Design and optimization of layout problem using metaheuristic algorithms

S. Suresh Balaji1* S. Ragunathan2, K. Visagavel3 S. Ganesh Kumar4

1 Department of Mechanical Engineering, Knowledge Institute of Technology, Salem, Tamilnadu, India.
2 Department of Mechanical Engineering, SRG Engineering College, Namakkal, Tamilnadu, India.
3 Department of Mechanical Engineering, Knowledge Institute of Technology, Salem, Tamilnadu, India
4 Department of Mechanical Engineering, Er.Perumal Manimekalai College of Engineering, Hosur, Tamilnadu, India

*Corresponding author mail ID: balajissindia@gmail.com, 9842775097

Abstract. Loop layout is a common layout in the Flexible Manufacturing Systems (FMS), machines arranged around the loop and materials are transported in unidirectional only. The intention of the loop layout problem (LLP) is the determination of regulating of machines around a loop, to minimize the maximum congestion among the family of parts. Artificial immune system (AIS) algorithm, Tabu Search (TS) and Improved Tabu Search (ITS) algorithms are employed to solve these loop layout problems. The algorithm is tested and validated through large and small-sized randomly generated hypothetical problems with minimum required machine sequence. The efficiency of algorithms is compared with existing algorithm for benchmark problems. Computational results reveal that ITS algorithm outperforms the AIS, TS and existing algorithm for large-sized hypothetical problems.

KEYWORDS
Layouts, Metaheuristics, Artificial Immune system, Tabu Search, Improved Tabu Search.
1. Introduction

One of the principal concerns for the layout of a loop flexible manufacturing process is distributing various resources for producing better efficiency. Machines in present FMS are a vital resource, thus it is crucial to ensure a machine does not need to remain idle based on the design of a material processing system. Hence, a vital factor in production system is to determine a layout so that, material movement between the machines should be most efficient. The various types of machine layout are in line layout, Open field layout, Loop layout, Ladder layout and Robot centered cell.

An overview of the loop layout problems are attempted by the researchers. Many researchers have been attended for solving LLPs. Common are linear programming, heuristic and meta-heuristic methods. Afentakis [1] proposed solutions for the LLPs of a loop network, developed the loop layout with a graph where in the nodes resembles the processing elements and the links of the Flexible Manufacturing System (FMS) were resembled by edges. Efficient material flow, operational simplicity and the flexibility are some features of the loop network. Cheng and Gen [2] suggested that using hybrid Genetic Algorithm technique and neighborhood search methodology to arrive at the min_sum and min_max LLP. Bennell et al. [3] described a TS methodology along with a randomized insertion technique to provide solutions for min_max LLP.

Yang et al. [4] detailed two-step heuristic for FMS with one loop directional flow patterns. The problem was in the form of integer linear programming, which might be defined optimally for similar instant smaller sizes. Many of the researchers had adopted mixed model for providing solution for new field layout and with closed loop configuration problems. Kumar et al. [5] suggested dual phase of which one of the phase involves AIS technique to determine solution for the unidirectional LLP by considering min_sum and min_max objectives and whereas, shortcuts are formed in the path of flow for the materials in the layout to improve the overall system performance. .

Kumar et al. [6] proposed a solution for determining the order of the machines are over a loop network, to reduce the count of loop traverse of machinable parts. Nearchou [7] determined DEA to provide solution to the LLP and a methodology for decoding the chromosomes for problems with a nature of permutation property. A simulated annealing technique to determine solution for and to reduce the cost of material handling was proposed by Chae and Peters [8].

Asefvaziri et al. [9] suggested ant colony method for improving loop-dependent network design with shortcut. Hong et al. [10] determined a modified ranking, local improvement with genetic algorithm. Lasrado and Nazzal [11] had implemented queuing theory and genetic algorithm to form closed loop configuration. Glover and Laguna [12] developed tabu search which is applied to lead optimization algorithms in the search for a globally optimal solution. Uneven area configuration layout cases where the motive is to determine a flexible bay configuration with desirable solutions [13]. Potts and Whitehead [14] proposed a combination of scheduling layout and machine layout for a manufacturing facility and the solutions were proposed with the help of a integer programming model. Yang et al. [15] and Yang peter [16] applied fabrication facilities in order to find solutions for making layout material processing facility design problem which is depends upon on a spine layout Systems.

Suhardini et al. [17] developed systematic layout planning to design the industry layout in order to increase production capacity. Kang et al. [18] developed a method for determining the optimal location of production cells in a material handling system with a central loop. The loop layout problem received very little attention in the extensive literature review (Perez-Gosende et al. [19]).

Alduaji and Hassin [20] devised a linear programming methodology for circular open field layout in designing FMSs with goal of reducing production costs to absolute minimum. Hungerlander et al. [21] described a directed circular facility layout problem in which the total weighted sum of all pairs of machine center-to-center distances measured is minimised. The best arrangement of a series of cutting tools on a tool turret is one of the most common real-world implementations of the loop layout problem.

One of the most important aspects to consider for the successful and cost-effective operation of an automated guided vehicle system is facility layout design technique (Gutta et al. [22]). Rai and Jayswal [23] developed particle swarm optimization algorithm for loop layout problem. The monarch butterfly optimization approach was presented by the layout design and quickest single loop material handling path were solved by Kim and Chae [24]. Wenhan et al. [25] described a hybrid population-based incremental learning technique for solving the closed-loop layout problem by simultaneously searching for the best facility placement order and the size of the rectangle loop. A multi stage stochastic programming approach was utilised by Mohammadia et al. [26] for a sustainable closed loop supply chain configuration design with financial decisions.

The major purpose of the circular layout challenge is to lower the overall transportation cost of material flow between facilities. To accomplish this, the appropriate facility placement sequence as well as the rectangle loop's optimal size must be determined at the same time. Although many metaheuristic-based solutions for solving the circular loop layout problem have been proposed, those approaches simply use metaheuristics to find the best facility placement sequence, with the enumeration method determining the ideal size of the rectangle loop (Deng et al. [27]).
Saravanan and Ganesh Kumar [28] reviewed the loop layout problem and it is described the min_max technique to reduce the congestion to a great extent of parts of the sub family with the minimization same as the other part reduction value. By the way the min_max approach can be improvised and it is determined that complicated problems must be adopted, recent optimization methodologies to produce optimal solutions for LLPS and the results can be evaluated with the earlier results of the algorithms. There is no benchmarking for the loop layout problem with distance for huge machines in the literature. However, the study still required to look into large-scale issues. Apply underutilised metaheuristics algorithms to test this type of loop layout problem.

In this paper, loop layout problems are used to: (a) consider large sized problems with unit distance to minimise maximum congestion among the family of parts; (b) consider large sized problems with unit distance to minimise maximum congestion among the family of parts; and (c) consider large sized problems with unit distance to minimise maximum congestion among the family of parts. (b) An improved tabu search strategy was developed for the min max congestion issue. (c) When compared to TS, AIS, and SFHA, the improved tabu search strategy for large loop layout difficulties has improved in benchmark problems. d) Incorporating a novel sort of solution reconstruction into the classic TS paradigm is the ITS method.

2. Problem Description

In this paper a loop layout is considered for a spine layout, as a spine design is relative to the loop arrangement system for an FMS. The dimensions and clearance among the machines which is considered equal with a unit distance. A FMS system to be a loop layout, the system where machines must be placed in a loop structure and the materials must be transferred in a cycle form as revealed in Fig.1.

In a unidirectional loop design, only a loading/unloading location exists where parts leaves and enter the manufacturing system. The number of machines $n$ and $M = \{0, 1, 2, …, n\}$, where 0 denotes the loading/unloading station. The loop arrangement model in the form of transformation of machines $(m_1, m_2, m_3, m_n)$ with a prefix of loading/unloading station 0. Every part is characterized by its machining operation to be performed, the sequencing of machines in which the machining is to be performed. By considering, a chosen part, let us consider that machining is performed in machine j and then followed by machine i. when the location of machine j is below machine i, then the part must move over the area, which is called a reload. The actual number of reloads required to finish the machining is termed as traffic congestion. Our aim to provide the best optimal layout sequence by optimization by considering a set of machining operation constraints performed through hypothetical test problems given in Table 1.

A min_sum case involves an attempt to minimize the congestion of all parts, while a min_max case ensures to minimize the highest congestion that exists among parts of the family [29].

1. The function for the min_max LLP is given below:

$$Cost_{\text{min-max}} = \max (\text{reload}_1, \text{reload}_2, \text{reload}_3)$$

(1)

The min_max goal is to take low in the quantity, mix of demand with high variability. Over time a high balanced loading of machines should be recorded so that the no of reload to be larger for this layout sequencing.

2. Each resolution is determined by the algorithm are not desirable solution however, few of the solutions are close enough to optimal. The efficiency of an algorithm for producing near optimal solution is calculated through the average percentage solution effort (SE %) is expressed through equation 2:

$$SE(\%) = \frac{n_{\text{opt}}}{n_{\text{total}}} \times 100$$

(2)

$n_{\text{opt}}$ defined as number of estimations performed through algorithm to get the optimal solution (very close to best solution).

$n_{\text{total}}$ defined as total number of estimations performed through the algorithm for all solutions

To make the things clear, illustration for first problem is presented here. The required machine progression for each part are given in Table 1

The layout considered is 13-9-6-19-11-18-5-8-16-3-17-4-1-14-12-15-7-10-2-20 which means a location of the machines in loop with machines 13 in the first position, followed through machine 9, and 6 etc. Only one loading/unloading, which taken as beginning point of the spine loop. Commonly, evaluating the demanded part progressions given above, part 1 has to visit 6,3 (upper row) and 4 (Lower row), and needs to crossing the loading/unloading (one reload) after that part 1, its continues to visit 18, 5, 1, 14, 7 and needs one reload after that part 1 visit machine 11 and requires one reload after that part 1 visit machine 13, 9, 16, 17, 2. Further, one reloads needs to visit machine 8, 15 and complete the process. Similarly, part 2 requires five reloads, part 3 requires five reloads, part 4 requires one reload, part 5 requires seven reloads, part 6 requires one reload, part 7 requires five reloads, part 8 requires three reloads, part 9 requires one reload and part 10 requires five reloads. For calculating the cost min_sum value.

$$\text{Cost}_{\text{min-max}} = \max (\text{part 1 reload, part 2 reload, part 3 reload, part 4 reload, part 5 reload, part 6 reload, part 7 reload, part 8 reload, part 9 reload, part 10 reload})$$

= 7

$$\text{Cost}_{\text{min-sum}} = \text{part 1 reload + Part 2 reload + part 3}$$
3. Methodology

3.1. Artificial Immune System Algorithm

AIS are an efficient search methodology resembling the procedure of analytical immunology and the immune system found in organisms. In AIS, clonal choice and affinity maturation of the immune system which causes immune system to grow, which also prevent from the infection of foreign organism in the human body are resembled (De Castro & Timmis [30]). These antibodies consist of antigens, which identify the antigen is multiplied by cloning based on the selection process. The cloned cells mutates produced the affinity in antibodies which the helps antigen to neutralize and eliminate the infectious organisms. New antibodies are developed by receptor editing and mutations, the clones are in proportion to the affinity to the antigen. As a route to transformation, a considerable number of bad cells are removed [31]. The working process of AIS is described in the figure 2.

3.1.1 AIS algorithm consists of the following steps:

Step-1: Initialization of a random population.

Step-2: Affinity assessment of every individual.

Step-3: Clonal Choice- Select the ‘n’ good string depending on the affinity.

Step-4: Clonal expansion- Clone the ‘n’ good strings with the reproduction size, increases with the affinity.

Step-5: Affinity maturation- Mutate every string to get the antibodies and then retain the strings for the next new population.

Step-6: Meta dynamics- Restore the ‘r’ strings with reproduced new ones. Lower affinity strings have the highest chances to be replaced.

Step-7: step 2 to 8 unless a particular level of criterion is achieved.

3.1.2 Numerical example of AIS:

Step-1: The first population (P) strings is obtained randomly up to the limited size. For instance, the sequence 8 6 4 3 5 7 2 10 13 9 12 11 14 18 17 16 15 20 19 considered a layout string.

Step-2: The objective function value (OFV) and the affinities are computed for all the randomly produced s in the population. The affinity value is determined using the formula:

\[ \text{Affinity} = \frac{1}{\text{OFV}} \]

Step-3: Choose the cloning individuals and choice of the individuals lies pon the affinity value.

Step-4: The cloning rate is obtained and described below:

\[ \text{Cloning rate} = \frac{(\text{Solution affinity value} \times \text{population size})}{\text{Total affinity in the population}} \]

This step involves the making of more individuals through cloning that involves minimum OFVs in comparison to maximum OFVs. This applied to determine the alternative clones that exist in the population.

Step-5: Taking in to account the existing string for the inverse and pairwise mutation as follows.

Inverse Mutation: It is obtained through reversing the machines between 5 and 14, as follows.

<table>
<thead>
<tr>
<th>Original String</th>
<th>Mutated String</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 6 19 4 3 5 7 2 10 13 9 12 11 14 18 17 16 15 20</td>
<td>14 18 37 16 15 20</td>
</tr>
</tbody>
</table>

Mutated string is lesser than the existing string then modified replaces the existing one. The pairwise mutation is obtained over the original string.

Pairwise Mutation: It is obtained through the exchange of the machines between 7 & 17 as revealed.

<table>
<thead>
<tr>
<th>Original String</th>
<th>Mutated String</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 6 19 4 3 5 7 2 10 13 9 12 11</td>
<td>14 18 37 16 15 20</td>
</tr>
</tbody>
</table>

As the post process after pairwise mutation, where OFV of the mutated string is smaller when compared to previous string then that of the original string is eliminated through mutated string: otherwise, the initial string is maintained.

Step-6: R% of the solutions which has the high value eliminated with the same R % of randomly created solutions.

Step-7: The step 2 to 7 is performed for the necessary iterations.

Parameter settings are: population size (P) = 60, the number of low-affinity antibodies to be replaced (R) = 20, termination criteria (n) = 300 iterations.

3.2. Tabu Search Algorithm

Normally, The TS framed (Glover [32] [33]) from the local search technique. Tabu Search procedure of the chains changing from one location to other position. The solution obtained by TS, is the best local optimal that can be determined. Accordingly, a superior part of the solution region is discovered while compare with local search Tabu Search involves more space for achieving better solutions. The return to the local optimal results in TS is depends.
upon the technique of prohibitions where few moves are frozen from period to period. But it accounts few complications which are maximum number of local optimal solution in the region, look out pattern, and this state is termed as deterministic chaos. The final local solution and the generations are left, but the solution look out is in narrow solution region. So, the search technique with restricted element of the key region if this portion does not have the global domain which shall ever can be identified.

3.3. Improved Tabu Search Algorithm

ITS recommended to overcome the difficulties of the conventional tabu search and idea of intensification and reconstruction structure which is grounded on three stages of intensification and reconstruction which accepts solution. The ITS proceeds new optimized solution which, further is regenerated, and this process continues. The best output identified through the iterative operation is determined, stored and this outcome would be the prospective consequential result of ITS. The same process is obtained for the number of creations to attain the optimal result (Battiti and Tecchioli [34]: Johnson [35]: Martin and Otto [36]: Schrimpf et al. [37]: Lourenco et al. [38]. The process of ITS, detailed in the figure 3.

3.3.1 ITS Procedure

Step 1: Initiate the first string S₀. Select the number of iterations k = 1. Select S₀ and set S₀=S₁.

Step 2: select the best Sₖ in the local string N (Sₖ).
(a) If the move Sₖ → Sc tabu list set Sₖ+₁ = Sₖ and go to 3
(b) If Sₖ → Sc is not tabu list set Sₖ+₁ = Sₖ

Increase the reverse move to the beginning of the Tabu list and remove the entry on the end. If the objective function G (Sc) < G (S₀), set S₀=Sc.

Step 3: Set k=k+1. Stop if end criteria completed; otherwise go to 2.

Step 4: Intensification- current string just reconstructed.

Step 5: Reconstruction- to keep a level of diversification

Step 6: Accept solution

Alternative’s exploitation and exploration determine the selection of candidate string for the diversification operator until a criterion is met.

3.2.1. ITS Numerical Example:

Numerical example illustrated through 5 m/cs & 3 parts problem. The required machine sequences for each part are assumed as given below.

Part 1 : 3-2-1
Part 2 : 1-2
Part 3 : 4-1-5

Initial sequence = 5 2 1 4 3 (Initial population size 4)
Step 1: S₀=S₁=(5-2-1-4-3). G (S₁) = 4. Set L={ }

Step 2: N(S₁)={ (2-5-1-4-3) (5-1-2-4-3) (5-2-4-1-3) (5-2-1-3-4) }

With respective load costs = { 4 , 4 , 3 , 4 }
S₀=S₁=S₂=(5-2-4-1-3)
Set L = {(1 , 4 )}

Step 3: K=2

Iteration 2:

N(S₂) = {(2-5-4-1-3) (5-4-2-1-3) (5-2-1-4-3) (5-2-1-3-4) }

With respective costs = {3 , 3 , 4}
Sc=S₃ = (2-5-4-1-3)
S₀=S₃
Set L = {(5 , 2) (1 , 4 )}

Step 3: Let K=3

Iteration 3:

N(S₃) = {(5-2-4-1-3) (2-4-5-1-3) (2-5-1-4-3) (2-5-1-3-4) }

With respective costs = { , 3 , 4}
Sc=S₄ = (2-4-5-1-3)
Set L = {(5 , 2) (2 , 1) }

Step 3: K=4, stop if criteria satisfied; otherwise go for step 2

Step 4: Intensification

This phase enhances the present solution and is applied to the freshly rebuilt solution, unlike the first iteration, which intensifies the initial solution (i.e., the “output” of construction). Experimentation has demonstrated that the tabu search-based improvement technique’s pricey runs are unnecessary. The simple tabu search iterations save a significant amount of CPU time. When used in conjunction with the diversification operators, however, classic tabu search might find more near-optimal solutions than extensive runs of basic tabu search.

Step 5: Reconstruction

It's a method of solution reconstruction that allows you to move away from the current local optimum and onto other sections of the solution space. As a result, maintaining a degree of variety at this step is critical: if it is too high, the resultant algorithm may resemble a clear random multi start, and if it is too minimum, the procedure may back to the same solutions on which the reconstruction was performed. It operates in the same manner as GA's mutation procedure does. During the reconstruction method, K percent of the layout solution's total nC2 (total number of pairings that may be exchanged) pair of facilities is swapped with locations.
Step 6: solution Acceptance

In exploration, only the presently best local optimum, is chosen, whereas only the currently best local optimum, is chosen. Each locally optimal solution might be a possible candidate for study diversification. Under extreme conditions, it is also possible to create a new solution from the ground up. It is feasible to achieve the so-called where you are approach, which is worth: in this situation, any new local optimum is accepted for the reconstruction process, regardless of solution quality. The exploitation approach is employed for candidate selection under the diversification operator in the described ITS-based heuristic strategy.

The ITS's usual flow is as follows. It all starts with the development of a one-of-a-kind solution using the traditional TS technique. As a result, the best option was identified first. A given solution is also reconstructed, resulting in a new one. The purpose of rebuilding is not to completely eliminate the current solution. However, since parts of this optimum may mirror those of the globally optimal solution, it is ideal that the ultimate solution inherit certain traits from prior local optimums. As soon as the perturbation procedure is completed, the result is passed to the TS method, which begins immediately. The ITS then returns a new optimized solution, which is rebuilt (or replaced with another local optimum), and so on. Throughout the iterative process, a better solution is discovered and remembered. This method is used for the needed iterations to get the best result.

Parameter setting are: Initial Population sizes = 19, 59 When the number of disimproving moves reaches to a maximum value, no neighbor is generated or an infeasible solution encountered for TS, \( K = 40 \% \). Number of iterations = 300.

4. Result and Discussions

To describe the efficacy of the AIS, TS and ITS, it is assessed with the hypothetical test problems mentioned in Table 1 and benchmark problem taken from Saravanan and Ganesh Kumar [39]. In this first 1 to 4 problems are randomly created hypothetical test instances has been examined. For the problem 1 consists of 20 machines with 10 parts where small machine and small part sized. For the problem 2 to 4 (60 machines 5 parts 60 machines 10 parts, 60 machines 15 parts) machines and the part size increases. The huge number of machines is assumed corresponding to previous papers, and efficacy of technique is tested. The results of AIS and ITS are exposed in Table 2. In this second, 5 & 6th test problems were borrowed for literature for layout problems with 50 machines, 10 and 20 parts. The comparison results of AIS, TS, ITS and existing SFHA are exposed in Table 3 & 4. The projected system is coded in the MATLAB language overall tests were computed on a Pentium IV Microsoft windows system.

The performance of the AIS and ITS is compared for the hypothetical problems based on the min_max approach, unit distance and solution effort. Further, the benchmark problems based on the min_sum and min_max approach without distance and solution effort. From the Table 2 and Figure 4, 5, 6 for the hypothetical test problem 1, AIS determine optimum result than ITS. Test problem 2 to 4 ITS outperforms AIS with minimum distance. For the test problems ITS better than AIS in terms of percentage solution effort (SE)\% spent through the methods, the lower value SE% indicate the quick convergence which leads the optimum result for large sized problems. Machine size and part size increased this will also change the simple problem to complex problem.

It can be incidental that the nearness of solutions in both the approaches, specifies the consistency of performance of a specific tool. Figure 7 the other method display scattered solutions for min_max and min_sum. In addition, min_max or min_sum, while implementing the two approaches. But ITS present’s minimum point for min_max and min_sum and display closer solutions for both which specifies the consistency of performance of the tool.

From the Table 3 and 4 and Figure 8, 9 for the benchmark test problem 5, ITS determine optimum result than IT. AIS and SFHA with a lower level of solution effort SFHA, on the other hand, outperforms the AIS and TS (Kannan et al. [40]; Venkumar and Chandrasekar [41]). ITS outperforms TS, AIS, and SFHA in Test Problem 6. When the ITS method was evaluated with the min max, the findings showed that it performed well for all of the large-sized test issues. The percentage solution effort represents the algorithm's pace of convergence towards the best answer. For the number of test issues, ITS outperformed TS, AIS (Ojha and Chow [42]), and SFHA with a faster convergence rate.

The Figure 10 show that the answer is converging within 300 cycles for configurations. Best congestion is 95 produced and the ITS gives almost the fast convergence for all the layout problems than TS and AIS. TS produced same results at the end of the iterations.

The results produced from the experiments, when the AIS is done well to discover the near optimal solution for the 20-machine size problem. When the problem size increases from 20 to 50 or 60 machines, the performance of ITS in determining the best solution improves. A large number of local optima across the solution space, repeating sequences or the presence of cycle search configuration, and chaotic attractors are some of the downsides of traditional TS. Although the TS chaotic attractors are distinguished by "getting stuck" Because of the absence of cycles and local optima, the search is still limited to a "small region" of the solution space (Glover and Laguna [43]). As a result, the search method will only look at a small portion of the solution space.

ITS, unlike TS, increases its search space with each generation to find a better solution and avoid local minima. Reconstruction and intensification are the two primary tactics. The first strategy requires reconstructing an existing solution that is for shifting away from the current local and to a new location in the solution space. The second technique provides a local refinement to a
“ruined” solution (Martin and Otto [44]) based on the classic TS methodology; theoretically, the enhanced solution is better than the prior iterations' responses. The ITS' principal purpose is to find near-optimal feasible solutions by repeating these stages over and again.

5. Conclusion

This paper proposes ITS & AIS to solve the loop design problems with min_max approach. Computational results show that ITS algorithm outperforms the AIS algorithm for small size problem. Further results show that ITS out performs AIS algorithm for large size problems with less solution effort through exploitation and exploration. ITS is a principle-based optimization policy that "reconstructs and improves" to provide better outcomes than other algorithms. In addition, for large-scale benchmark problems, ITS outperforms the TS, AIS, and SFHA algorithms. Min max and min sum provide a more accurate solution for both, indicating the tool's consistency of performance. The novel approach and mathematical model can be used to solve bidirectional LLPs in the future.

References


**Biography**
1. **S. Suresh Balaji** completed his Undergraduate in Mech. Engg from GCE, Salem. He completed his M.E. in Engineering Design specialization from Paavai Engineering College, Salem. He is currently working as Asst. Prof. at KIOT, Salem. He has 15 years of teaching experience. He has published 19 papers in various national, International journals and conferences.

2. **Dr. S. Ragunathan** has received B.E Degree in GCE, Salem in 1997 and M.E Degree from GCT, Coimbatore. He received his Ph.D in Robotics from Anna University, Chennai in 2008. He has been a Research Associate at the Anna University, Chennai from 2002 to 2006. Presently he is the Principal in SRG Engineering College, Namakkal, Tamilnadu. He has supervised 7 Ph.D. scholars, and at present, he is guiding 2 research scholars. He has published 48 papers in various journals and conferences. He is having 15 years of research experience and 25 years of teaching experience.

3. **Dr. K. Visagavel** has received his B.E. degree in Mech. Engg. And M.E. degree in Thermal Power Engineering from Annamalai University, in India and the Ph.D. degree from the Anna University, Chennai in 2010. Now, he is working as a Professor & Head, Department of Mech. Engg. in KIOT. He has supervised 5 Ph.D and published 91 papers in various journals and conferences. He has 11 years of research and 23 years of teaching experience.

4. **Dr. S. Ganesh Kumar** is an Professor / Mechanical Engineering at Er. Perumal Manimekalai College of Engineering. He completed his PhD from Anna University, Chennai. He has rich experience in various Institutions. He has Published 14 Technical papers in referred journals.

Table1. Minimum required sequence for hypothetical problem
<table>
<thead>
<tr>
<th>P.No</th>
<th>Number of Machines and Parts</th>
<th>Part No.</th>
<th>Machine progression</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20 &amp; 10</td>
<td>1</td>
<td>6-3-4-18-5-1-14-7-11-13-9-16-17-2-8-15</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>17-9-11-8-10-13-2-16-4-20-18-15-6-3-7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>13-2-6-3-14-12-15-7-8-1-10-7-20-19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>7-2-6-11-8-16-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>3-17-1-2-20-8-6-19-14-11-15-12-7-16-10-18-4-13-9-5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>9-2-6-7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>15-9-19-12-3-6-5-8-14-7-1-2-13-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>7-19-5-4-9-16-3-14-13-11-2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>9</td>
<td>3-4-1-6-11-20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>12-6-17-15-13-9-18-14-19-7-11-2-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>5-10-12-22-26-30-33-40-42-25-14-54-18-49-51-1-8-35</td>
</tr>
<tr>
<td>3</td>
<td>60 &amp; 10</td>
<td>1</td>
<td>5-20-32-43-6-8-12-55-58-19-16-17-35-39-22-40</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>1-7-15-23-29-36-44-5-8-14-24-33-41-48-57-51-55-60</td>
</tr>
<tr>
<td></td>
<td></td>
<td>9</td>
<td>3-4-6-32-36-11-19-52-59-2-44-47-49-24-29-20-46</td>
</tr>
<tr>
<td>4</td>
<td>60 &amp; 15</td>
<td>1</td>
<td>11-13-8-7-25-24-21-51-9-14-29-56-6-32-33-34-42-48-44-58-2-3-60-17-5-22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>57-60-59-4-1-15-37-14-5-7-50-26-24-44-36-42-10-2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>9-12-6-17-15-13-30-41-54-58-60-50-43-26-18-4-7-11-23-2-4-25-24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>33-6-3-41-18-53-56-30-5-1-14-24-38-26-7-11-36-47</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>7-22-60-11-31-18-51-44-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>42-51-6-8-14-9-11-3-35-22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>9</td>
<td>21-40-10-6-11-32</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>11-13-31-12-1-44-8-59-29</td>
</tr>
<tr>
<td></td>
<td></td>
<td>11</td>
<td>23-6-58-9-34</td>
</tr>
<tr>
<td></td>
<td></td>
<td>12</td>
<td>10-8-7-59-16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>15</td>
<td>21-41-6-11-22-52-14</td>
</tr>
</tbody>
</table>

Table 2. Outcomes of AIS and ITS for the hypothetical problem

<table>
<thead>
<tr>
<th>P.No</th>
<th>NoM &amp;NoP</th>
<th>Algorithm</th>
<th>Cost min_sum</th>
<th>Cost min_max</th>
<th>Unit distance</th>
<th>SE%</th>
<th>Congestion for each part</th>
<th>Desirable order of machines</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20 &amp; 10</td>
<td>AIS</td>
<td>37</td>
<td>7</td>
<td>814</td>
<td>27.2</td>
<td>4-5-5-1-7-1-5-3-1-5</td>
<td>13-9-6-19-11-18-5-8-16-3-17-4-1-14-12-15-7-10-2-20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ITS</td>
<td>40</td>
<td>8</td>
<td>880</td>
<td>27.5</td>
<td>6-5-4-1-8-1-4-3-1-7</td>
<td>11-12-8-16-3-10-18-14-5-13-7-15-20-17-4-1-9-2-6-19</td>
</tr>
<tr>
<td>Algorithm</td>
<td>Cost min_sum</td>
<td>Cost min_max</td>
<td>SE (%)</td>
<td>CPU time</td>
<td>Congestion for each part</td>
<td>Optimal order of machines</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>--------------</td>
<td>--------------</td>
<td>--------</td>
<td>----------</td>
<td>--------------------------</td>
<td>--------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Algorithm</td>
<td>Cost min_sum</td>
<td>Cost min_max</td>
<td>SE (%)</td>
<td>CPU time</td>
<td>Congestion for each part</td>
<td>Optimal order of machines</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>--------------</td>
<td>--------------</td>
<td>--------</td>
<td>----------</td>
<td>--------------------------</td>
<td>--------------------------</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4. Comparison of the Improved Tabu Search Algorithm with others for 50-machine, 20-part problem (Saravanan and Ganesh Kumar [16])
Fig 1. Arrangement of Loop arrangement
Fig. 2. Working Process of AIS algorithm

- Determine value and rate of cloning of every antibody
- Prepare clone antibodies
- Prepare inverse mutation
- Formulate pairwise mutation and OFV
- Calculate the OFV of every antibody and sort in ascending order
- Mutated String
- Yes = Is clone OFV > mutated OFV
- Mutated String
- No
- Cycle = cycle + 1
- Yes = Is clone OFV > mutated OFV
- Mutated String
- No
- Clone = True string
- After cloning and mutation new string
- Filter the string in ascending order of OFV and delete excess to obtain original pop size
- Receptor editing, robust replace heuristic and new strings for next cycle
- End
Fig. 3. The Process of the ITS algorithm

Figure 4. Comparison of ITS and AIS for large-sized loop layout problem with min_max
Figure 5. Comparison of ITS and AIS for large-sized loop layout problem with Unit distance

Figure 6. Comparison of ITS and AIS for large-sized loop layout problem with SE (%)

Figure 7. Comparison of Min_Max and Min_Sum for 20 M/Cs 10 parts, 60 M/Cs 5 parts, 60 M/Cs 10 parts and 60 M/Cs and 15 Parts problems

Figure 8. Comparison of the ITS with others for min_max approach, SE (%), CPU to the 50-machine, 10-part benchmark problem (Saravanan and Ganesh [16])
Figure 9. Comparison of the min_max and min_sum approach, to the 50-machines, 10, 20-parts benchmark problems (Saravanan and Ganesh [16])

Figure 10. Comparison of the algorithm convergence for min_sum approach to the 50-machine, 10-part problems