A bi-objective aggregate production planning problem with learning effect and machine deterioration: Modeling and solution

Article *in* Computers & Operations Research · March 2018

DOI: 10.1016/j.cor.2017.11.001

CITATIONS
12

READS
354

3 authors:

Esmaeil Mehdizadeh
Qazvin Islamic Azad University
62 PUBLICATIONS **443** CITATIONS

SEE PROFILE

Seyed Taghi Akhavan Niaki
Sharif University of Technology
537 PUBLICATIONS **6,976** CITATIONS

SEE PROFILE

Mojtaba Hemmati
Qazvin Islamic Azad University
16 PUBLICATIONS **188** CITATIONS

SEE PROFILE

A Bi-Objective Aggregate Production Planning Problem with Learning Effect and Machine Deterioration: Modeling and Solution

Esmaeil Mehdizadeh, Ph.D.
Faculty of Industrial and Mechanical Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran,
Phone: +98 28 33675784, E-mail: Emehdi@qiau.ac.ir

Seyed Taghi Akhavan Niaki, Ph.D.[1]
Sharif University of Technology, P.O. Box 11155-9414 Azadi Ave, Tehran 1458889694, Iran
Phone: +98 2166165740, E-mail: Niaki@Sharif.edu

Mojtaba Hemati, M.Sc.
Faculty of Industrial and Mechanical Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran,
Phone: +98 28 33670051, E-mail: M.hemmati84@yahoo.com

ABSTRACT

The learning effects of the workers and machine deterioration in an aggregate production planning (APP) problem have not been taken into account in the literature yet. These factors affect the performance of any real-world production system and require attention. In this paper, a bi-objective optimization model is developed for an APP problem with labor learning effect and machine deterioration. The first objective of this model is a quantitative function to maximize the profit by improving learning and reducing the failure cost of the system. The second objective is a qualitative function that aims to increase customer satisfaction. The aim of this article is to obtain appropriate levels of production rates in regular and overtimes, inventory and shortage levels, workers' hiring and firing levels, and the quantities of the products that are subcontracted. To demonstrate the validity of the proposed mathematical formulation, the multi-objective model is converted into a single-objective model using the fuzzy goal programming method, based on which computational experiments are performed on a set of random small-sized instances solved by the LINGO software. As the problem is shown NP-hard, a subpopulation genetic algorithm (SPGA) is proposed to solve large-size problems. In addition, two other meta-heuristics called weighted sum multi-objective genetic algorithm (WMOGA) and non-dominated sorting genetic algorithm II (NSGA-II) are utilized to solve a set of benchmark problems, in order to validate the results obtained and to assess the performance of the SPGA. For tuning the parameters, the Taguchi method is proposed in order to obtain high-quality solutions. Finally, the performances of the proposed algorithms are statistically compared together. The computational results show that SPGA compared to the other algorithms has a better performance in terms of some multi-objective optimization criteria.

Keywords: Aggregate production planning; bi-objective optimization; subpopulation genetic algorithm; learning effect; machine deterioration**.**

[1] Corresponding Author

1. Introduction

Aggregate production planning (APP) is an operational activity that provides an aggregate plan for production processes over a medium time range. The aim of APP is to set overall output levels in order to face with fluctuating or uncertain demands as well as to provide supply. It presents the optimal production quantity and production time of goods, parts, materials and other resources to minimize the total operational cost of the organization. In APP, the number of items to be outsourced, the amount of overtime labor, the numbers of workers to be hired and fired in each period, and the inventory level to be held in stock and to be backlogged for each period are determined. The primary required inputs for APP consists of information regarding the resources and the available facilities, demand forecast for the planning period, and cost of different alternatives and resources. The costs involve inventory holding, ordering, and production cost through various production alternatives such as subcontracting, backorders, and overtime. Organizational policies considering the utility of the above alternatives are also necessary.

The workers' learning, as well as machine deterioration, has a significant impact on how an aggregated production is planned. These factors affect the performance of any real-world production system and require attention. However, they have not been taken into account in the available models presented in the literature yet. That is why a bi-objective APP model is proposed in this paper to simultaneously consider the effects of labor learning and machine deterioration.

2. Literature review

As a technical level planning, APP attempts to determine the optimal quantity of production, inventory level, workforce, and etc., in each period with regard to some constraints to satisfy uncertain demands of all products (Mirzapour Al-e-Hashem, 2011). Based on the number of the objective functions considered in the models proposed in the literature, the APP models can be classified into two categories: (1) single objective function problems and (2) multi-objective function problems. A common objective function in the APP models is the minimization of the total cost of the system. In addition, maximization of service level, minimization of the changing of the workforce level, minimization of the risk, maximization of the profit, maximization of the customer satisfaction is other objective functions considered in the literature.

Over the last decades, numerous single-objective APP models have been studied in the literature, where various models have been developed to solve the APP problem. In a survey of models and methodologies for APP presented by Nam & Ogendar (1992), some researchers such as Bitran et al. (1982), Axsater (1986), and Ari & Axsater (1988) employed the hierarchical production planning (HPP) approach. Some other researchers such as Masud & Hwang (1980) and Tabucanon & Majumdar (1989) used a multi-criteria decision-making (MCDM) approach. Moreover, Techawiboonwong & Yenradee (2003) presented an APP for multiple product-types where the worker resource could be transferred to the production lines. They proposed a mathematical model in spreadsheet format, where the actual data were used to test and validate the results obtained using the proposed model. Hossain et al. (2016) used a GA approach for solving a real-world multi-product, multi-period aggregate production planning (APP) problem.

Hung & Hu (1998) solved an APP problem involving multiple products, multiple resources, multiple periods, setup times, and setup costs. If a particular product was to be manufactured in a specific period, then they assumed that each required machine should be set up for that product exactly once during the period. They formulated such production planning problem with setup decisions into the framework of a mixed-integer programming (MIP) problem. They proposed a heuristic algorithm to obtain near-optimum solutions. This algorithm iterates between linear programming solution phases and setup decision computations to solve the difficult MIP problem at hand.

While all the above-mentioned research discussed a single objective APP model which tried to minimize the total cost, other objective functions can also be considered in APP models. In other words, multiple objective functions that conflict each other can be considered in a practical APP model (Wang & Liang, 2004). Wang &Liang (2005) proposed a multi-objective APP model including three objective functions of minimizing the total costs, minimizing the carrying and back ordering costs, and minimizing the changing workforce level in a fuzzy environment. Leung & Chan (2009) developed a multi-objective APP model which attempts to maximize the profit, minimize the repairing costs, and maximize machine utilization regarding different operational constraints. Mirzapour Al-e-Hashem et al. (2012) addressed a multi-objective mixed-integer nonlinear programming model to deal with an APP considering two conflicting objectives as well as the uncertain nature of the supply chain. Sadeghi et al. (2013) implemented a goal programming approach to solve a three-objective optimization problem

developed for an APP. Their objectives included minimizing the total costs, minimizing carrying and back ordering costs, and minimizing the rate of changes in workforce level simultaneously. Entezaminia et al. (2016) proposed a multi-objective multi-period multi-product multi-site APP model in a green supply chain considering a reverse logistic (RL) network. In the model, minimizing the total losses and maximizing the total environmental scores of the products were the two objective functions. They demonstrated the trade-off between the conflicting objective functions by a set of Pareto-optimal solutions as generated by the LP-metrics method.

Based on the complexity involved in an APP problem that is usually modeled by a nonlinear mixed-integer programming model and due to NP-hardness of APP, using an exact or a hard-computing method is time–consuming, especially when the problem size increases (Fahimnia et al. 2006; Jiang et al. 2008; Partovi & Seifbarghy 2015). As such, some algorithms such as the hybrid algorithms proposed by Ganesh & Punniyamoorthy (2005) and Mohan Kumar & NoorulHaq (2005) and the Tabu search algorithm suggested by Baykasogluy(2006) and Pradenas & Pe~nailillo(2004) were implemented to solve APP. Ramezanian et al.(2012) presented a two-stage APP model with the goal of reducing the cost. They utilized a Tabu search and a genetic algorithm to solve their proposed mixed integer linear problem. Mirzapour et al.(2013) used a stochastic programming approach to solve a multi-period multi-product multi-site APP problem in a green supply chain for a medium-term planning horizon under the assumption of demand uncertainty. Their proposed model was a nonlinear mixed integer programming that was converted into a linear programming by employing some theoretical and numerical techniques. Wan & Yeh (2014) utilized a modified particle swarm optimization (MPSO) algorithm to solve an APP problem. This algorithm introduces the idea of sub-particles, a particular coding principle, and a modified operation procedure of particles to update rules in order to regulate the search processes for a particle swarm. They evaluated the performance of their MPSO with the ones of a standard PSO (SPSO) and a genetic algorithm (GA). Silva & Marins (2014) proposed a fuzzy goal programming model (FGP) for a real APP problem with a Brazilian sugar and ethanol milling company that takes into account uncertainty involved in APP. Mehdizadeh & Atashi-Abkenar (2014) developed a mixed integer linear programming (MILP) model for an integrated APP problem in a closed-loop supply chain with preventive maintenance. They implemented a genetic algorithm (GA), harmony search (HS) and vibration damping optimization (VDO) algorithm for solving the problem. Chakrabortty et al. (2015)

investigated the use of a particle swarm optimization (PSO) for an APP based on the potential environment. They offered a multi-product multi-period APP problem formulated as an integer linear programming. Modarres & Izadpanahi (2016) developed a robust optimization approach for an APP by focusing on energy saving to consider energy planning, demand, and production capacity simultaneously. That is why a meta-heuristic algorithm is proposed in the current paper to solve the APP problem with workers' learning effect and machine deterioration.

Among the meta-heuristics proposed to solve multi-objective APP problems, one can refer to the widely used Pareto-based non-dominated sorting genetic algorithm called NSGA-II, which is an extended version of the genetic algorithm (GA) proposed by Deb et al. (2002). This algorithm can be used in different scopes of operational management. In addition, the harmony search algorithm (HSA), as a music-inspired algorithm, is simple in concept and has just a few parameters. It is easy to be implemented and has been successfully applied to different problems including the mechanical structure design (Lee & Geem, 2004), pipe network optimization (Geem et al., 2002), and inventory models (Taleizadeh et al., 2011). Ramyar et al. (2017) presented a multi-objective model for a multi-product, multi-site APP problem of a supply chain. They utilized a Pareto-based multi-objective harmony search algorithm (MOHSA) to solve it. To demonstrate the performance of the presented algorithm, they used NSGA-II and NRGA to solve the problem as well.

A review of the literature on APP reveals that the learning effect of the workers has not been considered in the presented models yet. In addition, machine deterioration has not been taken into account. These factors affect the performance of any real-world production system and require attention. That is why a bi-objective APP model is proposed in this paper to simultaneously consider the effects of labor learning and machine deterioration. In other words, similar to the work in Hung & Hu (1998), a multiperiod multi-product multi-machine APP problem in a two-phase production system with setup costs and setup times is investigated that takes into account both the labor learning and machine deterioration. The first objective is a quantitative function to maximize the profit by improving labor learning as well as by reducing the failure cost of the system. The second objective is a qualitative function to increase customer satisfaction. The aim is to obtain appropriate levels of production rates in regular and overtimes, inventory and shortage levels, workers' hiring and firing levels, and the quantities of the products that are subcontracted. A subpopulation genetic algorithm (SPGA) is proposed to solve the bi-

objective APP problem. Due to the fact that the subpopulations are designed to explore specific areas, the solutions of Pareto-optimal are consistently spread over the border frontier. In order to distribute the solution uniformly, it seems really necessary to employ a uniform design method related to the scalarizing idea. In order to validate the results obtained and to assess the performance of the SPGA, two other meta-heuristics called weighted sum multi-objective genetic algorithm (WMOGA) and non-dominated sorting genetic algorithm II (NSGA-II) are utilized to solve a set of benchmark problems. To obtain better quality solutions, the parameters of all algorithms are calibrated using the Taguchi method. At the end, the performances of all the parameter-tuned solution algorithms are compared together statistically.

The rest of this paper is structured as follows: In Section 3, the bi-objective optimization model is developed. Section4 describes the proposed sub-population genetic algorithm (SPGA). Parameter tuning and computational results are provided in Section 5. Finally, conclusions are presented in Section 6.

3. Multi-objective optimization model

Multi-objective optimization problems (MOOP) are the ones where two or more objectives are optimized simultaneously. This kind of objective often conflict with each other and are inclusive of different units; i.e. some have to be minimized while others are to be maximized. Naturally, the final solution of MOOP is a set of solutions known as Pareto-solutions (Chankong & Haimes, 1983; Hans, 1988). The graph of such solutions produced in the objective function space is called the Pareto-front or the Pareto-optimal set. The MOOP consists of various objectives with several inequality and equality constraints. The general mathematical model of the MOOP problem is $Min\{f_1(x), f_2(x), ..., f_m(x)\}$ subject to $g_j(x) \le 0;\ j = 1, 2, ..., J$ with $h_k(x) = 0;\ k = 1, 2, ..., K$ where x is a vector of decision variables; $f_i(x)$ is the ith objective function and $g_j(x)$ and $h_k(x)$ are constraint vectors. The feasible region X is also limited based on the constraints, in which any given point $x \in X$ is considered a feasible solution. Not all the $f_i(x)$ function values have an optimum in X at a certain point x. Therefore, solutions to multi-objective problems are compared using the notion of Pareto dominance in the absence of preference information. In a minimization problem, a solution x_1 dominates a solution x_2 (also written as $x_1 \succ x_2$) for all objectives, if and only if the following conditions are realized:

- x_1 is no worse than x_2 in all the objectives, i.e. $f_i(x_1) \leq f_i(x_2); \forall i \in \{1, 2, ..., m\}$.
- x_1 is roughly better than x_2 for at least one objective, i.e. $f_i(x_1) < f_i(x_2); \exists i \in \{1, 2, ..., m\}$.

As mentioned above, a Pareto-optimal solution is not dominated by any other possible solution. Hence, the Pareto-front or Pareto-optimal set is constructed by the Pareto-optimal solutions of a multi-objective optimization problem.

3.1. A bi-objective optimization model for APP

This model is constructed based on two phases involved in a production system. In the first phase, the workers and the machines produce primary products. In the second phase, the workers and the machines assemble the products produced in the first phase along with the products purchased in order to provide finished products.

The assumptions involved in the proposed APP problem are as follows:

- Setup time is independent of the task sequence.
- The machines are available at any given time.
- Only one production manner is considered.
- Due to the learning effect, setup time is reduced when a more experienced worker does the setup.
- Due to machine deterioration, more production time and higher repair costs are required to satisfy the demands.
- The demands of the products in different periods are known constants

3.1.1. Decision variables

For a total number of N products, $k = 1,2,...,K$ products in the first phase, $i = 1,2,...,I$ products in the second phase, $l = 1,2,...,L$ machines in the first phase, $j = 1,2,...,J$ machines in the second phase, and $t = 1,2,...,T$ periods, the decision variables used in the mathematical model of the problem are:

P_{1kt} : Regular-time production quantity of product k in period t of the first phase

O_{1kt} : Overtime production quantity of product k in period t of the first phase

S_{1kt} : Subcontracting volume of product k in period t of the first phase (units)

I_{1kt} : The inventory of product k in period t of the first phase (units)

P_{2it} : Regular time production quantity of product i in period t of the second phase

O_{2it} : Over time production quantity of product i in period t of the second phase

S_{2it} : Subcontracting volume of product i in period t of the second phase (units)

I_{2it} : The inventory of product i in period t of the second phase (units)

B_{2it} : The backorder level of product i in period t of the second phase (units)

HH_t : The hiring level of the first labor group in period t (man-days)

LL_t : The layoff level of the first labor group in period t (man-days)

WW_t : The first labor group level in period t (man-days)

H_t : The hiring level of the second labor group in period t (man-days)

L_t : The layoff level of the second labor group in period t (man-days)

W_t : The second labor group level in period t (man-days)

C_{1k0} : Subcontracting volume of product k at the beginning of the planning horizon of the first phase (units)

Y_{1kt} : Setup time decision variable for product k in period t of the first phase (a binary integer variable)

Y_{2it} : Setup time decision variable for the product i in period t of the second phase (a binary integer variable).

3.1.2. Parameters

The parameters of the model are as follows:

D_{2it} : Demand of product i in period t of the second phase (units)

CP_{1kt} : Regular-time production cost of product k in period t of the first phase ($/units)

CO_{1kt} : Overtime production cost of product k in period t of the first phase ($/units)

CS_{1kt} : Subcontracting cost of product k in period t of the first phase($/ units)

CI_{1kt} : Inventory cost of product k in period t of the first phase ($/units)

CP_{2it} : Regular-time production cost of product i in period t of the second phase ($/units)

CO_{2it} : Over time production cost of product i in period t of the second phase ($/units)

CS_{2it} : Subcontracting cost of product i in period t of the second phase($/units)

SP_{2it} : Regular-time production price of product i in period t of the second phase ($/units)

SO_{2it} : Overtime production price of product i in period t of the second phase ($/units)

SS_{2it} : Subcontracting price of product i in period t of the second phase ($/units)

CI_{2it} : Inventory cost of product i in period t of the second phase ($/units)

CB_{2it}: Backorder cost of product i in period t of the second phase ($/ units)

a_{1kl} : The number of hours of machine l per unit of product k in the first phase (machine-days/unit)

a_{2ij}: The number of hours of machine j per unit of product i in the second phase (machine-days/unit)

u_{1kl} : The setup time for product k on machine l in the first phase (hours)

u_{2ij} : The setup time for product i on machine j in the second phase (hours)

r_{2ijt} : The setup cost of product i on machine j in period t of the second phase ($/machine-hours)

r_{1klt} : The setup cost of product k on machine l in period t of the first phase ($/machine-hours)

R_{jt} : The capacity of machine j in period t in the second phase (machine-hours)

RR_{lt} : The capacity of machine l in period t in the first phase (machine-hours)

CH_t : Cost to hire a labor group worker in period t in the second phase ($/man-days)

CHH_t : Cost to hire a labor group worker in period t in the first phase ($/man-days)

CL_t : Cost to lay off a labor group worker in period t in the second phase ($/man-days)

CLL_t : Cost to lay off a second labor group worker in period t in the first phase ($/man-days)

CW_t : The first labor group cost in period t in the second phase ($/man-days)

CWW_t : The second labor group cost in period t in the first phase ($/man-days)

I_{1k0}: The initial inventory of product k in period t of the first phase (units)

I_{2i0} : The initial inventory of product i in period t of the second phase (units)

W_0 : The initial workforce level in the second phase (man-days)

WW_0 : The initial workforce level in the first phase (man-days)

B_{2i0} : The initial backorder level of product i in period t of the second phase (units)

f_{ik} : The number of units of the first-phase product k required per unit of first-phase product i

e_{1k} : Labor hours required per unit of product k in the first phase (man-days/unit)

e_{2i} : Labor hours required per unit of product i in the second phase (man-days/unit)

Z_{1kt} : Labor cost of product k in period t of the first phase ($)

Z_{2it} : Labor cost of product i in period t of the second phase ($)

α_{1t} : The ratio of the regular-time of the first labor group available for use in overtime t

α_{2t} : The ratio of the regular-time of the second labor group available for use in overtime t

β_{jt} : The ratio of the regular-time capacity of machine j available for use in overtime t

β_{2lt} : The ratio of the regular-time capacity of machine l available for use in overtime t

f : The working hours of the labor per period (man-hour/man-day)

$W_{t\max}$:Maximum level of the first labor group available in period t in the first phase (man-days)

$WW_{t\max}$: Maximum level of the labor group available in period t in the second phase (man-days)

$S_{1kt\max}$: Maximum subcontracted volume available for product k in period t of the first phase (units)

$S_{2it\max}$: Maximum subcontracted volume available for product i in period t of the second phase (units)

Q_{1kl} : Maintenance cost of machine l to produce product k in the first phase

Q_{2ij} : Maintenance cost of machine j to produce product i in the second phase

q_{1kl} : Failure rate of machine l to produce product k in the first phase

q_{2ij} : Failure rate of machine j to produce product i in the second phase

br : The learning coefficient

lid :Lead time to deliver purchased goods (days)

M : A large number.

3.2. The proposed model

As previously mentioned, a bi-objective optimization model is developed in this paper for an aggregate production planning (APP) problem considering the labor learning effect and machine deterioration. The first objective function maximizes the profit by improving learning and reducing the failure cost of the system. The second objective function aims to increase the customer satisfaction. In Sections 2.2.1 and 2.2.2, these two objective functions are formulated based on the decision variables and the parameters defined in Section 2.1.

3.2.1. The first objective function

The first part of the quantitative objective function involves revenue obtained by selling produced and subcontracted products as

$$\sum_{i=1}^{N}\sum_{t=1}^{T}\left(SP_{2it}\times P_{2it}+SO_{2it}\times O_{2it}+SS_{2it}\times S_{2it}\right)$$

The total production and subcontracting costs of the first-phase and the second-phase products are:

$$\sum_{i=1}^{I}\sum_{t=1}^{T}\left(CP_{2it}\times P_{2it}+CO_{2it}\times O_{2it}+CS_{2it}\times S_{2it}\right)+\sum_{k=1}^{K}\sum_{t=1}^{T}\left(CP_{1kt}\times P_{1kt}+CO_{1kt}\times O_{1kt}+CS_{1kt}\times S_{1kt}\right)$$

The production cost corresponding to the learning effect for the first and the second-phase products is:

$$\sum_{i=1}^{I}\sum_{t=1}^{T}Z_{2it}e_{2it}\left(\left(\sum_{t=1}^{T}P_{2i(t-1)}\right)+1\right)^{br}\times P_{2it}^{(br+1)}+\sum_{k=1}^{K}\sum_{t=1}^{T}Z_{1kt}e_{1k}\left(\left(\sum_{t=1}^{T}P_{1k(t-1)}\right)+1\right)^{br}\times P_{1kt}^{(br+1)}$$

The setup cost for the first and the second-phase products on their corresponding machines is:

$$\sum_{l=1}^{L}\sum_{t=1}^{T}\sum_{k=1}^{K}r_{1klt}\times Y_{1kt}+\sum_{j=1}^{J}\sum_{t=1}^{T}\sum_{i=1}^{I}r_{2ljt}\times Y_{2it}$$

The inventory cost of the first-phase and the second-phase products are:

$$\sum_{t=1}^{T}\sum_{k=1}^{K}I_{1kt}\times CI_{1kt}+\sum_{t=1}^{T}\sum_{i=1}^{I}I_{2it}\times CI_{2it}$$

The backorder cost of the second-phase products is:

$$\sum_{t=1}^{T}\sum_{i=1}^{I}CB_{2it}\times B_{2it}$$

The total workforce, hiring, and layoff cost is obtained as:

$$\sum_{t=1}^{T} CW_t \times W_t + \sum_{t=1}^{T} CWW_t \times WW_t + \sum_{t=1}^{T} (CH_t \times H_t) + (CL_t \times L_t) + \sum_{t=1}^{T} (CHH_t \times HH_t) + (CLL_t \times LL_t)$$

Thus, the first objective function that aims to maximize the total profit is as follows:

$$\begin{aligned} Max\ OBJ_1 = & \sum_{i=1}^{N}\sum_{t=1}^{T}(SP_{2it} \times P_{2it} + SO_{2it} \times O_{2it} + SS_{2it} \times S_{2it}) - \sum_{i=1}^{I}\sum_{t=1}^{T}(CP_{2it} \times P_{2it} + CO_{2it} \times O_{2it} + CS_{2it} \times S_{2it}) \\ & - \sum_{k=1}^{K}\sum_{t=1}^{T}(CP_{1kt} \times P_{1kt} + CO_{1kt} \times O_{1kt} + CS_{1kt} \times S_{1kt}) - \sum_{i=1}^{I}\sum_{t=1}^{T} Z_{2it} e_{2it} \left(\left(\sum_{t=1}^{T} P_{2i(t-1)} \right) + 1 \right)^{br} \times P_{2it}^{(br+1)} \\ & - \sum_{k=1}^{K}\sum_{t=1}^{T} Z_{1kt} e_{1k} \left(\left(\sum_{t=1}^{T} P_{1k(t-1)} \right) + 1 \right)^{br} \times P_{1kt}^{(br+1)} - \sum_{l=1}^{L}\sum_{t=1}^{T}\sum_{k=1}^{K} r_{1klt} \times Y_{1kt} - \sum_{j=1}^{J}\sum_{t=1}^{T}\sum_{i=1}^{I} r_{2ljt} \times Y_{2it} \\ & - \sum_{t=1}^{T}\sum_{k=1}^{K} I_{1kt} \times CI_{1kt} - \sum_{t=1}^{T}\sum_{i=1}^{I} I_{2it} \times CI_{2it} - \sum_{t=1}^{T}\sum_{i=1}^{I} CB_{2it} \times B_{2it} - \sum_{t=1}^{T} CW_t \times W_t - \sum_{t=1}^{T} CWW_t \times WW_t \\ & - \sum_{t=1}^{T} (CH_t \times H_t) - (CL_t \times L_t) - \sum_{t=1}^{T} (CHH_t \times HH_t) - (CLL_t \times LL_t) \end{aligned} \quad (1)$$

3.2.2. The second objective function

In the second objective function, the costs associated with repairs and deterioration, which depend on the failure rate of the machines in the production periods of the first and the second phase, are minimized. Hence, the failure rate is increased in successive periods. In other words, we have:

$$Min\ OBJ_2 = \sum_{t=1}^{T}\sum_{i=1}^{N}\sum_{j=1}^{J} \sqrt[t]{\frac{q_{2ij}}{t}} \times Q_{2ij} \times P_{2ij} + \sum_{t=1}^{T}\sum_{k=1}^{K}\sum_{l=1}^{L} \sqrt[t]{\frac{q_{2ij}}{t}} \times Q_{1kl} \times P_{1kl} \quad (2)$$

Note that as fewer repairs may lead to more on-time deliveries, this objective function is indirectly related to customer satisfaction.

3.2.3. The constraints

The constraints are as follows:

$$P_{2it} + O_{2it} + S_{2it} + B_{2it} - B_{2i(t-1)} + I_{2i(t-1)} - I_{2it} = D_{2it} \qquad \forall i,t \quad (3)$$

$$P_{1kt} + O_{1kt} + S_{1kt} + I_{1k(t-1)} - I_{1kt} = \sum_{i=1}^{I} f_{ik} \times (P_{2i(t+lid)} + O_{2i(t+lid)}) \qquad \forall k,t \quad (4)$$

$$S_{0k} + I_{0k} - I_{2i0} = \sum_{i=1}^{I}\sum_{t=1}^{lid} f_{ik} \times (P_{2i(t+lid)} + O_{2i(t+lid)}) \qquad \forall k \quad (5)$$

$$\sum_{i=1}^{I} (A_{2ij} \times P_{2it} + u_{2ij} \times Y_{2it}) \le R_{jt} \qquad \forall j,t \quad (6)$$

$$\sum_{i=1}^{I} (A_{2ij} \times O_{2it}) \le \beta_{jt} \times R_{jt} \qquad \forall j,t \quad (7)$$

$$\sum_{k=1}^{K}(A_{1kl}\times P_{1kt}+u_{1kt}\times Y_{1kt})\le RR_{lt} \qquad \forall l,t \qquad (8)$$

$$\sum_{k=1}^{K}(A_{1kl}\times O_{1kl})\le \beta_{2lt}\times RR_{lt} \qquad \forall l,t \qquad (9)$$

$$P_{1kt}+O_{1kt}\le M.Y_{1kt} \qquad \forall k,t \qquad (10)$$

$$P_{2it}+O_{1it}\le M.Y_{2kt} \qquad \forall i,t \qquad (11)$$

$$W_t=W_{(t-1)}+H_t-L_t \qquad \forall t \qquad (12)$$

$$WW_t=WW_{(t-1)}+HH_t-LL_t \qquad \forall t \qquad (13)$$

$$\sum_{k=1}^{K}e_{1k}\left(\left(\sum_{t=1}^{T}P_{1k(t-1)}+1\right)^{br}\right)\times\left(P_{1kt}{}^{(br+1)}\right)\le f\times WW_t \qquad \forall t \qquad (14)$$

$$\sum_{k=1}^{K}e_{2k}\times O_{1kt}\le \alpha_{1t}\times f\times WW_t \qquad \forall t \qquad (15)$$

$$\sum_{i=1}^{I}e_{2i}\left(\left(\sum_{t=1}^{T}P_{2i(t-1)}+1\right)^{br}\right)\times\left(P_{2it}{}^{(br+1)}\right)\le f\times W_t \qquad \forall t \qquad (16)$$

$$\sum_{i=1}^{I}e_{2i}\times O_{2it}\le \alpha_{2t}\times f\times W_t \qquad \forall t \qquad (17)$$

$$W_t\le W_{\max t} \qquad \forall t \qquad (18)$$

$$WW_t\le WW_{\max t} \qquad \forall t \qquad (19)$$

$$S_{2it}\le S_{2it\max} \qquad \forall i,t \qquad (20)$$

$$B_{2it}\times I_{2it}=0 \qquad \forall i,t \qquad (21)$$

$$Y_{2it}=\{0,1\} \qquad \forall i,t \qquad (22)$$

$$Y_{1kt}=\{0,1\} \qquad \forall k,t \qquad (23)$$

Constraint (3) satisfies the market demands of the second-phase products. The production and subcontract of the first-phase products associated with the total production of the second-phase products are provided in Constraint (4). Constraint (5) assures that the initial inventory together with the subcontracting volume of the first-phase products at the beginning of the planning horizon should not be less than the total production of the second-phase products at the ordering time to satisfy the market demand. Constraint (6) ensures that the number of the products produced at regular times of the second phase does not exceed the available machine capacity, where setup times are included in machine capacity. Constraint (7) relates to the machine capacity in over time. Constraints (8) and (9) represent the machine capacity in the first phase. Note that the machines in the first and in the second phase are used to produce the products of the first phase and the second phase, respectively. Constraint (10) shows the limitation on the total quantity produced in regular and overtime period of the first phase. Constraint (11) is used for the second-phase products. Constraints (12) and (13) represent the

workforce levels of labor groups in the first and in the second phases. Workforce capacity at regular and overtime of each period for both labor groups taking into account the learning effect is limited in Constraints (14)–(17).The workforce of the first and the second phases are limited in Constraints (18) and (19), respectively. Constraint (20) is the limitation on the subcontracting volume of the products in each period of the second phase. Constraint (21) ensures that shortage and inventory are not possible simultaneously. Finally, Constraints (22) and (23) are the binary variables for machinery setup.It should be mentioned that all the variables of the model are non-negative.

3.2.4. Model validation

To demonstrate the validity of the proposed mathematical formulation, first, the multi-objective model is converted into a single-objective model using the fuzzy goal programming method. To do this, an auxiliary variable $(\mu_i ; i = 1,2)$ between zero and one is defined based on the prioritization of the decision maker for each objective to represent importance realization of the goal (the objective). Then, a secondary objective function is defined such that the goal becomes maximization of the weighted sum of both auxiliary variables. Note that while the weights are non-negative, their sum is equal to one. Finally, the membership inequalities are added to the single-objective maximization model shown in (24), where L_i is the lower bound, U_i is the upper bound, and G_i is the expected value, all for the ith objective specified by the manager.

$$
\begin{aligned}
&Max\left(w_1 \times \mu_1 + w_2 \times \mu_2\right) \\
&s.t: \\
&\mu_1 \geq \frac{OBJ_1 - L_1}{G_1 - L_1} \\
&\mu_2 \leq \frac{U_2 - OBJ_2}{U_2 - G_2} \\
&0 \leq \mu_1 \leq 1 \\
&0 \leq \mu_2 \leq 1 \\
&0 \leq w_1 \leq 1 \\
&0 \leq w_2 \leq 1 \\
&w_1 + w_2 = 1 \\
&\text{and the Constraints } (3) - (23)
\end{aligned}
\qquad (24)
$$

Table 1 contains the results obtained by solving the first 15 randomly generated problem instances shown in Table A1 of Appendix A when modeled using (24) and solved by the Lingo software.

Please insert Table 1 about here

The results in Table 1 clearly show that the proposed APP formulation is quite valid, as reasonable objective function values are obtained. Note that as the problem becomes larger, more computational time is required.

Due to NP-hardness of APP (Fahimnia et al. 2006), a meta-heuristic algorithm called sub-population genetic algorithm (SPGA) is proposed in Section 3 to solve the mixed integer linear programming (MILP) model developed above for a general two-phase aggregate production planning problem.

4. The proposed sub-population genetic algorithm (SPGA)

In addition to some sub-population approaches such as segregate genetic algorithm (Affenzeller, 2002), multi-population genetic algorithm (Cochran et al., 2003), hierarchical fair competition model (Goodman et al., 2005), and the two-phase sub-population GA (Chang et al., 2005) found in the literature, Chang et al. (2006) proposed a modified SPGA (SPGA-II) to solve parallel machine scheduling problems with minimization of the objectives such as makespan and tardiness. SPGA is mostly applied in cases where loss of diversity may result in premature solutions (Simoes & Costa, 2002). That is why a three-phase sub-population genetic algorithm is proposed in the current study to prevent the searching procedure from being trapped into a local optimal. The three major features of this SPGA include: (1) the original population is divided into many small sub-populations to check the solution space, in which each subpopulation is independent of or unrelated to each other; (2) the two objective functions are scalarised into a single objective, where each subpopulation is assigned a different weight. This ensures different searching directions. Moreover, an outside storage file is set up to record non-dominated solutions appeared during the searching process; (3) the substitution process is converged after a certain number of iterations where the selected sub-populations will replace the original populations if only they are in a higher position than the original ones.

Pareto-optimal solutions are consistently spread over the border frontier since the aim of subpopulations is exploring specific areas. For distributing the solution uniformly, it is required to employ a uniform design method regarding the scalarizing idea (Leung & Wang, 2000). However, in order to have results with better convergence and diversity, it is required that sub-populations establish a permanent relationship with each other. Therefore, this research considers the way of exchanging the information of these subpopulations by exploring different solution spaces. The proposed approach for solving combinatorial optimization problems is based on the concept of the global Pareto archives. However, the connection among the subpopulations of the proposed SPGA gives an opportunity that if a subpopulation can find a better solution; other subpopulations can also adapt it to enhance the quality of their solutions in that specific searching area, although the original pattern of SPGA does not permit Pareto archives to share their own information with each other. Similar to SPGA-II purposed by Changet al. (2006) and Behnamian et al. (2009), in this paper we adopt a 'preference-based method', the 'Pareto approach', and the ε-constraint method' to initiate from Phase 1 and to improve in Phase 2 with the intention of obtaining approximately efficient solutions.

The differences between the proposed SPGA and the available SPGA are as follows: (1) the main population in the initial phase is divided into several subpopulations, in which each subpopulation can communicate with the other using the same Pareto set assigned as a global archive. In this phase, a combination of regular weighting and the Min-Max method is used to reach a variety of appropriate Pareto solutions. (2) In the second phase, non-dominated solutions are unified as a big population to improve the Pareto-front. The primary quality of the Pareto solutions acquired from the first phase increases using the Pareto optimization and Non-dominated sorting method utilized in the selection strategy. The pseudo-code of the proposed SPGA is illustrated in Figure 1.

Please insert Figure 1 about here

4.1. Representation

A chromosome is provided by an integer $I \times T$ matrix shown in Figure 2, where I represents the number of products in the second phase and T is the number of periods. A gene in this matrix represents the total aggregate production quantity (TP) of a product obtained in a

given period of the second phase in regular-time production, overtime production, and subcontracting.

Please insert Figure 2 about here

4.2. Calculation of the production, subcontracting, hiring, and layoff variables

In what follows, the regular time production, overtime production, and the subcontracting variables associated with each product in a period along with the hiring and the layoff variables in each period are obtained.

4.2.1. *Production quantity of the second-phase products*

For each period, the second-phase products are first sorted in ascending order based on their regular time production cost as:

$$\begin{aligned} &Z_{[i]} = CP_{2it} \quad ; \quad i = 1,2,...,N \\ &Z_{[1]} \le Z_{[2]} \le ... \le Z_{[N]} \end{aligned} \tag{25}$$

Then, the regular time production quantities of the second-phase products are calculated by Eq. (26) based on the above sorting.

$$P_{2[i]t} = rand\left[0, Min\left\{Max\left(0, \frac{W_{t\max} f - \sum_{l<i}\sum_{t=1}^{T} e_{2l}\left(\left(P_{2i(t-1)}\right)+1\right)^{br} \times P_{2it}^{br+1}}{e_{2i}}\right), Max\left(0, \frac{R_{Jt_{\max}} - \sum_{l<i}\sum_{t=1}^{T} a_{2[l]j}\left(\left(P_{2i(t-1)}\right)+1\right)^{br} \times P_{2it}^{br+1}}{a_{2[i]j}}\right)\right\}\right] \tag{26}$$

Using Eq. (28), the overtime production quantities of the second-phase products are calculated based on their sorted overtime production costs in Eq. (27).

$$\begin{aligned} &E_{[i]} = CO_{2it} \quad ; \quad 1,2,...,N \\ &E_{[1]} \le E_{[2]} \le ... \le E_{[N]} \end{aligned} \tag{27}$$

$$O_{2[i]t} = rand\left[0, Min\left\{\begin{array}{l} Max\left(0, \dfrac{W_{t\max}\alpha_{2t} f - \sum_{l<i}\sum_{t=1}^{T} e_{2l}\left(\left(O_{2i(t-1)}\right)+1\right)^{br} \times O_{2it}^{br+1}}{e_{2i}}\right), \\ Max\left(0, \dfrac{\beta_{jt} \times RJ_{t\max} - \sum_{l<i}\sum_{t=1}^{T} a_{2[l]j}\left(\left(O_{2i(t-1)}\right)+1\right)^{br} \times O_{2it}^{br+1}}{a_{2[i]j}}\right) \end{array}\right\}\right] \tag{28}$$

The subcontracting volumes of the second-phase products are obtained by Eq. (29).

$$S_{2[i]t} = rand\left[0, S_{2it\max}\right] \tag{29}$$

The genes are created by the summation of the above three quantities as follows:

$$TP_{[i]t} = P_{2[i]t} + O_{2[i]t} + S_{2[i]t} \tag{30}$$

In addition, in order to determine the values of the other variables in Phase 2, the following set of equations is used:

$$W_t = \sum_{i=1}^{n} e_{2i} \times P_{2it} \tag{31}$$

$$W_t = W_{t-1} + H_t - L_t \tag{32}$$

$$TP_{[i]t} + B_{2it} - B_{2i(t-1)} + I_{2i(t-1)} - I_{2it} = D_{2it} \tag{33}$$

The variables of the first phase are calculated similarly.

4.3. Phase 1: the sub-population approach

In this approach, the population is divided into several subpopulations, which get different weights through scalarizing multiple objectives into a single objective (Chang et al., 2006). In the first phase of the proposed SPGA, the original population is divided into several small sub-populations, where the solution space is checked. Moreover, similar to SPGA-II purposed by Chang et al. (2006), the subpopulations communicate to each other in order to result in a better diversity and convergence. The subpopulations can benefit by sharing information and communicating with each other simultaneously since they explore different solution spaces and each one has the ability to find a new efficient solution. For this reason, the suggested SPGA benefits from a global Pareto archive. In this archive, each subpopulation is liked by a squad team with a pre-assigned goal hoping that the teams will be marching in the correct direction in order to meet the high pick of the landscape in terms of fitness performance. Therefore, each subpopulation focuses on a specific space, in which the whole population has unchanged solution diversity among the subpopulations. The appropriate diversity of the Pareto front is the primary aim of the first phase of the proposed SPGA. For this reason, a combination of a weighting approach and the Min-Max method is used.

Multi-objective optimization problems are usually solved by scalarization, where a problem with multiple objectives is converted into a single objective problem or a family of single objective optimization problems. The so-called “weighted aggregation” method is one of the most famous methods for this purpose. In this method, a linear combination of the objective

functions, $f_i(x)$; $i=1,2,...,m$, with nonnegative weights, w_i, is used to form an "aggregated function" $F(x)$ defined in Eq. (34).

$$F(x)=\sum_{i=1}^{m} w_i f_i(x)$$
$$\sum_{i=1}^{m} w_i = 1 \;\; ; \;\; w_i \geq 0 \tag{34}$$

Besides, the "dynamic weighted aggregation" (DWA) is the best method for aggregating the objective functions, since this method has shown better ability to estimate concave Pareto-fronts (Jin et al., 2000). DWA uses the following weight scheme to aggregate two objective functions:

$$\left(w_1(t), w_2(t)\right) = \left(|\sin(2\pi t/R)| \;,\; 1-w_1(t)\right), \tag{35}$$

where t refers to t-th subpopulation, $\left(t=1,2,...,\; Ns\right)$, and $R=200$.

The main shortcoming of the proposed weighting method is the need to scalarise the objectives. In this regard, the Min-Max method is employed in this paper, which aims at minimizing the distance of every solution from the best possible solution f^*.In other words, for the i-th objective function $f_i(x)$ with the best available solution f_i^*, we have $f^* = \left(f_1^*, f_1^*,..., f_m^*\right)^T$. Thus, the following achieved scalarizing function is minimized subject to the constraint of the problem:

$$Min\left[\sum_{i=1}^{m}\left(\left(f_i(x)-f_i^*\right)/f_i^*\right)^p\right]^{1/p},$$
$$s.t.\;\; X \in S \tag{36}$$
$$1 \leq p \leq \infty$$

where p provides various ways of measuring the distance. The most widely applied values of p are 1 for the simplest formulation, 2 for the Euclidean distance, and 3fortheCheff-norm (Andersson, 2000). The major challenge of this approach is to find the value of p that maximizes the satisfaction of the decision maker (DM). However, a combination of a weighting method and the Min-Max approach is used in the first phase of the proposed SPGA method since there is only one output and the DM has to accept it as a final solution. In other words, two problems are solved using the hybrid approach; one is the mono-solution of the Min-Max and the other is the scalarizing problem in the weighting method. Consequently, there would be two objective functions scalarized by:

$$\left[w\left(\left(f_1(x) - f_1^* \right) / f_1^* \right)^p + (1-w)\left(\left(f_2(x) - f_2^* \right) / f_2^* \right)^p \right]^{1/p} , \quad (37)$$

where $f_1(x)$ and $f_2(x)$ are the individual minima of each objective function, and $0 < w < 1$ denotes the weight (or the relative importance).

In this paper, the exponent p in Eq. (36) is assumed1. In addition, the normalization process is applied as well since the objective functions in Eqs. (1) and (2) possess different scales. Note that, in order to solve all the examples generated in this paper, based on which the minimum solution in all runs is f^* for each objective, 10 different seeds are used for each solution algorithm. These are substituted in Eq. (37).

The elitism strategy is used in the first stage of the proposed SPGA. It randomly chooses a number of individuals from the nondominated set into the mating pool. These individuals are used in the crossover operation described in Section 3.3.1. Note that the elitism strategy of each subpopulation is different from those of the others. In addition, in the selection operation of the first phase, the binary tournament selection is used, since there are few individuals in each subpopulation. Therefore, the binary tournament takes shorter time span than the roulette wheel. The smaller the objective value of each chromosome, the better the chance of its selection is.

4.3.1. Crossover

In order to generate a number of offspring in the crossover process, the chromosomes exchange genes through the breakage and reunion of two chromosomes. These offspring represent solutions that combine substructures of their parents. An arithmetic crossover is chosen in this paper in order to explore the solution space.

- **Arithmetic crossover (AC):** Arithmetic crossover generates an offspring by linearly combining two selective parents as shown in Figure 3. The principle of this operator is based on the following equation:

$$\tau_{it} = \gamma \times TP_{it} + (1-\gamma) \times TP''_{it} \quad ; \quad \gamma \in (0,1) \quad (38)$$

Please insert Figure 3 about here

4.3.2. Mutation

In the mutation operation, an offspring solution is generated by randomly modifying its parent's feature, through which a reasonable level of diversity is obtained in the population and

the search is facilitated by jumping out of local optimal solutions. In this paper, an arithmetic mutation with the following description is chosen.

- ***Arithmetic mutation (AM)*:** The arithmetic mutation is a mutation scheme that reduces the total aggregate production level of a randomly selected period (gene) in the parent chromosomes by the amount of Δ, where the same Δ is added to another randomly selected period of the parent chromosomes. Figure 4 illustrates this operation.

Please insert Figure 4 about here

- ***Exchange mutation (EM):):*** exchange mutation is a mutation scheme that swaps the value of the two randomly selected genes of the current solution together. Figure 5 shows this type of mutation operation.

Please insert Figure 5 about here

4.4. Phase 2: improvement of Pareto solutions

Since it is probable that SPGA miss searching some significant part of the spaces, a restore of the subpopulations into a big population at the beginning to improve the solution quality is required. Note that Pareto archives in the first phase of the proposed SPGA do not communicate with each other, making not to find a better possible solution (Chang et al., 2006).This is also mentioned in the second phase of the proposed SPGA, in which the elitism strategy for a global archive of the first phase collects the best-nondominated solution from all sub-populations as a global Pareto archive. It then uses a new fitness function based on non-dominated sorting. It is also expected that the global Pareto archive improves the solution quality with simultaneous diversity.

4.4.1. Non-dominated sorting

In a nondominated sorting procedure, a ranking method is used to emphasize on good solutions and to utilize niche method to keep up stable subpopulations of the good solutions. In this paper, the non-dominated sorting, adapted from Srinivas & Deb (1994), is carried out in which the population is ranked based on individual's non-domination. For this reason, the locally non-dominated individuals of the population are first identified from the current population.

Then, the first locally non-dominated frontier in the population is assumed to be constituted by all these individuals, where it is assigned a large *Dummy_Fitness_Value*. The same fitness value is assigned to give an equal reproductive potential to all these locally non-dominated individuals. Then, these classified individuals are shared with their dummy fitness values in order to maintain diversity in the population. Sharing is achieved by performing a selection operation using degraded fitness values obtained by dividing the original fitness value of an individual by a quantity proportional to the number of individuals presented in its niche. This causes multiple locally Pareto-optimal solutions to coexist in the population. In order to calculate the niche dimensions in a certain population, the concept of niche cubicle by Hyun et al. (1998) is utilized in this paper. A niche cubicle for a given individual is a rectangular region whose center is that particular individual. The size of the niche cubicle is computed using Eq. (39).For further clarification, consider a problem with m objectives. Assume that Max_{lt} and Min_{lt} are the maximum and the minimum of the *l*-th objective function in generation *t*, respectively. Then, the niche size for the *l*-th objective, σ_{lt}, is computed as:

$$\sigma_{lt} = \frac{Max_{lt} - Min_{lt}}{\sqrt[m]{Pop\_Size}} \quad ; \quad l = 1, 2, ..., m \,, \tag{39}$$

where Pop_Size is the size of the population. Interested readers are referred to Hyun et al. (1998) for more details.

The niche cubicle is calculated in every generation. the construction of niche cubicles in a two-objective problem is illustrated in Figure 6, in which two niche cubicles are shown for two arbitrarily chosen individuals X_1 and X_2.With equalized niche cubicles, the solution density of a niche cubicle can be simply measured by the number of individuals included in the cubicle. It is more probable that a solution located in a less dense cubicle to survive in the next generation. For example, if the niche cubicle of X_1 has less dense than X_2, then X_1 will have a higher survival probability than X_2 does. These locally non-dominated individuals are ignored temporarily after sharing to process the rest of the population, in the same way, to identify individuals for the second locally non-dominated frontier. Then, a new dummy fitness value is assigned to these locally non-dominated solutions that are kept smaller than the minimum shared dummy fitness of the previous frontier. This process continues until the entire population is classified into several frontiers (Mansouri, 2005).

Please insert Figure 6 about here

Therefore, the main characteristic of the second phase is that the subpopulations are merged into a big population. In other words, the size of the new population is equal to that of the original population. Note that not only the algorithm merges the subpopulations, but also the external memory of Pareto solution is merged and updated. Also, there are two more characteristics that make the second phase different from the first phase. The first characteristic is to evaluate the fitness function of the initial population using a desirability function that results in generating various Pareto solutions. The other difference is related to the selection strategy. The selection method in the second phase applies the binary tournament selection. The relationship between the initial phase and the second phase is shown in Figure 7.

Please insert Figure 7 about here

In order to validate the results obtained and to assess the performance of the proposed SPGA, two other meta-heuristics called weighted sum multi-objective genetic algorithm (WMOGA) and non-dominated sorting genetic algorithm II (NSGA-II) are utilized in the next two sections to solve a set of benchmark problems in Section 4.

4.4.2. WMOGA

One of the most famous methods in multiobjective optimization is the so-called weighted aggregation, based on which the objectives are usually formulated in a single objective form using the weighting sum approach. The pseudo-code of the utilized MOGAW is shown in Figure 8.

Please insert Figure 8 about here

4.4.3. NSGA-II

NSGA-II, as a well-known multi-objective evolutionary algorithm (MOEA), has been the most widely used methods that have been proven to do well in various real-world problems (Coello Coello, 2002). It is used in this research because there have been many investigations

ensuring that it often converges to Pareto-optimal set and is able to obtain solutions that are often spread well over the Pareto-optimal set. The pseudo-code of the utilized NSGA-II of this paper is presented by Algorithm1shown in Figure 9. NSGA-II takes the fast-non-dominated-sort mechanism demonstrated in Algorithm 2 (see Figure 9) to ensure good convergence. For more details on NSGA-II, one can refer to Srinivas et al. (1994).

Please insert Figure 9 about here

5. Computational study

The SPGA algorithm is coded in Matlab7.8 Software, where the code is executed on a PC with Intel Pentium 4, 1.67 GHz processor, 512MB memory. All the three algorithms are tested on 20 sets of multi-objective aggregate production planning problem instances from the literature, listed in Appendix A.

The performance of the proposed SPGA is evaluated and is compared to the ones obtained using WMOGA and NSGA-II in terms of the following performance measures.

5.1. Performance measures

As the two objectives conflict with each other, evaluating an MOEA is a challenging issue. It is well known that multi-objective optimization problems need multiple, but evenly distributed solutions to form a Pareto frontier to make them useful for decision makers. The performance of an MOEA is often evaluated in terms of some multi-objective performance measures. The four standard metrics used in this paper to evaluate the performances of the solution algorithms are as follows:

- The *spacing metric S* defined in Eq. (40) is utilized to express the distribution of individuals over the non-dominated region (Srinivas & Deb, 1994). This metric allows one to measure the uniformity of the spread of the points in the solution set.

$$S = \sqrt{\sum_{i=1}^{n} (d_i - \overline{d})^2 \Big/ (n-1)} \qquad (40)$$

In Eq. (40), d_i is the Euclidean distance between consecutive solutions in the obtained non-dominated set of solutions defined as follows:

$$d_i = \min_j \left\{ | f_1^i(x) - f_1^j(x) | + | f_2^i(x) - f_2^j(x) | \right\} \quad ; \quad \forall i, j = 1, 2, ..., n, \qquad (41)$$

and $\bar{d}$ is the sample mean value of all d_is . Note that when all the members spread uniformly and separately this metric takes the value of "0".

- The closeness between the Pareto solutions and the ideal point (0,0) is used as another measure called *mean ideal distance (MID)* defined in Eq. (42) (Zitzler & Thiele, 1998).

$$MID = \sum_{i=1}^{n} c_i / n, \tag{42}$$

where n is the number of non-dominated sets and $c_i = \sqrt{f_1^2 + f_2^2}$ for a bi-objective optimization problem. The lower the value of *MID*, the better the solution quality we have.

- Another criterion to measure the quality of non-dominated solutions is to pass the best smooth line using the slope and intercept and then calculate the area under linear regression curve (*ALC*). Since the problem considered in this research involves two objective functions, the area of the triangle established by the intersection points of the estimated line with the coordinate axes is used to determine *ALC*. The lesser the triangle area the better the set of non-dominated solutions is.
- Finally, the computational (CPU) time of running the algorithms to reach a near optimum solution is another criterion (Hajipour et al., 2014).

5.2. Parameters settings

The aim in this subsection is to determine the optimal parameter setting of the algorithms. Several parameters may influence the performance of an algorithm. For example, the larger the population sizes the better the solution quality with a higher computational expense. Moreover, the bigger the number of generations, the better the diversity is. However, it may also be a trade-off to reduce the number of generations. Besides, the crossover and mutation operators may have a significant impact on the solution quality. In this research, all the parameters of the three algorithms are tuned using the Taguchi method (details are not provided to save spaces) and are shown in Table 2.

Please insert Table 2 about here

5.3. Comparative analysis

In order to compare the performance of the proposed SPGA with the ones of WMOGA and NSGA-II in terms of the above-mentioned performance measures, all algorithms solve all 20

test problems. These problems are randomly created based on uniform distributions used to generate their parameters described in Tables A1 and A2 of Appendix A, with the expected value, upper and lower bounds of the quantitative objective in Table A3. The bounds of the parameters are considered to be close to real conditions. The size of a problem depends on (1) the number of products in the first phase $[K]$, (2) the number of products in the second phase $[I]$, (3) the number of machines in the first phase $[L]$, (4) the number of machines in the second phase $[J]$, and (5) the number of periods $[T]$. Table3 reports the CPU time required, the value of the *spacing metric* (*S*), *MID*, and *ALC* of all algorithms when they solve the problems.

Please insert Table 3 about here

The three algorithms are statistically compared based on the properties of their obtained solutions via the analysis of variance (ANOVA) test. These outputs are reported in Tables 4-7 in terms of the defined metrics. To clarify the statistical results better, interval-plots are presented in Figure 10.

Please insert Tables 4-7 about here

Please insert Figure 10 about here

The results in Table 3 show that the equality of the mean CPU times of the three algorithms cannot be rejected at 95% confidence (P-Value =0.489>0.05). Meanwhile, the upper left plot in Figure 10 indicates WMOGA is the fastest one. Then, NSGA-II comes into the picture and SPGA the slowest one.

Based on the results in Table 5, again while the equality of the mean *spacing* of the three algorithms cannot be rejected (P-Value =0.214>0.05), the upper right plot in Figure 10 shows NSGA-II the best, then SPGA is ranked second, and WMOGA the third.

The results in Table 6 show that while the three algorithms perform statistically equal in terms of the *MID* metric (P-Value =0.059>0.05), the lower left plot in Figure 10 indicate SPGA the best, NSGA-II the second, and WMOGA the third.

Finally, the results in Table 7 show that the three algorithms have statistically equal performance in terms of the *ALC* metric (P-Value =0.232>0.05). However, the lower right plot in Figure 10 indicates SPGA the best, NSGA-II the second, and WMOGA the third.

In short, the non-dominated solutions of the 20 test problems obtained using the three algorithms show the comparability of SPGA with NSGA-II and WMOGA. This somehow validates the solutions obtained using SPGA.

6. Conclusions and future extensions

In this paper, a bi-objective optimization model has been developed for an aggregate production planning (APP) problem with labor learning effect and machine deterioration. The aim of the first objective was to maximize the profit by improving learning as well as reducing the failure cost of the system. The second objective aimed at increasing customer satisfaction. The goal was to obtain appropriate levels of production rates in regular and overtimes, inventory and shortage levels, workers' hiring and firing levels, and the quantities of the products that were subcontracted. To demonstrate the validity of the proposed mathematical formulation, the multi-objective model was converted into a single-objective model using the fuzzy goal programming method, based on which computational experiments were performed on a set of random small-sized instances solved by the LINGO software. As the problem was classified NP-hard, a subpopulation genetic algorithm (SPGA) was proposed to obtain Pareto front. In order to validate the results obtained and to assess the performance of the SPGA, two other meta-heuristics called weighted sum multi-objective genetic algorithm (WMOGA) and non-dominated sorting genetic algorithm II (NSGA-II) were utilized to solve a set of 20 benchmark problems. In order to obtain better quality solutions, the parameters of all algorithms were tuned. At the end, the performances of all parameter-tuned algorithms were statistically compared together. The computational results showed that while all the solution algorithms had similar performances statistically, SPGA had a better performance compared to the other two algorithms in terms of *MID* and *ALC* metrics. However, in terms of the *mean CPU times* criterion WMOGA was the fastest and in terms of the *mean spacing* criteria, NSGA-II was the best.

In order to extend the application of the proposed methodology to cover more real-world problems, the demand can be considered uncertain. In addition, due to the high cost involved in an APP, monetary considerations such as the effect of time-valued money can be taken into account.

Acknowledgment

The authors are thankful for constructing comments of the respected anonymous reviewers. Taking care of the comments improved the presentation, significantly.

References

Affenzeller, M. (2002). New generic hybrids based upon genetic algorithms. *Lecture Notes in Computer Science*, 2527, 329-339.

Andersson, J. (2000).A survey of multi-objective optimization in engineering design. In Technical Report LiTH-IKP-R-1097, Department of Mechanical Engineering, Linköping University, Linköping, Sweden.

Axsater, S. (1986).On the feasibility of aggregate production plans.*Operations Research*, 34, 796–800.

Baykasogluy, A. (2006). MOAPPS 1.0: Aggregate production planning using the multiple objective Tabu search. *International Journal of Production Research*, 39, 3685–3702.

Behnamian, J., Zandieh, M., FatemiGhomi, S.M.T. (2009).A multi-phase covering Pareto-optimal front method to multi-objective parallel machine scheduling.*International Journal of Production Research*, 48, 4949-4976.

Bitran, G.R., Haas, E.A., Hax, A.C. (1982). Hierarchical production planning: A two stage system. *Operations Research*, 29, 232–251.

Chakrabortty, R.K., Akhtar Hasin, M.A., Sarker, R.A. & Essam, D.L. (2015). A possibilistic environment based particle swarm optimization for aggregate production planning. *Computers & Industrial Engineering*, 88, 366-377.

Chang, P.-C., Chen, S.-H., Hsieh, J.-Ch. (2006).A global archive sub-population genetic algorithm with adaptive strategy in multi-objective parallel-machine scheduling problem.*Lecture Notes in Computer Science*, 4221, 730-739.

Chang, P.-C., Chen, S.-H., Lin, K.-L.(2005). Two-phase sub-population genetic algorithm for parallel machine-scheduling problem.*Expert Systems with Applications*, 29, 705-712.

Chankong,V., Haimes, Y. (1983). Multi-objective decision making theory and methodology. New York: North-Holland.

Cochran, J.K., Horng, S.M., Fowler, J.W. (2003). A multi-population genetic algorithm to solve multi-objective scheduling problems for parallel machines.*Computers and Operations Research*, 30, 1087-1102.

CoelloCoello, C.A., Van Veldhuizen, D.A., Lamont, G.B. (2002).Evolutionary algorithms for solving multi objective problems, Kluwer Academic Publishers.

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T. (2002).A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6, 182–197

Entezaminia, A., Heydari, M.,Rahmani, D.(2016). A multi-objective model for multi-product multi-site aggregate production planning in a green supply chain: Considering collection and recycling centers. *Journal of Manufacturing Systems*,40,63–75.

Fahimnia, B., Luong, L.H.S., Marian, R.M. (2006). Modeling and optimization of aggregate production planning – A genetic algorithm. In: Proceedings of world academy of science, engineering and technology, pp. 169-174.

Ganesh, K., Punniyamoorthy, M. (2005).Optimization of continuous-time production planning using hybrid genetic algorithms-simulated annealing.*International Journal of Advanced Manufacturing Technology*, 26, 148–154.

Geem, Z.W., Kim, J.H., Loganathan, G.V. (2002). Harmony search optimization: application to pipe network design. *International Journal of Modeling and Simulation*, 22, 125–133.

Hajipour, V., Mehdizadeh E., and Tavakkoli-Moghaddam, R. (2014). A novel Pareto-based multi-objective vibration damping optimization algorithm to solve multi-objective optimization problems. *Scientia Iranica, Transactions E: Industrial Engineering*, 21(6), 2368-2378.

Hans, A.E. (1988). Multi Criteria optimization for highly accurate systems. Multi criteria optimization in engineering and sciences. W. Stadler (Ed.), *Mathematical Concepts and Methods in Science and Engineering*, 19, 309-352.

Hossain, Md. M., Nahar, K., Reza, S. and Shaifullah, K. M., (2016). Multi-period, multi-product, aggregate production planning under demand uncertainty by considering wastage cost and incentives. *World Review of Business Research,* 6(2), 170 – 185.

Hung, Y.F., Hu, Y.C. (1998). Solving mixed integer programming production planning problems with setups by shadow price information. *Computers and Operations Research*, 25, 1027–1042.

Hyun, C.J., Kim, Y.K., Kim, Y. (1998). A genetic algorithm for multiple objective sequencing problems in mixed model assembly lines. *Computers and Operations Research*, 25, 675-690.

Jiang, G., Kong, J., Li, G. (2008). Aggregate production planning model of production line in Iron and Steel enterprise based on genetic algorithm. *Intelligent Control and Automation, WCICA 2008, 7th World Congress on, pp.* 7716 – 7719.

Jin.Y, Olhofer, M., Sendho, B. (2000), "Dynamic weighted aggregation for evolutionary multi-objective optimization: Why does it work and how?" In Proc. GECCO 2000.

Lee, K.S., Geem, Z.W. (2004). A new structural optimization method based on the harmony search algorithm. *Computers and Structures*,82, 781–798.

Leung, S.C.H., Chan, S.S.W. (2009).A goal programming model for aggregate production planning with resource utilization constraint.*Computers & Industrial Engineering*, 56, 1053–1064.

Leung, Y.W., Wang Y.P. (2000). Multi-objective programming using uniform design and genetic algorithm.*IEEE Transactions on System Man Cybernetics C: Appl. Rev*., 30, 293-304.

Mansouri, S.A. (2005). A multi-objective genetic algorithm for mixed-model sequencing on JIT assembly lines.*European Journal of Operational Research*, 167, 696-716.

Masud, A.S.M., Hwang, C.L. (1980). An aggregate production planning model and application of three multiple objective decision methods.*International Journal of Production Research*, 18, 741–752.

Mehdizadeh, E., Atashi–Abkenar, A.A., (2014). An integrated aggregate production planning model with two-phase production system and maintenance costs. *International Journal of Applied Operational Research*, 4, 87-106.

Mirzapour Al-e-hashem, S.M.J., Baboli, A., Sazvar, Z. (2013). Stochastic aggregate production planning model in a green supply chain: Considering flexible lead times, nonlinear purchase and shortage cost functions. *European Journal of Operational Research*, 230, 26-41.

Mirzapour Al-e-Hashem, S.M.J., Malekly, H., Aryanezhad, M.B.(2011). A multi-objective robust optimization model for multi-product multi-site aggregate production planning in a supply chain under uncertainty.*International Journal of Production Economics,*134,28–42.

MirzapourAl-e-hashem, S.M.J., Aryanezhad, M.B., Sadjadi, S.J.(2012).An efficient algorithm to solve a multi-objective robust aggregate production planning in an uncertain environment. *International Journal of Advanced Manufacturing Technology*, 58, 765–782.

Modarres, M., Izadpanahi, E. (2016). Aggregate production planning by focusing on energy saving: A robust optimization approach. *Journal of Cleaner Production*, 133, 1074-1085.

Mohan Kumar, G., NoorulHaq, A. (2005). Hybrid genetic-ant colony algorithms for solving aggregate production plan. *Journal of Advanced Manufacturing Systems* (JAMS), 1, 103–111.

Nam, S.J., Ogendar, N.R. (1992). Aggregate production planning–a survey of models and methodologies.*European Journal of Operational Research*, 61, 255–272.

Partovi, F., Seifbarghy, M. (2015). Service centers location problem considering service diversity within queuing framework.*Scientia Iranica E*, 22, 1103-1116.

Pradenas, L., Peñailillo, F. (2004). Aggregate production planning problem: A new algorithm. *Electronic Notes in Discrete Mathematics*, 18, 193–199.

Ramezanian, R., Rahmani, D., Barzinpour, F., (2012). An aggregate production planning model for two phase production systems: Solving with genetic algorithm and tabu search. *Expert Systems with Applications*, 39, 1256–1263.

Ramyar, M., Mehdizadeh, M., Hadji Molana, M. (2017). Optimizing reliability and cost of system for aggregate production planning in supply chain. *Scientia Iranica*, Articles in Press, DOI: 10.24200/sci.2017.4398.

Sadeghi, M., Hajiagha, S.H.R., Hashemi S.S.(2013). A fuzzy grey goal programming approach for aggregate production planning. *International Journal of Advanced Manufacturing Technology*, 64, 1715–1727.

Schaffer, J.D. (1985). Multiple objective optimization with vector evaluated genetic algorithms. In J. D. Schaffer(Ed.), genetic algorithms and their applications: Proceedings of the first international conference on genetic algorithms, pp. 93-100. Hillsdale, NJ: Lawrence Erlbaum.

Silva, A.F.D., Marines, F.A.S. (2014).A fuzzy goal programming model for solving aggregate production-planning problems under uncertainty: A case study in a Brazilian sugar mill. *Energy Economics*, 45, 196-204.

Simoes, A., Costa, E. (2002). Parametric study to enhance genetic algorithm's performance when using transformation.In Proceedings of the genetic algorithm and evolution computation conference (GECCO'02). New York: Morgan Kaufmann Publishers, pp. 9-13.

Srinivas, N., Deb, K. (1994). Multi objective optimization using non-dominated sorting in genetic algorithms.*Evolutionary Computation*, 2, 221-248.

Tabucanon, M.T., Majumdar, S. (1989). Production planning in a ship repair company in Bangladesh. Proceedings of the International Conference on MCDM: Application in industry and service. Bangkok: Asian Institute of Technology.

Taleizadeh, A.A., Niaki, S.T.A., Barzinpour, F. (2011). Multiple-buyer multiple-vendor multi-product multi-constraint supply chain problem with stochastic demand and variable lead-time: A harmony search algorithm. *Applied Mathematics and Computation*,217, 9234-9253.

Tan, K.C., Lee, T.H., Khor, E.F. (2001). Evolutionary algorithms for multi-objective optimization: performance assessments and comparison. IEEE Congress on Evolutionary Computation, 979-986. Seoul, Korea.

Techawiboonwong, A., Yenradee, P. (2003).Aggregate production planning with workforce transferring plan for multiple product types. *Production Planning & Control: The Management of Operations*, 14, 447-458.

Wang, R.C., Liang T.F.(2004). Application of fuzzy multi-objective linear programming to aggregate production planning. *Computers & Industrial Engineering*, 46(1), 17–41.

Wang, R.C., Liang, T.T. (2005). Aggregate production planning with multiple fuzzy goals.*International Journal of Advanced Manufacturing Technology*, 25, 589–597.

Wang, S.C., Yeh, M.F., (2014).A modified particle swarm optimization for aggregate production planning. *Expert Systems with Applications*, 41, 3069-3077.

Zitzler, E., Thiele, L. (1998). Multi objective optimization using evolutionary algorithms: A comparative case study. Parallel Problem Solving from Nature, Germany, 292-301.

Zitzler, E., Thiele, L. (1999). Multi-objective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, 3, 257-271.

Algorithm: ***The Main Procedure of the Proposed Sub-Population Genetic Algorithm***

First phase

1: **Representation**: Encoding a solution.

2: **Initialization**:

(**a**) **Parameter setting**: set the number of population*(N)*, the number of subpopulation*(Ns)*,the number of individuals in each subpopulation *(n)*, The total number of generations in Phase 1 and 2 (*Max_Gen1*,*2*), the probability of order crossover *(Pc)*, the probability of mutation *(Pam and Pem)*.

(**b**) **Initial population generation**: Randomly generate an initial population.

3: Dividing Population

4: Assigning a weight to each objective

5: counter←0

6: **while** counter <*Max_Gen*1 **do**

7: **for** *i*= 1 to *Ns* do

8: Evaluation: Evaluating the fitness function for each of the subpopulation using a combination of the min-max method and the weighting method (combining the objectives into a scalar fitness function) to generate various Pareto solutions.

9: Finding Pareto solutions: Efficient solutions of the current population are copied into an archive that stores the best-nondominated front obtained.

10: Selection/Elitist: Binary tournament selection and Elitist selection are employed to reproduce the next generation.

11: Generate next generation: Generating by GA operator (i.e. Crossover and Mutation)

12: Crossover operation: Select *Ns* × *Pc* pairs of parents from the current population and perform crossover on the parents.

13: Arithmetic Mutation (AM): Select *Ns* × *Pam* chromosomes from current population and mutate the individual bits.

14: Exchange Mutation (EM): Select Ns × *Pem* chromosomes from the current population.

15: End for

16: *counter*←*counter* +1

17. **End While**

Second phase

18: *counter*←0

19: Merge sub-population

20: **while** counter <*Max_Gen2* **do**

21: **for** *i*= 1 to *N* do

22: Finding Pareto solutions

23: Evaluation: Evaluating the fitness function for each of the initial population to generate various Pareto solutions.

24: Selection/Elitist: Certain selection use Non-dominated sorting and Elitist selection is employed to reproduce the next generation.

25: Generate next generation: Generating by GA operator (i.e. Crossover and Mutation)

26: End for

27: *counter*←*counter* +1

28: End while

Fig.1. The general pseudo-code of the proposed SPGA

Period (*t*)	1	2	...	*T*-1	*T*
Type 1	TP_{11}	TP_{12}	…	$TP_{1(T-1)}$	TP_{1T}
Type 2	TP_{21}	TP_{22}	…	$TP_{2(T-1)}$	TP_{2T}
...			…		
Type I	TP_{I1}	TP_{I2}	…	$TP_{I(T-1)}$	TP_{IT}

Fig 2. Chromosome representation

Parent 1

TAP of product Type 1	TP_{11}	TP_{12}	TP_{13}	TP_{14}	TP_{15}	TP_{16}	TP_{17}	TP_{18}
TAPof product Type2	TP_{21}	TP_{22}	TP_{23}	TP_{24}	TP_{25}	TP_{26}	TP_{27}	TP_{28}

Parent 2

TAP of product Type 1	TP'_{11}	TP'_{12}	TP'_{13}	TP'_{14}	TP'_{15}	TP'_{16}	TP'_{17}	TP'_{18}
TAP of product Type2	TP'_{21}	TP'_{22}	TP'_{23}	TP'_{24}	TP'_{25}	TP'_{26}	TP'_{27}	TP'_{28}

Offspring

TAP of product Type 1	τ_{11}	τ_{12}	τ_{13}	τ_{14}	τ_{15}	τ_{16}	τ_{17}	τ_{18}
TAP of product Type2	τ_{21}	τ_{22}	τ_{23}	τ_{24}	τ_{25}	τ_{26}	τ_{27}	τ_{28}

Fig.3. Illustration of the arithmetic crossover

Parent

TAP of product type 1	TP_{11}	TP_{12}	TP_{13}	TP_{14}	TP_{15}	TP_{16}	TP_{17}	TP_{18}
TAP of product type2	TP_{21}	TP_{22}	TP_{23}	TP_{24}	TP_{25}	TP_{26}	TP_{27}	TP_{28}

Offspring

TAP of product type 1	TP_{11}	$TP_{12}-\Delta$	$TP_{13}+\Delta$	TP_{14}	TP_{15}	TP_{16}	TP_{17}	TP_{18}
TAP of product type2	TP_{21}	TP_{22}	TP_{23}	$TP_{24}-\Delta$	TP_{25}	$TP_{26}+\Delta$	TP_{27}	TP_{28}

Fig.4. Illustration of the arithmetic mutation

Parent

TAP of product type 1	TP_{11}	TP_{12}	TP_{13}	TP_{14}	TP_{15}	TP_{16}	TP_{17}	TP_{18}
TAP of product type2	TP_{21}	TP_{22}	TP_{23}	TP_{24}	TP_{25}	TP_{26}	TP_{27}	TP_{28}

Offspring

TAP of product type 1	TP_{11}	TP_{12}	TP_{17}	TP_{14}	TP_{15}	TP_{16}	TP_{13}	TP_{18}
TAP of product type2	TP_{21}	TP_{22}	TP_{23}	TP_{28}	TP_{25}	TP_{26}	TP_{27}	TP_{24}

Fig.5. Illustration of the exchange mutation

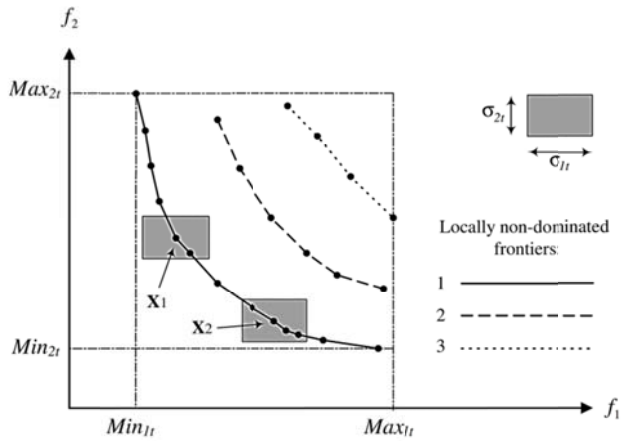

Fig.6. Niche cubicles along locally non-dominated frontier (Mansouri, 2005)

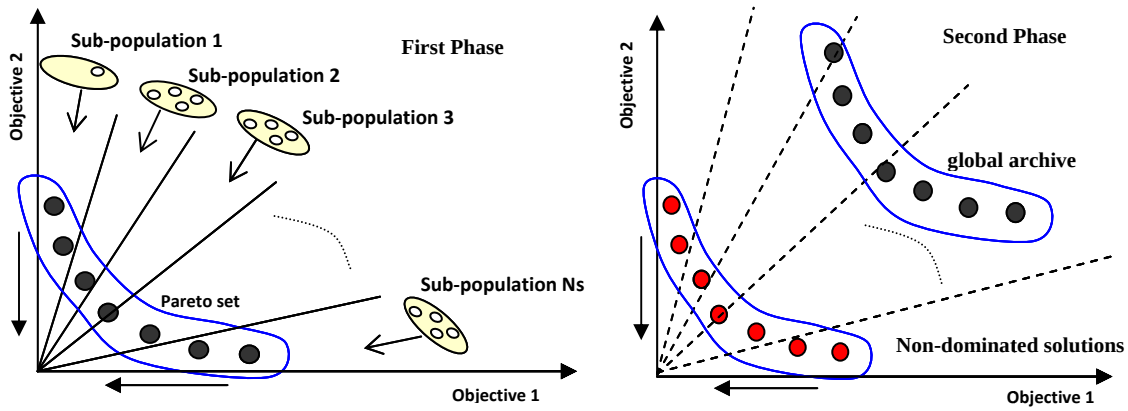

Fig.7. Relation between the first and the second phase

1: **Representation**: *Encode the problem into a search solution.*
2: **Initialization**:

- **Parameters setting**: *set the number of population (pop_size), Max_Gen (the total number of generations), Probability of order crossover (Pc), Probability of mutation (Pm), Probability of reproduction (Pr).*
- **Initial population generation**: *Randomly generate an initial population IPM.*

3: Assign weight to each objective
4: counter←0
5: **while** counter <*Max_Gen* **do**
6: **for** *i*= 1 to *pop_size* do
7: **Evaluation:** *Evaluate the fitness function for each of the solution using the hybrid min-maxmethod with the weighting method (to combine them into a scalar fitness function) to generate various Pareto solution.*
8: **Find Pareto solutions:** *Efficient solutions of the current population are copied into an archive that stores the best-nondominated front obtained.*
9: **Selection/Elitist:** *Roulette wheel selection and Elitist selection are employed to reproduce the next generation.*
10: **Crossover operation**: *Select pop_size × Pc pairs of parents from the current population:*
a. Determine the pairs of parents among the parent chromosomes.
b. Apply the crossover operator to produce two offspring corresponding to each pair.
c. Replace each offspring in the population instead of the parents.
11: **Mutation operation**: *Select pop_size × Pm chromosome from current population and mutate the individual bits (AM and EM).*
12: Update archive*: Efficient solutions of the current population are copied into an archive that stores the best-nondominated front obtained.*
13: **Generate next generation**:
14: End for
15: counter←counter +1
16. **End While**

Fig.8. The pseudo-code of the MOGAW

Algorithm1 The Pseudo-Code of NSGA-II

step.1: Set the parent vector $P = \varnothing$, the offspring vector $Q = \varnothing$, the collect vector $R = \varnothing$, and the generation number $t = 0$.
step.2: Initialize the parent vector P_0.
step.3: while t <the termination number **do**
(1) Combine the parent and offspring population via $R_t = P_t \cup Q_t$.
(2) Sort all solutions of R_t to get all non-dominated fronts F = fast-non-dominated-sort (R_t) where $F = (F_1, F_2, ...)$.
(3) Set c and i = 1.
(4) **while** the parent population size $| P_{t+1} | + | F_i | < N$ **do**
(a) Calculate the crowding-distance of F_i.
(b) Add the *ith* non-dominated front F_i to the parent pop P_{t+1}.
(c) $i = i + 1$.
end while
(5) Sort the F_i according to the crowding distance.
(6) Fill the parent pop P_{t+1} with the first $N - | P_{t+1} |$ elements of F_i.
(7) Generate the offspring population to Q_{t+1}.
(8) Set $t = t + 1$.
end while
step.4: the population in vector P is the non-dominated solutions.

Algorithm 2 The Pseudo-Code for the fast non-dominated sort (Function P)

step.1: For each population p in the P, get the solutions for which p dominates and save these solutions into S_p. Also calculate n_p, which is the number of dominating solutions.
step.2: Find the solutions whose $n_p = 0$ and add them to the first front F_1.
step.3: Initialize the front counter i = 1.
step.4:
while F_i is not empty **do**
Set the temp vector $Q = \varnothing$.
for each $p \in F_i$ **do**
for each $q \in Sp$ do
$n_p = n_p - 1$.
If $n_p = 0$ then add q to the Q.
end for
end for
$i = i + 1$ and the solutions in Q compose the F_i.
end while

Fig.9. The pseudo-code of the NSGA-II

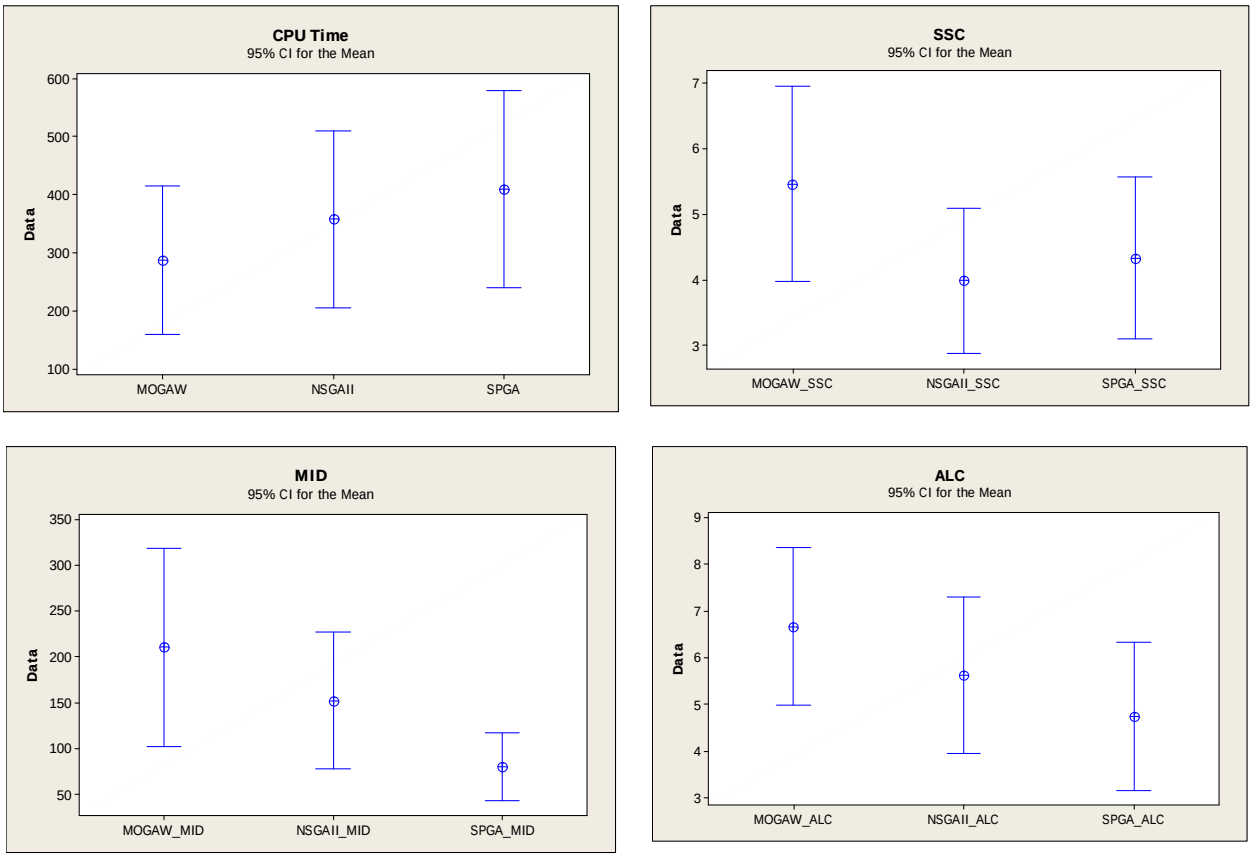

Fig.10. Comparability of SPGA in comparison with NSGA-II and MOGAW on CPU Time, MID, SSC and ALC metrics

Table 1: Solving various problems using the LINGO software

Problems	Lower Bound	Objective Function Value shown in (24)	CPU Time (s)
P1	0.880	0.880	3
P2	0.890	0.890	5
P3	0.780	0.780	6
P4	0.860	0.860	18
P5	0.900	0.900	45
P6	0.760	0.760	159
P7	0.856	0.856	189
P8	0.775	0.795	324
P9	0.882	0.892	423
P10	0. 892	0.900	632
P11	0.754	0.761	845
P12	0.837	0.856	1106
P13	0.774	0.795	1348
P14	0.871	0.891	1503
P15	0.755	0.760	2236
P16	0.697	-	4489
P17	0.765	-	7227
P18	0.635	-	12349
P19	0.797	-	18092
P20	0.647	-	26691

Table2: Parameter setting

Parameter setting SPGA		**Parameter setting NSGA-II**		**Parameter setting MOGAW**	
Parameter	**Value**	**Parameter**	**Value**	**Parameter**	**Value**
Pop_Size (*N*)	500	Pop_Size	500	Pop_Size	600
Number of sub-population (*Ns*)	50	Max_Gen	750	Max_Gen	800
n	10	Crossover rate (AM)	0.7	Crossover rate (AM)	0.6
Max_Gen phase 1	400	Crossover rate(EM)	0.2	Crossover rate(EM)	0.2
Max_Gen phase 2	600	Mutation rate	0.1	Mutation rate	0.2
Crossover rate (AM)	0.5	Elitist	top 20% of the population	Elitist	top 20% of the population
Crossover rate(EM)	0.3				
Mutation rate	0.1				
Elitist	Top 20% of the population				

Table 3: Evaluation of non-dominated solution for algorithms grouped by problem size and index type.

Problem No.	CPU Time			Spacing Metric (S)			MID			ALC ($\times 10^4$)		
	MOGAW	NSGAII	SPGA	MOGAW	NSGAII	SPGA	MOGAW	NSGAII	SPGA	MOGAW	NSGAII	SPGA
P1	16.85	27.56	36.60	9.33	8.11	9.00	59.90	55.23	40.94	1.93	0.93	0.46
P2	31.33	42.03	52.37	1.72	1.02	1.39	48.43	44.55	33.09	1.97	0.97	0.50
P3	38.76	51.39	67.20	1.71	1.01	1.38	56.52	52.09	38.63	2.29	1.29	0.83
P4	47.15	60.99	72.40	4.51	3.62	3.29	68.07	62.85	46.02	2.94	1.94	2.24
P5	58.21	71.31	90.40	5.93	4.61	5.24	80.77	73.26	52.79	2.97	1.97	0.97
P6	65.78	87.63	103.00	4.39	3.40	3.37	63.00	55.53	39.63	3.84	2.84	1.84
P7	76.31	88.38	101.43	3.16	2.64	2.54	52.31	40.30	17.84	4.12	3.12	2.12
P8	84.64	110.11	135.56	2.62	1.44	1.66	69.79	48.64	24.23	4.46	3.46	2.46
P9	90.59	115.16	142.10	2.75	1.90	2.33	77.30	54.13	27.82	6.07	5.07	4.07
P10	172.36	230.33	279.80	4.56	3.57	4.12	92.08	65.53	33.67	5.82	4.82	3.82
P11	218.94	289.21	348.00	2.32	1.19	1.13	93.02	59.96	30.97	7.03	6.03	5.03
P12	267.40	356.12	418.00	3.93	3.18	3.38	86.32	61.64	31.66	7.49	6.49	5.49
P13	294.41	379.93	452.20	4.79	3.89	4.46	91.56	65.68	33.75	8.21	7.21	5.61
P14	378.32	481.92	530.14	3.05	1.45	1.85	89.01	61.03	31.36	8.71	7.51	6.71
P15	545.92	698.13	796.31	5.93	5.54	4.65	233.77	175.63	84.31	8.25	7.25	6.25
P16	558.90	718.56	786.54	9.68	8.13	7.23	503.77	351.40	177.61	10.05	8.35	7.35
P17	582.53	746.25	803.34	10.58	5.80	6.45	600.72	394.82	199.65	10.51	9.51	8.51
P18	619.97	783.51	897.40	6.51	5.56	5.58	527.10	397.97	202.75	11.23	10.23	9.23
P19	789.34	825.12	929.76	10.41	6.39	7.40	642.04	458.66	236.16	11.71	10.71	9.71
P20	827.18	1005.25	1156.74	11.44	7.31	10.28	681.11	474.78	239.24	13.82	12.82	11.82

Table 4: Analysis of variance for the CPU time metric

Source	DF	SS	MS	F	P
Algorithms	2	149323	74661	0.73	0.489
Error	57	5869322	102971		
Total	59	6018645			

Table5: Analysis of variance for the *Spacing* metric

Source	DF	SS	MS	F	P
Algorithms	2	23.86	11.93	1.58	0.214
Error	57	429.39	7.53		
Total	59	453.25			

Table6: Analysis of variance for the *MID* metric

Source	DF	SS	MS	F	P
Algorithms	2	168886	84443	2.98	0.059
Error	57	1616620	28362		
Total	59	1785506			

Table7: Analysis of variance for the *ALC* metric

Source	DF	SS	MS	F	P
Algorithms	2	37.0	18.5	1.50	0.232
Error	57	702.3	12.3		
Total	59	739.2			

Appendix A. Randomly generated problem sets used in the experiments

I: number of products
K: number of products
J: number of machines
L: number of machines
T: number of periods

TableA1. Characteristics of the sample test problems

Problem No.	*I*	*T*	*J*	*L*	*K*
P1	2	2	1	1	2
P2	2	2	2	1	1
P3	3	2	2	2	2
P4	3	3	2	3	2
P5	3	2	4	4	1
P6	3	2	4	3	3
P7	3	6	3	4	3
P8	6	3	4	4	3
P9	3	5	3	4	5
P10	4	5	4	2	6
P11	4	4	3	4	6
P12	5	4	5	4	4
P13	6	3	7	5	5
P14	6	5	6	5	8
P15	6	7	4	4	10
P16	6	6	6	8	9
P17	6	7	4	5	11
P18	7	8	6	9	8
P19	8	8	5	4	9
P20	8	8	8	9	12

TableA2: Parameters of the problems

Parameter	Value	Parameter	Value	Parameter	Value
D_{2it}	U(10,1000)	R_{jt}, RR_{lt}	U(300 , 400)	β_{2lt}, β_{jt}	0.8
CP_{1kt}, CO_{1kt}	$U(50,70)$	CHH_t, CH_t	U(20 , 30)	F	50
CS_{1kt}, CS_{2it}	U(150,250)	CLL_t, CL_t	U(30 , 35)	$W_{t\max}, \mathrm{WW}_{t\max}$	200
CI_{1kt}, CI_{2it}	U(2,4)	CWW_t, CW_t	U(50 , 100)	$S_{2it\max}, S_{1kt\max}$	70
CP_{2it}, CO_{2it}	U(40,60)	I_{2i0}, I_{1k0}	U(0 , 100)	Q_{1kl}, Q_{2ij}	U(10 , 25)
SP_{2it}, SO_{2it}	U(500,2200)	W_0, WW_0	U(0 , 20)	q_{1kl}, q_{2ij}	U(0.2 , 0.3)
CB_{2it}	U(280,400)	f_{ik}	2	*Lid*	2
A_{2ij}, A_{1kl}	U(1,5)	E_{1k}, E_{2i}	U(3 , 5)	*Br*	0.8
u_{2ij}, u_{1kl}	U(0.1,2)	Z_{1kt}, Z_{2it}	U(5 , 10)		
$r_{2ijt}, \mathrm{r}_{1klt}$	U(10,20)	α_{2t}, α_t	0.8		

TableA3: Expected value, upper and lower bounds of the quantitative objective

Upper bound of the second objective (U_2)	Expected value of the second objective (G_2)	Lower bound of the first objective (L_1)	Expected value of the first objective (G_1)	Problems
4200	4000	39000	42000	P1
2600	2300	20000	24000	P2
5500	5000	50000	56000	P3
6000	5800	65000	70000	P4
3100	2700	15000	20000	P5
9350	9000	70000	80000	P6
19199	18000	170000	200000	P7
18500	17300	169000	176000	P8
17700	16100	350000	380000	P9
24000	20000	460000	480000	P10
19750	18000	420000	450000	P11
16800	15000	470000	500000	P12
16700	15600	469000	500000	P13
19500	18000	550000	600000	P14
32100	30000	439000	450000	P15
27500	25000	200000	230000	P16
25900	23000	630000	640000	P17
28600	26000	608000	625000	P18
36200	34000	730000	760000	P19
35100	32000	825000	840000	P20