A Heuristic Approach for Balancing Mixed-Model Assembly Line of Type-I using Genetic Algorithm

Abdullah-Al-Mamun and Mohammad Mamun. Chowdhury

Abstract— The Mixed Model Assembly Line is becoming more important than traditional single model due to the increased demand for higher productivity. In this paper, a set of procedures for mixed-model assembly line balancing problems (MALBP) is proposed to make it efficiently balance. The proposed procedure based on Genetic Algorithm can perform improved and efficient allocation of tasks to workstations for a pre-specified production rate and address some particular features, which are very common in a real world mixed model assembly lines (e.g., use of parallel workstations, zoning constraints, resource limitation). The main focus of this paper is to study and improve the existing Mixed Model Balancing framework. Here a heuristic is proposed to reassign the tasks that violate the capacity and zoning constraints. The new method minimizes the total number of workstation with higher efficiency and is found suitable for both small and large scale problem.

Keywords— Genetic Algorithm, Line Balancing, Mixed model, MALBP.

I. INTRODUCTION

In modern day to day life people are becoming more and more captivated to product that is customized to meet their personal need. For companies to be competitive they also need to manufacture products with high quality levels and at low costs. So the efficiency of assembly line is a much more important issue in the modern manufacturing system than ever before.

In Mixed Model Assembly Line Balancing problem (MALBP) different but similar model of same product is assembled simultaneously in the same assembly line. MALBP can be constructed in two ways [2]. For MALBP-I the number of workstations are minimized for a given cycle time. In MALBP-II the cycle time is minimized for a given number of workstations. In type I problem cycle time is fixed. So it is more often used in the design of a new assembly line. Type II problem is more suitable for existing line where increased production rate can be achieved by rearranging the tasks. Assembly line balancing problem was first mathematically formulated by selveson [1] and, since then, extensive research has been done in the area. Several procedures are formulated to balance mixed model assembly line.

Parallel workstation provides greater flexibility in design because it allows tasks greater than the cycle time. But with replication a worker perform greater number of tasks which contradicts the main advantage of assembly line: division of labor. So the replication process of workstation needs to be controlled. A limit on the number of parallel workstations to control the replication process was defined in [3, 4]. A limit was included on the number of tasks per workstation in [5]. These approaches allow the replication of workstations even when task time is much shorter than the cycle time. This may lead to excessive number of parallel workstations. This paper uses a mechanism to control the replication of the workstations which is based on the one originally developed by Simaria and Vilarinho for the single model assembly line balancing problem and later extended to the MALBP-I in [6]. Here a heuristic is proposed to assign tasks to workstations when applying Genetic Algorithm to solve the problem.

II. REPRESENTATION OF THE PROBLEM

There are M models of a product; N number of task needs to be performed to get the completed product. Every model may not need all N tasks to be performed for their completion. There is a fixed demand for each model (Dm) and the Cycle time of Output is C. The problem is to allocate the tasks to workstations in such a way that a minimum number of workstation is needed and the idle time of each workstation becomes a minimum. It will lead to minimum cost because minimizing workstation directly minimizes cost and smaller idle time save production time which ultimately reduce cost.

In the addressed problem, the assembly line is configured to produce a set of similar models of a product (m=1,…,M), in any order or mix, over a pre-specified planning horizon, P. The forecasted demand, over the planning horizon being Dm for model m, requires that the cycle time of the line to be,

\[ C = \frac{P}{\sum_{m=1}^{M} D_m} \] (1)
The overall proportion of the number of units of model \( m \) being assembled, i.e., the production share of each model, is computed by,

\[
q_m = \frac{D_m}{\sum_{p=1}^{P} D_p}
\]  

Each model has its own set of precedence relationships, but there is a subset of tasks common to all models. Hence, the precedence diagrams for all the models can be combined and the resulting one has \( N \) tasks \((i=1,...,N)\) are the task numbers of the tasks in the combined precedence diagram). The time required to perform task \( i \) on model \( m \), \( t_{im} \), may vary among models \((t_{im}=0 \text{ means that model } m \text{ does not require task } i)\).

The proposed approach uses a mechanism to control the replication of workstations, based on the approach originally developed for the single-model assembly line balancing problem [6]. The number of replicas of a workstation \( k \), \( r_k \), is determined by its longest task processing time (for all models) and it is given by

\[
r_k = \left\lceil \frac{\max_{m=1,...,M;i=1,...,N} \{ t_{im}x_{ik} \} }{\text{MRT}} \right\rceil (k = 1,...,S)
\]  

Where MRT (Minimum Replication Time) = Cycle Time.

\[
x_{ik} = \begin{cases} 1, & \text{If task } I \text{ is assigned to workstation } k \\ 0, & \text{Otherwise} \end{cases}
\]

\[
r_k = \begin{cases} 1, & \text{workstation can be replicated} \\ 0, & \text{Otherwise} \end{cases}
\]

\[
s_{km} = \text{Idle time of workstation } k \text{ due to model } m
\]

\[
S' = \text{Total number of workstation with replica}
\]

\[
S_k = \text{Idle time of station } k \left( \sum_{m=1}^{M} q_m \cdot S_{lm} \right)
\]

The objective function of the balancing problem is composed of three terms. First term indicates the minimizing the number of the workstation to which the last task of the precedence diagram was assigned. The second term is expressed by the term \( B_{b-s} \) which indicates the balance of workload within a workstation. The third term expressed by \( B_{w-s} \) which indicates the balance of workload between workstations. So it can be written as,

\[
\min Z = \sum_{k=1}^{S} k \cdot x_{ik}
\]

\[
+ \frac{S'}{S' \cdot 1} \sum_{m=1}^{M} q_m \left( \sum_{k=1}^{S} \frac{S_{lm}}{S'} \cdot \frac{1}{S'} \right)^2
\]

\[
+ \frac{M}{S'(M - 1)} \sum_{k=1}^{S} \sum_{m=1}^{M} q_m S_{lm} \left( \frac{1}{S_k} \right)^2
\]

Subject to,

\[
\sum_{k=1}^{S} x_{ik} = 1 \quad i = 1,2,...,N
\]

\[
\sum_{k=1}^{S} k x_{ak} - \sum_{k=1}^{S} k x_{bk} \leq 0 \quad a \in N, b \in Fa
\]

\[
\sum_{k=1}^{S} k x_{ak} - \sum_{k=1}^{S} k x_{bk} = 0 \quad (a,b) \in ZP
\]

\[
x_{ak} + x_{bk} \leq 1 \quad (a,b) \in ZN
\]

\[
s_{km} = \text{Idle time of workstation } k \text{ due to model } m
\]

\[
S' = \text{Total number of workstation with replica}
\]

\[
S_k = \text{Idle time of station } k \left( \sum_{m=1}^{M} q_m \cdot S_{lm} \right)
\]

III. GENETIC ALGORITHM APPROACH

Genetic algorithm is based on the heuristic concept for solving complex optimization problems which is based on manipulating a population of solutions by genetic operators like selection, crossover and mutation. The main challenge of the application of GA to the assembly line balancing problem is the development of good encoding schemes and genetic operators in order to attain feasible solutions.

A. Representation Scheme

The standard encoding scheme assigns directly the tasks to the workstations in a balancing solution. Each chromosome is a string of length \( N \) (number of tasks) where each element represents a task and the value of each element represents the workstation to which the corresponding task is assigned. Fig.1 shows the encoding of line balancing problem into chromosomes.

![Fig.1 Standard Encoding and corresponding balancing solution](image)

Five common priority rules are used to form the initial population,
• Maximum processing time for all models - \( \max_m(t_{im}) \)
• Maximum average processing time - \( t_i = \sum_m q_m \cdot t_{im} \)
• Maximum ranked positional weight
• Maximum number of direct successors
• Maximum number of total successors of combined precedence diagram.

Each time a task must be selected for assignment, from the set of available tasks, the heuristic randomly selects the priority rule to be used and create an initial population of 20 solutions.

When an unassigned task is found in the sorted array of a randomly selected priority rule, the formation of new workstation and replication follow the following rule,

1: **If** Task time > Cycle time
2: **If** the resulting number of replica ≤ MAXP
3: check precedence and zoning constraint before replication of current workstation
4: **else**
5: check precedence constraint before creating and replicating new workstation
6: **else**
7: **if** Idle Time < Task Time
8: check Precedence constraint before creating new workstation
9: **else**
10: check Precedence and zoning constraint before entering the task to current workstation
11: **if** task is still unassigned
12: select another priority rule randomly

B. Fitness Function
The main goal of MALBP of type-I is to minimize the number of workstations for a given cycle time. This goal can be achieved by calculating Weighted Line Efficiency (WE),

\[
WE = \sum_{m=1}^{M} q_m \left( \sum_{i=1}^{N} \frac{t_{im}}{S C} \right)
\]  \hspace{1cm} (5)

Now, the goal of genetic algorithms is to find the fittest individual over a set of generations. The fitness function is then, typically, a maximization function. In genetic algorithm, the better fit solutions survive across generation. The selection policy should ensure that, it happens every time. In this procedure, the fitness function is a combination of the objectives to achieve for MALBP-I, namely, the maximization of the weighted line efficiency (WE) and the smoothing of workloads between (\( B_{bs} \)) and within (\( B_{ws} \)) workstations. The fitness function is computed as follows,

\[
\max F = \lambda WE - B_{bs} - B_{ws}
\]  \hspace{1cm} (6)

The lower the number of operators and the values of function \( B_{bs} \) and \( B_{ws} \), the higher the value of \( F \). As \( B_{bs} \) and \( B_{ws} \) are within the value range \([0, 1]\), the term WE is dominant for \( \lambda > 1 \), so, the procedure minimizes the number of workstations before the secondary goals become active.

C. Selection and Genetic Operators
The main genetic operator is the crossover, which has the role to combine pieces of information from different individuals in the population. Here single point crossover is applied. Two parents (\( P_1 \) and \( P_2 \)) are selected randomly and a crossover point \( C_p \) is generated from 1 to \( S \). \( P_1 \) and \( P_2 \) in figure 2 may have different total number of workstations. Then the offspring workstation number will be that of parent with higher number of workstation. But the range of Crossover point will be the lower number of workstations between parents. To create the offspring \( O_1 \) workstation assignment from 1 to \( C_p \) is copied from \( P_1 \) and from \( C_p+1 \) to last workstation is copied from \( P_2 \). To produce the offspring \( O_2 \) workstation assignment from 1 to \( C_p \) is copied from \( P_2 \) and from \( C_p+1 \) to last workstation is copied from \( P_1 \). Fig.2 shows the generation of offspring through crossover.

![Fig. 2 Generation of offspring from parent solutions (Cp=5)](image)

After crossover Precedence and zoning constraints may be violated and the crossover produces some tasks without any workstation associated. These tasks and some other tasks are then reassigned. A two stage heuristic is proposed for this task reassignment problem. It does not reassign tasks of workstation with low workload. But if any predecessors of a task remain unassigned then all the successor of the task is made unassigned. In first stage no new workstation is created only replicas are formed for task time being larger than cycle time. Task those couldn’t be assigned in the first stage are assigned in second stage. In the second stage every time a task is assigned a new workstation is formed.
D. Mutation

Mutation is performed in iteration with probability 0.02. 10% of total tasks are randomly selected and made unassigned. These tasks are then again reassigned according proposed reassignment procedure. The new offspring produced by mutation do not replace the parent but added to the solution space for selection.

E. Replacement Strategy and Stopping Criterion

From 20 parents and 20 children, the best 20 solution are selected for next stage crossover and mutation. The process stops after 200 consecutive iterations.

IV. COMPUTATIONAL RESULT

The heuristic is tested for 14 problems, whose main characteristics are shown in the first columns of table 3 namely the number of tasks of the combined precedence diagram (N), the number of models (M) and the assembly line cycle time (C).

<table>
<thead>
<tr>
<th>Prob No</th>
<th>N</th>
<th>M</th>
<th>C</th>
<th>LB_{p_{\text{mix}}}</th>
<th>Solution from GA D(%)</th>
<th>D(%) from [6]</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>21</td>
<td>10</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>21</td>
<td>10</td>
<td>13</td>
<td>15</td>
<td>15</td>
<td>15.4</td>
</tr>
<tr>
<td>5</td>
<td>25</td>
<td>10</td>
<td>14</td>
<td>15.4</td>
<td>15</td>
<td>7.14</td>
</tr>
<tr>
<td>6</td>
<td>25</td>
<td>10</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>28</td>
<td>10</td>
<td>19</td>
<td>19</td>
<td>19</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>28</td>
<td>10</td>
<td>18</td>
<td>18.5</td>
<td>18</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>30</td>
<td>10</td>
<td>15</td>
<td>17</td>
<td>16</td>
<td>6.67</td>
</tr>
<tr>
<td>10</td>
<td>30</td>
<td>10</td>
<td>17</td>
<td>18.5</td>
<td>18</td>
<td>5.9</td>
</tr>
<tr>
<td>11</td>
<td>32</td>
<td>10</td>
<td>16</td>
<td>17</td>
<td>17</td>
<td>6.25</td>
</tr>
<tr>
<td>12</td>
<td>32</td>
<td>10</td>
<td>17</td>
<td>18.5</td>
<td>18</td>
<td>5.9</td>
</tr>
<tr>
<td>13</td>
<td>35</td>
<td>10</td>
<td>20</td>
<td>23.6</td>
<td>23</td>
<td>15</td>
</tr>
<tr>
<td>14</td>
<td>35</td>
<td>10</td>
<td>21</td>
<td>24.8</td>
<td>24</td>
<td>14.3</td>
</tr>
<tr>
<td>15</td>
<td>70</td>
<td>10</td>
<td>41</td>
<td>41.3</td>
<td>41</td>
<td>0</td>
</tr>
<tr>
<td>16</td>
<td>70</td>
<td>10</td>
<td>39</td>
<td>43</td>
<td>43</td>
<td>10.3</td>
</tr>
</tbody>
</table>

The precedence diagrams are taken from [7] except problem 5 and 6 which are taken from [6]. The task time of these problems are obtained from Dr. A.S. Simaria and are used in [6], [8], [9] and in many other paper by same author. A comparison between the minimum value generated by the proposed algorithm and the lower bound (LB_{p_{\text{mix}}}) is also given (D(\%)). The result shows that the Modified Genetic Algorithm produces satisfactory results for both small and large problem with acceptable amount of time.

V. CONCLUSION

In this research a modified procedure is proposed to solve Mixed Model Assembly Line Balancing Problem of Type-I using Genetic Algorithm Framework. The proposed model minimizes the total number of workstations and allows the user to control the replication process. The major contribution of this study is to define the reassignment process clearly to handle numerous conditions that can evolve during task allocation. It also controls the replication process of workstations.

VI. ACKNOWLEDGEMENT

The authors show their gratitude to the authority of Bangladesh University of Engineering and Technology (BUET) for their support and supervision as the research work was conducted under the department of Industrial and Production Engineering (IPE).

REFERENCES