Neural Comput & Applic
DOI 10.1007/s00521-017-3159-5

ORIGINAL ARTICLE

A new hybrid algorithm of simulated annealing and simplex downhill for solving multiple-objective aggregate production planning on fuzzy environment

A. A. Zaidan[1] · Bayda Atiya[2] · M. R. Abu Bakar[2] · B. B. Zaidan[1]

Received: 4 October 2016 / Accepted: 7 July 2017

Abstract Aggregate production planning (APP) is a significant level that seeks efficient production systems. In actual condition, APP decisions, production inputs, and relevant planning parameters are intrinsically imprecise, which results in significant complexities in the generation of master production schedules. Thus, this paper proposes a hybridization of a fuzzy programming, simulated annealing (SA), and simplex downhill (SD) algorithm called fuzzy–SASD to establish multiple-objective linear programming models and consequently resolve APP problems in a fuzzy environment. The proposed strategy is dependent on Zimmerman's approach for handling all inexact operating costs, data capacities, and demand variables. The SD algorithm is employed to balance exploitation and exploration in SA, thereby efficient and effective (speed and quality) solution for the APP model. The proposed approach produces rates for efficient solutions of APP in large-scale problems that are 33, 83, and 89% more efficient than those of particle swarm optimization (PSO), standard algorithm (SA), and genetic algorithm (GA), respectively. Moreover, the proposed approach produces a significantly low average rate for computational time at only 64, 77, and 24% compared with those of GA, PSO, and SA, respectively. Experimental results indicate that the fuzzy–SASD is the most effectual of all approaches.

Keywords Aggregate production planning · Simulated annealing · Multiobjective linear programming · Simplex downhill approach

✉ A. A. Zaidan
aws.alaa@fskik.upsi.edu.my

Bayda Atiya
Atiya@fsm.upm.edu.my

M. R. Abu Bakar
Bakar@fsm.upm.edu.my

B. B. Zaidan
bilal@fskik.upsi.edu.my

[1] Department of Computing Faculty of Arts, Computing and Creative Industry, Universiti Pendidikan Sultan Idris, Tanjong Malim, Malaysia

[2] Department of Mathematics, Faculty of Science, University Putra Malaysia, Serdang, Malaysia

1 Introduction

APP problems are considerably important in several manufacturing concerns. Managers are well aware that workforce and production decisions, in accordance with changing customer demands, can significantly affect a company economically. APP is defined as the planning of production quantities and time over a medium term of 3–18 months. During this period, the production to satisfy anticipated demand is determined. APP aims to set the overall production levels to satisfy the inconsistent or uncertain demands for each product category in the future. APP also considers policy and decision-making factors regarding suitable levels of hiring, overtime, layoff, backorder, subcontract work, and inventory [1]. Several APP models involving different levels of complexity have been launched since the 1950s. As denoted by [2], conventional approaches in addressing these problems can be categorized according to the following classifications: linear decision rule [3], linear programming [4], transportation method [5], management coefficient approach [6], search decision rule [8], simulation [7], and management coefficient approach [6]. In actual APP problems, input data or parameter values, including resource, demand, cost, and

Published online: 19 July 2017

objective functions, may be inaccurate because of partial or unavailable information [9]. The recent study by [10] classified prior research into factor uncertainties based on four primary approaches, namely (1) stochastic programming [11, 12], (2) fuzzy programming [13, 14], (3) stochastic dynamic programming [15], and (4) robust optimization [10].

The APP problems solved through stochastic programming techniques are based on theories, concepts, and methodologies of randomness theory. Therefore, the approach can only consider the restricted form of a given probability distribution function; thus, it cannot contribute significantly to decision-making in actual situations [16, 17]. Furthermore, the fuzzy approach is more efficient than stochastic programming approach, specifically when historical data are lacking; the former can also provide alternative models for imprecisions and uncertainties [16, 18–25]. In these studies, only the goals are defined as fuzzy values, and fuzzy models are resolved through their transformations into classically crisp mathematical programming problems [26]. Moreover, the approach may restrict the utilization of FMP because obtaining the crisp linear equivalent of a given fuzzy model in numerous situations may not be possible [27]. In recent decades, APP problems are characterized by high complexity and NP-hard problems. Thus, the research community aims to resolve complicated problems by using metaheuristic algorithms [28–33]. Although metaheuristic algorithms were successfully used to attack complex actual APP problems, no standard algorithm (SA) exists for all problems that depend on no-free-lunch theorem [34]. Therefore, modern concepts of self-adaptive modification of algorithms or hybrid algorithms that support the selection of a proper algorithm aim to overcome the implicit limitation of metaheuristics ones in solving actual APP problems. Given that the present study involved hybrid algorithms, we proposed a novel scheme that addresses the imprecision of operational coefficient values. This scheme was formulated as a multiple-objective linear programming (MOLP) model for resolving APP problems involving multiple periods and products. A hybrid optimizing algorithmic procedure that combines fuzzy programming, simulated annealing (SA), and simplex downhill (SD) algorithm called fuzzy–SASD was proposed to resolve the model.

The remaining part of this paper is outlined as follows: Sect. 2 provides the survey output of the research. Section 3 defines the ideal framework. Section 4 describes the mathematical formulations of multiple-objective APP problems. Section 5 offers procedural solutions. Section 6 validates the proposed model and demonstrates the effectiveness of the proposed method through computational study and results. Finally, a conclusion is provided in Sect. 7.

2 Literature review

Hybrid algorithms have been developed by combining two or more algorithms to improve overall search efficiencies. Several researchers have attempted to exploit the advantages of individual algorithms for a considerable purpose [35]. A hybrid model of the genetic algorithm (GA) and ant colony optimization was proposed to solve the APP of long-range policies of industries [36]. In addition, a mixed-integer linear programming was developed for a generalized two-phase APP scheme [37]. Subsequently, genetic algorithmic and tabu-type search methods were applied to resolve the APP model. In [38], modified particle swarm optimization (PSO) schemes were suggested to resolve the integer-based linear programming model for APP problem sets. Furthermore, the assumption that inexact parametric values that are deterministic can result in useless and impractical results [21, 39]. Several metaheuristic algorithms suffer from problems, such as traps in their localized optimal and slow convergence rate. Several researchers used the fuzzy approach with metaheuristic or hybrid metaheuristic algorithms. In [40], the fuzzy-integrated production distribution aggregate planning model is considered in the supply chain. The proposed model was solved through GA. Authors [27] proposed the tabu search (TS) algorithm to solve fuzzy goal programs for APP problems. Similarly, [26] proposed fuzzy ranking methods and TS algorithm for solving multiple-objective APP problems. Additionally, [41] proposed an interactive fuzzy-based GA approach to resolve APP problems in an uncertain environment. However, only two types of products were applied in their limited case study. In [42], a hybridized optimizing algorithm that combines fuzzy random simulation, a neural network, GA, and simultaneous perturbation stochastic approximation algorithm was suggested for solving APP decision-making problems in fuzzy environments. In addition, [43] proposed an integer linear programming model to resolve APP problems. This approach employs trilateral possibilities of distribution to handle all imprecise capacity data, operating costs, and demands. The authors subsequently utilized modified versions of probabilistic environments according to PSO methods to resolve APP modeling equations. Although these methods have adopted a fuzzy approach to handle inaccuracies in the environment and a hybrid algorithm for solving the model, the proposed model presents a singular objective.

We classified the major shortcomings of existing methods as follows [37, 43]): (1) All these methods are generally focused on resolution algorithms, but they do not consider generalized models. They are also incompatible with actual production systems. Thus, researchers presented no generalized and comprehensive model to formulate actual production environments. (2) Considering all parameters in an APP model is challenging, thereby making the method inefficient in terms of accuracy and runtime. (3) Most models for resolving APP problems are related to single objectives, and they are incompatible with actual production planning systems.

3 Conceptual framework

A. The problem was organized and devised as a multiple-objective linear programming model for resolving APP problems, which involve multiple periods and products that consider suitable levels of hiring and firing, overtime, production cost, and inventory as shown in Fig. 1.
B. To handle all imprecise parameters, we adopted a new method depending on Zimmerman's approach to transfer fuzzy data to crisp data as shown in Fig. 1.
C. To solve the outcome for APP problems, a new hybrid (SASD) was proposed to solve all aforementioned parameters of the said problem. Given the complexity and diversity of actual APP problems resulting from the planning process, modern approaches coherent with the innovation in technology over time and the complexities of the dynamic movement of the competition and market are required. Therefore, a single algorithm cannot be adopted for all real-world APP problems as shown in Fig. 1.

4 Mathematical programming model

The mathematical model of a MOLP for APP was proposed. We presumed that an industrial manufacturing company produces n-types of products to satisfy the market demands over a planning time horizon T.

4.1 Notational definitions

The notations utilized in the formulation of MOLP models for APP are as follows:

n quantity of production, $n = 1, 2, \ldots, N$.

t number of periods in the planning horizon, $t = 1, 2, \ldots, T$.

$c_{n\,t}$ cost of production for each ton of production n for period t.

h_t cost of hiring each labor in period t.

f_t cost layoff each labor in period t.

w_t cost of regular labor per period t.

i_{nt} inventory carrying cost for each ton of product t.

o_t cost of overtime labor for each man- hour for period t.

P_{nt} quantity of production n for period t.

D_{nt} predicted demand for product n per period t.

I_{nt} level of production inventory n for per period t.

W_t labor level for period t.

F_t labors fired for period t.

H_t labors hired for period t.

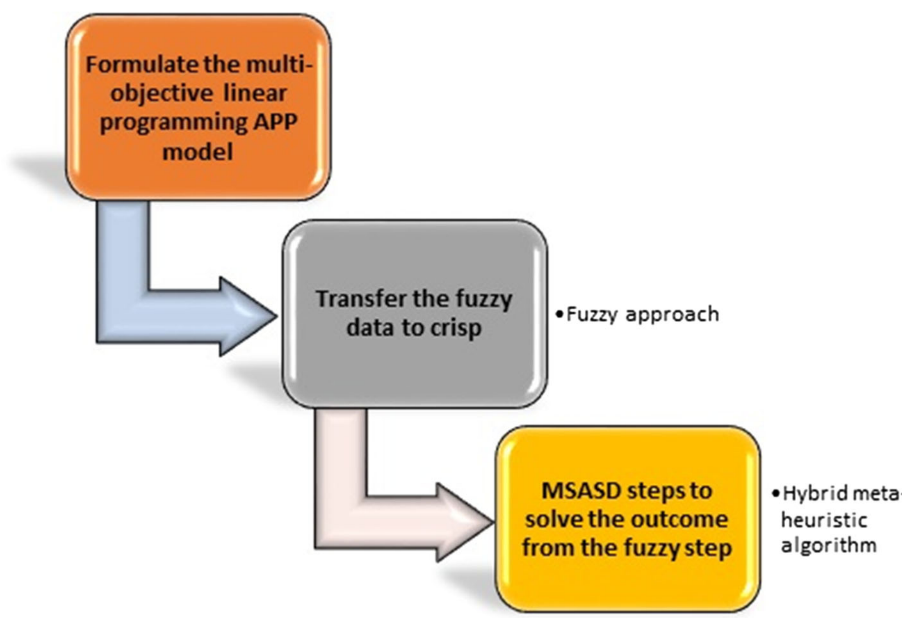

Fig. 1 Conceptual framework

O_t man-hours of overtime labor per period t.
K_n hours required for one ton of production n for each worker.
AR regular working hours for period t.
M_n required hours to produce one ton of product n.
AO overtime working hours allowed during for period t.

4.2 Objective functions

Multiple-objective approaches explicitly treat individual objectives and enable decision maker (DM) to investigate sets of alternative solutions. These approaches also allow the DM to attain more than one objective in selecting a course of action [44]. In the present study, two objective functions were solved simultaneously, that is, to minimize total production and workforce costs.

- Minimize production costs:

$$\text{Min } Z_1 = \sum_{n=1}^{N}\sum_{t=1}^{T} C_{nt}P_{nt} + i_{nt}I_{nt} \tag{1}$$

- Minimize workforce costs:

$$\text{Min } Z_2 = \sum_{t=1}^{T} w_tW_t + h_tH_t + f_tF_t + o_tO_t. \tag{2}$$

4.3 Subject to constraint

- Inventory level constraint:

$$P_{nt} + I_{n(t-1)} - I_{nt} = D_{nt} \quad \forall_n, \forall_t \tag{3}$$

- Production constraint:

$$\sum_{n=1}^{10} Mz_nP_{nt} - AR \times W_t - O_t \le 0 \quad \forall_t \tag{4}$$

- Overtime constraint:

$$O_t - AO \times W_t \le 0 \quad \forall_t \tag{5}$$

- Workforce-level constraint:

$$F_t - H_t + W_t - W_{t-1} = 0 \quad \forall_t \tag{6}$$

- Constraint of non-negativity on decision variables:

$$P_{nt,}, I_{nt,}, O_t, H_t, F_t, W_t \ge 0 \quad \forall n, \forall t$$

5 Model development

This study used fuzzy linear programming (FLP) based on SA and SD algorithms (fuzzy–SASD) to solve actual APP problems.

5.1 Fuzzy approach

In actual APP cases, a DM must frequently handle imprecise goals that regulate resource and capacity limits and usage within the operational areas of organizations. Fuzzy set theory exhibits high applicability in assessing imprecisely defined situations [1]. Zimmerman (1976) was the first to introduce fuzzy set theory into typical linear programming problems. He successively expanded the fuzzy programming methodology to common MOLP problems in 1978 [14].

FLP problems enable the membership functionality of fuzzy decisions and data utilized in sequence based on the trade-offs between the increasing cost of information and benefits of the model. Thus, the solution sets obtained from FLP methods can be realistically appreciated by the DM [19]. Similarly, Zimmerman improved a tolerance approach for resolving FLP problems encountered during allocation of resources and management of production plans [29, 45]. The present study included relevant inaccurate operating costs, data capacities, and forecast demand variables that were covered in Zimmerman's approach. However, the tolerance level for his approach was determined by the DM, depending on their experience, which can result in several solutions for the same problems. Therefore, a new method was utilized to determine the tolerance levels to make it the general means. This method can be applied by any DM, and it can obtain the same outcomes to the same problem and their model. FLP model for Zimmerman's approach is as follows:

$$\text{Min } C \quad X \tilde{\le} Z*$$

Subject to

$$AX \tilde{\le} T, \quad X \ge 0$$

where $\tilde{\le}$ is fuzzy inequalities, T is the tolerance level, and $Z*$ is an aspiration level of the DM. The linear membership function is defined as follows:

$$\mu_k(Z_k) = \begin{cases} 1 & Z_k \le Z_k^* \\ 1 - \dfrac{Z_k - Z_k^*}{T_k} & Z_k^* \le Z_k \le Z_k^* + T_k \\ 0 & Z_k > Z_k^* + T_k \end{cases} \tag{7}$$

where $k = 1, 2,\ldots, K$, which K is the number of objective functions. The application of this technique as a general procedure is as follows:

- *Step 1*
MOLP model for APP problem was composed, as described in Eqs. (1–6). Thereafter, each individual fuzzy objective function was solved to generate optimal solutions for each objective and denote aspiration levels (Z_k*).

- *Step 2*
  Tolerance level (T_k) for each objective was determined by taking the last two small decision variable values from the solution. Afterward, the minimum number was subtracted from the supreme number for each objective function. When generalized in this manner, any DM can utilize the values and achieve similar results. In previous cases, the DM assumed estimations to determine values for tolerance levels (T_k) according to their individual experiences, thereby inducing several solutions to the same problem.
- *Step 3*
  With FLP, the membership functions of fuzzy data and decisions can be applied sequentially on the basis of the trade-offs between the model benefits and increasing information costs [19]. Therefore, we applied the membership function of the Zimmerman approach for each objective to the APP model in Sect. (4.2) according to Eq. (7).

$$\mu_1(Z_1) = \begin{cases} 1 & Z_1 \le Z_1^* \\ 1 - \dfrac{Z_1 - Z_1^*}{T_1} & Z_1^* \le Z_1 \le Z_1^* + T \\ 0 & Z_1 > Z_1^* + T_1 \end{cases} \tag{8}$$

$$\mu_2(Z_2) = \begin{cases} 1 & Z_2 \le Z_2^* \\ 1 - \dfrac{Z_2 - Z_2^*}{T_2} & Z_2^* \le Z_2 \le Z_2^* + T \\ 0 & Z_2 > Z_2^* + T_2 \end{cases} \tag{9}$$

where µ represents membership function, and $\alpha = \mu_D(Z_D) = \min\{\mu_1(Z_1), \mu_2(Z_2)\}$. After solving these equations, the following equations were obtained:

$$\alpha \le 1 - ((Z_1 - Z_1*)/T_1) \tag{10}$$

$$\alpha \le 1 - ((Z_2 - Z_2*)/T_2). \tag{11}$$

After simplifying the equations, we obtained the following:

$$\alpha + Z_1/T_1 \le 1 + \frac{Z_1*}{T_1} \tag{12}$$

$$\alpha + Z_2/Z_2* \le 1 + \frac{Z_2*}{T_2}. \tag{13}$$

- *Step 4*
  The fuzzy decision-making for all the imprecise data of [46] and [47] methods was applied, and the complete equivalent single-objective linear programming model to solve the APP problem can be devised with the following: *Max* α was subjected to the following:

$$\alpha + Z_1/T_1 \le 1 + \frac{Z_1*}{T_1}$$

$$\alpha + Z_2/T_2 \le 1 + \frac{Z_2*}{T_2}$$

$$P_{nt} + I_{n(t-1)} - I_{nt} = D_{nt}$$

$$O_t - AO \times W_t \le 0$$

$$F_t - H_t + W_t - W_{(t-1)} = 0$$

$$\sum_{n=1}^{10} M_n S_{nt} - AR \times W_t - O_t \le 0$$

$$P_{nt}, I_{nt}, O_t, H_t, F_t, W_t \ge 0.$$

5.2 Hybrid SA–SD algorithm

All the forecasted demand variables, data capacities, and associated inexact operating costs were handled through a new method based on Zimmerman's approach. Given that APP belongs to the class of NP-hard, large-scale problems (with a large number of decision variables) cannot be tackled with mathematical programming solvers of the APP model [43]. In [31], metaheuristics show good performance for large-scale problems of APP. However, hybridization of metaheuristics results in robust solution methods. In creating the hybrid model, we considered two important components in modern metaheuristics, namely intensification and diversification (exploration and exploitation). For an efficient and effective (speed and quality) algorithm, it should be able to explore effectively the entire search space and intensify its search around the neighborhood for an optimal or nearly optimal solution. To optimize the speed and quality of any algorithm, exploration and exploitation should be balanced. A successful combination of these two key methodologies usually ensures that global optimality is attainable [35, 48]. The whole solution space is effectively explored by global methods, such as the GA [49], PSO [50], SA [51], and TS [52, 53], to localize the most suitable areas. Alternatively, the local methods of SD algorithm [54] and hill climbing are more effective than those of global methodologies to exploit the most suitable areas that were already identified [55]. The SA algorithm can effectively provide feasible solutions to a diverse range of practical optimization problems [56]. However, this algorithm is rarely applied in the APP field because of its avoidance of local optima by jumping away from them; consequently, its efficiency, that is, running time, is sacrificed [57].

To overcome issues in applying SA algorithm, a local search algorithm was chosen to balance between exploration and exploitation for SA. This paper proposed a new hybrid algorithm that combines the SD algorithm with the SA algorithm to solve the crisp MOLP APP model. We selected the SD algorithm for its local search, robustness, easy programming, and fast search features. Moreover, this algorithm is appropriate for optimizing functions whose derivatives may be expensive to evaluate or are unknown [55, 58]. According to the proposed approach, SA was initially allowed to search for the global optimums of specified objective functions. During the search, the solution was stagnant for a fixed number of iterations, and the SA algorithm was considered trapped into a local optimum. To improve its performance and lessen the deficiencies in problem solving, [59] tried to expand the search space by starting with N + 1 solutions, instead of one solution. The various stages of the recommended algorithm are as follows:

- *Step 1* Starting step.
  - The initial *n* solutions, such as $X_1, X_2, \ldots, X_{n+1}$, *set* $K = 1$, should be generated, and temperature T should be initialized.
- *Step 2* The fitness function for each X_i should be evaluated, and step 4 should be subsequently performed.
- *Step 3* Iteration step.
  - The *m* times should be followed.
  - New solution x_i' should be generated for each x_i by using a neighbor search.
  - If $f(x_i') \le f(x_i)$, then $x_i = x_i'$ and $f(x_i) = f(x_i')$ and then, step c should be conducted.
  - Otherwise, if $p = e^{\left(-\frac{|f(xi') - f(xi)|}{T}\right)} \le Ƶ, Ƶ \in (0, 1)$, then $x_i = x_i'$ and $f(x_i) = f(x_i')$ and step c should be carried out.
  - If $k = m$, then step 3 should be performed. Otherwise, $k = k+1$, and step ***a*** should be followed.
- *Step 4* Sort step.
  - $f(x_1) \le f(x_2) \le \ldots \le f(x_n \le f(x_{n+1})$, where $f(x_1)$ is the most suitable solution, and $f(x_{n+1})$ is the worst solution.
- *Step 5* Reflection step.
  - A point x_r should be determined by $x_r = m + \lambda (m - x_{n+1})$, and *f* should be evaluated for x_r, where *m* is the centroid of the *N* most suitable solutions, $m = \text{mean}(x(1{:}\ n))$, and $\lambda = 1$.
  - If $f(x_1) \le f(x_r) < f(x_n)$, then the worst solution is replaced with a reflected solution, that is, $x_n = x_r$.
- *Step 6* Expansion step.
  - If $f(x_r) < f(x_1)$ then generate a new point x_e by expansion, from $x_e = x_r + \beta(x_r - m)$, where $\beta = 2$.
  - If $f(x_e) < f(x_r)$ then replace x_{n+1} with x_e. else $x_{n+1} = x_r$.
- *Step 7* Contraction step.
  The two kinds of contraction are outside and inside contractions.
  1. Outside contraction:
     - If $f(x_n) \le f(x_r) < f(x_{n+1})$, then a new point x_c should be generated by contraction from $x_c = m + \gamma\ (x_{n+1} - m)$ *and* $\gamma = 0.5$.
     - If $f(x_c) < f(x_r)$, then $x_{n+1} = x_r$.
     - Otherwise, $x_{n+1} = x_r$.
  2. Inside contraction:
     - If $f(x_{n+1}) \le f(x_r)$, then a new point x_c should be generated by contraction from $x_c = m + \gamma\ (m - x_{n+1})$.
     - If $f(x_c) < f(x_r)$, then $x_{n+1} = x_r$.
     - Otherwise, $x_{n+1} = x_r$.
- *Step 8* Shrinkage step.
  - The *f* should be evaluated at the n solution x1, $x_{sj} = x_1 + \sigma\ (x_{sj} - x_1)$, $j = \{2, \ldots, n + 1\}$.
  - The vertices of the simplex in subsequent iteration included $x_1, v_2, \ldots, v_{n+1}$.
- *Step 9* Stopping step.
  - If $T = 0.01$ or $(|\ f(x_{n+1}) - f(x_1)|/f(x_{n+1})) < 1e^{-6}$, then end.
  - Otherwise, the temperature should be reduced by $T_t < \alpha\ T_{(t-1)}$, and step 3 should be followed.

6 Computational study and results

6.1 Case study

The general company for vegetable oil industry was used as a case study to exhibit the proposed model. The products of this company are detergent powder, liquid detergent, vegetable ghee, liquid oil, toilet soap, bay soap, chlorine bleach, shaving cream, shampoo, and toothpaste. To easily write these products in tables, each product is represented by a letter, namely *A*, *B*, *C*, *D*, *E*, *F*, *J*, *H*, *I*, *J*, and *K*, respectively. The periods of APP decision are six months. Tables 1 and 2 represent the costs of production and inventory, hours required to produce one ton for each product, and forecast demand for each product. The initial inventories for products *A*, *E*, *H*, and *J* are 105, 333, 0.25, and 1.8 tons, respectively. The initial worker level is 3313 workers. The cost of a regular worker per month is \$500/man, where the hours worked in one month are 140 h, and \$5.357 is the overtime cost per worker per hour. The

Table 1 Operating costs and data

	A	B	C	D	E	F	G	H	J	K
c_{nt}	328	385	451	1006	801	487	449	1007	496	739
i_{nt}	38	47	53.6	35.511	35.415	24.6	37.75	37.666	58.666	37
M_{n}	92	52.5	69	64	50	121	42	607	172	692

Table 2 Forecast demand for all products

Period	*A*	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*J*	*K*
1	3049.1	53.9	340.6	100	606.4	23.1	1.7	1.2	3.1	0.74
2	1664.1	50.9	708.1	152	482.7	26.5	3.3	2	1.8	1.1
3	1236.4	35.4	700	138	496.8	14.8	7.4	1.7	2.3	0.47
4	782.5	40.8	650	77	429.9	25	8.7	2.5	2.9	0.76
5	914.4	27.5	439	56	324.7	15	21.5	2.4	2.1	2.3
6	652.9	37.9	619.1	50	652.9	12.4	29.1	1.3	2.7	0.71

costs of hiring and firing are \$775 and \$581 dollars per worker, respectively. Hours of overtime allowed during the period are 60 h per period t. The hours of the regular employee per period are 140 h.

All the imprecise data were first tracked with a new approach based on the Zimmerman approach, in which membership functions to obtain the DM's satisfaction level must be within [0, 1]. Therefore, the aspiration and tolerance levels were determined by using the proposed method and used to solve each objective function individually. The aspiration and tolerance levels are $Z_1^* = 7{,}862{,}577$, $T_1 = 1234$, $Z_2^* = \$6{,}635{,}496$, and $T_2 = 207$. The linear membership function for multiple-objective fuzzy approach is $\alpha = 0.862$. Afterward, the crisp objective functions and decision variables were solved by a SASD algorithm.

6.2 Comparative evaluation of the performance of fuzzy–SASD, SA, GA, and PSO

In order to verify the performance of the fuzzy–SASD algorithm in resolving APP problem, we employed the case study data and examined various problem sizes. We considered four different algorithms, namely SA, PSO, GA, and our hybrid fuzzy–SASD. These algorithms were implemented on a PC with an Intel i5 CPU running at 1.8 GHz and with 4 GB RAM. In conducting the GA, SA, and PSO approaches, all imprecise data were first tracked with the new method for Zimmerman's approach. Hence, the APP problem was resolved using the SA, GA, and PSO. However, the parameters set for the GA population size, crossover, mutation rate, and generation were 100, 0.9, 0.05, and 103, respectively. For PSO experimentation, the weights for inertias were in the range of 0.9–0.4. The social (C_1) with cognitive coefficients (C_2) was 2, and the number of particles and iterations was 30 and 1000, respectively. In addition, the set parameters for the SA cooling rate α and the final temperature (T) were 0.95 and 0.001, respectively. Selecting the most applicable algorithm to solve APP problems can be difficult. For PSO, the results were satisfactory, but the method can exhibit longer runtimes compared those of others, as illustrated in Table 3. Conversely, the objective function of the GA and SA approaches provided reasonably adequate solutions that exhibited shorter runtimes than those of PSO, but they were costly. Comparison of these four algorithms showed that in the proposed fuzzy–SASD algorithm, the total costs for the first objective were \$7,093,894.56, and the costs for the second objective were \$5,898,154. In contrast to SA, PSO, and GA methods, the proposed fuzzy–SASD provided the least objective function values and the shortest runtimes.

The proposed fuzzy–SASD method ultimately offered the most practical solution for APP problems because it can generate improved decision-making with shorter runtimes than those of other algorithms. To generalize the proposed fuzzy–SASD algorithm for large-scale APP

Table 3 Results for each algorithm

Algorithm	Z1	Z2	*T* (s)
GA	7,560,278.12	6,790,206.86	1.3
SA	7,159,635.69	6,418,976.38	1.2
PSO	7,190,959.95	6,144,191.46	2.5
Fuzzy–SASD	7,093,894.5	5,898,154	0.8

Table 4 Parameters and statistical results of Lingo

Test no.	No. of products	No. of period	Lingo opt. z1	Lingo opt. z2
1	10	12	40,147,510	54,275,800
2	15	12	60,213,880	52,630,800
3	20	12	79,487,950	52,154,850
4	30	12	112,825,000	52,579,590
5	35	12	136,630,100	53,882,760
6	40	12	154,533,400	51,037,770

problems, we employed an artificial dataset comprising six problems with medium and large sizes and significantly equal constraints. Comparison of the performances of these algorithms showed that these examples were similarly encoded and resolved using the Lingo optimizing solver to determine the global optima and low bounds (LB) for each objective function. To represent the differences in performance among global optima and the objective functional values of the results of these algorithms, the quality measurement, that is, the percent deviation symbolized by %Dev, was expressed as follows:

$$\%Dev_{z1} = \frac{BOV_{fuzzy-sasd,SA,PSO,GA;z1} - LB_{Lingoz1}}{LB_{Lingoz1}} \times 100\%$$

$$\%Dev_{z2} = \frac{BOV_{fuzzy-sasd,SA,PSO,GA;z2} - LB_{Lingoz2}}{LB_{Lingoz2}} \times 100\%$$

where BOV (fuzzy–SASD, SA, GA, or PSO) is the value of objective obtained from fuzzy–SASD, SA, GA, and PSO, and LB (Lingo) is the low bound originated from Lingo for each objective.

During experimentation using the fuzzy–SASD algorithm, SA, PSO, and the GA, each of the six instances was executed in 30 trials and 1000 iterations as its stopping criterion. The statistical outcomes for each instance were collected solely from the most effective runs found in the 30 trials. The parametric sets of the SA, PSO, SA, and GA retained their previously noted similarities. The outcomes of experiments attained by Lingo, SA, fuzzy–SASD, PSO, and GA are depicted in Tables 4 and 5. A test number was assigned to each instance of the APP problem. Table 4 also contains the main parameters of each test instance of the APP problem, which are the number of products and periods. The first three instances were medium in size, and the last three ones were large-sized. Furthermore, this parameter depicted the Lingo LB. Table 5 illustrates the statistical results of applying fuzzy–SASD, SA, PSO, and GA in the trials. The results included the mean, most suitable, standard deviation, and percent deviation of their values according to the objectives. Runtimes were similarly incorporated in the tabular information. The statistical results demonstrated that the values for the most suitable objective function to each example were differentiated from their respective optimal values. Each case also displayed certain differences in their standard deviation, mean, percent deviation, and time of execution.

Comparable performance values of each algorithm are summarized in Table 6. The fuzzy–SASD provided better average standard deviations, percent deviations, and runtimes for all objective functions than those of SA, PSO, and GA. Assuming that the fuzzy–SASD outcome was 100%, the percent deviation gains were increased by 83, 33, and 89% for SA, PSO, and GA, respectively. The search discrepancies of the particle standard deviation corresponded to values 24, 77, and 64% less than those of SA, PSO, and GA, respectively.

Comparative analyses of all the various performance parameter values for fuzzy–SASD, GA, PSO, and SA are illustrated in Figs. 2 and 3. The average deviation percentages for each objective in each algorithm are presented in Fig. 2. Lingo optima results showed that the fuzzy–SASD algorithm demonstrated the lowest deviation percentages among all algorithms. Furthermore, computational runtimes for the fuzzy–SASD algorithm were consistent with its superior appropriateness over other algorithms, as shown in Fig. 3. In all problem cases, the fuzzy–SASD algorithm required shorter time to solve those instances relative to those of other algorithms. Additionally, the proposed fuzzy–SASD approach was more applicable to a wider range of the information for decision-making than those of other approaches, which largely emphasized multiple-objective APP problems. We concluded that fuzzy–SASD algorithm presented variants in feature accuracies, reliabilities, and convergence speeds in the optimization of APP problems, which are superior to those of the PSO, SA, and GA.

Table 5 Statistical results for all algorithms

Algorithm	Test no.	No.obj.	Best obj.	Mean	SD	% Dev	Com. time (s)
Fuzzy–SASD	1	Z1	28,747,819.62	39,995,238	7,671,995	0.37928	2.051042
		Z2	52,425,937.67	54,677,633	5,552,682	0.740354	
	2	Z1	41,771,447.3	60,466,948	9,083,679	0.51777	1.134375
		Z2	40,486,841.94	51,811,938	6,051,944	0.1107	
	3	Z1	55,681,621.82	79,447,366	10,936,781	0.051057	1.278646
		Z2	53,148,299.73	52,725,469	6,845,011	1.094086	
	4	Z1	89,964,226.68	1.17E+08	13,674,939	3.647045	2.182292
		Z2	58,711,531.44	52,461,831	5,943,123	0.223963	
	5	Z1	110,766,178.3	1.38E+08	12,277,269	0.846831	2.266667
		Z2	62,028,179.81	54,136,052	8,374,962	0.47008	
	6	Z1	121,951,104.2	1.55E+08	14,876,834	0.219584	2.292188
		Z2	47,529,475.7	51,409,644	7,503,551	0.219584	
Avg.		Z1			11,409,391	0.3357	1.45
		Z2			6,085,437	0.6122	
PSO	1	Z1	28,484,836.56	39,790,124	7,681,089	0.89018	34.74479
		Z2	63,626,962.56	60,878,994	7,380,405	12.166	
	2	Z1	42,340,931.19	60,301,075	9,351,528	0.698,386	19.61979
		Z2	48,050,806.49	57,216,801	6,627,862	9.36389	
	3	Z1	56,211,977.91	79,396,767	10,972,625	0.114713	19.55521
		Z2	56,024,117.38	57,926,701	6,619,469	11.06676	
	4	Z1	91,025,188.18	1.17E+08	13,772,925	3.978249	35.50781
		Z2	62,242,302.34	57,089,204	6,200,906	8.576738	
	5	Z1	110,788,243.4	1.38E+08	12,371,826	1.038194	35.70625
		Z2	63,858,918.11	58,369,440	8,208,574	8.326746	
	6	Z1	130,715,333.7	1.55E+08	14,927,381	0.237754	36.03594
		Z2	63,838,857.07	54,467,030	6,747,543	6.719063	
	Avg.	Z1			11,473,815	0.4766	2.56
		Z2			7,041,287.507	1.0434	
SA	1	Z1	28,812,177.72	40,142,480	7,659,850	0.01253	15.67708
		Z2	57,836,077.18	55,654,116	6,465,319	2.539468	
	2	Z1	42,096,077.09	60,695,570	9,320,722	1.11064	9.329688
		Z2	42,703,808.3	51,435,603	5,531,342	0.07795	
	3	Z1	56,515,106.84	79,836,011	10,976,262	0.478789	10.8026
		Z2	52,861,204.69	53,089,019	6,763,332	1.791145	
	4	Z1	90,243,160.44	1.18E+08	13,707,913	4.147242	24.775
		Z2	58,711,531.44	52,113,125	6,206,317	0.88716	
	5	Z1	111,067,544.3	1.38E+08	12,391,843	1.200831	26.01563
		Z2	62,028,179.81	54,103,907	8,730,223	0.410423	
	6	Z1	130,998,417.3	1.55E+08	14,987,986	0.48096	28.08438
		Z2	62,560,274.55	51,757,161	7,413,622	1.409526	
	Avg.	Z1			11,474,834	1.238499	1.7966
		Z2			7,186,150	1.185945	
GA	1	Z1	28,643,531.17	39,960,186	7,668,837	0.46659	93.35208
		Z2	61,832,225.04	59,656,705	7,112,881	9.914004	
	2	Z1	41,961,864.7	60,527,998	9,335,466	0.881016	51.71458
		Z2	46,409,018.33	55,638,903	6,166,056	6.452161	
	3	Z1	56,444,240.13	79700555	10,987,842	0.746794	53.67969
		Z2	54,497,796.72	56,019,042	6,542,154	7.7.409076	

Table 5 continued

Algorithm	Test no.	No.obj.	Best obj.	Mean	SD	% Dev	Com. time (s)
	4	Z1	90,338,598.62	1.17E+08	13,736,680	4.026205	95.22552
		Z2	59,997,192.58	54,710,454	5,823,265	4.052644	
	5	Z1	111,220,526.2	1.38E+08	12,301,748	1.176098	95.76875
		Z2	62,548,387.98	56,287,558	8,179,009	4.46302	
	6	Z1	131,372,563.2	1.55E+08	14,846,955	0.493563	96.4625
		Z2	62,948,717.91	53,908,841	6,736,797	5.625385	
	Avg.	Z1			1144575	0.5523	2.37
		Z2			693999	1.6009	

Table 6 Comparative performance evaluation of fuzzy–SASD, GA, PSO, and SA

algorithm	% Dev	Com. time (s)
Fuzzy–SASD	100	100
GA	189	164
PSO	133	177
SA	183	124

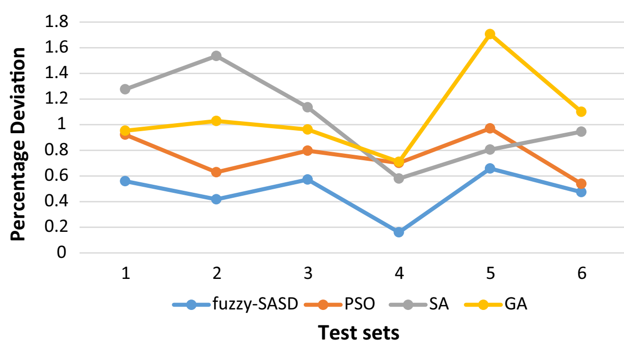

Fig. 2 Comparative performance evaluation for all datasets

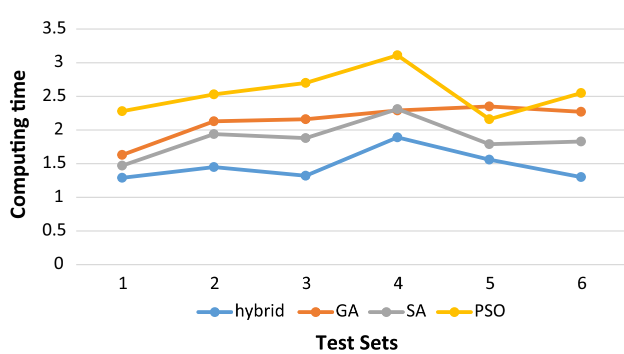

Fig. 3 Comparative runtime for each algorithm

7 Conclusion

In this paper, a new hybrid fuzzy–SASD optimization method was recommended to address the MOLP model for APP problems. SA has emerged as a popular and reliable approach for solving large complex problems, including scheduling, timetabling, and traveling salesman problems. However, few studies on SA have discussed uncertainties noted in APP problems. Hence, the present research provided a novel fuzzy–SASD approach for addressing an intricate a multiple-objective APP problem. This problem involved the minimization of total production and workforce-level costs. This research utilized the SD algorithm to provide a balance between exploitation and exploration for the SA algorithm, thereby providing an efficient convergence speed in optimizing the APP problem. On the contrary, a new method based on Zimmerman's approach was utilized to determine uncertainties associated with model parameters. An industrial case was exhibited to validate the viability of the proposed approach. This algorithm can be applied in vague and unspecified conditions of actual APP and scheduling problems through imprecise data. To improve the validation of the proposed fuzzy–SASD algorithm, its performance was compared with those of PSO, SA, and GA. The results indicated that the fuzzy–SASD algorithm variant more precisely solved APP problems than PSO, SA, and GA.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Jamalnia A, Soukhakian MA (2009) A hybrid fuzzy goal programming approach with different goal priorities to aggregate production planning. Comput Ind Eng 56(4):1474–1486

2. Saad GH (1982) An overview of production planning models: structural classification and empirical assessment. Int J Prod Res 20(1):105–114
3. Holt CC, Modigliani F, Simon HA (1955) A linear decision rule for production and employment scheduling. Manag Sci 2(1):1–30
4. Singhal K, Adlakha V (1989) Cost and shortage trade-offs in aggregate production planning. Decis Sci 20(1):158–165
5. Bowman EH (1956) Production scheduling by the transportation method of linear programming. Oper Res 4(1):100–103
6. Bowman EH (1963) Consistency and optimality in managerial decision making. Manag Sci 9(2):310–321
7. Byrne M, Bakir M (1999) Production planning using a hybrid simulation- analytical approach. Int J Prod Econ 59(1):305–311
8. Taubert WH (1968) A search decision rule for the aggregate scheduling problem. Manag Sci 14(6):B-343
9. Wang R-C, Liang T-F (2004) Application of fuzzy multi-objective linear programming to aggregate production planning. Comput Ind Eng 46(1):17–41
10. Al-e SMJM, Aryanezhad MB, Sadjadi SJ et al (2012) An efficient algorithm to solve a multi-objective robust aggregate production planning in an uncertain environment. Int J Adv Manuf Technol 58(5–8):765–782
11. Kazemi Zanjani M, Ait-Kadi D, Nourelfath M (2013) A stochastic programming approach for sawmill production planning. Int J Math Oper Res 5(1):1–18
12. Mirzapour Al-e Hashem S, Baboli A, Sazvar Z (2013) A stochastic aggregate production planning model in a green supply chain: Considering flexible lead times, nonlinear purchase and shortage cost functions. Eur J Oper Res 230(1):26–41
13. Iris C, Cevikcan E (2014) A fuzzy linear programming approach for aggregate production planning. In: Supply Chain Management Under Fuzziness, Springer, pp 355–374
14. Liang T-F, Cheng H-W, Chen P-Y, Shen K-H (2011) Application of fuzzy sets to aggregate production planning with multi-products and multi-time periods. IEEE Trans Fuzzy Syst 19(3):465–477
15. Kogut B, Kulatilaka N (1994) Operating flexibility, global manufacturing, and the option value of a multinational network. Manag Sci 40(1):123–139
16. Tang J, Wang D, Fung RY (2000) Fuzzy formulation for multi-product aggregate production planning. Prod Plan Control 11(7):670–676
17. Wang R-C, Liang T-F (2005) Aggregate production planning with multiple fuzzy goals. Int J Adv Manuf Technol 25(5–6):589–597
18. Tavakkoli-Moghaddam R, Rabbani M, Gharehgozli A Zaerpour N (2007) A fuzzy aggregate production planning model for make-to-stock environments. In: 2007 IEEE International conference on industrial engineering and engineering management, IEEE, pp 1609–1613
19. Wang R-C, Fang H-H (2001) Aggregate production planning with multiple objectives in a fuzzy environment. Eur J Oper Res 133(3):521–536
20. Fung RY, Tang J, Wang D (2003) Multiproduct aggregate production planning with fuzzy demands and fuzzy capacities. IEEE Trans Syst Man Cybern Part A Syst Hum 33(3):302–313
21. Yaghin RG, Torabi S, Ghomi SF (2012) Integrated markdown pricing and aggregate production planning in a two echelon supply chain: a hybrid fuzzy multiple objective approach. Appl Math Model 36(12):6011–6030
22. Omar MK, Jusoh MM, Omar M (2012) Investigating the benefits of fuzzy mathematical programming approach for solving aggregate production planning. In: 2012 IEEE International conference on fuzzy systems (FUZZ-IEEE), IEEE, pp 1–6
23. da Silva AF, Marins FAS (2014) A fuzzy goal programming model for solving aggregate production-planning problems under uncertainty: a case study in a Brazilian sugar mill. Energy Econ 45:196–204
24. Sisca FG, Fiasch_e M, Taisch M (2015) A novel hybrid modelling for aggregate production planning in a recon_gurable assembly unit for optoelectronics. In: International conference on neural information processing, Springer, 2015, pp 571–582
25. Gholamian N, Mahdavi I, Tavakkoli-Moghaddam R (2016) Multi-objective multi-product multi-site aggregate production planning in a supply chain under uncertainty: fuzzy multi-objective optimization. Int J Comput Integr Manuf 29(2):149–165
26. Baykasoglu A, Gocken T (2010) Multi-objective aggregate production planning with fuzzy parameters. Adv Eng Softw 41(9):1124–1131
27. Baykaso_glu A, Gocken T (2006) A tabu search approach to fuzzy goal programs and an application to aggregate production planning. Eng Optim 38(2):155–177
28. Stockton D, Quinn L (1995) Aggregate production planning using genetic algorithms. Proc Inst Mech Eng Part B J Eng Manuf 209(3):201–209
29. Wang D, Fang S-C (1997) A genetics-based approach for aggregated production planning in a fuzzy environment. IEEE Trans Syst Man Cybern Part A Syst Hum 27(5):636–645
30. Tavakkoli-Moghaddam R, Safaei N (2006) An evolutionary algorithm for a single-item resource-constrained aggregate production planning problem. In: IEEE Congress on evolutionary computation, 2006, CEC 2006, IEEE, pp 2851–2858
31. Raa B, Dullaert W, Aghezzaf E-H (2013) A matheuristic for aggregate production- distribution planning with mould sharing. Int J Prod Econ 145(1):29–37
32. Baykasoglu A (2001) Moapps 1.0: aggregate production planning using the multiple-objective tabu search. Int J Prod Res 39(16):3685–3702
33. Pradenas L, Penailillo F, Ferland J (2004) Aggregate production planning problem. a new algorithm. Electron Notes Discrete Math 18:193–199
34. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82
35. Yang X-S (2014) Nature-inspired optimization algorithms, 1st edn. Elsevier, UK. p 300
36. Kumar GM, Haq AN (2005) Hybrid geneticant colony algorithms for solving aggregate production plan. J Adv Manuf Syst 4(01):103–111
37. 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 Syst Appl 39(1):1256–1263
38. Wang S-C, Yeh M-F (2014) A modified particle swarm optimization for aggregate production planning. Expert Syst Appl 41(6):3069–3077
39. Madadi N, Wong KY (2014) A multiobjective fuzzy aggregate production planning model considering real capacity and quality of products. Math Probl Eng 2014:1–15
40. Aliev RA, Fazlollahi B, Guirimov B, Aliev RR (2007) Fuzzy-genetic approach to aggregate production-distribution planning in supply chain management. Inf Sci 177(20):4241–4255
41. Chakrabortty RK, Hasin MAA (2013) Solving an aggregate production planning problem by fuzzy based genetic algorithm (fbga) approach. Int J Fuzzy Logic Syst (IJFLS) 3(1):1–16
42. Ning Y, Tang W, Zhao R (2006) Multiproduct aggregate production planning in fuzzy random environments. World J Model Simul 2(5):312–321
43. Chakrabortty RK, Hasin MAA, Sarker RA, Essam DL (2015) A possibilistic environment based particle swarm optimization for aggregate production planning. Comput Ind Eng 88:366–377
44. Rosen LD, Canel C (2009) Aggregate production planning a multiple objective approach. J Stat Manag Syst 12(2):287–303
45. Lai Y, Hwang CL (1992) Fuzzy mathematical programming, 1st edn. Springer, Berlin

46. Bellman RE, Zadeh LA (1970) Decision-making in a fuzzy environment. Manag Sci 17(4):141
47. Zimmermann H-J (1978) Fuzzy programming and linear programming with several objective functions. Fuzzy Sets Syst 1(1):45–55
48. Yang X-S (2010) Nature-inspired metaheuristic Algorithms, 2nd edn. Luniver press, Beckington
49. Holland JH (1994) Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. Complex adaptive systems. Q Rev Biol 69(1):88–110
50. Eberhart RC, Kennedy J et al. (1995) A new optimizer using particle swarm theory. In: Proceedings of the sixth international symposium on micro machine and human science, Vol 1, New York, NY, pp 39–43
51. Kirkpatrick S, Gelatt CD, Vecchi MP et al (1983) Optimization by simulated annealing. Science 220(4598):671–680
52. Glover F (1989) Tabu search-part i. ORSA J Comput 1(3):190–206
53. Glover F (1990) Tabu searchpart ii. ORSA J Comput 2(1):4–32
54. Nelder JA, Mead R (1965) A simplex method for function minimization. Comput J 7(4):308–313
55. Chelouah R, Siarry P (2005) A hybrid method combining continuous tabu search and nelder—mead simplex algorithms for the global optimization of multiminima functions. Eur J Oper Res 161(3):636–654
56. Musharavati F, Hamouda A (2012) Enhanced simulated-annealing-based algorithms and their applications to process planning in reconfigurable manufacturing systems. Adv Eng Softw 45(1):80–90
57. Press WH, Teukolsky SA, Vettering WT, Flannery BP (1992) Numerical recipes in c-2'nd edition. p 22
58. Sarakhsi MK, Ghomi SF, Karimi B (2016) A new hybrid algorithm of scatter search and nelder{mead algorithms to optimize joint economic lot sizing problem. J Comput Appl Math 292:387–401
59. Abu Bakar MR, Bakheet AJK, Kamil F, Kalaf BA, Abbas IT, Soon LL (2016) Enhanced simulated annealing for solving aggregate production planning. Math Probl Eng 2016:1–9