual

Curriculum vitae

Md Majharul Islam Babor

Stuttgart, 70599

+4915908452678

majharul.babor@gmail.com

Nationality: Bangladesh
Date of birth: 10.07.1990

Experience

07.2020 – present

Research Associate • University of Hohenheim

- Data analysis for the PrO4Bake project
- Development of flow shop model and optimization algorithms
- Analysis of production efficiency of real manufacturing lines

Education

Master of Science, 2017-2020

University of Hohenheim, Stuttgart, Germany

Bioeconomy

Bachelor of Science, 2009-2013

Shahjalal University of Science & Technology, Sylhet, Bangladesh

Food Engineering and Tea Technology

Computer Skill

Programming language: MATLAB, Python, C

Language skill

English (Professional proficiency)

German (Basic)

Publications

Modeling and Optimization of Bakery Production Scheduling to Minimize Makespan and Oven Idle Time
M. Babor, O. Paquet-Durand, R. Kohlus and B. Hitzmann
Sci Rep 13, 235 (2023). https://doi.org/10.1038/s41598-022-26866-9

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

Application of Nature-Inspired Multi-Objective Optimization Algorithms to Improve the Bakery Production Efficiency
M. Babor and B. Hitzmann
Engineering Proceeding, 2022, https://doi.org/10.3390/ECP2022-12630

Optimization of No-Wait Flowshop Scheduling Problem in Bakery Production with Modified PSO, NEH and SA
M. Babor, J. Senge, C.M. Rosell, D. Rodrigo and B. Hitzmann
Processes, 2021, https://doi.org/10.3390/pr9112044

Traceability of Sweeteners in Soy Yogurt Using Linear Discriminant Analysis of Physicochemical and Sensory Parameters
M.R. Rana, **M. Babor** and A.A. Sabuz
J. of Agr. and Food Res., 2021, https://doi.org/10.1016/j.jafr.2021.100155

Model-Based Calibration of a Gas Sensor Array for On-line Monitoring of Ethanol Concentration in *Saccharomyces Cerevisiae* Batch Cultivation
A. Yousefi-Darani, **M. Babor**, O. Paquet-Durand and B. Hitzmann
Biosystems Eng., 2020, https://doi.org/10.1016/j.biosystemseng.2020.08.004

Other skills

Process optimization

Production scheduling model

Data analysis

Machine learning

Chemometrics

Modelling, monitoring of bioprocess

Stuttgart, 20/01/2023

Place, Date

Signature of the candidate

Declaration in lieu of an oath on independent work

This is a courtesy translation. Only the German version is legally binding. If there are any differences in the wording, interpretation, or meaning of the German and English versions, the German version shall prevail.

Annex 3

Declaration in lieu of an oath on independent work

according to Sec. 18(3) sentence 5 of the University of Hohenheim's Doctoral Regulations for the Faculties of Agricultural Sciences, Natural Sciences, and Business, Economics and Social Sciences

1. The dissertation submitted on the topic
Application of Nature-Inspired Optimization Algorithms to Improve the

Production Efficiency of Small and Medium-Sized Bakeries

is work done independently by me.

2. I only used the sources and aids listed and did not make use of any impermissible assistance from third parties. In particular, I marked all content taken word-for-word or paraphrased from other works.

3. I did not use the assistance of a commercial doctoral placement or advising agency.

4. I am aware of the importance of the declaration in lieu of oath and the criminal consequences of false or incomplete declarations in lieu of oath.

I confirm that the declaration above is correct. I declare in lieu of oath that I have declared only the truth to the best of my knowledge and have not omitted anything.

Stuttgart, 20/01/2023

MI Babor

Place, Date

Signature

The End