Proceedings of the 8th European Conference on Industrial Engineering and Operations Management
Paris, France, July 2-4, 2025

Proceedings of the International Conference on Industrial Engineering and Operations Management

Publisher: IEOM Society International, USA DOI: 10.46254/EU08.20250278
Published: July 2, 2025

Improving Welding Job Scheduling in Shipbuilding with Optimization Algorithms

Ángel Sánchez-Fernández and Elena Denisa-Vlad
Researcher
CEMI Navantia-UDC, Universidade da Coruña
Batallones s/n, 15403 Ferrol, Spain
angel.sanchez1@udc.es, e.denisa.vlad@udc.es

Ana Ortega-Sarceda
Researcher
Universidade da Coruña, Campus Industrial de Ferrol, CITENI
Campus de Esteiro s/n, 15403 Ferrol, Spain
ana.ortega@udc.es

Javier Pernas-Álvarez and Diego Crespo-Pereira
Professor, Business Department
Universidade da Coruña, Campus Industrial de Ferrol, CITENI, GII
Campus de Esteiro s/n, 15403 Ferrol, Spain
javier.pernas2@udc.es, diego.crespo@udc.es

Adolfo Lamas-Rodríguez, Belén Sañudo-Costoya and Santiago Tutor-Roca
Navantia S.A, S.M.E,
Taxonera. 11, 15403 Ferrol, Spain
alamasr@navantia.es, bsanudo@navantia.es, sjtutor@navantia.es

Abstract

This research addresses an optimization problem by scheduling welding jobs for the preassembly elements in robotic manufacturing cells within the shipbuilding industry. The key operational pillars include the required delivery time for the preassembly elements and their welding quality. To address these, constrained programming (CP) and several metaheuristic algorithms were proposed. To evaluate their performance in the case study, the IBM ILOG CPLEX module has been utilized as a Constrained Programming Optimizer (CPO) for modeling and solving the proposed job scheduling problem, ensuring constraint satisfaction and efficient solution exploration. Additionally, these algorithms have been integrated into the discrete event simulation (DES) model developed in Siemens Tecnomatix Plant Simulation using Python. The results obtained by metaheuristics were compared by varying the operational parameters of these algorithms with the GA Wizard algorithm integrated into the DES itself, observing that these external algorithms significantly outperform the GA Wizard. Conversely, the results show that for the proposed problem, CPO provides a significantly superior solution, achieving minimal delays. However, this technique has certain limitations that highlight the value of implementing metaheuristic algorithms for process optimization in the shipbuilding industry, particularly for more complex problems. Thus, a robust optimization methodology is proposed for job shop scheduling in ship manufacturing by integrating academic aspects, such as computational optimization methods, with the DES software commonly used in industry.

Keywords
Shipbuilding, Welding Job Scheduling, Constraint Programing, Metaheuristics, Optimization

1. Introduction
The current competitiveness in the shipbuilding manufacturing industry necessitates enhancing the efficiency of companies' production processes, positioning shipyards as a critical point for managing assembly processes. According to Gourdon and Steidl (2019), 70%-80% of the added value in ship production comes from intermediate products, which are largely processed within manufacturing cells located in workshops. Therefore, this study addresses the specific problem of welding-based assembly of preassembly elements to establish a solid optimization framework that can be extrapolated to other similar problems in the industry.

The problem is centered on scheduling welding jobs for preassembly elements, which consist of bases and reinforcements with varying geometries and processing times. These bases and reinforcements are grouped in the input buffers of the robotic cell, where two robots perform welding tasks. The first robot acts as a manipulator arm, positioning each element onto the assembly table, while the second robot executes the welding seams between the reinforcements and the base, as well as carrying out the assigned inspections of the preassembly elements. The key to optimizing this problem lies in meeting the predefined completion time for these tasks while maximizing the number of preassemblies undergoing inspection. This approach tackles job scheduling not only from a temporal perspective but also by considering the quality of these intermediate products.

To address this challenge, two alternative methodologies are presented. On one hand, to optimize the process using the proposed metaheuristic algorithms, the entire system is modeled in Plant Simulation, incorporating all operational specifications. Simultaneously, these external algorithms are coded to ensure seamless integration with the DES model, enabling performance evaluation and yielding a significantly improved solution. On the other hand, solving the problem through CP requires adapting the model to the syntax of the selected optimizer, which in this case is IBM ILOG CPLEX. This approach relies on interval decision variables, constraint programming, and the definition of the objective function to obtain an optimal scheduling solution.

The paper is structured as follows: Section 2 presents a literature review, outlining the current state of optimization techniques within the application domain. Section 3 introduces the methods implemented in this research for solving the problem. Section 4 details the software tools and data sources used in the study. Section 5 provides the computational results obtained through the proposed methods. Finally, Section 6 presents the conclusions drawn from this research.

1.1 Objectives
This research originates from a demand by the Spanish state-owned shipbuilding enterprise Navantia to optimize the job sequencing and visual inspection process in a robotic cell for preassembly welding. The proposed methodology must align with Navantia's planning and monitoring procedures or be easily integrated into its existing information systems. To achieve this, the study focuses on three key objectives:

- Identify state-of-the-art metaheuristics for job sequencing to maximize the visual inspection of preassemblies.
- Integrate the metaheuristics with models developed in Plant Simulation and provide an easy-to-use framework for non-expert decision-makers in Navantia.
- Evaluate the performance of metaheuristics against exact methods such as Constraint Programming.

2. Literature Review
The scheduling problem in manufacturing cells has been extensively studied in recent years by various researchers. A no-wait flowshop scheduling problem was analyzed by Ying et al. (2011), who employed three metaheuristic algorithms: Genetic Algorithm (GA), Simulated Annealing (SA), and Iterated Greedy (IG). These algorithms were selected due to the complexity of the problem, and the study demonstrated their effectiveness in addressing such scheduling challenges. These algorithms were also compared by Forghani and Ghomi (2020), who integrated three cellular manufacturing problems, with particular emphasis on cell scheduling, which is relevant to the focus of this study. They proposed a specific encoding scheme to represent the candidate solutions and found that SA outperformed other heuristic and metaheuristic approaches. In line with Industry 4.0 trends, Nejad et al. (2019) also investigated the scheduling problem of a real-life flexible robotic cell. They employed the Taguchi method for the initial combination

of algorithm parameters, demonstrating that the hybrid genetic algorithm outperformed both the traditional GA and SA.

However, while these studies focused on large-scale problems, Sahin and Alpay (2024) also explored an integrated cell formation and job scheduling problem, demonstrating that algorithms such as the GA and Marine Predators Algorithm (MPA) are highly effective, providing reasonable and applicable solutions for both large and medium-sized problems. They emphasized that for smaller problems, linear mathematical models can easily offer solutions. Lu et al. (2022) also considered different numbers of jobs and machines, considering the number of instances and categorizing them into small, medium, and large scales. A Variable-Neighborhood Search (VNS) was developed and compared with Jaya, GA, and Atomic Orbital Search (AOS), measuring computation time, machine finish probability range, and maximum probability value. The results demonstrated that VNS outperformed the other algorithms and is suitable for applications in high-complexity manufacturing industries, such as chip and aircraft production, which involve significant manufacturing complexities. Additionally, in the shipbuilding industry, Liao et al. (2022) studied the ship maintenance service scheduling problem, considering deteriorating maintenance time, distributed maintenance tasks, and limited maintenance teams. They compared the VNS with two heuristic methods, demonstrating that VNS offers advantages in terms of convergence speed and accuracy in this application domain.

Ibrahem et al. (2014) employed a novel iterative computational evolution model, Particle Swarm Optimization (PSO), and compared it with the GA to solve a cellular flowshop scheduling problem with family sequence setup time, aiming to minimize the total flow time. The performance of the proposed algorithms was evaluated using Design of Experiments (DoE), which demonstrated that the PSO-based metaheuristic outperformed GA in solving this type of problem, although GA was also capable of yielding high-quality solutions.

As demonstrated, a significant number of algorithms have been proposed to address various cellular flowshop scheduling problems. In this context, we aim to focus on the utility of population-based algorithms, as these have been shown to generally be more effective, as evidenced by Schlenkrich and Parragh (2022). Three main types of algorithms can be distinguished: the Genetic Algorithm (GA) as developed by Pezzella et al. (2007), the Particle Swarm Optimization (PSO) proposed by Eberhart and Kennedy (1995), and the Differential Evolution (DE) algorithm as proposed by Wu and Liu (2019).

Regarding the genetic algorithm, it is one of the most common strategies for job scheduling optimization. Luo et al. (2012) addressed a hybrid flowshop scheduling problem where jobs are grouped into families based on machine configurations, ultimately demonstrating that the GA is highly effective in reducing makespan. This makespan minimization is further corroborated by the research of Driss et al. (2015), in which a new genetic algorithm was proposed to minimize makespan in a flexible job scheduling problem (FJSP), with instances exceeding 10 jobs and 13 machines. For the efficient resolution of such problems, Zhang et al. (2012) propose a GA procedure with tabu search, considering transportation constraints. These types of constraints were also studied by Tao et al. (2019), who applied tabu search in transporter scheduling for assembly blocks in a shipyard. The PSO method is widely used for optimization in many fields. Rose and Coenen (2015) compared it with three other metaheuristic algorithms for solving a constraint satisfaction problem in ship outfitting scheduling, concluding that the solution quality was quite poor compared to the other algorithms. However, as observed, Ibrahem et al. (2014) solved a cellular flowshop scheduling problem, demonstrating that this algorithm can offer good performance in such a problem. Finally, regarding DE, it has been compared by Lunardi et al. (2021) with other methods for a real-world scheduling problem, also proposing a DE with tabu search and observing the great effectiveness of the solution it provides. Xu et al. (2013) introduce a hybrid discrete differential evolution algorithm that integrates DE with local search to address the lot splitting problem in FJSP. Similarly, Yuang and Xu (2013) also combine DE and local search to solve the FJSP.

Alternatively, exact optimization methods such as CP have been widely adopted as a complement to mixed-integer linear programming (MILP) for medium- and large-scale problems. This shift stems from the inability of MILP to efficiently solve problems of such complexity within acceptable computational times, as demonstrated in the studies conducted by Verbiest et al. (2019) and Dauzére-Pérès et al. (2024). Although it was not initially considered an optimization technique in literature reviews such as those conducted by Chaudhry and Khan (2015) and Xie et al. (2019), it is now widely applied as such. In fact, scheduling is one of the most successful applications of CP as demonstrated by Laborie (2018). Similarly, Da Col and Teppan (2022) anticipate a growing adoption of CP in the industrial domain.

Due to the computational nature of CP, its performance heavily depends on the optimization solver employed. In this study, the Constrained-Programming Optimizer (CPO) from IBM, as introduced by Heinz and Beck (2012) and Laborie et al. (2018), is utilized. This solver follows a 'model-and-run' approach, implementing an automatic search-based optimization algorithm that dynamically combines various techniques, adapting their selection based on the problem size and optimization progress. Given the increasing application of CP in scheduling problems, several studies highlight its effectiveness. For instance, Meng et al. (2020) employed CPO in a Flexible Job Shop Scheduling (FJSP) problem for makespan minimization, demonstrating that it outperforms four Mixed-Integer Linear Programming (MILP) models by providing high-quality solutions for both small-sized and large-sized instances. Furthermore, the CP model surpasses other state-of-the-art algorithms, achieving the best-known solutions for 11 benchmark problems. The superiority of CP in scheduling problems has been consistently demonstrated in the literature. Laborie (2009b) showed that CP outperformed state-of-the-art ad hoc approaches across various scheduling problems. A decade later, Laborie (2018) reaffirmed these findings, demonstrating that a standalone CP model surpassed all previously proposed methods and successfully solved all 335 benchmark instances considered in the study.

In brief, the main highlights of this study are the development and application of metaheuristic algorithms, as studied in the literature, to solve scheduling problems in modern manufacturing cells used in industry. The study particularly emphasizes the performance of population-based algorithms in terms of solution quality and computation time, while also recognizing the relevance of exact methods, such as CP, in providing optimal solutions for these types of problems.

3. Methods

3.1 Case Study: Optimizing Job Sequencing in a Robotic Welding Cell

The case study focuses on a robotic welding cell responsible for assembling reinforcement profiles onto base structures, key components of modular shipbuilding blocks. This robotic workstation, located in a test environment at the Innovation and Robotics Center, is designed for full automation, handling material picking, welding operations, and optional visual inspection. The system comprises two robotic arms, a picking robot and a welding robot, each tasked with distinct operations to streamline the assembly process. Figure 1 shows the robotic cell in Navantia's facilities.

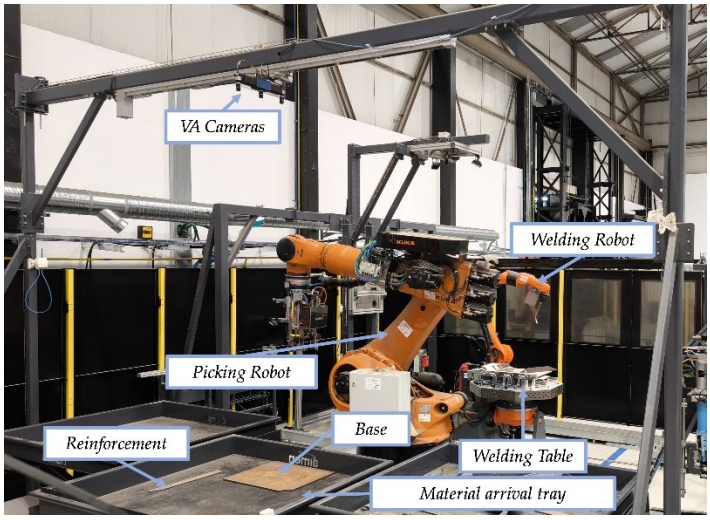

Figure 1. Real-world robotic welding cell with labeled components. Image courtesy of Navantia©

Due to the large variety of minor assemblies processed (over 1,000 different configurations, varying in size, topology, and reinforcement types) sequencing optimization presents significant challenges.

Upstream, the required steel bases and reinforcement profiles are cut in a randomized sequence dictated by the nesting process. As a result, an intermediate sorting step is required before welding. To improve efficiency, Navantia is evaluating the impact of pre-sorting materials before they enter the robotic welding cell, ensuring they arrive in a

structured sequence rather than in a purely random order. Additionally, due to time constraints, visual inspection cannot be performed on every assembly. This raises a critical research question:

How should job sequencing be optimized to maximize the number of preassemblies undergoing visual inspection while maintaining cycle time constraints?

3.2 Optimization Algorithms

The current research applies two different strategies to optimize job scheduling in a robotic manufacturing cell within the shipbuilding industry: metaheuristic approaches—Genetic Algorithm (GA), Particle Swarm Optimization (PSO), and Differential Evolution (DE)—along with an exact method, Constraint Programming (CP). All methods aim to minimize delays while ensuring constraint satisfaction.

Each metaheuristic algorithm is tailored to address the specific problem at hand:

- Genetic Algorithm (GA): GA operates through an evolutionary process, starting with a randomly generated population of solutions. It employs uniform crossover and partially mapped crossover (PMX) for offspring generation, while uniform mutation introduces diversity. Selection follows a tournament-based approach, and an elitist strategy ensures that the best solution is retained across generations.

- Differential Evolution (DE): DE follows the DE/rand/1/bin strategy, where mutation is performed by perturbing a base solution with a scaled difference between two other individuals. Crossover is applied based on a probability CR, determining whether genes are inherited from the mutant or the original solution. An elitism mechanism retains the best solution found at each iteration. The mutation operator follows the equation:

$$\upsilon_{i,G+1} = x_{r1,G} + F \cdot (x_{r2,G} - x_{r3,G})$$

  where $\upsilon_{i,G+1}$ is the mutant vector for the next generation (G+1), $x_{r1,G}$ a randomly selected base vector from the current population, $x_{r2,G}$ and $x_{r3,G}$ two randomly chosen distinct vectors from the population and F the scaling factor.

- Particle Swarm Optimization (PSO): PSO optimizes solutions by treating them as particles moving through a search space. Each particle updates its position based on its own best-known position and the best-known position of the entire swarm. Velocity updates follow:

$$v_i^{t+1} = w\, v_i^t + c_1\, r_1 \left(p_{best,i}^t - x_i^t\right) + c_2\, r_2 \left(g_{best,}^t - x_i^t\right)$$

  where w is the inertia weight, c_1 and c_2 are acceleration coefficients, $p_{best,i}^t$ the position of the best solution the particle i has achieved so far, g_{best}^t the best position so far by any particle, and r_1, r_2 are random numbers between 0 and 1. The updated velocity adjusts each particle's position, and an elitist strategy retains the global best solution.

- Simulated Annealing (SA): SA is a probabilistic optimization algorithm inspired by the annealing process in metallurgy. At each iteration, a neighboring solution is generated through a mutation operation, and the algorithm decides whether to accept this new solution based on a probability calculated as $e^{-\Delta/T}$, where Δ is the difference in solution quality and T the current temperature. At high temperatures, SA explores widely by frequently accepting worse solutions, while at low temperatures, it behaves more like a greedy algorithm by rarely accepting worse solutions.

- Random Search (RS): RS serves as a baseline approach in which solutions are generated purely at random in each iteration, without any guided search strategy.

As an exact optimization method, CP formulates the scheduling problem using a constraint satisfaction framework. Constraints are modeled explicitly, and a solver systematically searches for feasible and optimal solutions. Unlike metaheuristic approaches, CP guarantees optimality but may become computationally expensive for large-scale problems.

All metaheuristic algorithms are parameter-tuned and integrated into a Discrete Event Simulation (DES) model. The implementation is performed using Python (DEAP, PySwarms) and Plant Simulation for evaluation. Their performance is compared against the built-in GA Wizard (Bojic et al. 2023), evaluating both solution quality and computational efficiency.

4. Results and Discussion

To carry out the simulations and obtain the results, the algorithms are coded in Python 3.9 using the DEAP library, PySwarms library, and the Python API provided by CPLEX. For the exact methods, CP Optimizer 22.1.0.0 was employed as optimization engine. Computational experiments were executed on a 14-core 12th Gen Intel(R) Core (TM) i7-12700H1576 2.70 GHz processor. The data used in the following experiments is based on real data provided by Navantia, which has been distorted for confidentiality reasons.

In addition to solving the main problem, which involved 90 preassembly elements, three additional scenarios have been configured to analyze the behavior of these optimization techniques as problem complexity increases. These scenarios have been set up as shown in Table 1.

Table 1. Data and configuration of the four scenarios for the job scheduling problem

	Nº Preassembly	DueDate Interval (s)	
Scenario 0 (main)	90	[26000	26000]
Scenario 1	100	[27328	40992]
Scenario 2	250	[69450	104175]
Scenario 3	500	[138608	207912]

As observed, the core of the problem is based on a fixed due date, by which all preassembly elements must complete both welding operations and any required inspections. In the three proposed scenarios, besides increasing the number of preassembly elements, the due dates have been constrained differently. Considering that the earliest and latest possible due dates correspond to the cases of performing no inspections and all inspections, respectively, each preassembly element has been assigned a random due date within a 20% interval, with its mean value positioned between these minimum and maximum due dates.

This approach forces the scheduling problem to search for the optimal solution while ensuring compliance with individual due dates, simultaneously determining whether an inspection can be performed or not. Accordingly, the objective function to be evaluated is defined as follows:

$$fitness = n_{inspections} - 1000 \cdot n_{delayed}$$

With $n_{inspections}$ and $n_{delayed}$ being the number of inspections and the number of preassemblies delayed, ensuring that, as previously mentioned, the problem lies in maximizing this function while seeking a positive fitness value. Otherwise, the constraint of not delivering any delayed preassemblies would not be satisfied.

Plant Simulation is used as the evaluation model for the objective function. In each iteration of the metaheuristic algorithm, it determines the number of delayed preassemblies. The evaluation model takes as input the proposed schedule—indicating the sequence of preassemblies and whether or not each one is inspected—as generated by the algorithm. This schedule is fed into the production model in Plant Simulation, which is integrated with the optimization framework via a Python interface.

Additionally, this hyperparameter search was performed by parallelizing the execution across multiple cores, so the average values of fitness and computation time will be presented.

5.1 Numerical Results

To obtain the results, a hyperparameter search was first performed in **Scenario 1**. As will be shown below, this approach allows for significantly faster evaluations.

5.1.1 Genetic Algorithm

The following hyperparameters were evaluated: n pop [50, 100]; n gen [100, 500, 1000]; and crossover, mutation, and individual probabilities of [0.2, 0.4, 0.6, 0.8]. Accordingly, Table 2 presents the results where the probability values were fixed at the best observed combination for different population sizes and generation numbers.

Table 2. Genetic Algorithm Results from Hyperparameter Search

n pop	n gen	prob. Cx	prob. Mut	prob. Indv	Fitness Mean
50	100	0.8	0.4	0.2	60
	500	0.8	0.4	0.2	62
	1000	0.8	0.4	0.2	62
100	100	0.8	0.4	0.2	62
	500	0.8	0.4	0.2	65
	1000	0.8	0.4	0.2	67

5.1.2 Differential Evolution

Similarly to the Genetic Algorithm, in Differential Evolution, the possible combinations were evaluated using the following hyperparameters: n_pop [50, 100]; n_gen [100, 500, 1000]; and scaling factor and crossover probability of [0.2, 0.4, 0.6, 0.8]. In this case, it was concluded that the optimal values for the scaling factor and crossover probability are 0.8 and 0.2, respectively, for which the results presented in Table 3 were obtained. It can be observed that for higher values of population size and number of generations, a significantly elevated mean fitness is achieved, outperforming the other proposed algorithms.

Table 3. Differential Evolution Results from Hyperparameter Search

n pop	n gen	prob. F	prob. CR	Fitness Mean
50	100	0.2	0.8	65
	500	0.2	0.8	74
	1000	0.2	0.8	74
100	100	0.2	0.8	66
	500	0.2	0.8	77
	1000	0.2	0.8	77

5.1.3 Particle Swarm Optimization

For the PSO, the hyperparameters corresponding to the PySwarms library have been set, with the number of neighbours (k) and the neighbourhood topology fixed at 3 and 1, respectively. Additionally, the following hyperparameter combinations have been evaluated: c_1 and c_2 [1, 1.5, 2]; w [0.4, 0.65, 0.9]; n particles [20, 40, 60]; and n iterations [500, 1000]. Accordingly, the optimal combination of acceleration coefficients and inertia was determined to be [2, 1.5, 0.4], with the corresponding results presented in Table 4. Additionally, a highly positive effect of increasing the number of particles on the optimization results can be observed.

Table 4. Particle Swarm Optimization Results from Hyperparameter Search

n particles	n iterations	C1	C2	w	Fitness Mean
20	500	2	1.5	0.4	59
	1000	2	1.5	0.4	60
40	500	2	1.5	0.4	64
	1000	2	1.5	0.4	62
60	500	2	1.5	0.4	64
	1000	2	1.5	0.4	69

5.1.4 Simulated Annealing

For the SA, a set of hyperparameters was considered, for which an initial temperature T_o and number of iterations were swept over the range of [1000, 10000], while the values [1, 10, 100] were selected for the number of iterations at temperature T_i and [0.8, 0.895, 0.99] for the cooling rate. Fixing the cooling rate at 0.99, the results shown in Table 5 are obtained.

Table 5. Simulated Annealing Results from Hyperparameter Search

To	n iter	n iter at Ti	alpha	Fitness Mean
1000	1000	10	0.99	56
		100	0.99	59
	10000	10	0.99	57
		100	0.99	58
10000	1000	10	0.99	57
		100	0.99	58
	10000	10	0.99	58
		100	0.99	56

5.1.5 CPO

On the other hand, the optimization problem has been solved using CP for all the scenarios presented in the problem. Table 6 shows the results of the CPO model across all scenarios, indicating that the global maximum inspection was achieved with the variables, constraints, and KPIs presented.

Table 6. CPO Results for All Scenarios

Scenario	Nº Variables	Nº Constraints	GAP (%)	CPU Time (s)	Nº Inspections
0	271	92	0.00 %	0.18	25
1	301	102	0.00 %	1.01	78
2	751	252	0.00 %	19.40	197
3	1501	502	0.00 %	30.05	389

The performance of the metaheuristic algorithms connected to PlantSimulation has been compared by rigorously selecting the previously evaluated hyperparameters for each one to keep computation times within a reasonable limit, which required reducing the number of algorithm evaluations. A purely random search was incorporated to assess their potential, along with the results of the GA Wizard optimization integrated within the DES model. As shown in Table 7, the GA Wizard not only has a very high computation time but also fails to find any solution that avoids delivering delayed preassemblies.

5.1.6 Comparative Analysis Across All Scenarios

Table 7. Results of the Optimization of the Metaheuristics and CPO Across All the Proposed Scenarios

	Scenario 0		Scenario 1		Scenario 2		Scenario 3	
	C.Time (s)	Best Fitness	C. Time (s)	Best Fitness	C. Time (s)	Best Fitness	C. Time (s)	Best Fitness
GA Wizard	1620	-1983	-	-	-	-	-	-
GA	1304	21	782	**59**	1218	**129**	1132	**223**
PSO	786	**22**	606	51	838	110	1013	197
DE	1322	21	806	52	1053	116	1293	199
SA	405	19	345	49	446	111	522	-2754
RANDOM	431	-16969	310	51	540	-894	510	-5772
CPO	0.2	25	1.0	78	19.4	197	30.0	389

It can be observed that the optimization of the problem using all the metaheuristic algorithms yields good results. For the problem with fewer preassemblies, the PSO is the metaheuristic that provides the best result based on the best fitness value obtained. On the other hand, as the problem size increases, the GA yields the best results. However, it is proposed to evaluate multiple algorithms in the optimization experiments to expand the range of possible solutions and select the one that provides the best result for the problem size to be solved.

When comparing CPO results with approaches that combine metaheuristics and simulation models, CPO proves to be an effective alternative for solving this type of problem. It has the potential to achieve high-quality solutions in shorter computation times. However, CPO presents several limitations compared to the combination of external algorithms with DES models. One key drawback is the potential loss of reliability due to the simplifications required in problem modelling. In this work, only tasks with or without inspections were considered, but for more complex processes with a high number and variety of tasks, the formulation may need to be simplified, potentially compromising the accuracy of the results. This trend is clearly reflected in Table 7, which illustrates how increasing the number of preassembly elements and tightening due dates lead to a significant rise in CPU time.

As a summary of the optimization techniques performance, the following affirmations can be done:

- GA Wizard underperforms compared to the alternative approaches, particularly in terms of reliability.
- GA outperforms other metaheuristics on larger instances, although it is noticeably slower than the others.
- PSO achieves the best performance for small instances, while also maintaining keeping a favourable ratio between fitness value and computation time for larger problem sizes.
- DE exhibits similar computation times to GA but achieves significantly lower fitness values across all scenarios.
- SA is the fastest metaheuristic implemented, yielding suitable solutions for small to medium-sized instances.
- CPO outperforms the other techniques, achieving optimal solutions within reasonable computation times.

By applying the proposed optimization environment and analyzing the results within Navantia's operational context, several practical implications can be identified.

If quick optimizations are required for a relatively small number of preassemblies, PSO and SA prove to be highly practical algorithms. On the other hand, if the objective is to achieve higher-quality solutions, GA is the most suitable choice, though at the cost of increased computation time.

On the other hand, CPO is ideal for processes that are not overly complex and involve moderately sized instances, as it can guarantee the optimal solution within a short time frame. This is particularly feasible when experienced modelers are available to quickly translate the process into a constraint programming model. However, for more complex processes, implementing CPO can become excessively difficult, and it may even fail to find a solution if the problem size is too large.

This highlights an additional advantage of using metaheuristics: Navantia already employs discrete event simulation in many of its production workflows. As such, incorporating metaheuristic-based optimization—especially approaches compatible with Plant Simulation—offers a highly promising and non-disruptive alternative. Nevertheless, CPO should remain under consideration as a future optimization tool, given the high quality of its solutions in problems of the kind analyzed in this study.

6. Conclusion

In this work, a welding job scheduling problem proposed by Navantia was addressed using optimization tools such as metaheuristics and exact methods. A comprehensive literature review was conducted to identify optimization techniques in manufacturing, with a particular focus on welding robotic cells used in shipbuilding. This review led to the implementation of Genetic Algorithm, Differential Evolution, Particle Swarm Optimization and Simulated Annealing alongside Constraint Programming using CPO. Furthermore, a highly effective integration between DES software and metaheuristics was developed, creating a framework that can be applied to decision-making in Navantia's scheduling problems, facilitating improvements in intermediate processes.

The metaheuristic algorithms demonstrated significant improvements over the GA Wizard. However, CPO achieved the most optimal solution, minimizing delays effectively, albeit with potential scalability limitations for highly complex scenarios.

A key contribution of this study is the successful integration of advanced optimization methods with Navantia's planning and monitoring procedures, ensuring that the proposed framework can be seamlessly adapted to real-world industrial settings. While CPO showcased strong potential for job scheduling, the flexibility and computational efficiency of metaheuristics reinforce their value, particularly for larger or more dynamic production environments.
Future work should explore hybrid optimization strategies that leverage both CPO and metaheuristics, potentially combining their strengths to handle more complex scheduling problems. Moreover, expanding the framework to integrate Navantia's existing information systems could further enhance its industrial applicability, paving the way for more intelligent and automated decision-making processes in shipbuilding manufacturing.

Acknowledgements

This work has been supported by Xunta de Galicia through Axencia Galega de Innovación (GAIN) by grant IN853C 2022/01, Centro Mixto de Investigación UDC-NAVANTIA "O estaleiro do futuro", which is ongoing until the end of September 2025. The support was inherited from both starting and consolidation stages of the same project throughout 2015-2018 and 2018-2021, respectively. This stage is also co-funded by ERDF funds from the EU in the framework of program FEDER Galicia 2021-2027.

References

Bojic, S., Maslaric, M., Mircetic, D., Nikolicic, S., and Todorovic, V., Simulation and Genetic Algorithm-based approach for multi-objective optimization of production planning: A case study in industry, Advances in Production Engineering & Management, vol. 18, no. 2, pp. 250-262, 2023.

Chaudhry, I. A. and Khan, A. A., A research survey: Review of flexible job shop scheduling techniques, International Transactions in Operational Research, vol. 23, pp. 551–591, 2015.

IBM ILOG CPLEX Python API, Available: https://www.ibm.com/docs/en/icos/22.1.2, Accessed on Nov. 14, 2024.

Da Col, G. and Teppan, E. C., Industrial-size job shop scheduling with constraint programming, Operations Research Perspectives, vol. 9, 2022.

Dauzère-Pérès, S., Ding, J., Shen, L., and Tamssaouet, K., The flexible job shop scheduling problem: A review, European Journal of Operational Research, vol. 314, no. 2, pp. 409–432, 2024.

DEAP documentation, Available: https://deap.readthedocs.io/en/master/, Accessed on Dec. 9, 2024.

Driss, I., Mouss, K. N., and Laggoun, A., A new genetic algorithm for flexible job-shop scheduling problems, Journal of Mechanical Science and Technology, vol. 29, no. 3, pp. 1273–1281, 2015.

Eberhart, R., and Kennedy, J., A new optimizer using particle swarm theory, in MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, pp. 39–43, 1995.

Forghani, K., and Fatemi Ghomi, S.M.T., Joint cell formation, cell scheduling, and group layout problem in virtual and classical cellular manufacturing systems, Appl. Soft Comput., vol. 97, 106719, 2020.

Gourdon, K., and Steidl, C., Global value chains and the shipbuilding industry, OECD Science, Technology and Industry Policy Papers, 2019.

Heinz, S., Ku, W. Y., and Beck, J. C., Recent improvements using constraint integer programming for resource allocation and scheduling, in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Vol. 7874 LNCS, pp. 12–27, 2012.

Ibrahem, A., Elmekkawy, T., and Qingjin, P., Robust Metaheuristics for Scheduling Cellular Flowshop with Family Sequence-Dependent Setup Times, Proceedings of the 47th CIRP Conference on Manufacturing Systems, 2014.

Laborie, P. 2009b. "IBM ILOG CP Optimizer for detailed scheduling illustrated on three problems." In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Vol. 5547 LNCS, pp. 148–162, 2009.

Laborie, P. 2009a. "IBM ILOG CP Optimizer for detailed scheduling illustrated on three problems," Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 5547 LNCS, pp. 148–162, 2009.

Laborie, P., "An update on the comparison of MIP, CP and hybrid approaches for mixed resource allocation and scheduling." In Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Vol. 10848 LNCS, pp. 403–411., 2018.

Liao, B., Lu, S., Jiang, T., and Zhu, X., A variable neighborhood search and mixed-integer programming models for a distributed maintenance service network scheduling problem, International Journal of Production Research, vol. 62, no. 20, pp. 7466-7485, 2024.

Lu, S., Ma, C., Kong, M., Zhou, Z., and Liu, X., Solving a Stochastic Hierarchical Scheduling Problem by VNS-Based Metaheuristic with Locally Assisted Algorithms, Applied Soft Computing, vol. 130, pp. 109719, 2022.

Luo, H., Huang, G. Q., Shi, Y., Qu, T., and Zhang, Y. F., Hybrid flowshop scheduling with family setup time and inconsistent family formation, International Journal of Production Research, vol. 50, no. 6, pp. 1457-1475, 2012.

Lunardi, W. T., Birgin, E. G., Ronconi, D. P., and Voos, H., Metaheuristics for the online printing shop scheduling problem, European Journal of Operational Research, vol. 293, no. 2, pp. 419-441, 2021.

Meng, L., Zhang, C., Ren, Y., Zhang, B., and Lv, C., Mixed-integer linear programming and constraint programming formulations for solving distributed flexible job shop scheduling problem, Computers & Industrial Engineering, vol. 142, pp. 106347, 2020.

Ghadiri Nejad, M., Shavarani, S.M., Güden, H., et al., Process sequencing for a pick-and-place robot in a real-life flexible robotic cell, International Journal of Advanced Manufacturing Technology, vol. 103, pp. 3613–3627, 2019.

Pezzella, F., Morganti, G., and Ciaschetti, G., A genetic algorithm for the Flexible Job-shop Scheduling Problem, Computers & Operations Research, vol. 35, no. 10, pp. 3202-3212, 2008.

PySwarms, Available: https://pyswarms.readthedocs.io/en/latest/, Accessed on March, 14, 2025.

Rose, C. D. and Coenen, J. M. G., Comparing four metaheuristics for solving a constraint satisfaction problem for ship outfitting scheduling, International Journal of Production Research, vol. 53, no. 19, pp. 5782–5796, 2015.

Sahin, B. and Alpay, S., Integrated cell formation and part scheduling: A new mathematical model along with two meta-heuristics and a case study for truck industry, Scientia Iranica, vol. 31, no. 11, pp. 888–905, 2024.

Schlenkrich, M. and Parragh, S. N., Solving large scale industrial production scheduling problems with complex constraints: an overview of the state-of-the-art, Procedia Computer Science, vol. 217, pp. 1028–1037, 2022.

Tao, N. R., Jiang, Z. H., Liu, J. F., Xia, B. X. and Li, B. H., A metaheuristic algorithm to transporter scheduling for assembly blocks in a shipyard considering precedence and cooperating constraints, Discrete Dynamics in Nature and Society, vol. 2019, 2019.

Verbiest, F., Cornelissens, T. and Springael, J., A matheuristic approach for the design of multiproduct batch plants with parallel production lines, European Journal of Operational Research, vol. 273, no. 3, pp. 933-947, 2019.

Wu, X. and Liu, X., Differential evolution algorithm for solving distributed flexible job shop scheduling problem, Jisuanji Jicheng Zhizao Xitong/Computer Integrated Manufacturing Systems, vol. 25, no. 10, pp. 2539-2558, 2019.

Xie, J., Gao, L., Peng, K., Li, X. and Li, H., Review on flexible job shop scheduling, IET Collaborative Intelligent Manufacturing, vol. 1, no. 3, pp. 67-77, 2019.

Xu, X., Li, L., Fan, L., Zhang, J., Yang, X. and Wang, W., Hybrid discrete differential evolution algorithm for lot splitting with capacity constraints in flexible job scheduling, Mathematical Problems in Engineering, vol. 2013.

Ying, K.-C., Lee, Z.-J., Lu, C.-C. and Lin, S.-W., Metaheuristics for scheduling a no-wait flowshop manufacturing cell with sequence-dependent family setups. The International Journal Of Advanced Manufacturing Technology, vol. 58, no. 5-8, pp. 671-682, 2011.

Yuan, Y. and Xu, H., Flexible job shop scheduling using hybrid differential evolution algorithms, Computers & Industrial Engineering, vol. 65, no. 2, pp. 246-260, 2013.

Zhang, Q., Manier, H. and Manier, M. A., A genetic algorithm with tabu search procedure for flexible job shop scheduling with transportation constraints and bounded processing times, Computers and Operations Research, vol. 39, no. 7, pp. 1713–1723, 2012.