A hybrid GA-PSO algorithm for *seru* scheduling problem with dynamic resource allocation

Ling Shen, Zhe Zhang* and Xiaoling Song

School of Economics and Management,
Nanjing University of Science and Technology,
Nanjing, China
Email: zhangzhe@njust.edu.cn
*Corresponding author

Yong Yin

Graduate School of Business,
Doshisha University,
Karasuma-Imadegawa, Kamigyo-ku, Kyoto, Japan

Abstract: As a new type flexible production mode, *seru* production is widely used in Japanese enterprises to deal with the manufacturing market with volatile and diversified demand. In practical *seru* production, product processing time may be related to resource allocation, i.e., more resource allocation, less processing time. Thus, this paper attempts to solve *seru* scheduling problems with dynamic resource allocation, along with which the learning effect of workers is also considered. A combinatorial optimisation model is proposed to minimise the makespan, and a hybrid GA-PSO algorithm with nonlinear inertia weight is specifically designed to solve the proposed model. Finally, a numerical example is presented to verify the effectiveness of hybrid algorithm. The computational results indicate that hybrid GA-PSO algorithm is efficient, and dynamic resource allocation should be considered in *seru* scheduling problems.

[Submitted 1 February 2021; Accepted 30 April 2021]

Keywords: *seru* scheduling; GA-PSO; dynamic resource allocation; learning effect.

Reference to this paper should be made as follows: Shen, L., Zhang, Z., Song, X. and Yin, Y. (2023) 'A hybrid GA-PSO algorithm for *seru* scheduling problem with dynamic resource allocation', *Int. J. Manufacturing Research*, Vol. 18, No. 1, pp.100–124.

Biographical notes: Ling Shen is a Master candidate of School of Economics and Management, Nanjing University of Science and Technology. Her research interest is *seru* scheduling problem.

Zhe Zhang received her PhD from Sichuan University in December 2011. She is an Associate Professor of Nanjing University of Science and Technology. Her current research interests are in the areas of *seru* production systems, production scheduling, advanced manufacturing and so on.

Xiaoling Song obtained her PhD from the Sichuan University in 2016. She is an Associate Professor of Nanjing University of Science and Technology. Her current research interests are in the areas of complex uncertainty, layout of construction temporary facilities.

Yong Yin has graduated at Tohoku University in Japan. He is a Professor of Graduate School of Business in Doshisha University. His current research interests are in the areas of beyond lean, *seru* production systems, sustainable development, production and operations management and so on.

1 Introduction

Seru is an innovation production mode converted from the traditional Japanese assembly line, and it contains not only the efficiency of traditional assembly line but also the flexibility to cope with frequent product variety changes and product demand fluctuations (Lian et al., 2018; Stecke et al., 2012). In 1990s, Japan entered a period of economic downturn, and the manufacturing company began to seek new production way in order to meet the fluctuant market requirements. In this situation, *seru* was born in Sony in 1992. Sony dismantled a long assembly line and built several shorter assembly lines dedicated to one or more products. These short assembly lines are constantly shortened and innovative in layout, and eventually evolved into a work unit with one or more workers and a very small work space (Yu and Tang, 2019). *Seru* only includes a few workers and simple equipment to complete most or even all the work and it has the following characteristics: kanketsu (completion), majime (association) and jiritsu (autonomy) (Stecke et al., 2012). Generally, there are three types of *seru*: divisional *seru*, rotating *seru*, and *yatai* (Yu and Tang, 2019). In fact, *seru* has been widely praised and popularised because it can reduce lead time, work in process inventory, finished product inventory, cost, labour force and workshop space, and positively affect profit, product quality and labour incentive. For example, from 1999 to 2003, Canon dismantled a large number of traditional assembly lines and form many *seru*s. As a result, Canon saved about 7,200,000 square metres of workshop space, reduced 10,000 personnel, and reduced the average period of work in process from three days to six hours. At the same time, Canon paid attention to the cross training of worker's skills, and improved the production efficiency and enterprise benefits by improving the working efficiency of workers (D&M Nikkei Mechanical, 2003).

There are few studies on *seru* production system due to its short history and most of them are in Japanese. Fortunately, in recent years, with the practice of *seru* in the other countries and regions outside of Japan, *seru* has been favoured by academic circles at Japan and abroad. At present, the related research on *seru* is mainly about the *seru* formation and *seru* loading. Kaku et al. (2009) thought there are three aspects to the *seru* formation: how many *seru*s should be formatted, how many workers should be assigned in each *seru* and how many workers should be rested when a conveyor assembly line is converted into *seru*s. Liu et al. (2013) thought that the multi-skilled worker was the most important precondition in *seru* production system. They formulated a mathematical model to solve the training and assignment problem with multiple objectives which aim to minimise the total training cost and to balance the total processing times among multi-skilled workers in each *seru*. Yu et al. (2013) presented a multi-objective line-cell conversion model with the two goals of reducing worker(s) and increasing productivity simultaneously. They also identified several mathematical insights on solution space of the multi-objective line-cell conversion model and prove that it was an NP-hard problem (Yu et al., 2013). Later, Yu et al. (2014) developed a multi-objective optimisation model with the aim of minimising the total throughput time and the total labour hours to investigate the superiority of *seru* production system. They developed a NSGA-II based algorithm and verified the reliability of the algorithm by comparing with the enumeration method (Yu et al., 2014). Moreover, Yu et al. (2016) studied the scheduling rules of *seru* loading when the traditional assembly line is converted to *seru* and found that scheduling rules of shortest processing time was better than first come first service. Luo et al. (2016) considered a *seru* loading problem with worker-operation assignment, where the workers did not exist as limited resources, but fixed values can be fully allocated. Shao et al. (2016) designed a heuristic algorithm to solve the problem of *seru* formation and worker assignment by queuing theory to deal with the stochastic orders. Luo et al. (2017) considered a *seru* loading problem under uncertainty, and they proposed a fuzzy random bi-objective model with the objective of minimising the makespan and the totalt ardiness penalty cost. Yu et al. (2017) focused on investigating the fundamental principles of converting an assembly line to a hybrid system with *seru*s and a short line. They formulated several main models, clarified the complexity and analysed the properties of line-hybrid *seru* system conversion (Yu et al., 2017). Aboelfotoh et al. (2018) used the neural network to help enterprise decide whether to convert line to *seru*, and discussed the relationships among worker-task skill levels and the mean processing times, and the preferred manufacturing strategy (Abdel-Kader, 2018). Wu et al. (2018) studied the problem of cross-trained worker assignment in divisional and rotating *seru*, compared throughput performance between the two *seru* types. They constructed new mathematical models to solve the cross-trained worker assignment problem which can help the enterprise choose which suitable *seru* type to accommodate their production environment (Wu et al., 2018). Lian et al. (2018) solved the problem of worker grouping, *seru* loading, and task assignment simultaneously, and proposed a mathematical model with objectives of improving the inter-*seru* and inter-worker workload balance, and they came to the conclusion that the differences in worker's competency can impact the performance of *seru* production system. Yu et al. (2018) studied how to improve the balancing of *seru* production system, and discussed the fundamental principles of *seru* system balancing and presented an improved algorithm based on ε-constraint to find the optimal solution by cutting non-Pareto-optimal solutions. Zwierzyński and Ahmad (2018) made a review of the literature in the field of *seru* production, and they also considered that *seru* is better than traditional assembly line by a simulation experiment. Sun et al. (2019) developed a cooperative co-evolution algorithm with the aim of minimising makespan to solve the *seru* formation and *seru* scheduling simultaneously. Wang et al. (2019) discussed an order acceptance and scheduling problem considering lot-splitting, and proposed an improved genetic algorithm (GA) with the goal of maximising net profit to solve the problem. Yılmaz (2019) discussed a workforce scheduling problem in *seru* production system with two objectives, that of minimising makespan and reducing workload imbalance among workers. In this paper, we will also study the production scheduling problems in *seru* production system.

However, in the practical production process, processing time is not fixed but closely related to the resource allocation: when the number of additional resources is relatively large, it takes less processing time to complete the same number of products. Though, in other areas of research, there are some articles that consider resource constraints which could affect the job/product processing, and it is worth learning when study the *seru* scheduling problem. O'lafsson and Shi (2000) defined that the workers as a kind of limited renewable resource, and studied the scheduling in parallel manufacturing systems with flexible resources. Then, they presented a nested partitions method to solve the proposed scheduling problem (O'lafsson and Shi, 2000). Gao and Pan (2016) put forward that the scheduling problem not only the scheduling of machines but also other limited resource, and considered a multi-resource-constrained with the objective of minimising the makespan or completion time of all of the jobs. Afzalirad and Rezaeian (2016) solved an unrelated parallel machine scheduling problem with some restricted resources such as labours, tools, fixtures and proposed some methods: an exact approach and two meta-heuristic algorithms to find optimal or near optimal solutions. Fanjul-Peyro et al. (2017) took plant personnel as additional resources into account when they studied unrelated parallel machine scheduling problem, with the objective of minimisation of the makespan. Afzalirad and Shafipour (2018) considered other required resources like labours, tools in an unrelated parallel machine scheduling problem with the aim of minimising makespan. Costa et al. (2020) proposed a novel hybrid backtracking search optimisation algorithm, named BSATS to solve the scheduling problem in hybrid flow shop with limited human resource constraint and compared it with several evolutionary algorithm from the relevant literature. Sekkal and Belkaid (2020) took an identical parallel machine scheduling problem into account by minimising the makespan and resources cost. Also, they proposed a multi-objective simulated annealing (SA) algorithm and developed a two-steps algorithm by dividing the original problem into two sub problems to solve the problem (Sekkal and Belkaid, 2020). Watermeyer and Zimmermann (2020) considered the resource-constrained project scheduling with partially renewable resources and presented a branch-and-bound procedure for the problem with the objective to minimise the project duration. Thus, in this paper, we will focus on the scheduling problem in *seru* production systems with the dynamic resource allocation.

Besides, many scholars consider the change of processing time caused by learning effect when they study scheduling problems. Especially, *seru* is a manual production, learning effect can certainly be found. For example, the longer a multi-worker works, the more skilful the multi-worker get at this product and the shorter time it takes to process a single product. Mosheiov (2001) thought that learning effect is realistic and important and proved that the computational effort is greater for scheduling problem considering learning effect but polynomial-time solutions still exist. Okołowski and Gawiejnowicz (2010) used the general DeJong's learning effect curve:

$$p_{jr} = p_j \left(M + (1-M) r^{\alpha} \right)$$

to describe the impact of learning effect on processing time. The parameter M represents the part of job processing time that is limited by some conditions and cannot be shortened. They also developed two exact algorithms to solve the parallel-machine scheduling problem with the objective of minimising the makespan (Okołowski and Gawiejnowicz, 2010). Ji et al. (2015) discussed scheduling problem with DeJong's learning effect and proved the problem with the goal of minimising total completion time is polynomially solvable in both single machine and parallel machine production system. Lu et al. (2016) considered the resource allocation problem with learning effect and deteriorating jobs with the aim of minimising the cost function including the total load, the total completion time, the total absolute deviation of completion time and the total resource cost. They considered that the job processing-time is related to the worker's learning effect and the number of resources (Lu et al., 2016). Hosseinian et al. (2019) studied a multi-skilled resource-constrained project scheduling problem (RCPSP) with learning effect on job-processing time. They proposed that it is possible for workforces to improve their efficiencies by learning from more efficient staff members, thus the job-processing time is not fixed but changed. They developed an improved non-dominated sorting genetic algorithm II to solve the problem and validate the solutions obtained by the IM-NSGA-II by comparing it with other algorithms (Hosseinian et al., 2019). Wu et al. (2019) considered that the job-processing time decreases as the process proceeds, they compared four modified heuristics and three metaheuristics by simulation experiments and found that SA has a better performance on solving the scheduling problem with the goal of minimising total tardiness of the orders (Wu et al., 2019). Bai and Zhao (2020) considered a scheduling problem on single machine with DeJong's learning effect and presented a fully polynomial-time approximation scheme for it. They also tried to solve the problem with deteriorating jobs, DeJong's learning effect and an availability constraint with polynomial-time approximation scheme (Bai and Zhao, 2020). In these papers, many scholars discussed the scheduling problem considering learning effect in different scenarios. Thus, in this paper, we also adopt the DeJong's learning effect to simulate processing time in the actual *seru* production system. A summary of some related literature about *seru* production system, resource conditions and Dejong's learning effect has been provided in Table 1.

Motivated by the literatures mentioned above, we find that the resource allocation and learning effects are hardly considered in *seru* production scheduling problem. However, some scholars have begun to consider the impact of these two factors on production scheduling in other production systems. Because the *seru* production system is a human-centred production and learning effect is a common phenomenon in the manual production system, hence, it is natural to consider the learning effect in *seru* production scheduling. In the related literature of *seru* production system, some scholars have discussed the worker assignment but did not take it as limited resource of scheduling problem. We find that some articles about parallel machine production system clearly pointed out that workers can be considered as a kind of resource, and also proposed that jigs, fixtures, pallets, dies should be defined as constrained resources. Thus, in this paper we will first discuss a *seru* scheduling problems with dynamic resource allocation considering learning effect. We will take processing sequences and due date into account, and hope that it can guide *seru* production practice.

Seru scheduling problem is about that when the manufacturing company receives a set of product orders with different demands, it will choose a certain execute mode which contains a certain number of dynamic resources for each order when assigning the order to one exact *seru*. Besides, the learning effect will influence the processing time of a single product due to its processing position. By reasonably arranging the production system, determining which *seru* the order is assigned to and which execute mode selected for each order, it can help to improve the performance of the *seru* production system and minimise the makespan. Hence, a model with minimisation of the makespan based on the above requirements is established. In addition, the hybrid algorithm which combines the advantages of two different algorithms has better performance than the original algorithm

(Chen et al., 2020; Li et al., 2021; Liang et al., 2020; Luo et al., 2020; Sun et al., 2019). For example, Li et al. (2021) used a cooperative co-evolution differential evolution to choose the optimal feature subsets. Sun et al. (2019) developed a cooperative co-evolution algorithm to solve the *seru* formation and *seru* scheduling simultaneously. Khan et al. (2019) used hybrid particle swarm optimisation-genetic algorithm (PSO-GA) algorithm for traveling salesman problems. Hence, we will improve a new hybrid GA-PSO algorithm to solve the *seru* scheduling problems with dynamic resource allocation considering learning effect.

Table 1 Summary of the literature about *seru* production system, resource allocation and Dejong's learning effect

Reference	*Production system*	*Objective function(s)*	*Main characteristics of the content*
Luo et al. (2016)	*Seru*	Minimising the makespan	They considered the workers did not exist as limited resources, but fixed values can be fully allocated.
Wu et al. (2018)	*Seru*	Minimising the throughput of *seru* and workload balance of workers	They studied the problem of cross-trained worker assignment under considering skill levels (SLs) and several practical constraints in *seru* production system.
Lian et al. (2018)	*Seru*	The inter-*seru* and inter-worker workload balance	They found that homogenous workers can bring the high level of inter-worker workload balance whereas heterogenous workers with diversified competency perform well in balancing inter-*seru* workload.
Yılmaz (2019)	*Seru*	Minimising makespan and reducing workload imbalance among workers	They provided a novel optimisation model to compare two operational strategies by considering the heterogeneity inherent of workers.
O'lafsson and Shi (2000)	Parallel machine	Minimising the makespan	They defined that the workers as a kind of limited renewable resource.
Gao and Pan (2016)	Job Shop	Minimising the makespan	They took into account the expensive or limited resources in the actual production such as labour, maintenance equipment, and tooling.
Afzalirad and Rezaeian (2016)	Unrelated parallel machine	Minimising the makespan	They consider labours, tools, jigs, fixtures, pallets, dies and industrial robots as restricted resources, and proposed an exact approach and two meta-heuristic algorithms.
Fanjul-Peyro et al. (2017)	Unrelated parallel machine	Minimising the makespan	They defined the resources required for job processing is determined by itself and the characteristic of the machine.
Costa et al. (2020)	Hybrid flow shop	Minimising the makespan	They considered workforce as a critical resource and proposed a novel hybrid backtracking search optimisation algorithm.

Table 1 Summary of the literature about *seru* production system, resource allocation and Dejong's learning effect (continued)

Reference	*Production system*	*Objective function(s)*	*Main characteristics of the content*
Sekkal and Belkaid (2020)	Identical Parallel Machine	Minimising the makespan and resources cost	They studied a scheduling problem with deterioration effect and resources consumption constraints.
Watermeyer and Zimmermann (2020)	Project scheduling	Minimising the project duration	They used a branch-and-bound method to solve the project scheduling with partially renewable resources and general temporal constraints.
Mosheiov (2001)	Single machine, parallel machine	Minimising flow time, minimising the weighted sum of completion deviation and so on	He proved that the computational effort is greater for scheduling problem considering learning effect but polynomial-time solutions still exist.
Okołowski and Gawiejnowicz (2010)	Parallel Machine System	Minimising the makespan	They studied the scheduling problem with DeJong's learning effect.
Ji et al. (2015)	Single machine, parallel machine production system	Minimising total completion time	They proved that the scheduling problem considering DeJong's learning effect is polynomially solvable.
Lu et al. (2016)	Unrelated parallel machine	Optimal sequence of jobs and the optimal resource allocation for minimising the cost function	They proved that the scheduling problem with deteriorating jobs, resource allocation and learning effects is polynomial time solvable if the number of machines is a given constant.
Hosseinian et al. (2019)	Project scheduling	Minimising the makespan and total costs of project, simultaneously	They proposed that it is possible for workforces to improve their efficiencies by learning from more efficient staff members.
Wu et al. (2019)	Multiple machines	Minimising total tardiness of the orders	Wu considered that the job-processing time decreases as the process proceeds.

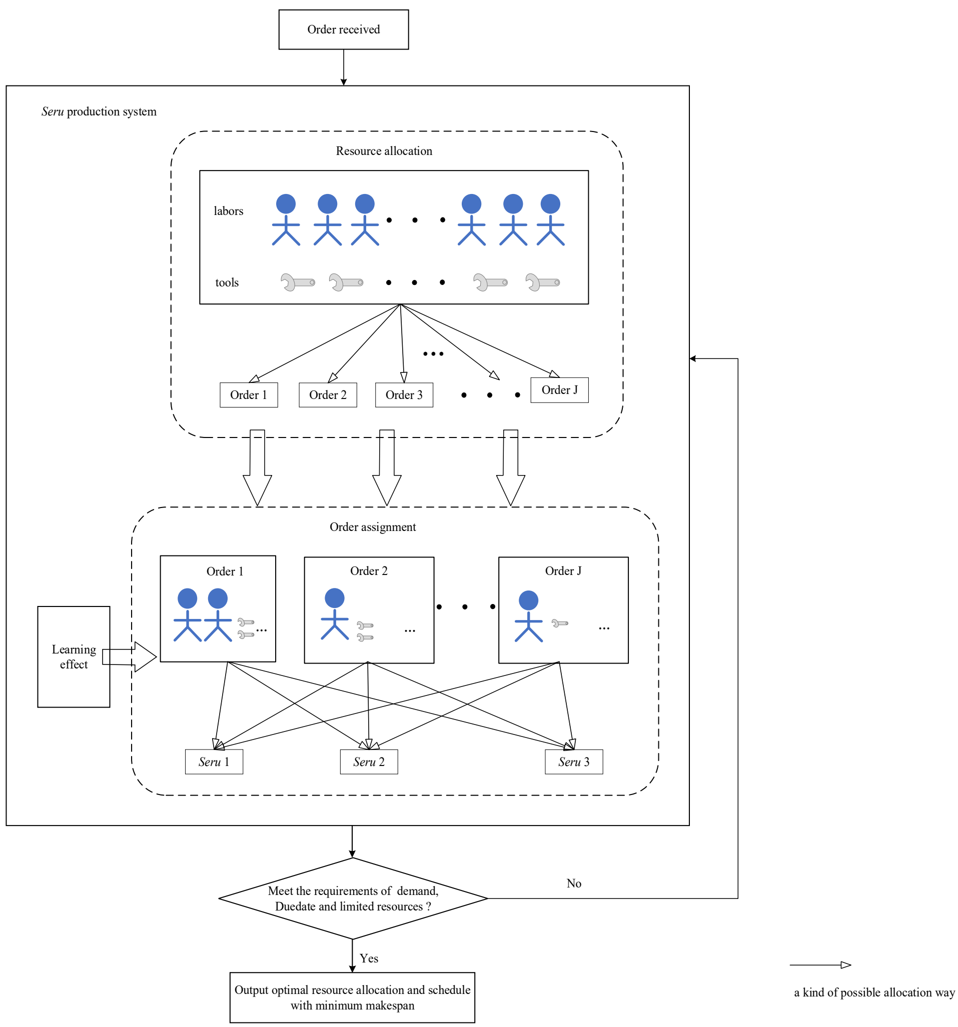
The paper is organised as follows. Section 2 describes the *seru* scheduling problem with dynamic resource allocation considering learning effect. Then, a hybrid GA-PSO algorithm is designed as the solution method in Section 3. In Section 4, computational experiments and analysis are presented. Section 5 summarises the paper and puts forward the future research.

2 Problem description and mathematical model

2.1 Problem description

Seru scheduling problem to be solved in this paper considers dynamic resource allocation with learning effect, which concentrates on dynamic resource allocation, assigning orders to *seru*s and determine the processing sequence to minimise the makespan. The decision process of *seru* scheduling problem is shown in Figure 1.

Figure 1 The flow chart of the decision process for *seru* scheduling problem (see online version for colours)



2.1.1 Assumption

1 The production system discussed in this paper is a pure *seru* production system.

2 The *seru* production system is constructed already, thus the reconfiguration is not considered.

3 Each order only can be assembled in one *seru*.

4 The available resource of *seru* production system is limited.

2.1.2 Notation

Indices

i *seru* index

j order index

m mode index

s position index

t time period index

k resource index

Parameters

D_j due date of order j

Q_j the quantity of product in order j

$\overline{p}_{jm}$ the normal processing time of each product of order j assembled in mode m

p_{jsm} the actual processing time of the product of order j at the s^{th} position in a sequence when assembled in mode m

p_{jm} the actual processing time of order j when assembled in mode m

p_j the actual processing time of order i

T the available production time if each *seru*

r_{jmk} the unit of resource k allocated to order j in mode m

R_k the total quantity of resource in *seru* production system.

Decision variables

$$x_{ij} = \begin{cases} 1, & \text{if order } j \text{ is allocated to } seru\ i \\ 0, & \text{otherwise} \end{cases}$$

$$z_{jmt} = \begin{cases} 1, & \text{order } j \text{ completes assembling in mode } m \text{ at time } t \\ 0, & \text{otherwise} \end{cases}$$

2.2 Modelling

The objective of the *seru* scheduling with dynamic resource allocation considering learning effect is minimising the makespan, which refers to the maximum completion time of all *seru*s in the *seru* production system, i.e.,

$$\min MS = \max_{\{i=1,2,\cdots I\}} C_i = \max_{\{i=1,2,\cdots I\}} \sum_{j=1}^{J} x_{ij} p_j \tag{1}$$

First, each order is assigned exactly to one *seru*.

$$\sum_{j=1}^{J} x_{ij} = 1 \tag{2}$$

Also, for each order, its execute mode should be selected, which means each order can only be assembled in one mode.

$$\sum_{m \in \{1,2,\cdots M\}} \sum_{t=1}^{T} z_{jmt} = 1 \tag{3}$$

Then, for each order j whose execute is determined, the amount of resource assigned to the order and its actual processing time are also determined:

$$p_j = \sum_{m \in \{1,2,\cdots,M\}} p_{jm} \sum_{t=1}^{T} z_{jmt} \tag{4}$$

Because of the learning effect, the total processing time of each order assembled in mode m is the sum of the actual processing time of every product in the k^{th} position in a sequence for s = 1, 2, 3, ···, Q_j (Q_j is the quantity of product in order j) of this order assembled in mode m.

$$p_{jm} = \sum_{s=1}^{Q_j} p_{jsm} \tag{5}$$

Besides, the quantity of each resource is limited. Therefore, the resource k consumed at each time period cannot exceed its available quantity.

$$\sum_{j=1}^{J} \sum_{m \in \{1,2,\cdots M\}} r_{jmk} \sum_{\Gamma=t}^{t+p_{jm}-1} z_{jm\Gamma} \le R_k \tag{6}$$

In addition, each order should be assembled before its due date.

$$FT_j = \sum_{t=1}^{T} t \sum_{m \in \{1,2,\cdots M\}} z_{jmt} \le D_j \tag{7}$$

Finally, the completion time of the whole *seru* production system should be earlier that its available production time

$$C_i = \sum_{j=1}^{J} x_{ij} p_j \leq T \tag{8}$$

To sum up, the mathematic model of the *seru* scheduling problem with dynamic resource allocation considering learning effect is as follows:

$$\begin{cases} \min \ MS = \max\limits_{\{i=1,2,\cdots I\}} \sum\limits_{j=1}^{J} x_{ij} p_j \\ s.t. \begin{cases} \sum\limits_{j=1}^{J} x_{ij} = 1 \\ \sum\limits_{m=1}^{M} \sum\limits_{t=1}^{T} z_{jmt} = 1 \\ p_{jm} = \sum\limits_{s=1}^{Q_j} p_{jsm} \\ p_j = \sum\limits_{m \in \{1,2,\cdots,M\}} p_{jm} \sum\limits_{t=1}^{T} z_{jmt} \\ \sum\limits_{j=1}^{J} \sum\limits_{m \in \{1,2,\cdots M\}} r_{jm} \sum\limits_{\Gamma=t}^{t+p_{jm}-1} z_{jm\Gamma} \leq R \\ FT_j = \sum\limits_{\mathrm{t}=1}^{T} t \sum\limits_{m \in \{1,2,\cdots M\}} z_{jmt} \leq D_j \\ C_i = \sum\limits_{j=1}^{J} x_{ij} p_j \leq T \end{cases} \end{cases} \tag{9}$$

3 Hybrid GA-PAO algorithm

In this paper, we adopt particle swarm optimisation (PSO) algorithm to tackle the *seru* scheduling problem with dynamic resource allocation considering learning effect. PSO is an evolution computing method which is population-based, many scholars have used it to solve scheduling problems and resource allocation problems. Zhang et al. (2005) used PSO to study the RCPSP, and got the good results by the algorithm. And they found that the permutation-based PSO outperformed the priority-based PSO (Zhang et al., 2005). Chen et al. (2013) discussed the parallel machine scheduling problem with the goal of minimising the makespan. They proposed a mixed integer linear programming model and developed a hybrid algorithm based on the variable neighbourhood search and PSO which is good at solving large size problems. Hulett et al. (2017) studied a scheduling problem on non-identical machines, they proposed a PSO algorithm and a heuristic algorithm to group the jobs into batches and schedule them. Compared with some other algorithms through some random instances, they found the PSO has a good performance in smaller problem instances and produced a better result in short time in larger problem instances (Hulett et al., 2017). Kasravi et al. (2019) proved the effectiveness of a hybrid approach by studying a resource-constrained railway project scheduling problem. The proposed ICA/PSO algorithm provided better solutions because of the weakness of PSO is improved by combination with imperialist competitive algorithm (Kasravi et al., 2019). Huang et al. (2019) studied a job shop scheduling problem, they improved the performance of PSO algorithm by combining PSO with SA algorithm and adding adaptive weight to improve inertia weight which could make the proposed algorithm dynamically adjust parameter factors according to fitness. Zhang et al. (2019) applied PSO algorithm and neural network to a job-shop scheduling problem. They proved that the proposed algorithm is good at solving large scale problems because the weakness of conventional PSO was eliminated by neural network (Zhang et al., 2019). Marichelvam et al. (2020) discussed a hybrid flow-shop scheduling problem considering human factors and developed an improved PSO algorithm to solve it. They proposed a mathematical model with the goal of minimising the weighted sum of the makespan and total flow time. Through a computational example, they proved the effectiveness of the proposed algorithm (Marichelvam et al., 2020). Ding and Gu (2020) proposed a hybrid PSO algorithm to solve the job-shop scheduling problem and verified its good performance by several FJSP instance. Therefore, we know that conventional PSO algorithm can get results in a short time, but it is likely to fall into the local optimal solution. Inspired by the related literature, we find that the performance of PSO algorithm can be improved by combining the GA introducing nonlinear inertia weight. Hachimi et al. (2012) improved the hybrid algorithm with the crossover and mutation operations of GA but also by mechanisms of PSO. They compared the proposed algorithm with the other algorithm, and found the GA-PSO algorithm has a good performance (Hachimi et al., 2012). Abdel-Kader (2018) used the selection and crossover operators of GA algorithm to evolve stagnated particles in the swarm to improve the performance of the proposed algorithm. They evaluate the hybrid algorithm by 123 benchmarks and came to a conclusion that it is better than other algorithms in solving the job shop scheduling problem (Abdel-Kader, 2018). Mahardhika (2021) used the selection, crossover, mutation operators of GA and velocity update mechanism of PSO to generate new individuals in the new hybrid algorithm, he chose 3 functions to compare GA, PSO and hybrid GA-PSO algorithm, and found that the hybrid algorithm has better results for optimisation problems. Yu et al. (2020) solved the job shop scheduling problem with the aim of minimising the makespan by an improved PSO algorithm. They add nonlinear inertia weight to PSO to improve local search ability and Gaussian mutation strategy to reduce probability of the algorithm falling into the local optimal solution (Yu et al., 2020). A summary of some related literature about PSO algorithm has been provided in Table 2.

Therefore, in this paper, we propose a hybrid PSO algorithm with GA. We try to use the crossover operator of GA to improve the global search ability of the hybrid algorithm, and use the mutation operator to search locally around the global optimal solution to improve the accuracy of the final optimal solution. Besides, we add nonlinear inertia weight to the hybrid algorithm to enhance the performance of the proposed algorithm. The specific steps of PSO algorithm and hybrid GA-PSO algorithm are described in the following paragraphs.

Table 2 Summary of the literature about PSO

Reference	*Production system*	*Objective function(s)*	*The proposed methodology*	*Main characteristics of the proposed methodology*
Zhang et al. (2005)	Resource-constrained project	Minimising the project duration time	A PSO algorithm	Design and compare the two PSO with different solution representations, i.e., priority-based representation and permutation-based representation
Chen et al. (2013)	Parallel machine	Minimising the makespan	A PSO algorithm with variable neighbourhood search	Two parts of the decision-making process: First, VNS is used to decide the processing sequence of orders. Second, the PSO is used to determine the machine allocation
Hulett et al. (2017)	Batch processing machines	Minimising the total weighted tardiness	A PSO algorithm	Some different job sequence rules
Kasravi et al. (2019)	Resource-constrained project	Minimising the project duration time	A hybrid algorithm based on particle swarm optimisation (PSO) and imperialist competitive algorithm (ICA)	The algorithm steps in consistent with the ICA steps
Huang et al. (2019)	Job shop	Minimising the machining time	An improved hybrid PSO algorithm	Utilising the probabilistic mutation capability of SA and adaptive inertia weight.
Zhang et al. (2019)	Job shop	Minimising the makespan	Particle swarm optimisation algorithm and particle and neural network	PSO is used to optimised the network weights
Marichelvam et al. (2020)	Hybrid flow shop	minimising of the weighted sum of the makespan and total flow time	An improved particle swarm optimisation with variable neighbourhood search	The variable neighbourhood search (VNS) algorithm used to attain the optimal solutions consuming less computational time.
Ding and Gu (2020)	Job shop	Minimising the maximum completion time	Hybrid of human learning optimisation algorithm and particle swarm optimisation algorithm	The individual learning ability of every particle and the search capacity of the proposed improved PSO are further promoted
Hachimi et al. (2012)	Some well-known benchmark test functions		A new hybrid genetic algorithm and particle swarm optimisation	The PSO is introduced to replace the mutation operator of GA
Abdel-Kader (2018)	Job shop	Minimising the makespan of the jobs	An improved PSO algorithm with genetic and neighbourhood-based diversity operators	The diversity enhancement operator is used to improve the population diversity and selection and crossover operators are introduced to evolve stagnated particles
Mahardhika (2021)		Three optimisation function problems	A hybrid algorithm that combined GA and PSO	The selection, crossover, mutation operators of GA and velocity update mechanism of PSO are used to generate new individuals
Yu et al. (xxxx)	Job shop	Minimising the makespan by arranging the processing scheduling sequence of all jobs	A hybrid particle swarm optimisation algorithm with nonlinear inertial weight and Gaussian mutation	Nonlinear inertia weight improves local search capabilities of PSO, while Gaussian mutation strategy improves the global search ability of NGPSO

The PSO algorithm is inspired by the cooperative behaviour of birds in a swarm (Kennedy and Eberhart, 1995). It is assumed that every feasible solution of the problem is a bird in the search space, which is what we usually call a particle. Each particle has its own position and velocity, which makes them follow the current optimal particle to search in the solution space. The formula for updating particle position and velocity is as follows:

$$v_{t+1} = \omega * v_t + c_1 * r_1 * \left(p^* - x_t\right) + c_2 * r_2 * \left(g^* - x_t\right) \tag{10}$$

$$x_{t+1} = x_t + v_{t+1} \tag{11}$$

where v_t and x_t represent the current velocity and position of particles in the iterative search; v_{t+1} and x_{t+1} represent the velocity and position of the updated particles; p^* represents the best position of the current particle; p^* represents the global best position that all particles in the swarm have reached so far. ω is the inertia weight; c_1, c_2 are the personal and social confidence coefficients; r_1, r_2 are random numbers generated between [0, 1].

To overcome the problem of premature convergence of conventional PSO algorithm, crossover operator and mutation operator of GA and nonlinear inertia weight would be introduced to improve the proposed algorithm in the next subsection.

3.1 Encoding of GA-PSO

The hybrid GA-PSO algorithm divides the particle code into three parts, which represent order assignment, processing sequence and execute mode. The first part represents order assignment. As shown in Figure 2, this encoding means that order 1, 7, 9, 10 are assigned to *seru* 1, order 2, 4, 8 are assigned to *seru* 2, order 3, 5, 6 are assigned to *seru* 3.

Figure 2 The coding of order assignment

1	2	3	2	3	3	1	2	1	1

The second part represents processing sequence. For instance, an example of processing sequence is given in Figure 3. The processing sequence of orders is 5, 2, 7, 4, 8, 6, 3, 9, 10.

Figure 3 The coding of processing sequence

9	2	7	4	1	6	3	5	8	10

The third part represents execute mode. In Figure 4, order 1, 10 are assembled in execute mode 1, order 2 is assembled in mode 2, order 5, 9 are assembled in mode 3, and order 3, 4, 6, 7, 8 are assembled in execute mode 4.

Figure 4 The coding of execute mode

1	2	4	4	3	4	4	4	3	1

3.2 Crossover operator improves the global search capability of PSO

Crossover operator could help to generate new solutions which improve the global search ability of GA. For this reason, we attempt to introduce the crossover operator in the process of updating to improve the global search ability of hybrid GA-PSO algorithm. The specific examples of crossover are shown in the following figures. The coding forms of order assignment and execute mode are the same, so we choose the same crossover way. As shown in Figures 5 and 6, two particles (P1,P2) are selected randomly, and several consecutive gene locations are randomly selected, then they are exchanged to generate two new particles (C1, C2).

Figure 5 The crossover of order assignment

P1:	1	2	3	2	3	3	1	2	1	1	⟶	C1:	3	1	2	1	3	3	1	2	1	1
P2:	3	1	2	1	1	2	2	3	1	3	⟶	C2:	1	2	3	2	1	2	2	3	1	3

Figure 6 The crossover of execute mode

P1:	1	2	4	4	3	4	4	4	3	1	⟶	C1:	4	4	4	4	1	4	4	4	3	1
P2:	4	4	4	4	1	4	1	1	1	4	⟶	C2:	1	2	4	4	3	4	1	1	1	4

Because of the characteristics of the processing sequence part, we choose the way of multipoint crossover and order crossover. First, two particles (P1, P2) are selected randomly and some gene locations are chosen randomly, then two new particles (C1, C2) are generated according to the principle that each number can only appear once time. The crossover of processing sequence is shown in Figure 7.

Figure 7 The crossover of processing sequence

P1:	9	2	7	4	1	6	3	5	8	10	⟶	C1:	4	9	6	2	7	1	3	5	8	10
P2:	10	4	9	5	7	8	2	6	1	3	⟶	C2:	5	8	10	4	9	7	2	6	1	3

3.3 Mutation operator improves the local search capability of PSO

Mutation operator is an assist operator which can improve the local search ability of PSO. Thus, for improving the accuracy of final optimal solution of the PSO algorithm, the mutation operator in GA is introduced to determine the final global best solution by local search near the current global best solution.

The mutation way of three parts of the particle is single point variation. Figures 8–10 show the mutation progress in order assignment, processing sequence and execute mode. First, two gene locations are chosen randomly in the current particle, then, the new particle is generated by exchanging the chose genes.

Figure 8 The mutation of order assignment

P1: → C1:

1	2	3	2	3	3	1	2	1	1	→	1	3	3	2	3	2	1	2	1	1

Figure 9 The mutation of processing sequence

P1: → C1:

9	2	7	4	1	6	3	5	8	10	→	9	6	7	4	1	2	3	5	8	10

Figure 10 The mutation of execute mode

P1: → C1:

1	2	4	4	3	4	4	4	3	1	→	1	4	4	4	3	2	4	4	3	1

3.4 Nonlinear inertia weight of hybrid GA-PSO

Inertia weight (ω) is the ability of particles to keep the state of the previous moment, which is particularly important in PSO. It indicates the influence of pre-particle velocity on current particle velocity, and determines the search ability of the algorithm (Yu et al., 2020). ω decreases linearly with the iteration progress in traditional PSO algorithm, which makes it difficult to jump out the neighbourhood of the local extreme point. In order to overcome the deficiency of linear inertia weight and improve the performance of the hybrid algorithm, it can be changed into nonlinear variable in different ways. In this paper, the relationship between inertia weight and iteration progress is defined as nonlinear correlation by introducing trigonometric function. As equation (12) shows, at the beginning of iteration progress, ω decreases slowly, which makes the algorithm have strong global search ability to find the optimal point, and avoid falling into the local optimum too early. At the end of iteration, ω changes quickly and converges to the global optimal value quickly after finding the local optimal value. And we also set the minimum value ($\omega_{\min}$) to make ω gradually get close to the minimum threshold, which can carry out more precise localisation search and improve the efficiency of operation.

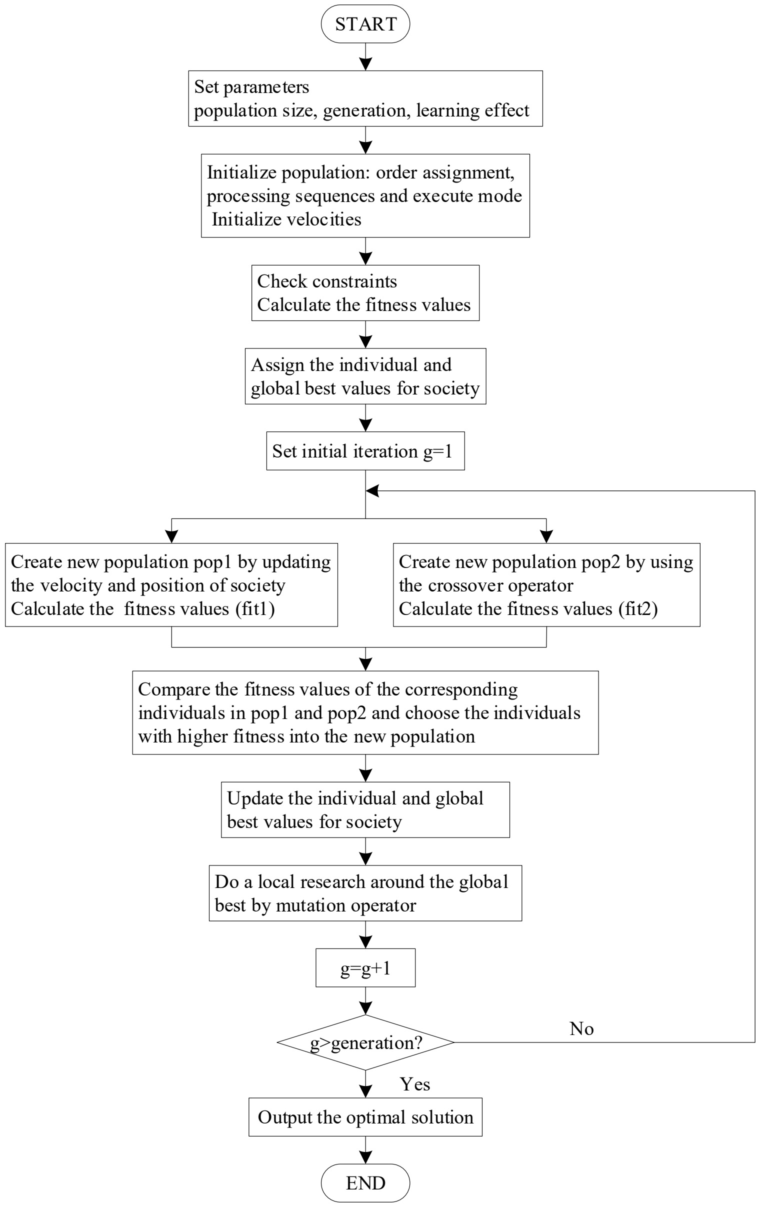
The ω update formula is as follows:

$$\omega = \omega_{\min} + \left(\omega_{\max} - \omega_{\min}\right) * \cos\left(pi * i \div (2 * T)\right) \tag{12}$$

where $\omega_{\min}$ is the minimum value of ω, $\omega_{\max}$ is the maximum value of ω, i is the current iteration, T is the maximum value of iterations. In this paper, $\omega_{\max} = 0.9$, $\omega_{\min} = 0.4$.

The overall process of GA-PSO algorithm is shown in Figure 11.

Figure 11 The overall process of GA-PSO algorithm



4 Computational experiments and analysis

4.1 Data collection

In order to evaluate the effectiveness of the GA-PSO algorithm, a randomly generated instance was made. The main parameters of the hybrid algorithm were configured as follows: swarm size $N = 600$, upper bound of inertia weight $\omega_{\max} = 0.9$, lower bound of inertia weight $\omega_{\min} = 0.4$, learning factors $c_1 = c_2 = 1.5$, maximum number of iterations $T = 200$. In this section, the discussed *seru* scheduling problem contains 3 *serus*, 10 orders, 2 kinds of resources, and 4 execute modes. We considered the available production time of the *seru* production system is 2,400 minutes. Tables 3 and 4 show some other related data. In Table 3, D_j represents the due date of order j, Q_j represents the quantity of order j, $\overline{p}_{j1}$ means the normal processing time of each product in order j assembled in mode 1, p_{j1} and means the normal processing time of all products in order j assembled in mode 1 without learning effect. In Table 4, R_1 and R_2 represent the detailed quantities of resource 1 and resource 2, $\overline{p}_{jm}$ is the actual processing time of the product assembled first in mode m of order j calculated according to equation (13), p_{jm} is the actual processing time of order j assembled in mode m considering learning effect calculated according to equation (14) referenced by Ji et al. (2015).

$$\overline{p}_{jm} = \overline{p}_{j1}\left(1-\left(1-\beta_1 \cdot \frac{1}{o_{m1}}\right)\right)\cdot\left(1-\left(1-\beta_2 \cdot \frac{1}{o_{m2}}\right)\right)\cdots\left(1-\left(1-\beta_k \cdot \frac{1}{o_{mk}}\right)\right) \tag{13}$$

where β_k is a constant to determine the acceleration achieved when resource k is available, and o_{mk} means the ratio of the quantity of resource k in execute mode m and execute mode1. In this paper, we considered the equal medium reductions in product processing time, thus set the $\beta_1 = \beta_2 = \cdots = \beta_k = 0.5$

$$p_{jm} = \sum_{s=1}^{Q_j} p_{jsm} = \sum_{s=1}^{Q_j} \overline{p}_{jm}\left[Z + (1-Z)s^{\alpha}\right] \tag{14}$$

where α is the common learning index, and $\alpha \le 0$. Z is the incompressible factor ($0 \le Z \le 1$). In this paper, we assume that the α of each order is different and random, $\alpha = (-1, -0.5, -0.2, -0.3, -0.6, -0.4, -0.8, -0.5, -0.6, -0.9)$, and $Z = 0.5$.

Comparing the normal product processing time and normal order completion time in Table 3 and the actual product processing time and actual order completion time in Table 4 by the formula (13) and formula (14), we find that with the advancement of the processing progress and the increase of resource quantity, the processing time of single product is reduced, and as a result, the completion time of the whole order is also reduced.

Table 3 Relevant data about due date, demand, and normal processing time

Order	D_j	Q_j	$\overline{p}_{j1}$	p_{j1}	*Order*	D_j	Q_j	$\overline{p}_{j1}$	p_{j1}
1	1,920	30	25	750	6	2,400	45	48	2,160
2	2,360	70	22	1,540	7	1,680	20	12	240
3	2,400	80	26	2,080	8	2,320	60	30	1,800
4	1,800	50	26	1,300	9	2,400	60	17	1,020
5	2,380	30	13	390	10	2,250	40	42	1,680

Table 4 Some related data about order and the actual product processing time

Order	*Mode*	R_1	R_2	$\overline{p}_{jm}$	p_{jm}	*Order*	*Mode*	R_1	R_2	$\overline{p}_{jm}$	p_{jm}
1	1	2	1	25	425	6	1	2	1	48	1,448
	2	4	1	19	323		2	4	1	36	1,086
	3	2	2	19	323		3	2	2	36	1,086
	4	4	2	14	238		4	4	2	27	815
2	1	2	1	22	939	7	1	2	1	12	148
	2	4	1	17	725		2	4	1	9	111
	3	2	2	17	725		3	2	2	9	111
	4	4	2	12	512		4	4	2	7	86
3	1	2	1	26	1,574	8	1	2	1	30	1,111
	2	4	1	20	1,211		2	4	1	23	852
	3	2	2	20	1,211		3	2	2	23	852
	4	4	2	15	908		4	4	2	17	630
4	1	2	1	26	927	9	1	2	1	17	603
	2	4	1	20	714		2	4	1	13	461
	3	2	2	20	714		3	2	2	13	461
	4	4	2	15	535		4	4	2	10	355
5	1	2	1	13	246	10	1	2	1	42	946
	2	4	1	10	189		2	4	1	32	721
	3	2	2	10	189		3	2	2	32	721
	4	4	2	7	132		4	4	2	24	541

4.2 Results and analysis

The computational results under the above premise are shown in Figure 12. It can be seen that the proposed algorithm reaches the state of convergence and obtain the optimal solution at about 30 iterations, fast and effectively. Figure 13 is the result of the above problem without considering dynamic resource allocation and learning effect by PSO algorithm. Compared Figure 12 with Figure 13, it shows that the hybrid GA-PSO algorithm has the characteristics of fast convergence as the conventional PSO algorithm. Besides, because of the crossover operator and mutation operator, its' ability of local search has been improved.

In Figure 13, the makespan is 4360 minutes, that is, the enterprise cannot complete all the orders within the available production time, but in Figure 12, through reasonable resource allocation, the enterprise finally completes the orders ahead of time. In addition, the actual makespan (1,873 minutes) is shorter than the expected by considering the learning effect, which conforms to the actual production situation.

Figure 14 illustrates the optimal solution of this instance. Orders 8, 5, 6, 1 are assembled in *seru* 1, order 4, 10 are assembled in *seru* 2, orders 7, 2, 9, 3 are assembled in *seru* 3. The number in brackets is the execute mode in which the order is assigned. The production time of 3 *seru*s are 1,872 minutes, 1,873 minutes and 1,861 minutes, respectively. Hence, the makespan of this *seru* production system is 1,873 minutes.

Figure 12 The makespan in each iteration (see online version for colours)

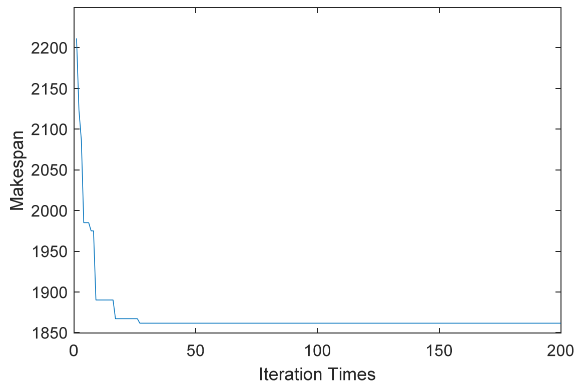


Figure 13 The makespan without considering resource allocation and learning effect (see online version for colours)

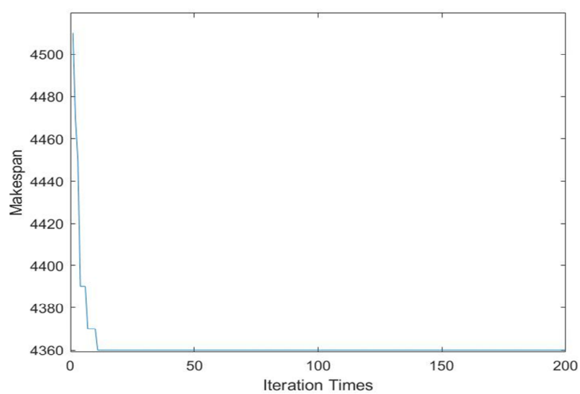
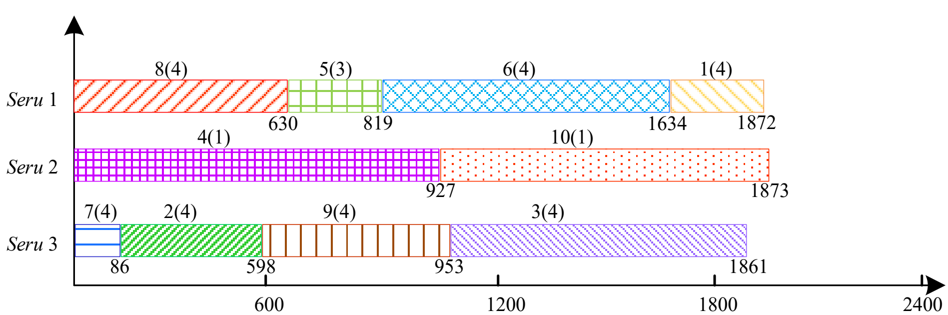


Figure 14 The optimal schedule (see online version for colours)



4.3 *Comparison with PSO and GA*

To better analyse the effective of the effectiveness of the hybrid GA-PSO algorithm, we designed a conventional PSO algorithm and GA to run the same numerical experiment.

As we can see, the makespan in Figure 12 is smaller, and the convergence speed is slower than Figure 15, but much faster than Figure 16. So, by comparing Figure 12, Figure 15 and Figure 16, it is concluded that the hybrid GA-PSO algorithm has better performance on solving *seru* scheduling problems.

Figure 15 The makespan and iteration of conventional PSO (see online version for colours)

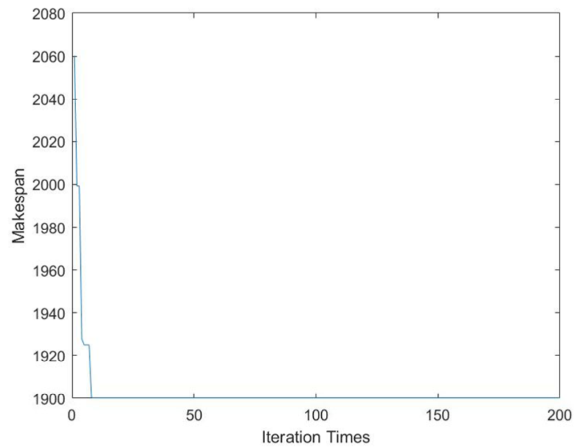


Figure 16 The makespan and iteration of GA (see online version for colours)

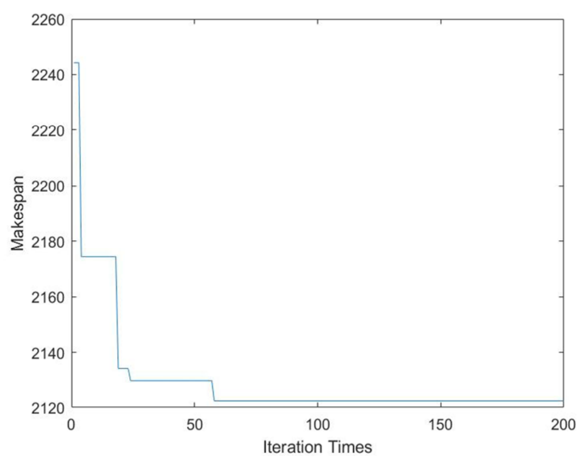


Table 5 shows the makespan and CPU time and of the hybrid GA-PSO algorithm, PSO and GA. According to these data, we can know that the makespan value of conventional PSO and GA are larger than the makespan of the proposed algorithm. Though the running time of conventional PSO and GA is less than GA-PAO, more accurate results can be obtained by dozens of seconds. Therefore, it can be concluded that the hybrid GA-PSO algorithm is better on *seru* scheduling problem with dynamic resource allocation considering learning effect than conventional PSO and GA.

Table 5 Results of hybrid GA-PSO, PSO and GA

No.	*Hybrid GA-PSO*		*PSO*		*GA*	
	Makespan (min)	*CPU time(s)*	*Makespan (min)*	*CPU time(s)*	*Makespan (min)*	*CPU time(s)*
1	1,873	76.457	1,900	37.275	2,122	39.729
2	1,918	95.26	1,956	41.854	2,105	44.173
3	1,896	87.996	2,087	38.872	2,044	38.328
4	1,936	84.768	1,994	37.148	2,105	37.47
5	1,910	72.432	2,224	38.665	2,054	38.412
6	1,932	74.404	2,064	37.036	2,060	38.241
7	1,948	78.98	2,050	44.052	2,063	37.674
8	1,867	75.758	2,258	43.285	2,157	37.816
9	1,910	77.178	2,223	36.998	2,074	37.961
10	1,893	75.697	2,144	37.198	2,194	38.508
Average	1,908.3	79.893	2,090	39.2383	2,097.8	38.8312
SD	25.1835	6.8270	115.04	2.6288	46.3331	1.8762

5 Conclusions

This paper has first studied the *seru* scheduling with dynamic resources and considering processing sequence and learning effect. A hybrid GA-PSO algorithm is designed to solve this complex combinatorial optimisation problem. The hybrid GA-PSO combines the crossover and mutation operators with the update mechanism of PSO, and improves the performance by introducing the nonlinear inertia weight. When compared to the conventional PSO and GA, the results show that the hybrid algorithm has a good performance. The proposed hybrid GA-PSO algorithm obtains the optimal value more accurately, and running results are more stable.

The result of the computational experiments indicates that the makespan changes with the quantity of additional resources and the learning effect affects the processing time of a single product which is closely related to the makespan of the *seru* production system.

Future research will focus on applying the proposed model and hybrid algorithm to various *seru* types considering different characteristics. In addition, considering the practical processing progress, *seru* scheduling problems under stochastic environment, such random processing time, resource quantity, etc., are also worth studying.

Acknowledgements

This research was sponsored by National Natural Science Foundation of China (Grant No. 71401075, 71801129), the Natural Science Foundation of Jiangsu Province (Grant No. BK20180452), and the Fundamental Research Funds for the Central Universities (Grant No. 30920010021). We would like to give our great appreciation to all the reviewers and editors who contributed this research.

References

Abdel-Kader, R.F. (2018) 'An improved PSO algorithm with genetic and neighborhood-based diversity operators for the job shop scheduling problem', *Applied Artificial Intelligence*, Vol. 32, No. 5, pp.433–462.

Aboelfotoh, A., Süer, G. and Abdullah, M. (2018) 'Selection of assembly systems; assembly lines vs. seru systems', *Procedia Computer Science*, Vol. 140, pp.351–358.

Afzalirad, M. and Rezaeian, J. (2016) 'Resource-constrained unrelated parallel machine scheduling problem with sequence dependent setup times, precedence constraints and machine eligibility restrictions', *Computers & Industrial Engineering*, August, Vol. 98, pp.40–52.

Afzalirad, M. and Shafipour, M.(2018) 'Design of an efficient genetic algorithm for resource-constrained unrelated parallel machine scheduling problem with machine eligibility restrictions', *Journal of Intelligent Manufacturing*, Vol. 29, No. 2, pp.423–437.

Bai, M.Z. and Zhao, Y.F. (2020) 'A fully polynomial-time approximation scheme for total completion time minimization on a single machine with DeJong's learning effect and an availability constraint', *Engineering Optimization*, Vol. 52, No. 8, pp.1313–1322.

Chen, Y., Cheng, C., Wang, L. and Chen T. (2013) 'A hybrid approach based on the variable neighborhood search and particle swarm optimization for parallel machine scheduling problems – a case study for solar cell industry', *International Journal of Production Economics*, Vol. 141, No. 1, pp.66–78.

Chen, Y., He, F., Li, H., Zhang, D. and Wu Y. (2020) 'A full migration BBO algorithm with enhanced population quality bounds for multimodal biomedical image registration', *Applied Soft Computing Journal*, August, Vol. 93, p.106335.

Costa, A., Fernandez-Viagas, V. and Framinan, J.M. (2020) 'Solving the hybrid flow shop scheduling problem with limited human resource constraint', *Computers & Industrial Engineering*, August, Vol. 146, p.106545.

D&M Nikkei Mechanical (2003) 'The challenge of Canon-Part 3', *Mechanical*, Vol. 588, No. 9, pp.70–73.

Ding, H. and Gu, X. (2020) 'Hybrid of human learning optimization algorithm and particle swarm optimization algorithm with scheduling strategies for the flexible job-shop scheduling problem', *Neurocomputing*, November, Vol. 414, pp.313–332.

Fanjul-Peyro, L., Perea, F. and Ruiz, R. (2017) 'Models and matheuristics for the unrelated parallel machine scheduling problem with additional resources', *European Journal of Operational Research*, Vol. 2016, No. 2, pp.482–493.

Gao, L. and Pan, Q. (2016) 'A shuffled multi-swarm micro-migrating birds optimizer for a multi-resource-constrained flexible job shop scheduling problem', *Information Sciences*, Vol. 372, No. 1, pp.655–676.

Hachimi, H., Ellaia, R. and Hami, A.E. (2012) 'A new hybrid genetic algorithm and particle swarm optimization', *Key Engineering Materials*, January, Vol. 498, pp.115–125.

Hosseinian, A.H., Baradaran, V. and Bashiri. M. (2019) 'Modeling of the time-dependent multi-skilled RCPSP considering learning effect', *Journal of Modelling in Management*, Vol. 14, No. 2, pp.521–558.

Huang, M., Liu, Q. and Liang, X. (2019) 'The application of improved hybrid particle swarm optimization algorithm in job shop scheduling problem', *2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT)*.

Hulett, M., Damodaran, P. and Amouie, M. (2017) 'Scheduling non-identical parallel batch processing machines to minimize total weighted tardiness using particle swarm optimization', *Computers & Industrial Engineering*, November, Vol. 113, pp.425–436.

Ji, M., Yao, D., Yang, Q. and Cheng, T. (2015) 'Machine scheduling with DeJong's learning effect', *Computers& Industrial Engineering*, February, Vol. 80, pp.195–200.

Kaku, I., Gong, J., Tang, J. and Yin, Y. (2009) 'Modelling and numerical analysis of line-cell conversion problems', *International Journal of Production Research*, Vol. 47, No. 8, pp.2055–2078.

Kasravi, M., Mahmoudi, A. and Feylizadeh, M. (2019) 'A novel algorithm for solving resource-constrained project scheduling problems: a case study', *Journal of Advances in Management Research*, Vol. 16, No. 2, pp.194–215.

Kennedy, J. and Eberhart, R. (1995) 'Particle swarm optimization', *Proceedings of IEEE International Conference on Neural Networks*, pp.1942–1948.

Khan, I., Pal, S. and Maiti, M.K. (2019) 'A hybrid PSO-GA algorithm for traveling salesman problems in different environments', *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, Vol. 27, No. 5, pp.693–717.

Li, H., He, F., Chen, Y. and Pan, Y. (2021) 'MLFS-CCDE: multi-objective large-scale feature selection by cooperative coevolutionary differential evolution', *Memetic Computing*, Vol. 3, No. 1, pp.1–18.

Lian, J., Liu, C., Li, W. and Yin, Y. (2018) 'Multi-skilled worker assignment in seru production systems considering worker heterogeneity', *Computers & Industrial Engineering*, April, Vol. 118, pp.366–382.

Liang, Y., He, F. and Zeng X. (2020) '3D mesh simplification with feature preservation based on whale optimization algorithm and differential evolution', *Integrated Computer-Aided Engineering*, Vol. 27, No. 4, pp.417–435.

Liu, C., Yang, N., Li, W., Lian, J., Evans, S. and Yin, Y. (2013) 'Training and assignment of multi-skilled workers for implementing seru production systems', *International Journal of Advanced Manufacturing Technology*, Vol. 69, Nos. 5–8, pp.937–959.

Lu, Y., Jin, J., Ji, J. and Wang, J. (2016) 'Resource-dependent scheduling with deteriorating jobs and learning effects on unrelated parallel machine', *Neural Computing and Applications*, Vol. 27, No. 7, pp.1993–2000.

Luo, J., He, F., Li, H., Zeng, X. and Liang Y. (2020) 'A novel whale optimization algorithm with filtering disturbance and non-linear step', *International Journal of Bio-Inspired Computation*, in press.

Luo, L., Zhang, Z. and Yin, Y. (2016) 'Seru loading with worker-operation assignment in single period', *IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*.

Luo, L., Zhang, Z. and Yin, Y. (2017) 'Modelling and numerical analysis of seru loading problem under uncertainty', *European Journal of Industrial Engineering*, Vol. 11, No. 2, pp.185–204.

Mahardhika, T. (2021) 'Hybrid algorithm as alternative method for optimization, a combination genetic algorithm and particle swarm optimization', *Journal of Physics: Conference Series*, Vol. 1764, No. 1, p.012040.

Marichelvam, M., Geetha, M. and Tosun, Ö. (2020) 'An improved particle swarm optimization algorithm to solve hybrid flowshop scheduling problems with the effect of human factors – a case study', *Computers & Operations Research*, February, Vol. 114, p.104812.

Mosheiov, G. (2001) 'Scheduling problems with a learning effect', *European Journal of Operational Research*, Vol. 132, No. 3, pp.687–693.

O'lafsson, S. and Shi, L. (2000) 'A method for scheduling in parallel manufacturing systems with flexible resources', *IEE Transactions*, Vol. 32, No. 2, pp.135–146.

Okołowski, D. and Gawiejnowicz, S. (2010) 'Exact and heuristic algorithms for parallel-machine scheduling with DeJong's learning effect', *Computers & Industrial Engineering*, Vol. 59, No. 2, pp.272–279.

Sekkal, N. and Belkaid, F. (2020) 'A multi-objective simulated annealing to solve an identical parallel machine scheduling problem with deterioration effect and resources consumption constraints', *Journal of Combinatorial Optimization*, Vol. 40, No. 3 pp.660–696.

Shao, L., Zhang, Z. and Yin, Y. (2016) 'A bi-objective combination optimization model for line-seru conversion based on queuing theory', *International Journal of Manufacturing Research*, Vol. 11, No. 4, pp.322–338.

Stecke, K.E., Yin, Y., Kaku, I. and Murase, Y. (2012) 'Seru: the organizational extension of JIT for a super-talent factory', *International Journal of Strategic Decision Sciences*, Vol. 3, No. 1, pp.106–119.

Sun, W., Wu, Y., Lou, Q. and Yu, Y. (2019) 'A cooperative coevolution algorithm for the seru production with minimizing makespan', *IEEE Access*, January, Vol. 7, pp.5662–5670.

Wang, L., Zhang, Z. and Yin, Y. (2019) 'Order acceptance and scheduling considering lot-spitting in seru production system', *IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pp.1305–1309.

Watermeyer, K. and Zimmermann, J. (2020) 'A branch-and-bound procedure for the resource-constrained project scheduling problem with partially renewable resources and general temporal constraints', *OR Spectrum*, Vol. 42, No. 2 pp.427–460.

Wu, C., Lin, W., and Zhang, X. et al. (2019) 'Tardiness minimisation for a customer order scheduling problem with sum-of-processing-time-based learning effect', *Journal of the Operational Research Society*, Vol.70, No.3, pp.487–501.

Wu, L., Chan, F.T.S., Niu, B. and Li, L. (2018) 'Cross-trained worker assignment and comparative analysis on throughput of divisional and rotating seru', *Industrial Management & Data Systems*, Vol. 118, No. 5, pp.1114–1136.

Yılmaz, Ö. (2019) 'Operational strategies for seru production system: a bi-objective optimisation model and solution methods', *International Journal of Production Research*, Vol. 58, No. 11, pp.3195–3219.

Yu, H., Gao, Y., Wang, L. and Meng, J. (2020) 'A hybrid particle swarm optimization algorithm enhanced with nonlinear inertial weight and Gaussian mutation for job shop scheduling problems', *Mathematics*, Vol. 8, No. 8, p.1355.

Yu, Y. and Tang, J.F. (2019) 'Review of seru production', *Frontiers of Engineering Management*, Vol. 6, No. 2, pp.183–192.

Yu, Y., Sun, W., Tang, J. and Wang, J.W. (2017) 'Line-hybrid seru system conversion: models, complexities, properties, solutions and insights', *Computers & Industrial Engineering*, January, Vol. 103, pp.282–299.

Yu, Y., Tang, J., Gong, J., Yin, Y. and Kaku, I. (2014) 'Mathematical analysis and solutions for multi-objective line-cell conversion problem', *European Journal of Operational Research*, Vol. 236, No. 2, pp.774–786.

Yu, Y., Tang, J., Sun, W., Yin, Y. and Kaku, I. (2013) 'Reducing worker(s) by converting assembly line into a pure cell system', *International Journal of Production Economics*, Vol. 145, No. 2, pp.799–806.

Yu, Y., Wang, J.W., Ma, K. and Sun, W. (2018) 'Seru system balancing: definition, formulation, and solution', *Computers & Industrial Engineering*, August, Vol. 122, pp.318–325.

Yu, Y., Wang, S.H., Tang, J.F., Kaku, I., Sun, W. (2016) 'Complexity of line-seru conversion for different scheduling rules and two improved exact algorithms for the multi-objective optimization', *SpringerPlus*, Vol. 5, No. 1, pp.1–26.

Zhang, H., Li, X. D., Li, H. and Huang, F.L. (2005) 'Particle swarm optimization-based schemes for resource-constrained project scheduling', *Automation in Construction*, Vol. 14, No. 3, pp.393–404.

Zhang, Z., Guan, Z.L., Zhang, J. and Xie, X. (2019) 'A novel job-shop scheduling strategy based on particle swarm optimization and neural network', *International Journal of Simulation Modeling*, Vol. 18, No. 4, pp.699–707.

Zwierzyński, P. and Ahmad, H. (2018) 'Seru production as an alternative to a traditional assembly line', *Engineering Management in Production and Services*, Vol. 10, No. 3, pp.62–69.