APPLICATION OF THE DOWNHILL SIMPLEX ALGORITHM FOR SOLVING AGGREGATE PRODUCTION PLANNING PROBLEMS

Abdul Jabbar K. Bakheet [2], Bayda Atiya kalaf [1,4,*], Batool Atiya [3]

[1] *University Putra Malaysia, Faculty of science, Department of Mathematics*

[2] *University of Baghdad, Faculty of Administration and Economics, Department of Statistic*

[3] *University of Baghdad, Faculty of Administration and Economics, Department of Industrial Management*

[4] *University of Baghdad, Faculty of Education for Pure Science /Ibn AL-Haytham, Department of Mathematics*

* Corresponding addresses: hbama75@yahoo.com

ABSTRACT: *Simplex downhill algorithm (SDA) is a direct search method that uses geometric relationships to aid in finding approximate solutions to complex and NP-hard optimization problems. Due to aggregate production planning belongs to the class of NP-hard problems in production planning, in this study employs SDA for solving multi-objective linear programming model for aggregate production planning problems. The proposed model minimizes total production and work-force costs simultaneously. This study is the first attempt to solve an APP problem by using SDA. The experimental results demonstrate that the SDA was efficient and faster than genetic algorithm (GA).*

Keyword: Aggregate Production Planning, Simplex Downhill Algorithm, Genetic Algorithm.

1 INTRODUCTION

Aggregate production planning (APP) is an operational plan for a production process in advance of 3 to 18 months. In the field of operations planning, APP falls between the broad decisions of long-range planning and the highly specific and detailed short range planning decisions of production and operations management. Relevant planning forms that involve a master production schedule, material requirement planning, and capacity requirement planning depend on APP decisions in a hierarchical way [1- 3].

Numerous APP models and solutions with various degree of sophistication have been introduced since early 1950. According to [4], all traditional approaches of APP problems may be classified into six categories as follows: linear programming (LP) [5], linear decision rule (LDR)[6], transportation method [7], management coefficient approach [8], search decision rule [9], simulation [10].

However, considering all practical parameters in an APP model makes the model difficult and non-optimally solvable. In recent decades, depending on more as assumptions made and advanced modeling approaches invented, the APP problem has become quite complex and large scale. There is a trend in the research community to solve the large complex problems by using meta-heuristic optimization techniques. This is mainly due to the time-consuming and unsuitability of classical techniques in many circumstances. Meta-heuristic algorithms, such as tabu search (TS)[11], genetic algorithm (GA) [12], simulated annealing algorithm (SA)[13], harmony search algorithm (HS) [14], and particle swarm optimization algorithm (PSO) [15], have been developed to solve APP.

[16] proposed (FAIHSA) mechanisms on an improved HSA to solve a multi-objective APP problem. Esmaeil and Amir [17] introduced HSA to solve two mixed-integer linear programming models for APP system with return products and machine breakdowns. Ramazanian et al.[18] reformulated a multi-objective non-linear programming model to single objective, in which PSO algorithm is used to solve the model. [19] modified PSOA for integer linear programming to APP, and a modified operation procedure of particles to the modernization rules was employed to govern the search processes for a particle swarm. Rakibul et al. [20] applied PSO for constrained optimization of APP under vague demand.

In 2001 [21] extended Masud's model (1980) [22] by including subcontracting and setup decisions and considered multiple-objective tabu search algorithm which was proposed by [23] and [24]. Likewise, Kumar and Haq adopted the genetic algorithm (GA), ant colony algorithm (AGA), and hybrid genetic-ant colony algorithm (HGA) to solve an APP problem. Based on the results that were obtained, GA and HGA displayed relatively good performance [25].

Although SDA is a heuristic direct-search method [26] and still a method of choice for many practitioners in the fields of statistics, engineering, and the physical and medical sciences because it is easy to code and very easy to use [27], until now SDA has not been applied for solving APP problems.

In this paper, SDA was introduced at the first tie to solve a multi-objective linear programming model for APP problems. The rest of the paper is organized as follows. Section 2 describes the simplex downhill algorithm. Section 3 presents the mathematical model. Section 4 introduces the solution procedure. Numerical results and discussion are presented in Section 5. Section 6 concludes the paper.

2 Simplex Downhill Algorithm

Simplex downhill algorithm (SDA) is a mathematical method that uses geometric relationships to aid in finding approximate solutions to solve complex problems and applies a simplex structure to attain the optimization function. The basis of SDA was introduced in 1962 [28], but Nelder and Mead [29] adopted and modified this algorithm in 1965 to modern form. which is also called Nelder-Mead or Amoeba. The bene t of this method is it does not require an evaluation of the derivative of the function. but only guesses number of solutions for each decision variable.

The concept simplex in SDA is quite different from Dantzigs simplex method for linear programming. Simplex is a simple geometric shape de ned by the convex hull of N + 1 vertices in N-dimensional space.

In 2 dimensions, a simplex represents a triangle, while in 3 dimensions it represents a tetrahedron as we show in figure 1.

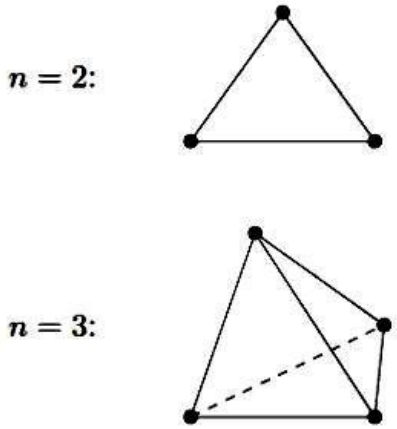

Figure 1: Geometric shapes of simplex downhill algorithm

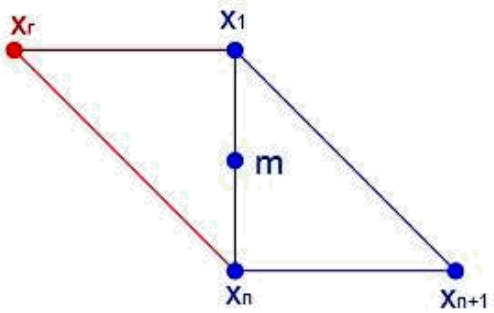

Figure 2:Reflection step

The idea of SDA generates a sequence of simplice which converges to minimize. That means we generate N + 1 points (vertex) in an N-dimensional space. then the vertices sorted by ascending order such us: $f(x_1) \le f(x_2) \le \ldots \le f(x_n) \le f(x_{n+1})$, where x_{n+1} is worse point (solution) and x_1 best point(solution). The objective function value is determined at each of these points. The algorithm iteration updates the worst point by four operations: reflection, expansion, contraction, and shrinkage. And we will explain details for each operation:

- Reflection:

compute the reflection point x_r from $x_r = m + \lambda (m - x_{n+1})$ and evaluate f for x_r, where m is the centroid of the N best points in the vertices of the simplex;

m = mean (x (1 : n)) and $\lambda = 1$.

If $f(x_1) \le f(x_r) < f(x_n)$, then replace the worst point with a reflected point $x_{n+1} = x_r$. As we shown in Fig.2.

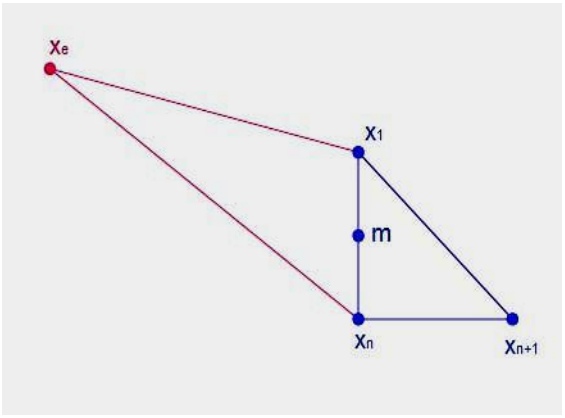

Figure 3: Expansion step

Expansion:

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 $x_{n+1} = x_r$. Figure 3 illustrate expansion step.

Contraction:

We have two kind of contraction, outside contraction and inside contraction.

Outside Contraction: If $f(x_n) \le f(x_r) < f(x_{n+1})$, then generate a new point x_c by contraction, from $x_c = m + \gamma (x_r - m)$ and $\gamma = 0.5$.

- If $f(x_c) < f(x_r)$, then replace x_{n+1} with x_c.
- else $x_{n+1} = x_r$.

- **Inside contraction**: If $f(x_{n+1}) \le f(x_r)$, generate a new point x_c by $x_c = m + \gamma (m - x_{n+1})$.

- If $f(x_c) < f(x_r)$ then $x_{n+1} = x_c$.
- else $x_{n+1} = x_r$.

As we shown in Figure 4

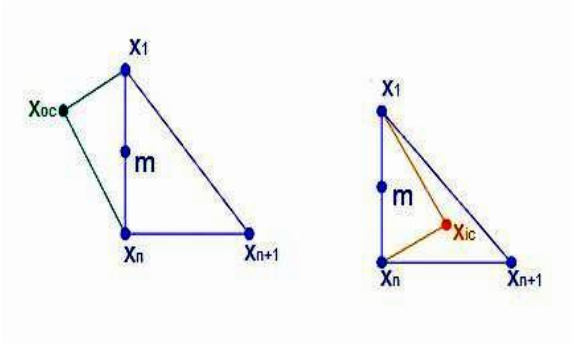

Figure 4:Outside and Inside Contraction step

If the three steps in above are fails, then a shrinkage step is used.

- **Shrinkage Step:**

If the three operations in above are fails then shrinkage step is used shrinkage start with calculate the n new vertices, and just keep the best one x_1 and

$x_{sj} = x_1 + \sigma (x_{sj} - x_1)$, $j = \{2 \ldots n+1\}$ and $\sigma = 0.5$. The (unordered) vertices of the simplex at the next iteration consist of x_1, x_{s2}, …, $x_{s\,n+1}$. we can see this step in Figure 5.

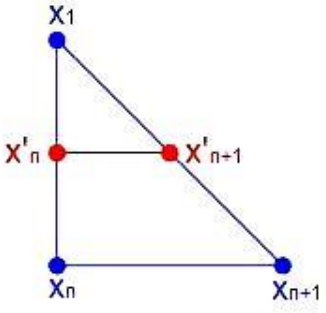

Figure 5: Shrinkage Step

5

3 Problem formulation

The mathematical model of multi-objective linear programming for aggregate production planning is proposed, and describe descried below. Assume that a company produced N types of products to fulfill market demand over the planning horizon T. The objectives of this APP decision are to minimize total production costs and workforce costs.

3.1 Notation

n : number of products, $n = 1,2,\ldots, N$.

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

.c_{nt} : production cost per ton of product n per period t, (dolar /ton).

i_{nt} : inventory carrying cost per ton of product n per period t, (dolar/ton).

h_t : hiring cost per worker in period t, (dolar, worker).

f_t : firing cost per worker in period t, (dolar, worker).

o_t : cost per man- hour of overtime labor per period t.

w_t : cost of regular labor per period t.

D_{nt} : forecasted demand for product n per period t, (tons).

H_{max} : Maximum hiring in each period.

F_{max}: Maximum firing in each period.

M_n : hours required to produce one ton of product n.

A_R : working regular hours per period t.

A_O working overtime hours, which allowed during per period t.

K_n : hours required to produce one ton for product n per workers.

P_{nt} : production of product n per period t, (tons).

I_{nt} : inventory level of product n per period t, (tons).

O_t : man-hours of overtime labor per period t.

W_t: workforce level per period t, (workers).

H_t : hired workers per period t, (workers).

F_t : fired workers per period t, (workers).

Objective function:

- Minimize production costs.

$$\text{Min } Z_1 = \sum_{n=1}^{10} \sum_{t=1}^{6} C_{nt} P_{nt} + i_{nt} I_{nt} \quad (1)$$

- Minimize workforce costs.

$$\text{Min } Z_2 = \sum_{t=1}^{6} w_t W_t + h_t H_t + f_t F_t + o_t O_t \quad (2)$$

Constraint

- inventory level constraint:

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

- Workforce level constraints:

$$F_t - H_t + W_t - W_{t-1} = 0 \qquad \forall_t$$

$$F_t \le F_{max}$$

$$H_t \le H_{max}$$

Overtime constraint:

$$O_t - AO * W_t \le 0 \qquad \forall_t$$

- Production constraint

$$\sum_{n=1}^{10} M_n P_{nt} - AR * W_t - O_t \le 0 \qquad \forall_t$$

- non-negativity constraint

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

4 Solution approach

This study proposes simplex downhill algorithm to use multi-objective linear programming model for APP problems.

The SDA procedure can be summarized as follows:

Step 1: n = number of parameters (decision variables) required for output;

let X is a vector for all decision variables such as

$X = [P_{nt} I_{nt} H_t F_t W_t O_t]$ and generate $n+1$ solutions for X.

Step 2: Find the objective function for each X_i, $i = 1 \ldots, n+1$,

Sort the result of $f(x_i)$ such that $f(x_1) \le f(x_2) \le \ldots \le f(x_n) \le f(x_{n+1})$.

Step 3: Generate a trial point x_r by reflection,

If $f(x_1) \le f(x_r) < f(x_n)$, then $x_{n+1} = x_r$ and go to step 5.

else If $f(x_r) < f(x_1)$ then generate a new point x_e by expansion,

If $f(x_e) < f(x_r)$ then $x_{n+1} = x_e$ and go to step 5.

else replace x_{n+1} with x_r and go to step 5.

else $f(x_n) \le f(x_r)$, generate a new point x_c by contraction,

If $f(x_c) < f(x_r)$, then replace x_{n+1} with x_c. go to step 5.

Step 4: shrinkage step. calculate the n new vertices, and just keep the best one x_1, $x_j = x_1 + \sigma (x_j - x_1)$; j=1, …, n.

Step 5: stopping step: if $| f(x_{n+1} - f(x_1) | / f(x_{n+1} < e^{-6})$ then end else go to step 2.

In addition, SD algorithm depicting in Fig. 6

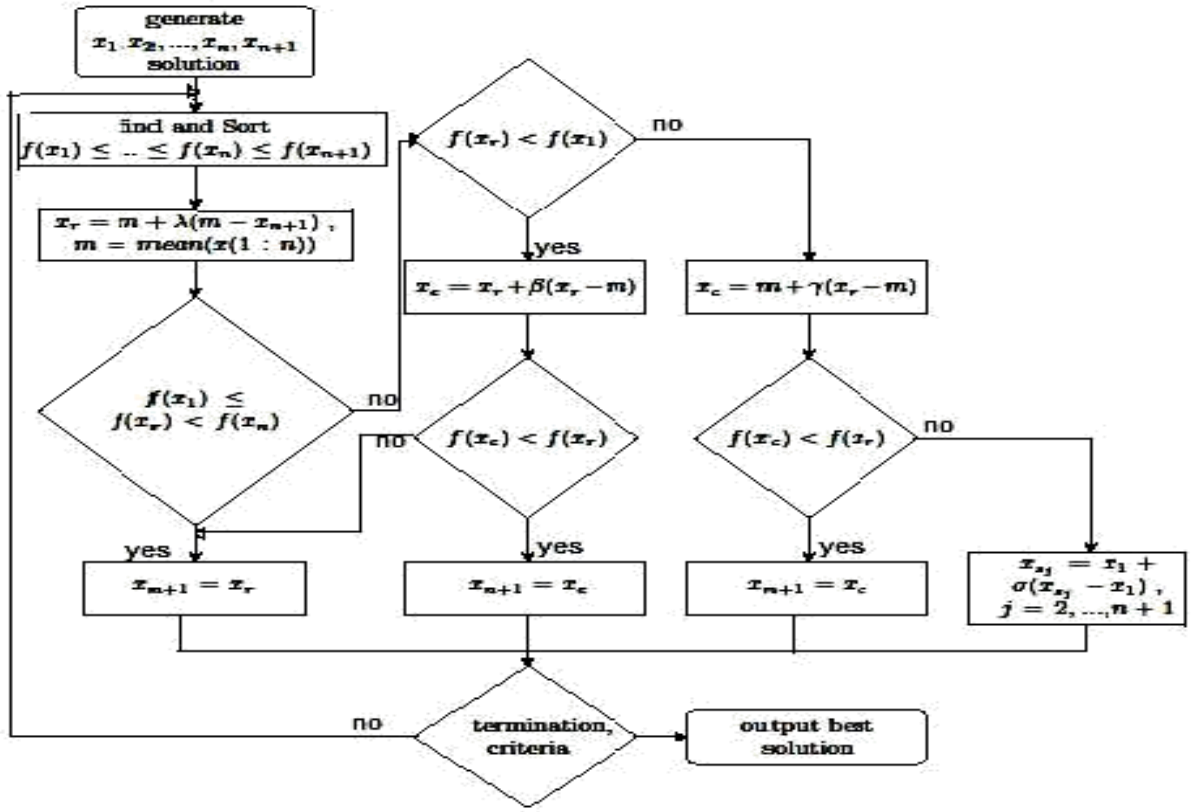

Figure 6: Flowchart for SDA procedure

5 Computational study

5.1 case study

The proposed model is applied in the General Company for Vegetable Oils. This company of the most important companies in the Arab world to vegetable oils industry. It produces ten types of products. We represented for each product by letter, A=solid detergent, B= liquid detergent, C = fat solid, D= liquid oil, E = toilet soap, F =liquid soap, G=detergent bleach, H =shaving cream, J =shampoo, and K = toothpaste. The time horizon of APP decision six months. The production costs, inventory costs forecast demand for each product were summarized in tables 1 3 respectively. The initial inventory for solid detergent is105 tons, toilet soap is333 tons, shaving cream is 0:25 tons and shampoo is 1:8 tons. The initial worker level is 3313 workers. The costs of regular worker per month is 500 dollar /man, the hours worked in the in one month is 140 hours. 5:357 is overtime costs per worker in hour. The costs associated with hiring and ring are 774:910 and 581:182 per worker, respectively. Hours of overtime, which allowed during the period is 60 hours per period t. hours of regular worker per period is140 hours.

Table 1:Production and inventory costs in dollar

	A	B	C	D	E	F	G
^{c}nt	3285	385.082	450.7	1005.776	800.834	486.999	449.419
^{i}nt	38	47	53.6	35.511	35.415	24.6	37.75

Table 2: Hours required to produce one ton of product

Product	A	B	C	D	E	F	G	H	J	K
Hours	92	525	69	64	50	121	42	607	172	692

Table 3:Forecast demand for all products

Period	A	B	C	D	E	F	G	H	J	K
1	3049.1	539	340.6	100	606.4	23.1	1.7	1.2	3.1	0/74
2	1664.1	509	708.1	152	482.7	265	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	275	439	56	324.7	15	215	2.4	2.1	2.3
6	652.9	379	619.1	50	652.9	12.4	29.1	1.3	2.7	0.71

5.2 Evaluation of the SDA with Other Algorithms

In order to evaluate the performance of the SD algorithm for the APP problem, we made use of that case study data. For better comparison, we considered three different algorithms; Genetic Algorithm (GA), harmony search (HS), and SD algorithm. The algorithms were Coded with Mat lab R2015a and were executed on a PC with an Intel Core i5 processor, 4.00 GB RAM, and a 1.80 GHz CPU.

To carry out the GA, we set some parameters for GA of the population size, the crossover rate, the mutation rate and the generations as 100, 0.9, 0.05 and 103 respectively. choosing the best algorithm for solving APP problems is not an easy task.

Table 4 summered the results for algorithms SDA and GA. This table indicated that, SDA provided better results (7111902 and 577053) than GA (7560276 6790206) for each objective function. In addition, Therefore, the results of GA were quite satisfactory thought it exhibit longer runtime (33) compared with SDA (12). Therefore, SDA can solve APP problems through an interactive decision-making process. Furthermore, objective function value and run time for SDA best than GA as shown in table 4. Thus, SDA was adopted in the implementation of the model in the company. Tables 5 to 7 present the solutionsfor each decision variable for a multi-objectives linear programming to APP problem. Finally, we conclude that the SDA provided better results for APP decision problems in terms of cost and time than GA.

Table 4:Results for each algorithm

algorithm	Z1	Z2	time
SDA	7111902	577053	12 s.
GA	7560278	6790206	33 s.

Table 5: Production yield

product	P1	P2	P3	P4	P5	P6
A	2944.82	1664.1	1236.4	782.5	914.4	652.9
B	53.9	50.9	35.4	40.8	27.5	37.9
C	341.6	708.02	700	650.2	439	619.02
D	100	152	138	77	56	50
E	144.071	482.7	496.8	429.9	324.7	652.9
F	23.101	26.501	14.801	25	14.7563	13.1611
G	1.0934	2	1.7	2.5	3.2532	0.49984
H	29.1	0.7477	2	1.7	2.5	2.4
J	3.2686	1.8	2.3	2.9	2.1	2.7
K	0.74143	1.10543	0.470777	0.6852	2.600935	0.128868

Table 6: Inventory levels

product	P1	P2	P3	P4	P5	P6
A	0	0	0	0	0	0
B	0	0	0	0	0	0
C	0	0	0	0	0	0
D	0	0	0	0	0	0
E	0.1612	0.1612	0.1612	0.1612	0.1612	0.1612
F	0	0	0	0	0.2308	0
G	0	0	0	0	0	0
H	0.1434	0.1434	0.1434	0.1434	0.1946	0.1263
J	1.9686	1.9686	1.9686	1.9686	1.9686	1.9686
K	0.004	0	0	0	0.5142	0

Table 4:Results for each algorithm

product	P_1	P_2	P_3	P_4	P_5	P_6
W_t	1915	1682	1453	1147	1041	1042
H_t	0	0	0	0	0	1
F_t	1400	233	229	306	106	0
O_t	4.2337	8.6316	0	0	0	0

6 CONCLUSION

SDA performs without partial derivatives for each independent variable of the function, robust, easy to be programmed and fast. Therefore, in this study presented a SDA for solving multi-objective linear programming model of APP problems for the first time. The proposed model attempt to minimizes the total production and workforce costs. The model was applied to solve the APP problem of the General Company for Vegetable Oils. The proposed algorithmwas also compared with GA. The results showed that the SDA is competitive and can provide an efficient solution within a short time.

REFERENCES

[1] A. Jamalnia, M. A. Soukhakian, A hybrid fuzzy goal programming approach with different goal priorities to aggregate production planning, Computers & Industrial Engineering 56 (4) (2009) 1474-1486.

•[2] L. Ozdamar, M. A. Bozyel, S. I. Birbil, A hierarchical decision support system for production planning (with case study), European Journal of Operational Research 104 (3) (1998) 403-422.

[3] T.-F. Liang, H.-W. Cheng, P.-Y. Chen, K.-H. Shen, Application of fuzzy sets to aggregate production planning with multiproducts and multitime periods, IEEE Transactions on Fuzzy Systems 19 (3) (2011) 465-477.

[4] G. H. SAAD, An overview of production planning models: structural classification and empirical assessment, THE INTERNATIONAL JOURNAL OF PRODUCTION RESEARCH 20 (1) (1982) 105-114.

[5] K. Singhal, V. Adlakha, Cost and shortage trade-offs in aggregate production planning, Decision Sciences 20 (1) (1989) 158-165.

[6] C. C. Holt, F. Modigliani, H. A. Simon, A linear decision rule for production and employment scheduling, Management Science 2 (1) (1955) 1-30.

[7] E. H. Bowman, Production scheduling by the transportation method of linear programming, Operations Research 4 (1) (1956) 100-103.

[8] E. H. Bowman, Consistency and optimality in managerial decision making, Management Science 9 (2) (1963) 310-321.

[9] W. H. Taubert, A search decision rule for the aggregate scheduling problem, Management Science 14 (6) (1968) B-343.

[10] M. Byrne, M. Bakir, Production planning using a hybrid simulation analytical approach, International Journal of Production Economics 59 (1) (1999) 305-311.

[11] F. Glover, Tabu searchpart ii, ORSA Journal on computing 2 (1) (1990) 4-32.

[12] J. H. Holland, Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence., U Michigan Press, 1975.

[13] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, et al., Optimization by simulated annealing, science 220 (4598) (1983) 671-680.

[14] Z. W. Geem, J. H. Kim, G. Loganathan, A new heuristic optimization algorithm: harmony search, Simulation 76 (2) (2001) 60-68.

[15] J. Kennedy, R, Eberhart, Particle swarm optimization 1.

[16] P. Luangpaiboon, P. Aungkulanon, Integrated approaches to enhance aggregate production planning with inventory uncertainty based on improved harmony search algorithm, in: Proceedings of World Academy of Science, Engineering and Technology, no. 73, World Academy of Science, Engineer-ing and Technology (WASET), 2013, p. 243.

[17] E. Mehdizadeh, A. A. A. Abkenar, et al., Harmony search algorithm for solving two aggregate production planning models with breakdowns and maintenance, in: Proceedings of the International Management Conference, Vol. 8, Faculty of Management, Academy of Economic Studies, Bucharest, Romania, 2014, pp. 306-320.

[18] M. Ramazanian, A. Modares, Application of particle swarm optimization algorithm.

[19] S.-C. Wang, M.-F. Yeh, A modified particle swarm optimization for aggregate production planning, Expert Systems with Applications 41 (6) (2014) 3069-3077.

[20] M. R. Islam, M. S. Aziz, M. M. H. Muftee, M. S. Hossain, Application of particle swarm optimization in aggregate production planning and comparison with genetic algorithm.

[21] A. Baykasoglu, Moapps 1.0: aggregate production planning using the multiple-objective tabu search, International Journal of Production Re-search 39 (16) (2001) 3685-3702.

[22] A. S. Masud, C. Hwang, An aggregate production planning model and application of three multiple objective decision methods, International Journal of Production Research 18 (6) (1980) 741-752.

[23] A. Baykasoglu, S. Owen, N. Gindy, Solution of goal programming models using a basic taboo search algorithm, Journal of the Operational Research Society 50 (9) (1999) 960-973.

[24] A. BAYKASOGLU, S. OWEN, N. GINDY, A taboo search based approach to find the pareto optimal set in multiple objective optimization, Engineering Optimization 31 (6) (1999) 731-748.

[25] G. M. Kumar, A. N. Haq, Hybrid geneticant colony algorithms for solving aggregate production plan, Journal of Advanced Manufacturing Systems 4 (01) (2005) 103-111.

[26] K. Ragsdell, A. Ravindran, G. Reklaitis, Engineering optimization: Methods and applications (1983).

[27] R. Chelouah, P. Siarry, A hybrid method combining continuous tabu search and nelder{mead simplex algorithms for the global optimization of multiminima functions, European Journal of Operational Research 161 (3) (2005) 636-654.

[28] W. Spendley, G. R. Hext, F. R. Himsworth, Sequential application of simplex designs in optimization and evolutionary operation, Technometrics 4 (4) (1962) 441-461.

[29] J. A. Nelder, R. Mead, A simplex method for function minimization, The computer journal 7 (4) (1965) 308-313.