Prince Jerome Christopher J ✉
Lingadurai K

https://doi.org/10.21278/TOF.492062623
ISSN 1333-1124
eISSN 1849-1391

THE HYBRID APPROACH OF A MACHINE LEARNING ALGORITHM FOR ADDRESSING THE PERMUTATION FLOW-SHOP SCHEDULING PROBLEM

Summary

The existing numerous challenges encompassing raw materials, labour, electricity, machining time, and customer constraints mean that engineering and industrial facilities are under constant pressure to meet demand and increase productivity in the manufacturing sector. Traditional optimisation methods often fall short in effectively addressing these challenges. Recognising the need for advanced solutions, this paper introduces the PFSP as a critical aspect of manufacturing planning, aiming to minimise makespan. Makespan minimisation in PFSP is particularly challenging due to varying processing times and the number of jobs. To tackle this issue, the study advocates for an investigation of machine learning algorithms, emphasising the application of a hybrid approach that integrates the strengths of the Q-learning algorithm and the iterated greedy algorithm (IGA). Hybridized Q-learning along with the IGA algorithm is proposed as a novel modified algorithm. By combining Q-learning with the iterated greedy algorithm, the QIGA enhances its ability to explore the solution space thoroughly. In order to determine the optimal sequence for processing "n" jobs on "m" machines and improve the efficiency of the permutation flow-shop scheduling problem based on makespan, the QIG algorithm was proposed. To evaluate its effectiveness, the algorithm was tested using Taillard benchmark problems. The findings indicate that the suggested algorithm's performance surpasses that of traditional heuristics and metaheuristics approaches and also matches the optimal upper bound makespan value in all instances. Leveraging this effectiveness, two case studies were undertaken, and their outcomes were compared with alternative algorithms. The derived findings and schedules not only contribute to efficient product completion with minimised makespan time but also lead to heightened productivity for manufacturers. QIGA can be integrated with automated manufacturing systems, facilitating real-time scheduling and decision-making. The integration supports Industry 4.0 initiatives and enhances the automation of production processes.

Key words: *makespan, heuristics, metaheuristics, flow shop*

1. Introduction

Scheduling in production plays a crucial role in manufacturing decision-making, alongside various other management tools. In the business context, production scheduling is recognised as a crucial connection, facilitating a seamless information flow between the tactical and operational

tiers of an organisation. Advanced Planning and Scheduling (APS) systems strive to develop effective scheduling solutions, aiming to lower production costs, enhance throughput, and guarantee punctual deliveries. This is achieved through the utilisation of advanced algorithms and optimisation approaches [1]. The experimental results showcase the superior performance of PSOVNS approaches over conventional PSO, crediting the effectiveness of the recently proposed velocity and position update formulas for their ability to pinpoint optimal solutions. On average, the PSOVNS-6 methods exhibit a 58.43% reduction in computational time compared to the original PSO, while concurrently enhancing solution quality by 11.71% [2].

The efficacy of MCO-GA becomes apparent when applied to tackle a scheduling challenge in an actual wood production facility. When compared to advanced metaheuristic algorithms like MGLS, OBL_HSA, and HSA, MCO-GA consistently outperforms them in over 60% of cases, demonstrating superior performance. Additionally, MCO-GA exhibits faster computational speed than OBL_HSA and MGLS. These results underscore the efficiency of MCO-GA in resolving the HFSPMT issue and highlight its capacity to improve coordination and scheduling in cooperative systems of manufacturing [3]. Conducting experiments to assess the efficacy and efficiency of the proposed approach reveals promising results. The algorithm proves both efficient and effective, as evidenced by the findings, showcasing its ability to identify viable solutions within a three-minute timeframe with a mere 1% gap from the lower bound [4]. The study focused on a sand-casting workshop engaged in the production of diverse castings, with particular emphasis on examining batch and single-piece coupling processing features. Addressing this issue, an enhanced cuckoo algorithm was introduced, accompanied by the development of a corresponding scheduling model. The primary aim of the model is to minimise the maximum completion time in the production process [5]. Proceeding to evaluate the effectiveness and the execution time of the suggested methods through the application of two commonly employed statistical tests – Wald and Analysis of Variance (ANOVA) – unmistakably establishes the PPSOGA algorithm as a dependable and computationally efficient approach [6].

The Taillard benchmark results revealed that, in comparison to five contemporary algorithms in the literature, our HGSA algorithm exhibited superior performance concerning the most widely recognised upper limits on the makespan. Ultimately, the criteria of the makespan were utilised to enhance 109 out of 120 problem instances [7]. Recent approaches using reinforcement learning show slow convergence and limited accuracy. To address this, the author proposes an expert-driven imitation learning model with a graph-based encoder for better feature representation. The model outperforms state-of-the-art reinforcement learning methods, reducing network parameters by 63% and narrowing the solution gap from 6.8% to 1.3% on average, especially in scenarios with up to 1000 jobs. [8]. The permutation flow-shop scheduling (PFSS) problem is NP-hard, with many algorithms struggling to achieve both accuracy and efficiency. To improve this, a hybrid grey wolf optimiser (HGWO) is proposed, featuring enhanced initialisation, a levy flight strategy for balance, and crossover, mutation, and critical block exchange to avoid local optima. A variable neighbourhood descent strategy further boosts convergence accuracy. Comparative experiments show that HGWO outperforms existing methods [9]. In comparison to Gupta's heuristic, this algorithm significantly outperforms in delivering results. It not only provides an enhanced partial makespan but also achieves the minimum makespan. Furthermore, the proposed algorithm presents multiple alternatives for optimising the final outcome [10]. Based on the experimental outcomes, the suggested QVNS-NSGA-II exhibits superior performance in quantity, quality, and computational efficiency of Pareto solutions compared to NSGA-II, an enhanced Jaya optimisation algorithm, and a modified MOEA/D. Moreover, the implications of sensitivity analysis are diverse from a managerial perspective. Implementing the suggested strategy can empower manufacturers in hybrid flow shops to enhance productivity and sustainability [11].

A comparison was made between the standard QL algorithm, the hybrid NEH-QL approach, and the proposed enhanced QL (IQL) which were evaluated based on their makespan value and stability in simulation experiments. The optimised outcomes unequivocally demonstrate the superiority of the suggested algorithm compared to other algorithms under consideration. In conclusion, despite the limited research in this domain, the QL algorithm emerges as a user-friendly method for addressing the permutation flow-shop scheduling problem (PFSP) [12]. The modified NEH heuristic has demonstrated superior performance compared to the Johnson-based heuristic for relatively small-sized problems. Upon conducting a comparative analysis of the two metaheuristics with a focus on neighbourhood exploration, it becomes evident that the iterated greedy (IG) algorithm achieves the optimal outcome for medium and large-sized problems [13]. The statistical analysis and extensive experiments demonstrate that the suggested multi-objective model surpasses other models in minimising the relative percentage deviation. Moreover, the study demonstrates that the BRIG algorithm, despite its simplicity, stands as a cutting-edge technique, surpassing the Particle Swarm Optimisation approach in performance [14]. The attained results, surpassing rival approaches despite their novelty, underscore the efficacy of the proposed HIGT. Additionally, by incorporating considerations such as setup times, time or machine breakdowns, and adjusting the assignment rule to align with the objective function, there is potential for introducing numerous future research studies to enhance the realism of the problem. Furthermore, exploring alternative blocking techniques becomes pertinent when investigating innovative production systems [15].

The literature review highlights the complexity of the permutation flow-shop scheduling problems (PFSP), which are NP-hard and are challenging to solve optimally. While heuristics and metaheuristics have been widely used, they often fall short of providing optimal solutions. To address this, researchers are increasingly exploring machine learning techniques. In this study, the integration of Q learning with the IG method was introduced to leverage the strengths of both algorithms. The ability of the Q-learning algorithm to adaptively learn and improve policies over time combined with the effectiveness of the IG method in guiding the search process toward high-quality solutions. This integrated approach aims to enhance the efficiency and accuracy of solving PFSP.

The paper is organised as follows: Section 2 presents the problem statement and formulates a mathematical model. In Section 3, a detailed discussion on the hybrid approach of the Q-learning algorithm is presented. The findings and discussions are delineated in Section 4, and validation of the case studies is conducted in Section 5. Section 6 elaborates on the study's conclusion and the future scope of the study.

2. Problem Statement

2.1 Permutation Flow-Shop Scheduling Problem

In the permutation flow-shop scheduling problem (PFSP), there exists a set $N = \{1, 2, \ldots, n\}$ comprising "n" number of independent jobs, which needs to be scheduled for processing across a set of "m" machines designated as $M = \{1, 2, \ldots, m\}$, as depicted in Fig.1. A solution for PFSP is represented as $\Pi := (\pi_1, \ldots, \pi_n)$, a permutation where each π_j denotes the job's index, indicating its position within the sequence Π. Additionally, $p_i\pi_j$ is used to reference the time needed to process job π_j on machine i. The primary objective when addressing PFSP is the identification of the most efficient permutation Π, which minimises a specific criterion, often linked to the reduction of makespan or total completion time. The PFSP is significant for optimising job scheduling across multiple machines within diverse industrial and operational settings. The following mathematical model [16] represents the permutation flow-shop sequencing problem (PFSP) for the calculation of the makespan value.

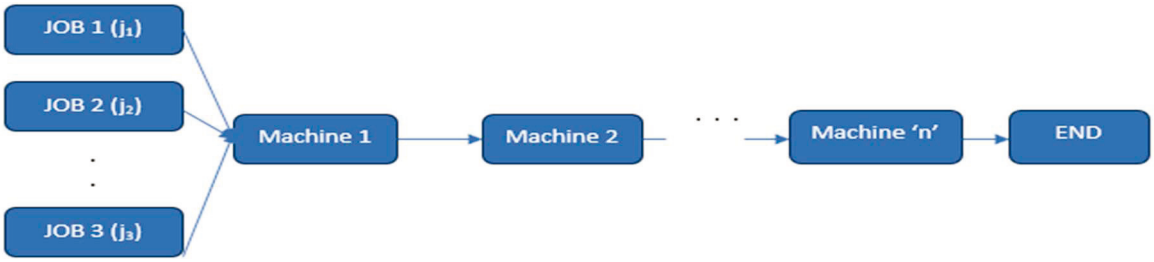

Fig. 1 Layout of the Permutation Flow-Shop Scheduling Problem

$$(j_1, i_1) = PT_{(j1,i1)},\ b=1,\ z=1 \tag{1}$$

$$c\,(j_b, i_1) = c\,(j_{b-1}, i_1) + PT\,(j_b, i_1),\ b = 2,3,\ldots,n. \tag{2}$$

$$c\,(j_b, i_z) = c\,(j_1, i_{z-1}) + PT\,(j_1, i_z),\ z = 2,3,\ldots,m. \tag{3}$$

$$(j_b, i_z) = \max\,(c\,(j_{b-1}, i_z),\ c\,(j_b, i_{z-1}))\ + PT(j_b, i_z),\ b=2,3,\ldots,n,\ z=2,3,\ldots,m. \tag{4}$$

The first equation seems to represent the completion time of the first job (j_1) on the first machine (i_1). It is given by the processing time (PT) of the job on that specific machine. The second equation defines the completion time for jobs (j_b) starting from the second job onwards (b = 2, 3, ..., n) on the first machine (i_1). The time required to finish each job is determined by adding the completion time of the preceding job (c (j_{b-1}, i_1)) to the processing time of the present job (PT (j_b, i_1)). The third equation represents the completion time for job j_b on machine i_z, where z ranges from 2 to m. The fourth represents a recurrence relation for calculating the completion time of job j_b on machine i_z in the context of a scheduling or optimisation problem.

3. Methodology

3.1 Hybrid Q – Learning Algorithm

The integration of Q-learning with the iterated greedy (IG) meta-heuristic addresses the NP-hard permutation flow-shop scheduling problem (PFSP) with the goal of minimising makespan, as illustrated in Fig. 2. This combination enhances the exploration capabilities of the IG algorithm by employing dynamic perturbation operators. In practice, utilising Q-learning with the IG algorithm involves several stages and introduces additional parameters, which increases computational demands. Despite challenges such as parameter optimisation and defining states, Q-learning significantly improves solution quality and reduces computational time. The Q-learning iterated greedy (QIG) algorithm innovates by incorporating an adaptive perturbation mechanism, contrasting with the original IG algorithm's static parameter approach.

A key innovation of QIG is its adaptive operator selection. Unlike the original IG algorithm, which uses a constant value for perturbation, QIG dynamically selects perturbation operators based on real-time search states and historical performance data. This dynamic adjustment allows QIG to enhance exploration and effectively address local optima by tailoring its perturbation strategy to the current search context.

In QIG, Q-learning plays a pivotal role by guiding decision-making through learned rewards. It assigns "credit" to perturbation operators based on their historical performance and immediate rewards from the PFSP environment. When selecting the next perturbation operator, QIG considers both the current search state and the credits assigned to operators. Operators with higher credits are more likely to be chosen, reflecting their past effectiveness in specific contexts. This dynamic selection process enables QIG to adjust its level of exploration as needed, improving its ability to explore solutions effectively and avoid local optima. Overall, QIG's adaptive approach enhances the static IG algorithm by making more informed decisions during the search process. This leads to better performance and a higher likelihood of discovering high-quality solutions, particularly for complex problems like PFSP.

Integrating Q-learning with the iterated greedy (IG) method combines the adaptive learning capabilities of Q-learning with the efficiency of the IG approach. This integration provides several benefits: it balances exploration and exploitation by leveraging Q-learning's ability to explore new solutions while the IG method focuses on immediate gains, which enhances solution quality. The IG method accelerates the search process, and when combined with Q-learning, it helps achieve quicker convergence on effective solutions. Moreover, Q-learning's adaptability contributes to the overall robustness of the approach, making it applicable to various problem instances.

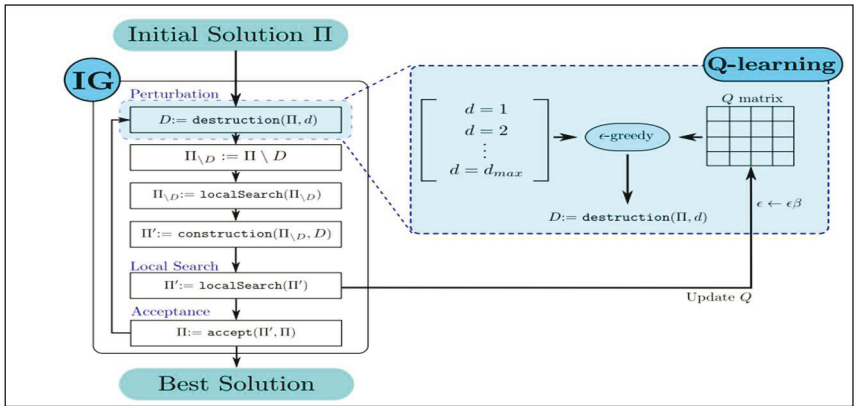

Fig. 2 Hybrid Q-Learning Algorithm

4. Experimental Design

The current segments outline a thorough series of experiments designed to evaluate the performance of the newly suggested QIG algorithm. The experiments involve a diverse range of permutation flow-shop scheduling problem (PFSP) instances. To assess the effectiveness of the algorithm, Taillard's benchmark instances from the OR Library were used. These instances vary in problem size, including 20x5, 20x10, 20x20, 50x5, 50x10, 50x20, and 100x5 configurations. Each benchmark instance is characterised by the number of machines, jobs, and their respective processing times. This benchmark set reflects the challenges encountered in real-time scenarios.

The selection of benchmark problems was made randomly to ensure a fair evaluation. The results obtained from running the proposed algorithm were then compared against the upper-bound optimum values established for these benchmark problems. The algorithm was implemented in Python, utilising a computing environment equipped with a CORE i5 processor, 8 GB of RAM, and a CPU operating at 2.3 GHz. This setup was chosen to ensure that the experiments are conducted under consistent hardware conditions.

Initially, experiments were conducted to determine the optimal process parameters. Table 1 displays the combinations of process parameters.

Table 1 Process Parameters

Temperature Scale (T)	Epsilon Greedy (ϵ)	Epsilon Decay (β)	Learning Rate (α)	Discount Factor (γ)	Size of Episode (E)	Local or Global Improvement Weight (η)
0.5	0.7	0.990	0.5	0.7	1	0.2
0.75	0.85	0.995	0.75	0.85	4	0.5
1	1	1	1	1	7	0.8

The effectiveness of meta-heuristics is significantly influenced by the settings of their parameters, which necessitate careful tuning. The parameter values have a direct impact on the

algorithm's capacity to maintain an equilibrium between exploration and exploitation in its quest to discover solutions that are close to, or even at, optimality.

The optimum parameters after carrying out the different experiments are: T = 0.7, $\epsilon = 0.8$, $\beta = 0.996$, $\alpha = 0.6$, $\gamma = 0.8$, $E = 6$, $\eta = 0.3$. Using these optimal parameters, the proposed algorithm was performed on various Taillard benchmark instances depicted in Table 2.

Table 2 Experimental Result of Various Benchmark Instances

Benchmark instances		Optimal Makespan	PSO	HGA	HGSA	PSONEHVNS	QL	IQL	QIG
Ta001	20x5	1278	1300	1486	1324	1297	1297	1278	1278
Ta002		1359	1376	1528	1442	1373	1366	1365	1361
Ta003		1081	1125	1460	1125	1100	1098	1098	1090
Ta004		1293	1364	1588	1469	1400	1307	1307	1300
Ta005		1236	1250	1449	1300	1291	1250	1240	1236
Ta011	20X10	1582	1735	2011	1713	1660	1594	1586	1584
Ta012		1659	1727	2166	1800	1718	1684	1678	1659
Ta013		1496	1940	1633	1555	1521	1517	1515	1513
Ta014		1378	1505	1811	1516	1434	1399	1399	1385
Ta015		1419	1559	1933	1573	1492	1450	1450	1429
Ta021	20X20	2297	2365	2973	2452	2331	2330	2311	2309
Ta022		2100	2279	2582	2280	2177	2111	2110	2104
Ta023		2326	2466	3013	2480	2387	2354	2342	2342
Ta024		2223	2362	3001	2370	2304	2248	2230	2229
Ta025		2291	2429	3003	2507	2358	2310	2302	2301
Ta031	50X5	2724	2773	3161	2773	2731	2729	2724	2724
Ta032		2834	2934	3432	2988	2906	2843	2842	2838
Ta033		2621	2676	3211	2705	2638	2631	2622	2622
Ta034		2751	2824	3339	2871	2785	2762	2755	2754
Ta035		2863	2873	3356	2912	2864	2864	2863	2863
Ta041	50X10	3025	3240	4318	3363	3059	3198	3073	3059
Ta042		2892	3093	4155	3260	3020	2934	2957	2955
Ta043		2864	3139	4322	3266	3055	2939	2939	2932
Ta044		3064	3236	4283	3389	3124	3115	3115	3092
Ta045		2986	3186	4322	3320	3129	3059	3057	3052
Ta051	50X20	3875	4192	6129	4387	4134	4105	4010	3976
Ta052		3715	4217	5788	5788	4067	3864	3835	3728
Ta053		3668	4178	5863	5863	3981	3808	3808	3789
Ta054		3752	4067	5958	5958	4052	3855	3844	3844
Ta055		3635	3999	5862	5862	3969	3815	3811	3741
Ta061	100X5	5493	5527	6397	5536	5493	5493	5493	5493
Ta062		5268	5327	6234	5302	5290	5290	5289	5289
Ta063		5175	5253	6121	5221	5213	5177	5177	5177
Ta064		5014	5078	6026	5044	5023	5023	5021	5021
Ta065		5250	5323	6200	5358	5253	5266	5265	5265

Subsequently, the result of the makespan value obtained from the suggested algorithm was evaluated with the results obtained from other metaheuristics approaches such as the particle swarm optimisation algorithm, the hybrid genetic algorithm, the hybrid genetic simulated annealing algorithm, the hybrid particle swarm optimisation algorithm, the Q-learning algorithm and the improved Q-learning algorithm and also with the upper bound optimum values of the benchmark instances.

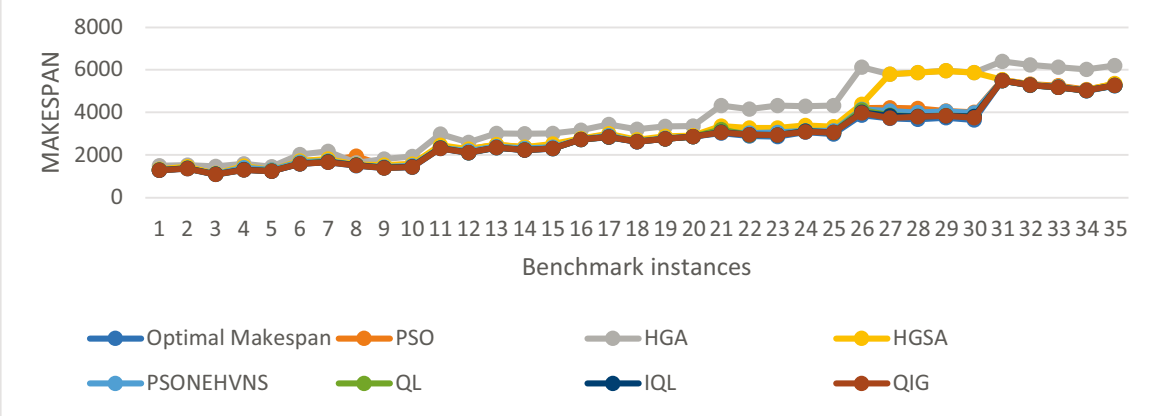

Fig. 3 Comparison of the Makespan Value of the Proposed Algorithm

Figure 3 illustrates that the makespan values achieved by the suggested algorithm match the upper bound values of the benchmark instances. Furthermore, when compared to the outcomes of other metaheuristic algorithms, the suggested algorithm demonstrates superior performance.

Traditional heuristics and metaheuristics often rely on fixed rules and can produce suboptimal solutions for complex or dynamic problems, whereas QIGA's adaptive learning and hybrid approach dynamically adjust to changing conditions, exploring a broader solution space and finding better solutions. Conventional metaheuristics may struggle with local optima and require extensive parameter tuning, but QIGA's reinforcement learning component improves its ability to escape local optima and offers greater flexibility for problem-specific tuning. Pure machine learning approaches may lack domain-specific insights and require substantial computational resources, whereas QIGA combines machine learning with the iterated greedy algorithm to integrate data-driven learning with expert knowledge, resulting in more efficient and effective scheduling solutions.

5. Evaluation Metrices

The evaluation of the proposed algorithm's performance involves the use of two key metrics: relative percentage deviation (RPD) and average relative percentage deviation (ARPD). RPD serves as a metric to gauge the relative difference between two values and is calculated using the formula:

$$\mathbf{RPD} = \mathbf{[(HS - BS) / BS] * 100},$$

where:

- HS represents the makespan value of the proposed algorithm.
- BS represents the optimum upper bound value.

The average relative percentage deviation (ARPD) is a measure of the average relative difference between a set of values and their corresponding reference values:

$$\mathbf{ARPD = RPD / N}.$$

The relative percentage deviation (RPD) and average relative deviation (ARD) are determined for the permutation flow shop problem (PFSP) to assess the performance of an algorithm in comparison to a reference or optimal solution. These metrics provide insights into the accuracy and efficiency of the solution method employed.

5.1 Comparative Results of the Evaluation Metrices

In Figures 4 and 5, the RPD and ARPD values of the proposed algorithm and other metaheuristics algorithms are presented. The findings indicate that when compared to other algorithms, the proposed algorithm demonstrates lower RPD – relative percentage deviation and ARPD – average relative percentage deviation.

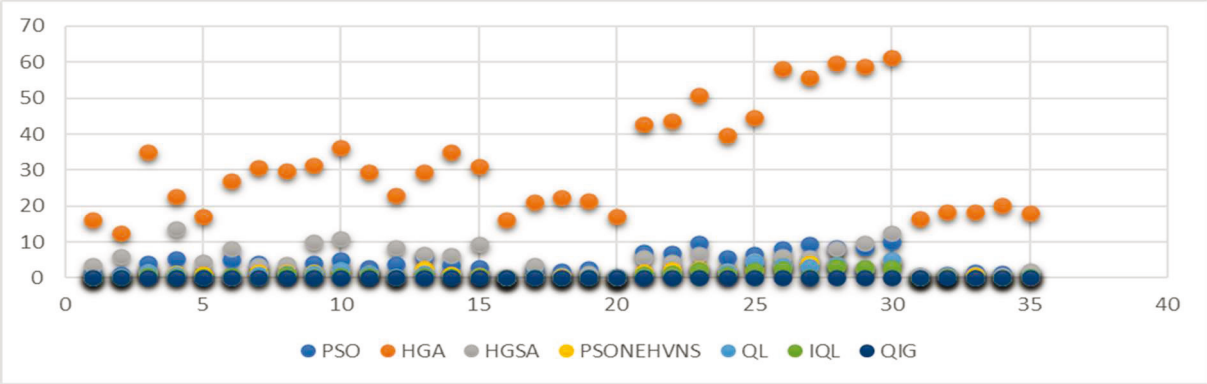

Fig. 4 RPD values of QIG and Metaheuristics Algorithm

Across all benchmark instances, the QIG algorithm consistently outperformed the others. These results underscore that the RPD and ARPD values for the proposed algorithm are notably smaller, highlighting its superior performance.

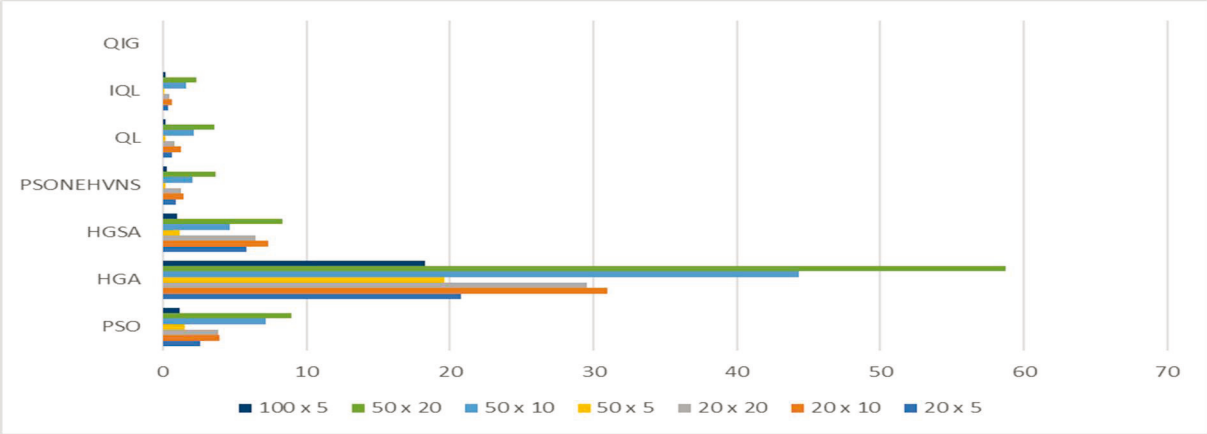

Fig. 5 ARPD values of QIG and Metaheuristics Algorithm

5.2 Box Plot

A box plot, alternatively referred to as a box-and-whisker plot, visually illustrates the distribution of a dataset. It offers a concise overview of important dataset features, encompassing the minimum, first quartile (Q1), median (second quartile, Q2), third quartile (Q3), and maximum. The box plot is particularly useful for comparing distributions and identifying the presence of outliers.

Figure 6 shows the comparison of the relative percentage deviation (RPD) values of different algorithms, including PSO, HGA, HGSA, PSONEHVNS, QL, IQL, and QIG. The QIG algorithm has a lower RPD value and a shorter box in the box plot compared to the other algorithms, which suggests that, on average, QIG performs better and has less variability in its predictions. It could be considered a more stable and accurate algorithm for the given task or dataset in terms of relative percentage deviation.

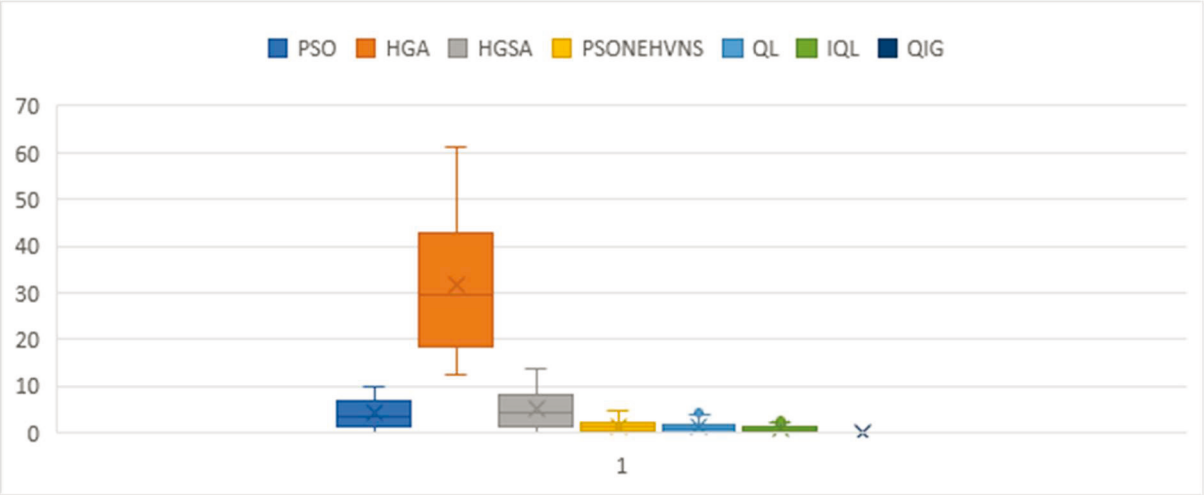

Fig. 6 Box Plot

6. Case Study Validation

6.1 Case Study 1: Plastic Toy Manufacturing

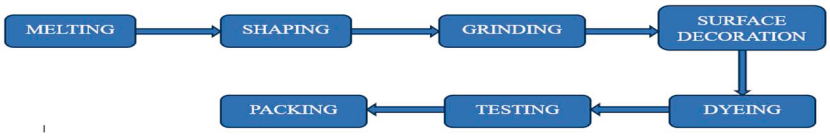

Fig. 7 Process Sequence of the Plastic Toy Manufacturing Industry

The algorithm's efficacy was assessed through its application to a case study in the plastic toy manufacturing industry. The company adopts a permutation flow-shop layout which is shown in Figure 7, where seven jobs are to be processed across seven machines, each with specified processing times as detailed in Table 2.

Table 2 Processing Time of all Jobs on Each Machine in the Plastic Toy Manufacturing Industry

Jobs / Machines	M/C 1	M/C 2	M/C 3	M/C 4	M/C 5	M/C 6	M/C 7
JOB 1	692	310	832	630	258	147	255
JOB 2	581	582	14	214	147	753	806
JOB 3	475	475	785	578	852	2	699
JOB 4	23	196	696	214	586	356	877
JOB 5	158	325	530	785	325	565	412
JOB 6	796	874	214	236	896	898	302
JOB 7	542	205	578	963	325	800	120

The problem was addressed using the QIG algorithm, and its performance was contrasted with that of the Q-learning and IQL algorithms. The results indicate that QIG outperformed the other algorithms, as evidenced by a superior makespan value of 6590 seconds and an optimal sequence of 4-3-1-5-6-2-0, accompanied by efficient computational times, as outlined in Table 3 and Figure 8.

Table 3 Performance of the Proposed Algorithm

Algorithm / Results	Optimal Makespan Value	Computational Time
Q – Learning	6622	0.34
IQL	6622	0.48
QIG	6590	0.051

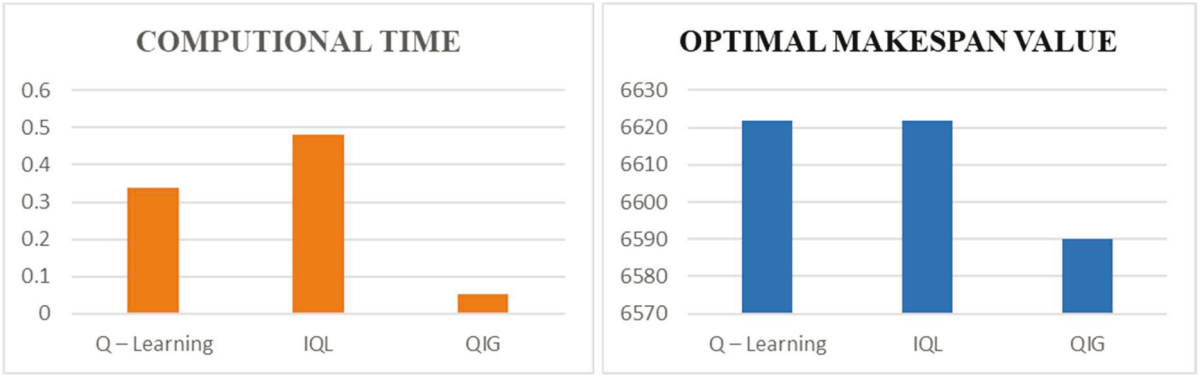

Fig. 8 Computational Time and Optimum Makespan Value

6.2 Case Study 2: Stator Core Manufacturing

The effectiveness of the suggested algorithm was assessed by implementing the above procedure in a case study in the stator core manufacturing industry. Data were gathered from a prominent automobile company in India, revealing a discrepancy between demand and production levels. To address this gap and enhance overall productivity, there is a need for an optimal schedule that minimises makespan. The number of jobs for processing, the sequence of machines, and the varying processing times are shown in Table 4.

Table 4 Processing Time of the Stator Core Manufacturing Industry

Part No	Blanking & Curling	Riveting	Coining	T'Bolt Milling	Groove Milling	Second Coining	OD-Turning
0	27	18	28	0	22	28	44
1	27	18	21	0	22	21	31
2	27	24	28	35	0	28	74
3	27	18	30	35	0	30	31
4	27	18	32	35	0	32	74
5	27	24	30	0	22	30	44
6	27	24	28	35	0	28	44
7	27	24	21	35	0	21	31
8	27	24	32	35	0	32	74

The optimal sequence for the above-mentioned problem is 1-0-7-5-3-4-2-8-6, resulting in a makespan value of 566 seconds. Table 5 illustrates the IN and OUT times for each job on every machine.

Table 5 In and Out Time of Each Job in Each Machine

Jobs	Blanking & Curling		Riveting		Coining		T'Bolt Milling		Groove Milling		Second Coining		OD-Turning	
	IN	OUT	IN	OUT	IN	OUT	IN	OUT	IN	OUT	IN	OUT	IN	OUT
1	0	27	27	45	45	66	66	66	66	88	88	109	109	140
0	27	54	54	72	72	100	100	100	100	122	122	150	150	194
7	54	81	81	105	105	126	126	161	161	161	161	182	194	225
5	81	108	108	132	132	162	162	162	162	184	184	214	225	269
3	108	135	135	153	162	192	192	227	227	227	227	257	269	300
4	135	162	162	180	192	224	227	262	262	262	262	294	300	374
2	162	189	189	213	224	252	262	297	297	297	297	325	374	448
8	189	216	216	240	252	284	297	332	332	332	332	364	448	522
6	216	243	243	267	267	295	332	367	367	367	367	395	522	**566**

In the manufacturing sector, there is a lack of adherence to a structured schedule, and attempts are made to meet demand through a trial-and-error approach. Establishing an optimal schedule is crucial for the manufacturer to efficiently process all jobs, minimising makespan time, thereby enhancing overall productivity and meeting demand more effectively.

7. Conclusion

- The permutation flow-shop scheduling problem, a combinatorial optimisation problem, aims to minimise the makespan value, where traditional approaches have proven inadequate for addressing its challenges. To address this, a hybrid approach utilising Q-learning algorithms was proposed and investigated in the context of solving the permutation flow-shop scheduling problem (PFSP), particularly by applying it to Taillard benchmark problems.
- The makespan value of the QIG algorithm aligns with the optimal upper bound value observed in benchmark instances. The results demonstrated that the suggested Q-learning iterated greedy algorithm (QIG) surpassed existing heuristics and meta-heuristics with less computational time. QIG can dynamically improve decision-making by balancing exploration and exploitation, thus avoiding local optima and potentially finding near-optimal solutions. Additionally, QIG's customisable reward structures can prioritise specific scheduling objectives, leading to improved solution quality and computational efficiency compared to traditional methods.
- Building on the success observed in benchmark instances, two case studies were conducted to implement these algorithms in real-world scenarios, including manufacturing plants where real-time scheduling decisions are crucial.
- The algorithms were subjected to testing with real-time data, and their performance was evaluated based on metrics such as makespan, efficiency, and practicality. The findings underscored the algorithm's effectiveness in practical settings, further highlighting its potential for application beyond theoretical benchmarks.
- In industrial settings, scheduling is primarily accomplished through trial-and-error methods. The findings and schedules derived from case studies aid manufacturers in achieving efficient product completion with minimised makespan time, ultimately leading to enhanced productivity.
- In future, the creation of a dynamic scheduling approach involves integrating the Q-learning algorithm to dynamically adapt and modify the production schedule in real-time, responding to variations in the production system or external factors. Expanding the practical application of the algorithm to diverse manufacturing settings and sectors is essential. Collaborating with industry partners can facilitate real-world testing and offer valuable insights.

REFERENCES

[1] Tasnim Mraihi, Olfa Belkahla Driss, Hind Bril EL-Haouzi, Distributed Permutation Flow Shop Scheduling Problem with Worker flexibility: Review, trends and model proposition, Expert Systems with Applications, Volume 238, Part C, 2024. https://doi.org/10.1016/j.eswa.2023.121947

[2] Kongkidakhon Worasan, Kanchana Sethanan, Rapeepan Pitakaso, Thitipong Jamrus, Karn Moonsri, Paulina Golinska-Dawson, A hybridization of PSO and VNS to solve the machinery allocation and scheduling problem under a machinery sharing arrangement, Intelligent Systems with Applications, Volume 18, 2023. https://doi.org/10.1016/j.iswa.2023.200206

[3] Yuxiang Guan, Yuning Chen, Zhongxue Gan, Zhuo Zou, Wenchao Ding, Hongda Zhang, Yi Liu, Chun Ouyang, Hybrid flow-shop scheduling in collaborative manufacturing with a multi-crossover-operator genetic algorithm, Journal of Industrial Information Integration, Volume 36, 2023. https://doi.org/10.1016/j.jii.2023.100514

[4] Yan Qiao, NaiQi Wu, YunFang He, ZhiWu Li, Tao Chen, Adaptive genetic algorithm for two-stage hybrid flow-shop scheduling with sequence-independent setup time and no-interruption requirement, Expert Systems with Applications, Volume 208, 2022. https://doi.org/10.1016/j.eswa.2022.118068

[5] Xixing Li, Xing Guo, Hongtao Tang, Rui Wu, Jiayi Liu, an improved cuckoo search algorithm for the hybrid flow-shop scheduling problem in sand casting enterprises considering batch processing, Computers & Industrial Engineering, Volume 176, 2023. https://doi.org/10.1016/j.cie.2022.108921

[6] Arash Amirteimoori, Iraj Mahdavi, Maghsud Solimanpur, Sadia Samar Ali, Erfan Babaee Tirkolaee, A parallel hybrid PSO-GA algorithm for the flexible flow-shop scheduling with transportation, Computers & Industrial Engineering, Volume 173, 2022. https://doi.org/10.1016/j.cie.2022.108672

[7] Wei, H.; Li, S.; Jiang, H.; Hu, J.; Hu, J. Hybrid Genetic Simulated Annealing Algorithm for Improved Flow Shop Scheduling with Makespan Criterion. Appl. Sci. 2018, 8, 2621. https://doi.org/10.3390/app8122621

[8] Longkang Li, Siyuan Liang, Zihao Zhu, Chris Ding, Hongyuan Zha, Baoyuan Wu. 2024 Learning to optimize permutation flowshop Scheduling via Graph - Based Imitation Learning. Proceedings of AAAI conference on Artificial Intelligence. VL – 38. https://doi.org/10.1609/aaai.v38i18.29998

[9] Shuilin Chen, Jianguo Zheng: Hybrid grey wolf optimizer for solving permutation flowshop scheduling problem. Wiley Concurrency and computation practice and Experience, 2024. https://doi.org/10.1002/cpe.7942

[10] Sandeep Kumar, Pooja Jadon International Journal of Computer Science and Information Technologies, Vol. 5 (4), 2014, 5057-5061.

[11] Yanhe Jia, Qi Yan, Hongfeng Wang, Q-learning driven multi-population memetic algorithm for distributed three-stage assembly hybrid flow shop scheduling with flexible preventive maintenance, Expert Systems with Applications, Volume 232, 2023. https://doi.org/10.1016/j.eswa.2023.120837

[12] Peize Li, Qiang Xue, Ziteng Zhang, Jian Chen, Dequn Zhou, Multi-objective energy-efficient hybrid flow shop scheduling using Q-learning and GVNS driven NSGA-II, Computers & Operations Research, Volume 159, 2023. https://doi.org/10.1016/j.cor.2023.106360

[13] Zimiao He, Kunlan Wang, Hanxiao Li, Hong Song, Zhongjie Lin, Kaizhou Ali Sadollah Gao, Improved Q-learning algorithm for solving permutation flow shop scheduling problems. IET Collab. Intell.Manuf.4(1),35-44(2022). https://doi.org/10.1049/cim2.12042

[14] Jabrane Belabid, Said Aqil, Karam Allali, "Solving Permutation Flow Shop Scheduling Problem with Sequence-Independent Setup Time", Journal of Applied Mathematics, vol. 2020, Article ID 7132469, 2020. https://doi.org/10.1155/2020/7132469

[15] Mohanad Al-Behadili, Djamila Ouelhadj & Dylan Jones (2020) Multi-objective biased randomised iterated greedy for robust permutation flow shop scheduling problem under disturbances, Journal of the Operational Research Society, 71:11, 1847-1859, https://doi.org/10.1080/01605682.2019.1630330

[16] Ahmed Missaoui, Younes Boujelbene 2020 An effective iterated greedy algorithm for blocking hybrid flow shop problem with due date window Oper. Res. 55 (2021), 1603-1616. https://doi.org/10.1051/ro/2021076

[17] Mohamed Abdel-Basset, Reda Mohamed, Mohamed Abouhawwash, Ripon K. Chakrabortty and Michael J. Ryan 2021, "A Simple and Effective Approach for Tackling the Permutation Flow Shop Scheduling Problem" Mathematics, 9, 270. https://doi.org/10.3390/math9030270

Submitted: 26.12.2023

Accepted: 24.10.2024

Prince Jerome Christopher J*
Principal cum Instructor
CSI Karur Industrial Training Institute,
Karur, India
Lingadurai K
Professor and Dean
Department of Mechanical Engineering,
Anna University, Regional Campus,
Madurai, India
*Corresponding author:
hardhickvihas@gmail.com