Hippopotamus Optimization for Dynamic Flexible Job Shop Scheduling under Machine Tool Breakdowns

Katarina Z. Brenjo
Research assistant
University of Belgrade
Faculty of Mechanical Engineering

Aleksandar V. Jokić
Teaching Assistant
University of Belgrade
Faculty of Mechanical Engineering

Milica M. Petrović
Associate Professor
University of Belgrade
Faculty of Mechanical Engineering

Dynamic flexible job shop scheduling under machine tool breakdowns represents a complex and highly constrained combinatorial optimization problem in modern manufacturing systems. This research paper proposes an integrated optimization framework that simultaneously determines operation sequencing, machine tool assignment, tool selection, and tool orientation with the objective function of minimizing makespan. A unified multi-string solution representation is developed to simultaneously model all decision layers. Three biologically inspired metaheuristic algorithms, Genetic Algorithm (GA), Particle Swarm Optimization (PSO), and Hippopotamus Optimization (HO), are implemented using the proposed encoding scheme. A rescheduling strategy is introduced to preserve completed operations while rescheduling the affected operations after machine tool failures. Experimental verification demonstrates that the integrated framework effectively handles dynamic disturbances and significantly improves scheduling performance. Comparative analysis shows that the hippopotamus optimization algorithm achieves superior convergence behavior and better objective function values than the other approaches. The proposed method provides a robust framework for resilient scheduling under multiple resource constraints.

Keywords: *dynamic flexible job shop scheduling, rescheduling, machine breakdown, genetic algorithm, particle swarm optimization, hippopotamus optimization.*

1. INTRODUCTION

Modern manufacturing systems are increasingly exposed not only to conventional disturbances but also to cyber-physical threats. With the rapid digitalization of production systems and the integration of Industry 4.0 principles, cyber-attacks have become a significant cause of disturbances, often resulting in temporary machine tool unavailability or unexpected resource constraints. As a consequence, manufacturing systems are required to perform in highly dynamic environments where machine tool failures, resource interdependencies, and complex operational constraints frequently occur. In such environments, the Dynamic Flexible Job Shop Scheduling Problem (DFJSSP) represents a fundamental operational challenge that directly affects production efficiency, robustness, and overall system resilience. Machine tool breakdowns, whether caused by physical failure or by cyber-attacks, significantly intensify this challenge by compromising the feasibility of existing schedules and demanding structured rescheduling approaches to ensure operational stability while maintaining predefined performance criteria and optimization objectives. Therefore, the ability to effectively reschedule operations after cyber-induced or physical disturbances becomes essential for minimizing total processing time, preserving system performance, and enhancing cyber resilience in smart manufacturing environments.

Flexible job shop scheduling extends the classical job shop problem by allowing each operation to be processed on alternative machine tools, thereby increasing system flexibility while expanding the solution space. Previous studies mainly concentrate on operation sequencing and machine tool assignment. However, modern manufacturing systems frequently involve additional flexibilities such as tool selection and tool orientation (Tool Access Direction – TAD), which significantly affect processing feasibility and overall performance. The integrated optimization of these different types of flexibility remains insufficiently investigated, particularly in dynamic environments.

Over the past decades, the classical Job Shop Scheduling Problem (JSSP) has gradually evolved toward more realistic and flexible manufacturing systems, leading to the development of the Flexible Job Shop Scheduling Problem (FJSSP), in which each operation can be processed on one of several compatible machine tools, thereby introducing an additional level of flexibility while significantly increasing the combinatorial complexity of the problem. Unlike the JSSP, where the primary decision concerns only the sequencing of operations, the FJSSP requires simultaneous resolution of two interdependent decision-making sub–

Received: March 2026, Accepted: April 2026
Correspondence to: Katarina Brenjo
Faculty of Mechanical Engineering,
Kraljice Marije 16, 11120 Belgrade 35, Serbia
E-mail: kbrenjo@mas.bg.ac.rs
doi: 10.5937/fme2602354B

problems: (i) machine assignment for each operation and (ii) sequencing of operations on each selected machine [1]. In the literature, solution approaches for the FJSSP are generally classified into four main categories: exact methods, which guarantee optimal results but are computationally feasible only for small and medium-sized problems [2], heuristic approaches based on priority dispatching rules, learning-based approaches, and metaheuristic algorithms (such as Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Simulated Annealing (SA), and hybrid strategies), which have become dominant in recent research due to their ability to provide high-quality solutions within acceptable computational times for NP-hard problems [3]. Recent studies emphasize the transition from single-objective optimization to multi-objective optimization frameworks that incorporate energy efficiency, operational costs, sustainability, and robustness, particularly within the context of Industry 4.0 [4]. Recent research directions also include combining different metaheuristic strategies, incorporating problem-specific knowledge to improve search efficiency and convergence behavior [5], and integrating metaheuristic methods with data-driven and reinforcement learning approaches [6]. In study [7], a collaborative multi-agent reinforcement learning framework models the scheduling problem as a multi-objective Markov Decision Process, integrating makespan and energy consumption, and employs disjunctive graph representations and dual-agent decision mechanisms. In contrast, [8] enhances classical NSGA-II by introducing adaptive crossover and mutation operators and an improved elite preservation strategy, thereby strengthening convergence toward the Pareto front in multi-objective FJSSP. Swarm-based approaches are further extended in [9], where an improved Harris Hawk Optimization (HHO) algorithm introduces elitism, chaotic perturbation, and nonlinear escaping energy mechanisms to reduce the risk of early convergence in static scheduling scenarios. Additionally, a broader comparative analysis of recent population-based metaheuristics is presented in [10], highlighting differences in exploration-exploitation balance, robustness, and convergence behavior, thereby providing insight into the structural advantages of emerging biologically inspired algorithms for complex optimization problems. An improved dung beetle optimization algorithm, combined with SA and a single-layer encoding scheme, is developed to solve the single-objective FJSSP [11]. The incorporation of local search enhances convergence speed and improves solution quality, further confirming the effectiveness of biologically inspired swarm intelligence methods for complex scheduling problems.

As introduced earlier, the increasing exposure of manufacturing systems to real-time disturbances has led to the development of the DFJSSP, which explicitly considers unpredictable events such as random job arrivals [12] and machine tool breakdowns [13]. Recent studies have explored Reinforcement Learning (RL) as a promising framework for dynamic scheduling – rescheduling, by modeling DFJSSP as a sequential decision-making process in which decision strategies are continuously updated through interaction with the environment [14]. Multi-agent RL architectures further decompose scheduling decisions into cooperative agents responsible for job selection and machine tool assignment, aiming to enhance adaptability under disturbances [15]. Authors in [16] propose a robust pro-active scheduling framework where machine tool repair time is embedded into a disjunctive graph and optimized using the proximal policy optimization algorithm, while [17] develops a dual-objective deep RL model based on Double Deep Q-Network enhanced with an attention mechanism to handle machine breakdowns while optimizing machine offset and total tardiness. These RL-based methods demonstrate strong adaptability and learning capability in highly dynamic environments. However, their performance strongly depends on the availability of sufficient training data and on the careful design of the reward function, which can make their direct applicability to real-time rescheduling difficult.

Unlike RL-based approaches, metaheuristic algorithms remain the most widely adopted solution strategy for DFJSSP due to their flexibility, robustness, and without requiring a prior training phase. Early work [18] addressed DFJSSP with random machine tool breakdowns by proposing a two-stage Hybrid Genetic Algorithm (HGA). In this approach, the first stage minimizes the makespan in a deterministic setting, while the second stage incorporates expected breakdown information to improve robustness and schedule stability. This study clearly justifies the use of GA in machine breakdown environments due to its flexibility in handling routing and sequencing decisions within a complex combinatorial space. Subsequent research [19] further formalized the concept of robustness under machine breakdowns and integrated breakdown-related information directly into evolutionary search mechanisms. By incorporating probabilistic knowledge of failures into the optimization process, GA-based methods demonstrated improved resistance to schedule degradation after disturbances. Evolutionary variants such as NSGA-II and NRGA were further applied to improve stability under simulated machine breakdown conditions [20], while extended problem formulations that incorporate breakdowns, new job arrivals, and energy-related objectives continue to rely on population-based evolutionary search mechanisms [21]. Research in [22] presents an IoT-based dynamic scheduling framework that combines a GA with rule-based repair strategies to address random machine breakdowns in DFJSSP. The integration of real-time data within an Industry 4.0 based Manufacturing Execution System environment improves scheduling robustness and adaptability. A complete and right-shift rescheduling strategy for DFJSSP with machine failure based on GA is proposed in [23]. Their results confirm that complete rescheduling outperforms simple right-shift repair strategies, especially under severe machine failures. The study demonstrates the strength of GA in dynamic rescheduling environments, though parameter tuning and premature convergence remain challenges. In parallel with GA developments, the Particle Swarm Optimization (PSO) algorithm has been widely explored for DFJSSP under machine tool breakdowns. The two-stage PSO proposed in [24] first constructs a predictive schedule and then

adjusts it upon a machine breakdown, maintaining a minimized makespan while improving robustness and stability. PSO is justified in this context due to its relatively simple structure, limited parameter adjustment, and strong global search capability. Further improvements of PSO-based scheduling under machine breakdowns are presented in [25], where several PSO variants are proposed to reduce computational time and enhance rescheduling performance. These variants aim to improve robustness and stability in the presence of disturbances, while also addressing the runtime limitations of classical PSO implementations.

In addition to classical evolutionary strategies, recent studies have explored novel biologically inspired swarm optimization algorithms. Authors in [26] propose a Discrete Improved Gray Wolf Optimization (DIGWO) algorithm for dynamic distributed FJSSP considering random job arrivals and machine breakdowns. Their predictive-reactive hybrid strategy transforms dynamic scheduling into static subproblems within scheduling windows, optimizing multiple objectives including makespan, tardiness, factory load, and stability. The DIGWO algorithm enhances traditional GWO through hybrid initialization, prey-search strategies, and an external archive mechanism, demonstrating superior performance compared to conventional multi-objective evolutionary algorithms. Hippopotamus Optimization (HO) algorithm, as a recently proposed biologically inspired metaheuristic [27], introduces a three-phase search mechanism (group movement, defensive strategy, and predator avoidance) that naturally balances exploration and exploitation. Compared to traditional swarm algorithms, HO exhibits strong global exploration in early iterations, adaptive local exploitation in later iterations, simplified parameter structure, and high suitability for discrete adaptation. The HO algorithm has demonstrated extensive applicability across various engineering and computational domains. Its effectiveness has been confirmed in mechanical component design optimization [28], global and engineering design problems [29], solar photovoltaic output prediction [30], music genre classification tasks [31], and IoT healthcare security enhancement [32]. These studies collectively indicate the flexibility and robustness of HO in solving diverse nonlinear and complex optimization problems. Most importantly, HO remains insufficiently investigated in the context of DFJSSP under machine breakdown scenarios. Therefore, the adaptation to a multi-string discrete encoding framework for DFJSSP represents both methodological novelty and a contribution within this research.

Overall, the literature confirms that GA and PSO are highly competitive metaheuristic approaches for DFJSSP under machine breakdowns, primarily due to their flexibility in handling discrete encoding, routing decisions, and rescheduling. At the same time, many existing approaches that rely on simplified machine breakdown assumptions may suffer from premature convergence or involve considerable computational effort in dynamic environments. These limitations highlight the need for more structured and robust search mechanisms that can maintain solution quality while preserving computational efficiency. In this context, recent biologically inspired optimization strategies, such as the HO algorithm, offer a promising alternative framework for further investigation. Therefore, the proposed HO approach is systematically compared with GA and PSO, which represent the established benchmark methods in DFJSSP research.

Existing studies predominantly focus on sequencing and machine allocation decisions, while additional resource-related layers, such as tool selection and tool orientation [33,34], are rarely incorporated within a unified optimization framework. Furthermore, rescheduling strategies are often limited to either complete rescheduling or simplified repair mechanisms, without a structured integration of partial rescheduling under multi-resource constraints. These research challenges motivate further research of an integrated multi-string optimization framework capable of simultaneously addressing sequencing, machine allocation, and resource-related decisions under breakdown conditions. Accordingly, the proposed research introduces a unified encoding and structured rescheduling strategy and evaluates GA, PSO, and HO within this integrated DFJSSP framework.

The main motivation for this research arises from the increasing complexity of modern manufacturing systems, where cyber-induced disturbances and machine tool breakdowns require efficient and reliable rescheduling under multiple interdependent resource constraints. In contrast to most existing studies that primarily focus on operation sequencing and machine allocation, the proposed approach integrates additional decision layers, including tool selection and tool orientation, within a unified optimization framework.

The novelty of this study is achieved through the development of a unified multi-string encoding scheme and a structured partial rescheduling strategy that preserves completed operations while updating only the operations affected by disturbances. Furthermore, a discrete adaptation of the Hippopotamus Optimization (HO) algorithm is introduced for solving the DFJSSP. Compared to existing approaches, which typically address resource decisions separately or rely on simplified rescheduling mechanisms, the proposed framework enables more robust and computationally efficient adaptation to disturbances while maintaining feasibility and improving scheduling performance.

The research paper is structured as follows. Section 2 presents the problem formulation of the DFJSSP, including its mathematical model, fundamental constraints, and explicit modeling of machine breakdown disturbances in manufacturing systems. Section 3 introduces the integrated multi-string encoding and decoding scheme, which provides a unified representation of operation sequencing, machine assignment, tool selection, and tool orientation. Section 4 describes the applied optimization algorithms and their adaptation to the proposed discrete scheduling framework. Section 5 provides the experimental verification and comparative analyses of the proposed approach under dynamic conditions. Finally, Section 6 concludes the paper and outlines potential directions for future research.

2. PROBLEM FORMULATION

The problem considered in this research paper originates from the FJSSP, in which each job consists of a predefined sequence of operations, while each operation may be processed on one of several alternative machine tools and requires the selection of compatible tools and feasible tool approach directions. This structural flexibility significantly increases the combinatorial complexity of the scheduling problem, as alternative resource selection decisions become integrated with sequencing decisions.

Let:

- $N = \{1,2,\ldots, n\}$ denote the set of jobs,
- $M = \{1,2,\ldots, m\}$ the set of machines,
- $T = \{1,2,\ldots, \tau\}$ denote the set of available tools,
- $D = \{1,2,\ldots, \delta\}$ denote the set of feasible tool orientations.

Each job i includes an ordered set of operations (1):

$$O_i = \left\{O_{i1}, O_{i2}, ..., O_{ik_i}\right\}, i \in N \tag{1}$$

where K_i is the total number of operations required to complete job i. Operations of the same job must follow a predefined processing order (2):

$$\begin{aligned} &O_{ij} \prec O_{i,j+1}, \forall i \in N \\ &j = 1,..., K_{i-1} \end{aligned} \tag{2}$$

This requirement prevents parallel processing of operations within the same job and ensures sequential consistency of the manufacturing process. Each operation O_{ij} can be performed according to alternative process plans, each defined by a specific set of feasible machine tools (3):

$$M_{ij} \subseteq M \tag{3}$$

If operation O_{ij} is assigned to machine tool $k \in M_{ij}$, its processing time is denoted by P_{ijk}. Machine processing capacity is limited to a single operation at any time. The presence of alternative process plans introduces resource flexibility, meaning that the process plan for each job is not fixed in advance but is determined during the optimization. Consequently, machine tool assignment decisions directly determine operation processing times, transportation requirements, and the resulting operation sequencing on each machine tool within the manufacturing system. In addition to machine tool selection, each operation requires:

- a compatible tool $t \in T_{ij}$,
- a feasible tool approach direction $d \in D_{ij}$.

The introduction of tool and orientation constraints transforms the scheduling problem into a multi-resource allocation framework. Machine selection decisions can no longer be treated independently of tool compatibility and orientation feasibility, thereby significantly increasing the dimensionality and structural interdependence of the search space within the manufacturing system. If sequential operations of job i are processed on different machine tools k and l, a transportation time (4) is incurred – $TT(k,l)$:

$$\begin{aligned} TT = \left[tt_{kl}\right]_{m\times m}, \ \ tt_{kl} \geq 0, \\ tt_{kk} = 0 \end{aligned} \tag{4}$$

Therefore, the start time of operation $O_{i,j+1}$ is defined as (5):

$$S_{j,i+1} \geq C_{ij} + \tau\tau_{kl} \tag{5}$$

Where C_{ij} denotes the completion time of operation O_{ij}, k represents the machine tool assigned to operation O_{ij} and l is the machine assigned to the subsequent operation O_{ij+1}. Beyond processing and transportation times, additional setup times are required due to changes in machine tool resources. When two successive operations processed on the same machine require different tools, a tool change time (6) is incurred:

$$t_{change}^{tool}\left(t_{prev}, t_{new}\right) \tag{6}$$

where the special case is defined in (7):

$$t_{change}^{tool} = 0 \text{if } t_{prev} = t_{new} \tag{7}$$

Similarly, changes in tool access directions (TAD) may require additional setup time (8):

$$t_{change}^{tad}\left(d_{prev}, d_{new}\right) \tag{8}$$

The completion time of operation O_{ij} processed on machine k is defined as (9):

$$C_{ij} = S_{ij} + p_{ijk} \tag{9}$$

The start time S_{ij} of operation O_{ij} is defined as (10):

$$\begin{aligned} S_{ij} = \max\left(C_{i,j-1} + tt_{k'k}, C_{prev}\right) + + \\ t_{change}^{tool} + t_{change}^{tad} \end{aligned} \tag{10}$$

Where k´ represents the machine tool assigned to operation $O_{i,j\text{-}1}$, k is the machine assigned to the operation O_{ij} and C_{prev} denotes the completion time of the operation processed immediately before O_{ij} on the machine tool k.

The completion time of job i is defined as (11):

$$C_i = C_{iKi} \tag{11}$$

In both initial scheduling and rescheduling after the machine tool breakdowns, the objective is to minimize total processing time – makespan, defined as the maximum completion time of all jobs (12):

$$\min C_{\max} = \max_{i\in N} C_i \tag{12}$$

Minimizing makespan ensures global efficiency of the manufacturing system by reducing total production time while maintaining feasibility under dynamic disturbances.

2.1 Mathematical formulation of DFJSSP

The present subsection extends the formulation to incorporate dynamic machine breakdown events, thereby representing the DFJSSP. In modern manufacturing

environments, particularly those exposed to cyber-attacks, failures may often occur unexpectedly. Such disturbances affect ongoing operations, invalidate previously optimized schedules, and require immediate rescheduling.

The machine tool breakdown modeling adopted in this study follows the structural framework introduced in [35], in which disturbances are incorporated by updating job and machine release times. The machine breakdown is defined by a failed machine $r \in M$, a breakdown occurrence time t_{mb}, a failure duration t_{dur}, a job release time r_i and a machine tool release time r_k.

The failed machine is unavailable during the interval $[t_{mb},t_{mb}+t_{dur}]$. At the moment of failure, all operations in the system are classified according to the breakdown time t_{mb}. The operations are structured into the following three categories:

- completed operations: operations satisfying $C_{ij} \leq t_{mb}$ are considered completed prior to the disturbance; these operations remain unaffected by the breakdown, preserve their original scheduling order, and are excluded from any rescheduling process,
- unaffected ongoing operations: operations satisfying $S_{ij}<t_{mb}<C_{ij}$ and processed on machine tools $k \neq r$ are ongoing at the time of failure but remain unaffected; they continue processing without interruption and retain their previously assigned machine, tool, and tool orientation,
- affected operations: operations that satisfy $S_{ij}<t_{mb}<C_{ij}$ and that are processed on the failed machine tool $k = r$, as well as all the operations with $S_{ij} \geq t_{mb}$, are subject to the rescheduling; this group includes interrupted operations on failed machine tools and all operations that have not yet started, while their structural feasibility and scheduled start times are directly impacted by the breakdown event within the manufacturing system.

This classification ensures that the disturbance impacts only the relevant subset of decision variables, while maintaining maximum stability of the remaining schedule. To integrate the disturbance into the scheduling framework, the earliest time at which job i may resume processing is defined through an updated job release time r_i. For an operation O_{ij} previously assigned to machine tool k, the release time is defined as (13):

$$r_i = \begin{cases} C_{ij}, & k \neq r\,(\text{unaffected}), \\ t_{mb}, & k = r\,(\text{reassigned}), \\ t_{mb} + \tau_{dur}, & k = r\,(\text{resumed}). \end{cases} \tag{13}$$

Similarly, machine tool availability is updated through machine release time r_k (14):

$$r_k = \begin{cases} C_{prev}, & k \neq r \\ t_{mb} + t_{dur}, & k = r \end{cases} \tag{14}$$

where C_{prev} denotes the completion time of the last operation processed on machine tool k.

During rescheduling, the originally defined constraints continue to apply within the manufacturing system. Operation start times must continue to respect machine tool availability, transportation times, and sequence-dependent setup requirements, including tool and orientation adjustments. After the machine tool breakdown, the objective function remains the minimization of makespan, while feasibility conditions are modified through updated release times and temporary machine unavailability. Once release times are updated, a structured rescheduling is required to preserve completed operations, consistently update system states, and generate a feasible rescheduled process plan. By embedding the updated release times r_i and r_k directly into the optimization process, the scheduling algorithm dynamically adapts machine tool assignments and sequencing decisions to the dynamic conditions of the manufacturing system.

2.2 Rescheduling strategy

Once the disturbance has been formally integrated into the scheduling framework by updating job and machine release times, a systematic rescheduling is required to generate a feasible and efficient schedule following a machine breakdown. At the breakdown time t_{mb}, the previously generated schedule is analyzed and all operations are categorized according to the classification defined in Subsection 2.1. Completed operations and unaffected ongoing operations are preserved and remain fixed in the schedule. Only affected operations, including interrupted operations on the failed machine and operations scheduled to start after t_{mb}, are subject to the Ω rescheduling process. This strategy ensures schedule stability while minimizing structural changes in the manufacturing system.

Let the multi-string chromosome representation of a solution be defined as (15):

$$X = \left(X^{sc}, X^{pp}, X^{m}, X^{tool}, X^{tad} \right) \tag{15}$$

where the strings encode operation sequencing, process plan selection, machine tool assignment, tool allocation, and tool orientation, respectively. Each string corresponds to a specific decision layer, while the elements within each string represent genes associated with the corresponding operations. Under machine tool breakdown, genes corresponding to completed and unaffected operations remain fixed, while only the subset associated with affected operations is subject to rescheduling. The effective decision vector, therefore, becomes: $X^{MB} \subset X$, which reduces the original search space Ω to a smaller feasible subspace Ω^{MB}.

During rescheduling, start times must satisfy operational constraints, machine capacity constraints, transportation times, tool and orientation setup times. The breakdown modifies feasibility exclusively through updated release times r_i and r_k, as well as through the temporary unavailability of the failed machine r during the interval $[t_{mb},t_{mb}+t_{dur}]$.

For each affected operation O_{ij}, the feasible machine set is dynamically adjusted. If alternative machine tools exist within M_{ij}, reassignment is allowed as long as machine release constraints are satisfied. If no alternative machine tool is available, the operation must be

postponed until the failed machine tool becomes operational again, ensuring (16):

$$S_{ij}^{new} \geq t_{mb} + t_{dur} \tag{16}$$

The final schedule is obtained by combining the preserved subset of the original schedule M^{freeze} (representing the fixed part of the schedule unaffected by the machine breakdown) with the newly optimized subset of all affected operations generated from X^{MB} (17):

$$M^{final} = Decode\left(X^{MB}, M^{freeze}\right) \tag{17}$$

where the decoding procedure recalculates all operation start and completion times. During decoding, all defined constraints and times, as well as the temporary unavailability of the failed machine, are simultaneously enforced. This guarantees global feasibility while maintaining consistency with completed operations.

The proposed partial rescheduling mechanism is based on two complementary principles: preservation of completed operations and elimination of infeasible process alternatives. Completed and unaffected operations remain fixed, thereby maintaining schedule stability and avoiding unnecessary modifications within the manufacturing system. At the same time, process plan alternatives incompatible with the fixed operations are removed from the feasible set. As a result, the optimizer explores only process plan continuations consistent with previously realized operations, thereby confining the metaheuristic to operate within a constrained subspace $\Omega^{MB} \subset \Omega$. Although the rescheduling framework is more elaborate from an implementation, the optimization process itself explores a smaller and structurally restricted search space compared to complete rescheduling from scratch. This is particularly relevant in cyber-secure manufacturing environments, where major schedule changes may introduce instability across interdependent resources.

3. INTEGRATED MULTI-STRING ENCODING AND DECODING FRAMEWORK

The DFJSSP defined in Section 2 requires a representation capable of simultaneously modeling operation sequencing, alternative process plan selection, machine assignment, tool allocation, tool access directions (TAD), transportation times, and sequence-dependent tool and setup change times. Since these decision layers are strongly interdependent, a decoupled or single-string representation is insufficient. To preserve structural feasibility, each candidate solution is represented using an integrated multi-string encoding scheme. This section formally defines both the encoding structure of candidate solutions and the decoding process used to generate feasible schedules.

3.1 Integrated encoding

Let n denote the number of jobs and let q denote the maximum number of operations per job. For each job $i \in N = \{1,...,n\}$ the real number of operations is $K_i \leq q$ and the set of operations is defined as (1). To ensure a uniform chromosome structure across all individuals, strings of fixed lengths $L = n \cdot q$ are used. If $K_i < q$, the remaining positions are filled with zero values. Each candidate solution is encoded according to (15). For a population of size P, the strings X^{sc}, X^{m}, X^{tool}, X^{tad} are of size $P \times L$, whereas X^{pp} is of size $P \times n$. The fixed length structure $L = n \cdot q$ ensures uniform chromosome size for all individuals, which simplifies the implementation of crossover, mutation, and other update operators. Zero-value genes guarantee structural consistency without affecting feasibility, since they are explicitly ignored during decoding. The same integrated representation is used for all metaheuristic algorithms considered in this research, while only the search dynamics differ across them.

The operation sequencing string is defined as (18):

$$\begin{aligned} &X^{sc} = \left[s_1, s_2, ..., s_p, ..., s_L\right] \\ &s_p \in \{0,1,2,...,n\} \end{aligned} \tag{18}$$

When $s_p = i$ and $s_p > 0$ the decoder schedules the next unprocessed operation O_{ij} of job i. If $s_p = 0$, the position corresponds to a zero-value gene and is ignored during decoding. Each job index i appears exactly K_i times in X^{sc}, while the remaining positions are zeros.

The process plan string is given in (19):

$$\begin{aligned} &X^{pp} = \left[pp_1, pp_2, ..., pp_p, ..., pp_n\right] \\ &pp_i \in \{1,...,P_i\} \end{aligned} \tag{19}$$

and pp_p denotes the number of alternative process plan available for job i.

Machine, tool, and orientation assignments are defined by (20)-(22), respectively:

$$\begin{aligned} &X^{m} = \left[m_1, m_2, ..., m_p, ..., m_L\right] \\ &m_p \in \{0,...,m\} \end{aligned} \tag{20}$$

$$\begin{aligned} &X^{tool} = \left[t_1, t_2, ..., t_p, ...t_L\right] \\ &t_p \in \{0,...,\tau\} \end{aligned} \tag{21}$$

$$\begin{aligned} &X^{tad} = \left[d_1, d_2, ..., d_p, ..., d_L\right] \\ &d_p \in \{0,...,\delta\} \end{aligned} \tag{22}$$

where non-zero values of m_p represents the index of the selected machine tool for processing the corresponding operation, non-zero values of t_p denotes the selected cutting tool from the available set, and non-zero values of d_p represent tool access directions, corresponding to the admissible orientations {+x,-x,+y,-y,+z,-z}. These directions are internally mapped to integer values within the encoding. For any position $s_p = 0$, the corresponding values satisfy $m_p = t_p = d_p = 0$, and the position is ignored in the decoding phase.

3.2 Integrated decoding

The schedule is generated through a forward state-based decoding procedure that sequentially evaluates positions

$p = 1,\ldots,L$ of the sequencing string X^{sc}. If $s_p \neq 0$, the value of s_p denotes job i; otherwise, the position is disregarded. The decoder schedules the next operation O_{ij} of job i, where the operation index j is determined by counting the previous occurrences of job i in the sequencing string X^{sc}. The decoding method generates the schedule sequentially while updating the system state after each assignment. Throughout the decoding process, relevant information about jobs and machine tools is maintained. The job state includes the current ready time, previously assigned machine, last used tool and tool orientation, while the machine state updates the latest completion time and occupied time intervals.

At each position of the sequencing string X^{sc}, the corresponding operation O_{ij} is identified, while the associated machine $k=m_p$, tool t_p and orientation d_p are retrieved from the encoding. Feasibility is verified by ensuring that the selected resources belong to the feasible sets ($k \in M_{ij}, t_p \in T_{ij}, d_p \in D_{ij}$), defined for the corresponding operation.

If the operation is not the first operation of the job, transportation time from the machine assigned to the previous operation is evaluated (T_{ij}^{trans}). Otherwise, no transportation time is introduced. Tool and setup change times ($t_{change}^{tool}, t_{change}^{setup}$) are then determined based on differences between previously used and newly assigned resources.

The earliest feasible start time of operation O_{ij} is obtained by simultaneously considering job readiness and machine tool availability. Specifically, the decoder compares the completion time of the previous operation of the same job (including transportation and required tool and orientation adjustments) with the release time of the selected machine tool (including resource adjustments on machine k). The earliest start time is then determined as (23):

$$S_{ij} = \max\left(C_{i,j-1} + T_{ij}^{trans} + \Gamma_{job}(\cdot), r_k + \Gamma_{machine}(\cdot)\right) \quad (23)$$

where $\Gamma_{job}(\cdot)$represents the changeover time evaluated with respect to the previously completed operation of the same job, while $\Gamma_{machine}(\cdot)$denotes the changeover time evaluated relative to the last operation scheduled on machine k.

Once the start time is determined, the corresponding completion time is evaluated, and both job and machine information are updated before proceeding to the next position in the sequencing string. This sequential evaluation continues until all operations are scheduled, after which the makespan $C_{\max}$ is obtained as the maximum completion time across all jobs [36].

4. OPTIMIZATION ALGORITHMS

To ensure methodologically consistent comparison of different optimization strategies within the DFJSSP, a unified metaheuristic framework is adopted. The proposed framework integrates the Genetic Algorithm (GA), Particle Swarm Optimization (PSO), and the newly developed Hippopotamus Optimization (HO) algorithm. The overall decision space is decomposed according to the adopted multi-string encoding (24):

$$X_h = \left(X_h^{sc}, X_h^{pp}, X_h^{m}, X_h^{tool}, X_h^{tad} \right) \quad (24)$$

In the proposed approach, all considered metaheuristic algorithms operate on a population of N candidate solutions, indexed by $h = 1,2,\ldots,N$, where each index h corresponds to a distinct chromosome in the population.

Initially, GA is employed to optimize the sequencing and process plan encoding strings (X_h^{sc}, X_h^{pp}), generating feasible scheduling structures that satisfy the required processing order of operations and alternative process plan requirements. While GA efficiently generates admissible schedules in discrete search spaces, further improvements to machine assignment decisions require more detailed adjustments at the resource level. Therefore, for the remaining decision variables ($X_h^{m}, X_h^{tool}, X_h^{tad}$) three different optimization strategies, GA, PSO, and the proposed discrete HO, are comparatively analyzed under the same representation to ensure a fair and consistent performance evaluation.

4.1 Genetic Algorithm

Genetic Algorithm (GA) is a population-based metaheuristic which has been extensively applied to solve complex engineering optimization problems [37]. Moreover, it has been extensively applied to FJSSP and DFJSSP, particularly for problems involving alternative process plans and multi-string encodings. GA is particularly suitable due to its population-based search mechanism, robustness in discrete combinatorial spaces, and natural compatibility with chromosome-based representations. In this research paper, GA is applied following the same approach as in our previous studies [38] and in accordance with established approaches from the literature.

GA is employed to generate the initial feasible scheduling plan by optimizing X_h^{sc} and X_h^{pp}, ensuring that operational constraints are satisfied and that each job is assigned one of its available alternative process plans. The GA is subsequently applied to optimize the resource assignment strings ($X_h^{m}, X_h^{tool}, X_h^{tad}$). The evolutionary process follows the standard GA method (25):

$$\begin{aligned} &Population^{g} \rightarrow Selection \rightarrow \\ &Crossover \rightarrow Mutation \rightarrow \\ &Population^{g+1} \end{aligned} \quad (25)$$

To prevent the degradation of solution quality, an elitist strategy is incorporated. The chromosome with the best objective function is directly copied into the next generation, formally expressed as (26):

$$x_{best}^{g} \in Population^{g+1} \quad (26)$$

where X_{best}^{g} denotes the chromosome with the minimum objective function value in generation g. This

approach ensures that the best solution found so far is retained in the evolutionary process while allowing further improvements in subsequent generations. By iteratively applying crossover and mutation operators, GA generates a diverse yet feasible set of candidate solutions, which subsequently serve as fixed scheduling frameworks for the resource optimization phase.

4.2 Particle Swarm Optimization

Particle Swarm Optimization (PSO), originally introduced by [39], is a population-based algorithm inspired by social behavior in biological swarms and has been widely applied in various engineering applications [40,41]. PSO has demonstrated competitive performance in various scheduling and combinatorial optimization problems due to its simple structure, fast convergence, and efficient balance between exploration and exploitation [24,25]. In PSO, candidate solutions are updated by iteratively adjusting particle velocity and position. These updates are guided by both the particle's personal best solution and the global best solution found by the swarm, enabling coordinated exploration of the search space and progressive improvement of scheduling solutions.

Unlike GA, which operates in discrete generations indexed by *g*, PSO evolves through iterative time steps indexed by *t*. Let $t = 1,2,\ldots, T$ denote the iteration index of the PSO algorithm. For each particle *h*, velocity and position updates are defined as (27) and (28), respectively:

$$V_h^{t+1} = wV_h^t + c_1 r_1 \left(pbest_h - X_h^t \right) + \\ +c_2 r_2 \left(gbest - X_h^t \right) \tag{27}$$

$$X_h^{t+1} = X_h^t + V_h^{t+1} \tag{28}$$

where:

- w is the inertia weight,
- c_1 and c_2 are cognitive and social acceleration coefficients,
- $r_1 \in (0,1)$ and $r_2 \in (0,1)$,
- $Pbest_h$ is the local best solution found by particle h,
- gbest is the global best solution in the swarm.

In the proposed framework, PSO is implemented exclusively on the resource decision variables ($X_h^m, X_h^{tool}, X_h^{tad}$), while X_h^{sc} and X_h^{pp} remain fixed after GA optimization. Continuous position updates are mapped to admissible machine, tool, and orientation selections through feasibility recovery strategies, ensuring consistency with the operational constraints defined by X_h^{sc} and X_h^{pp}.

4.3 Hippopotamus Optimization

Although GA and PSO have demonstrated strong performance in scheduling optimization, their exploration-exploitation balance may become limited in highly dynamic environments affected by disturbances such as machine tool breakdowns. In this context, maintaining both solution diversity and convergence stability becomes essential. For this reason, a newly adapted optimization strategy, the discrete Hippopotamus Optimization (HO) algorithm, is introduced.

The Hippopotamus Optimization algorithm, originally proposed in [27], is a biologically inspired population-based metaheuristic designed to balance exploration and exploitation through a three-phase behavioral structure. The algorithm models characteristic hippopotamus behaviors (coordinated group movement, defensive response to predators, and retreat toward safe areas) through mathematical operators that guide the search process. These strategies promote global exploration in early iterations and gradual intensification toward high-quality solutions as the search progresses.

In its original formulation, HO operates in a continuous search space using real-valued vectors. However, the DFJSSP addressed in this research paper represents a discrete and combinatorial optimization problem. Therefore, the standard HO framework is adapted to operate within a structured multi-string encoding scheme that simultaneously represents machine assignment, tool selection, and tool orientation decisions. Although position updates are performed using the arithmetic rules of the original HO model, all candidate solutions are discretized and adjusted to admissible values before fitness evaluation to ensure resource feasibility. The proposed adaptation performs coordinated optimization of all resource decision variables within a unified representation. This adaptation enables HO to operate effectively in complex dynamic scheduling environments while maintaining convergence stability and solution quality. The mathematical formulation of the three-phase updating mechanism adapted for the proposed discrete scheduling framework is presented below.

Following the general principles of HO, the search process is performed over a population of *N* individuals X_h, where $h = 1,2,\ldots, N$. Each individual represents a resource-level decision vector (29):

$$X_h = \left(X_h^m, X_h^{tool}, X_h^{tad} \right) \tag{29}$$

where each segment contains *L* components corresponding to the total number of operations. In accordance with the sets defined in Section 2, the decision variables satisfy (30):

$$X_m^h(k) \in M \quad X_h^{tool}(k) \in T, \\ X_h^{tad}(k) \in D \\ k = 1,\ldots, L \tag{30}$$

where *M*, *T*, and *D* denote the sets of machines, tools, and tool orientations in the manufacturing system.

Since the updating rules of the original HO algorithm are defined in a continuous search space, each updated component is adjusted to its admissible index range. The lower and upper bounds are defined component-wise as (31) and (32), respectively:

$$lb_i = 1, \forall j \tag{31}$$

where *m*, τ, and δ denote the total number of machines, tools, and tool approach directions in the manufacturing

system, respectively. The first L components correspond to machine assignments, the next L components to tool selections, and the final L components to tool approach directions. After each position update, boundary control is applied to ensure that all components remain within their admissible ranges.

The quality of each individual is evaluated through the decoding-based objective function (33):

$$f_h = F\left(X_h\right) \tag{33}$$

where $F(\cdot)$ denotes the makespan obtained after comp–lete schedule decoding under operational and resource constraints. At iteration t, the dominant hippopotamus is defined as (34):

$$D^t = \arg \min_{h \in [1,\ldots,N]} f_h \tag{34}$$

and serves as the reference for the three-phase updating mechanism of the proposed HO algorithm.

Phase 1 – Hippopotamus Position Update (Exploration)

In accordance with the original HO framework, Phase 1 performs global exploration by updating only the first half of the population, $h = 1,\ldots,\frac{N}{2}$. The domi–nant hippopotamus D^t denotes the best solution at iteration t. Phase 1 consists of two behavioral subphases inspired by male and female hippopotamus movement.

Male subphase (Dominant attraction)

In the male subphase, each individual moves toward the dominant hippopotamus. The update rule is defined as (35):

$$X_h^{t+1} = X_h^t + r_1\left(D^t - I_1 X_h^t\right) \tag{35}$$

where $r_1 \in (0,1)$ and $r_2 \in (0,1)$.

This movement promotes intensification around the dominant solution while preserving stochastic diversity.

Female subphase (Group influence and diversification)

In the female subphase, a local group mean MG_h is evaluated from a randomly selected hippopotamus. An exponentially decreasing control mechanism is introdu–ced to regulate the transition between guided movement and diversification during the search process. In the early stages of the search, the exponential factor rema–ins high, guiding movement toward the dominant solution. As it decreases in later iterations, diversifi–cation mechanisms are activated (36):

$$r_5 \in (0,1) \tag{36}$$

where $r_1 \in (0,1)$ and $r_2 \in (0,1)$.

In later stages, diversification is enhanced either by moving relative to the group mean or by random reinitialization within admissible bounds. After each update, boundary control is applied to ensure feasibility, and the new position is accepted only if it improves the objective function value. Overall, Phase 1 integrates dominant attraction (male behavior) and collective group adjustment (female behavior), promoting strong global exploration while maintaining convergence stability within the discrete scheduling framework.

Phase 2 – Defensive behavior against predator (Exploration)

Phase 2 models the defensive behavior of hippo–potamuses when confronted with a predator. A predator position is randomly generated within the admissible search space (37):

$$Predator_j = lb_j + r_3\left(ub_j - lb_j\right) \tag{37}$$

where $r_3 \in (0,1)$.

In contrast to simplified implementations, predator fitness is explicitly evaluated using the objective func–tion based on decoding. In the proposed discrete sche–duling framework, the predator position is evaluated using the sequencing and process plan components of the currently best individual, ensuring consistent eva–luation within the structured multi-string representation.

This phase updates the second half of the population $h = \left\lceil \frac{N}{2} \right\rceil + 1,\ldots,N$. The distance between the h-th hippopotamus and the predator is defined as (38):

$$D_h = \left| Predator - X_h^t \right| \tag{38}$$

where the absolute value is applied component-wise.

A Lévy-distributed random-step RL is introduced to simulate sudden and irregular predator movement. If the predator exhibits better fitness value than the current hippopotamus, indicating a stronger threat, a more aggressive defensive movement is performed (39):

$$X_h^{t+1} = \left(RL + Predator\right) + F \cdot \frac{1}{D_h} \tag{39}$$

where F is constructed from uniformly distributed ran–dom parameters, following the original HO formulation. Otherwise, a moderated defensive reaction is applied (40):

$$X_h^{t+1} = \left(RL + Predator\right) + F \cdot \frac{1}{2D_h + r'} \tag{40}$$

where $r' \in (0,1)^m$ is a uniformly distributed random ve–ctor introduced to ensure numerical stability and inc–rease stochastic variability.

After boundary control is applied to preserve feasi–bility, the new position is accepted only if it improves the objective function value. Overall, Phase 2 comple–ments the cooperative exploration of Phase 1 by intro–ducing controlled stochastic modifications driven by predator–prey interaction, thereby strengthening global search diversity and reducing the risk of convergence to local minima.

Phase 3 – Escaping from predator (Exploitation)

Phase 3 models the escaping behavior of hippopo–tamuses when defensive reaction is insufficient. In this situation, the hippopotamus attempts to move to a

nearby, safer position, representing a localized exploi–tation step in the search space. To simulate this beha–vior, a local search region is defined around the current position. The local lower and upper bounds are progres–sively reduced as iterations advance (41):

$$lb_j^{local} = \frac{lb_j}{t},\ ub_j^{local} = \frac{ub_j}{t} \tag{41}$$

where t denotes the current iteration.

A new candidate position is generated in the vicinity of the current solution (42):

$$X_h^{t+1} = X_h^t + r_5\left(lb^{local} + s_1\left(ub^{local} - lb^{local}\right)\right) \tag{42}$$

where $r_5 \in (0,1)$ and s_1 is a stochastic coefficient selected from predefined random scenarios, following the original HO formulation.

This mechanism enables controlled local modifica–tion around the current solution, thereby strengthening exploitation capability. After boundary control is applied, the new position replaces the current one only if it improves the objective function value. Overall, Phase 3 enhances local search improvement by guiding individuals toward safer neighboring regions, comple–menting the exploratory mechanisms of Phases 1 and 2 and supporting convergence stability in the discrete scheduling framework.

Collectively, the three phases establish a structured balance among cooperative exploration (Phase 1), adap–tive predator–prey adjustment (Phase 2), and localized exploitation (Phase 3), forming a unified discrete opti–mization framework developed for dynamic environ–ments. The bounding mechanism ensures that all up–dated components remain within their admissible dis–crete domains, thereby preserving the structural feasi–bility of the multi-string encoding without requiring additional repair strategies. The computational effort of the proposed discrete HO algorithm is primarily deter–mined by the decoding-based fitness evaluation, which remains consistent with the complexity of the DFJSSP model. Through this coordinated design, the proposed HO framework maintains search diversity, enhances robustness to premature convergence, and ensures stable search behavior in highly dynamic manufacturing environments affected by machine tool breakdowns.

5. EXPERIMENTAL RESULTS AND COMPARATIVE ANALYSIS

To evaluate the effectiveness and robustness of the pro–posed multi-string HO-based scheduling framework, comprehensive experiments were conducted in both static and dynamic manufacturing environments. This analysis evaluates final solution quality, convergence behavior, stability, and recovery capability under mac–hine tool breakdowns.

To ensure methodological consistency and a fair comparison, all considered algorithms (GA, PSO, and HO) were implemented in MATLAB® using the same discrete multi-string encoding and decoding scheme. Each candidate solution is evaluated under the same scheduling framework, ensuring that any observed performance differences arise exclusively from the respective algorithms' search mechanisms.

The experimental study is divided into two main parts. The first part considers the static scheduling scenario, where no disturbances are introduced. This setup serves as a reference evaluation to assess optimi–zation performance, convergence behavior, and algo–rithm stability under stable conditions. The second part focuses on dynamic rescheduling under machine tool breakdowns. In this scenario, a disturbance interrupts the ongoing manufacturing process, requiring partial rescheduling, while ensuring that completed operations remain fixed. This experimental setup allows the evalu–ation of algorithm adaptability, recovery speed, and ro–bustness under real manufacturing conditions.

Performance evaluation includes the best and average makespan values, the standard deviation, con–vergence analysis, and stability comparison using box–plot distributions. Additionally, statistical validation tests were applied to verify the significance of observed differences.

5.1 Experimental setup

The considered manufacturing system includes three jobs (1, 3, and 8) selected from dataset [42], which are processed on nine machine tools. The dataset is deve–loped for research on smart manufacturing and process optimization, with a particular focus on cybersecurity aspects. It comprises 3D models of representative parts (jobs) together with their geometric and functional cha–racteristics, as well as alternative process plan networks describing feasible manufacturing routes across diffe–rent machines, tools, and Tool Access Directions (TADs). Furthermore, it provides detailed technical specifications and corresponding operation processing times for each alternative.

Accordingly, the scheduling model incorporates al–ternative process plans with operation sequencing, machine tool assignment, tool allocation, and tool ori–entation constraints, as well as transportation and tool setup times. Transportation times are defined by a sym–metric machine-to-machine time matrix *TT*, while tool and setup change times are predefined and embedded in the evaluation function. The objective function in both static and dynamic instances is the minimization of makespan.

Three population-based metaheuristic algorithms are compared: Genetic Algorithm (GA), Particle Swarm Optimization (PSO), and the proposed discrete Hippo–potamus Optimization (HO). To maintain methodo–logical consistency, all algorithms perform within the same multi-string encoding and decoding structure.

For the initial scheduling problem without distur–bances, a population size of 40 individuals and 30 gene–rations was used, with 30 independent runs per algo–rithm. For the dynamic rescheduling, the population size and number of generations were set to 20, with 20 independent repetitions. The following parameters were applied without modification in both the initial scheduling and the rescheduling. Elitism was applied, with two elite individuals preserved per generation, while the crossover probability in the GA was 1. For

PSO, the inertia weight decreases linearly between 1.2 and 0.4, with acceleration constants $c_1 = c_2 = 2$. In the HO-based framework, the discrete predator phase adaptation incorporates an exponentially decreasing exploration factor and a Levy-based diversification strategy. All candidate solutions are constrained within the admissible search space and evaluated through the complete scheduling fitness function, which verifies the feasibility of machine, tool, and orientation assignments.

In the rescheduling phase, machine tool breakdowns are introduced on machines M2, M3, M5, and M7 during predefined time intervals. Completed operations remain fixed, while interrupted operations and operations scheduled after t_{mb} are rescheduled under updated machine availability constraints.

5.2 Static scheduling results

The optimal process plan generated after scheduling serves as a baseline evaluation of independent optimization performance without disturbances. The analysis focuses on convergence behavior, solution stability, and overall solution quality.

The best schedule obtained by HO achieves a makespan of 249s and the corresponding Gantt chart is shown in Figure 1. The optimal schedule demonstrates balanced machine utilization, without significant waiting time toward the end of the schedule. This confirms that the multi-string decoding preserves structural feasibility while enabling efficient search.

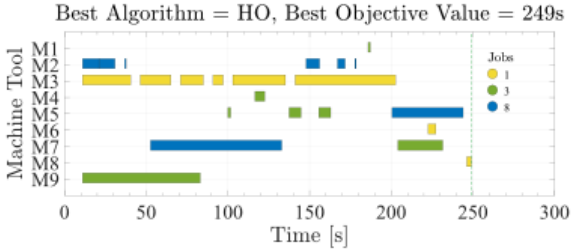

Figure 1. Gantt chart of the best scheduling solution

The convergence behavior averaged over 30 independent runs is presented in Figure 2.

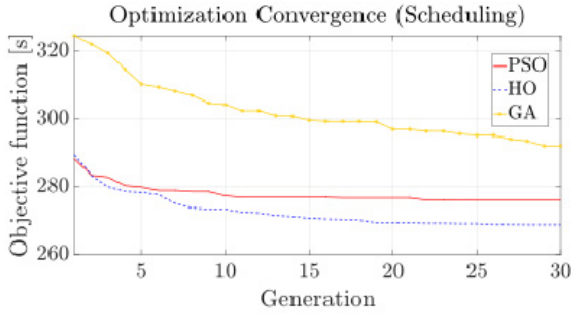

Figure 2. Convergence behavior in scheduling

HO exhibits the fastest early-stage convergence and maintains gradual improvement across generations. PSO demonstrates smoother but slower refinement and earlier stagnation, while GA continues gradual improvement but converges to a higher objective value, indicating weaker exploitation capability in this instance.

Stability analysis is illustrated in Figure 3. HO achieves the lowest median makespan, while PSO exhibits slightly narrower dispersion, although a single outlier indicates occasional instability. GA shows both a higher median and increased variability.

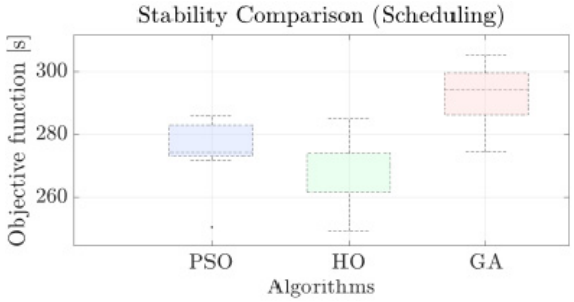

Figure 3. Stability comparison in scheduling

The numerical results summarized in Table 1 confirm that HO achieves the lowest average makespan, while PSO exhibits slightly lower performance variability. GA shows inferior performance in both average value and stability.

Table 1. Scheduling performance (30 runs)

Algorithm	Best [s]	Mean [s]	Std. Dev. [s]
GA	274.50	292.10	8.47
PSO	250.20	275.97	7.62
HO	249.00	268.75	9.52

Wilcoxon rank-sum tests indicate statistically significant improvement of HO over GA and PSO ($p < 0.05$), while the Friedman analysis confirms overall differences among the compared algorithms. However, the difference between HO and PSO is less significant in the scheduling, suggesting that the instance remains moderately easy under stable conditions.

5.3 Rescheduling results under machine breakdown

The dynamic rescheduling introduces four machine tool failures with the following downtime intervals: M2 (50s–100s), M3 (50s–90s), M5 (70s–130s), and M7 (140s–180s), creating sudden feasibility changes and interdependent effects across jobs. The purpose of this experiment is to evaluate algorithm robustness, adaptability, and recovery capability.

The best rescheduled solution is also obtained by HO with a makespan of 301.8s. The corresponding Gantt chart is shown in Figure 4. The optimal rescheduled process plan clearly reflects partial solution decoding: operations completed prior to the disturbance remain fixed, while the affected operations are reassigned in accordance with the updated machine availa–bility constraints. This result confirms the effectiveness of the proposed rescheduling strategy in maintaining schedule feasibility under multiple machine tool breakdowns.

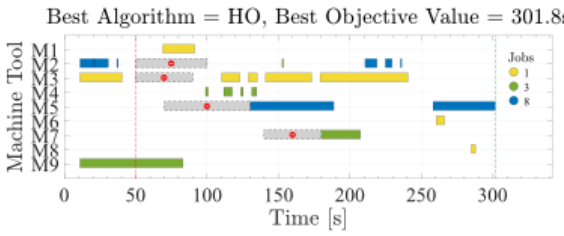

Figure 4. Gantt chart of the best rescheduling solution under machine breakdowns

The convergence behavior represents the mean performance across 20 independent runs and is presented in Figure 5. HO demonstrates efficient adaptation in early generations and maintains stable improvement. PSO exhibits slower adaptation, while GA converges to intermediate values but remains inferior to HO.

The stability comparison is illustrated in Figure 6, while the detailed numerical indicators are summarized in Table 2. The results further confirm that HO achieves lower variability across independent runs, indicating more consistent performance under the same experimental conditions.

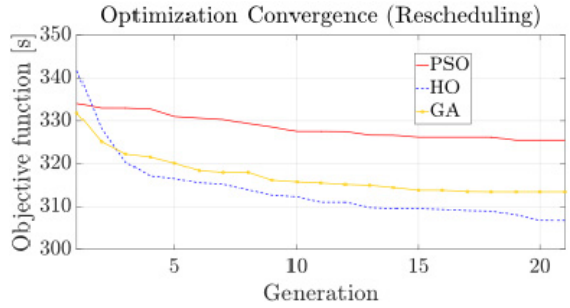

Figure 5. Convergence behavior under machine breakdowns

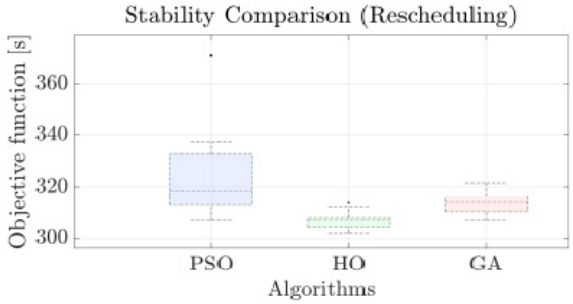

Figure 6. Stability comparison under machine breakdown

HO not only achieves the lowest average makespan (306.76s) but also demonstrates significantly lower dispersion (3.63s) than PSO. Several PSO runs yield significantly higher makespan values, increasing solution variability and indicating sensitivity to dynamic disturbances.

Table 2. Rescheduling performance under machine breakdowns (20 runs)

Algorithm	Best [s]	Mean [s]	Std. Dev. [s]
GA	306.80	313.45	4.64
PSO	306.80	325.49	21.33
HO	301.80	306.76	3.63

Wilcoxon rank-sum tests reveal statistically significant improvements of HO over PSO ($p = 0.000014$) and GA ($p = 0.000090$). The Friedman test further confirms overall differences among algorithms ($p = 0.000033$). Bonferroni corrected post-hoc comparisons validate the superiority of HO over both PSO ($p = 0.000046$) and GA ($p = 0.002316$), whereas no significant difference is observed between PSO and GA during rescheduling after multiple machine tool breakdowns.

6. DISCUSSION

The experimental evaluation reveals different behaviour of the algorithms under static and dynamic conditions. In the scheduling, all three metaheuristic algorithms achieve high-quality feasible solutions, with HO and PSO exhibiting comparable convergence behavior and final makespan values. Although HO obtains the best overall solution and slightly better average performance, the difference in solution quality between HO and PSO remains limited under stable feasibility conditions, indicating that multiple population-based strategies can effectively explore the search space when constraints remain unchanged.

However, the dynamic rescheduling under machine tool breakdowns introduces significantly stronger differentiation. The occurrence of multiple machine downtime intervals induces sudden changes in feasibility and structural discontinuities in the search space. Under such disturbances, algorithms that rely primarily on gradual solution improvement adapt less efficiently when global feasibility conditions change. PSO exhibits increased variability, largely driven by the global-best solution, whereas GA converges to lower-quality solutions with limited adaptive flexibility.

In contrast, the discrete HO framework maintains controlled exploration through its predator-phase dynamics, enabling the adaptive reallocation of search effort immediately after the disturbance. The integrated multi-string representation further supports simultaneous modification of machine, tool, and orientation decisions, enabling consistent rescheduling of affected operations without excessive solution changes. The significantly lower variance observed for HO indicates that its performance is not only superior in absolute terms but also stable under disturbance conditions.

Overall, the results indicate that while established metaheuristics such as GA and PSO remain competitive in static manufacturing environments, the HO-based approach demonstrates clear advantages in rescheduling after disturbances. The combination of adaptive exploration mechanisms and structurally consistent encoding enhances recovery efficiency and robustness, confirming the suitability of the proposed framework for DFJSSP, where resource availability may change unexpectedly.

7. CONCLUSION AND FUTURE RESEARCH

This research paper addressed the Dynamic Flexible Job Shop Scheduling Problem (DFJSSP) under multiple machine tool breakdowns, explicitly considering both physical failures and cyber-induced disturbances, through an integrated multi-string discrete optimization framework. A discrete adaptation of the Hippopotamus Optimization (HO) algorithm was developed and systematically compared with established metaheuristics, GA and PSO, within both static and dynamic manufacturing environments. In addition, a structured partial rescheduling strategy was introduced to maintain completed operations fixed while updating only the affected ones after breakdown, thereby minimizing unnecessary schedule modifications and enhancing operational stability.

The experimental verification demonstrated that multiple population-based approaches can achieve high-quality solutions in undisturbed environments. While HO achieved the best overall makespan and favorable

convergence behavior, performance differences between HO and PSO remained limited under unchanged feasibility conditions.

However, rescheduling under machine tool breakdowns demonstrated significantly greater performance divergence. The introduction of multiple downtime intervals led to sudden changes in feasibility and structural discontinuities in the solution space. Under such conditions, HO consistently achieved the lowest average makespan and significantly lower variability across independent runs. Statistical validation through the Wilcoxon and Friedman tests confirmed the superiority of the proposed approach in a dynamic manufacturing environment.

The improved robustness of HO is mainly associated with two structural components. First, the predator-phase exploration strategy, reinforced by time-adaptive exploration control and Lévy-flight-based exploration dynamics, enables controlled diversification following disturbance. Second, the integrated multi-string encoding allows coordinated modification of machine assignment, tool allocation, and orientation decisions, preserving feasibility while enabling efficient rescheduling.

Overall, the results indicate that while established metaheuristic algorithms remain competitive during scheduling, the proposed discrete HO framework improves adaptive response capability in rescheduling after the disturbances. The method demonstrates strong adaptability, structural stability, and recovery efficiency, which enhances its suitability for dynamic manufacturing systems exposed to unexpected resource unavailability.

Future research may further explore the applicability of alternative biologically inspired optimization algorithms and perform comprehensive comparative analyses to better understand their performance in dynamic environments. In addition, the proposed framework can be extended toward multi-objective optimization, larger-scale problem instances, and tighter integration with manufacturing systems exposed to cyber-induced disturbances.

ACKNOWLEDGMENT

This research was supported by the Science Fund of the Republic of Serbia, Grant No. 17801, Cybersecurity of Motion Control Systems in Industry 4.0 - MCSecurity, as well as by the Ministry of Science, Technological Development, and Innovations of the Serbian Government under Contract No. 451-03-33/2026-03/200105 and 451-03-34/2026-03/200105.

REFERENCES

[1] Dauzère-Pérès, S., Ding, J., Shen, L. Tamssaouet, K.: The flexible job shop scheduling problem: A review, European Journal of Operational Research, Vol. 314, No. 2, pp. 409-432, 2024. https://doi.org/10.1016/j.ejor.2023. 05.017

[2] Guliashki, V., Kirilov, L., Marinova, G.: Methods and Algorithms for Flexible Job Shop Scheduling – A State of the Art, Cybernetics and Information Technologies, Vol. 25, No. 2, pp. 3-33, 2025. https://doi.org/10.2478/cait-2025-0009

[3] Hajariwala, D.C., Patil, S.S., Patil, S.M.: A Review of Metaheuristic Algorithms for Job Shop Scheduling, Engineering Access, Vol. 11, No. 1, pp. 65-91, 2025. https://ph02.tci-thaijo.org/index.php/mijet/article/view/254023

[4] Zhang, W., Bao, X., Hao, X. Gen, M.: Metaheuristics for multi-objective scheduling problems in Industry 4.0 and 5.0: A state-of-the-arts survey, Frontiers in Industrial Engineering, Vol. 3, 1540022, 2025. https://doi.org/10.3389/fieng.2025.1540022

[5] Gui, L., Li, X., Zhang, Q., Gao, L.: Domain Knowledge Used in Meta-Heuristic Algorithms for the Job-Shop Scheduling Problem: Review and Analysis, Tsinghua Science and Technology, Vol. 29, No. 5, pp. 1368-1389, 2024. https://doi.org/10.26599/TST.2023.9010140

[6] Momenikorbekandi, A., Kalganova, T.: Intelligent Scheduling Methods for Optimisation of Job Shop Scheduling Problems in the Manufacturing Sector: A Systematic Review, Electronics, Vol. 14, No. 8, 1663, 2025. https://doi.org/10.3390/electronics14081663

[7] Li, J., Li, S., He, P. and Li, H.: A multi objective collaborative reinforcement learning algorithm for flexible job shop scheduling, Scientific Reports, Vol. 15, 22838, 2025. https://doi.org/10.1038/s41598-025-03681-6

[8] Jiang, M., Wei, H., Peng, D. and Chen, S.: An Improved NSGA-II Method for Solving Multi-objective Flexible Job-shop Scheduling Problems, A preprint submitted to Research Square, 2025. https://doi.org/10.21203/rs.3.rs-5794286/v1

[9] Lv, Z., Zhao, Y., Kang, H., Gao, Z., Qin, Y.: An Improved Harris Hawk Optimization Algorithm for Flexible Job Shop Scheduling Problem, Computers, Materials & Continua, Vol. 78, No. 2, pp. 2337-2345, 2024. https://doi.org/10.32604/cmc.2023. 045826

[10] Asklany, A.M. et al.: Comparative Analysis of Metaheuristic Algorithms for Unconstrained Optimization Problems, Assiut University Journal of Multidisciplinary Scientific Research, Vol. 54, No. 2, pp. 358-385, 2025. DOI: 10.21608/aunj.2025.347716. 1112

[11] Yao, S., Guo, Y., Yang, B., Lv, Y., Ceccarelli, M., Dai, X., Carbone, G.: Single-objective flexible job-shop scheduling problem based on improved dung beetle optimization, STEM Education, Vol. 4, No. 3, pp. 299-327, 2024. https://dx.doi.org/10.3934/steme.2024018

[12] Ben Ali, K., Bechikh, S., Louati, A., Louati, H., Kariri, E.: Dynamic Job Shop Scheduling Problem with New Job Arrivals Using Hybrid Genetic Algorithm, IEEE Access, Vol. 12, pp. 85338-85354, 2024. https://doi.org/10.1109/ACCESS.2024.3401080

[13] Liu, S.Q., Ong, H.L., Ng, K.M.: Metaheuristics for minimizing the makespan of the dynamic shop scheduling problem, Advances in Engineering Software, Vol. 36, No. 3, pp. 199-205, 2005. https://doi.org/10.1016/j.advengsoft.2004.10.002

[14] Ngwu, C., Liu, Y. and Wu, R.: Reinforcement learning in dynamic job shop scheduling: a comprehensive review of AI-driven approaches in modern manufacturing, Journal of Intelligent Manufacturing, Vol. 37, pp. 1093-1108, 2026. https://doi.org/10.1007/s10845-025-02585-6

[15] Pu, Y., Li, F., Rahimifard, S.: Multi-Agent Rein–forcement Learning for Job Shop Scheduling in Dynamic Environments, Sustainability, Vol. 16, No. 8, 3234, 2024. https://doi.org/10.3390/su1608 3234

[16] Gan, X., Zuo, Y., Yang, G., Zhang, A., Tao, F.: Dynamic scheduling for dual-objective job shop with machine breakdown by reinforcement lea–rning, in: *Proceedings of the Institution of Mecha–nical Engineers*, Part B: Journal of Engineering Manufacture, Vol. 238, No. 1-2, pp. 3-17, 2024. https://doi.org/10.1177/09544054231167086

[17] Wu, R., Zheng, J., Yin, X.: Dynamic Scheduling for Multi-Objective Flexible Job Shops with Machine Breakdown by Deep Reinforcement Learning, Processes, Vol. 13, No. 4, 1246, 2025. https://doi. org/10.3390/pr13041246

[18] Al-Hinai, N., ElMekkawy, T.Y.: Robust and stable flexible job shop scheduling with random machine breakdowns using a hybrid genetic algorithm, In–ternational Journal of Production Economics, Vol. 132, No. 2, pp. 279-291, 2011. https://doi.org/10. 1016/j.ijpe.2011.04.020

[19] Xiong, J., Xing, L.N., Chen, Y.W.: Robust scheduling for multi-objective flexible job-shop problems with random machine breakdowns, Inter–national Journal of Production Economics, Vol. 141, No. 1, pp. 112-126, 2013. https://doi.org/10. 1016/j.ijpe.2012.04.015

[20] 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, Computers & Operations Research, Vol. 73, pp. 56-66, 2016. https://doi.org/10.1016/j.cor.2016.03.009

[21] Li, Y., He, Y., Wang, Y., Tao, F., Sutherland, J.W.: An optimization method for energy-conscious production in flexible machining job shops with dynamic job arrivals and machine breakdowns, Journal of Cleaner Production, Vol. 254, 120009, 2020. https://doi.org/10.1016/j.jclepro.2020.120009

[22] Tariq, A., Khan, S.A., But, W.H., Javaid, A., Shehryar, T.: An IoT-enabled real-time dynamic scheduler for flexible job shop scheduling (FJSS) in an Industry 4.0-based manufacturing execution system (MES 4.0), IEEE Access, Vol. 12, pp. 49653-49666, 2024. https://doi.org/10.1109/ACCESS.2024.3384252

[23] Liang, Z., Zhong, P., Zhang, C., Yang, W., Xiong, W., Yang, S., Meng, J.: A genetic algorithm-based approach for flexible job shop rescheduling prob–lem with machine failure interference, Eksploatacja i Niezawodność – Maintenance and Reliability, Vol. 25, No. 4, 171784, 2023. http://dx.doi.org/ 10.17531/ ein/171784

[24] Nouiri, M., Bekrar, A., Jemai, A., Trentesaux, D., Ammari, A.C. and Niar, S.: Two Stage particle swarm optimization to solve the Flexible job shop predictive scheduling problem considering possible machine breakdowns, Computers & Industrial Engineering, Vol. 112, pp. 595-606, 2017. https:// doi.org/10.1016/j.cie.2017.03.006

[25] Zarrouk, R., Bennour, I.E., Jemai, A.: Performance improvement of the particle swarm optimisation algorithm for the flexible job shop problem under machines breakdown, International Journal of Intelligent Engineering Informatics, Vol. 6, No. 3-4, pp. 396-416, 2018. https://doi.org/10. 1504/ IJIEI.2018.091876

[26] Wang, C., Chen, J., Xu, B., Liu, S.: A Discrete Improved Gray Wolf Optimization Algorithm for Dynamic Distributed Flexible Job Shop Scheduling Considering Random Job Arrivals and Machine Breakdowns, Processes, Vol. 13, No. 7, 1987, 2025. https://doi.org/10.3390/pr13071987

[27] Amiri, M.H., Mehrabi Hashjin, N., Montazeri, M., Mirjalili, S., Khodadadi, N.: Hippopotamus optimi–zation algorithm: a novel nature-inspired optimi–zation algorithm, Scientific Reports, Vol. 14, 5032, 2024. https://doi.org/10.1038/s41598-024-54910-3

[28] Mehta, P., Sait, S.M., Yildiz, B.S., Yildiz, A.R.: Enhanced hippopotamus optimization algorithm and artificial neural network for mechanical com–ponent design, Materials Testing, Vol. 67, No. 4, pp. 655-662, 2025. https://doi.org/10.1515/mt-2024-0514

[29] Han, T., Wang, H., Li, T., Liu, Q., Huang, Y.: MHO: A modified hippopotamus optimization algorithm for global optimization and engineering design problems, Biomimetics, Vol. 10, No. 2, 90, 2025. https://doi.org/10.3390/biomimetics10020090

[30] Wang, H., Binti Mansor, N.N., Mokhlis, H.B.: Novel hybrid optimization technique for solar photovoltaic output prediction using improved hippopotamus algorithm, Applied Sciences, Vol. 14, No. 17, 7803, 2024. https://doi.org/10.3390/app14177803

[31] Indiravathi, G., Rajesh, D.: Hippopotamus optimi–zation algorithm with long short-term memory for music genre classification, International Journal of Intelligent Engineering and Systems, Vol. 18, No. 1, pp. 805-816, 2025. https://doi.org/10.22266/ ijies2025.0229.57

[32] Abbas, S.M., et al.: Decision tree based on hippo–potamus optimization algorithm for securing IoT healthcare systems, Journal of Theoretical and Applied Infor–mation Technology, Vol. 103, No. 12, pp. 5345-5353, 2025. https://www.jatit.org/ volumes/Vol103 No12/26Vol103No12.pdf

[33] Petrović, M., Miljković, Z., Jokić, A.: A novel met–hodology for optimal single mobile robot sche–duling using whale optimization algorithm, Applied Soft Computing, Vol. 81, 105520, 2019. https:// doi.org/10.1016/j.asoc.2019.105520

[34] Petrović, M., Jokić, A., Miljković, Z., Kulesza, Z.: Multi-objective scheduling of a single mobile robot

based on the grey wolf optimization algorithm, Applied Soft Computing, Vol. 131, 109784, 2022. https://doi.org/10.1016/j.asoc.2022.109784

[35] Brenjo, K., Jokić, A., Petrović, M.: Dynamic Flexible Job Shop Scheduling Problem Based on Genetic Algorithm, in: Proceedings of the 40th International Conference on Production Engi–neering – ICPES 2025, pp. 243-254, 2025. https://doi.org/10.46793/ICPES25.247B

[36] Petrović, M., Miljković, Z., Babić, B.: Integration of Process Planning, Scheduling, and Mobile Robot Navigation Based on TRIZ and Multi-Agent Methodology, FME Transactions, Vol. 41, No. 2, pp. 120-129, 2013. https://www.mas.bg.ac.rs/_media/istrazivanje/fme/vol41/2/06_mpetrovic.pdf

[37] Pham, N. D.: Genetic Algorithm Application in Multi-Objective Optimization of Structural Para–meters and PID Controller Parameters on Bus Driver Seat's Suspension System, FME Transac–tions, Vol. 53, No. 3, pp. 426-436, 2025. https://doi.org/10.5937/fme2503426D

[38] Brenjo, K., Jokić, A., Petrović, M.: Optimization of Flexible Job Shop Scheduling with Machine Tool Breakdowns, in: Proceedings of the 10th International Conference Transport and Logistics – TIL 2025, pp. 29-35, 2025. http://dx.doi.org/10.46793/TIL2025.029B

[39] Kennedy, J., Eberhart, R.: Particle swarm optimi–zation, in: Proceedings of the IEEE International Conference on Neural Networks (ICNN'95), Vol. 4, pp. 1942-1948, 1995. https://doi.org/10.1109/ICNN.1995.488968

[40] Abe, A.: Non-Linear Control Technique of a Pen–dulum via Cable Length Manipulation: Application of Particle Swarm Optimization to Controller De–sign, FME Transactions, Vol. 41, No. 4, pp. 265-270, 2013. https://www.mas.bg.ac.rs/_media/ istrazivanje/fme/vol41/4/01_aabe.pdf

[41] Rehman, S., Khan, S. A., Alhems, L. M.: The Effect of Acceleration Coefficients in Particle Swarm Optimization Algorithm with Application to Wind Farm Layout Design, FME Transactions, Vol. 48, No. 4, pp. 922-930, 2020. https://doi.org/10. 5937/fme2004922R

[42] Brenjo, K., Petrovic, M.: Dataset of flexible process plans for representative parts produced within manufacturing system affected by cyber-attacks [Data set], Zenodo, 2025. https://doi.org/10.5281/zenodo.15131252

ДИНАМИЧКО ФЛЕКСИБИЛНО ТЕРМИНИРАЊЕ ТЕХНОЛОШКИХ ПРОЦЕСА УСЛЕД ОТКАЗА РАДА МАШИНА АЛАТКИ ПРИМЕНОМ АЛГОРИТМА ОПТИМИЗАЦИЈЕ ИНСПИРИСАНОГ ПОНАШАЊЕМ НИЛСКИХ КОЊА

К.З. Брењо, А.В. Јокић, М.М. Петровић

Динамичко флексибилно терминирање технолошких процеса услед отказа рада машина алатки представља један од сложених комбинаторно-оптимизационих проблема у савременим технолошким системима. У овом раду предложен је оригиналан приступ опти–мизацији технолошких процеса, који подразумева истовремено одређивање редоследа извођења опера–ција, доделу машина алатки, избор алата и оријен–тација алата, при чему је функција циља миними–зација укупног времена потребног за обраду свих делова (енгл. *makespan*). Развијен је јединствен начин представљања решења заснован на структури са више стрингова (енгл. *multi-string*), који омогућава исто–времено кодирање и декодирање свих релевантних нивоа одлучивања. У циљу оптимизације планова терминирања, имплементирана су три биолошки инс–пирисана метахеуристичка алгоритма: генетички ал–горитам (енгл. *Genetic Algorithm – GA*), алгоритам инспирисан интелигенцијом роја честица (енгл. *Particle Swarm Optimization – PSO*) и алгоритам инспирисан понашањем нилских коња (енгл. *Hippo–potamus Optimi–zation – HO*). Уведена је стратегија ретерминирања на основу које операције завршене пре тренутка настанка поремећаја остају непроме–њене, док се операције на које су утицали откази машина алатки поново терминирају – ретерминирају. Добијени експериментални резултати указују да пре–дложени интегрисани приступ ефикасно решава ути–цај динамичких поремећаја и значајно унапређује пе–рформансе технолошког система. Компаративна ана–лиза показује да алгоритам инспирисан понашањем нилских коња остварује бржу конвергенцију и опти–малне вредности функције циља у односу на остале анализиране приступе. Предложена методологија о–могућава робустан приступ ретерминирању у усло–вима вишеструких ограничења производних ресурса.