Application of MATLAB to Create Initial Solution for Tabu Search in Parallel Assembly Lines Balancing

G.R.Esmaeilian, S.Sulaiman ,N.Ismail, M.M.H.M.Ahmad, M.Hamedi

Department of Mechanical and Manufacturing, Faculty of Engineering, University Putra Malaysia,
43400, Serdang, Selangor, Malaysia.

Abstract

In comparison with the exact mathematical methods, the heuristic models are different to provide solutions close to the optimal one, saving time of processing. With the appearance of the Tabu Search by Fred Glover in 1986, different applications have been arisen from the procedure to solve different problems for the classic problems of assembly line balancing and parallel mixed model assembly line. Here, an adaptation to an existing code appears which is applied to the problem of allocating and assigning mixed model tasks for the reconfiguration of distributed generation and balance with parallel assembly line. The model provides optimal or near optimal results in terms of results obtained from calculations compared to the other methods.

Keywords:
Parallel assembly lines, Heuristic method, MATLAB.

1. Introduction

The assembly line can be defined as the movement of the work piece from one task to the next. The tasks that must be performed on the product are divided among workers, so that each worker performs the same operation on the product, which was passed before him. The balance of an assembly line is the assignment of these tasks along a production line in order to increase the production efficiency.

Many publications are available concerning the design, balancing and scheduling for Single, Multi and Mixed-Product lines. Line balancing was the main design issue in the early studies of assembly line design and is addressed in many publications [1]. The focus of this research was the single product assembly line with deterministic task times. The lines were assumed to be dedicated and were mainly balanced for a known cycle time in the simplest form [2].

The problem of balancing an assembly line is a classic Industrial Engineering problem. Even though much of the work in this area goes back to the mid-1950s and early 1960s, the basic structure of the problem is relevant to the design of production systems today, even in automated plants [3]. The assembly line balancing problem defined as assigning tasks to the workstations that minimize the amount of idle time of the line with satisfied specific condition. The first condition is that the total task time assigned to each workstation should be less than or equal to the cycle time (the time interval between two successive outputs). The second condition is the task assignments should follow the sequential processing order of the tasks [4].

Tabu search is a mathematical optimization method, fitting to the class of local search techniques that enhances the performance of a local search method by using memory structures: once a potential solution has been determined, it is marked as "tabu" (thus the name) so that the algorithm does not visit that possibility repeatedly. Tabu search is generally attributed to Fred Glover. It also has been applied successfully to maximum satisfying ability problems[5].

The performed researches in the 70s and 80s focused almost exclusively on the development of exact methods to solve the basic assembly line problem. Tabu search is a "higher level" heuristic procedure for solving optimization problems, designed to guide other methods (or their component processes) to escape the trap of local optimality. Tabu search usually has obtained optimal and near optimal solutions to a wide varieties of classical and practical problems.

Tabu search (TS) is a meta-heuristic strategy for solving combinatorial optimization problems. Tabu search was introduced by Glover[5, 6] as a technique to overcome local optimality. The underlying idea is to forbid some search directions at a present iteration in order to avoid cycling, but to be able to escape from a local optimal point. This strategy can make use of any local improvement technique [7-10]. Furthermore, tabu search is an Adaptive Memory Programming (AMP) [11] that can be superimposed on many other methods. The major theme behind TS is to incorporate flexible memory (short-term and/or long-term) functions into the search procedure. Hence, the search process can avoid a move that reinstate past solutions and prevents being trapped at locally

Manuscript received October 5, 2008
Manuscript revised October 20, 2008

optimized solutions. This method is distinct from the SA and GA methods in that SA and GA have low memory capability and are probabilistic random search methods, while TS takes advantage of the history of the search process and embeds it into the search process. Many problems are successfully solved using tabu search.

2. MATHEMATICAL MODEL

Here, the adaptation of a TS code will be introduced, designed for ALB to find the balancing solution. This research studies the problem of assembly line design, focusing on lines paralleling and job distributing. Three problems are focused including model formulations, cycle time minimizing by the given number of stations, the productivity maximizing with determining parallel lines and scheduling mixed model parallel assembly lines by using Tabu search. The latter formulation can be demonstrated by several examples and compare with another research method that uses some numerical example, for other kinds of assembly system, which try to reduce their required cycle time. It is shown that the problem of Mixed-Model assembly line can be treated as a special case of parallel assembly lines problem. The algorithm is designed to investigate and demonstrate the influence of system parameters, such as number of product in mixed-model and different cycle time, on the balancing improvement due to line paralleling. Using an NP-hard formulation developed for the combined problem of parallel line with equipment selection; the approach presented here is quite different from [4, 12]approaches and the goal is to balance mixed-model assembly line together with parallel specification. A newly developed programming model for the balancing mixed-model parallel line is hereby presented. It is known that all the assembly line balancing problems have an NP-hard nature, and an optimal solution for middle or large-scale problems is not sufficient. Therefore, it is not suitable for practical applications, it has been developed by using MATLAB software to calculate and compare the result. The mathematical model of the problem is given as follows:

$$ST_{kh} = \sum_{h=1}^{H} \sum_{k=1}^{NS_h} x_{imhk} \times t_{im}$$

$$S = \bigcup_{h=1}^{H} \max\{ST_{kh} \to \forall k\}$$

$$ST_{kh_i} \bigcap ST_{kh_j} = \varnothing$$

$$i, j \in S, i \neq j$$

$$C = \max\{S\}$$

$$objective: \min C$$

Objective function: $$\min\left(\max\left(\sum_{h=1}^{H} \sum_{k=1}^{K_H} x_{imhk} \times t_{im}\right)\right) \quad (1)$$

Constraints:

$$\sum_{h=1}^{h=H} \sum_{k=1}^{NS_h} X_{imhk} = 1 \qquad for \qquad i = 1,...,N_m$$

$$m = 1,...,|M|. \quad (2)$$

$$\sum_{m=1}^{|M|} \sum_{k=1}^{NS_h} \sum_{i=1}^{N_m} X_{imkh} \times t_{im} \le C_h \qquad for \qquad h = 1,...,H.$$

(3)

$$X_{imkh} \in \{0,1\} \qquad for \quad i, m, h, k.$$

$$NS_h = \left\lceil (\sum_{T} t_{im}) / C_m \right\rceil^{+} \quad (4)$$

The first step consists of reading data from excel sheet and categorized them in to different matrix for each line and followed by calculating the primary number of workstations according to equation 4 to each line considering the present line. The algorithm will start from the initial number of workstation.

The second step categorizes tasks. For each line a task will be selected which is the number of all tasks which must precede task before selected i must be zero. So, this section will be repeated to find i. After that the algorithm starts to calculate E_i and L_i considering to equation number 5 and 6.

$$E_{ih} = \left\lceil (t_{im} + \sum_{i \in P_{im}} t_{im}) / C_m \right\rceil^{+} \quad (5)$$

$$L_{ih} = NS_h - \left\lceil (t_{ih} + \sum_{i \in F_{im}} t_{im}) / C_m \right\rceil^{-} \quad (6)$$

Once the minimum ST_{kh} is selected, the position of the minimum ST_{kh} in the selected line is k. Task k should be assigned to other workstations which are not outside of E_i and L_i in a way the ST_{kh} should not be bigger than the cycle time .

In continue, after assigning task i , this task will be dropped from the set of tasks and one will less from each of remaining task in NP. In addition, this circle continues until all tasks for all lines completed and assigned to the best place. The heuristic method coding is shown as continues:

```
E = zeros(NSmax,M);
L = zeros(NSmax,M);
X = zeros(NSmax,M,M,NSmax);
ST = zeros(NSmax,M);
ST1 = zeros(NSmax,M);
h=1;
NS0=0;
NS1=0;
while h<=M
   m = h;
   t = T(m:m,:);
   t = t(1:NS(m));
   Im = I(m:m, :);
   Im = Im(1:NS(m))';
   NSh = floor(sum(t)/C(m))+1;
   NS0=NS1+NS0;
   NS1 = NS(m);
   if m ~= 1
      np = NP((NS0+1):(NS0+NS1),:);
   else
      np = NP(1:NS1);
   end
   a1 = numel(np);
   while a1 > 0
      a2 = sum(np == 0);
      while a2 > 0
         b=find(np == 0);
         c=b(1,1);
         U=Im(b);
         i=U(1,1);
         z=find(Im == i);
         np(c)= [];
         Im(z)= [];
         stpi=0;
         a3=NS0+i;
         if NP(a3) > 0
            for j=1:NP(a3)
               s=P(a3,j);
               stpi=stpi+T(h,s);
            end
         else
            stpi=0;
         end
         E(i,h)=floor((T(h,i)+stpi)/C(h))+1;
         %disp(sprintf('E(%d,%d) = %0.0g', i, h, E(i,h)));
         stfi=0;
         a4=NS0+i;
         if NF(a4) > 0
            for j=1:NF(a4)
               s=F(a4,j);
               stfi=stfi+T(h,s);
            end
         else
            stfi = 0;
         end
         L(i,h) = NSh-floor((T(h,i)+stfi)/C(h));
         %disp(sprintf('L(%d,%d) = %0.0f', i, h, L(i,h)));
         k=E(i,h);
         a5 = X(i,m,h,k);
         while a5 ~= 1
            ST(k,h) = T(h,i)*X(i,m,h,k)+ST(k,h);
            ST1(k,h) = ST(k,h)+T(h,i);
            if k <= L(i,h)
               if ST1(k,h) <= C(h);
                  ST(k,h)=ST1(k,h);
                  X(i,m,h,k)=1;
               else
                  k=k+1;
               end
            else
            NSh = NSh+1;
            L(i,h) = k;
               if ST1(k,h) <= C(h)
                  X(i,m,h,k) = 1;
                  ST(k,h)=ST1(k,h);
                  L(i,h) = k;
               end
            end
            a5 = X(i,m,h,k);
            if a5 == 1
               disp(sprintf('E(%d,%d) = %0.0g', i, h, E(i,h)));
               disp(sprintf('L(%d,%d) = %0.0f', i, h, L(i,h)));
               disp(sprintf('X(%d,%d,%d,%d) = %0.0f', i, m, h, k,
X(i,m,h,k)))
               disp(sprintf('ST(%d,%d) = %0.0f', k, h, ST(k,h)));
            end
         end
```

3. Numerical example

The precedence diagram and task times are used from the sample of literature for this numerical example, despite precedence diagram and relative data of task times of line one in line number two are repeated.

In the new obtained balance, all products have been produced for each cycle time in lines 1, 2, and 3, respectively. Therefore, for Sawyer problem that is shown in Figure (1) new cycle times required for the new balance is presented. Consequently, overall line cycle time are improved by a new balance.

In Figure 2, cycle time improvement for given example from Tonge are presented. In addition, repeated test and

the result are shown in Figure 3 to 5 for Arcuse1, Mertens and Jaeschke.

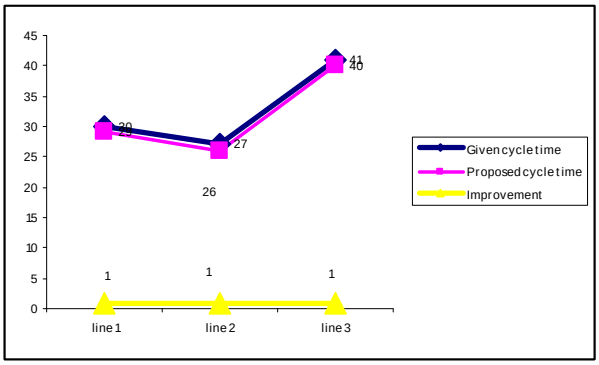

Figure 1. Sawyer problem for three parallel lines.

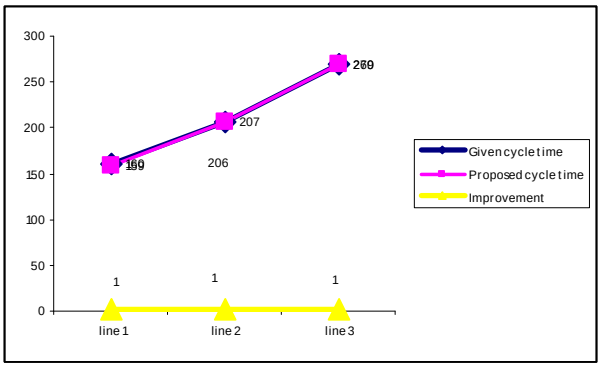

Figure 2. Tonge problem for three parallel lines.

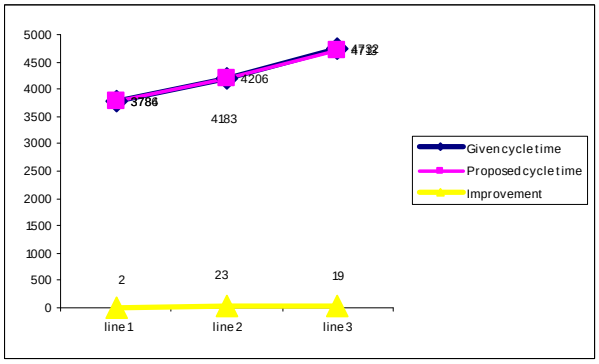

Figure 3. Arcuse 1 problem for three parallel lines.

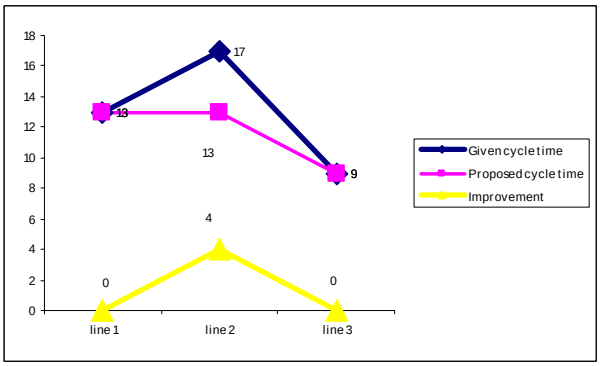

Figure 4. Mertens problem for three parallel lines.

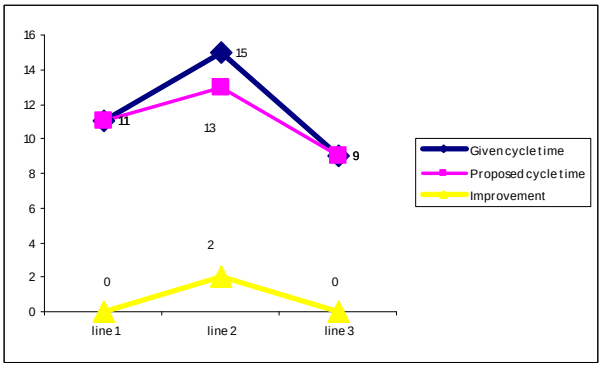

Figure 5. Jaeschke problem for three parallel lines.

4. Computational results

The performance of the proposed initial solution procedure is tested on the five well-known test problems in the ALB literature. Each problem consists of a number of tasks, task times, precedence relations and a number of cycle times. The data of line number 2 and 3 are same with number 1 by different cycle time and some added or dropped tasks. The number of stations and obtained cycle time from the procedure is compared with theoretical minimum number of stations and the independent balance of the lines for each of them. The number of stations obtained from the proposed procedure is generally less than the results of the independent balance of the lines.

In the future, more efficiently techniques should be developed to calculate and improved the cycle time in mixed model that give a better solution than the proposed procedure parallel assembly line.

5. CONCLUSIONS

In this paper, the TS technique has been applied to make exhaustive selecting and analyzing all the possibilities of a

configuration in a single iteration. This is important to fortify the mechanism search. In this study, new procedures and a mathematical model on the mixed model assembly line balancing problem with parallel lines are proposed. The studied procedure was calculated based on one numerical example. Several well-known test problems in the ALB literature are solved using the mixed model parallel assembly line procedure and the mathematical model. With this computational model, an initial optimal solution has been achieved. The obtained results from the procedure are compared with the optimal solutions, the theoretical minimum number of stations and initial cycle time calculated for each problem. Comparison of results shows that the performance of the procedure is sufficient and the proposed model provides a significant improvement in assembly line.

6. Acknowledgments

The author wishes to thank University Putra Malaysia for the financial support, department of mechanical and manufacturing to conduct the research and the anonymous referees for their praiseworthy accuracy and readiness in carrying out the reviews, and for their valuable suggestions, which led to an improvement of the quality of the presented work.

References

1. Ghosh, S. and R.J. Gagnon, *A comprehensive literature review and analysis of the design, balancing and scheduling of assembly lines.* International Journal of Production Research, 1989: p. 637-670.
2. Khan, A. and A.J. Day, *A Knowledge Based Design Methodology for manufacturing assembly lines.* Computers and Industrial Engineering, 2002. **41**: p. 441-467.
3. Nahmias, S., *Production and Operations Analysis*. second ed. 1993: Irwin, Homewood, IL
4. Gökçen, H., K. Agpak, and R. Benzer, *Balancing of parallel assembly lines.* International Journal of Production Economics, 2006. **103**(2): p. 600-609.
5. Glover, F., *Tabu Search-Part 1.* Operations Research Society of America journal on Computing, 1989. **1**(3): p. 190-206.
6. Glover, F., *Tabu search, Part II.* Operations Research Society of America journal on Computing, 1989. **2**: p. 4 - 32.
7. Chiang, W.-C. and R. Russell, *A reactive tabu search metaheuristic for the vehicle routing problem with time windows.* INFORMS Journal on Computing, 1997: p. forthcoming.
8. Chiang, W.-C. and P. Kouvelis, *An improved tabu search heuristic for solving facility layout design problems.* International Journal of Production Research 1996. **34**: p. 2565 - 2585.
9. Chiang, W.-C. and P. Kouvelis, *Simulated annealing and tabu search approaches for unidirectional flowpath design for Automated Guided Vehicle systems.* Annals of Operations Research, 1994. **50**.
10. Skorin-Kapov, J., *Tabu search applied to the quadratic assignment problem.* Operations Research Society of America journal on Computing, 1990. **2**: p. 33- 44.
11. Glover, F., *Tabu search and adaptive memory programming - advances, applications, and challenges, in: Interfaces in Computer Science and Operations Research*, in *Kluwer Academic*. 1996, Helgason and Kennington.
12. Süer, G.A., *Designing Parallel Assembly Lines.* Computers ind. Eng, 1998. **35**(3--4): p. 467-470.

ABBREVIATIONS

TS: Tabu Search

N_m : Number of task in model m^{ih}

i : Index of task,

h : Index of parallel line $H = \{1,...,|M|\}$ $h \in H$

k : Index of station in parallel line $K = \{1,...,|K|\}$ $k \in K$

m : Index of model $M = \{1,...,|M|\}$ $m \in M$

C_m : Cycle time of model m^{th}

C_h : Cycle time of line h^{th} in parallel status

t_{im} : Time of task i in model m

P_{im} : Set of all tasks which must precede task i in model m

F_{im} : Set of all tasks which must follow task i in model m

NS_h : Number of station in line h^{th} in parallel status

I_m : Set of task ID in model m^{ih}

NP_{ih} : Number of all tasks which must precede task i^{th} (predecessors) in model m^{th}

E_{ih} : Earliest station for task i in line h

L_{ih} : Latest station to which task i can be assigned in line h

$\lceil x \rceil^+$: The smallest integer greater than or equal to x

$\lceil x \rceil^-$: The greatest integer smaller than or equal to x

ST_{kh} : The sum of total tasks assigned to station k^{th} in line h^{th} on parallel status

x_{imhk} : 1 if task i from model m in line h is assigned to station k ; 0 otherwise