Journal of Quality Engineering and Production Optimization
Vol. 1, No. 1, PP. 11-20, 2015

A New Optimization via Invasive Weeds Algorithm for Dynamic Facility Layout Problem

Farnaz Barzinpour[1], Mohammad Mohammadpour Omran[1], Seyed Farzad Hoseini[1], Kaveh Fahimi[1] and Farshid Samaei[2]
[1] *Department of Industrial Engineering, Iran University of Science & Technology, Tehran, Iran*
[2] *Department of Civil and Engineering, Ports & Maritime Organization, Bandar Abbas, Iran*

***Corresponding Author:** Seyed Farzad Hoseini (E-mail: F-Hoseini@ind.iust.ac.ir)

***Abstract-** The dynamic facility layout problem (DFLP) is the problem of finding positions of departments on the plant floor for multiple periods (material flows between departments change during the planning horizon) such that departments do not overlap, and the sum of the material handling and rearrangement costs is minimized. In this paper a new optimization algorithm inspired from colonizing weeds, Invasive Weeds Optimization (IWO) is utilized to solve the well-known DFLP. IWO is a simple algorithm which uses basic characteristics of a colony of weeds such as proliferation, growth and competition.*
A set of reference numerical problems is taken in order to evaluate the efficiency of the algorithm compared with the Dynamic Programming method which had been applied to solve the addressed problem. In order to verify the efficiency of the proposed algorithm a wide range of experiments are carried out to compare the proposed algorithm. Computational results have indicated that the DIWO algorithm is capable of obtaining optimal solutions for small and medium-scaled problems very efficiently.

***Keywords:** Discrete Invasive Weed Optimization, Dynamic Facility layout Problem, Dynamic Programming.*

I. INTRODUCTION

Nowadays, due to competitive and global markets, there are more changeability in design of products and life cycle of products has become too short. Therefore, in most industries, products are under alternation and this issue has direct influence on layout of facilities. Since the changes of demand increases costs of the organization because of adaptation; hence, organizations seek methods to satisfy demands with less costs. In this regard, recently, Dynamic Facility Layout Planning (DFLP) has been considered by many researchers. DFLP, based on prediction of changes that might occur in material flow in future, divides the future into time periods and yields an efficient layout by minimizing summation of material transportation and rearrangement costs [1].

Plenty of factors are involved in proposing an efficient design for an industrial unit. In most studies, transportation cost of materials has been considered as the most significant criterion. However, by considering the current competiveness and short life cycle, analyzing this criterion merely may not be appropriate. It is obvious that the corresponding criteria should be considered under different conditions.

If we consider T periods, and N parts, maximum number of facility layout plans for DFLP is $(N!)^T$. For example, N=6 and T=5, so we have $(6!)^5 = 1.93\times10^{14}$ as facility layout plans. This calculation shows that even for the small scaled problem there is a huge complexity, so the problem is NP-Hard [2].

II. LITERATURE REVIEW

Rosenblatt was the first one who developed an optimization approach based on a dynamic programming model for the DFLP. Rosenblatt proposed two heuristics that were based on dynamic programming model, each of which simply

Manuscript Received: 8 June 2014 and revised 21 Oct 2014 *ISSN: 2423-3781*
Accepted: 28 Dec 2014

considers a set of limited good layouts for a single period. Urban developed a steepest-descent heuristic based on a pairwise exchange idea, which is similar to CRAFT [3]. Lacksonen and Enscore introduced and compared five heuristics to solve the DFLP, which were based on dynamic programming, a branch and bound algorithm, a cutting plane algorithm, cut trees, and CRAFT [4].

It should be mentioned that in addition to exact algorithms, many meta-heuristic algorithms have been reported in the literature such as a genetic algorithm by [5] and a tabu search (TS) heuristic by [6]. This TS heuristic is a two-stage search process that incorporates diversification and intensification strategies. Baykasoglu and Gindy developed a simulated annealing (SA) heuristic for the DFLP, in which they used the upper and lower bounds of the solution of a given problem instance to determine the SA parameters [7]. Balakrishnan et al. presented a hybrid genetic algorithm [8]. Erel et al. introduced a new heuristic algorithm to solve the DFLP. They used weighted flow data from various time periods to develop viable layouts and suggested the shortest path for solving the DFLP [9]. McKendall and Shang developed three hybrid ant systems (HAS) [10]. McKendall et al. introduced two (SA) heuristics. The first one (SAI) is a direct adaptation of SA for the DFLP while the second one (SAII) is the same as SAI except that it incorporates an added look-ahead/look-back strategy [11]. A hybrid meta-heuristic algorithm based on a genetic algorithm and tabu search was introduced by Rodriguez et al. [12]. Krishnan et al. used a novel tool, the “Dynamic From-Between Chart,” for an analysis of redesigned layouts. This tool models changes in the production rates using a continuous function [13]. Ripon et al. developed Pareto-optimal solutions for multi-objective DFLP under uncertainty. They investigated an evolutionary approach to solve the multi-objective dynamic facility layout problem under uncertainty that presents the layout as a set of Pareto-optimal solutions. In addition, the approach proposed in this paper is tested using a backward pass heuristic to determine its effectiveness in optimizing multiple objectives [14]. Emami and Nookabadi modeled DFLP as a multi-objective optimization problem. In the proposed model the adjacency-based objective aims at maximizing adjacency scores between the facilities in a facility layout problem [15]. For an extensive review on the DFLP, one can refer to the studies presented by[16], and [17].

In this paper, we first introduce the problem formulation for the DFLP in Section 3. In Section 4, the proposed algorithm is introduced. In Section 5, computational results are summarized, and finally some concluding remarks are presented in Section 6.

III. PROBLEM FORMULATION

In this section, we have formulated the mathematical model for the DFLP adopted from Balakrishnan et al. The assumptions are described as follows:

(1) Equal-sized facilities and locations are considered, (2) Shapes and dimensions of the shop floor are not considered, (3) The number of periods in the planning horizon is known and (4) Distance between the facilities is determined a priori.

A. *Indexing sets*

i, j are indices for facilities, $i, j = 1, 2, ..., M, i \neq j$;

k, l are indices for facility locations, $k, l = 1, 2, ..., M, k \neq l$;

t is the index for periods, $t = 1, 2, ..., P$;

B. *Parameters*

M s the total number of locations and facilities; P is the number of periods; f_{tik} is the flow cost for unit distance from facility i to k in period t; d_{tij} is the distance from location j to1 in period t; A_{tij1} is the cost of shifting facility I between locations j and 1 in period t.

C. *Decision variables*

The decision variables of the model, X_{tij} and Y_{tij} ,are defined as follows:

$$X_{tij} = \begin{cases} 1 & \text{If facility i is assigned to location j in period t,} \\ 0 & \text{Otherwise} \end{cases}$$

$$Y_{tij} = \begin{cases} 1 & \text{if facility i is shifting between locations j and l at the beginning of period t,} \\ 0 & \text{Otherwise} \end{cases}$$

D. Mathematical model

The quadratic assignment problem (QAP) for the DFLP is presented as follows:

$$\text{Minimize } Z = \sum_{t=1}^{P}\sum_{i=1}^{M}\sum_{k=1}^{M}\sum_{j=1}^{M}\sum_{l=1}^{M} f_{tik} d_{tjl} X_{tij} X_{tkl} + \sum_{t=2}^{P}\sum_{i=1}^{M}\sum_{j=1}^{M}\sum_{l=1}^{M} A_{tijl} Y_{tijl} \tag{1}$$

$$\sum_{j=1}^{M} x_{tij} = 1 \quad , \qquad i = 1,2,...,M \text{ and } t = 1,2,...,P \tag{2}$$

$$\sum_{i=1}^{M} x_{tij} = 1 \quad , \qquad j = 1,2,...,M \text{ and } t = 1,2,...,P \tag{3}$$

$$Y_{tijl} = X_{(t-1)ij} X_{til} \ , \qquad i, j, l = 1,2,...,M \ , \ t = 1,2,...,P \tag{4}$$

$$X_{tij}, X_{tkl}, Y_{tijl} \in \{0,1\}, \qquad i, j, l = 1,2,...,M, \ t = 2,3,...,P \tag{5}$$

The objective function in Equation (1) minimizes the sum of the material flow and layout rearrangement cost during the planning horizon. Constraints in Equations (2) and (3) ensure that each facility location is assigned to one location at each period, respectively. Constraint in Equation (4) adds the rearrangement cost to the material flow cost if a facility is shifted between locations in consecutive periods. Lastly, the restrictions on the decision variables are given in Equation (5).

IV. SOLUTION METHOD

To solve the proposed multi-objective facility layout problem, we have used a priority-based method which is introduced for solving multi-objective optimization problems. In priority-based multi-objective optimization, each solution is corresponded with a specific vector of objectives' importance. A simple method for solving a multi-objective problem is to transform different objectives to one objective function by using weighted factor in which the weight of each multiplier is a ratio of preference factor of the corresponding objective. In fact, this method transforms a multi-objective problem to a single-objective problem. In this research a similar weighted priority is assigned to all objectives.

V. DISCRETE INVASIVE WEEDS ALGORITHM

Inspiring from colonization of invasive weeds, a common farming phenomenon, Mehrabian and lucas introduced a new evolutionary algorithm for solving continuous operational research problems [18]. They named this new algorithm as Invasive Weed Optimization (IWO). IWO is a simple algorithm which uses basic characteristics of a colony of weeds such as proliferation, growth and competition, that mimics the process of weeds colonization and distribution. Despite its recent development, it has shown successful results in a variety of practical applications like optimization and tuning of a robust controller [16], optimal positioning of piezoelectric actuators [17], developing a recommender system [18], cooperative multiple task assignment of the UAVs [19], etc. Due to its wide range applicability and relative fast convergence rate, we are motivated to apply this algorithm to the model.

According to the prevalent definition, a weed is a plant which grows in an undesirable place. Weeds have a very strong and adaptive nature which makes them undesirable plants in farming.

The proposed algorithm is described as follows:

A. Initialize a population

A population of initial solutions is randomly generated.

B. Reproduction

A member of the population of plants is allowed to reproduce depending on its own and the colony's lowest and highest fitness functions. The number of seeds that plant j can produce is calculated by Equation (6):

$$Seed(j) = \left\lceil \frac{(S_{\max} - S_{\min}) \times (Fitness(j) - Fit_{\min})}{Fit_{\max} - Fit_{\min}} \right\rceil \tag{6}$$

where S_{max} is the maximum number of seeds, S_{min} is the minimum number of seeds, Fit_{max} is the maximum fitness in the colony, Fit_{min} is the minimum fitness in the colony and $\lceil \mathrm{O} \rceil$ denotes the smallest integer which is larger than or equal to the enclosed number.

C. Dispersal

The generated seeds are being randomly distributed over search space by some “swap move” or “insertion move”, which the number of moves is generated randomly according to discrete uniform distribution with a minimum value of 1 and the maximum number of possible movements (σ). The maximum number of possible movements will be reduced from a previously defined initial value ($\sigma_{initial}$) to a final value (σ_{final}) in each step (generation), which is given in Equation (7):

$$\sigma_{iter} = round\left(\sigma_{final} + \left(\frac{iter_{\max} - iter}{iter_{\max}} \right)^n \times \left(\sigma_{initial} - \sigma_{final} \right) \right), \tag{7}$$

where $iter_{max}$ is the maximum number of iterations, σ_{iter} is the maximum number of possible movements at the present time step and *n* is the nonlinear modulation index usually set to 3.

D. Competitive exclusion

After some iterations, the number of plants in a colony will reach its maximum number of plants in the colony (p_{max}). Hence, a mechanism to eliminate the plants with poor fitness in the generation activates, which works as follows:

When the offspring spread over the search space according to the dispersal mechanism, they are ranked together with their parents' as a colony of weeds. Next, weeds with lower fitness are eliminated to reach the maximum allowable population size in a colony.

TABLE I. DIWO PARAMETERS

Symbol	Definition
N_0	Number of initial weeds
it_{max}	Maximum number of iterations
p_{max}	Maximum number of plants population
s_{max}	Maximum number of seeds
s_{min}	Minimum number of seeds
n	Nonlinear modulation index
$\sigma_{initial}$	Initial value of maximum number of possible movements
σ_{final}	Final value of maximum number of possible movements

Set DIWO parameters (N_0, it_{max}, p_{max}, s_{max}, s_{min}, n, $\sigma_{initial}$, σ_{final});
Generate N_0 random solutions;
$Iter = 0$;
Phase I:
REPEAT (Population of colony < P_{max} OR $Iter < it_{max}$)
1- Calculate Fit_{min} and Fit_{max} of colony;
2- Calculate σ_{iter} by equation (19);
3- REPEAT (for each weed in the colony)
a. Calculate number of seeds for the weed by equation (18);
b. X = weed (as parent);
c. σ = Random integer number from σ_{final} to σ_{iter};
d. REPEAT (for σ times)
i. Y = Generate a solution from X by Swap movement;
i. $X = Y$;
END REPEAT;
e. Add X to colony;
END REPEAT;
4- $Iter = Iter + 1$;
END REPEAT;
Phase II:
REPEAT ($Iter < it_{max}$)
1- Select best P_{max} plants among parents and offspring (competitive exclusion);
2- Calculate Fit_{min} and Fit_{max} of colony;
3- Calculate σ_{iter} by equation (19);
4- REPEAT (for each weed in the colony)
a. Calculate number of seeds for the weed by equation (18);
b. X = Parent of seed;
c. σ = Random integer number from σ_{final} to σ_{iter};
d. REPEAT (for σ times)
i. Y = Generate a solution from X by Swap move;
i. $X = Y$;
END REPEAT;
e. Add X to colony;
END REPEAT;
5- $Iter = Iter + 1$;
END REPEAT;
RETURN (Best Solution);

Fig. 1: Pseudo code of the DIWO

E. Algorithm structure

The DIWO parameters are specified in Table 2. The proposed algorithm contains two phases, which phase I is applied to reach the population at least pmax and then phase II is applied until the stopping criterion is reached. The second phase contains the exclusion mechanism. In comparison with the genetic algorithm, an individual is called a weed, a child is a seed, and the population is a colony. The procedure of DIWO is shown in Figure 2.

The general framework of the Invasive weeds algorithm is depicted in Figure 2.

F. Solution representation

To solve a problem with a meta-heuristic algorithm, we first need to propose a solution representation. We name facilities and locations with digits 1, 2… n. A solution is a set of several strings of digits and each string is a sequence of n digits. Each digit represents a facility and location of each digit represents the location of that facility. In other words, digit i in location j is interpreted as locating facility i in location j. For example, a solution representation of a DFLP problem with 6 facilities and 5 periods is given in Fig 3. In this figure, each row represents facilities locations of each period. For example, facility 2 is in location 1, facility 4 is in location 2, facility 6 is in location 3, facility 3 is in location 4, facility 5 is in location 5 and facility 1 is in location 6.

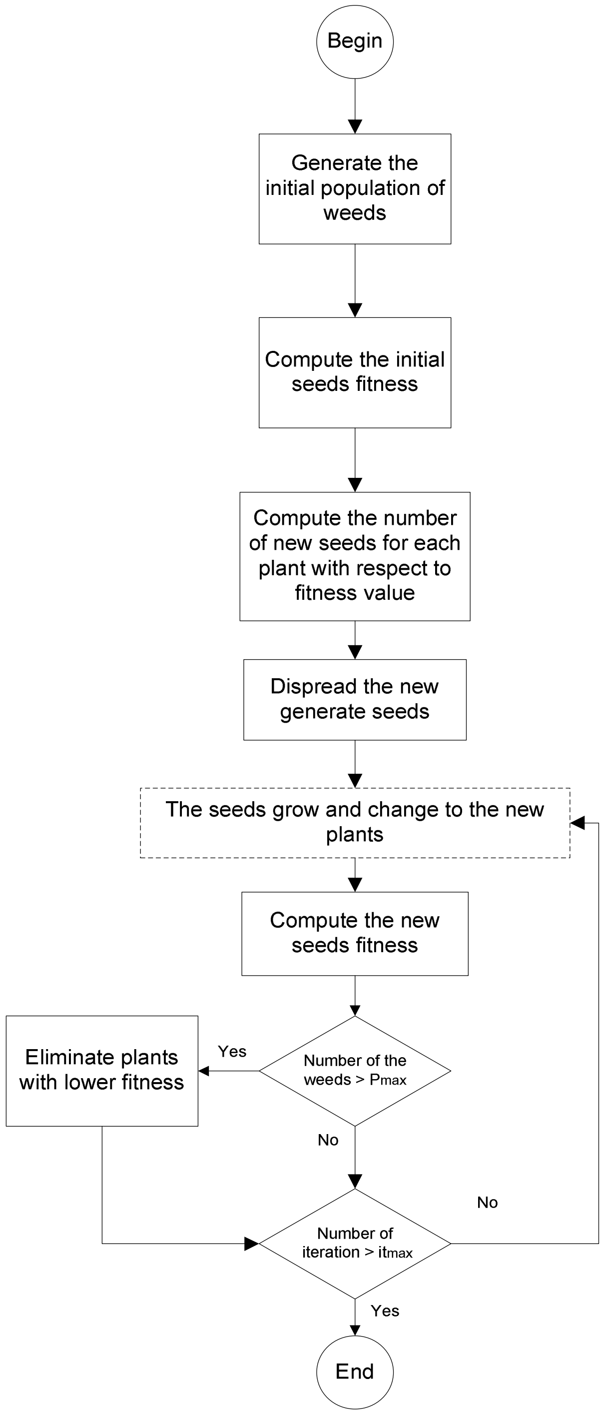

Fig. 2: Invasive Weeds Algorithm framework

4	2	6	3	5	1
2	4	6	3	5	1
2	4	6	5	3	1
2	4	1	5	3	6
2	4	6	5	3	1

Fig. 3. Solution representation

After generating random solutions, fitness of plants is calculated. In the next step, plants are ordered according to their fitness functions. Then, number of new seeds is determined according to a linear relation between amount of fitness, S_{min} and S_{max}. In the next step, the newly generated seeds are scattered in solution space according to Equations (1-3) and fitness of newly grown plants is calculated. If number of plants exceeds a predefined maximum number (P_{max}), number of (P- P_{max}) plants with minimum fitness is eliminated to have P_{max} plants. Otherwise, previous step is repeated.

In the final step, the algorithm checks the termination condition. If the termination condition is met, the algorithm is terminated; otherwise, new seeds are generated and distributed in the solution space.

VI. DYNAMIC PROGRAMMING

In this paper, dynamic programming technique is used to solve small-sized instances (N6T5 and N6T10 instances). In this technique, for each period i we create a level L_i and all the possible static layouts of period i are considered as the states of L_i. Dynamic programming technique solves the problem using a recursive technique. In fact, for each period, a layout is selected among all predefined layouts of that period such that sum of the material flow and rearrangement costs of the planning horizon is minimized. In the following, parameters and solution procedure are described.

The optimal solution methodology is developed using Rosenblatt dynamic programing as the following recursive relationship:

$$C^{*}_{tm} = min\left\{C^{*}_{t-1} + R_{km}\right\} + Q^{m}_{t}, \tag{8}$$

where R_{km} is the rearrangement cost from layout A_k to layout A_m, Q_m is the material handling cost for layout A_m in period t and C^{*}_{tm} is the minimum total costs for all periods up to t, where layout A_m is being used in period t.

A backward approach is used to solve the recursive relation in Equation (8). Each period of planning horizon corresponds with a level and each static layout corresponds with a state.

VII. COMPUTATIONAL RESULTS

This section presents the computational results of the proposed DIWO algorithm applied to the 32 test problems obtained from Balakrishnan and Cheng. The proposed algorithm was programmed using C++ programming language and a set of the test problems was solved on a PC with a 2.83 GHz Core-i3-530M CPU with 8.00GB of RAM.

As mentioned before, IWO involves some parameters which should be tuned to provide DFLP best solutions. We have run the algorithm for each problem set several times using the different values of the parameters and then have selected the best parameters based on solution results. Based on the experimental results, all the best parameter settings for the proposed DIWO algorithm are given in Table 2. In this study, the nonlinear modulation index is set equal to 3. Tables 3-6 summarize the results obtained by the DIWO algorithm.

For each dataset, the results for the DIWO algorithm were compared with the best results obtained by the GA presented by (Balakrishnan and Cheng, 2000; Balakrishnan et al., 2003), the DP presented by Erel et al. (2003), the

HAS presented by McKendall and Shang (2006), and the SA presented by (Baykasoglu and Gindy, 2004; McKendall et al., 2006).

Tables 3 and 4 show the results for the test problems where M=6, P=5 (Problems 1-8) and M=6, P=10 (Problems 9-16), respectively. The DIWO algorithm obtained the optimum solutions for all the 1-16 test problems.

Tables 5 and 6 give the results for the test problems where M=15, P=5 (Problems 17-24) and M=15, P=10 (Problems 25-32), respectively. The DIWO algorithm obtained 9 optimum solutions of 16 problems and 7 near optimal solutions of 16 problems, while the DP was unable to reach a feasible solution.

Since the varied heuristic algorithms in the literature use different computing systems, programming language compilers, coding techniques and so on, it is very difficult to compare their computation time. Hence, we did not make comparison between computation times of the algorithms in this study.

TABLE II. PARAMETERS USED IN THIS STUDY

Parameters	N_0	S_{max}	S_{min}	P_{max}	it_{max}	σ_{initial}	σ_{final}
M=6, P=5	20	4	0	20	8000	4	1
M=6, P=10	25	4	0	40	20000	6	1
M=15, P=5	30	8	0	80	32000	8	1
M=15, P=10	60	10	0	160	50000	10	1

TABLE III. SOLUTION RESULTS FOR PROBLEMS WITH M=6, P=5

Problem no.	1	2	3	4	5	6	7	8
GA	106419	104834	104320	106515	105628	104053	106439	103771
HAS	106419	104834	104320	106399	105628	103985	106439	103771
SA	106419	104834	104320	106399	105628	103985	106439	103771
DP	106419	104834	104320	106399	105628	103985	106439	103771
DIWO best solution	106419	104834	104320	106399	105628	103985	106439	103771
DIWO average solution	106419	104834	104320	106399	105628	103985	106439	103771
Deviation (%)	0	0	0	0	0	0	0	0

TABLE IV. SOLUTION RESULTS FOR PROBLEMS WITH M=6, P=10

Problem no.	9	10	11	12	13	14	15	16
GA	214313	212134	207987	212741	210944	210000	215452	212588
HAS	214313	212134	207987	212530	210906	209932	214252	212588
SA	214313	212134	207987	212530	210906	209932	214252	212588
DP	214313	212134	207987	212530	210906	209932	214252	212588
DIWO best solution	214313	212134	207987	212530	210906	209932	214252	212588
DIWO average solution	214313	212134	207987	212565	210906	209932	214318	212588
Deviation (%)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

TABLE V. SOLUTION RESULTS FOR PROBLEMS WITH M=15, P=5

Problem no.	17	18	19	20	21	22	23	24
GA	484090	485352	489898	484625	489885	488640	489378	500779
HAS	480453	484761	488748	484446	487722	486685	486853	491016
SA	480453	484761	488748	484405	487882	487147	486779	490812
DIWO best solution	480453	484761	488748	484446	487882	487147	486779	490812
DIWO average solution	481945	485610	489072	486659	489043	489426	487724	492463
Deviation (%)	0.00	0.00	0.00	0.01	0.03	0.00	0.00	0.00

TABLE VI. SOLUTION RESULTS FOR PROBLEMS WITH M=15, P=10

Problem no.	25	26	27	28	29	30	31	32
GA	987887	980638	985886	976025	982778	973912	982872	987789
HAS	980351	978271	978027	974694	979196	971548	980752	985707
SA	979468	978065	982396	972797	977188	967617	979114	983672
DIWO best solution	979468	983826	988635	972797	979196	967617	982872	985707
DIWO average solution	995291	995045	1090201	986528	990486	972762	998904	1000053
Deviation (%)	0.00	0.59	1.08	0.00	0.21	0.00	0.38	0.21

VIII. CONCLUSION AND FUTURE RESEARCH

Heuristics or suboptimal algorithms are often used to obtain solutions for DFLP instances. DIWO is a numerical stochastic search algorithm inspired from natural behavior of weeds colonizing in opportunity spaces for function optimization. In small scaled problems, Dynamic Programming was used and for larger scales meta-heuristic algorithm DIWO was applied. By using available data in the literature, this method was assessed. Computational results indicated that the proposed DIWO is capable of obtaining optimal solutions for small and medium-scaled problems very efficiently. In addition, the quality of the solutions obtained by DIWO for large-scaled problems was evaluated through empirical comparisons with best-known results. Computational results demonstrated that the proposed DIWO significantly provides satisfied solutions.

Modeling DFLP with production uncertainty can be considered as a future research. In addition, the time-dimension comparison of the developed algorithms to solve DFLP can be the other interesting issue for future studies.

REFERENCES

1. Rosenblatt, M.J. (1986). "The dynamics of plant layout", *Management Science*, 32(1) pp. 76-86.
2. Lin, X., Floudas, C., & Kallrath, J. (2005). "Global Solution Approach for a Nonconvex MINLP Problem in Product Portfolio Optimization", *Journal of Global Optimization*, 32(3) pp.417-431.
3. Urban, L. (1998). "Solution procedures for the dynamic facility layout problem," *Annals of Operations Research, 76* pp. 323–342.
4. Lacksonen, T.A., & Enscore E.E.(1993). "Quadratic assignment algorithms for the dynamic layout problem". *International Journal of Production Researc*h, 31(3) 503–517.
5. Balakrishnan, J., & Cheng, C.H.(2000). "Genetic search and the dynamic layout problem," *Computers and Operations Research*, 27(6) pp. 587–593.
6. Kaku, B. K., & Mazzola, J. B.(1997)."A tabu-search heuristic for the dynamic plant layout problem," *INFORMS Journal on Computing*, 9(4) pp. 374–384.

7. Baykasoglu, A., & Gindy, N. N. Z. (2001). "A simulated annealing algorithm for dynamic facility layout problem," *Computers and Operations Research*, 28(14) pp. 1403–1426.

8. Balakrishnan, J., Cheng, C. H., Conway, D. G., & Lau, C. M.(2003). "*A hybrid genetic algorithm for the dynamic plant layout problem*," International Journal of Production Economics, 86(2) pp. 107–120.

9. Erel, E., Ghosh, J. B, & Simon, J. T.(2003). "New heuristic for the dynamic layout problem," *Journal of the Operational Research Society*, 56(8) p. 1001.

10. McKendall, A. R., & Shang, J.(2006). "Hybrid ant systems for the dynamic facility layout problem," *Computers & Operations Research*, 33(3) pp. 790–803.

11. McKendall, A. R., Shang, J., & Kuppusamy, S.(2006). "Simulated annealing heuristics for the dynamic facility layout problem," *Computers and Operations Research*, 33(8) pp. 2431–2444.

12. Rodriguez, J. M., MacPhee, F. C., Bonham, D. J., & Bhavsar, V. C.(2006). "Solving the dynamic plant layout problem using a new hybrid meta-heuristic algorithm," *International Journal of High Performance Computing and Networking*, 4(5-6)pp. 286–294.

13. Krishnan, K. K., Cheraghi, S. H., & Nayak, C. N.(2006). "Dynamic from-between charts: a new tool for solving dynamic facility layout problems*," International Journal of Industrial and Systems Engineering*, 11(1-2) pp. 182–200.

14. Ripon, Kazi Shah Nawaz, et al (2011). "Dynamic facility layout problem under uncertainty: a Pareto-optimality based multi-objective evolutionary approach." *Central European Journal of Computer Science*, 1.4 pp. 375-386.

15. Emami, S., & Nookabadi, A.S. (2013)."Managing a new multi-objective model for the dynamic facility layout problem", *The International Journal of Advanced Manufacturing Technology*", 68.9-12 pp. 2215-2228.

16. Balakrishnan, J., & Cheng, C. H. (1998) ."Dynamic layout algorithms: a state-of-the-art survey", 26(4) pp. 507–521.

17. Kulturel-Konak, S. (2007). "Approaches to uncertainties in facility layout problems: perspectives at the beginning of the 21st Century," *Journal of Intelligent Manufacturing*, 18(2) pp. 273–284.

18. Mehrabian, A. R., & Lucas, C.(2006) ."A novel numerical optimization algorithm inspired from weed colonization." *Journal of Ecological Informatics*, vol. 1 pp. 355–366.

19. Mehrabian, A., & Yousefi-Koma, A. (2007). "Optimal positioning of piezoelectric actuators on a smart fin using bio-inspired algorithms", *Aerospace Science and Technology*, vol. 11 pp. 174–182.

20. Rad, H., Lucas, C. (2008). "A recomniender system based on invasive weed optimization algorithm", in: 2007 *IEEE Congress on Evolutionary Computation*, CEC 2007 pp. 4297–4304.

21. Ghalenoei, M., Hajimirsadeghi, H., Lucas, C. (2009)."Discrete invasive weed optimization algorithm: application to cooperative multiple task assignment of UAVs", in: Proceedings of the IEEE Conference on Decision and Control, pp. 1665–1670.