Journal of Applied Engineering and Technological Science
Vol 6(2) 2025: 1280-1296

FLEXIBLE JOB SHOP SCHEDULING OPTIMIZATION USING GENETIC ALGORITHM FOR HANDLING DYNAMIC FACTORS

Masmur Tarigan[1*], Ford Lumban Gaol[2], Alexander AS Gunawan[3], Widodo Budiharto[4]
Computer Science Department, Esa Unggul University, Jakarta, Indonesia[1]
Department of Doctor of Computer Science, BINUS - Graduate Program, Bina Nusantara University, Jakarta, Indonesia[2]
Computer Science Department, School of Computer Science, Bina Nusantara University, Jakarta, Indonesia[34]
masmur.tarigan@esaunggul.ac.id

Received: 30 July 2024, Revised: 24 February 2025, Accepted: 18 March 2025
**Corresponding Author*

ABSTRACT
This research introduces the Genetic Adaptive Scheduling System (GASS), a novel framework designed to optimize scheduling in Flexible Job Shop Scheduling Problems (FJSP). Due to its complexity, FJSP presents significant challenges stemming from machine flexibility, dynamic routing, and operation precedence constraints. GASS addresses these challenges by incorporating real-time, dynamic data, enabling the system to adapt to machine downtimes, fluctuating job priorities, and process variability. Leveraging advanced genetic algorithm techniques, GASS integrates enhanced mutation and selection processes that dynamically adjust setup times, prioritize urgent tasks, and balance machine workloads to minimize makespan effectively. Empirical results demonstrate that GASS achieves up to a 45.3% reduction in makespan within the flexible packaging industry, showcasing its ability to enhance scheduling efficiency and adaptability. The research highlights the system's scalability and potential applicability across diverse industries, including printing, electronics, pharmaceuticals, and food manufacturing, where operational flexibility and efficiency are critical. By bridging existing gaps and integrating real-time constraints into scheduling models, GASS provides practical solutions for modern manufacturing environments. The findings contribute to the advancement of optimization techniques in FJSP, offering valuable insights for researchers and practitioners seeking efficient, scalable, and adaptive scheduling systems.

Keywords: *Flexible Job Shop Scheduling, Genetic Adaptive Scheduling System, Dynamic Scheduling Optimization, Manufacturing Process Efficiency, Real-Time Production Scheduling.*

1. Introduction

Due to its relevance to efficient resource and production time allocation, Flexible Job Shop Scheduling (FJSS) has come to the forefront of many modern manufacturing research efforts. The FJSS problem belongs to optimization problems and is an NP-hard problem that becomes more complex due to the consideration of dynamic factors in real production environments (Psarommatis, 2020; Shao, 2021; Türkyılmaz, 2020; F. Zhang, 2021, 2023; Zhuang et al., 2019). As a powerful metaheuristic approach, genetic algorithms (GA) have been successfully used to address this issue. The VSH developed a mathematical model for FJSS, whose objective function is the maximization of total profit, which accounts for the cost of raw material and selling price, as well as the changing demand throughout each period. This methodology effectively identifies the economical production quantity on different machines to fulfill customer demand at each period (Awad, 2021; Luo, 2020; Xu, 2020).

Moreover, a comprehensive scheduling model would incorporate dynamic factors—most importantly, setup times, job priorities, and machine downtimes that can change throughout an operation. Consequently, the scheduling system has the flexibility to accommodate fluctuating production conditions in real time, resulting in enhanced operational efficiency (Y. Li et al., 2022). In this context, GA-based approaches have demonstrated superior schedule quality, taking into account the actual dynamics of the production environment.

The application of GA for FJSS scheduling taking dynamic factors into account has been investigated in few industrial case studies under the heading of practical implementation. It has been shown that it enhances the efficiency of production and makes it easier to adapt to changes in demand and operating conditions. Hence, the use of GA in FJSS scheduling is a useful approach to solve complex problems in dynamic manufacturing environment.

There is a great variety of problems, ranging from simple to very complex ones but the Flexible Job Shop Problem (FJSP) in particular - or more general solutions named General manufacturing scheduling problem GSTP. FJSP entails more complex task scheduling because of the necessity for processing different products using various machines and with distinct operating sequences. Efficient scheduling is also very important to make the most of your resources and minimize lead times and production overall. Fig. In this example given in Fig. 1, the FJSP scheduling problem is a real-world challenge of a flexible packaging manufacturing environment where each job presents an intricate structure and demands several operations to be conducted by undistinguishable machines available at the production shop floor.

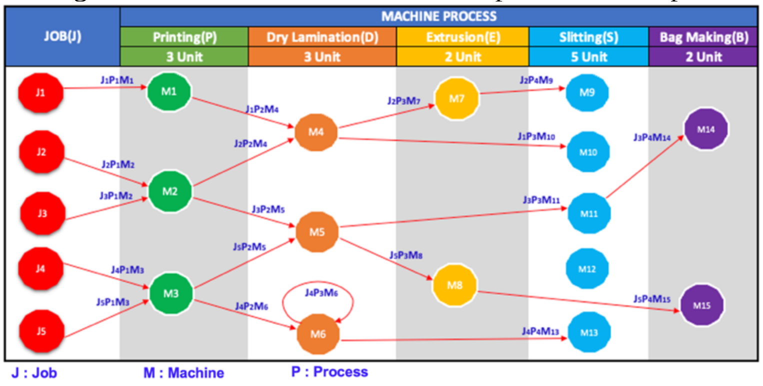


Fig. 1. FJSP in Flexible Packaging Manufacture(Tarigan et al., 2023)

In particular, genetic algorithms (a field of evolutionary optimization techniques) have been widely adopted for complex scheduling problems. It uses selection, crossover, and mutation genetic processes to emulate natural evolutionary processes by searching for the best solution. A search is an acceptable approach for FJSP as the problem of finding a scheduling solution involves a large and complex search space.

The main aim of this thesis is to formulate a mathematical model for scheduling the FJSP (Flexible Job Shop Problem) by using Genetic Algorithms. This model is our attempt to put in place an automated queuing mechanism implemented with real data picked up from the manufacturing industry. Automated queuing ensures that jobs will be processed efficiently, taking into account priority and the product category or product length in running meters. The approach aims to minimize the total processing time (TP), which is referred to using makespan, to achieve optimal schedules and improve efficiency in manufacturing.

Challenge: The most significant challenge of this research is to design such a scheduling model that could assimilate the real-time manufacturing data into scheduling models and can automate queue maintenance by prioritizing them on given parameters like priority level, product type, or runtimes. The approach uses a genetic algorithm to identify the best plan, which minimizes makespan of the maximum task. We are coming up with good solutions for navigating the complexities of FJSP.

The goal of this query is to develop a concrete scheduling model for the Flexible Job Shop Problem (FJSP) that can well incorporate trusted data coming from practice in manufacturing. We use evolutionary algorithms to Optimize the Scheduling of Flexible Job Shop problems (FJSP). Utilize an extremely successful scheduling strategy for impeccable time management of the entire process. The best practice is to establish a rigorous testing and validation procedure based on real data in order to evaluate the utility of your models for working in practical production scenarios.

This is an area of production scheduling and optimization that we anticipate will be materially affected by the research, in particular, building a new mathematical model for FJSP scheduling, including real data from the manufacturing industry. The research literature, to this end, seeks primarily to illustrate the potential of genetic algorithms in modeling complex and dynamic scheduling problems. The industrial space is one sector that can implement realistic strategies to increase efficiency and productivity. So with that, the researchers have reviewed in this study how to use artificial intelligence technology for better scheduling across the Industry 4.0 era (Ghaleb, 2020; Y. Li, 2020; S. Zhang, 2021a).

Genetic Algorithms (GAs) have emerged as a powerful optimization tool due to their ability to navigate large and complex search spaces effectively. Compared to other techniques like tabu search or simulated annealing, GAs offer enhanced adaptability and scalability, making them suitable for addressing the unique challenges of FJSP. However, existing literature reveals gaps in real-time adaptability and integration of dynamic data in FJSP solutions. This study aims to bridge these gaps by developing a GASS model that incorporates real-world constraints and dynamic scheduling requirements.

2. Literature Review

Flexible Job Shop Scheduling (FJSP)

The Flexible Job Shop Scheduling Problem (FJSP) is a modified version of the traditional job shop scheduling problem. It allows for more flexibility in choosing machines for each operation, making it more suitable for modern production conditions. FJSP can handle a variety of tasks and machines with varying capacities. (Fan et al., 2022)described FJSP as one of the most intricate scheduling problems due to the numerous variables and constraints involved. (Huang & Yang, 2019) highlighted that FJSP presents significant optimization challenges because of its extensive machine selection options (Lei, 2024; Tian, 2023; K. Zhu, 2023).

Utilizing Genetic Algorithms for Optimizing Scheduling

The genetic algorithm process depicted in Fig. 2 begins with the creation of an initial population. The chromosomes in this population represent a set of people or potential solutions. We assess each participant using a fitness function, based on either makespan or total completion time to judge the quality of the resulting solution (F. Zhang, 2022; S. Zhang, 2021b; X. Zhang et al., 2022). Next, we conduct a selection process to identify the most exceptional individuals who will act as parents in the reproductive process. Techniques like tournament selection achieve this by increasing the likelihood of choosing individuals with higher levels of fitness. Crossover, also known as recombination, is a genetic process that involves combining the genetic material of two parental individuals to create a new individual. The process involves the exchange of specific genes between the parents. On the other hand, mutation is a process that alters one or more genes in an individual's chromosomes. The former brings new variables in play which prevent the user from going directly to a local maximum known solution. After crossover and mutation, the newly generated individuals undergo an evaluation using a fitness function. Hence, the selection process for a population of tomorrow means keeping those who are best. This process repeats or creates new solutions until the termination criterion is satisfied (e.g., a maximum number of iterations, an acceptable best fitness value, and no improvement in fitness over fewer consecutive generations). This finally gives us the best solution to the Job Scheduling Problem (Tarigan et al., 2023).

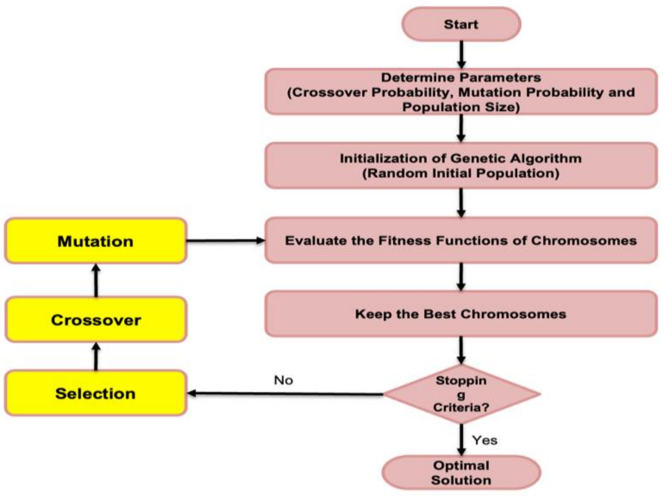

Start
Determine Parameters (Crossover Probability, Mutation Probability and Population Size)
Initialization of Genetic Algorithm (Random Initial Population)
Evaluate the Fitness Functions of Chromosomes
Keep the Best Chromosomes
Stoppin g Criteria?
No
Yes
Optimal Solution
Selection
Crossover
Mutation

Fig. 2. Flow Diagram of Genetic Algorithm

Genetic Algorithms (GA) are commonly used evolution optimization technique in the area of Scheduling. A genetic algorithm (GA) applies the mechanism of natural selection to evolve solutions through generations, utilizing operations like, but limited to - selection, crossover, and mutation. (Zhuang et al., 2019) indicate that in the case of complex scheduling problems like FJSP, the Genetic Algorithm(GA) is an effective approach due to its ability for a wide exploration search space and handle multiple constraints, etc. (Singh & Sundar, 2019) have discussed that the GA parameters such as population size, mutation rate, and crossover rate play an important role in better performance of genetic algorithm to solve scheduling problems.

Manufacturing systems automate queuing.

It is a system that organizes and regulates the order in which jobs are processed, which is influenced by factors such as priority, product category, and the required time to produce a running meter of products (Abderrahim, 2020; Qin, 2021; Samsonov, 2021; Zhou, 2020). Automated queuing systems see to it that jobs are done efficiently and within expected timeframes, hence, minimizing the time client waits and maximizing production. For instance, their research, (Hong & Chien, 2020) found that automatic queuing integrated system into the production system improves production operation and reduces machine downtimes. Automated queuing is ideal when one is managing a dynamic production environment, facing consistent changes in the types of jobs to be done depending on market demands. (Rossit et al., 2019), for example, assert that automated queuing has been useful in dynamic environments.

Authentic data from the manufacturing sector

The validation of such scheduling models needs to use authentic data taken from the industry. Data includes specifics such as job type, processing time for the jobs, and level of cuproproteins to name a few. It is stressed by (Baykasoğlu et al., 2020) that to increase the accuracy and reliability of solutions in real production cases, hence realistic data-based scheduling models should be developed. To this end, (K. Li et al., 2021) claim that the inclusion of realistic data increases model realism and practical applicability as it correctly describes all impediments and lead times suffered in daily production.

State of The Art

Genetic algorithms and other techniques have been used by researchers, many studies were performed to enhance the efficiency of FJSP scheduling. Designed genetic algorithms integrated with real-time data for job shop scheduling (Y. Li et al., 2022; Meng, 2023; Soares, 2020; Xie, 2023), and reported an advance in the performance of gene expression programmers by addressing various medical imaging problems more quickly. In (Sana et al., 2019), genetic algorithms are used for dynamic job shop scheduling concerning the algorithmized adaptation to an alteration, which enables it to change on production conditions. An improved genetic algorithm for optimizing the makespan in job shop scheduling was successfully applied to reduce the total processing time (Umam et al., 2021).

Another study was conducted by (Zheng et al. (2022), illustrating a remarkable improvement in the quality of scheduling using a data-driven genetic algorithm that exploits real-time information from the production environment. Open access: H. Zhu et al. (2019) highlight that it is essential to integrate real-time data into genetic algorithm methodologies for the final schedule to be accurate and efficient. In their research, Luo et al. (2020) presented a workshop scheduling problem and solved it by using an improved genetic algorithm. The results indicated that their method outperformed prior works. A hybrid method that combines the genetic algorithm with other optimization techniques to improve production schedules has been proposed (Wang & Zhu, 2021). Their method exhibited promising results across all experimental settings.

The paper of (Zhao & Zhang, 2021) extensively focuses on the application of artificial intelligence in flexible job scheduling. Their study provides insights into present trends and challenges within this area. The present paper provides an efficient and feasible approach for the manufacturing sector, the considered genetic algorithms technique serves as one of the advanced technology applications in scheduling too.

3. Research Methods

In this work, we use a quantitative research method to develop and validate an algorithmic framework for solving Flexible Job-Shop Scheduling Problems (FJSP) based on genetic algorithms. Considering the nature of this study, we used a quantitative approach because it can provide accurate measurements and allow us to statistically analyze those data so that reliable conclusions will be capable of being drawn.

Data Source

This study used secondary data from Internal Company Records: This is a type of past production operations records. This information consists of how long the process takes, the productivity of machines, and output. Real-time information - e.g., processing times, downtimes, or machinery maintenance Interviews - We will conduct interviews with plant managers and machine operators so we can truly understand the day to day challenges that onsite personnel experience, in addition to what preferences they have for scheduling. Feedback Surveys can help collect feedback from employees on the work shift they want to perform or whether they are happy with it.

The firm's information system collects data in electronic format using a data gathering method depicted in Fig. 3. We then merge this data into a centralized database for analysis. We subject the raw data to processing and verification procedures to ensure its accuracy and comprehensiveness before using it in simulations or scheduling models.

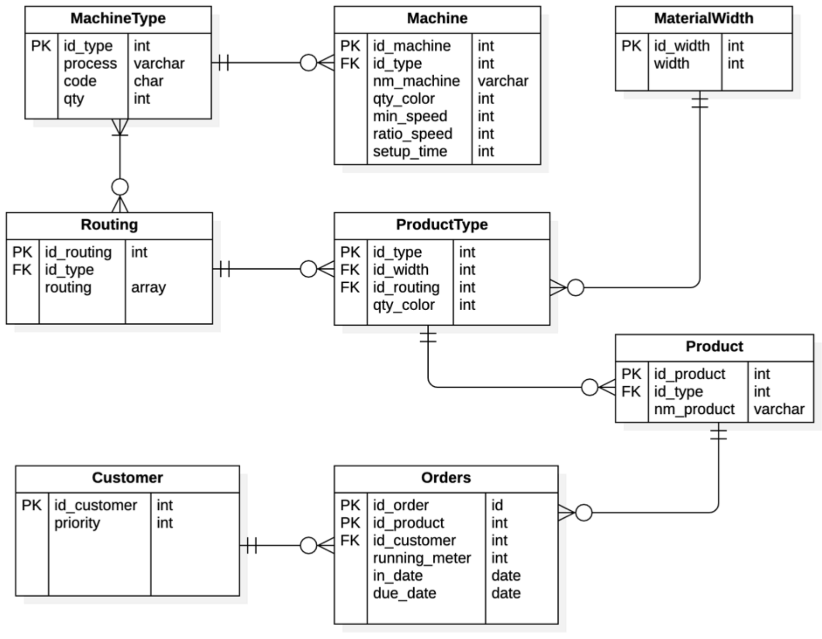
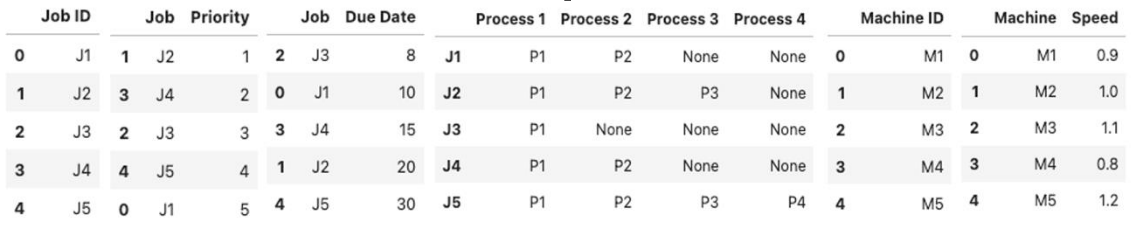


Fig. 3. Entity-Relationship Diagram Data Source

The provided diagram depicts the database structure of a production scheduling system, showcasing multiple primary tables and the links that exist between them. The *MachineType* table contains data regarding the precise type of machine utilized in the production process. The machine table, which provides more detailed information about each individual machine, links to this table. The variable *MaterialWidth* represents the width of the material utilized and is associated with *ProductType*, which represents the product's type, including both the material width and the number of colors. The routing table contains the manufacturing process's sequential order and is linked to machine types and orders. The Orders table stores customer order details, such as product, routing, and machine used. The customer entity maintains customer data and their priorities, which are associated with orders. *ProductType* is associated with Product, which contains precise product information. These linkages facilitate streamlined data integration and administration in the production scheduling system, ensuring effective control of all aspects of production, including machines, materials, and client orders.

Prior to utilization in a genetic algorithm, the data must undergo preprocessing to transform a non-uniform format into a standardized format (Fig. 4). This encompasses the process of normalizing data, imputing missing values, and converting categorical data into a numerical format suitable for mathematical computations.



	Job ID
0	J1
1	J2
2	J3
3	J4
4	J5

	Job	Priority
1	J2	1
3	J4	2
2	J3	3
4	J5	4
0	J1	5

	Job	Due Date
2	J3	8
0	J1	10
3	J4	15
1	J2	20
4	J5	30

	Process 1	Process 2	Process 3	Process 4
J1	P1	P2	None	None
J2	P1	P2	P3	None
J3	P1	None	None	None
J4	P1	P2	None	None
J5	P1	P2	P3	P4

	Machine ID
0	M1
1	M2
2	M3
3	M4
4	M5

	Machine	Speed
0	M1	0.9
1	M2	1.0
2	M3	1.1
3	M4	0.8
4	M5	1.2

Fig. 4. Standard Format Data for FJSP Scheduling

Mathematical model development

The problem's formulation

Flexible job shop scheduling (FJSP) challenges arise when a manufacturing company needs to schedule different types of work on many machines efficiently. The primary objective is to decrease the makespan which refers to the overall duration needed to finish all the tasks. The following mathematical model provides a concise representation of the fundamental framework of the FJSP problem:

Decision Variable:

x_{jpi}	:	x_{jpi} =1 if process p of job j is assigned to machine i, and $x_{jpi} = 0$ otherwise
S_{jpi}	:	Start time of process p of job j on machine i
C_{jpi}	:	Process completion time p of job j on machine i
C_{max}	:	Maximum completion time for all processes p for all jobs j
$y_{jj'i}$	:	$y_{jj'i} = 1$ if job j' processed after job j on machine i, and $y_{jj'i} = 0$ otherwise
d_j	:	Due date of job j

Objective Function:

$min\ C_{max}$	:	Minimum Makespan (Find the most optimal completion time for all work)
C_{max}	=	$max_{i\in M, j\in J, p\in P}\ C_{jpi}$

Constraints:

$$\sum_{i\in M_{jp}} x_{jpi} = 1\ \ \forall j \in J, \forall p \in P_j \tag{1}$$

where every process of every job must be processed exactly once on the appropriate machine.

$$\sum_{j\in J}\sum_{p\in P} x_{ijp} \le 1\ \ \forall i \in M \tag{2}$$

where each machine can only process one job for a particular process until it is finished.

$$C_{jpi} = S_{jpi} + p_{jp} \cdot v_i \cdot x_{jpi}\ \ \forall i \in M, \forall j \in J, \forall p \in P_j \tag{3}$$

where process completion time p of job j on machine i.

$$C_{max} \ge C_{jpi}\ \forall i \in M, \forall j \in J, \forall p \in P_j \tag{4}$$

where process completion time p of job j on machine i smaller equals to maximum completion time for all processes p for all jobs j.

$$x_{jpi} \in \{0,1\} \ \forall i \in M, \forall j \in J, \forall p \in P_j \tag{5}$$

where decision variable x_{iin} is binary numbers.

$$s_{jj'i} = \begin{cases} t_{ji} & if\ job\ type\ of\ j\ \neq j' \\ \alpha \cdot t_{ji} & if\ job\ type\ of\ j\ = j' \end{cases} \tag{6}$$

where actual setup time is reduced when working on the same job type as the previous job on the same machine.

$$S_{(j'p')i} \geq C_{jpi} + s_{jj'i} \cdot y_{jj'i} \ \forall i \in M, \forall (j,p) \neq (j',p') \tag{7}$$

where start time takes into account setup time.

$$C_{jpi} \leq d_j \ \forall i \in M, \forall j \in J, \forall p \in P_j \tag{8}$$

where work completed before the due date.

$$S_{(jp')i} \geq C_{(jp)i} \ \ \forall i \in M, \forall j \in J, \forall p, p' \in P_j\ with\ p'\ start\ after\ p \tag{9}$$

where each process in one job must complete before the next process can start.

$$\sum_{i \in M} x_{jpi} \leq 1 \ \ \forall j \in J, \forall p \in P_j\ which\ is\ finished \tag{10}$$

where all previously completed processes are not reassigned to any machine.

Job Definition with Dynamic Characteristics

This study classifies occupations as dynamic by considering three primary factors: product type, running meter, and priority. The term "product type" describes the classification of the item under manufacture, which includes a variety of items with unique characteristics and production needs. Running Meter: This is a quantity of measurement that calculates how much material is required to manufacture one product. It has a significant contribution to production planning and scheduling. Priority: When you define the list of things to do and their hierarchy based on either market demand or production needs. By combining these 3 factors the system can monitor and manage tasks on a case-by-case basis allowing for more efficient production that can be adjusted according to changes in demand, work schedules, deadlines, etc.

Genetic Algorithm Implementation

Execution A genetic algorithm execution contains three initial population individuals or potential solutions, commonly referred to as chromosomes, is a members of an initial went on calmly enacted by its slime balls. The fitness function evaluates each generation to assess the final solution quality, usually in terms of makespan or total completion time for production scheduling.

Techniques such as roulette wheel selection select the most exceptional individuals as parents, giving those with superior fitness a higher probability of selection. The crossover, or recombination, process involves merging two parent individuals to generate new individuals through the exchange of their genes. Mutation follows, altering one or more genes in the individual chromosomes to introduce fresh variation and prevent premature convergence to a local solution. After the crossover and mutation process, a fitness function reevaluates the newly created individuals. Then, we carry out a new generation selection to determine the population of the next generation, retaining only the best individuals. We continue this iterative process for multiple generations until we meet a specific termination condition, such as a predetermined number of generations or a lack of substantial improvement in fitness values. The final objective is to discover the most optimal solution to an intricate scheduling problem.

Individual Representation

Each entity within the genetic algorithm corresponds to a single prospective resolution for the scheduling predicament. Chromosomes, which contain data regarding job sequences and machine assignments, represent these individuals. Here is the proposed pseudocode that describes the procedure for creating individuals using the FJSP model.

Pseudocode to create individual:

```
FUNCTION create_individual
    INITIALIZE machine_end_times TO dictionary with all machines set to 0
    INITIALIZE job_end_times TO dictionary with all jobs set to 0
    INITIALIZE last_job_type_on_machine TO dictionary with all machines set to None
    INITIALIZE job_process_allocation TO dictionary of dictionaries for each job and its processes set to None

    SHUFFLE job_list

    FOR EACH job IN job_list
        FOR EACH proc_index, proc IN enumerate(processes[job])
            INITIALIZE best_end_time TO infinity
            INITIALIZE best_machine TO None
            INITIALIZE best_start_time TO None
            INITIALIZE best_process_time TO None
            INITIALIZE best_setup_time TO None

            FOR EACH machine IN available_machines[(job, proc)]
                SET current_job_type TO job_types[job]

                IF last_job_type_on_machine[machine] EQUALS current_job_type
                    SET setup_time_current TO setup_time[(machine, current_job_type)] * alpha
                ELSE
                    SET setup_time_current TO setup_time[(machine, current_job_type)]

                CALCULATE process_time_adjusted AS processing_time[(job, proc)] * machine_speed[machine]
                CALCULATE start_time AS MAX of machine_end_times[machine] and job_end_times[job]

                CALCULATE end_time AS start_time + process_time_adjusted + setup_time_current
                IF end_time < best_end_time
                    UPDATE best_end_time TO end_time
                    UPDATE best_machine TO machine
                    UPDATE best_start_time TO start_time
                    UPDATE best_process_time TO process_time_adjusted
                    UPDATE best_setup_time TO setup_time_current

            APPEND (job, proc, best_machine, best_start_time, best_process_time, best_setup_time) TO ind
            UPDATE machine_end_times[best_machine] TO best_end_time
            UPDATE job_end_times[job] TO best_end_time
            UPDATE last_job_type_on_machine[best_machine] TO job_types[job]
            UPDATE job_process_allocation[job][proc] TO best_machine

    RETURN creator.Individual(ind)
END FUNCTION
```

Fitness Evaluation

We use the fitness function to evaluate each person's excellence based on the resulting make-up. Individuals with a shorter makespan are believed to have higher fitness. The implementation uses pseudocode, as depicted below.

Pseudocode to evaluate the individual:

```
Function evaluate(individual)
    Initialize:
        job_process_done as dictionary of jobs with lists of boolean (false)
        job_end_times as dictionary with job keys and zero values
        machine_end_times as dictionary with machine keys and zero values
        penalties set to 0

    For each gene in individual
        Extract job, proc, machine, start_time, process_time, setup_time from gene
        proc_index = index of proc in job's process list

        Check if process already done or out of sequence:
            If yes, add 1000 to penalties

        If machine is None:
            Add 10000 to penalties
            Continue to next gene

        Calculate actual start time:
            actual_start = maximum of machine_end_times[machine] and
                                  job_end_times[job] + setup_time
            end_time = actual_start + process_time

        If due date of current job equals the minimum of all due dates:
            Find jobs with the same due date
            Select job with lowest priority value from these jobs
            If current job is not the selected job:
                Continue to next iteration

        Manage job and machine timings:
            If actual_start < machine_end_times[machine]:
                Add 1000 to penalties
            Else:
                Update machine_end_times[machine] to end_time

        Update job end times:
            job_end_times[job] = end_time

        Mark process as done:
            Set job_process_done[job][proc_index] to true

        Check for due date violations:
            If job_end_times[job] > due_dates[job]:
                Calculate delay = job_end_times[job] - due_dates[job]
                Add delay * late_penalty_factor to penalties

    Compute total fitness:
        max_end_time = find maximum value in job_end_times
        total_fitness = max_end_time + penalties

```

```
50 :     Return total_fitness
51 :   End Function
```

Genetic Operations

The execution phases of a genetic algorithm consist of three primary stages: selection, crossover, and mutation. The selection process uses a tournament selection method to determine the reproductive capability of individuals based on their fitness value. During the crossover stage, the crossover operator chooses two individuals as parents and merges them to generate new progeny. The mutation stage occurs when people experience genetic mutations, which include altering small parts of their chromosomes and bringing new variants into the population.

Algorithm Parameters

The genetic algorithm is executed with the following parameters: population size, which represents the total count of individuals in a population; crossover probability, which denotes the likelihood of selecting two individuals for the crossover process; mutation probability, which indicates the likelihood of an individual undergoing a genetic mutation; and the number of generations, which signifies the total number of iterations performed by the algorithm.

Model Validation and Verification

Authentic data from the industrial industry serves as the validation for the model. We executed multiple test scenarios to assess the model's performance across different production settings. We compare the findings of the genetic algorithm with those of classical scheduling algorithms to assess its superiority in minimizing makespan.
We conducted experimentation using the parameters and data specified in Table 1.

Table 1 - Parameters of Genetic Algorithm

Parameter	Value
Population Size	100
Iteration Cycles	50
Crossover Rate	0.5
Crossover	Two Point Crossover
Mutation Rate	0.2

We conduct an experiment using real data and specified parameters to verify the successful execution of the genetic algorithm, which includes the individual creation function and fitness evaluation function tailored to the FJSP mathematical model. The experiment utilizes data that conforms to the format depicted in Fig. 4 above.

Executing the genetic algorithm model with the parameters listed in Table 1 yields the results shown in Table 2 below:

Table 2 - Output Genetic Algorithm Processing

Generation	Evaluations	Min	Max	Avg	Std
0	100	19437,6	403924,39	124022,435	119737,604
1	71	19437,6	16781833,3	457124,287	1746894,45
2	72	19437,6	8902560,03	559018,4	1560520,09
3	64	19437,6	4821242,06	178435,374	599782,226
4	62	19437,6	10785124,6	228200,947	1102360,05
5	55	19437,6	6280934,81	277160,279	1007041,46
6	59	19437,6	6759818,39	209885,271	793325,421
7	72	19437,6	7297970,96	432877,257	1324457,04
8	49	19437,6	3671933,85	106150,453	374423,028
9	65	19437,6	1491297,77	91870,0958	201631,431
10	64	19437,6	4108470,42	117675,655	429386,364
11	55	19437,6	792138,24	69627,981	135709,056
12	54	19437,6	4762791,7	107960,393	490773,183

13	53	19437,6	789138,24	44958,3339	107861,82
14	58	19437,6	3200881,77	75981,9919	359201,668
15	54	19437,6	1773886	90011,0427	277372,581
16	65	19437,6	504981,746	36281,7654	63415,7611
17	60	19437,6	1991480,79	57484,2642	206436,592
18	57	19437,6	434502,028	32314,63	57911,751
19	56	19437,6	1097112,06	37206,7141	117043,817
20	63	19437,6	997610,046	39334,4485	114680,941
21	48	19437,6	714664,569	29662,0678	70173,043
22	61	19437,6	2583451,78	51230,3994	255970,937
23	54	19437,6	2333563,52	74713,094	275503,494
24	57	19437,6	630696,934	35100,9773	72482,4448
25	52	19437,6	1841130,48	49721,6673	207417,777
26	61	19437,6	405827,593	28828,1265	46524,5268
27	56	19437,6	2597247,5	87815,8148	320329,138
28	59	19437,6	1847543,01	53817,2176	203993,441
29	48	19437,6	1173889,99	39816,6459	122979,469
30	63	19437,6	2154731,48	53816,8303	220529,887
31	72	19437,6	1620819,64	43411,0812	169051,725
32	60	19437,6	3158325,4	122629,185	476905,137
33	61	19437,6	515750,378	26813,2249	49372,1196
34	55	19437,6	1689160,35	54977,9955	203349,922
35	57	19437,6	2261273,75	56956,7813	241489,41
36	48	19437,6	5446138,41	89076,9857	546259,038
37	55	19437,6	262361,544	26852,0144	34772,7626
38	62	19437,6	1151732,63	43009,2897	123874,808
39	49	19437,6	1224703,61	55867,9959	165951,322
40	47	19437,6	2229909,45	61490,2475	241300,353
41	57	19437,6	1384854,33	37479,5284	139011,81
42	64	19437,6	1232839,93	54346,6454	158726,525
43	65	19437,6	1597624,2	61960,7218	214574,825
44	51	19437,6	1252527,84	58964,0339	200669,681
45	56	19437,6	2647005,9	69176,9255	295462,49
46	70	19437,6	244130,857	24886,4946	24049,265
47	57	19437,6	1337193,73	55828,0065	189262,955
48	60	19437,6	1477005,83	44464,4103	152787,525
49	53	19437,6	786102,844	36261,2875	103377,165
50	70	19437,6	1352691,44	52086,5605	187421,39

Analysis of the Results

The experiment yielded results that demonstrate the capabilities of the constructed model when using the genetic algorithm refer to Table 2 above, as shown in Fig. 5 below. The diagram illustrates the fitness statistics throughout multiple generations of the genetic algorithm. The graph illustrates three fundamental metrics: the minimal fitness, the average fitness, and the standard deviation of fitness for each generation. The red line represents the minimal fitness value, which has a tendency to fall and thereafter stabilize as generations advance, suggesting the algorithm's ability to discover superior solutions. The blue line depicts the mean fitness, which exhibits a declining and stabilizing pattern, indicating an enhancement in the overall quality of solutions within the population. The grey region represents the standard deviation of fitness, which indicates the degree of variety in the population. This variation tends to diminish as time progresses, suggesting that the algorithm is moving towards the ideal solution. In general, the graph demonstrates that the genetic algorithm is effectively optimizing the answer, achieving convergence within a specific number of generations.

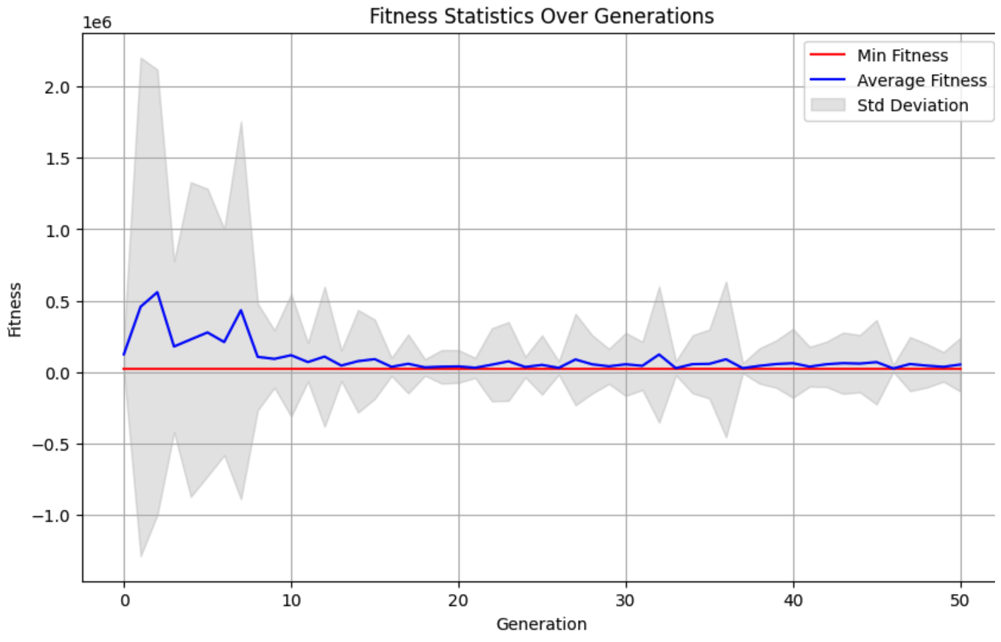


Fig. 5. Fitness Statistic Over Generations

4. Results and Discussions

The extraction stage is the initial phase, where we collect and validate data from multiple sources to ensure precision. Subsequently, in the transform phase, the acquired data undergoes processing and organization, ensuring its readiness for utilization. After conversion, the load phase sends the processed data to the repository. During the execution stage, the system applies the genetic algorithm to the data repository to identify the most optimal solution. e system presents the optimal schedule derived from the optimization process. In these stages, the study ensures a systematic execution of each step, from extraction to algorithm execution, to achieve the best possible scheduling results.

Table 3 - Data Processing for Flexible Packaging Manufacture

Job	Type	Running Meter	Colour	Process	Code	Min Speed	Ratio Speed	Setup Time
J57	22	162223	9	P1	M3	200	0,5	150
J57	22	162223	9	P2	M4	200	1	90
J57	22	162223	9	P2	M5	200	0,75	90
J57	22	162223	9	P2	M6	200	0,5	90
J57	22	162223	9	P2	M7	200	1	90
J57	22	162223	9	P3	M8	200	1	90
J57	22	162223	9	P4	M9	80	1	60
J57	22	162223	9	P4	M10	80	0,75	60
J57	22	162223	9	P4	M11	80	0,75	60
J57	22	162223	9	P4	M12	80	0,5	60

J57	22	162223	9	P4	M13	80	0,125	60
J57	22	162223	9	P4	M14	80	1	90
J57	22	162223	9	P5	M15	80	0,75	90

Table 3 displays the outcomes of data processing for a production task involving product type J57. This particular product has a production length of 162223 meters and utilizes a total of 9 distinct colors. The provided data contains comprehensive information for each production process, including a Process List with the sequence number (No), the name and code of the machine used, the number of colors processed at each stage, the minimum machine speed (MinSpeed), the machine speed ratio (Ratio Speed), and the machine setup time (Setup Time). This means during the first procedure we will print at a B1 printing machine (M3) with a minimum velocity of 200, velocity ratio of.5, and setup duration =150 if, for example, you would like to produce tickets. Its existence allows us to schedule merged production schedules - and plan accordingly how each operation would be executed at full efficiency, about the machine capacity that is available for use. This access provides more accurate scheduling and lets you pull insights on areas that can be improved operationally.

Results

We conducted a series of experiments using a Genetic Adaptive Scheduling System (GASS) model to improve the production scheduling process before creating a Gantt chart to display the most efficient scheduling. The goal of this experiment is to reduce the longest job duration and optimize task distribution among the available computers. We anticipate that the GA model, by leveraging authentic industry data and powerful hardware, will deliver a scheduling solution that not only minimizes machine idle time but also enhances overall productivity. We will display the outcomes of this model through a Gantt chart, which will illustrate the order and duration of each task on the respective machines. This will provide a clear and comprehensive depiction of the optimal scheduling that emerges.

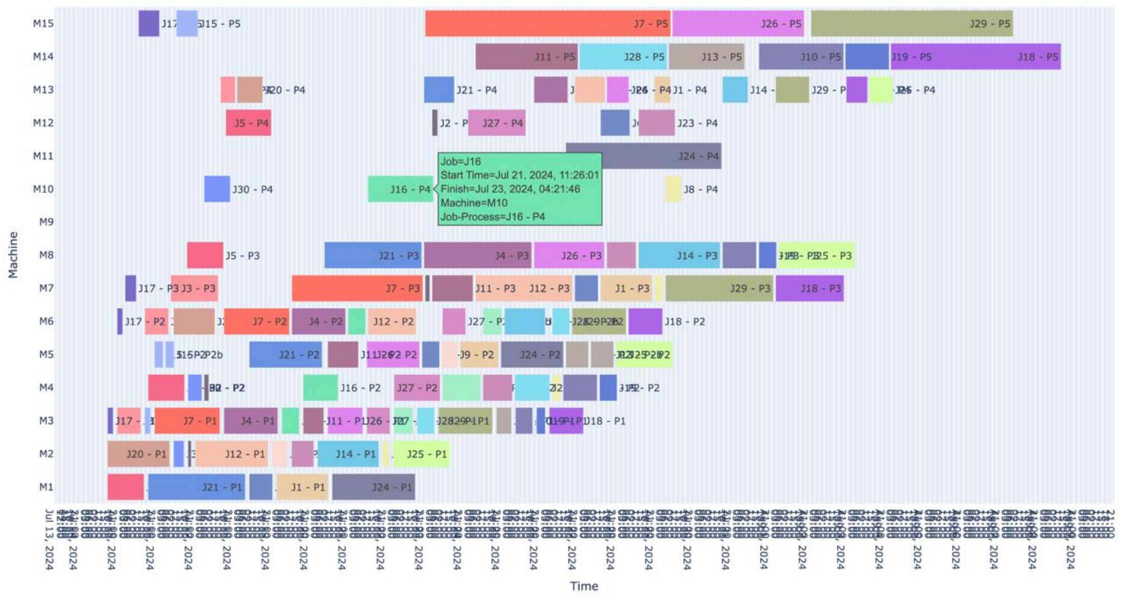


Fig. 6. Gantt Chart Output by Genetic Algorithm Model

Fig. 6 Gantt Chart Output by Genetic Algorithm Model. The Order and Machine Efficiency Information is the most capable plan of production for order work to machines Task-wise. Every colored block on this plot is a unique task that some machine needs to do within its allotted time. Taking into account variables such as product type, linear meters to run, work importance, and machine capacity the model determines a schedule in an automated manner. The outcomes show that the proposed model is capable of assigning jobs to machines accurately, solving allocation problems without idle state, and zero downtime in order to satisfy continuous goods stream. As a result, industries are now in perfect condition to go with the schedule made for them and improve both operational efficiency as well as effectiveness while

meeting their specific dynamic production requirements. This automation plan allows enterprises to address market demands quickly, utilize resources efficiently, and enhance production.

We conducted the Genetic Adaptive Scheduling System (GASS) model experiment on an Apple M2 Pro laptop with 16GB of memory. Table 4 below displays the results of this experiment. This hardware guarantees efficient and rapid execution of the scheduling process and makespan computation by harnessing the laptop's high processing power and extensive memory capacity. The experimental results demonstrate the superiority of the Genetic Adaptive Scheduling System (GASS) approach in generating a more efficient schedule when compared to traditional approaches, in terms of both scheduling time and overall work completion time (makespan).

Table 4 - Comparison of Conventional Method vs GASS Method

Job Qty	Conventional Method		GASS Method	
	Time	Makespan	Time	Makespan
10	50	15,19	0,49	5,4
15	75	22,785	0,49	8,1
20	100	30,38	0,49	10,8
25	125	37,975	0,49	13,5
30	150	45,57	0,49	16,2
35	175	53,165	0,49	18,9
40	200	60,76	0,49	21,6
45	225	68,355	0,49	24,3
50	250	75,95	0,49	27

In Table 4 shown above, performances of traditional scheduling methods were compared with GASS-based techniques for different job quantities. In both timetabling, the "Time" column shows the time taken to generate a schedule, and one side of "Makespan' row demonstrates the total used for execution (makespan) by each method.

This table also shows that the scheduling time is frequently reduced (to 0.49 unit-time) in the GA technique compared to the conventional method. The larger the number of jobs (from 50 to more than say around 250 units a year) this advantage becomes even more pronounced. In addition, the GASS model results with far better makespan than a typical approach. As an example, the results of 10 jobs should be a makespan of respectively 5.4 (GASS methodology) and a lengthened to approximately three times as long if using traditional methods ending in about 15.19 As the number of jobs increases, this trend is confirmed as again (always) GASS results in lower makespans. For example, for 50 jobs the makespan is equal to 27 when obtained by the GASS method and the conventional approach yields a value of makespan as high as 75.95.

5. Conclusion

This study presents a comprehensive dataset designed to support research and development in Flexible Job Shop Scheduling Problems (FJSS), particularly for dynamic production environments. The dataset incorporates critical real-world factors such as machine flexibility, varying processing times, job priorities, setup times, and machine downtimes, offering a robust foundation for advanced optimization techniques.

The dataset is specifically tailored for the implementation and evaluation of the Genetic Adaptive Scheduling System (GASS), a modified genetic algorithm framework. By integrating enhanced mutation and selection processes, GASS provides significant improvements in scheduling efficiency, achieving up to a 45.3% reduction in makespan within the flexible packaging industry. This performance underscores its potential as a scalable and adaptable solution for dynamic scheduling challenges.

The inclusion of real-world constraints ensures the dataset's applicability across various industries, including printing, electronics, and pharmaceuticals. Researchers can leverage this dataset to explore new methods for addressing complex scheduling problems, including multi-objective optimization, dynamic adaptability, and real-time decision-making.

Future work should focus on expanding the dataset to include additional dynamic factors, such as variable demand patterns and energy constraints, and testing its applicability in different industrial scenarios. Moreover, integrating the dataset with machine learning models could provide further insights into predictive and adaptive scheduling strategies.

Acknowledgment

We would like to thank PT. Intikemas Putra Makmur for the great help and cooperation This observation facility, and the data sources that help enable it are things we thank for every day; our work in the Lapp Constituent Survey would not be possible without a steady flow of this kind of data. Observation in a real industry was given and a detail understanding of the system along with practical data that played a significant role indirectly in making development more accurate as well as an activity based scheduling model. We would like to thank PT. Intikemas Putra Makmur, for their invaluable assistance in supporting us throughout the research of development process!!

References

Abderrahim, M. (2020). Manufacturing 4.0 operations scheduling with AGV battery management constraints. *Energies*, *13*(18). https://doi.org/10.3390/en13184948

Awad, M. A. (2021). An Efficient Modified Genetic Algorithm for Integrated Process Planning-Job Scheduling. *2021 International Mobile, Intelligent, and Ubiquitous Computing Conference, MIUCC 2021*, 319–323. https://doi.org/10.1109/MIUCC52538.2021.9447610

Baykasoğlu, A., Madenoğlu, F. S., & Hamzadayı, A. (2020). Greedy randomized adaptive search for dynamic flexible job-shop scheduling. *Journal of Manufacturing Systems*, *56*, 425–451. https://doi.org/10.1016/j.jmsy.2020.06.005

Fan, J., Zhang, C., Liu, Q., Shen, W., & Gao, L. (2022). An improved genetic algorithm for flexible job shop scheduling problem considering reconfigurable machine tools with limited auxiliary modules. *Journal of Manufacturing Systems*, *62*, 650–667. https://doi.org/10.1016/j.jmsy.2022.01.014

Ghaleb, M. (2020). Real-time production scheduling in the Industry-4.0 context: Addressing uncertainties in job arrivals and machine breakdowns. *Computers and Operations Research*, *123*. https://doi.org/10.1016/j.cor.2020.105031

Hong, T. Y., & Chien, C. F. (2020). A simulation-based dynamic scheduling and dispatching system with multi-criteria performance evaluation for Industry 3.5 and an empirical study for sustainable TFT-LCD array manufacturing. *International Journal of Production Research*, *58*(24), 7531–7547. https://doi.org/10.1080/00207543.2020.1777342

Huang, X., & Yang, L. (2019). A hybrid genetic algorithm for multi-objective flexible job shop scheduling problem considering transportation time. *International Journal of Intelligent Computing and Cybernetics*, *12*(2), 154–174. https://doi.org/10.1108/IJICC-10-2018-0136

Lei, K. (2024). Large-Scale Dynamic Scheduling for Flexible Job-Shop with Random Arrivals of New Jobs by Hierarchical Reinforcement Learning. *IEEE Transactions on Industrial Informatics*, *20*(1), 1007–1018. https://doi.org/10.1109/TII.2023.3272661

Li, K., Deng, Q., Zhang, L., Fan, Q., Gong, G., & Ding, S. (2021). An effective MCTS-based algorithm for minimizing makespan in dynamic flexible job shop scheduling problem. *Computers and Industrial Engineering*, *155*. https://doi.org/10.1016/j.cie.2021.107211

Li, Y. (2020). Machine learning and optimization for production rescheduling in Industry 4.0. *International Journal of Advanced Manufacturing Technology*, *110*(9), 2445–2463. https://doi.org/10.1007/s00170-020-05850-5

Li, Y., Gu, W., Yuan, M., & Tang, Y. (2022). Real-time data-driven dynamic scheduling for flexible job shop with insufficient transportation resources using hybrid deep Q network. *Robotics and Computer-Integrated Manufacturing*, *74*. https://doi.org/10.1016/j.rcim.2021.102283

Luo, X. (2020). Improved genetic algorithm for solving flexible job shop scheduling problem. *Procedia Computer Science*, *166*, 480–485. https://doi.org/10.1016/j.procs.2020.02.061

Luo, X., Qian, Q., & Fu, Y. F. (2020). Improved genetic algorithm for solving flexible job shop scheduling problem. *Procedia Computer Science*, *166*, 480–485. https://doi.org/10.1016/j.procs.2020.02.061

Meng, L. (2023). An Improved Genetic Algorithm for Solving the Multi-AGV Flexible Job Shop Scheduling Problem. *Sensors*, *23*(8). https://doi.org/10.3390/s23083815

Psarommatis, F. (2020). Improved heuristics algorithms for re-scheduling flexible job shops in the era of zero defect manufacturing. *Procedia Manufacturing*, *51*, 1485–1490. https://doi.org/10.1016/j.promfg.2020.10.206

Qin, Z. (2021). Self-organizing manufacturing network: A paradigm towards smart manufacturing in mass personalization. *Journal of Manufacturing Systems*, *60*, 35–47. https://doi.org/10.1016/j.jmsy.2021.04.016

Rossit, D. A., Tohmé, F., & Frutos, M. (2019). A data-driven scheduling approach to smart manufacturing. *Journal of Industrial Information Integration*, *15*, 69–79. https://doi.org/10.1016/j.jii.2019.04.003

Samsonov, V. (2021). Manufacturing control in job shop environments with reinforcement learning. *ICAART 2021 - Proceedings of the 13th International Conference on Agents and Artificial Intelligence*, *2*, 589–597. https://www.scopus.com/inward/record.uri?partnerID=HzOxMe3b&scp=85103850187&origin=inward

Sana, S. S., Ospina-Mateus, H., Arrieta, F. G., & Chedid, J. A. (2019). Application of genetic algorithm to job scheduling under ergonomic constraints in manufacturing industry. *Journal of Ambient Intelligence and Humanized Computing*, *10*(5), 2063–2090. https://doi.org/10.1007/s12652-018-0814-3

Shao, W. (2021). Effective constructive heuristics for distributed no-wait flexible flow shop scheduling problem. *Computers and Operations Research*, *136*. https://doi.org/10.1016/j.cor.2021.105482

Singh, K., & Sundar, S. (2019). A hybrid steady-state genetic algorithm for the min-degree constrained minimum spanning tree problem. *European Journal of Operational Research*, *276*(1), 88–105. https://doi.org/10.1016/j.ejor.2019.01.002

Soares, L. C. R. (2020). Biased random-key genetic algorithm for scheduling identical parallel machines with tooling constraints. *European Journal of Operational Research*, *285*(3), 955–964. https://doi.org/10.1016/j.ejor.2020.02.047

Tarigan, M., Gaol, F. L., Mauritsius, T., & Budiharto, W. (2023). Scheduling Production in the Flexible Packaging Industry Using Mathematical Models and Genetic Algorithms. *2023 International Conference on Informatics Engineering, Science & Technology (INCITEST)*, 1–5. https://doi.org/10.1109/INCITEST59455.2023.10397033

Tian, Z. (2023). Dynamic energy-efficient scheduling of multi-variety and small batch flexible job-shop: A case study for the aerospace industry. *Computers and Industrial Engineering*, *178*. https://doi.org/10.1016/j.cie.2023.109111

Türkyılmaz, A. (2020). A research survey: heuristic approaches for solving multi objective flexible job shop problems. *Journal of Intelligent Manufacturing*, *31*(8), 1949–1983. https://doi.org/10.1007/s10845-020-01547-4

Umam, M. S., Mustafid, M., & Suryono, S. (2021). A hybrid genetic algorithm and tabu search for minimizing makespan in flow shop scheduling problem. *Journal of King Saud University - Computer and Information Sciences*. https://doi.org/10.1016/j.jksuci.2021.08.025

Wang, Y., & Zhu, Q. (2021). A Hybrid Genetic Algorithm for Flexible Job Shop Scheduling Problem with Sequence-Dependent Setup Times and Job Lag Times. *IEEE Access*, *9*, 104864–104873. https://doi.org/10.1109/ACCESS.2021.3096007

Xie, J. (2023). A hybrid genetic tabu search algorithm for distributed flexible job shop scheduling problems. *Journal of Manufacturing Systems*, *71*, 82–94. https://doi.org/10.1016/j.jmsy.2023.09.002

Xu, B. (2020). Genetic programming with delayed routing formultiobjective dynamic flexible job shop scheduling. *Evolutionary Computation*, *29*(1), 75–105. https://doi.org/10.1162/evco_a_00273

Zhang, F. (2021). Evolving Scheduling Heuristics via Genetic Programming with Feature Selection in Dynamic Flexible Job-Shop Scheduling. *IEEE Transactions on Cybernetics*, *51*(4), 1797–1811. https://doi.org/10.1109/TCYB.2020.3024849

Zhang, F. (2022). Multitask Genetic Programming-Based Generative Hyperheuristics: A Case Study in Dynamic Scheduling. *IEEE Transactions on Cybernetics*, *52*(10), 10515–10528. https://doi.org/10.1109/TCYB.2021.3065340

Zhang, F. (2023). Multitask Multiobjective Genetic Programming for Automated Scheduling Heuristic Learning in Dynamic Flexible Job-Shop Scheduling. *IEEE Transactions on Cybernetics*, *53*(7), 4473–4486. https://doi.org/10.1109/TCYB.2022.3196887

Zhang, S. (2021a). A hybrid multi-objective approach for real-time flexible production scheduling and rescheduling under dynamic environment in Industry 4.0 context. *Computers and Operations Research*, *132*. https://doi.org/10.1016/j.cor.2021.105267

Zhang, S. (2021b). Dual resource constrained flexible job shop scheduling based on improved quantum genetic algorithm. *Machines*, *9*(6). https://doi.org/10.3390/machines9060108

Zhang, X., Liao, Z., Ma, L., & Yao, J. (2022). Hierarchical multistrategy genetic algorithm for integrated process planning and scheduling. *Journal of Intelligent Manufacturing*, *33*(1), 223–246. https://doi.org/10.1007/s10845-020-01659-x

Zhao, Y., & Zhang, H. (2021). Application of machine learning and rule scheduling in a job-shop production control system. *International Journal of Simulation Modelling*, *20*(2), 410–421. https://doi.org/10.2507/IJSIMM20-2-CO10

Zheng, P., Zhang, P., Wang, J., Zhang, J., Yang, C., & Jin, Y. (2022). A data-driven robust optimization method for the assembly job-shop scheduling problem under uncertainty. *International Journal of Computer Integrated Manufacturing*, *35*(10–11), 1043–1058. https://doi.org/10.1080/0951192X.2020.1803506

Zhou, L. (2020). Deep reinforcement learning-based dynamic scheduling in smart manufacturing. *Procedia CIRP*, *93*, 383–388. https://doi.org/10.1016/j.procir.2020.05.163

Zhu, H., Chen, M., Zhang, Z., & Tang, D. (2019). An adaptive real-time scheduling method for flexible job shop scheduling problem with combined processing constraint. *IEEE Access*, *7*, 125113–125121. https://doi.org/10.1109/ACCESS.2019.2938548

Zhu, K. (2023). Dynamic distributed flexible job-shop scheduling problem considering operation inspection. *Expert Systems with Applications*, *224*. https://doi.org/10.1016/j.eswa.2023.119840

Zhuang, Z., Huang, Z., Chen, L., & Qin, W. (2019). A Heuristic Rule Based on Complex Network for Open Shop Scheduling Problem with Sequence-Dependent Setup Times and Delivery Times. *IEEE Access*, *7*, 140946–140956. https://doi.org/10.1109/ACCESS.2019.2944296