Identification of Promising Infant Clusters to Obtain Improved Block Layout Designs

Mustahsan Mir, Ahmed Hassanin, Mohammed A. Al-Saleh

***Abstract*—**The layout optimization of building blocks of unequal areas has applications in many disciplines including VLSI floorplanning, macrocell placement, unequal-area facilities layout optimization, and plant or machine layout design. A number of heuristics and some analytical and hybrid techniques have been published to solve this problem. This paper presents an efficient high-quality building-block layout design technique especially suited for solving large-size problems. The higher efficiency and improved quality of optimized solutions are made possible by introducing the concept of Promising Infant Clusters in a constructive placement procedure. The results presented in the paper demonstrate the improved performance of the presented technique for benchmark problems in comparison with published heuristic, analytic, and hybrid techniques.

***Keywords*—**Block layout problem, building-block layout design, CAD, optimization, search techniques.

I. Introduction

THE Block Layout Problem (BLP) involves optimal placement of unequal-area rectangular blocks on a 2-D plane such that a certain objective function is optimized subject to specified set of constraints. It has applications in many fields including VLSI floorplan design [1]-[10], macrocell placement optimization [11]-[14], unequal-area facilities layout optimization [15]-[26], and plant or machine layout design [27]-[30]. The difference in various above-mentioned applications is the nature of objective or cost function and constraints. For instance, in VLSI floorplan design and macrocell placement optimization a commonly used objective is minimization of total wirelength while in the case facilities or machine layout design the objective is generally to minimize the total material handling cost. One important constraint that is common to all applications is non-overlapping of blocks. As no general solution exists for this complex optimization problem, numerous heuristic techniques including those based on simulated annealing [10], [17], [26], genetic algorithms [15], [28], Tabu search [21], [24], particle swarm optimization [21], [25], ant colony optimization [4], [29], and other heuristics [6], [9], [23], [30] have been developed to efficiently solve this problem for a near-optimal solution. The heuristic techniques, in general, either do not consider the actual dimensions of the blocks or assume them to be all having the same areas during the initial phase of finding the relative placement but have been quite successful as they are capable of finding good solutions taking into account a variety of practical constraints for various applications. However, for problems where the variation in the sizes of blocks does not justify the equal-area assumption, the optimized layouts will have significant overlaps. To remove these overlaps, the blocks are arbitrarily moved and the resulting overlap-free layout is no more an optimal layout. In contrast, some highly efficient analytical techniques for large-scale block layout problems have successfully incorporated the geometrical specifications in their mathematical formulation and therefore have the distinct advantage of producing optimal layouts without any overlaps [12], [14], [19], [22], [31]. However, most analytical techniques are not as flexible as heuristic techniques for incorporating various applicable constraints in the optimization process. By combining the salient features of both heuristics and analytical techniques, some hybrid techniques for block layout optimization have also been published [16], [20].

A highly effective analytical placement technique for solving BLP is based on cluster boundary search algorithm [31]. Like other constructive techniques, it also builds the layout block by block. However, unlike other placement techniques, the algorithm efficiently explores all feasible search space for the new block to ensure that it is placed, without any overlaps, at its optimum position with respect to the already placed cluster of blocks. While the search algorithm is quite effective in finding the optimum position of one block at a time, the ordering of blocks still affects the quality of optimized layouts. In order to minimize the impact of ordering on the quality of optimal layout design and reduce search time, an analytical technique based on modified cluster boundary search algorithm with improved ordering criteria has been developed [22]. While the improved ordering criterion has resulted in obtaining better layout designs, a number of different firing orders are still required in order to select the best layout among various designs achieved using these firing orders. And if the number of firing orders is too large, the computation time becomes quite excessive. Thus, while increasing the number of firing orders increases the probability of finding a better layout design, it also significantly increases the computation cost.

The analytical technique presented in this paper is a constructive placement technique primarily based on cluster boundary search algorithm with three important developments. These include identification and utilization of “Promising Infant Clusters” to minimize computation time, a semi-deterministic ordering criterion for obtaining multiple firing orders, and an enhanced boundary search procedure for

M. Mir is with Ajman University of Science & Technology in UAE (phone: 0097167056763; fax: 0097167438888; e-mail: m.mir@ ajman.ac.ae).
A. Hassanin is with RWTH Aachen, Germany.
M. A. Al-Saleh is Umm Al-Qura University, Saudi Arabia.

efficient placement of new block at its optimum position with respect to the already placed cluster of blocks. Its performance is compared with other techniques for some published benchmark problems and it is shown that the performance of the presented technique in obtaining improved layout designs is better than other published techniques.

II. Problem Formulation

Consider N rectangular blocks, with fixed dimensions, to be placed on a continuous 2-D plane without any overlaps such that a specified cost function is minimized. The position of ith block is defined by the coordinates of its centroid (x_i,y_i). Let (L_i,W_i) denote the length and width of ith block along the X- and Y-axes, respectively. Let matrix $\{\alpha_{ij}\}$ define the connectivity relationship for all pairs of modules in a VLSI/macrocell placement problem or the weighted flow relationship between all pairs of facilities in a facility or machine layout problem. The cost function F to be minimized is defined as:

$$F = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \alpha_{ij} d_{ij} \qquad (1)$$

where, d_{ij} is the distance measured between the centroids of blocks i and j and could be either of the following three distance norms:

a) Euclidean distance:

$$d_{ij} = ((x_i - x_j)^2 + (y_i - y_j)^2)^{1/2} \qquad (2)$$

b) Squared Euclidean distance:

$$d_{ij} = (x_i - x_j)^2 + (y_i - y_j)^2 \qquad (3)$$

c) Rectilinear distance:

$$d_{ij} = |x_i - x_j| + |y_i - y_j| \qquad (4)$$

Subject to:

$$|x_i - x_j| \geq \frac{wi+wj}{2}$$

or

$$|y_i - y_j| \geq \frac{hi+hj}{2} \qquad (5)$$

Equation (5) ensures that there is no overlapping of blocks at any stage of the optimization process. The OR operator used here is logical OR function, that is, either one or both constraints are satisfied at each stage of the optimization process. In other words, blocks i and j will overlap each other if neither of the two constraints specified by (5) is satisfied.

III. Promising Infant Clusters

In a constructive technique the ordering criterion plays a very important role in determining the quality of optimized layout design. While a better ordering criterion will definitely contribute in obtaining improved layout designs, a number of different firing orders are still required in order to select the best layout design among various designs obtained using these firing orders. However, if the number of firing orders is too large, the computation time becomes quite excessive. Thus, while increasing the number of firing orders increases the probability of finding a better layout design, it also significantly increases the computational cost. In the presented constructive placement technique, the analogy of "catch them young" is used for early identification of firing orders that are expected to produce better layout designs. In this way, a large number of firing orders can be tried and after only partial placement of few core blocks, the promising firing orders are identified and utilized for complete layout designs. In other words, if n defines the total number of firing orders, partial layouts of core blocks will be obtained for n firing orders but complete layout designs will be obtained only for few promising firing orders. This is explained below.

For layout design of N blocks, let there be M number of core blocks (M < N), as defined by the user. Let there be total "n" firing orders. In the first phase, "n" partial layouts will be constructed block by block, for only M core blocks. These partial layouts, called "Infant Clusters", are obtained for n different firing orders. Among the n Infant Clusters, K best Infant Clusters (with least cost function values) are identified for further development. These K Infant Clusters are called Promising Infant Clusters (PICs). For these PICs, M core blocks have already been placed but the remaining (N – M) blocks are yet to be placed. For each of the K PICs, remaining blocks are placed one at a time using W different firing orders and finally the solution with the least value of cost function is selected as the optimal layout design.

It is important to note that in constructive boundary search procedure the computational cost is proportional to the number of placed blocks. Thus identification of Promising Infant Clusters in initial phase and their full development for all N blocks in the second phase will allow a much larger number of firing orders to be attempted for the same computational cost as is possible with conventional approach of obtaining full layout designs for all firing orders. This two-phase PICs-based optimization process involving multiple solutions is explained in Fig. 1.

IV. Semi-Deterministic Ordering Criterion

It is a common practice to use either a completely random or purely deterministic ordering criterion for a constructive placement procedure for solving block layout problems. While complete random ordering has no logical basis, a purely deterministic ordering criterion restricts the number of firing orders to a limited number. For this reason, it was decided that some kind of randomness needs to be introduced in the deterministic firing orders to provide additional semi-deterministic firing orders. Let there be D number of deterministic firing orders, as defined by the user. The user will also define the number of semi-deterministic firing orders

S for each deterministic firing order. Thus the total number of firing orders "n" is determined as:

$$n = D(S+1) \tag{6}$$

A. Deterministic Firing Orders

The procedure for determining each of the D deterministic firing orders is explained below.

Each firing order comprises of a Lead Block and Follower Blocks. The following ordering function φ_i for ith block is determined for all blocks (i = 1 to N):

$$\varphi_i = A_i^{\gamma} \sum_{j=1}^{N} \alpha_{ij} \tag{7}$$

where A_i is the area of the ith block (w_i x h_i) and γ specifies the user-defined weight on areas of blocks for determining the ordering function. This means that blocks that have more connectivity or flow relationship with other blocks will tend to have a larger value of ordering function. If $\gamma > 0$ then larger size blocks will contribute more towards a higher value of the ordering function. On the other hand, if $\gamma < 0$ then smaller size blocks will contribute more towards a higher value of the ordering function. The blocks' areas will have no impact on the ordering if γ is selected as zero.

The N values of the ordering function φ_i are sorted in the descending order. The first block in this sorted list is taken as the Lead Block for the first deterministic firing order, the second block in the sorted list is taken as the Lead Block for the second deterministic firing order, and so on until Lead Blocks are defined for all D deterministic firing orders.

To determine the Follower Blocks for each deterministic firing order, the ordering function, defined in (7), is calculated for all unplaced blocks with one difference; the summation is carried out only with respect to the already placed blocks. Thus, for the first Follower Block, the ordering function will be determined with respect the Lead Block only, and for the second Follower Block the ordering function will be determined by doing the summation for the Lead Block and the first Follower Block, and so on. This process is repeated for each of deterministic firing order.

B. Semi-Deterministic Firing Orders

For each deterministic firing order, S semi-deterministic firing orders are determined by randomly swapping two of the Follower Blocks in that deterministic firing order. In other words, the Lead Block will remain the same for a deterministic firing order and its S semi-deterministic firing orders. As an example, consider that N=6, D=3, and S=2. The total number of firing orders is given as:

$$n = 3(2+1) = 9$$

Let the first deterministic firing order be given as:

B5, B3, B1, B6, B4, B2

Then its two semi-deterministic firing orders can be as:

B5, B3, B4, B6, B1, B2
B5, B4, B1, B6, B3, B2

That is, block B5 always remains the Lead Block and two of the Follower Blocks are swapped to get a semi-deterministic firing order.

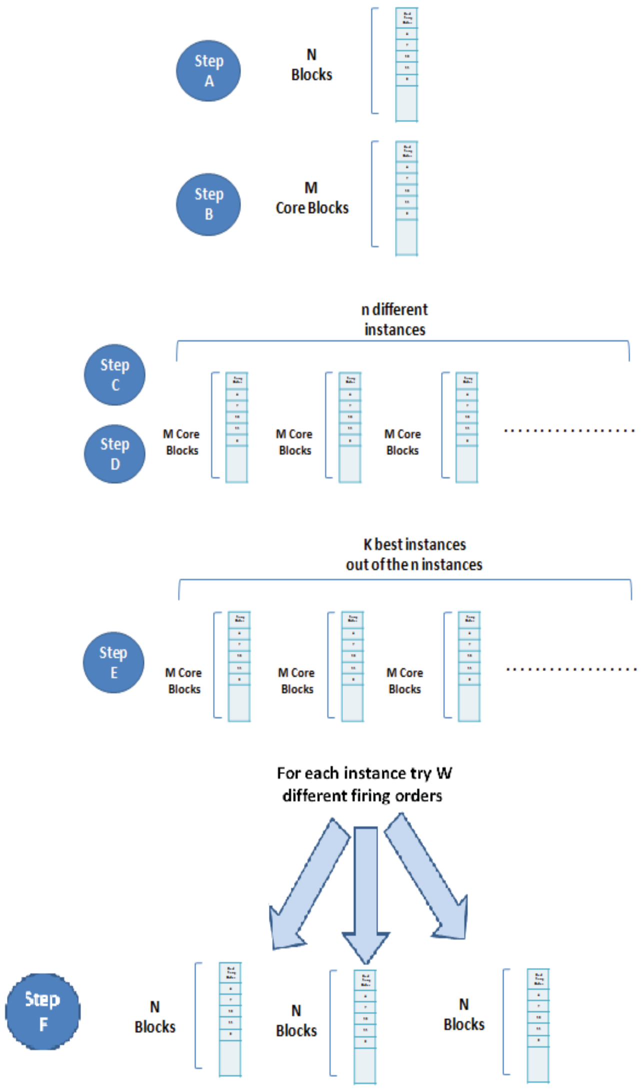

Fig. 1 Two-phase PICs-based optimization process optimal placement

C. Search Strategy

The following procedure is repeated for each instance of the "n" firing orders. Place the Lead Block in the center of a two-dimensional continuous plane and then place the first Follower Block at its optimal position. This optimal position will be somewhere along the edges (sides) of the Lead Block because placing it away from the edges will increase the value of the cost function and placing it any closer will result in overlapping with the Lead Block. To determine the optimal position of the Follower Block, one-dimensional (1-D) search is carried out, using the modified quadratic-fit procedure [32], along all the four sides of the Lead Block and for both possible orientations of the Follower Block. Among the eight optimal positions, the Follower Block is placed at a position (with appropriate orientation) that corresponds to the minimum value of the cost function.

The above procedure is repeated for the next Follower Block in order. However, this time 1-D search will be carried out along each edge of the boundary formed by the placed blocks. Among all the feasible optimal positions at different edges of the boundary, the new block will be placed at a position (with appropriate orientation) that corresponds to the minimum (best) value of the cost function.

D. Enhanced Features

In the original cluster boundary search algorithm [31] the cluster boundary is recalculated every time a new block is placed. This process is time-consuming especially for large-size layout problems. Furthermore, at every position determined by the 1-D search, the algorithm will need to check if the newly placed block overlaps with other blocks. This process would require further computation time. Since for the presented technique the number of firing orders "n" is usually large, it is important that the optimization procedure is fast enough to minimize the total computation time. This consideration has led to the development of enhanced cluster boundary search procedure that rectifies the above two limitations. It updates the cluster boundary rather than recalculate it at the start of every iteration and it defines an overlap-free path so that 1-D search will be carried out without any need to check for overlapping at every placement. With these two enhancements, the presented search procedure becomes highly computationally efficient.

The enhanced search procedure is explained with the help of Fig. 2. A cluster of four blocks is shown in Fig. 2 (a) where the right-most (fourth) block has just been placed at its optimal position after carrying out the above-mentioned search procedure. Prior to placing this fourth block at its optimal position the cluster boundary corresponding to the three already placed blocks was as shown in Fig. 2 (b). After the fourth block is placed at its optimal position there is no need to re-calculate the cluster boundary. Instead, the earlier boundary of Fig. 2 (b) is simply updated to represent the latest cluster boundary as shown in Fig. 2 (c). The next step is to define the overlap-free search path for the fifth block, as shown in Fig. 2 (d). The optimal position for fifth block is searched along this overlap-free path and when it is placed at its optimal position, the cluster boundary is again updated, as shown in Fig. 2 (e).

VI. Implementation and Testing

The above-mentioned procedure was implemented in a computer program written in C++ and C#. The program for the presented technique named PICET (Promising Infant Clusters based Enhanced Technique) was run on Acer laptop using Intel Core i5 CPU running at 2.27 GHz and having 4GB RAM. Its performance was compared for some published benchmark problems [16], [34]-[36] as well as for two large-size benchmark problems taken from VIP-PLANOT [33], a well-known commercially available software package for general-purpose block layout optimization. VIP-PLANOPT uses a pseudo-exhaustive search procedure and produces high-quality layout designs especially for large-size problems. For all test problems the cost function is the same as defined by (1) except for the 11-block problem where the cost function is twice that defined by (1). Also, the distance norms used were the same as specified for given benchmark problems. Both positive and negative values of γ were tested. The problems IDs are as mentioned in [16] and [33].

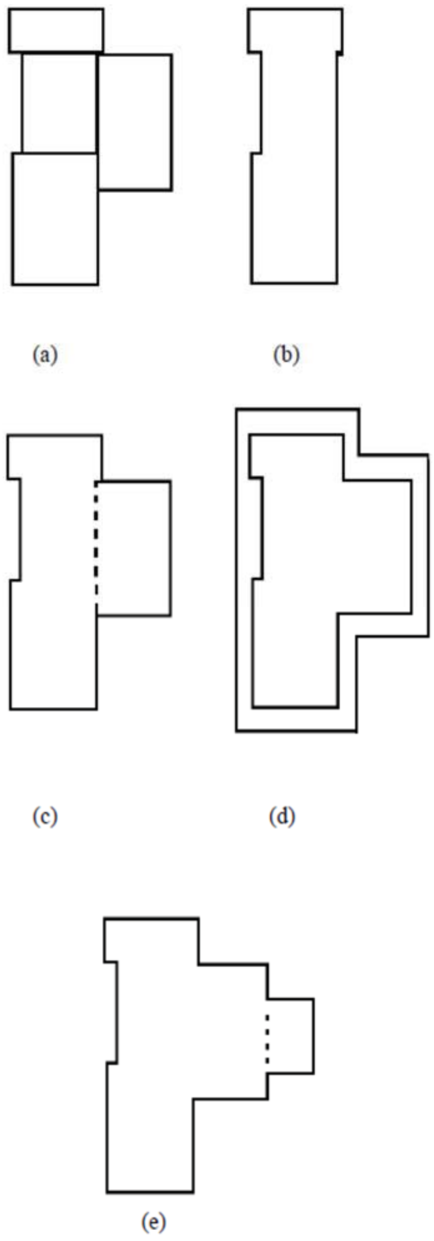

Fig. 2 Enhanced cluster boundary search procedure

The test results are presented in Table I. As can be seen from the presented results, PICET generated the best layout designs for small-size as well as for large-size problems. For the 50-block benchmark problem it took 27 seconds, for the 100-block benchmark problem it took 42 seconds, and for 125-block problem it took 54 seconds to obtain the optimal layouts without any overlaps. This shows that the presented technique can obtain high-quality layouts with low computational cost. For instance, for the 100-block benchmark the cost of optimal layout obtained by PICET is about 10% lower than that obtained by PLANOPT. Similarly, for 125-block benchmark problem the cost for PICET layout is 12.16% lower than that obtained by PLANOPT.

TABLE I
COMPARISON OF RESULTS FOR BENCHMARK PROBLEMS

Problem ID	No. of Blocks	PLANOPT Cost [33]	Published Costs	PICET Cost
OUH-006-Dxx [37]	6	3379	3314 [34] 3274 [16]	3185
OUH-011-IMx [38]	11	2730	2714 [35] 2626 [16]	2418
OUH-050-EOS [33]	50	78,224	71,291 [36] 71,151 [16]	70,657
L100 [33]	100	538,193	-------	484,432
L125B [33]	125	1,096,800	-------	963,426

The optimal layout designs obtained by PICET for 50 and 125 block benchmark problems are shown in Figs. 3 and 4, respectively. It is interesting to note that for many test problems negative weight on blocks' areas (γ) yielded better results. This is quite obvious for the layout design shown in Fig. 3 for the 50-block benchmark problem. The layout was obtained by using $\gamma = -0.75$ and the number of core blocks as 16. As may be noted, the small-size blocks are mostly in the inner part and most of the large-size blocks are on the outer side. The same value of $\gamma = -0.75$ was used for the 125-block layout shown in Fig. 4. Once again, the inner core is mostly composed of small-size blocks while larger-size blocks are on the outer side of the layout. One possible reason for achieving better layout designs in many cases is that when starting with smaller size blocks more blocks with high connectivity or flow relationship form a cluster with reduced cost due to smaller distances among them. With a high positive value of γ, the large size blocks will form an initial cluster occupying large space and therefore smaller blocks with relatively high values of connectivity or flow relationship will have to be placed at larger distances resulting in higher value of cost function.

VII. CONCLUSION

The concept of Promising Infant Clusters is introduced to minimize the impact of ordering on a constructive placement technique for block layout optimization. The enhanced cluster boundary search procedure used in the presented technique is highly efficient and thereby multiple firing orders based on a semi-deterministic ordering criterion can be attempted with minimal computational cost. The test results with small- and large-size benchmark problems, involving up to 125 unequal-area blocks, have shown that the presented technique can generate high-quality layouts with small computational cost.

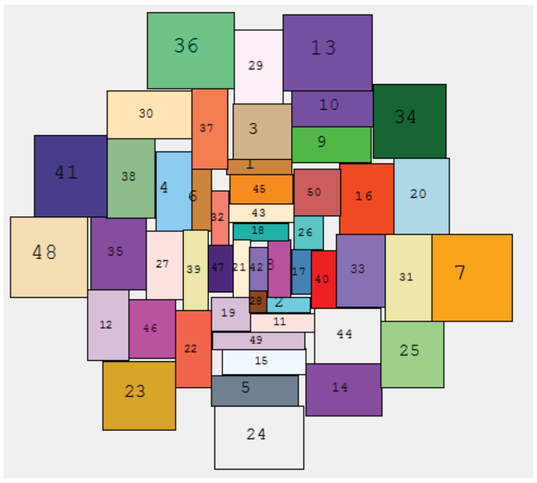
Fig. 3 Optimal layout for 50 block benchmark

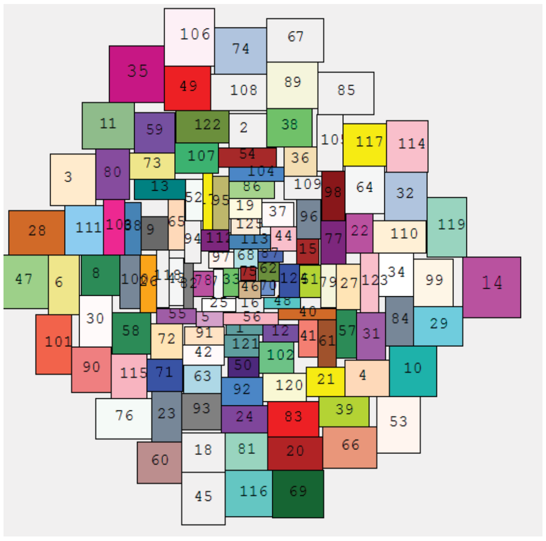
Fig. 4 Optimal layout for 125 block benchmark

REFERENCES

[1] J. Funke, S. Hougardy, J. Schneider, "An exact algorithm for wirelength optimal placements in VLSI design", *Integration, the VLSI Journal*, July 2015.

[2] D. Y. Zaporozhets, D. V. Zaruba, and V. V. Kureichik. "Hierarchical Approach for VLSI Components Placement", *Artificial Intelligence Perspectives and Applications*, pp. 79-87, 2015.

[3] Y. Matsuoka and H. Zhao, "Multi-Objective Optimization Techniques for VLSI Placements", *International Conference on IT Convergence and Security (ICITCS)*, 2014.

[4] C. Hoo, et. al., "Hierarchical congregated ant system for bottom-up VLSI placements", *Engineering Applications of Artificial Intelligence*, Vol. 26, No. 1, pp. 584–602, Jan 2013.

[5] M. F. Anjos and F. Liers, "Global approaches for facility layout and VLSI floorplanning", *Handbook on Semidefinite, Conic and Polynomial Optimization*, Springer US, pp. 849-877, 2012.

[6] D. Hill, et al., "Algorithms and techniques for VLSI layout synthesis", *Springer Science & Business Media*, Vol. 65, 2012.

[7] I. L. Markov, J. Hu, and M. C. Kim. "Progress and challenges in VLSI placement research", *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2012.

[8] B. S. Babu, R. R. Swetha, and K. A. S. Devi. "Comparison of hierarchical mixed-size placement algorithms for VLSI physical synthesis", *International Conference on Communication Systems and Network Technologies (CSNT)*, 2011.

[9] S. Chen, S. Dong, X. Hong, Y. Ma and C.K. Cheng, “VLSI block placement with alignment constraints”, *IEEE Trans. on Circuits and Systems,* Vol. 53, No. 8, pp. 622-626, Aug. 2006.

[10] S.Y. Ho, S.J. Ho, Y.K. Lin, and W. C. Chu, “An orthogonal simulated annealing algorithm for large floorplanning problems”, *IEEE Trans. on VLSI Systems* Vol. 12, No. 8, pp. 874-877, Aug. 2004.

[11] J. Hou, Q. Zhou and X. Hong, “A Thermal-Driven Force-Directed Macro Cell Placement Algorithm,” *International Conference on Communications, Circuits and Systems,* pp. 2420-2423, June 2006.

[12] M. Mir, "Analytical technique for macrocell placement optimization with multiple constraints", *10th IEEE Int’l Conference on Electronics, Circuits and Systems*, pp. 503-506, Dec. 14-17, 2003.

[13] M. Sarrafzadeh, M. Wang, and X. Yang. "Macro-Cell Placement," *Modern Placement Techniques*. Springer US, 2003. 159-174.

[14] M. Mir and M. Al-Saleh, “A constructive procedure for optimizing the placement of macrocells", *Proceedings of 2001 IEEE Int’l. Symposium on Circuits and Systems*, Vol. 5, pp. 57-60, 2001.

[15] J. F. Gonçalves, and M. C. Resende, “A biased random-key genetic algorithm for the unequal area facility layout problem”, *European Journal of Operational Research*, Vol. 246, No. 1, pp. 86-107, October 2015.

[16] I.A. Tasadduq, M.H. Imam, and A. Ahmad, “A hybrid algorithm for optimising facility layout”, *South African Journal of Industrial Engineering*, Vol 26, No. 1, pp 120-134, 2015.

[17] R. Matai, “Solving multi objective facility layout problem by modified simulated annealing,” *Applied Mathematics and Computation*, vol. 261, pp. 302–311, Jun. 2015.

[18] B. Ulutas and A. A. Islier, “Dynamic facility layout problem in footwear industry”, *Journal of Manufacturing Systems*, Vol. 36, pp. 55–61, July 2015.

[19] A. Tasadduq, M.H. Imam and A. Ahmad, “A novel adaptive boundary search algorithm for solving facility layout problems,” *The 43rd Atlantic Schools of Business Conference*, Canada, September 27-29, 2013.

[20] M. Mir and M. H. Imam, "A Hybrid Optimization Approach for Layout Design of Unequal-Area Facilities", Journal of Computers and Industrial Engineering, Vol. 39, pp. 49-63, 2001.

[21] W-C Chian, G. Mudunuri, G. Cai, W. Zhu and X. Xu, “Two-Stage Tabu-Particle Swarm Algorithms for Facility Layout Problem with Size Constraints,” *IEEE Congress on Evolutionary Computation*, pp. 1679-1686, June 2011.

[22] M. Al-Saleh, M. Mir, and A. Hassanin, “Comparison of Enhanced Constructive Layout Optimization Technique with Tabu-Search and Particle Swarm Optimization Methodologies”, Proceedings of 5th International IEOM Conference, pp. 475-481, UAE, March 3-5, 2015.

[23] P. Arikaran, V. Jayabalan and R. Senthilkumar, “Analysis of Unequal Areas Facility Layout Problems,” *Intl. Journal of Engineering* (IJE), Vol. 4, pp. 44-51, Jan. 2010.

[24] A. Drira, H. Pierreval and S. H. Gabouj, “Facility Layout Problems: A Survey,” *Annual Reviews in Control*, Vol. 31, pp. 255-267, 2007.

[25] S. P. Singh and R. R. K Sharma, “A Review of Different Approaches to the Facility Layout Problems,” *Intl. J. Manufacturing Technology*, Vol. 30, pp. 425-433, 2006.

[26] A. R. McKendall, J. Shang, and Kuppusamy, “Simulated Annealing Heuristics for the Dynamic Facility Layout Problem” *Computers and Operations Research*, Vol.33, pp. 2431-2444, 2006.

[27] P. S. Welgama and P. R. Gibson, “A construction algorithm for the machine layout problem with fixed pick-up and drop-off points”, International Journal of Production Research, Vol. 31, pp. 2575-2590, 1993.

[28] J. Balakrishnan, et al. "A hybrid genetic algorithm for the dynamic plant layout problem." *International Journal of Production Economics* 86.2 (2003): 107-120.

[29] P. Corry, and E. Kozan. "Ant colony optimisation for machine layout problems." *Computational optimization and applications* 28.3 (2004): 287-310.

[30] G. Moslemipour and T. S. Lee. "Intelligent design of a dynamic machine layout in uncertain environment of flexible manufacturing systems." *Journal of Intelligent Manufacturing* 23.5 (2012): 1849-1860.

[31] M. H. Imam and M. Mir, "Cluster Boundary Search Algorithm for Building-Block Layout Optimization", Journal of Advances in Engineering Software, Vol. 29, No. 2, pp. 165-173, 1998.

[32] D. G. Leunberger, *Introduction to Linear and Non-linear Programming*, Addison-Wesley, Massachusetts,1973.

[33] Engineering Optimization Software, VIP-PLANOPT: www.planopt.com

[34] A. R. Ahmad, O. Basir, K. Hassanein and M. H. Imam, “An effective module placement strategy for genetic algorithms based layout design”, *International Journal of Production Research,* 44, pp. 1545-1567, 2006.

[35] A. R. Ahmad, *An intelligent expert system for decision analysis & support in multi-attribute layout optimization*. PhD Thesis, University of Waterloo, Canada, 2005. Available online: https://uwspace.uwaterloo.ca/handle/10012/785.

[36] S. Kulturel-Konak and A. Konak, “Linear programming based genetic algorithm for the unequal area facility layout problem”, *International Journal of Production Research,* 51, pp. 4302-4324, 2013.

[37] S. K. Das, “A facility layout method for flexible manufacturing systems”, Int. *J. Prod. Res.,* 31, pp. 279-297, 1993.

[38] M. H. Imam and M. Mir, “Nonlinear programming approach to automated topology optimization”, *Computer Aided Design,* 21, pp. 107-115, 1989.