Makespan Minimization of Machines and Automated Guided Vehicles Schedule Using Binary Particle Swarm Optimization

Muhammad Hafidz Fazli bin Md Fauadi and Tomohiro Murata

***Abstract*— An efficient and optimized Automated Guided Vehicles (AGVs) operation plays a critical role in improving the performance of a Flexible Manufacturing System (FMS). Among the main elements in the implementation of AGV is task scheduling. This is due to the fact that efficient scheduling would enable the increment of productivity and reducing delivery cost whilst optimally utilizes the entire fleet. In this research, Binary Particle Swarm Optimization (BPSO) is used to optimize simultaneous machines and AGVs scheduling process with makespan minimization function. It is proven that the method is capable to provide better solution compared to others.**

***Index Terms*— Flexible Manufacturing System, Automated Guided Vehicle, Particle Swarm Optimization**

I. Introduction

Rapid development of information technology has made the competition in manufacturing industry becoming more complex and stiff. Manufacturers are able to deliver product in relatively shorter time than ever. Thus, to win market share, managing information of a manufacturing company is very crucial in order to ensure that the information could be use when the company needs them.

Over the last few years, researchers had been intensely discussing about the implementation of Flexible Manufacturing System (FMS). While there are certain scientific advancements made, it is obvious that for the implementation to be a success, there are many problems need to be resolved. One of them is regarding simultaneous scheduling of machines and AGV operation.

There are many elements of FMS scheduling. However, the more important factor that should be considered is scheduling of multiple AGV. This is due to the fact that in typical shop floor environment, AGV is shared by more than one machine. Assigning a non-optimal delivery would put other machines in longer idle time than it should be. On the other hand, delaying a delivery means delaying the processing chain of the material. Furthermore, efficient AGV task allocation method would be able to increase productivity and reduce delivery cost whilst optimally utilizes the entire fleet. It is especially important when dealing with large fleet of AGVs.

Among the researches conducted on AGV scheduling discipline are on hybrid approach to address scheduling and routing of AGV [1-3], multi-attribute dispatching rules [4-6] and deadlock-resolution [7, 8].

II. Machines and AGV Scheduling for Resource-Constrained FMS

A. Introduction to FMS

FMS is a highly automated machine cell, consisting of a group of processing workstations (usually CNC machine tools), interconnected by an automated material handling, automated storage system and controlled by a distributed computer system (Groover, 2004). FMS is the key to an automated factory.

Since the term FMS was coined, various numbers of researches had been done in order to increase the capability and to explore the potential it could bring. Although there are significant advancement had been achieved, there are still plenty rooms for improvement to ensure that the benefit could be fully gained.

B. Resource-Constraint FMS

Resource-constrained FMS scheduling problem inherits the characteristics of combinatorial problem. Utilizing limited number of machines and automated vehicles, the main goal is to search for the best solution to solve a given set of problems. Over the years, it has attracted attentions from worldwide researchers. Typically mathematical optimization or heuristic methods had been applied to solve the problem rather than theoretical method. This is due to the reason that they are more applicable to be applied in actual environment. One of the approaches normally used to solve the problem is constrained optimization technique.

This study is based on single objective function where total operation completion time is the parameters that need to be minimized. Total operation completion time,

$$O_{ij} = T_{ij} + P_{ij}, \quad (1)$$

where i= job, j= operation, T_{ij}= traveling time, P_{ij}= operation processing time.

Manuscript submitted on December 16, 2009. This work was partially supported by the Public Services Department of Malaysia (JPA) and Universiti Teknikal Malaysia Melaka (UTeM).

Muhammad Hafidz Fazli bin Md Fauadi is a PhD student in Graduate School of Information, Production and Systems Engineering, Waseda University. (Corresponding author's address: 2-7 Hibikino, Wakamatsu-ku, Kitakyushu-shi, Fukuoka, Japan. E-mail: hafidz.waseda@yahoo.com).

Tomohiro Murata is serving as a Professor in Graduate School of Information, Production and Systems Engineering, Waseda University, Japan. (E-mail: t-murata@waseda.jp)

Job completion time, $C_i = \sum_{i=1}^{n} O_{ij}$ (2)

Makespan = Max (C1, C2, C3,…Cn).

As the scheduling involves combinatorial problem, it is important to ensure that a suitable methodology is selected to optimize the problem. In addition to the ability of finding optimal solution, the method also has to be capable to find the solution as quick as possible. Particle Swarm Optimization (PSO) possesses both criteria mentioned.

III. Particle Swarm Optimization

A. Standard PSO

PSO is categorized as swarm intelligence algorithm. It is a population based algorithm that is inspired by the social dynamics and emergent behavior that arises in socially organized colonies [12-14]. It exploits a population of particles to search for promising regions of the search space (swarm). While each particle randomly moves within the search space with a specified velocity, it stores data of the best position it ever encountered. This is known as personal best (Pbest) position.

Upon finishing each iteration, the Pbest position obtained by all individuals of the swarm is communicated to all of the particles in the population. The best value of Pbest will be selected as the global best position (Gbest) to represent the best position within the population. Each particle will search for best solution until it find stopping criteria. The movement of the particles towards the optimum is governed by equations similar to the following:

$$V^t_{i(d+1)} = W \times V^t_{id} + C_1 \times Rand \times (P_{best} - X^t_{id}) + C_2 \times Rand \times (G_{best} - X^t_{id}) \quad (3)$$

$$X^t_{i(d+1)} = X^t_{id} + V^t_{id} \quad (4)$$

Where W is inertial weight, c_1 and c_2 are constants (usually $c_1 = c_2 = 2$), r_1 and r_2 are uniform random numbers in [0,1], P_i is the best position vector of particle i^{th} until iteration t, P_{best} is the best position vector of all particles so far, x_{id} is the current position vector of particle i^{th}, and v_{id} is the current velocity parameter assigned for particle i^{th}.

For Eq. (2), the first part represents the inertial weight of the previous velocity. The second part corresponds to the cognition part, which represents the personal achievement of the particle. The third part is for the social part, which represents the cooperation among particles.

B. Binary PSO (BPSO)

In solving binary/ discrete problems, Kennedy and Eberhart [12] have deployed the PSO to search in binary spaces by applying a sigmoid transformation to the velocity component Eq. (5). It employs the concept of velocity as a probability that a bit (position) takes on one or zero. In the BPSO, Eq. (3) for updating the velocity remains unchanged, but Eq. (4) for updating the position is replaced by Eq. (6).

$$sigmoid(v^t_{id}) = \frac{1}{1+e^{-v^k_{id}}} \quad (5)$$

$$x^t_{id} = \begin{cases} 1, \leftarrow rand < sigmoid(v^k_{id}) \\ 0, otherwise \end{cases} \quad (6)$$

C. Utilizing BPSO to Solve Scheduling Problem

This section describes how BPSO is implemented to solve the simultaneous machines and AGVs scheduling problem. Among the most important matters of contention when designing any PSO algorithm lies on how to represent the solutions, of which particles bear the necessary information related to the problem-to-be solved. In order to map the relationship between the PSO particles and the problem domain, each particle will corresponds to a candidate solution of the scheduling problem.

In the proposed method each particle represents a feasible solution for task assignment using a vector of r elements, and each element is an integer value between 1 to n. Fig. 1 shows an illustrative example where each row represents the particles which correspond to a task assignment that assigns five tasks to three processors, and $Particle_{particle3,T4}$=P2 means that in particle 3 the Task 4 is assigned to Processor 2.

Differences between the current position of the k[th] particle, X^t_k, and the position with global best value P^t_k (or P^t_g) can be presented by an array of n element. Each element shows that whether the content of the resulting element in X^t_k is different from the desired one (best global value) or not.

If yes, the element gets its value from P^t_k (or P^t_g). For those elements that have the same content in X^t_k and P^t_k (or P^t_g), their corresponding jobs are listed based on specified rules and are assigned to machines successively, whenever a machine becomes free. For this research, the well-known longest processing time (LPT) was utilized as the main rule. It is due to the reason that in term of minimizing makespan, LPT proved to perform better than other conventional method [3, 4 and 13]. Apart from LPT, precedence and machine assignment constraints had also been considered during the scheduling process. Fig. 1 illustrates the operation's working principle.

Particle Number	T1	T2	T3	T4	T_n
Particle 1	P1	P2	P3	P4	P5
Particle 2	P3	P2	P2	P3	P5
Particle 3	P1	P1	P1	P2	P5
Particle 4	P2	P2	P3	P3	P1
Particle n	P_{n1}	P_{n2}	P_{n3}	P_{n4}	P_{n5}

	Job1	Job2	Job3	Job4
A (P^t_g)	1	2	3	4

	Job1	Job2	Job3	Job4
B (X^t_k)	1	3	3	5

Subtract	Job1	Job2	Job3	Job4
A - B	0	2	0	4

Fig. 1 Mapping representation of BPSO – FMS scheduling

Let the number of tasks be T and number of machines and AGVs available be M. The proposed BPSO algorithm for the task allocation process is summarized as the following:

Let M be the number of machines and AGVs.
Let T be the number of tasks.
Let P be the size of BPSO population.
Let *PSO*[*i*] be the position of the i^{th} particle in the entire population with T-dimensional vector, whose entries' values belong to the set {1,…, M}
Then *PSO*[*i*][*j*] be the processor number to which the j^{th} task in the i^{th} particle is assigned.
Let *fitness*[*i*] be the objective function of the i^{th} particle according to (1)
Let *V*[*i*] be the traveled distance (or velocity) of a i^{th} particle represented as an M-dimensional real-coded vector.
Let G_{best} be an index to global-best position.
Let $P_{best}[i]$ be the position of the local-best position.
Let P_{best}_*fitness*[*i*] be the local-best fitness for the best position visited by the i^{th} particle.

Initialization: For each particle *i* in the population:
i) For each task j, initialize *PSO*[*i*][*j*] randomly from the set {1,…,N}
ii) Initialize *V*[*i*] randomly
iii) Evaluate *fitness*[*i*]
iv) Initialize G_{best} with the index of the particle with the best fitness (lowest cost) among the population.
v) Initialize $P_{best}[i]$ with a copy of *PSO*[*i*] ≤ P

Optimization Process: Repeat until a number of generations, equal to twice the total number of tasks, are passed:
i) Find Gbest such that fitness[G_{best}] ≥ *fitness*[*i*] ≤ P
ii) For each particle i:
- $P_{best}[i]$ = *PSO*[*i*] if *fitness*[*i*] > P_{best}_*fitness* [*i*] ≤ P
- Update *V*[*i*] according to (3)
- Update *PSO*[*i*] according to (5) and (6)

iii) Evaluate *fitness*[*i*] ≤ P

IV. Experimental Setup

The FMS selected as the case in this work has the configuration as shown in Fig. 2. The case and data set is adopted from [11] was originated by [9]. In the case study, there are 10 job sets with each possessing four to eight different job sequences, dedicated machines and numbers were specified within the parenthesis is the processing time of a particular job (refer Table V). Based on the job sets and four different layouts, 82 problems are generated.

The problems are grouped into two categories. The first category contain problem sets which t_i/p_i ratios are greater than 0.25 while second category consists problems whose t_i/p_i ratios are lesser than 0.25. A code is used to represent the example problems. The digits succeeding EX indicate the job set and the layout respectively. Meanwhile, for second category, another digit is appended to the code. In this case, having a 0 or 1 as the last digit implies that the process times had been doubled or tripled, respectively. Furthermore, travel times are halved.

There are four machines consist of computer numerical machines (CNCs) and two AGVs for material delivery purpose. While the types and number of machines is fixed, the speed of the vehicles is constant at 40 m/min. Furthermore, loading and unloading times are constant at 0.5 min each.

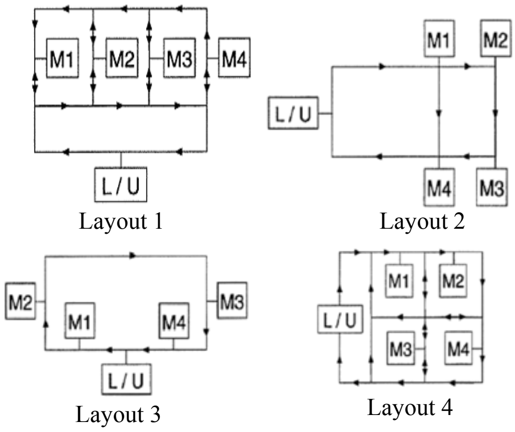

Fig. 2 Layouts for the case study

It is assumed that there is sufficient buffer space for input/output operations at each machine. Loading/ unloading equipments such as pallets are sufficiently allocated. Furthermore, the machine-to-machine distance and the distance between loading/ unloading machines are known.

The distance matrix of load/unload stations to machines and machine-to-machine distances for all layouts are shown in Appendix (Table IV). The load/unload (L/U) station acts as the distribution center for incoming raw materials and as the collection center for outgoing finished parts. All vehicles start from the L/U station initially though it does not need to return to L/U station in between delivery job.

V. Simulation Results And Discussion

Analysis had been conducted using MATLAB software. For the BPSO algorithm, V_{max} = 4, V_{min} = -4, c_1 = 2, c_2 = 2, swarm size is set to be 70 and the maximum of iterative generations I_{max} is set to be 400. For the reason that PSO and it variants inherit heuristic attributes, 10 runs had been conducted for every set of problem in the study. Average completion time of all the run had been taken as the completion time for the set.

Optimized task assignment of machine and AGV has been conducted. The research possesses offline scheduling behavior where complete set of task, number of machines and number of vehicles are established prior to the task assignment process. This is different to the online scheduling that is based on real-time scheduling where task assignment is mainly based on the delivery attributes. The outcomes discussed in this paper are compared to STW [9], UGA [10] and AGA [11]. While detailed result obtained based on the proposed methodology for the described FMS environment is given in Appendix (Table VI and Table VII), the contribution of BPSO in minimizing average makespan is depicted in Table I and Table II.

For t_i/p_i ratio >0.25 category, BPSO managed to improve the makespan for Layout 2 and Layout 3 as depicted in Table

ISBN: 978-988-18210-5-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

I. Meanwhile, for t_i/p_i ratio <0.25, although BPSO couldn't improved average makespan for any layout, results from other cases as shown in Table I and Table II proved that BPSO is able to provide optimal solution in minimizing scheduling makespan particularly for FMS.

Furthermore, comparison of actual makespan among all of the methodologies had also been analysed. Based on Table VI, out of 40 sets of problem, BPSO proved to be better in 15 cases when compared to the other methods while the results were on par for the other 12 problems. On the other hand, referring to the problem category with t_i/p_i ratio <0.25 as shown in Table VII, BPSO improved three results and equals the solution of 31 problems.

The comparison of improvements made is listed shown in Table III. The numbers represent total number of problem sets either categorized as I –Improved makespan, E – Equal to existing best makespan or Y – yet to be improved. From the results obtained, it is clear that BPSO successfully contributed to the minimization of makespan time.

It is found that BPSO is able to outperform other optimization methods for t_i/p_i ratio > 0.25. However, for t_i/p_i ratio < 0.25, BPSO only managed to improve solution of three cases. There is possibility that the algorithm might be trapped in local minima. This is corresponding to the searching mechanism of BPSO where upon having a P^t_g value; particles tend to move surrounding the position due to the social element characteristics. This will be one of the aspects for future improvement. In general, BPSO still bettered other methodologies noticeably.

In order to ensure the results obtained are statistically acceptable, analysis on makespan minimization characteristic over iteration had been conducted. To further explain about the minimization characteristic, two graphs are included as in Fig. 3 and Fig. 4. As it will be tedious to represent makespan minimization behavior for all of the 82 problem sets, we had normalized makespan minimization data into percentage value.

TABLE I COMPARISON OF AVERAGE MAKESPAN FOR T_i/P_i RATIO >0.25

	STW	UGA	AGA	BPSO
Layout 1	118.6	116.6	115.5	116.2
Layout 2	99.6	96.4	96.8	93.6
Layout 3	102.8	100.5	100.9	99.9
Layout 4	128.0	125.3	123.5	125.4

TABLE II COMPARISON FOR AVERAGE MAKESPAN FOR T_i/P_i RATIO <0.25

	STW	UGA	AGA	BPSO
Layout 1	167.6	164.9	167.4	166.2
Layout 2	164.6	162.1	163.8	162.2
Layout 3	165.4	163.1	164.5	164.7
Layout 4	194.2	187.4	188.8	189.3

TABLE III COMPARISON OF IMPROVEMENTS MADE BY BPSO

Method	t_i/p_i ratio > 0.25			t_i/p_i ratio < 0.25		
	I	E	Y	I	E	Y
BPSO	15	12	13	3	31	8
AGA	11	13	16	0	33	9
UGA	0	9	31	5	31	6
STW	0	6	34	0	21	21

This is to enable the calculation of mean average for all of the cases. The makespan value after first generation is used as the maximum value while the final accepted makespan value is used to represent 100% convergence. First quartile and third quartile values are used to represent the convergence variation between problem cases.

Referring to Fig. 3 and Fig. 4, both graphs illustrate BPSO convergence rate (t_i/p_i > 0.25) and (t_i/p_i < 0.25) respectively. It is shown that convergence variation for t_i/p_i ratio < 0.25 is smaller than t_i/p_i ratio > 0.25. According to [15], the variation is a normal outcome for any BPSO utilizing static topology. Since BPSO is more suitable for large search space, it becomes more stable for t_i/p_i < 0.25 category. However, both graphs also indicate that on average, 100% convergence could be achieved after 200 iterations for most of the cases.

VI. CONCLUSION

Based on the analysis conducted, it is found that BPSO managed to provide a better optimization solution particularly for simultaneous scheduling of machines and automated vehicles in production environment. For future study, more consideration would be given on establishing unique BPSO optimization method. Other BPSO variations would be considered not only to shorten the tasks completion time but also to shorten calculation time. Another limitation of the work is that it deals with single objective problem. Future work would consider multiple objectives so as to reflect actual industrial applications.

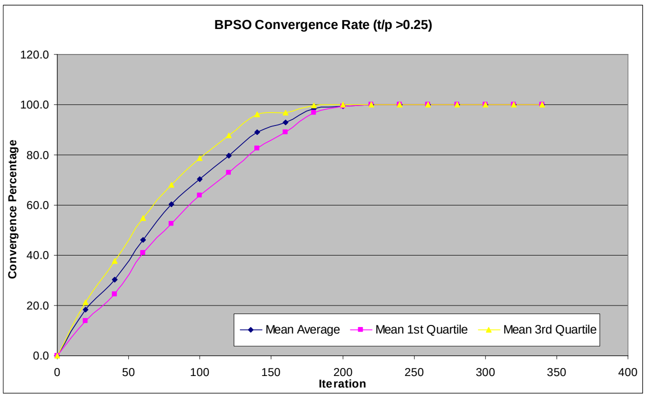

Fig. 3 BPSO Convergence Rate (t/p > 0.25)

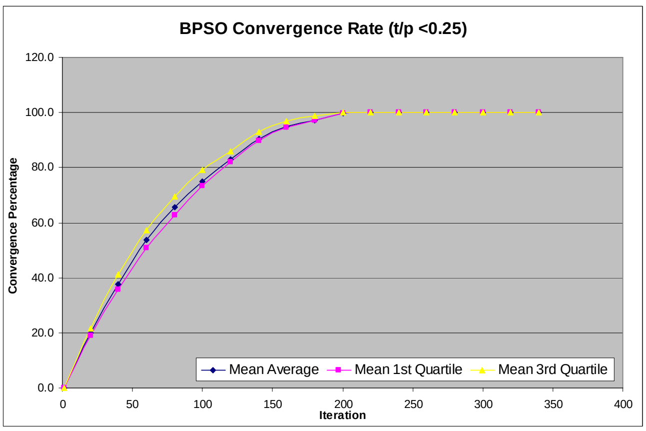

Fig. 4 BPSO Convergence Rate (t/p < 0.25)

ISBN: 978-988-18210-5-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

APPENDIX

TABLE IV MACHINE-TO-MACHINE DISTANCE CHART

	Layout 1					Layout 2					Layout 3					Layout 4				
From - To	LU	M1	M2	M3	M4	LU	M1	M2	M3	M4	LU	M1	M2	M3	M4	LU	M1	M2	M3	M4
L/U	0	6	8	10	12	0	4	6	8	6	0	2	4	10	12	0	4	8	10	14
M1	12	0	6	8	10	6	0	2	4	2	12	0	2	8	10	18	0	4	6	10
M2	10	6	0	6	8	8	12	0	2	4	10	12	0	6	8	20	14	0	8	6
M3	8	8	6	0	6	6	10	12	0	2	4	6	8	0	2	12	8	6	0	6
M4	6	10	8	6	0	4	8	10	12	0	2	4	6	12	0	14	14	12	6	0

TABLE V PROBLEM SETS (JOB SEQUENCE WITH MACHINE PROCESSING TIME DETAIL)

Job Set 1 Job 1: M1(8); M2(16); M4(12) Job 2: M1(20); M3(10); M2(18) Job 3: M3(12); M4(8); M1(15) Job 4: M4(14); M2(18) Job 5: M3(10); M1(15) Job Set 2 Job 1: M1(10); M4(18) Job 2: M2(10); M4(18) Job 3: M1(10); M3(20); Job 4: M2(10); M3(15); M4(12) Job 5: M1(10); M2(15); M4(12) Job 6: M1(10); M2(15); M3(12) Job Set 3 Job 1: M1(16); M3(15) Job 2: M2(18); M4(15) Job 3: M1(20); M2(10) Job 4: M3(15); M4(10) Job 5: M1(8); M2(10); M3(15); 4(17) Job 6: M2(10); M3(15); M4(8); M1(15)	Job Set 4 Job 1: M4(11); M1(10); M2(7) Job 2: M3(12); M2(10); M4(8) Job3:M2(7);M3(10); M1(9); M3(8) Job4:M2(7);M4(8); M1(12); M2(6) Job 5: M1(9); M2(7); M4(8); M2(10); M3(8) Job Set 5 Job 1: M1(6); M2(12); M4(9) Job 2: M1(18); M3(6); M2(15) Job 3: M3(9); M4(3); M1(12) Job 4: M4(6); M2(15) Job 5: M3(3); M1(9) Job Set 6 Job 1: M1(9); M2(11); M4(7) Job 2: M1(19); M2(20); M4(13) Job 3: M2(14); M3(20); M4(9) Job 4: M2(14); M3(20); M4(9) Job 5: M1(11); M3(16); M4(8) Job 6: M1(1O); M3(12); M4(10)	Job Set 7 Job 1: M1(6); M4(6) Job 2: M2(11); M4(9) Job 3: M2(9); M4(7) Job 4: M3(16); M4(7) Job 5: M1(9); M3(18) Job 6: M2(13); M3(19); M4(6) Job 7: M1(10); M2(9); M3(13) Job 8: M1(11); M2(9); M4(8) Job Set 8 Job 1: M2(12); M3(21); M4(11) Job 2: M2(12); M3(21); M4(11) Job 3: M2(12); M3(21); M4(11) Job 4: M2(12); M3(21); M4(11) Job 5: M1(10); M2(14); M3(18); M4(9) Job 6: M1(10); M2(14); M3(18); M4(9)	Job Set 9 Job 1: M3(9); M1(12); M2(9); M4(6) Job 2: M3(16); M2(11); M4(9) Job 3: M1(21); M2(18); M4(7) Job 4: M2(20); M3(22); M4(11) Job 5: M3(14); M1(16); M2(13); M4(9) Job Set 10 Job 1: M1(11); M3(19); M2(16); M4(13) Job 2: M2(21); M3(16); M4(14) Job 3: M3(8); M2(10); M1(14); M4(9) Job 4: M2(13); M3(20); M4(10) Job 5: M1(9); M3(16); M4(18) Job 6: M2(19); M1(21); M3(11); M4(15)

TABLE VI RESULT COMPARISON OF JOB MAKESPAN FOR T_I/P_I RATIO >0.25

Problem	t_i/p_i ratio	STW	UGA	AGA	BPSO
EX11	0.59	96	96	96	96
EX21	0.61	105	104	102	101
EX31	0.59	105	105	99	105
EX41	0.91	118	116	112	118
EX51	0.85	89	87	87	87
EX61	0.78	120	121	118	120
EX71	0.78	119	118	115	125
EX81	0.58	161	152	161	142
EX91	0.61	120	117	118	115
EX101	0.55	153	150	147	153
EX12	0.47	82	82	82	82
EX22	0.49	80	76	76	76
EX32	0.47	88	85	85	80
EX42	0.73	93	88	88	88
EX52	0.68	69	69	69	72
EX62	0.54	100	98	98	90
EX72	0.62	90	85	79	75
EX82	0.46	151	142	151	137
EX92	0.49	104	102	104	100
EX102	0.44	139	137	136	136
EX13	0.52	84	84	84	84
EX23	0.54	86	86	86	86
EX33	0.51	86	86	86	84
EX43	0.8	95	91	89	91
EX53	0.74	76	75	74	76
EX63	0.54	104	104	104	101
EX73	0.68	91	88	86	94
EX83	0.5	153	143	153	141
EX93	0.53	110	105	106	102
EX103	0.49	143	143	141	140
EX14	0.74	108	103	103	103
EX24	0.77	116	113	108	113
EX34	0.74	116	113	111	119
EX44	1.14	126	126	126	126
EX54	1.06	99	97	96	96
EX64	0.78	120	123	120	120
EX74	0.97	136	128	127	126
EX84	0.72	163	163	163	158
EX94	0.76	125	123	122	122
EX104	0.69	171	164	159	171

ISBN: 978-988-18210-5-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

TABLE VII RESULT COMPARISON OF JOB MAKESPAN FOR T_i/P_i RATIO <0.25

Problem	t_i/p_i ratio	STW	UGA	AGA	BPSO
EX110	0.15	126	126	126	126
EX210	0.15	148	148	148	136
EX310	0.15	150	148	150	150
EX410	0.15	121	119	119	119
EX510	0.21	102	102	102	102
EX610	0.16	186	186	186	186
EX710	0.19	137	137	137	137
EX810	0.14	292	271	292	292
EX910	0.15	176	176	176	176
EX1010	0.14	238	236	238	238
EX120	0.12	123	123	123	123
EX220	0.12	143	143	143	143
EX320	0.12	148	145	145	132
EX420	0.12	116	114	114	114
EX520	0.17	100	100	100	100
EX620	0.12	183	181	181	181
EX720	0.15	136	136	136	136
EX820	0.11	287	268	287	287
EX920	0.12	174	173	173	170
EX1020	0.11	236	238	236	236
EX130	0.13	122	122	122	122
EX230	0.13	146	146	146	146
EX330	0.13	149	146	146	146
EX430	0.13	116	114	114	114
EX530	0.18	99	99	99	99
EX630	0.14	184	182	182	182
EX730	0.17	137	137	137	137
EX830	0.13	288	270	288	288
EX930	0.13	176	174	174	176
EX1030	0.12	237	241	237	237
EX140	0.18	124	124	124	124
EX241	0.13	217	217	217	217
EX340	0.18	151	151	151	151
EX341	0.12	222	221	221	221
EX441	0.19	179	172	172	179
EX541	0.18	154	148	148	148
EX640	0.19	185	184	184	184
EX740	0.24	138	137	137	137
EX741	0.16	203	203	203	203
EX840	0.18	293	273	293	293
EX940	0.19	177	175	175	175
EX1040	0.17	240	244	240	240

REFERENCES

[1] E. K. Bish , F. Y. Chen , Y. T. Leong, B. L. Nelson, J. W. C. Ng, and D. Simchi-Levi, "Dispatching vehicles in a mega container terminal *OR Spectrum,* vol. 27, Number 4 pp. 491 - 506 2005.

[2] M. Grunow , H. Günther, and M. Lehmann, "Dispatching multi-load AGVs in highly automated seaport container terminals, *OR Spectrum,* vol. 26, Number 2, 211-235 2004.

[3] A. I. Corréa, A. Langevin, and L.-M. Rousseau, "Scheduling and routing of automated guided vehicles: A hybrid approach, *Computers & Operations Research,* 2005.

[4] T. Le-Anh and M. B. M. De Koster, "On-line dispatching rules for vehicle-based internal transport systems, *International Journal of Production Research,* vol. 43, Number 8 / April 15, 2005 pp. 1711 - 1728 2005.

[5] G. Desaulniers, A. Langevin, D. Riopel, and B. Villeneuve, "Dispatching and Conflict-Free Routing of Automated Guided Vehicles: An Exact Approach, *International Journal of Flexible Manufacturing Systems* vol. 15, Number 4, pp. 309 - 331, 2003.

[6] Iris F.A. Vis, René de Koster, Kees Jan Roodbergen, Leon W.P. Peeters (2001). Determination of the number of AGVs required at a semi-automated container terminal. Journal of the Operational Research Society. Vol 52 pp 409-417.

[7] NaiQi Wu and MengChu Zhou (2007).Shortest Routing of Bidirectional Automated Guided Vehicles Avoiding Deadlock and Blocking. IEEE/ASME Transactions on Mechatronics. Vol. 12 Issue 1 pp 63-72

[8] Mariagrazia Dotoli and Maria Pia Fanti (2007). Deadlock Detection and Avoidance Strategies for Automated Storage and Retrieval Systems. IEEE Transactions on Systems, Man and Cybernetics – Part C: Applications and Reviews, Vol 37, No 4, July 2007.

[9] Ümit Bilge and Gündüz Ulusoy (1995). A Time Window Approach to Simultaneous Scheduling of Machines and Material Handling System in an FMS. Journal of Operations Research. Vol. 43, No 6, pp. 1058-1070

[10] Ulusoy, Gündüz, Sivrikaya-Serifoglu, Funda and Bilge, Ümit (1997). A genetic algorithm approach to the simultaneous scheduling of machines and automated guided vehicles. Journal of Computers Operational Research. Vol 24, No 4, pp 335-351. Elsevier Ltd. 1997

[11] Tamer F. Abdelmaguid, Ashraf O Nassef, Badawia A. Kamal and Mohamed F. Hassan (2004). A Hybrid GA/ Heuristic approach to the Simultaneous Scheduling of Machines and Automated Guided Vehicle. International Research of Production Research. Vol 42, No 2, 267-281. Taylor and Francis Group.

[12] Kennedy J, Eberhart RC. A discrete binary version of the particle swarm algorithm. In: Proceedings of the World Multi-Conference on Systemic, Cybernetics and Informatics. NJ: Piscatawary; 1997.

[13] J.Jerald, P.Asokan, G.Prabaharan and R.Saravanan (2005). Scheduling optimisation of flexible manufacturing systems using particle swarm optimisation algorithm. The International Journal of Advanced Manufacturing Technology, Volume 25, Numbers 9-10. pp 964-971. Springer London.

[14] Ayed Salman, Imtiaz Ahmad and Sabah Al-Madani (2002). Particle swarm optimization for task assignment problem. Microprocessors and Microsystems. Vol 26, Issue 8. P.p 363-371. Elsevier Science B.V.

[15] Riccardo Poli, James Kennedy and Tim Blackwell (2007). Particle Swarm optimization - An overview. Swarm Intelligence Journal. Volume 1, Number 1. Pages 33-57. Springer New York

ISBN: 978-988-18210-5-8
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)