Job scheduling problem using NSGA-II with six different policies

Juarez Leyva Jorge Antonio
García Carrizoza André
Cuevas León Carlos Ricardo
jantoniojl07@gmail.com
andycarritos@gmail.com
carlos.cuevaleon@gmail.com
Instituto Politécnico Nacional - Escuela Superior de Cómputo
Gustavo A. Madero, Mexico City, Mexico

Miriam Pescador Rojas
mpescadorr@ipn.mx
Instituto Politécnico Nacional - Centro de Investigación y de Estudios Avanzados
Gustavo A. Madero, Mexico City, Mexico

ABSTRACT

This work addresses a multi-objective job shop scheduling problem using a bioinspired genetic algorithm combined with problem-specific preprocessing strategies. The goal is to simultaneously minimize makespan and energy consumption across heterogeneous machines. Several deterministic preprocessing policies are proposed to construct fixed operation ordering arrays and auxiliary precedence indices, allowing all schedules to be evaluated efficiently using a single aptitude function. These policies include FIFO, LTP, STP, and their Round Robin variants, enabling structured exploration of different scheduling behaviors.

The genetic algorithm follows an NSGA-II-inspired framework based on Pareto dominance and crowding distance to ensure convergence and solution diversity. Uniform crossover, multiple mutation operators, and an adaptive parameter adjustment mechanism guided by hypervolume feedback are employed. The approach was evaluated in three scenarios of increasing complexity using 30 independent executions. Results show that Round Robin-based policies consistently achieve better trade-offs between time and energy, especially in larger instances, while preprocessing significantly improves efficiency and interpretability.

KEYWORDS

NSGA-II, Multiploide, Task Assignment,

ACM Reference Format:
Juarez Leyva Jorge Antonio, García Carrizoza André, Cuevas León Carlos Ricardo, and Miriam Pescador Rojas. 2026. Job scheduling problem using NSGA-II with six different policies . In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The orchestration of complex operations in modern Command and Control (C2) systems and autonomous clusters, such as Unmanned Surface Vehicles (USVs), presents a formidable challenge in the field of combinatorial optimization [4]. The Task Assignment Problem (TAP) in these domains is inherently dynamic, characterized by high levels of uncertainty and the frequent emergence of unexpected events, including platform failures, environmental shifts, and sudden task arrivals [3]. Unlike traditional static scheduling, which assumes a deterministic and stable environment, operational contexts in the battlefield or maritime search missions require adaptive mechanisms capable of real-time reconfiguration while maintaining organizational stability.

This study addresses the TAP through a multi-objective lens, seeking to balance two fundamentally conflicting goals: the minimization of the makespan (C_{max}), representing the total mission duration, and the optimization of total energy consumption (E_{total}). The trade-off between these objectives is critical; while aggressive resource allocation may reduce completion time, it often leads to prohibitive energy costs and organizational destabilization. This interplay necessitates the generation of a Pareto-optimal set of solutions, providing decision-makers with a range of feasible strategies rather than a single, potentially fragile, optimal point [2].

To solve this hybrid integer nonlinear programming problem under strong constraints, we propose an improved version of the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [2]. Our approach extends the classical framework by incorporating three primary technical innovations:

- **Multi-Policy Encoding:** We utilize a multi-ploid representation where individuals are encoded with various dispatching policies (e.g., RR-FIFO, LTP, STP), allowing the evolutionary process to explore diverse scheduling heuristics simultaneously, building upon classical dispatching theory [1].
- **Adaptive Parameter Control:** Crossover (p_c) and mutation (p_m) probabilities are dynamically adjusted based on hypervolume improvement [5], ensuring an optimal balance between exploration and exploitation as the search progresses.
- **Efficient Constraint Handling:** A specialized preprocessing stage utilizing auxiliary arrays (*orden* and *last_idx*) guarantees that all precedence and machine availability constraints are satisfied *a priori*, reducing the fitness evaluation complexity to $O(N)$ and eliminating the need for costly repair mechanisms.

The performance of the proposed adaptive NSGA-II is rigorously evaluated across multiple scenarios of increasing complexity. Our

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

results demonstrate that as system scale grows, the choice of scheduling policy becomes a dominant factor in solution quality, with round-robin variants consistently providing superior trade-offs in mission-critical environments. This work contributes both a theoretical framework and empirical evidence for the use of adaptive evolutionary algorithms in the resilient management of autonomous and organizational systems.

2 RELATED WORKS

2.1 A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II

The foundation of modern multi-objective optimization is largely defined by the NSGA-II framework [2]. This study focuses on improving the adaptability of Command and Control (C2) organizations operating in complex and uncertain environments. Unlike static scheduling approaches, the authors recognize that real operational contexts are affected by unexpected events such as platform failures or sudden task additions. This dynamic perspective motivates the need for adaptive optimization methods capable of responding to evolving battlefield conditions.

A key contribution of the work by [2] is the efficient non-dominated sorting and crowding distance mechanisms, which allow for a well-distributed Pareto front. By explicitly modeling conflicting objectives, the approach avoids overly aggressive reallocations that may destabilize the organization while still ensuring mission effectiveness.

2.2 Multi-USV Task Assignment Based on NSGA II-MC

The coordination and task allocation of multiple Unmanned Surface Vehicles (USVs) has gained increasing importance with the advancement of autonomous systems [3]. The dispersion of underwater targets introduces uncertainty, making it necessary to balance detection effectiveness with operational efficiency.

The work of [3] formulates the USV task allocation as a bi-objective optimization model that seeks to maximize target detection probability while minimizing collaborative search time. To handle uncertainty, adaptive genetic operators are incorporated to balance exploration and exploitation. Furthermore, performance metrics such as those proposed by [5] are often used in these contexts to evaluate the quality of the generated Pareto sets, specifically through the hypervolume indicator.

2.3 Dynamic Task Assignment in Command and Control

In the field of Command and Control (C2), dynamic task assignment is characterized as an event-driven process [4]. Real operational contexts are continuously affected by unexpected events such as platform damage or sudden task emergence. By framing the problem as discrete at the micro level, the study establishes a practical foundation for modeling real-time organizational adaptation under uncertainty.

The model produced in [4] provides decision-makers with multiple feasible adjustment strategies through a Pareto front. This focus aligns the optimization framework with the Resource Constrained Project Scheduling Problem (RCPSP), reinforcing the practical relevance of the proposed approach for real-world military decision-making.

2.4 Dispatching Rules and Scheduling Policies

A critical component of solving the Task Assignment Problem effectively is the selection of appropriate dispatching rules. The classic survey by [1] categorizes various heuristics such as First-In-First-Out (FIFO) and Shortest Processing Time (SPT), which have been widely used in manufacturing.

Our work builds upon these foundations by integrating these traditional rules into a **multi-policy encoding** strategy. Unlike the previously discussed works that often rely on a single priority rule, our proposed adaptive NSGA-II explores multiple policies simultaneously (including Round-Robin variants), allowing the evolutionary process to identify the most resilient scheduling strategy for a given operational complexity.

3 PROPOSED SOLUTION

Mathematical Formulation

Let the set of operations be indexed by $1, \ldots, N$ and the set of machines by $1, \ldots, m$. We define:

- O_i: identifier of the operation at position i in the ordering array (`orden`).
- C_i: machine assigned to operation O_i (`chromosome`).
- L_i: index of the predecessor operation that must be completed before O_i can start (`last_idx`).
- $T_{o,r}$: processing time of operation o on machine r.
- $E_{o,r}$: energy consumed by operation o on machine r.

For each position $i = 1, \ldots, N$, with operation $o = O_i$ and assigned machine $r = C_i$, the processing duration is defined as:

$$d_i = T_{o,r}$$

Let f_{L_i} denote the completion time of the predecessor operation L_i (with $f_0 = 0$ if $L_i = 0$). The start time of operation i is given by:

$$s_i = \max(f_{L_i}, M_r)$$

where M_r represents the time at which machine r becomes available. The completion time is then:

$$f_i = s_i + d_i$$

and the machine availability is updated as:

$$M_r \leftarrow f_i$$

The total energy consumed is computed as:

$$E_{\text{total}} = \sum_{i=1}^{N} E_{O_i,C_i}$$

and the *makespan* of the individual is obtained as:

$$C_{\max} = \max_{r \in \{1,\ldots,m\}} M_r$$

Therefore, the fitness function returns the objective vector:

$$\text{Fitness}(O, C) = (C_{\max}, E_{\text{total}})$$

where $C_{\max}$ represents the total completion time (makespan) and E_{total} the total energy consumed by the sequence represented by the individual.

Policies and Construction of the Ordering Array

The differences among individuals arise mainly from the method used to construct the `orden` array, which depends on the scheduling policy employed:

- **FIFO (First-In First-Out):** operations are processed in the order in which they arrive, without any reordering.
- **LTP (Longest Time Processing):** operations are sorted in descending order according to their average processing time:

$$\bar{T}_o = \frac{1}{m} \sum_{r=1}^{m} T_{o,r}$$

- **STP (Shortest Time Processing):** identical to LTP, but sorted in ascending order (shorter operations first).
- **RR-FIFO (Round Robin FIFO):** operations are grouped by job and interleaved column by column, such that each job contributes one operation per turn while respecting its internal order.
- **RR-LTP:** jobs are ordered according to their total average processing time (as in LTP) and interleaved following the Round Robin policy.
- **RR-ECA:** similar to the previous case, but the initial ordering is performed according to the average energy consumption per job.

Each policy produces a distinct ordering vector, representing different scheduling strategies and priority criteria among jobs.

3.1 Preprocessing and aptitude

After defining the different scheduling policies and the construction of the ordering array, a preprocessing stage is introduced in order to simplify the evaluation of individuals during the optimization process. This preprocessing step transforms the original constrained scheduling problem into a representation that allows the fitness function to be evaluated efficiently, while preserving all precedence and machine availability constraints. In particular, two auxiliary arrays are constructed: the operation order array (`orden`) and the precedence index array (`last_idx`).

The `orden` array defines a fixed global sequence in which operations are considered for scheduling, according to a selected policy such as FIFO, LTP, STP, or Round Robin–based strategies. In parallel, the `last_idx` array encodes the precedence constraints between operations by storing, for each operation, the index of its immediate predecessor in the sequence. A value of zero indicates that the operation has no predecessor and can be scheduled as soon as a machine is available. This representation eliminates the need for dynamic constraint checking or graph traversal during fitness evaluation.

Thanks to this preprocessing, the aptitude (fitness) evaluation of an individual can be performed using a single forward pass over the operation sequence. For each operation, the starting time is computed as the maximum between the completion time of its predecessor and the availability time of the assigned machine. This mechanism ensures that both technological precedence constraints and machine constraints are always satisfied, without requiring any repair mechanisms or infeasible solution handling.

The aptitude function simultaneously computes two objective values: the makespan, defined as the maximum completion time across all machines, and the total energy consumption, computed as the sum of the energy costs associated with each operation–machine assignment. Additionally, a detailed schedule is generated during the evaluation process, which is later used for visualization and analysis of the obtained solutions. The use of preprocessed arrays guarantees that this schedule is always feasible.

Algorithm 1: Multi-objective Genetic Algorithm with Adaptive Parameters

```
Input: Population size P, number of generations G,
       crossover probability p_c, mutation probabilities p_m
Output: Final population with Pareto-optimal solutions
Initialize population with random individuals;
Evaluate fitness (makespan, energy) for all individuals;
for g ← 1 to G do
    Assign Pareto fronts and crowding distances;
    Initialize empty offspring population;
    while offspring size < P do
        Select two parents randomly;
        if rand < p_c then
            Apply uniform crossover to generate two
             offspring;
        else
            Copy parents directly;
        Apply inter-chromosome mutation;
        Apply exchange mutation;
        Apply displacement mutation;
        Evaluate offspring fitness;
        Add offspring to population;
    Merge parent and offspring populations;
    Select next generation using tournament selection;
    if g mod k = 0 then
        Adapt crossover and mutation probabilities based
         on hypervolume improvement;
return Final population;
```

Computational Complexity

The fitness evaluation has a time complexity of $O(N)$ per individual, since each operation requires only constant-time accesses and updates to the vectors M, f, and L. This allows an efficient evaluation of a large number of individuals at each generation of the genetic algorithm, while preserving feasibility with respect to precedence constraints and machine availability.

3.2 Genetic Algorithm

The proposed solution is based on a multi-objective genetic algorithm inspired by the NSGA-II framework, specifically adapted to address a flexible job scheduling problem with multiple scheduling policies. Each individual in the population represents a complete

Algorithm 2: Preprocessing for Operation Ordering and Precedence Constraints

Input: Set of jobs and operations
Output: Ordering array `orden`, precedence array `last_idx`
Initialize empty arrays `orden`, `last_idx`;
Set current index to 1;
foreach *job* **do**
  foreach *operation in job* **do**
    Append operation to `orden`;
    if *operation is first in job* **then**
      Append 0 to `last_idx`;
    else
      Append index of previous operation to `last_idx`;
    Increment current index;
return `orden`, `last_idx`;

scheduling solution composed of multiple chromosomes, where each chromosome corresponds to a different dispatching policy. This representation allows the algorithm to explore multiple scheduling strategies simultaneously while maintaining a unified evolutionary process.

An individual is encoded as a set of chromosomes, where each chromosome assigns machines to a fixed sequence of operations. The sequence itself is determined externally by predefined policies such as FIFO, LTP, STP, and Round Robin variants. This separation between operation ordering and machine assignment reduces the complexity of the search space while preserving feasibility with respect to precedence constraints. The diversity among individuals is mainly driven by variations in machine assignments and the interaction between different policies.

The fitness evaluation of each chromosome is performed by simulating the execution of operations while respecting machine availability and precedence constraints. Two objective functions are considered: the makespan, defined as the total completion time of all operations, and the total energy consumption. These objectives are conflicting in nature and are jointly optimized using Pareto dominance. The resulting fitness values allow the population to be ranked into non-dominated fronts and assigned crowding distances following the NSGA-II methodology.

Genetic variation is introduced through uniform crossover and multiple mutation operators. The crossover operator exchanges machine assignments between two parent individuals at the gene level, promoting the combination of different scheduling patterns. Mutation operators include inter-chromosome swaps, intra-chromosome exchanges, and displacement mutations, which collectively enhance population diversity and prevent premature convergence. These operators enable both local refinements and global exploration of the solution space.

Selection and replacement are performed using a tournament-based strategy that compares individuals according to their Pareto front rank and crowding distance. For each chromosome position, the better candidate is selected from competing individuals, ensuring that elite solutions are preserved while maintaining diversity.

Algorithm 3: Multiobjective Genetic Algorithm with Adaptive Parameters

Input: Population size N, number of generations G, crossover probability p_c, mutation probabilities p_m, objective functions
Output: Final population of non-dominated solutions
Initialize population P with N random individuals;
Evaluate all individuals in P;
for $g \leftarrow 1$ **to** G **do**
  Assign Pareto fronts and crowding distances to P;
  $O \leftarrow \emptyset$;
  while $|O| < N$ **do**
    Select two parents A, B randomly from P;
    if *rand()* $< p_c$ **then**
      Apply uniform crossover to A and B to generate offspring
    else
      Copy parents as offspring
    Apply mutation operators to offspring;
    Evaluate offspring;
    Add offspring to O;
  $P \leftarrow P \cup O$;
  Assign Pareto fronts and crowding distances to P;
  Select next generation using tournament selection;
  Compute hypervolume indicator;
  if *generation* $\bmod k = 0$ **then**
    Adapt crossover and mutation probabilities based on hypervolume change;
return P

Additionally, adaptive adjustment of crossover and mutation probabilities is incorporated during the evolutionary process to balance exploration and exploitation, allowing the algorithm to dynamically respond to the observed search performance.

4 TESTS AND RESULTS

The proposed NSGA-II-based scheduling approach was evaluated under three different problem scenarios in order to assess its robustness, scalability, and sensitivity to problem complexity. For each scenario, the algorithm was executed using ten different random seeds, resulting in a total of thirty independent executions. The scenarios differ significantly in size: the first scenario considers 5 operations, 4 machines, and 6 jobs; the second scenario increases the complexity to 8 operations, 10 machines, and 10 jobs; while the third scenario represents the most challenging case with 15 operations, 5 machines, and 12 jobs. This experimental design allows a systematic analysis of convergence behavior, policy dominance, and schedule structure across increasing levels of difficulty.

Regarding convergence behavior, clear differences are observed among the three scenarios. In the first scenario, the Pareto fronts converge rapidly, stabilizing around generation 100 with no significant generation of new non-dominated solutions afterward. This behavior suggests that the problem is relatively simple and that the search space is quickly exhausted. In contrast, the second and third

scenarios show sustained improvement over a longer horizon, with noticeable Pareto front expansion until approximately generation 150. These results indicate that increased problem complexity delays convergence and benefits more from extended evolutionary search.

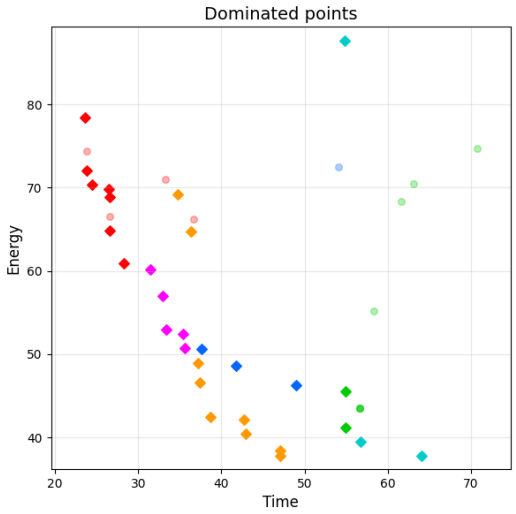

Figure 1: Pareto front for first scenario on last iteration

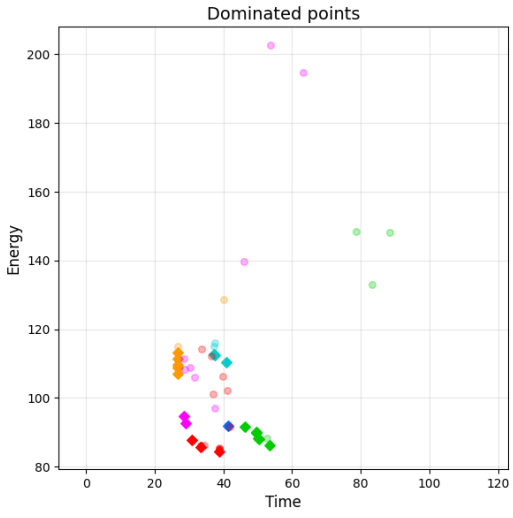

Figure 2: Pareto front for second scenario on last iteration

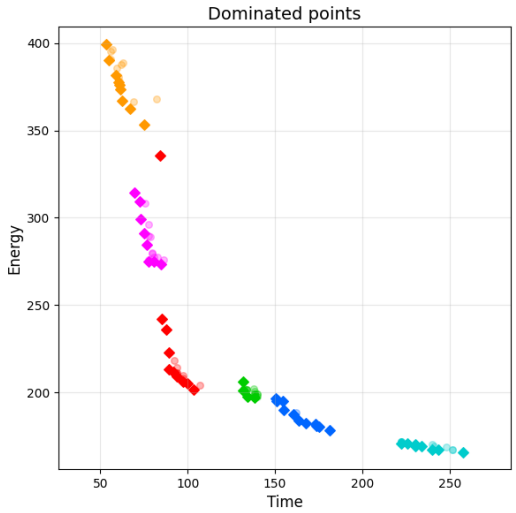

Figure 3: Pareto front for third scenario on last iteration

An analysis of the Pareto front distributions reveals meaningful structural patterns. In the first scenario, Pareto fronts associated with each policy are relatively well distributed and exhibit limited overlap. Round-robin-based policies (RR-FIFO, RR-LTP, and RR-ECA) clearly outperform their non-round-robin counterparts. In particular, RR-LTP tends to achieve superior makespan minimization, while RR-FIFO and RR-ECA alternate between better energy minimization and balanced trade-offs. Non-round-robin policies such as FIFO, LTP, and STP consistently populate dominated regions of the objective space, highlighting the importance of rotation-based task assignment even in simpler scenarios.

The second scenario presents a notably different Pareto structure. Here, the Pareto fronts become more compact and cluster around a smaller region of the objective space, forming an almost circular distribution. In this case, RR-FIFO exhibits strong performance in makespan minimization, while STP becomes more competitive in energy minimization. LTP emerges as the weakest policy, with its Pareto front consistently dominated by those of other strategies. This behavior suggests that as system complexity increases, naive prioritization of long tasks leads to inefficient resource utilization and poor trade-offs.

The third scenario provides the clearest separation between policies and offers the most insightful results. Unlike the previous scenarios, the Pareto fronts corresponding to each policy are almost entirely disjoint. RR-FIFO and RR-ECA dominate makespan minimization, achieving solutions as low as 130 seconds with energy consumption around 188 units. RR-LTP occupies a central region, acting as a compromise between time and energy. Meanwhile, FIFO and LTP remain near the energy-minimization region but at the cost of significantly increased makespan, while STP produces the lowest-energy solutions overall. This separation strongly indicates that policy choice becomes increasingly critical as problem complexity grows.

Seed variability analysis reveals that although the exact position of each Pareto front varies across runs, the relative ranking

of policies remains consistent. While some seeds generate wider or more dispersed fronts—particularly in the second scenario—the global structure of dominance is preserved. The spread between extreme solutions is relatively large, reflecting the stochastic nature of evolutionary algorithms. However, extreme solutions such as minimum-time and minimum-energy schedules are frequently rediscovered across different seeds, indicating acceptable robustness.

The analysis of 180 generated schedules provides further insight into the structural behavior of the solutions. As evolution progresses, schedules exhibit a clear reduction in idle times and a more compact arrangement of operations. Early-generation schedules contain fragmented execution patterns and frequent machine inactivity, while late-generation schedules show tighter operation blocks and improved resource utilization. This structural evolution is especially evident in scenarios two and three, where increased complexity amplifies inefficiencies in suboptimal schedules.

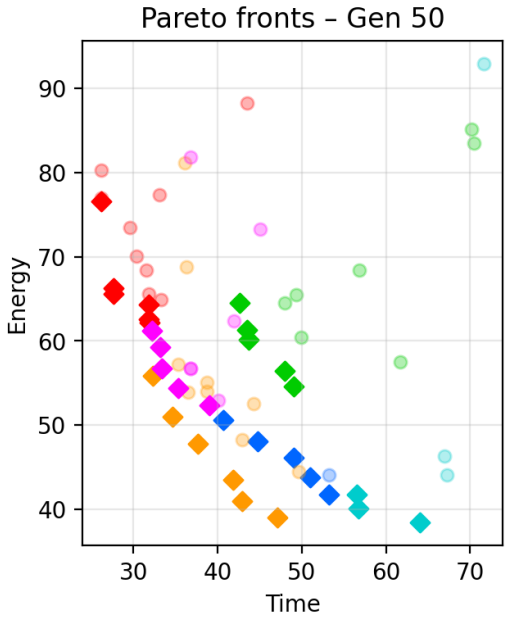

Figure 4: Dominated and non dominated point at 50 generations of first scenario

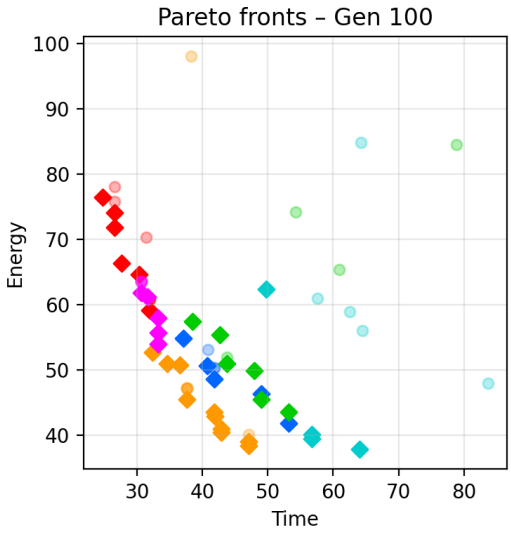

Figure 5: Dominated and non dominated point at 100 generations of first scenario

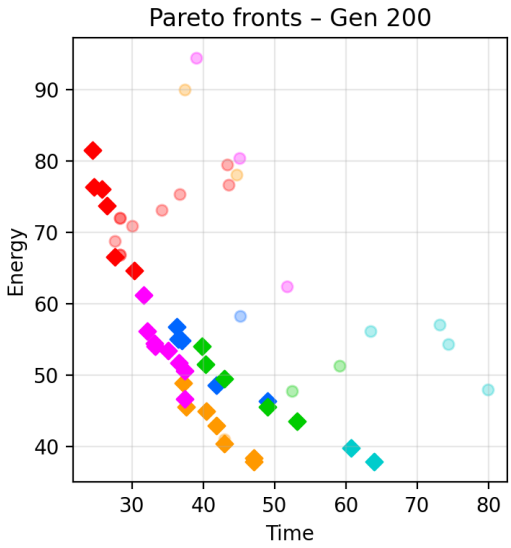

Figure 6: Dominated and non dominated point at 200 generations of first scenario

Policy-specific scheduling behavior is also evident from the Gantt-like visualizations. FIFO, LTP, and STP frequently leave one or more machines underutilized or completely unused, especially when optimizing for energy. In contrast, round-robin-based policies distribute tasks more evenly across machines, particularly when minimizing makespan. RR-ECA and RR-FIFO consistently activate all available machines in time-optimal solutions, confirming their effectiveness in exploiting parallelism.

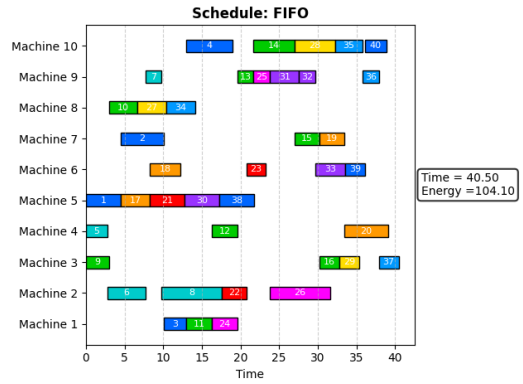

Figure 7: Schedule solution for FIFO on the second scenario

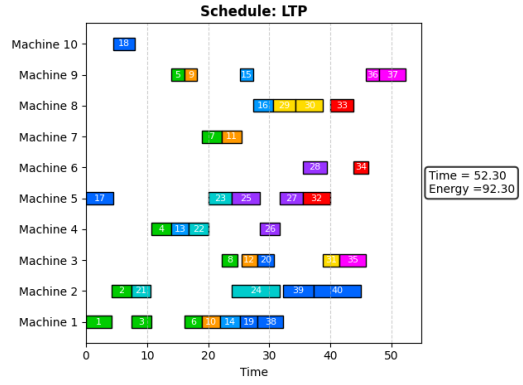

Figure 8: Schedule solution for LTP on the second scenario

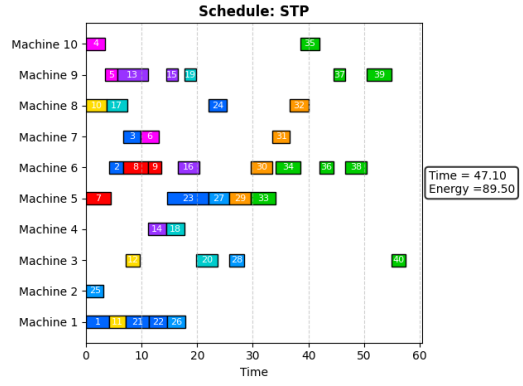

Figure 9: Schedule solution for STP on the second scenario

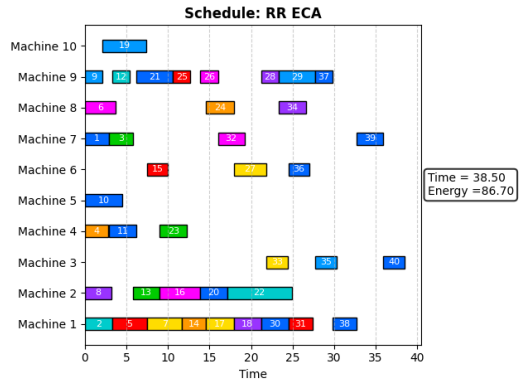

Figure 10: Schedule solution for RR ECA on the second scenario

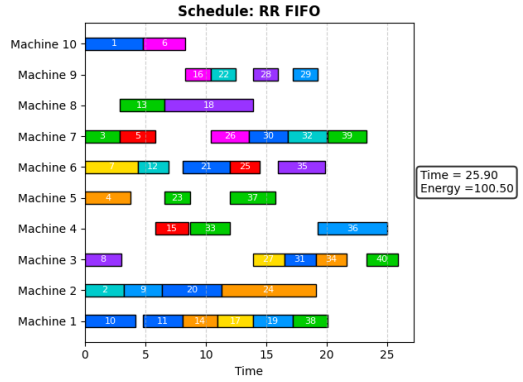

Figure 11: Schedule solution for RR FIFO on the second scenario

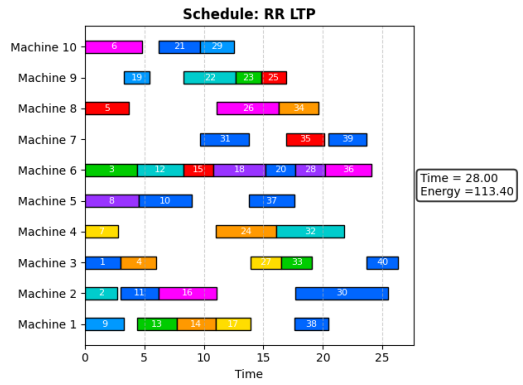

Figure 12: Schedule solution for RR LTP on the second scenario

Energy-time trade-offs are visually distinguishable in the schedules. Energy-efficient solutions often feature increased idle times

and selective machine usage, whereas time-optimal schedules are densely packed and rely on aggressive machine utilization. Intermediate Pareto solutions, especially those associated with RR-LTP, display balanced structures that avoid excessive idle time while limiting unnecessary energy consumption. These visual patterns align well with the quantitative Pareto front results.

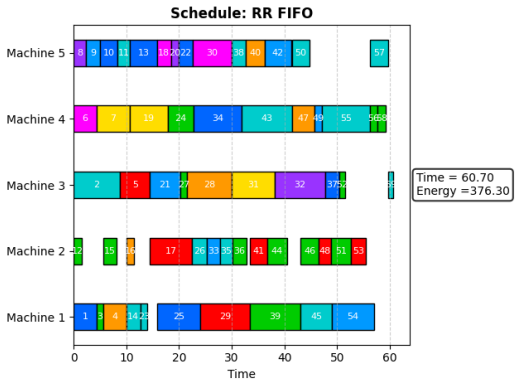

Figure 13: Schedule with minimum time on third scenario

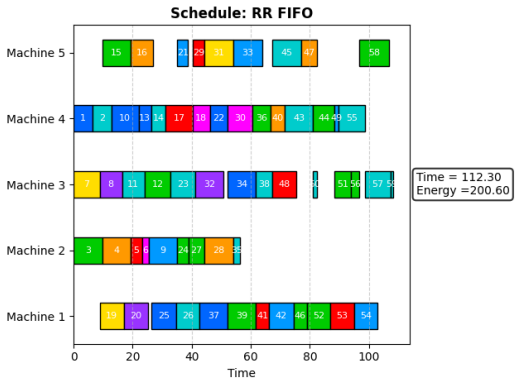

Figure 14: Schedule with middle values of time and energy on third scenario

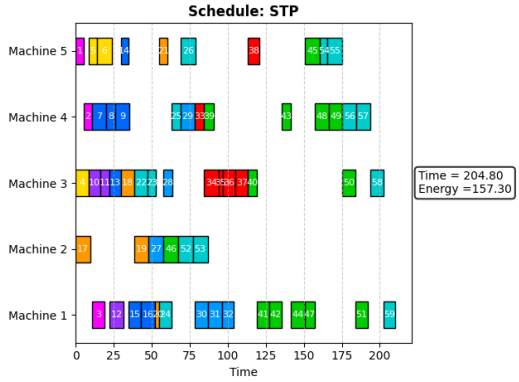

Figure 15: Schedule with minimum energy on third scenario

The impact of adaptive crossover and mutation mechanisms appears subtle. While minor improvements in convergence speed and occasional escape from local optima are observed, the overall Pareto front structure remains similar to that obtained with fixed parameters. This suggests that while parameter adaptation may enhance search dynamics, policy selection and problem structure exert a stronger influence on final solution quality.

5 CONCLUSIONS

This work presented a multi-objective solution approach for the flexible job shop scheduling problem, integrating NSGA-II with multiple construction policies and a preprocessing-based fitness evaluation. By decoupling operation ordering from machine assignment through structured preprocessing, the proposed framework achieves efficient evaluation while preserving solution diversity. The experimental design, involving three scenarios of increasing complexity and multiple random seeds, allowed for a comprehensive assessment of convergence behavior, robustness, and policy effectiveness.

The results demonstrate that round-robin-based policies consistently outperform traditional FIFO, LTP, and STP strategies, particularly as problem complexity increases. In simpler scenarios, all policies converge rapidly, although round-robin variants still yield superior Pareto fronts. As the problem size grows, the dominance of round-robin policies becomes more pronounced, producing better trade-offs between makespan and energy consumption, as well as clearer separation in the Pareto space. These findings confirm that balanced task rotation and improved machine utilization are critical factors in multi-objective scheduling performance.

Visual analysis of the generated schedules further supports the quantitative results. Time-optimal solutions exhibit dense operation packing and extensive machine utilization, whereas energy-efficient schedules rely on selective machine activation and increased idle periods. Intermediate Pareto solutions provide balanced structures that mitigate excessive energy use without significantly increasing makespan. The evolution of schedules across generations highlights the algorithm's ability to progressively reduce idle times and improve operational compactness, reinforcing the effectiveness of the evolutionary search process.

As future work, several extensions can be explored to further enhance the proposed approach. First, the preprocessing policies could be made adaptive or learned online, allowing the ordering strategy to evolve jointly with the genetic algorithm instead of being fixed beforehand. Second, additional objectives such as machine workload balance, tardiness, or maintenance costs could be incorporated to better reflect real industrial environments. From an algorithmic perspective, hybridizing the proposed method with local search or decomposition-based multi-objective techniques may improve convergence speed and solution diversity

REFERENCES

[1] John H Blackstone, Don T Phillips, and Gary L Hogg. 1982. A survey of dispatching rules for manufacturing job shop operations. *International Journal of Production Research* 20, 1 (1982), 27–45.

[2] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2 (2002), 182–197.

[3] Yu Liu et al. 2021. An exact algorithm for task allocation of multiple unmanned surface vehicles with minimum task time. *Journal of Marine Science and Engineering* 9, 8 (2021), 807.

[4] Shao-fei Wang et al. 2016. The solution of target assignment problem in command and control decision-making behaviour simulation. In *Proceedings of the 2016 International Conference on Robots and Intelligent System (ICRIS)*. IEEE.

[5] Eckart Zitzler and Lothar Thiele. 1999. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation* 3, 4, 257–271.

Received 13 January 2026