SINERGI Vol. 26, No. 2, June 2022: 137-144
http://publikasi.mercubuana.ac.id/index.php/sinergi
http://doi.org/10.22441/sinergi.2022.2.002

How to boost the flow shop manufacturing agility using hybrid Genetic Tabu Search in scheduling

Moch Saiful Umam[1*] Jutono Gondohanindijo[2]
[1]Magister Program of Information System, School of Postgraduate Studies, Diponegoro University, Indonesia
[2]Department of Informatics, Faculty of Engineering and Informatics, AKI University, Indonesia

Abstract
The hybridization between evolutionary genetic algorithm and tabu search has been proposed in this paper to address flow shop scheduling. It accommodates jobs that need to be rearranged and executed on identical machines serially. High agility is required in the manufacturing process, especially for the garment industry to be able to stand facing competitors. The manufacturing related to scheduling to deliver a product as early as possible, the tardiness, and waiting time are also concerned. A Genetic Algorithm was widely used to deal with this; which finds an optimal solution to the problems because it can obtain a more optimal solution. Unfortunately, it is easy to get stuck in optimum local (early convergence is faster). The tabu search algorithm works as a local explorer to better find and exploit the optimum local area, which can be combined with a Genetic Algorithm. This study aims to minimize the three objectives mentioned above to increase production agility. These strategies are evaluated on Taillard benchmark problems to show the significance of the proposed algorithm. The outcomes prove that the hybrid mechanism can boost the solution quality by 2.75% compared to our previous work and can resolve all of Taillard instances better. It has been proven by a 0.28% percentage relative deviation, which shows the error rate is lower and means better.

Keywords:
Agility;
Flow Shop;
Genetic Algorithm;
Production Scheduling;

Article History:
Received: May 2, 2021
Revised: December 15, 2021
Accepted: December 26, 2021
Published: June 1, 2022

Corresponding Author:
Moch Saiful Umam
Magister Program of Information System, School of Postgraduate Studies, Diponegoro University, Indonesia
Email: itgov@yandex.com

INTRODUCTION

The fashion industry's growth reached the international business level across all sectors – from independent boutiques to worldwide brands, from raw material to finished goods selling, and also comes from small retailers to big wholesalers [1]. The fashion industry business model must provide services or goods as quickly as possible to the shopper, known for its agility due to information technology innovation, short product cycles under competitive market pressures, changing consumer needs, and unreliable demand conditions [2]. Agility is strongly influenced by activities in the production process, especially in the scheduling of products to be made [3], so good production scheduling is needed to adapt to varying market changes. In addition, scheduling is one of the most critical elements in business administration and information systems because a good schedule allows management to have all necessary parts at hand when they are needed [4].

Production scheduling is defined as a process that combines sales forecasting and manufacturing planning [5]. They are important to control resources in a complex production environment with many different machines and jobs. While the two are separate, when combined, they have the potential to increase a business's productivity, accuracy, and agility [6]. A well-known scheduling problem that attracts many researchers is flow shop scheduling which is used when the product is similar in nature and requires almost identical processing time for each job, such as garment production [7]. A flow shop is a family of scheduling problems for which there is a precedence relationship between tasks and also capacity limits on resources along with precedence constraints [8]. Production scheduling means controlling the resources (machines) to produce useful results at minimum cost with maximum effectiveness. Unfortunately, a problem occurs in scheduling when several tasks have to be completed in a given time with limited resources. Still, there is no clear instruction on which tasks should be completed first and which later.

A Genetic Algorithm (GA) is widely utilized to deal with problems in scheduling [9][10] and is categorized as an evolutionary algorithm that is motivated by genetic evolution through natural selection [11]. The algorithm selects the optimal solution for its fitness or objective function and puts it into a new generation. In some cases, those solutions are then modified by operators such as crossover, mutation, or some other manipulating algorithms. The common sense of GA is that it copies the parent's strategy and modifies it for producing children, which are realistic solutions hard to find through common optimization algorithms. Unfortunately, GA often suffers from the drawbacks of premature convergence and weak exploitation capabilities trapped in optimum local [12].

In other places, there is a good algorithm in performing exploitation for flow shop where the goal is to seek for a result with high-quality criteria, rather than merely find a solution called with tabu search [13, 14, 15]. The Tabu Search (TS) idea is quite straightforward. The algorithm begins with a solution and then explores its neighborhood for a suitable new neighborhood. The search approach extends beyond local optimality to examine the solution space by allowing moves to neighbors with poorer makespans. The important aspects of the exploration track are selectively memorized (via a tabu list), and proactive decisions are taken so that they can steer the exploration away from optimum local toward different areas inside the solution space. If a halting criterion is met, then the algorithm terminates itself.

This study introduces a hybrid method to address the GA's shortcomings. These are tabu search algorithms combined with GA to increase the manufacturing agility in the garment. The rest of the article's structure is mentioned as the following. A concise summary of the research that has been done is presented in the materials. The method section also discusses the fusion of the proposed GA and TS algorithms. Then, experimental studies are presented in the next section, and conclusions with future works in the conclusion section.

METHOD

An improved hybridization between GA and TS methods is proposed for achieving manufacturing agility using three steps. First, data gathered from the Taillard dataset is partitioned into several subsets. Second, GA is applied to the first subset to get the initial population. Third, TS is utilized to enhance the solution quality found by GA.

The Proposed Algorithms

Throughout this section, the multi-objective scenario is given first as the paper's contribution to developing an optimization algorithm based on partial opposed-based learning. Notably, the initialization uses a partial opposed-based approach, consistent with our previous work [16]. There are a few parallels between them. In detail, both algorithms randomize, then divide the population into two sections and create one of them using an opposed-based strategy. According to our earlier work, this algorithm is only applicable to single-objective optimization problems. Although the number of objectives in this paper is significantly greater than in our prior work, it is designed to deal with multi-objective optimization problems. Thus, it is started by describing a multi-objective problem, followed by GA and TS parts.

Multi-Objective Problem

The paper considers the multi-objective optimizer for scheduling problems to discover a feasible solution under three different objectives: finding the shortest job completion time, minimal tardiness, and total waiting time between jobs at the same time. The optimal solution is declared as the following:

$$\text{Min } f_a(x), f_b(x), f_c(x) \tag{1}$$

Supplied with $x \in X$; where $f_a(x), f_b(x), f_c(x)$ are the objectives to be lessened, x is the decision vector, while the X is the decision space.

Inserting TS into GA

The key idea of the GA is based on the fact that human beings make decisions by comparing the current situation with the previous situation. In order to make GA more intelligent, TS helps GA by integrating TS into the GA process to result in a more optimal solution. It uses fine-tuned parameter configuration provided in Table 1 and contains some phases.

Table 1. Configuration for GA-TS

No	Configuration	Size
1	Iteration	1000
2	Population	100
3	Crossover	0.5
4	Mutation	0.1
5	Tournament length	5
6	Tabu length	5

Phase 1: encoding the solution
- 1.1 Gene ← set of job
- 1.2 Chromosome ← set of job sequence
- 1.3 Distribute time t needed on every gene

Phase 2: population initialization
- 2.1 Generate *n* size population randomly
- 2.2 Split population becomes two section
- 2.3 Process section part by partial opposition based strategy to take the best fitness by comparing opposite point

Phase 3: fitness checking
- 3.1 If iteration is not reached, do
- 3.2 Evaluate population by fitness objective function
- 3.3 Update the solution

Phase 4: parent selection
- 4.1 E ← elitist fitness
- 4.2 F ← current fittest solution
- 4.3 For an individual to tournament size, do
- 4.4 NF ← new fittest solution resulted from population
- 4.5 If NF > F
- 4.6 F ← NF
- 4.7 Return F and save to elitist

Phase 5: offspring generation
- 5.1 // crossover (two-point)
- 5.2 Randomly decide on two chromosome
- 5.3 Choose two barriers to exchanging chromosome
- 5.4 Exchange chromosome inside the barrier to generating solution
- 5.5 // mutation (swap)
- 5.6 Select two points in a chromosome
- 5.7 Swap the chosen point to generate a new solution

Phase 6: tabu search task
- 6.1 Rule the tabu list size, aspiration criteria, and stop regulation
- 6.2 Make a move to search solution space using insertion and swap strategy
- 6.3 Renew the solution using the best new one which is not saved in the tabu list
- 6.4 Stop exploration when stop regulation fulfilled

Phase 7: decoding the solution
- 7.1 Gathering job sequence
- 7.2 Gathering machine with processing time
- 7.3 Load Gantt chart construction

Visually, the algorithm runs using flow as shown in Figure 1, starting with solution encoding as a sequence table followed by initializing the population using the partial opposed-based method from our previous study. Then do a fitness check. This paper has three objectives function to minimize the makespan or the shortest of job finished time, minimal tardiness, and total waiting time, respectively:

$$\text{Min } f_1 = C(J_i, m) \tag{2}$$

$$\text{Min } f_2 = \sum_{i=1}^{j} \sum_{j=2}^{m} T(J_i, m) \tag{3}$$

$$\text{Min } f_3 = \sum_{i=1}^{j} \sum_{j=2}^{m} W(J_i, m) \tag{4}$$

Where:

j : set of job
m : set of machine
C(Ji, m) : completion time job J, machine m
T(Ji, m) : job J tardiness which is processed on machine m
W(Ji, m) : waiting time job J, machine m
($i = 1, 2, \ldots, n$) and ($j = 1, 2, \ldots, m$)

This study assumes that operation preemption is prohibited. Thus, a machine can execute a job operation if only the previous operation has already finished its process. Also, processing time may be zero because not all operations are processed on all machines. A flow shop is how to rearrange the order of the jobs using an identical machining sequence.

Next, two chromosomes are selected as a parent using tournament selection which is commonly used by the researcher [17, 18, 19], and the best chromosomes are saved to the elitist solution. It is intended to generate offspring using two-point crossover [20] and swap mutation [21] to maintain diversity for entering the reproduction phase 5.

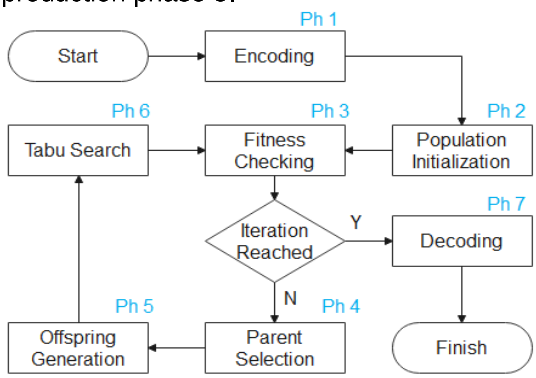

Figure 1. Proposed GA-TS Algorithm

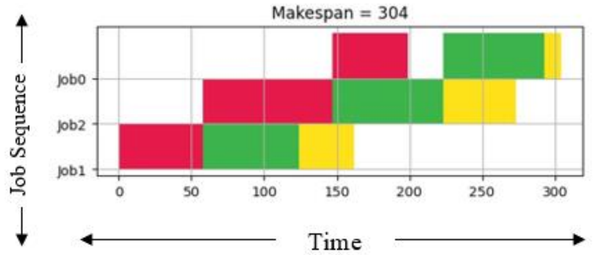

Figure 2. Example of Gantt Chart

After passing the genetic operator, the chromosome is processed by tabu search exploiting the solution space using the insertion and swap strategy [22] to result in the new solution. Finally, the algorithm decodes the optimal solution as the Gantt chart, which provides the job sequence and needs time to make products. Figure 2 shows the chart of how to schedule three different jobs and types of machinery.

RESULTS AND DISCUSSION

The hybrid GA-TS is applied to the Taillard problem [23] to provide a fair computational experiment throughout this section. This dataset is publicly available and contains the results of a competition to examine the best possible solution to a problem. The dataset contains 120 instances, each of which has a different solution. Therefore, the compared algorithms treated all instances for simplicity. The proposed algorithm is executed by employing a 1.9 GHz processor, using a memory of 4 GB, and coded in Python language. The best solution is presented in Table 2, which summarizes the findings. Additionally, for numerical analysis, we collect the Percentage of Relative Deviation (PRD) for the 120 instances over ten number runs to show the average error among a solution of the proposed algorithm and the lowest known upper bound values, where a lower PRD indicates a better algorithm, formulated as:

$$PRD = \frac{\sum_{i=1}^{N}\left(\frac{GATS_2-UB}{UB} x\,100\right)}{N} \quad (5)$$

Where:

N : number of instances
$GATS_2$: best solution from proposed GA-TS
UB : Taillard upper bound, the best known solution for Taillard

Regarding the hybrid strategy, GA has been hybridized to achieve one or more objectives with various algorithms, such as tabu search [16], which successfully improves the solution quality by 115 of 120 Taillard instances over hybrid genetic simulated annealing five other GA cooperation with PRD 3.05%. Thus, this proposed algorithm ($GATS_2$) is compared to the algorithm in our previous work ($GATS_1$), genetic algorithm variable neighborhood search (GAVNS) by [24], which is superior to the single variable neighborhood search algorithm, and finally compared to hybrid evolution strategy (HES_{SA}) studied by [25]. The proposed GA-TS solves each Taillard instance. The best-known upper bounds or optimal solutions for these problems are utilized to facilitate comparison. We compare algorithm performance using the percentage of increase (PI) between the current algorithm upper bound and Taillard upper bound, calculated as:

$$PI = \frac{GATS_2-UB}{UB} x100 \quad (6)$$

Here we go for the experiment result. Please note that the last column of Table 2 indicates the percentage of increase for $GATS_2$ compared to every Taillard upper bound and can be used to calculate PRD.

Table 2. Experimental Results

Problem	UB	GAVNS	$GATS_1$	HES_{SA}	$GATS_2$	PI
Tai001	1278	1486	1282	1278	1278	0,00
Tai002	1359	1528	1373	1359	1359	0,00
Tai003	1081	1460	1098	1081	1081	0,00
Tai004	1293	1588	1310	1293	1293	0,00
Tai005	1235	1449	1277	1235	1235	0,00
Tai006	1195	1481	1224	1195	1195	0,00
Tai007	1239	1483	1251	1239	1239	0,00
Tai008	1206	1482	1229	1206	1206	0,00
Tai009	1230	1469	1257	1230	1230	0,00
Tai010	1108	1377	1140	1108	1108	0,00
Tai011	1582	2044	1622	1582	1582	0,00
Tai012	1659	2166	1706	1659	1659	0,00
Tai013	1496	1940	1555	1496	1496	0,00
Tai014	1377	1811	1407	1377	1377	0,00
Tai015	1419	1933	1481	1419	1419	0,00
Tai016	1397	1892	1440	1397	1397	0,00
Tai017	1484	1963	1556	1484	1484	0,00
Tai018	1538	2057	1584	1538	1538	0,00
Tai019	1593	1973	1616	1593	1593	0,00
Tai020	1591	2051	1646	1591	1591	0,00
Tai021	2297	2973	2331	2297	2297	0,00
Tai022	2099	2852	2169	2099	2099	0,00
Tai023	2326	3013	2389	2326	2326	0,00
Tai024	2223	3001	2306	2223	2223	0,00
Tai025	2291	3003	2361	2291	2291	0,00
Tai026	2226	2998	2297	2226	2226	0,00
Tai027	2273	3052	2337	2273	2273	0,00
Tai028	2200	2839	2249	2200	2200	0,00
Tai029	2237	3009	2303	2237	2237	0,00
Tai030	2178	2979	2287	2178	2178	0,00
Tai031	2724	3161	2730	2724	2724	0,00
Tai032	2834	3432	2890	2836	**2834**	0,00
Tai033	2621	3211	2622	2621	2621	0,00

Problem	UB	GAVNS	GATS$_1$	HES$_{SA}$	GATS$_2$	PI
Tai034	2751	3339	2780	2751	2751	0,00
Tai035	2863	3356	2904	2863	2863	0,00
Tai036	2829	3347	2867	2829	2829	0,00
Tai037	2725	3231	2755	2725	2725	0,00
Tai038	2683	3235	2701	2686	**2683**	0,00
Tai039	2552	3072	2601	2552	2552	0,00
Tai040	2782	3317	2783	2782	2782	0,00
Tai041	2991	4274	3100	3024	3024	1,10
Tai042	2867	4177	3017	2882	2882	0,52
Tai043	2839	4099	3015	2852	2852	0,46
Tai044	3063	4399	3124	3063	3063	0,00
Tai045	2976	4322	3123	2982	2982	0,20
Tai046	3006	4289	3188	3006	3006	0,00
Tai047	3093	4420	3226	3122	**3099**	0,19
Tai048	3037	4318	3207	3042	**3038**	0,03
Tai049	2897	4155	3045	2911	**2902**	0,17
Tai050	3065	4283	3233	3077	3077	0,39
Tai051	3850	6129	4064	3889	3889	1,01
Tai052	3704	5725	3910	**3714**	3720	0,43
Tai053	3640	5862	3875	3667	3667	0,74
Tai054	3720	5788	3904	3754	3754	0,91
Tai055	3610	5886	3929	3644	3644	0,94
Tai056	3681	5863	3967	3708	3708	0,73
Tai057	3704	5962	3968	3754	3754	1,35
Tai058	3691	5926	3996	3711	3711	0,54
Tai059	3743	5876	4064	3772	3772	0,77
Tai060	3756	5958	3954	3778	3778	0,59
Tai061	5493	6402	5502	5493	5493	0,00
Tai062	5268	6240	5301	5268	5268	0,00
Tai063	5175	6133	5213	5175	5175	0,00
Tai064	5014	6025	5041	5014	5014	0,00
Tai065	5250	6198	5323	5250	5250	0,00
Tai066	5135	6087	5171	5135	5135	0,00
Tai067	5246	6255	5320	5246	5246	0,00
Tai068	5094	6130	5127	5094	5094	0,00
Tai069	5448	6381	5506	5448	5448	0,00
Tai070	5322	6384	5386	5322	5322	0,00
Tai071	5770	8079	5962	5776	**5770**	0,00
Tai072	5349	7886	5594	5360	**5349**	0,00
Tai073	5676	8028	5790	5677	5677	0,02
Tai074	5781	8348	5939	5792	**5781**	0,00
Tai075	5467	7958	5637	5467	5467	0,00
Tai076	5303	7814	5401	5311	**5304**	0,02
Tai077	5595	7866	5667	5596	5596	0,02
Tai078	5617	7913	5633	5625	5625	0,14
Tai079	5871	8166	5926	5891	**5875**	0,07
Tai080	5845	8117	5867	5845	5845	0,00
Tai081	6202	10700	6337	6257	6257	0,89
Tai082	6183	10594	6403	6223	6223	0,65
Tai083	6271	10611	6509	6342	**6325**	0,86
Tai084	6269	10607	6409	6303	6303	0,54
Tai085	6314	10539	6425	6380	6380	1,05
Tai086	6364	10677	6419	**6427**	6431	1,05
Tai087	6268	10835	6428	6306	6306	0,61
Tai088	6401	10840	6540	6472	6472	1,11
Tai089	6275	10723	6611	6380	**6330**	0,88
Tai090	6434	10798	6514	6485	**6456**	0,34

Problem	UB	GAVNS	GATS$_1$	HES$_{SA}$	GATS$_2$	PI
Tai091	10862	15319	11103	10872	10872	0,09
Tai092	10480	15126	10637	10487	10487	0,07
Tai093	10922	15398	11220	10941	**10922**	0,00
Tai094	10889	15240	11075	10889	10889	0,00
Tai095	10524	15259	10756	**10524**	10526	0,02
Tai096	10326	15116	10465	10346	**10330**	0,04
Tai097	10854	15415	11174	10868	10868	0,13
Tai098	10730	15279	11002	10741	**10731**	0,01
Tai099	10438	15135	10721	**10451**	10454	0,15
Tai100	10657	15340	10785	10680	10680	0,22
Tai101	11195	19740	11528	11287	**11280**	0,76
Tai102	11203	20112	11650	11277	**11272**	0,62
Tai103	11281	19937	12163	11418	**11378**	0,86
Tai104	11275	19961	12098	11376	11376	0,90
Tai105	11259	19849	11979	11365	**11310**	0,45
Tai106	11176	19942	11651	11330	**11265**	0,80
Tai107	11360	20112	11957	**11398**	11430	0,62
Tai108	11334	20064	11716	11433	**11398**	0,56
Tai109	11192	19918	12120	11356	**11265**	0,65
Tai110	11288	19942	12004	11446	**11355**	0,59
Tai111	26059	-	26771	26187	26187	0,49
Tai112	26520	-	27014	26799	**26779**	0,98
Tai113	26371	-	27491	26496	**26494**	0,47
Tai114	26456	-	26902	**26612**	26618	0,61
Tai115	26334	-	26790	26514	**26500**	0,63
Tai116	26477	-	27297	26661	**26647**	0,64
Tai117	26389	-	26758	26529	26529	0,53
Tai118	26560	-	27134	**26750**	26772	0,80
Tai119	26005	-	27636	26223	26223	0,84
Tai120	26457	-	27049	26619	**26617**	0,60

Table 2 demonstrates that the GA-TS method makes deeper exploitation inside the solution space and discovers a higher quality of solutions. Visual charts of the data can be seen in Figure 3 for 1 to 60 instances and Figure 4 for 61 to 120 instances. We know from these two figures that HES$_{SA}$ and GATS$_2$ are very close in the resulting solution, so we bold the better solution between the two algorithms in Table 2.

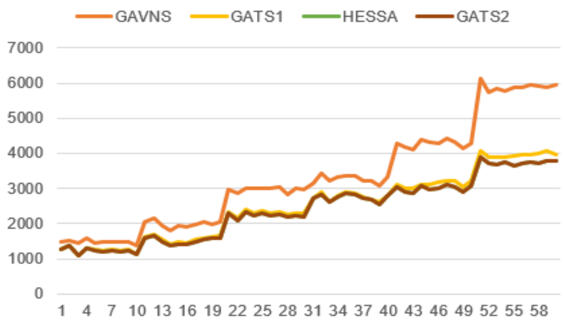

Figure 3. Solution for Tai001 – Tai060

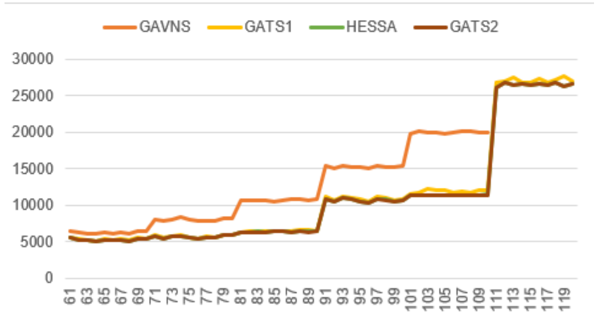

Figure 4. Solution for Tai061 – Tai120

From the computation time used, we can average the time needed in Table 3. The table shows that the more significant problem instance to be solved, it needs more time for the proposed GA-TS to solve.

Based on Table 2, the PI column's sum can be calculated that the sum is 33.46. So, the PRD for the GA-TS algorithm is calculated as follows.

$$PRD = \frac{33.46}{120} = 0.28\%$$

A comparison of PRD for another algorithm is presented in Table 4.

There is a difference between GAVNS, it is only used for the first 110 of the Taillard instance, so we calculate for the $GATS_2$ twice to make a fair comparison. The first contains 120 instances, and the second contains the first 110 instances. The result that both GA-TS using 110 or 120 instances achieve the lowest PRD, which is the best, can be visually conferred in Figure 5.

The main goal of this hybridization is to combine the power and flexibility of both algorithms so that each algorithm contributes to the solution process to solve the objective function. Therefore, any problem solved using both GA and TS can benefit from this algorithm since its hybridization feature brings significant improvements in solution quality.

Table 3. Average Execution Time of GA-TS

Problem	Time (s)	Problem	Time (s)
20 ~ 5	0,994	100 ~ 5	9,797
20 ~ 10	2,337	100 ~ 10	27,335
20 ~ 20	5,872	100 ~ 20	65,923
50 ~ 5	3,671	200 ~ 10	76,429
50 ~ 10	13,138	200 ~ 20	246,892
50 ~ 20	34,058	500 ~ 20	839,868

Table 4. PRD Comparison

Algorithm	Instance	∑ PI	PRD (%)
GAVNS	110	4588,16	41,71
GATS1	120	366,09	3,05
HESSA	120	40,67	0,34
GATS2-120	120	33,46	0,28
GATS2-110	110	26,87	0,24

Table 5. Percentage of Increase

Problem Size	PI of $GATS_2$ Compared to		
	GAVNS	$GATS_1$	HES_{SA}
20 ~ 5	21,10	1,78	0,00
20 ~ 10	31,01	3,15	0,00
20 ~ 20	32,97	3,04	0,00
50 ~ 5	19,50	0,98	0,02
50 ~ 10	42,81	4,52	0,12
50 ~ 20	57,70	5,97	-0,02
100 ~ 5	18,67	0,85	0,00
100 ~ 10	42,43	2,00	0,09
100 ~ 20	68,43	1,75	0,14
200 ~ 10	42,96	2,04	0,04
200 ~ 20	76,10	4,89	0,32
500 ~ 20	-	2,06	0,01
Average	**41,24**	**2,75**	**0,06**

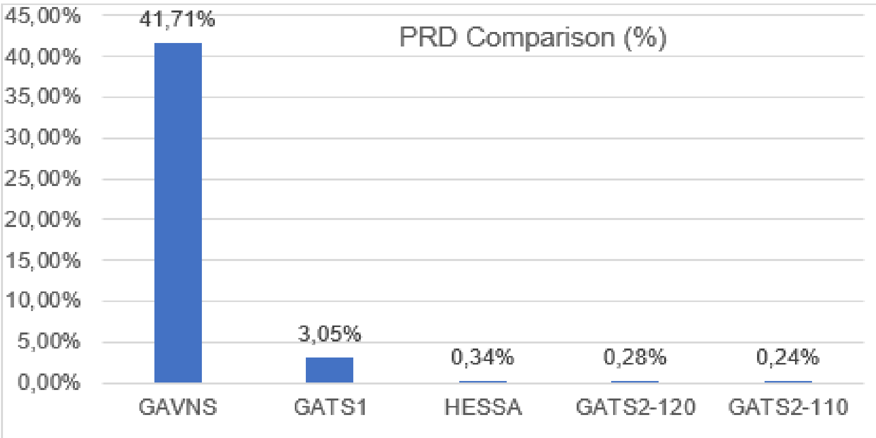

Figure 5. PRD Comparison

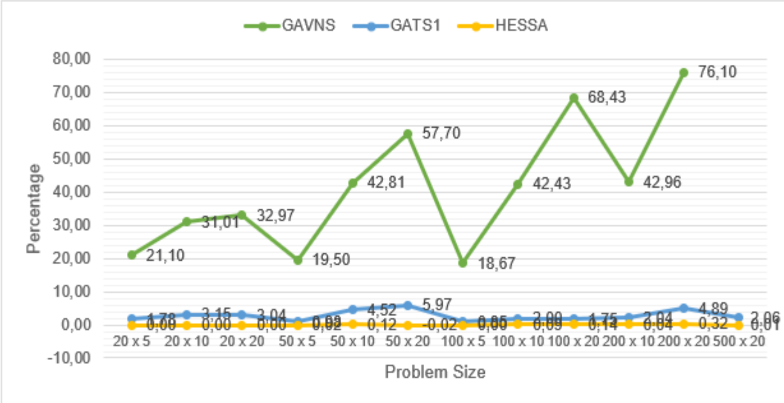

Figure 6. PI of GA-TS Compared

We can measure the improvement using a percentage increase provided in Table 5, which compares the proposed GA-TS ($GATS_2$) algorithm improvement with other algorithms.

The GA-TS algorithm can improve solution quality in all Taillard instances. Figure 6 shows the percentage of achieved improvement. The GA-TS improved HESSA by 0.06% on average, improved the GAVNS by 41.24% on average, and improved our previous work on GA-TS by 2.75% on average.

CONCLUSION

This paper gave GA-TS hybridization to deal with the scheduling problem of flow shop to improve solution quality. By using our partial opposed initialization in GA, it is effective in addressing multi-objective scenarios. In addition, proper parameter tuning such as two-point crossover and simple swap mutation help GA-TS to perform their task easily. By employing the insertion and swap method for TS, the GA perform exploitation deeply and results in better solution quality. The experiment shows that the GA-TS has the lowest PRD (0.28%) and can improve all Taillard instances. Also can improve the previous work by 2.75%. Future work will examine the extensive use of another evolutionary algorithm combined with GA because of its superiority.

REFERENCES

[1] B. Ly, "Competitive advantage and internationalization of a circular economy model in apparel multinationals," *Cogent Business & Management*, vol. 8, no. 1, p. 1944012, Jan. 2021, doi: 10.1080/ 23311975.2021.1944012.

[2] G. Gonda, E. Gorgenyi-Hegyes, R. J. Nathan, and M. Fekete-Farkas, "Competitive factors of fashion retail sector with special focus on SMEs," *Economies*, vol. 8, no. 4, p. 95, Nov. 2020, doi: 10.3390/ economies8040095.

[3] E. S. A. Nasr, A. Afify, M. Osman, and S. A. Elatty, "Scheduling flow shop manufacturing systems considering agility issues," *International Journal of Intelligent Collaborative Enterprise*, vol. 4, no. 3, p. 205, 2014, doi: 10.1504/ijcent.2014.066331.

[4] P. Jonsson and L. Kjellsdotter Ivert, "Improving performance with sophisticated master production scheduling," *International Journal of Production Economics*, vol. 168, pp. 118–130, Oct. 2015, doi: 10.1016/ j.ijpe.2015.06.012.

[5] S. Zheng, J. Gao, and J. Xu, "Research on production planning and scheduling based on improved collaborative optimization," *Concurrent Engineering: Research and Applications (CERA)*, vol. 27, no. 2, pp. 99–111, Jun. 2019, doi: 10.1177/ 1063293X19842253.

[6] Budianto, Surachman, D. Hadiwidjojo, and Rofiaty, "The effect of manufacturing agility competencies on lean manufacturing in increasing operational performance," *Uncertain Supply Chain Management*, vol. 9, no. 1, pp. 195–204, 2021, doi: 10.5267/ j.uscm.2020.10.001.

[7] W. K. Wong, C. K. Chan, and W. H. Ip, "A hybrid flowshop scheduling model for apparel manufacture," *International Journal of Clothing Science and Technolog*, vol. 13, no. 2, pp. 115–131, 2001, doi: 10.1108/09556220110390737.

[8] A. Goli, E. Babaee Tirkolaee, and M. Soltani, "A robust just-in-time flow shop scheduling problem with outsourcing option on subcontractors," *Production & Manufacturing Research*, vol. 7, no. 1, pp. 294–315, Jan. 2019, doi: 10.1080/ 21693277.2019.1620651.

[9] F. Werner, "Genetic algorithms for shop

scheduling problems: A survey," in *Mathematical Research Summaries*, vol. 2, pp. 15, 2017

[10] A. Ariani and S. Samsuryadi, “Classification of kidney disease using Genetic Modified KNN and Artificial Bee Colony algorithm,” *SINERGI*, vol. 25, no. 2, pp. 177-184, 2021, doi: 10.22441/sinergi.2021.2.009

[11] F. A. Omara and M. M. Arafa, "Genetic algorithms for task scheduling problem," Journal of Parallel and Distributed Computing, vol. 70, no. 1, pp. 13-22, 2010, doi: 10.1016/j.jpdc.2009.09.009

[12] I. Vlašić, M. Đurasević, and D. Jakobović, "Improving genetic algorithm performance by population initialisation with dispatching rules," *Computers & Industrial Engineering*, vol. 137, no. 1, pp. 1–38, Nov. 2019, doi: 10.1016/j.cie.2019.106030.

[13] V. A. Armentano and J. E. C. Arroyo, "An application of a multi-objective tabu search algorithm to a bicriteria flowshop problem," *Journal of Heuristics*, vol. 10, no. 5, pp. 463–481, 2004, doi: 10.1023/ B:HEUR.0000045320.79875.e3.

[14] B. Ekşioğlu, S. D. Ekşioğlu, and P. Jain, "A tabu search algorithm for the flowshop scheduling problem with changing neighborhoods," *Computers & Industrial Engineering*, vol. 54, no. 1, pp. 1–11, Feb. 2008, doi: 10.1016/j.cie.2007.04.004.

[15] S. H. Jayady, and H. Antong, “Theme Identification using Machine Learning Techniques,” *Journal of Integrated and Advanced Engineering (JIAE)*, vol. 1, no. 2, pp. 123-134, 2021, doi: 10.51662/jiae.v1i2. 24

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

[17] E. C. Osuna and D. Sudholt, "Runtime analysis of restricted tournament selection for bimodal optimisation," *Evolutionary Computation*, pp. 1–26, 2021, doi: 10.1162/evco_a_00292.

[18] X. Han, L. Zheng, L. Wang, H. Zheng, and X. Wang, "Fireworks algorithm based on dynamic search and tournament selection," *International Journal of Computers and Applications*, vol. 43, no. 6, pp. 577–588, 2021, doi: 10.1080/1206212X. 2019. 1590034.

[19] A. Shukla, H. M. Pandey and D. Mehrotra, "Comparative review of selection techniques in genetic algorithm," *2015 International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE)*, 2015, pp. 515-519, doi: 10.1109/ABLAZE.2015.7154916.

[20] B. Kiraz, A. A. Bidgoli, H. Ebrahimpour-komleh and S. Rahnamayan, "A Novel Collective Crossover Operator for Genetic Algorithms," *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2020, pp. 4204-4209, doi: 10.1109/SMC42975.2020.9282841.

[21] K. W. Kim, M. Gen, and G. Yamazaki, "Hybrid genetic algorithm with fuzzy logic for resource-constrained project scheduling," *Applied Soft Computing*, vol. 2, no. 3, pp. 174–188, 2003, doi: 10.1016/S1568-4946(02)00065-0.

[22] S. J. Beaty, "Genetic algorithms versus tabu search for instruction scheduling," *Artificial Neural Nets and Genetic Algorithms*, pp. 496–501, 1993, doi: 10.1007/978-3-7091-7533-0_72.

[23] E. Taillard, "Benchmarks for basic scheduling problems," *The European Journal of Operational Research (EJOR).*, vol. 64, no. 2, pp. 278–285, 1993, doi: 10.1016/0377-2217(93)90182-M.

[24] B. Jarboui, M. Eddaly, and P. Siarry, "A hybrid genetic algorithm for solving no-wait flowshop scheduling problems," *International Journal of Advanced Manufacturing Technology*, vol. 54, no. 9–12, pp. 1129–1143, 2011, doi: 10.1007/s00170-010-3009-4.

[25] B. Khurshid, S. Maqsood, M. Omair, B. Sarkar, I. Ahmad and K. Muhammad, "An Improved Evolution Strategy Hybridization With Simulated Annealing for Permutation Flow Shop Scheduling Problems," in *IEEE Access*, vol. 9, pp. 94505-94522, 2021, doi: 10.1109/ACCESS.2021.3093336.