RESEARCH ARTICLE

Multi-objective AGV scheduling in an FMS using a hybrid of genetic algorithm and particle swarm optimization

Maryam Mousavi, Hwa Jen Yap*, Siti Nurmaya Musa, Farzad Tahriri, Siti Zawiah Md Dawal

Centre for Product Design and Manufacturing, Department of Mechanical Engineering, Faculty of Engineering, University of Malaya, Kuala Lumpur, Malaysia

* hjyap737@um.edu.my

Citation: Mousavi M, Yap HJ, Musa SN, Tahriri F, Md Dawal SZ (2017) Multi-objective AGV scheduling in an FMS using a hybrid of genetic algorithm and particle swarm optimization. PLoS ONE 12(3): e0169817. doi:10.1371/journal.pone.0169817

Editor: Wen-Bo Du, Beihang University, CHINA

Received: November 19, 2015

Accepted: December 22, 2016

Published: March 6, 2017

Data Availability Statement: All relevant data are within the paper and its Supporting Information files.

Funding: The authors would like to acknowledge University of Malaya's financial support of this project under UMRG Top Down Programme (Grant No. RP027-14AET). The authors would also like to acknowledge the Ministry of Higher Education of Malaysia for the facility support under High Impact Research Grant UM.C/HIR/MOHE/ENG/35 (D000035-16001).

Abstract

Flexible manufacturing system (FMS) enhances the firm's flexibility and responsiveness to the ever-changing customer demand by providing a fast product diversification capability. Performance of an FMS is highly dependent upon the accuracy of scheduling policy for the components of the system, such as automated guided vehicles (AGVs). An AGV as a mobile robot provides remarkable industrial capabilities for material and goods transportation within a manufacturing facility or a warehouse. Allocating AGVs to tasks, while considering the cost and time of operations, defines the AGV scheduling process. Multi-objective scheduling of AGVs, unlike single objective practices, is a complex and combinatorial process. In the main draw of the research, a mathematical model was developed and integrated with evolutionary algorithms (genetic algorithm (GA), particle swarm optimization (PSO), and hybrid GA-PSO) to optimize the task scheduling of AGVs with the objectives of minimizing makespan and number of AGVs while considering the AGVs' battery charge. Assessment of the numerical examples' scheduling before and after the optimization proved the applicability of all the three algorithms in decreasing the makespan and AGV numbers. The hybrid GA-PSO produced the optimum result and outperformed the other two algorithms, in which the mean of AGVs operation efficiency was found to be 69.4, 74, and 79.8 percent in PSO, GA, and hybrid GA-PSO, respectively. Evaluation and validation of the model was performed by simulation via Flexsim software.

Introduction

General

In today's competitive market, customer satisfaction plays a key role in companies' market share. Therefore, organizations have shifted their strategy from manufacturing large quantities of a single product to a range of products, and improving the quality and on-time delivery. To meet these challenges, organizations should have a flexible manufacturing platform. In automated manufacturing environments, FMS provides wide flexibility and concurrent production of a wide variety of parts in small capacities. It comprises material handling

Competing interests: The authors have declared that no competing interests exist.

devices like robots and AGVs, automated storage and retrieval systems (AS/RS) and workstations. AGVs are extensively used in FMSs because of their flexible structure and high compatibility [1, 2].

AGVs are driverless mobile vehicles that are computer-controlled (usually battery operated) and equipped with different guidance systems (optical, magnetic, laser, etc.) for automated functionality. They are categorized into two groups of load towing and load carrying (forked, mandrel, unit load deck, etc.) [3]. AGVs are well-suited for long-distance horizontal movement of materials from/to multiple destinations. They are also apt for repetitive/predictable material transportations and/or dangerous tasks [4, 5]. AGVs control system can be incorporated into the computer control of the production and storage equipment; thus, all the shop floor operations would be controlled using a computer system.

FMS performance increases by better coordination and scheduling of its components like AGV [6–8]. The term 'scheduling' refers to the process of allocating AGVs to tasks, taking into account the cost and required time for the operations to be done [9]. Efficient scheduling therefore would increase the productivity and reduce the delivery cost whilst the entire fleet is optimally utilized [7]. Although the AGV scheduling context has been studied before, given the diversity in objectives, limitations and considerations in scheduling problems, it is still an open area of research to improve it for real environment results. Improvement in the performance of an FMS can be expected by efficiently utilizing its resources and properly integrating and synchronizing their scheduling [9–11]. Literature has shown tendency toward multi-task scheduling of AGV systems and FMSs, in which the makespan minimization criterion is accompanied with several other criteria to entertain an actual-practice scheduling [12–14]. Heuristic techniques and evolutionary algorithms (EA) are the common optimization methods used to solve a multi-task scheduling problem [3, 15–17]. Some of the distinct researches in scheduling context that can assist new researchers in finding a pertinent literature to their specific objectives and methodologies are [18–21].

In the majority of the earlier works, makespan minimization was the main objective in scheduling practices as it keeps the resources utilization rate at a balanced level and ensures proper utilization of expensive FMSs [10, 22, 23]. However, those studies have discounted the importance of equal utilization of all the resources. Allocating a large number of AGVs may lead to a shorter makespan but it escalates the idle time of AGVs and costs [24]. AGVs are such expensive devices that determining the type and the appropriate number of them in an FMS can positively influence the profitability of the business [14, 17, 25–27]. Another issue in AGV scheduling is the charge of an AGV's battery, where many studies do not consider the AGV's battery charge and that leads to unrealistic scheduling models. Battery management in an AGV System (AGVS) is crucial as it can reduce the costs and increase the efficiency of the system [28, 29]. To address the above concerns, this research aims to schedule AGVs in an FMS environment by developing a multi-objective mathematical model that minimizes the makespan and number of AGVs while considering the AGV's battery charge. The model will be optimized using three evolutionary algorithms (genetic algorithm (GA), particle swarm optimization (PSO), and hybrid GA-PSO) and validated through simulation in Flexsim software.

Problem descriptions and assumptions

The type and operation of FMS vary according to the different configurations being used. Therefore, the system configuration has to be established precisely prior to the scheduling of AGVs. The system configuration, assumptions and objective criteria in this study are presented in the following sections.

Model derivation

This section explains the mathematical model development for AGV scheduling using the three criteria selected based on the literature review. The three criteria are categorized into two main objectives: (1) minimizing the makespan and (2) minimizing the number of AGVs while considering the AGV's battery charge. Each sub-section explains the mathematical definitions used to develop the model. However, prior to the model development it is necessary to define the conditions and limitations considered in the model framework. Thus, the following conditions were defined for the model development:

- All AGVs have unit-load capacity.
- AGVs and machines operate continuously without breakdown.
- There are no traffic problems, collision, deadlock or conflict.
- AGV loading and unloading times are fixed and considered in travel times.
- AGVs can always park at their pick-up/drop-off (P/D) locations.
- The velocity of the vehicles is constant and vehicles move forward only.
- Loading/unloading (L/U) equipment such as pallets are sufficiently allocated as well as input and output buffer for machines.
- The machine-to-machine distance and L/U point-to-machine distances are known.
- Each machine operates only one product at a time.
- The setup times are included in the time of production.
- The AGVs are stored in the home until dispatching commands are allocated.

Notations

n	Total number of jobs
j, j'	Indexes of jobs, genes', and dimensions' code, $j, j' = 1, 2, \ldots, n$
m_j	Total number of operations for each job j
i, i'	Indexes of operations, $i, i' = 1, 2, \ldots, m_{j,j'}$
θ	Total number of operations for all of jobs
z	Number of AGVs
a, a'	Index of AGVs, $a, a' = 1, \ldots, z$
y	Index of new AGV
J_j	Job number j
O_{ji}	Operation i from job j
M_{ji}	Assigned machine for O_{ji}
p_{ji}	Processing time of O_{ji}
p^s_{ji}	Start time of processing O_{ji}
p^e_{ji}	End time of processing O_{ji}
H	Load/unloading point (Home)
A^a	AGV number a
T_{ji}	Related task to O_{ji} (Moving from $M_{j(i-1)}$ to M_{ji} or H to M_{ji})
T^a_{ji}	Assigned A^a to do task T_{ji}
T^a	A collection of operations that have done by A^a
T	A collection of T^a
MS	Makespan

PS	Population size for GA
r	Index of chromosomes, $r = 1, \ldots, PS$
e	Index of genes, e = 1, . . ., θ
C_r	Chromosome
G_e	Gene
CR	Crossover rate
Pm	Mutation rate
$G_{\max}$	Maximum gene code
$Iter_{\max}$	The maximum iterations
$Iter$	The current iteration number
t	Iteration number
S^t	Swarm size at iteration *(t)*
α	Index of particles, $\alpha = 1, \ldots, S^t$
PR_α	Particle
d	Dimension, $d = 1, \ldots, \theta$
ω	Inertia factor
$\omega_{\max}$	Maximum inertia factor
$\omega_{\min}$	Minimum inertia factor
$v^t_{\alpha d}$	The velocity of α^{th} particle on d^{th} dimension at iteration *(t)*
$v^{t+1}_{\alpha d}$	The velocity of α^{th} particle on d^{th} dimension at iteration *(t+1)*
V^t_α	The velocity of α^{th} particle in the swarm at iteration *(t)*
$q^t_{\alpha d}$	The position of α^{th} particle on d^{th} dimension at iteration *(t)*
$q^{t+1}_{\alpha d}$	The position of α^{th} particle on d^{th} dimension at iteration *(t+1)*
Q^t_α	The position of α^{th} particle in the swarm at iteration *(t)*
$B^t_{\alpha d}$	The best position of α^{th} particle on d^{th} dimension found so far
G^t_d	The global best position of the swarm on d^{th} dimension found so far
φ_1 and φ_2	Uniformly distributed random numbers in the interval [0, 1]
C_1	Self-confidence
C_2	Swarm confidence
NA	Number of AGVs to do all the operations
CTO_{ji}	The time that operation O_{ji} completes
$CChA^a$	Current battery charge of A^a
$ChHT^a_{ji}$	Charge that A^a needed for doing the task T_{ji} and back home
ChT^a_{ji}	The battery charge that A^a consumes for doing T_{ji}
ChA^a	The total battery charge that A^a consumes for all of its operations
CA^a	Current position of A^a,(Can be H, M_{ji}, $M_{j'i'}$, and $M_{j(i-1)}$)
tCA^a	Time of current position of A^a
$tT^a_{ji}H$	Time that A^a arrives home after doing T_{ji}
PT^a_{ji}	Pick up point of A^a doing T_{ji}, (P represents pick up point and can be H, M_{ji}, $M_{j'i'}$, and $M_{j(i-1)}$)
tPT^a_{ji}	Pick up time of A^a doing T_{ji}
rPT^a_{ji}	The time that A^a reaches pick up place of T_{ji}
DT^a_{ji}	Drop off point of A^a doing T_{ji},(D represents drop off point and can be H, M_{ji}, $M_{j'i'}$, and $M_{j(i-1)}$)
tDT^a_{ji}	Drop off time of A^a doing T_{ji}
rDT^a_{ji}	The time that A^a reaches drop off place of T_{ji}
μ	A large positive number

$tCPT_{ji}^{a}$	The travel time of A^a from its current point to reach the start point of T_{ji}
γ	A coefficient for transforming energy consumption to time
UT_{ji}^{a}	Unloaded time of A^a doing T_{ji}
UtA^a	Total unloaded time of A^a
ItA^a	Total idle time of A^a
WT_{ji}^{a}	Waiting time of A^a doing T_{ji}
WtA^a	Total waiting time of A^a
LT_{ji}^{a}	loaded time of A^a doing T_{ji}
LtA^a	Total loaded time of A^a
RT_{ji}^{a}	Running time (loaded + unloaded) of A^a doing T_{ji}
RtA^a	Total running time (loaded + unloaded) of A^a
$\overline{TuCh}$	The time that AGV is being charged
BU	Battery usage percentage of A^a
AE	Efficiency of AGV's operation (%)
λ	A coefficient for determining when a new AGV should be added
L	Number of objectives
β	Index of δ, $\beta = 1, \ldots, L$
δ	The β^{th} weight of the β^{th} objective function
ψ	A ratio to make balance among objectives with different ranges of value
$f(x)$	Fitness function

Minimizing the makespan. This step involves calculating makespan (MS) which is the time required for all operations to be completed. Makespan can be expressed by

$$MS = max\left\{\left(tDT_{ji}^{a} + p_{ji}\right)\right\} \tag{1}$$

$$tDT_{ji}^{a} = tCA^{a} + UT_{ji}^{a} + WT_{ji}^{a} + LT_{ji}^{a} \tag{2}$$

Subject to:

$$CTO_{ji} \geq p_{ji}^{s} \quad \forall i = 1 \tag{3}$$

$$tPT_{ji}^{a} \geq 0 \quad \forall\, T_{ji}^{a} \in T^{a} \tag{4}$$

$$p_{ji}^{s} \;\geq\; tPT_{j(i+1)}^{a} - p_{ji} \quad \forall j, i \tag{5}$$

$$p_{ji}^{s} - p_{j(i-1)}^{s} \geq p_{ji} + LT_{ji}^{a} \quad \forall j,\; i = 2, \ldots,\; m_{j} \tag{6}$$

$$\begin{aligned}&(p_{ji}^{s} - p_{j'i'}^{s} - p_{j'i'} + \mu|M_{ji} - M_{j'i'}| \geq 0) \;\vee \\ &\qquad \left(p_{j'i'}^{s} - p_{ji}^{s} - p_{ji} + \mu|M_{ji} - M_{j'i'}| \geq 0\right) \quad \forall (j, i, j', i')\end{aligned} \tag{7}$$

$$\begin{aligned}&(tPT_{jm_j}^{a} - tPT_{j'i'}^{a} - LT_{j'i'}^{a} + \mu|T_{ji}^{a} - T_{j'i'}^{a}| \geq 0) \;\vee \\ &\qquad \left(tPT_{j'i'}^{a} - tPT_{jm_j}^{a} - LT_{ji}^{a} + \mu|T_{ji}^{a} - T_{j'i'}^{a}| \geq 0\right) \forall \left(j, m_j, j', i'\right)\end{aligned} \tag{8}$$

Constraint number 3 ensures the feasibility of completion time of the first operation of each job. Constraints number 4 and 5 ensure the feasibility of pick up time of operations.

Inequality number 6 describes the operations precedency constraint. Inequalities number 7 and 8 represent the operation and the AGV un-overlapping constraints respectively.

Minimizing the number of AGVs. This step involves calculating the number of AGVs, which is denoted by *NA* by considering the AGVs battery charge sufficiency. Number of AGV can be expressed by

$$NA = \max\{a\} | \; T^a \in T \tag{9}$$

Subject to

$$A^a \text{ is assigned to } T_{ji} \text{ (to create } T^a_{ji}) \; if \begin{cases} ChA^a \geq ChHT^a_{ji} \\ \quad \wedge \\ \begin{cases} tCPT^a_{ji} < tCPT^{a'}_{ji} \\ \quad \vee \\ tCPT^a_{ji} = tCPT^{a'}_{ji} \quad \wedge \quad a < a' \end{cases} \end{cases} \tag{10}$$

$$ChHT^a_{ji} = \gamma(UT^a_{ji} + LT^a_{ji} + (tT^a_{ji}H - tDT^a_{ji})) \tag{11}$$

$$A^y|_{y \text{ is a new AGV}} \text{ is assigned to } T_{ji} \text{ (to create } T^y_{ji}) \\ if \left\{ tCPT^y_{ji} + RT^y_{ji} \leq \lambda\left(tCPT^a_{ji} + RT^a_{ji}\right)\right\} \tag{12}$$

where Eq 10 makes sure the assigned AGV has enough battery charge to do the job and return home, while it chooses the AGV which takes less time to reach the point. Eq 11 calculates the charge that AGV needs to do the job and return home. As battery-run-time of an AGV and battery-charging-time can be defined depending on the type of batteries used, charging methods, charge rate, application, manufacturer, and assignments the vehicles perform, γ has been defined to adopt to any kind of battery, charging method, etc. The automatic and opportunity battery charging considered here, which on average, an AGV charges for 10–12 minutes every hour [30, 31]. Eq 12 determines the suitable time for adding a new AGV.

Multi-objective evaluation. Decision makers refer to choosing a solution out of all the efficient solutions as a posteriori approach. Pareto is one well-known pioneer in multi-objective optimization problems. In this method, Pareto-optimal set is a group of best trade-off schedules, and Pareto-front refers to a set of Pareto solutions [32]. Overall fitness function formulation for two objectives is described by

$$f(x) = \delta_1 f_1(x) + \psi(1 - \delta_1) f_2(x) \tag{13}$$

Where δ_1 is the weight of first objective function and ψ is a ratio to make balance among objectives with different ranges of value [33–35], which is defined by

$$\psi = \frac{max\, f_1(x)}{max\, f_2(x)} \tag{14}$$

AGV specifications

This step involves calculating specifications of AGV number *a* including its total running time denoted by RtA^a (loaded (LtA^a) + unloaded time (UtA^a)), its waiting time (WtA^a), its idle time

(ItA^a), its consumed battery charge (ChA^a), its battery usage percentage (BU), and its efficiency (AE) by Eqs 15 to 27.

$$UtA^a = \sum_{\substack{j,i \\ T^a_{ji} \in T^a}} UT^a_{ji} \tag{15}$$

$$UT^a_{ji} = tPT^a_{ji} - tCA^a = \begin{cases} tHT^a_{ji} - tM_{j(i-1)} & if \quad CA^a = M_{j(i-1)} \quad \wedge \quad i = 1 \\ tHT^a_{ji} - tM_{ji} & if \quad CA^a = M_{ji} \quad \wedge \quad i = 1 \\ tHT^a_{ji} - tM_{j'i'} & if \quad CA^a = M_{j'i'} \quad \wedge \quad i = 1 \\ 0 & if \quad \begin{cases} CA^a = H \quad \wedge \quad i = 1 \\ \vee \\ CA^a = M_{j(i-1)} \quad \wedge \quad i \neq 1 \end{cases} \\ tM_{j(i-1)}T^a_{ji} - tH & if \quad CA^a = H \quad \wedge \quad i \neq 1 \\ tM_{j(i-1)}T^a_{ji} - tM_{ji} & if \quad CA^a = M_{ji} \quad \wedge \quad i \neq 1 \\ tM_{j(i-1)}T^a_{ji} - tM_{j'i'} & if \quad CA^a = M_{j'i'} \quad \wedge \quad i \neq 1 \end{cases} \tag{16}$$

$$LtA^a = \sum_{\substack{j,i \\ T^a_{ji} \in T^a}} LT^a_{ji} \tag{17}$$

$$LT^a_{ji} = tDT^a_{ji} - tPT^a_{ji} = \begin{cases} tM_{ji}T^a_{ji} - tHT^a_{ji} & if \quad PT^a_{ji} = H \quad \wedge \quad i = 1 \\ tM_{ji}T^a_{ji} - tM_{j(i-1)}T^a_{ji} & if \quad PT^a_{ji} = M_{j(i-1)} \quad \wedge \quad i \neq 1 \end{cases} \tag{18}$$

$$WtA^a = \sum_{\substack{j,i \\ T^a_{ji} \in T^a}} WT^a_{ji} \tag{19}$$

$$WT^a_{ji} = \begin{cases} p^e_{j(i-1)} - rPT^a_{ji} & if \quad rPT^a_{ji} < p^e_{j(i-1)} \quad \wedge \quad rDT^a_{ji} \geq p^e_{j'i'} \quad \wedge \quad i \neq 1 \\ \left(p^e_{j(i-1)} - rPT^a_{ji}\right) + \left(p^e_{j'i'} - rDT^a_{ji}\right) & if \quad rPT^a_{ji} < p^e_{j(i-1)} \quad \wedge \quad rDT^a_{ji} < p^e_{j'i'} \quad \wedge \quad i \neq 1 \\ p^e_{j'i'} - rDT^a_{ji} & if \quad \begin{cases} rDT^a_{ji} < p^e_{j'i'} \quad \wedge \quad i = 1 \\ \vee \\ rDT^a_{ji} < p^e_{j'i'} \quad \wedge \quad rPT^a_{ji} \geq p^e_{j(i-1)} \quad \wedge \quad i \neq 1 \end{cases} \\ 0 & if \quad \begin{cases} rDT^a_{ji} \geq p^e_{j'i'} \quad \wedge \quad i = 1 \\ \vee \\ rDT^a_{ji} \geq p^e_{j'i'} \quad \wedge \quad rPT^a_{ji} \geq p^e_{j(i-1)} \quad \wedge \quad i \neq 1 \end{cases} \end{cases} \tag{20}$$

Subject to $\quad M_{ji} \triangleq M_{j'i'}$

$$ItA^a = MS - RtA^a - \overline{TuCh} \tag{21}$$

$$RtA^a = \sum_{\substack{j,i \\ T^a_{ji} \in T^a}} RT^a_{ji} \tag{22}$$

$$RT^a_{ji} = LT^a_{ji} + UT^a_{ji} \tag{23}$$

$$ChA^a = \sum_{\substack{j,i \\ T^a_{ji} \in T^a}} ChT^a_{ji} \tag{24}$$

$$ChT^a_{ji} = \gamma(LT^a_{ji} + UT^a_{ji}) \tag{25}$$

$$AE = \frac{ChA^a}{\gamma ItA^a} \times 100 \tag{26}$$

$$BU = \frac{ChA^a}{\gamma\left(MS - \overline{TuCh}\right)} \times 100 \tag{27}$$

Proposed algorithms

Three different evolutionary algorithms (EA) have been developed to optimize the mathematical AGV scheduling model. The three algorithms (GA, PSO, and hybrid GA-PSO) are later evaluated in terms of their strength and suitability for the scheduling problem.

Genetic algorithm

GA is a search algorithm based on the mechanics of the natural selection process. The major steps of GA algorithm development according to the study objective are described in this section. However, readers for a thorough review of the GA are referred to publications of [36–40].

Step 1. Initializing parameters. It involves setting the parameters of the GA and creating the first generation of chromosomes based on the notations section. The general schematic for reading data for the problem is presented in Table 1. The first column shows a chromosome *(C_r)* and the second one shows the genes (G_e) of the chromosome. The encoding of each gene is presented in the third column, which will be discussed later.

Step 2. Initializing population. A set of chromosomes is needed to create a population. For constructing a chromosome, it is necessary to define a proper genetic representation (encoding) due to its significant effects on all the subsequent steps of the GA.

Chromosome representation and encoding. As it is shown in Table 1, each chromosome is formed by genes. The order of genes represents the priority of operations, which decreases from left to right; and the genes' code defines operations related to each job. Gene's code are the same as their job number so that all the genes related to J_1 operations have the code '*1*' and subsequently the code '*2*' is given to all the genes related to the operations of J_2, and so on. As the operations of each job are expected to be performed sequentially, the repetition of genes' code represents the corresponding operation number of the job as clearly described in the following example.

Table 1. General schematic for reading data.

GA			PSO						
Chromosome *(C_r)*	Gene number *(G_e)*	Gene code *(j)*	Particle *(PR_α)*	Dimension Number *(d)*	Dimension code *(j)*	Job *(J_j)*	Operation *(O_{ji})*	Machine *(M_{ji})*	Processing time *(p_{ji})*
C_1	G_1	1	PR_1	1	1	J_1	O_{11}	M_{11}	p_{11}
	G_2	1		2	1		O_{12}	M_{12}	p_{12}
	.	.		.	.		.	.	.
	.	.		.	.		.	.	.
	.	.		.	.		.	.	.
	G_{m_1}	1		m_1	1		O_{1m_1}	M_{1m_1}	p_{1m_1}
	.	2		.	2	J_2	O_{21}	M_{21}	p_{21}
	.	.		.	.		.	.	.
	.	.		.	.		.	.	.
	.	.		.	.		.	.	.
	.	2		.	2		O_{2m_2}	M_{2m_2}	p_{2m_2}
	.	.		.	.	.	.	.	
	.	.		.	.	.	.	.	
	.	.		.	.	.	.	.	
		n		.	n	J_n	O_{n1}	M_{n1}	p_{n1}
	.	.		.	.		.	.	.
	.	.		.	.		.	.	.
	.	.		.	.		.	.	.
	G_θ	n		θ	n		O_{nm_n}	M_{nm_n}	p_{nm_n}

doi:10.1371/journal.pone.0169817.t001

The number of genes in each chromosome equals the number of total operations in a job-set, which is expressed by:

$$\theta = \sum_{j,j'=1}^{j,j'=n} m_{j,j'} \tag{28}$$

Chromosome generating. A chromosome *(C_r)* is a random construct of operations, which is expressed by

$$C_r = \left\{ \left(m_{j,j'} \right) | j,j' = 1,2,\ldots,n \right\} = \left\{ \underbrace{(O_{11}, O_{12}, \ldots, O_{1n})}_{m_1}, \underbrace{(O_{21}, O_{22}, \ldots, O_{2n})}_{m_2}, \ldots, \underbrace{(O_{n1}, O_{n2}, \ldots, O_{nn})}_{m_j} \right\} \tag{29}$$

where *j*, *j'* are indexes of jobs, *j*, *j'* = *1*, *2*, . . ., *n* and $m_{j,j'}$ = number of operations for each job. O_{ji} is the operation *i* of job *j*.

Chromosome coding and generating is explained below via an example of 3 jobs (J_1, J_2, *and* J_3). Each job has 4, 3, and 5 operations respectively. Overall, there is θ = 4+3+5 = 12 operations. Therefore, the chromosome is a random construct of $[\underbrace{1111}_{4}\ \underbrace{222}_{3}\ \underbrace{33333}_{5}]$. A sample could be [221132313133]. Here, code '*1*', '*2*', and '*3*' imply operations of J_1, J_2, and J_3 respectively. From the left, the first '*2*' represents the first operation of J_2, the second '*2*' represents the second operation of J_2, the first '*1*' represents the first operation of J_1, and so on.

Step 3. Multi-objective evaluation. After initializing the population size, each chromosome is evaluated by minimizing the makespan and the AGV numbers, while considering the

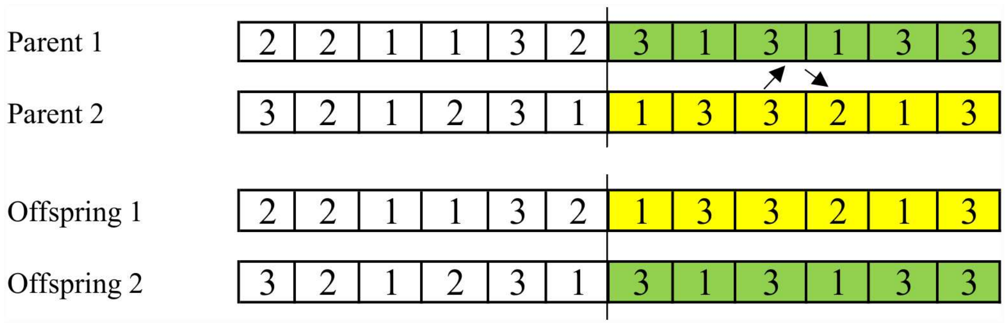

Fig 1. Example of one-point crossover.

doi:10.1371/journal.pone.0169817.g001

battery charge of AGV that are defined by Eqs 1 to 12. Then, the total fitness values of the efficient frontiers will be calculated based on Eq 13.

Step 4. New population. New population will be produced based on the below sub-steps: selection, crossover, elitism, and mutation operation.

Selection. To constantly enhance the population overall fitness, selection helps to discard the bad/weak designs and only keep the best ones in the population. It increases the likelihood of selection of fitter individuals for the next generation. There are a few different selection methods but their basis is the same. The tournament candidate selection, which is a proportionate random selection method, is used in this study. In this method, every individual in the population is paired at random with another. The fitness values of each pair is compared. The fitter individual of the pair moves on to the next round, while the other is disqualified. This continues until there are a number of winners equal to the desired number of parents. Then, this last group of winners is paired as the parents for new individuals [41].

Crossover. Crossover operator generates two new chromosomes for the next generation out of two selected chromosomes by exchanging some of their genes. This study employs two crossover operators based on partial strings exchange; a one-point crossover and a two-point crossover [42], where the one-point crossover is illustrated in Fig 1 based on the example in step 2.

The offspring of crossover between the strings may not produce a legal encoding, for example, uncorrected number of operations per job may be seen. Therefore, they should be repaired and legalized. For repairing mechanism, counting from the left, the redundant genes will be deleted and compensate the missing ones, in order for each offspring to comprise all the operations of all the jobs. Repair mechanism is shown in Fig 2.

Legal chromosomes for the example in step 2 should include four code '*1*', three code '*2*', and five code '*3*'. In Fig 2a, counting from the left, in offspring 1, code '*2*' is repeated four times, but there are three operations for J_2, so it should repeat three times. There is one code '*2*' that is redundant and should be replaced by the missing code. Code '*1*' is repeated four times, which is correct, but code '*3*' is repeated only four times, which should be 5 times. So in Fig 2b, the 4th code '*2*' will be replaced by number '*3*'. In offspring 2, number '*2*' is repeated two times and number '*3*' is repeated six times, so the last code '*3*' will be changed to code '*2*'.

The number of crossovers is calculated based on the crossover rate (*CR*) and population size (*PS*) using

$$Number\ of\ crossovers = \frac{(CR \times PS)}{2} \tag{30}$$

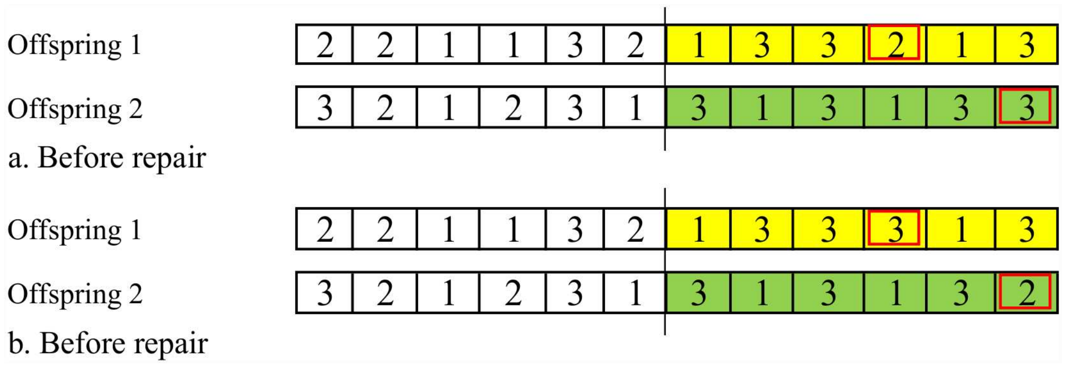

Fig 2. An example of repairing offsprings out of one-point crossover.

doi:10.1371/journal.pone.0169817.g002

Mutation. Mutation is another important operator of GA that initiates extra variability in a population to create and maintain the diversity. Mutation is not applied on chromosomes that are immune. The number of mutations in each generation is calculated using Eq (31) based on the mutation rate (*Pm*), population size (*PS*), and maximum gene code ($G_{\max}$).

$$\text{Number of mutations} \cong (PS \times G_{\max}) \times Pm \tag{31}$$

Shift mutation is used in this study [43] and it is shown in Fig 3. Based on the coding used in this study, chromosomes produced out of shift mutation are legal and no need to be repaired.

Elitism. The first three best chromosomes from each generation are transferred directly to the next generation in the elitism step to avoid annihilation. It is possible to maintain a fixed fitness value in some generations, but elitism makes sure they will never deteriorate.

Step 5. Termination. The loop of chromosome generation is terminated when the number of generation reaches its maximum, then the elite chromosome returns as the best solution.

Particle swarm optimization

PSO is a population based stochastic technique inspired by social behaviour of bird flocking or fish schooling. Extensive reviews on PSO algorithm development can be found in [44–46]. PSO with limited information has been studied to avoid the waste of information [47]. Whereby population topologies and their performance had be studied [48]. Gao & collaborators proposed a method called Selectively-informed PSO (SIPSO) to allow the particles to learn at difference strategies based on their connections [49]. The PSO configuration for the mathematical model is described in details in the following steps:

Step 1. Initializing parameters. Initialization involves setting the parameters of the PSO and creating a group of particles to make the initial swarm. The general scheme for reading the

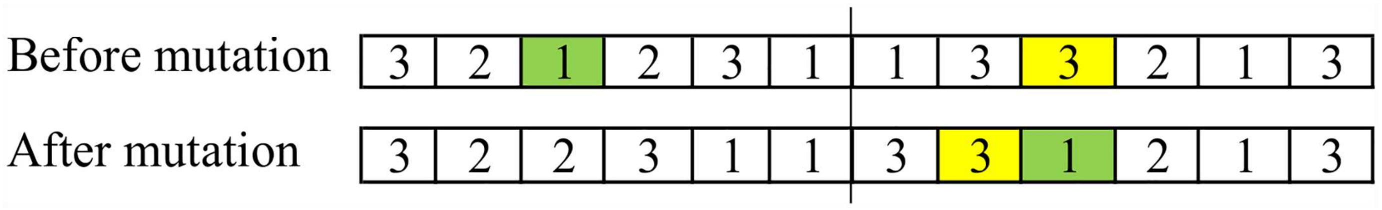

Fig 3. Example of shift mutation operator.

doi:10.1371/journal.pone.0169817.g003

data in the problem is presented in Table 1. The third column shows a particle (PR_α) and the forth one shows dimensions of the particle *(d)*. The dimensions' codes are presented in the third column, which will be discussed later.

Step 2. Initializing population (swarm). A group of particles are needed to create a swarm. Each particle has position *(Q)* and velocity *(V)* in the search space at iteration *(t)*, where they are described briefly in the following sub-steps:

Particle position. First position of particle is filled by two digit numbers for '*d*' dimensions of the particle using Eq 32. The number of dimensions is equal to the total number of operations, which is calculated by Eq 28.

$$q^0_{\alpha d} = q_{\min} + (q_{\max} - q_{\min}) \times \varphi_1 \tag{32}$$

where $q_{\min} = 0$, $q_{\max} = 10$ and φ_1 is a uniform random number between 0 and 1.

Particle velocity. Initial velocities for the PSO particles are generated by the formula below:

$$v^0_{\alpha d} = v_{\min} + (v_{\max} - v_{\min}) \times \varphi_2 \tag{33}$$

where $v_{min} = 0$, $v_{\max} = 10$ and φ_2 is a uniform random number between 0 and 1.

Step 3. Particle representation and encoding. Every possible sequence of operations is considered as a particle, where the dimension of the particle represents each operation. Three sub-steps for encoding a particle are as follows: applying smallest position value (SPV) rule, assigning the dimensions' codes to the particles, and identifying sequence of operations in each job.

1. **Applying smallest position value (SPV) rule.** SPV is a rule that facilitates transformation of the continuous PSO algorithm to discrete cases applicable to all types of the scheduling problem [50]. As an example for better understanding of SPV rule, the corresponding sequence of a given continuous position like [0.3, 1.2, 0.9, 2.4] would be [4, 2, 3, 1]. In a descending order, '*0.3*' is the smallest value and its sequence will be '*4*'; '*2.4*' is the largest so its order in the group will be '*1*'.

Assigning the dimensions' codes to the particles. In this stage, the dimension's codes as it is shown in the 6th column of Table 1 are assigned to the particles. Dimension's codes are based on the job number.

Identifying sequence of operations in each job. From the left side, the first appearance of a job number is assumed the first operation of that job (i.e., O_{j1}). Similarly, the second time repetition of the same job number stands for the second operation of the same job (i.e., O_{j2}) and so on. Once the first encountered generated number is assigned to the first operation of a job, this technique automatically handles the precedence constraints.

The stages of encoding an example with 3 jobs are shown in Table 2. Each job has 4, 3, and 5 operations respectively. The total operations are 12, which means the particle sample will have 12 dimensions being randomly generated using Eq 32 and shown in the first row of Table 2. In the second row of the Table, based on SPV rule, the numbers of 1 to 12 are assigned to the particles in an ascending order. In the third row, the dimensions' codes based on the job

Table 2. Encoding of a sample particle.

Particle sample	0.2	0.37	0.17	0.51	0.73	0.42	0.93	0.35	0.69	0.84	0.65	0.05
Applying SPV rule (giving the numbers from one based on ascending order)	3	5	2	7	10	6	12	4	9	11	8	1
Assigning the dimensions' codes to the particles	1	2	1	2	3	2	3	1	3	3	3	1
Identifying sequence of operations in each job	O_{11}	O_{21}	O_{12}	O_{22}	O_{31}	O_{23}	O_{32}	O_{13}	O_{33}	O_{34}	O_{35}	O_{14}

doi:10.1371/journal.pone.0169817.t002

Table 3. AGV travel time among L/U point and machines for example 1.

	L/U	M_1	M_2	M_3	M_4	M_5	M_6
L/U	0	5	7	10	12	17	19
M_1	15	0	2	5	7	12	14
M_2	13	18	0	3	5	10	12
M_3	10	15	17	0	2	7	9
M_4	8	13	15	18	0	5	7
M_5	3	8	10	13	15	0	2
M_6	1	6	8	11	13	18	0

doi:10.1371/journal.pone.0169817.t003

numbers are given to the particles as follows: first four numbers are assigned to the first job, so their code is '*1*', followed by the second three numbers assigned to the second job, so their code is '*2*' and the remaining five numbers are assigned to the third job and their code is '*3*'. The sequence of operations in each job is shown in the fourth row of the Table 3. From the left, the first particle has the code '*1*', so it belongs to job 1 and it is the first code '*1*', which makes it the first operation of job 1 denoted by O_{11}; the next code is '*2*', so it belongs to job 2, but as it is the first code '*2*', it is the first operation of job 2. The same structure is followed for the remained 10 operations.

Step 4. Multi-objective evaluation. Once the swarm is generated, each particle is evaluated by minimizing the makespan and AGV numbers, while considering the AGV battery charge, which are defined by Eqs 1 to 12. Then, the total fitness values of the efficient frontiers will be calculated based on Eq 13.

Personal best. B_α^t represents the best position associated with the best permutation and fitness value of the particle α obtained so far and is called the personal best. For each particle, B_α^t can be determined and updated at each iteration.

Global best. G^t denotes the best position of the globally best particle achieved so far in the whole swarm.

Step 5. New swarm. To produce a new swarm, the position and velocity of the particles should be updated. Updated particles will be evaluated again according to the step four and the best local and global particle will be determined. This procedure will be repeated up to a point where the termination criterion is satisfied. The updating procedure is explained as follows:

1. **Updating the velocity of each particle.** The velocity of each particle is updated using

$$v_{\alpha d}^{t+1} = \omega\, v_{\alpha d}^{t} + C_1\varphi_1(B_{\alpha d}^{t} - q_{\alpha d}^{t}) + C_2\varphi_2(G_d^{t} - q_{\alpha d}^{t}) \qquad (34)$$

where C_1 is self-confidence while C_2 is swarm confidence (common values of C_1 and C_2 varies between 0.1 and 0.5 but the values between 0.1 and 1 has been tested as well. In some literature, the value of 2 have also been observed). Inertia weight (ω) is a parameter to control the impact of the previous velocity on the current velocity [51, 52]. Let ω be varying with time by the following linear decreasing function.

$$\omega = \omega_{\max} - Iter \times \frac{\omega_{\max} - \omega_{\min}}{Iter_{\max}} \qquad (35)$$

2. **Updating the position of each particle.** The position of particle is updated using the updated velocity as below:

$$q_{\alpha d}^{t+1} = q_{\alpha d}^{t} + v_{\alpha d}^{t+1} \tag{36}$$

Step 6. Termination. The loop of swarm groups is terminated when it reaches the maximum number of iteration, then the particle with global best returns as the best solution.

Hybrid GA and PSO

The PSO algorithm is a more robust optimization algorithms compared to many other algorithms as it can work almost independent from the problem. It does not require extensive prior-knowledge regarding the problem except the fitness evaluation of each particle [53]. On the other hand, GA has the capability of simultaneous evaluation of many points in the search area, which increases the probability of finding the global solution of the problem. Hybridization of EAs has been studied in many researches [54–56]. Generally, PSO functions based on the social interaction knowledge and all the individual particles will be considered in each generation. Unlike PSO, fitter chromosomes will be chosen in GA and the weaker ones will fade away from generation to generation [57]. Hence, by integrating the advantages of the compensatory properties of PSO and GA, their hybrid is used to obtain better result [58–60]. In the proposed GA-PSO algorithm for this study, after generating and evaluating the initial swarm and after position and velocity updating, the crossover operation has been used in the GA segment to avoid premature convergence; and a mutation operation was applied to maintain the diversity of the swarms. Elitism step was also performed to improve immune particle filter. Fig 4 illustrates the steps of hybrid GA-PSO and some parts of programming codes are listed in S1 Appendix.

Computational results and discussion

To validate the model, two numerical examples have been used. The first example had 6 jobs ($J_1, \ldots, J_6$) processing on 6 machines ($M_1, \ldots, M_6$), and each job with 2 to 5 operations. The second one with 15 jobs ($J_1, \ldots, J_{15}$) processing on 10 machines ($M_1, \ldots, M_{10}$), and each job with 1 to 5 operations [23, 61]. Tables 3 and 4 show the AGV travel time among L/U point and machines and Tables 5 and 6 demonstrate the processing time of every operation on the machines for both the examples.

The makespan of scheduling before optimization by a random sequence and assigning one AGV to each of the six jobs for example 1 is shown in Fig 5. However, illustration of before optimization for example 2 was not possible due to its big figure size and detail.

Based on the experimental approach, the best setting of hybrid GA-PSO parameters was found to be the crossover and mutation rates of 0.2 and 0.08 respectively, $C_1 = 0.01$, $C_2 = 0.9$, $\omega_{\min} = 0.01$, and $\omega_{\max} = 0.5$. The algorithms were run 30 times, each run with a population size of 100 in 100 iterations, and their first two best results based on different AGV numbers are shown in Table 7.

The third column of Table 7 shows the best result of each algorithm, and the forth column shows the best result of each algorithm using a different number of AGVs compared with the first column. The fitness value in Table 7 has been calculated based on Eq 13, $\psi = \frac{max\ (MS)}{max\ (NA)}$, and $\delta_1 = \frac{2}{3}$. *Max* (*MS*) was presumed to be equal to the sum of travel times and operation times. *Max* (*NA*) was presumed equal to the whole number of operations. All the steps were repeated

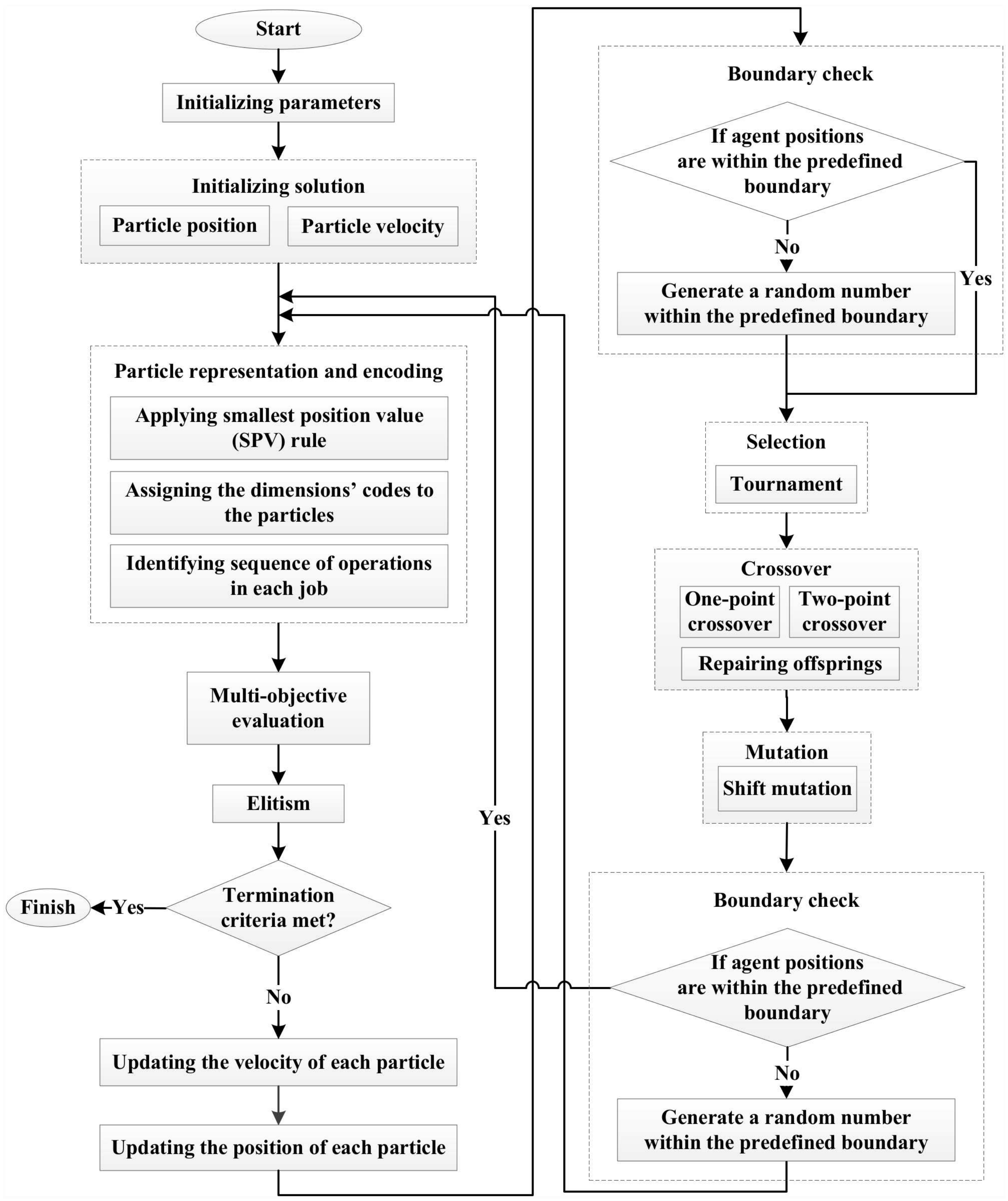
Start
Initializing parameters
Initializing solution
Particle position
Particle velocity
Particle representation and encoding
Applying smallest position value (SPV) rule
Assigning the dimensions' codes to the particles
Identifying sequence of operations in each job
Multi-objective evaluation
Elitism
Termination criteria met?
Yes
Finish
No
Updating the velocity of each particle
Updating the position of each particle
Boundary check
If agent positions are within the predefined boundary
No
Generate a random number within the predefined boundary
Yes
Selection
Tournament
Crossover
One-point crossover
Two-point crossover
Repairing offsprings
Mutation
Shift mutation
Boundary check
If agent positions are within the predefined boundary
Yes
No
Generate a random number within the predefined boundary

Fig 4. Flowchart of hybrid GA-PSO.

doi:10.1371/journal.pone.0169817.g004

Table 4. AGV travel time among L/U point and machines for example 2.

Min	L/U	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
L/U	0	8	12	18	13	12	22	9	21	20	25
M_1	8	0	6	17	20	16	4	5	12	14	19
M_2	12	6	0	19	8	7	9	17	24	13	21
M_3	18	17	19	0	12	5	4	9	6	13	10
M_4	13	20	8	12	0	10	15	12	21	4	18
M_5	12	16	7	5	10	0	5	23	18	17	9
M_6	22	4	9	4	15	5	0	2	12	15	17
M_7	9	5	17	9	12	23	2	0	14	19	23
M_8	21	12	24	6	21	18	12	14	0	11	18
M_9	20	14	13	13	4	17	15	19	11	0	17
M_{10}	25	19	21	10	18	9	17	23	18	17	0

doi:10.1371/journal.pone.0169817.t004

Table 5. The processing time of every operation on the machines for example 1.

Job	1	1	1	1	2	2	2	3	3	3	3	3	4	4	4	5	5	6	6
Operation	1	2	3	4	1	2	3	1	2	3	4	5	1	2	3	1	2	1	2
Machine	M_2	M_4	M_5	M_6	M_3	M_4	M_6	M_1	M_2	M_3	M_1	M_4	M_6	M_4	M_5	M_5	M_1	M_1	M_2
Operation time	30	21	24	27	15	24	13	16	21	18	14	25	25	19	20	33	21	27	31

doi:10.1371/journal.pone.0169817.t005

for example 2. Fig 6 shows the performance of all the three algorithms at examples 1 and 2 based on the third and fifth column of the Table 7, respectively.

After the optimization, all the three algorithms were proved successful in decreasing the makespan and the required number of AGVs, and the optimized model using hybrid GA-PSO obtained the best result.

Fig 7 demonstrates the optimized sequence of Fig 5 using only 3 AGVs which is obtained by hybrid GA-PSO. In Figs 7 and 8, although the battery charge of AGV was considered, the path of AGVs going home for recharging is not shown to avoid extra complexity. Fig 8 shows the optimized sequence of example 2 obtained using hybrid GA-PSO (nearly half of the time points are not shown due to space limitations).

Table 6. The processing time of every operation on the machines for example 2.

Job	1	1	1	1	1	2	2	2	3	3	4	4	4	4	5	6
Operation	1	2	3	4	5	1	2	3	1	2	1	2	3	4	1	1
Machine	M_7	M_1	M_9	M_8	M_3	M_2	M_{10}	M_6	M_5	M_4	M_1	M_3	M_5	M_7	M_{10}	M_8
Operation time	19	21	14	10	11	15	21	22	30	26	1	12	19	14	29	12
Job	6	6	7	7	8	8	8	8	8	9	9	9	10	10	10	10
Operation	2	3	1	2	1	2	3	4	5	1	2	3	1	2	3	4
Machine	M_6	M_3	M_2	M_7	M_9	M_3	M_4	M_5	M_8	M_{10}	M_9	M_5	M_5	M_1	M_8	M_2
Operation time	24	17	29	16	9	21	24	10	12	20	10	16	17	5	24	20
Job	11	11	11	11	12	13	13	14	14	14	14	14	15	15	15	
Operation	1	2	3	4	1	1	2	1	2	3	4	5	1	2	3	
Machine	M_7	M_9	M_8	M_1	M_6	M_{10}	M_1	M_3	M_4	M_2	M_{10}	M_9	M_4	M_8	M_7	
Operation time	11	11	14	15	49	29	30	12	11	9	10	10	15	21	4	

doi:10.1371/journal.pone.0169817.t006

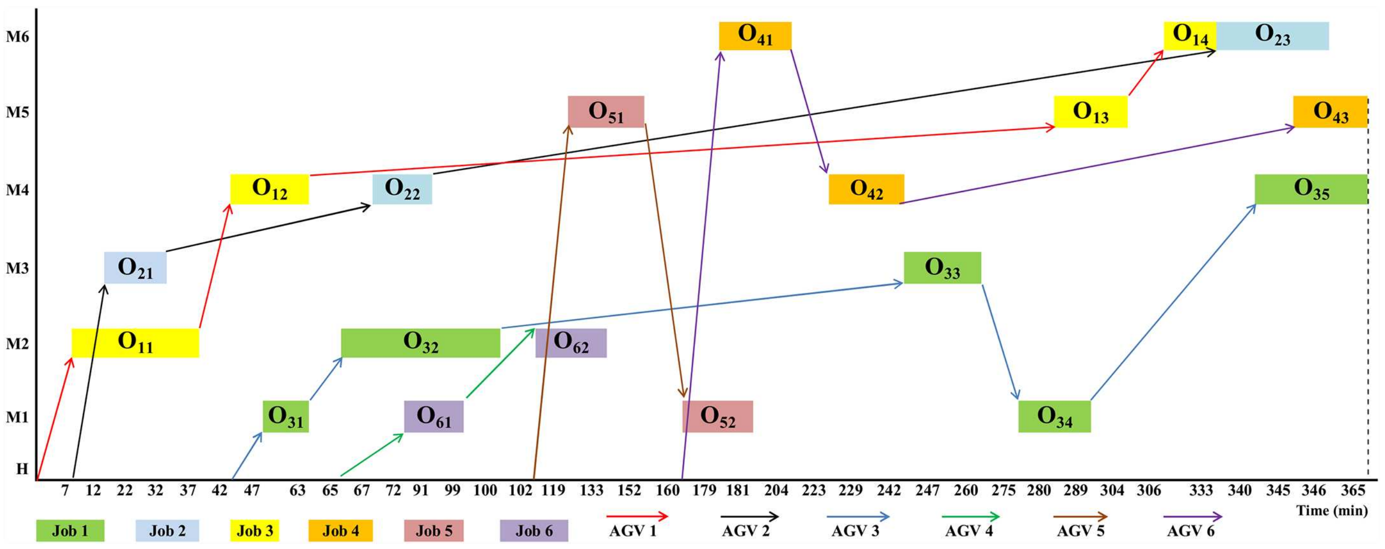

Fig 5. Gantt chart of a random sequence of the example using the six AGVs before optimization.

doi:10.1371/journal.pone.0169817.g005

To investigate the effect of optimization methods on AGV scheduling, AGVs' specification both before and after the optimization for example 1 were explored. The studied specifications are AGVs' total running time (loaded and unloaded), idle time, battery usage, and operation efficiency computed using Eqs 15 to 27. In Fig 9, prior to the optimization, the AGVs total running time is low because a higher number of AGVs are employed with no intention to use their highest potential, compared to the optimized schedule, thus the idle time of AGVs has increased dramatically. The AGV number four (AGV_4) had the highest operation efficiency (37.3%) before the optimization; although the scheduling model was designed to sequentially appoint tasks to AGVs based on their numbers' order, so that AGV number one (AGV_1) would have the highest operation efficiency level.

In GA-PSO, the makespan, number of AGVs and their idle time have been reduced, and consequently efficiency of AGVs' operation has enhanced (Fig 9). Potency of hybrid GA-PSO in solving scheduling problems and its superiority against its constituting algorithms have also

Table 7. Test results of optimization algorithms (The first two best result of each) for both examples.

Algorithms		Example 1			Example 2		
		The first two best results		Mean computational time	The first two best results		Mean computational time
PSO	Fitness value	167.0877	172.5614	58.9164 Sec	625.9148	640.5532	261.0752 Sec
	Makespan	184	170		792	765	
	Number of AGV	3	4		6	8	
GA	Fitness value	163.0877	164.9474	60.3881 Sec	589.9148	590.234	269.0021 Sec
	Makespan	178	203		738	714	
	Number of AGV	3	2		6	7	
Hybrid GA-PSO	Fitness value	161.7544	162.2807	61.5003 Sec	566.2623	568.2339	273.7746 Sec
	Makespan	176	199		727	681	
	Number of AGV	3	2		5	7	

doi:10.1371/journal.pone.0169817.t007

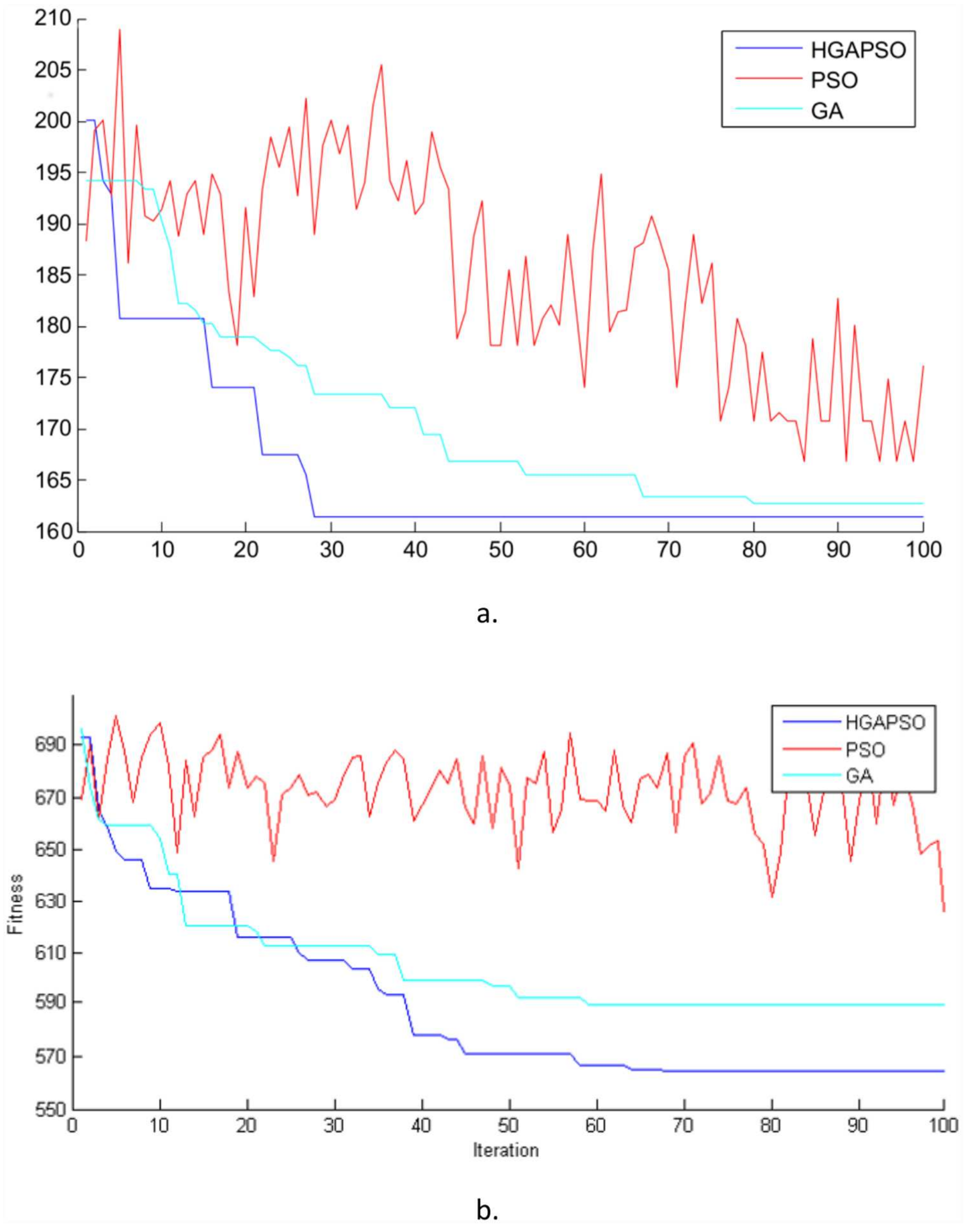

Fig 6. Performance of the different algorithms a. Example 1, b. Example 2.

doi:10.1371/journal.pone.0169817.g006

been largely mentioned in other published studies [62–65]. Overall, application of the hybrid GA-PSO in scheduling studies is concluded to be more effective than its constituting EAs.

Simulation by Flexsim

In order to prove the feasibility of the proposed model, a simulation practice based on the above example has been performed using the Flexsim software. Fig 10 shows a scene from the simulation space. The simulation outcome confirmed the optimization results by obtaining equal makespan magnitude to all the three algorithms. The experimental results proved the validity and feasibility of the model, which provide useful reference for further research on the

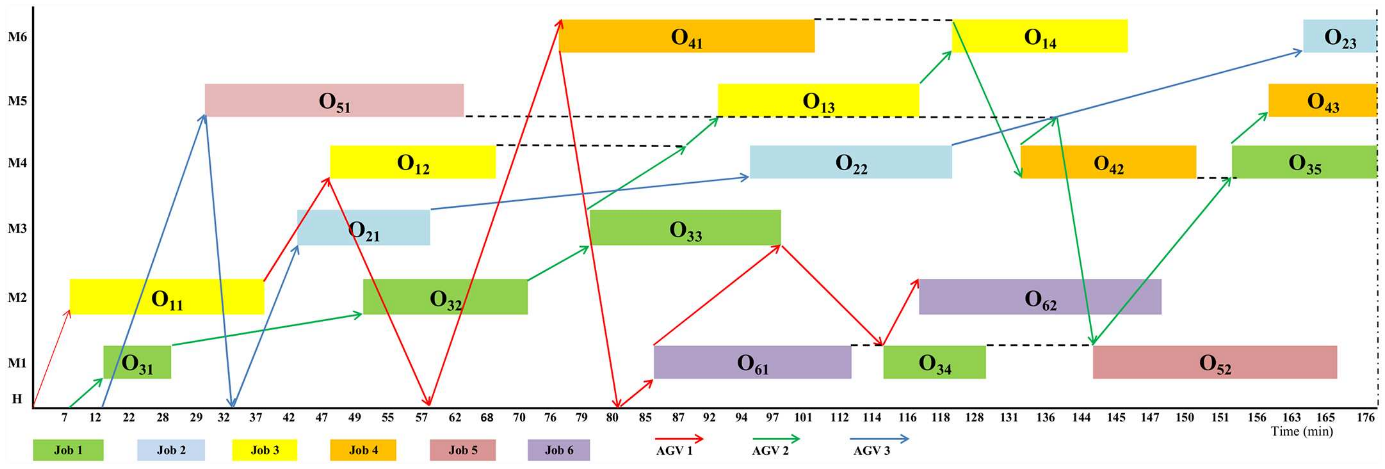

Fig 7. Gantt chart of the schedule of the example 1 after optimization by GA-PSO that employs three AGVs.

doi:10.1371/journal.pone.0169817.g007

scheduling of AGV. Other experiments were also simulated to check the suitability and compatibility of the model to any kind of FMS configuration and environment. It can also be utilized for optimizing the objectives separately as well as in a combination.

Conclusion

This research focused on the multi-objective AGV scheduling in an FMS using GA, PSO, and hybrid GA-PSO algorithms. A model was developed for the task scheduling of AGVs considering multiple objectives of minimizing makespan and number of AGVs, while considering the battery charge of AGVs. Using the numerical examples, near-optimum schedules for the combined objective functions were obtained. The inter-comparison of the three algorithms results showed that the hybrid GA-PSO yields the least makespan and AGV numbers. Literature has also largely exhibited the excellence of hybrid GA-PSO over its constituents in solving the scheduling problems [62–65], however the scheduling model proposed in this study distinguishes it from previous studies. The scheduling problem was further scrutinized by comparing the AGVs characteristics such as the total running time (loaded and unloaded), idle time, battery usage, and operation efficiency—before and after the optimization. It was found that

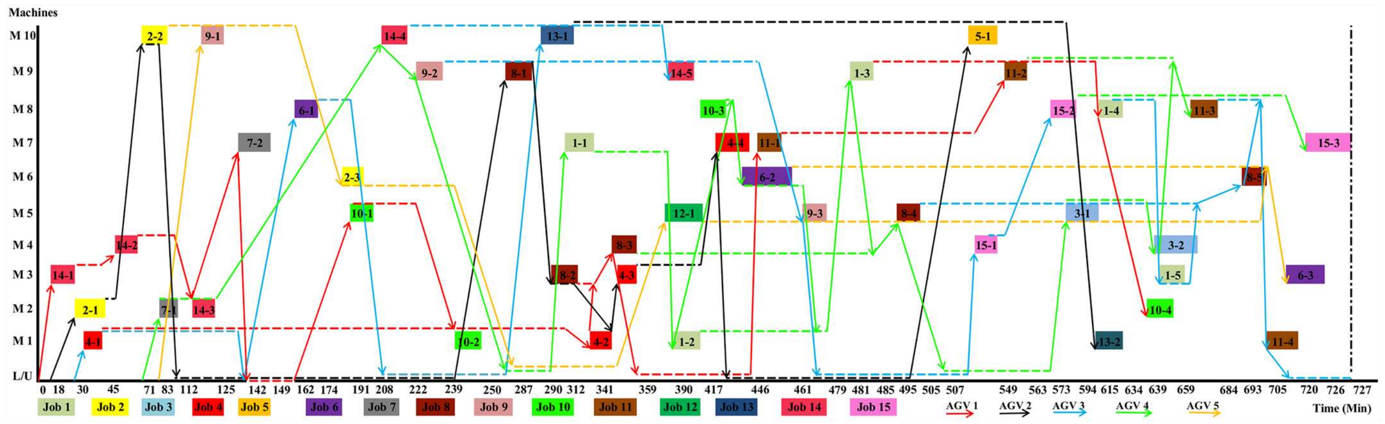

Fig 8. Gantt chart of the schedule of the example 2 after optimization by GA-PSO that employs five AGVs.

doi:10.1371/journal.pone.0169817.g008

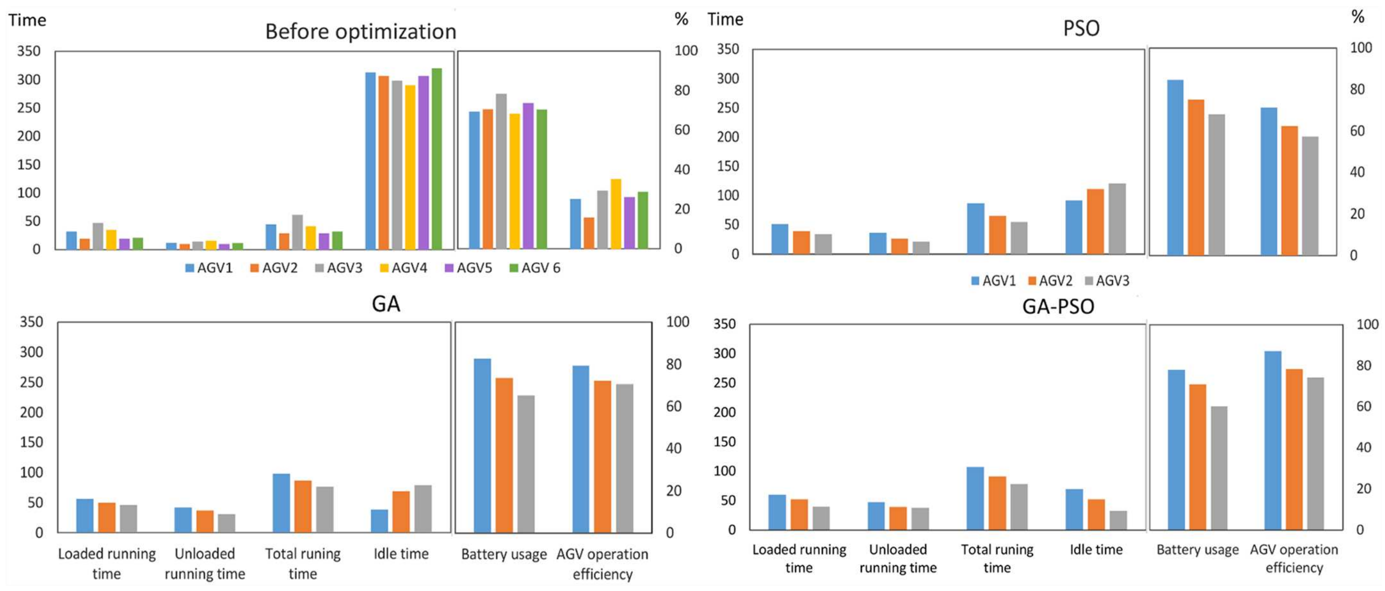

Fig 9. AGVs specification before and after optimization by PSO, GA, GA-PSO for the example 1.

doi:10.1371/journal.pone.0169817.g009

after the optimization, despite a small rise in AGVs' total running time (loaded and unloaded), the AGVs' idle time reduction enhanced the AGVs' operation efficiency. This leads to effective utilization of AGVs and hence the overall efficiency of the system will be enhanced. In line with the experimental results, the AGV system simulation using the Flexsim software has also proved the feasibility of the developed model and suitability of the optimization algorithms for

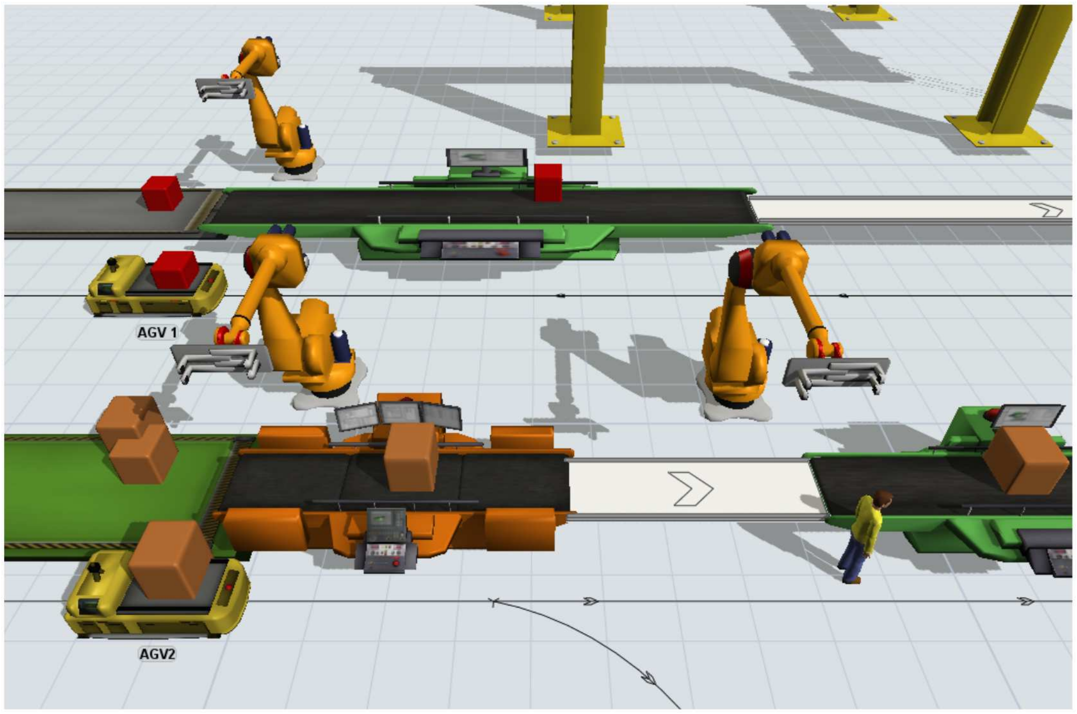

Fig 10. Simulation of the sample by Flexsim software.

doi:10.1371/journal.pone.0169817.g010

the scheduling problem. The developed model can be adopted to any FMS with different configuration and environment, and it can be applied for optimizing the objectives separately or in a combinatorial fashion. This research framework can be stretched out to examine newer algorithms and hybrids, and also employ more criteria in the scheduling model for further studies in this context.

Supporting information

S1 Appendix. Programming codes for hybrid GA-PSO.
(PDF)

Acknowledgments

The authors would like to acknowledge University of Malaya's financial support of this project under UMRG Top Down Programme (Grant No. RP027-14AET). The authors would also like to acknowledge the Ministry of Higher Education of Malaysia for the facility support under High Impact Research Grant UM.C/HIR/MOHE/ENG/35 (D000035-16001).

Author Contributions

Conceptualization: MM FT.

Data curation: MM.

Formal analysis: MM.

Funding acquisition: SZMD HJY.

Investigation: MM.

Methodology: MM HJY SNM FT.

Project administration: MM HJY.

Resources: HJY SNM FT SZMD.

Software: MM FT.

Supervision: HJY SNM FT.

Validation: MM HJY FT.

Visualization: MM.

Writing – original draft: MM.

Writing – review & editing: MM.

References

1. Blazewicz J, Eiselt HA, Finke G, Laporte G, Weglarz J. Scheduling tasks and vehicles in a flexible manufacturing system. International Journal of Flexible Manufacturing Systems. 1991; 4(1):5–16.
2. Reddy B, Rao C. Flexible manufacturing systems modelling and performance evaluation using AutoMod. International Journal of Simulation Modelling. 2011; 10(2):78–90.
3. Vasava AS. Scheduling of automated guided vehicle in different flexible manufacturing system environment. International Journal of Innovative Research in Advanced Engineering (IJIRAE). 2014; 1(8):262–7.

4. Sabuncuoglu I, Hommertzheim DL. Dynamic dispatching algorithm for scheduling machines and automated guided vehicles in a flexible manufacturing system. The International Journal Of Production Research. 1992; 30(5):1059–79.
5. Ren NF, Liu D, Zhao Y, Ge XB. AGV Scheduling Optimizing Research of Collaborative Manufacturing System Based on Improved Genetic Algorithm. Applied Mechanics and Materials. 2013; 300:55–61.
6. Anwar MF, Nagi R. Integrated scheduling of material handling and manufacturing activities for just-in-time production of complex assemblies. International Journal of Production Research. 1998; 36 (3):653–81.
7. Fauadi MHFBM, Murata T, editors. Makespan Minimization of Machines and Automated Guided Vehicles Schedule Using Binary Particle Swarm Optimization. Proceedings of the International MultiConference of Engineers and Computer Scientists (IMECS); 2010; Hong Kong: Citeseer.
8. Pan XY, Wu J, Zhang QW, Lai D, Xie HL, Zhang C. A Case Study of AGV Scheduling for Production Material Handling. Applied Mechanics and Materials. 2013; 411:2351–4.
9. Udhayakumar P, Kumanan S. Task scheduling of AGV in FMS using non-traditional optimization techniques. International Journal of Simulation Modelling. 2010; 9(1):28–39.
10. Zheng K, Tang D, Gu W, Dai M. Distributed control of multi-AGV system based on regional control model. Production Engineering. 2013; 7(4):433–41.
11. Kumar MS, Janardhana R, Rao C. Simultaneous scheduling of machines and vehicles in an FMS environment with alternative routing. The International Journal of Advanced Manufacturing Technology. 2011; 53(1–4):339–51.
12. Fazlollahtabar H, Shafieian SH. An Optimal Path in an AGV-based Manufacturing System with Intelligent Agents. Journal for Manufacturing Science and Production. 2014; 14(2):87–102.
13. Novas JM, Henning GP. Integrated scheduling of resource-constrained flexible manufacturing systems using constraint programming. Expert Systems with Applications. 2014; 41(5):2286–99.
14. Kato F, Shin S, editors. Multistep optimal scheduling of Automated Guided Vehicles in a semiconductor fabrication. Proceedings of SICE Annual Conference 2010; 2010: IEEE.
15. Ventura JA, Pazhani S, Mendoza A. Finding optimal dwell points for automated guided vehicles in general guide-path layouts. International Journal of Production Economics. 2015; 170:850–61. http://dx.doi.org/10.1016/j.ijpe.2015.03.007.
16. Rashmi M, Bansal S. Task Scheduling of Automated Guided Vehicle in Flexible Manufacturing System using Ant Colony Optimization. International Journal of Latest Trends in Engineering and Technology (IJLTET). 2014; 4(1):177–81.
17. Wang HF, Chan CH. Multi-objective optimisation of automated guided dispatching and vehicle routing system. International Journal of Modelling in Operations Management. 2014; 4(1):35–52. http://dx.doi.org/10.1504/IJMOM.2014.063585.
18. Cai Q, Tang D, Zheng K, Zhu H, Wu X, Lu X. Multi-AGV scheduling optimization based on neuro-endocrine coordination mechanism. International Journal on Smart Sensing and Intelligent Systems. 2014; 7 (4):1613–30.
19. Suzuki T, Hirogaki T, Aoyama E, Ogawa K, Ito T, editors. Influence of the Number of AGVs on Products Conveyance Efficiency in AGV Transportation System Based on Knowledge of Taxis. ASME 2014 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference; 2014; New York, USA: American Society of Mechanical Engineers.
20. Nageswararao M, Narayanarao K, Ranagajanardhana G. Simultaneous Scheduling of Machines and AGVs in Flexible Manufacturing System with Mean Tardiness Criterion by using HGVHA. INROADS-An International Journal of Jaipur National University. 2014; 3(1s):62–8.
21. Kaplanoğlu V, Şahi C, Baykasoğlu A, Erol R, Ekinci A, Demirtaş M, editors. A Multi-Agent Based Approach to Dynamic Scheduling of Machines and Automated Guided Vehicles (AGV) in Manufacturing Systems by Considering AGV Breakdowns. 4th IAJC/ISAM Joint International Conference on engineering and related technologies; 2014; Orlando, Florida.
22. Huang D, Zhang G, editors. Scheduling control of AGV system based on game theory. 6th International Conference on Advanced Infocomm Technology (ICAIT); 2013; Hsinchu, Taiwan: IEEE.
23. Saidi-Mehrabad M, Dehnavi-Arani S, Evazabadian F, Mahmoodian V. An Ant Colony Algorithm (ACA) for solving the new integrated model of job shop scheduling and conflict-free routing of AGVs. Computers & Industrial Engineering. 2015; 86:2–13. http://dx.doi.org/10.1016/j.cie.2015.01.003.
24. Azimi P. Alleviating the collision states and fleet optimization by introducing a new generation of automated guided vehicle systems. Modelling and Simulation in Engineering. 2011; 2011:2.
25. Aized T. Modelling and performance maximization of an integrated automated guided vehicle system using coloured Petri net and response surface methods. Computers & Industrial Engineering. 2009; 57 (3):822–31. http://dx.doi.org/10.1016/j.cie.2009.02.009.

26. Liang Y, Lin L, Gen M, Chien CF, editors. A hybrid evolutionary algorithm for FMS optimization with AGV dispatching. Computers and Industrial Engineering 42; 2012; Cape Town, South Africa.

27. Wang JB, Hou LY, Li W, Zheng XJ. Simulating an AGV Scheduling in Job Workshop for Optimal Configuration. In: Peilong Xu HS, Yiqian Wang and Pin Wang, editor. Advanced Materials Research. 926–930: Trans Tech Publ; 2014. p. 1562–5.

28. Kawakami T, Takata S. Battery Life Cycle Management for Automatic Guided Vehicle Systems. Design for Innovative Value Towards a Sustainable Society: Springer; 2012. p. 403–8.

29. Oliveira MM, Galdames JPM, Vivaldini K, Magalhaes DV, Becker M, editors. Battery state estimation for applications in intelligent warehouses. IEEE International Conference on Robotics and Automation (ICRA); 2011; Shanghai: IEEE.

30. Inc. EA. Battery charging systems for automated guided vehicles 2006 [cited 2016 3 March]. http://www.egeminusa.com/pages/agvs/agvs_battery_charging.html.

31. INSTITUUT AK. General Technology Description of AGV-systems The Netherlands2015 [cited 2016 11 March]. http://www.frog.nl/Oplossingen/AGV_Kennis_Instituut.

32. Pareto V. Corso di economia politica: P. Boringhieri; 1961.

33. Eichfelder G. Adaptive scalarization methods in multiobjective optimization: Springer; 2008.

34. Ghane-Kanafi A, Khorram E. A new scalarization method for finding the efficient frontier in non-convex multi-objective problems. Applied Mathematical Modelling. 2015; 39(23):7483–98. http://dx.doi.org/10.1016/j.apm.2015.03.022.

35. Giagkiozis I, Fleming PJ. Methods for multi-objective optimization: An analysis. Information Sciences. 2015; 293:338–50. http://dx.doi.org/10.1016/j.ins.2014.08.071.

36. Holland JH. Adaptation in natural and artificial system: an introduction with application to biology, control and artificial intelligence: The University of Michigan Press; 1975.

37. Joshi G. Review of Genetic Algorithm: An Optimization Technique. International Journal of Advanced Research in Computer Science and Software Engineering. 2014; 4(4):802–5.

38. Elsayed SM, Sarker RA, Essam DL. A new genetic algorithm for solving optimization problems. Engineering Applications of Artificial Intelligence. 2014; 27:57–69.

39. Thakur M, Meghwani SS, Jalota H. A modified real coded genetic algorithm for constrained optimization. Applied Mathematics and Computation. 2014; 235:292–317.

40. Beheshti Z, Shamsuddin SMH. A review of population-based meta-heuristic algorithms. Int J Adv Soft Comput Appl. 2013; 5(1):1–35.

41. Chudasama C, Shah S, Panchal M, editors. Comparison of parents selection methods of genetic algorithm for TSP. International Conference on Computer Communication and Networks CSI-COMNET-2011, Proceedings; 2011.

42. Spears WM, Anand V. A study of crossover operators in genetic programming: Springer; 1991.

43. Nearchou AC. The effect of various operators on the genetic search for large scheduling problems. International Journal of Production Economics. 2004; 88(2):191–203.

44. Kennedy J, Eberhart RC, Shi Y. Swarm intelligence. US: Morgan Kaufmann; 2001.

45. Song ML, editor A Study of Single-objective Particle Swarm Optimization and Multi-objective Particle Swarm Optimization. Applied Mechanics and Materials; 2014: Trans Tech Publ.

46. Li X, Yao X. Cooperatively coevolving particle swarms for large scale optimization. Evolutionary Computation, IEEE Transactions on. 2012; 16(2):210–24.

47. Du W-B, Gao Y, Liu C, Zheng Z, Wang Z. Adequate is better: particle swarm optimization with limited-information. Applied Mathematics and Computation. 2015; 268:832–8.

48. Kennedy J, Mendes R, editors. Population structure and particle swarm performance. Evolutionary Computation, 2002 CEC'02 Proceedings of the 2002 Congress on; 2002: IEEE.

49. Gao Y, Du W, Yan G. Selectively-informed particle swarm optimization. Scientific reports. 2015;5.

50. Tasgetiren MF, Sevkli M, Liang YC, Gencyilmaz G, editors. Particle swarm optimization algorithm for single machine total weighted tardiness problem. Evolutionary Computation CEC2004; 2004: IEEE.

51. Kuo H, Horng SJ, Kao TW, Lin TL, Lee CL, Terano T, et al. An efficient flow-shop scheduling algorithm based on a hybrid particle swarm optimization model. Expert systems with applications. 2009; 36(3):7027–32. http://dx.doi.org/10.1016/j.eswa.2008.08.054.

52. Xia W, Wu Z. An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems. Computers & Industrial Engineering. 2005; 48(2):409–25. http://dx.doi.org/10.1016/j.cie.2005.01.018.

53. Xia W, Wu Z. A hybrid particle swarm optimization approach for the job-shop scheduling problem. The International Journal of Advanced Manufacturing Technology. 2006; 29(3–4):360–6.

54. Chelouah R, Siarry P. Genetic and Nelder–Mead algorithms hybridized for a more accurate global optimization of continuous multiminima functions. European Journal of Operational Research. 2003; 148 (2):335–48. http://dx.doi.org/10.1016/S0377-2217(02)00401-0.

55. Fan SKS, Zahara E. A hybrid simplex search and particle swarm optimization for unconstrained optimization. European Journal of Operational Research. 2007; 181(2):527–48. http://dx.doi.org/10.1016/j.ejor.2006.06.034.

56. Kao YT, Zahara E. A hybrid genetic algorithm and particle swarm optimization for multimodal functions. Applied Soft Computing. 2008; 8(2):849–57. http://dx.doi.org/10.1016/j.asoc.2007.07.002.

57. Liou CD, Hsieh YC, Chen YY. A new encoding scheme-based hybrid algorithm for minimising two-machine flow-shop group scheduling problem. International Journal of Systems Science. 2013; 44 (1):77–93. http://dx.doi.org/10.1080/00207721.2011.581396.

58. Mehta M. Hybrid Genetic Algorithm with PSO Effect for Combinatorial Optimization Problems. International Journal of Advanced Computer Research. 2012; 2(4):300–5.

59. Wu C-H, Dong N, Ip W-H, Chan C-Y, Yung K-L, Chen Z-Q. Chaotic hybrid algorithm and its application in circle detection. Applications of Evolutionary Computation: Springer; 2010. p. 302–11.

60. Wang L, Si G, editors. Optimal location management in mobile computing with hybrid genetic algorithm and particle swarm optimization (GA-PSO). Electronics, Circuits, and Systems (ICECS), 2010 17th IEEE International Conference on; 2010: IEEE.

61. Zeng C, Tang J, Yan C. Scheduling of no buffer job shop cells with blocking constraints and automated guided vehicles. Applied Soft Computing. 2014; 24:1033–46.

62. Jamrus T, Chien CF, Gen M, Sethanan K, editors. Hybrid Particle Swarm Optimization with Genetic Operators and Cauchy Distribution for Flexible Job-shop Scheduling Problem. 14th Asia Pacific Industrial Engineering and Management Systems; 2013; Cebu, Philippines.

63. Tang J, Zhang G, Lin B, Zhang B. A hybrid PSO/GA algorithm for job shop scheduling problem. Advances in Swarm Intelligence: Springer Berlin Heidelberg; 2010. p. 566–73.

64. Kaveh A, Malakouti Rad S. Hybrid genetic algorithm and particle swarm optimization for the force method-based simultaneous analysis and design. Iranian Journal of Science and Technology, Transaction B: Engineering. 2010; 34(B1):15–34.

65. Premalatha K, Natarajan A. Hybrid PSO and GA for global maximization. International Journal of Open Problems in Computational Mathematics. 2009; 2(4):597–608.