International Journal of Intelligent Information Systems
2015; 4(2-1): 49-58
Published online February 10, 2015 (http://www.sciencepublishinggroup.com/j/ijiis)
doi: 10.11648/j.ijiis.s.2015040201.17
ISSN: 2328-7675 (Print); ISSN: 2328-7683 (Online)

On heijunka design of assembly load balancing problem: Genetic algorithm & ameliorative procedure-combined approach

Zhi Zhuo Hou[1], Hiroshi Katayama[2], Reakook Hwang[3]

[1]Department of Industrial and Management System Engineering, Graduate School of Creative Science and Engineering, Waseda University, Tokyo, Japan
[2]Department of Industrial and Management System Engineering, Faculty of Science and Engineering, Waseda University, Tokyo, Japan
[3]Samsung Economic Research Institute, Seoul, Korea

Email address:

breathlesc@ruri.waseda.jp (Zhi Zhuo Hou)

To cite this article:

Zhi Zhuo Hou, Hiroshi Katayama, Reakook Hwang. On Heijunka Design of Assembly Load Balancing Problem: Genetic Algorithm & Ameliorative Procedure-Combined Approach. *International Journal of Intelligent Information Systems*. Special Issue: Logistics Optimization Using Evolutionary Computation Techniques. Vol. 4, No. 2-1, 2015, pp. 49-58. doi: 10.11648/j.ijiis.s.2015040201.17

Abstract: Mixed-model straight/U-shaped assembly line has been recognized as a relevant component of Just-In-Time (JIT) production line system. For this system, "Heijunka" design is also challenged as both the task assignment and the production sequence affect the workload imbalance among workstations. In this context and recognizing uncertain task time environment that is often observed in actual manufacturing scene, this research addresses the Line Balancing Problem (LBP) and the Product Sequencing Problem (PSP) jointly and proposes a mathematical model with stochastic task time which is subjected to normal distribution. The objectives of this model are to maximize line efficiency and to minimize the variation of work overload time. A Multi-objective Genetic Algorithm (MOGA) and an Ameliorative Structure of Multi-objective Genetic Algorithm (ASMOGA) with Priority-based Chromosome (PBC) are applied to solve this problem. At last, this research conducts an experimental simulation on a set of benchmark problems to verify the outperformance of the proposed algorithm.

Keywords: Mixed-Model Assembly Line, Load Balancing, Multi-Objective Genetic Algorithm, Ameliorative Procedure

1. Introduction

This research focuses on load balancing, which is an important concept of resource management in operational systems. The study aims to equalize or reduce imbalance of workload among resources in processing systems. This objective is important because fair assignment of workload on each resource enables high utilization of fixed assets. This can be achieved through task assignment procedure. Due to the trend towards a larger scale complex structure of processing systems, this concept becomes one of the most critical issues in operations and production management. The pay back of swollen fixed cost of existing configurations is highly depending on the utilization of resources in such systems. For this reason, this research highlights load balancing problem for assembly line production systems.

Assembly line is a typical flow-line production system that normally consists of sequence of workstations. These workstations are connected by material transport system such as belt conveyors. The transport system moves the products along the line at constant speed where products are evenly distributed. At each workstation, a group of trained workers repeatedly perform predetermined tasks on a partially finished product in fixed time called cycle time. These tasks are the elementary components of precedence graph which represents relation of tasks of assembly line production system. Namely, each task is subjected to sequence constraints and performs a compulsory operation to complete a final product.

Assembly line systems are categorized by two features: The first feature is layout configuration: Straight-shaped Assembly Line (SAL) and U-shaped Assembly Line (UAL). Most of the new assembly lines tend to be arranged in terms of UAL rather than SAL since UAL can be more flexible to work across both sides of the line while workers can only work on adjacent sections in case of SAL. Consequently, UAL has been viewed as an integral component of JIT production principle.

The second feature is capability of manufacturing variety: Single-Model Assembly Line (SMAL) and Mixed-Model Assembly Line (MMAL). In the early days, companies mainly employed SMAL to produce a volume of single products. However, it is difficult to satisfy the various requirements of customers. Hence MMAL was developed to meet such high level of demands.

This study deals with the Mixed-model Straight/U-shaped Assembly Line (MMS/UAL) due to the advantages mentioned above. Comparing to SAL, the optimal design of MAL needs to handle both LBP and PSP, which are closely interrelated. The optimization of LBP highly depends on PSP, which, in turn, is affected by LBP. Especially for MMS/UAL, the workload of a workstation is not only related to the assigned tasks, but also depending on the model sequence processed at this workstation. Regarding this point, load balancing problem is extended in terms of multi-decision criteria design problem. "Heijunka" criterion, one of the new multi-decision criteria, has been developed to solve this situation. "Heijunka" is Japanese word meaning level equalization. It is one of the LEAN tools used by manufacturing companies, but can be applied to any operations framework.

Two objective functions are considered in this "Heijunka" design for both LBP and PSP: 1) maximizing line efficiency and 2) minimizing the maximum of work overload ratio. In order to define the mathematical model with both two objectives, task times are assumed to be stochastic variables that subjects to normal distributions. Since there are numerous uncertain factors in real-life applications (such as workers' skill level, mentation, task complexity, machine breakdowns and non-JIT etc.), deterministic task times may be quite unrealistic even in full-automatic assembly lines. Under this stochastic assumption, both the objectives are obliged to be normally distributed. The procurement objective 1) is to maximize the mean and to minimize the variance of line efficiency variable. On the other hand, objective 2) is proposed for leveling the variance of work overload times among workstations based on the idea of "Heijunka".

In this research, an approach for this multi-criteria MMS/UAL model is developed by utilizing MOGA.

For computational complexity, line balancing and model sequencing are both known to be of the NP-hard class of combinatorial optimization problems, so the mixed-model straight/U-shaped line balancing and sequencing problem is also NP-hard. Many meta-hierarchical design procedures for this sort of problem is proposed to actualize significant improvement. Within these approaches, GA is known as a stochastic search algorithm that mimics the process of natural selections and genetic mutations. The implementation of genetic operators makes GA very effective in performing global search, while most of conventional heuristic methods usually hard to overcome local search. Since the task precedence graph among different models are actually integrated, a PBC is advanced to encode the solution of this problem into a chromosome efficiently. In addition, to improve the performance of MOGA, an ASMOGA is presented. In order to optimize the outcomes of GA, the evaluation method is proposed as the weight mapping function of line efficiency and work overload time, and the selection operator is considered as roulette wheel selection procedure.

At last, experimental simulation on the objective problems are conducted by comparing the performance of MOGA and ASMOGA. Furthermore, the attained performance of straight lines and the performance of U-lines are compared to confirm the improvement.

The structure of this dissertation is organized as following: section 2 describes the assembly load balancing problem by constructing the mathematical model of the problem and illustrates the procedure of task assignment; section 3 proposes a genetic algorithm for solving this problem; section 4 provides the experimental results on a set of benchmark problems; section 5 concludes this paper.

2. Assembly Line Load Balancing Problem

As mentioned before, two objectives of maximizing line efficiency and minimizing work overload time are proposed for both LBP and PSP in MMU/SAL system. Since calculating the value of both objectives highly relies on line organization, an elaborate task assignment procedure for both straight line and U-shaped line is necessary. Therefore, Chapter 2 proposes a detailed mathematical model and a proper approach of task assignment.

2.1. Mathematical Model

2.1.1. Notations

1) Indices:

i task index, $i = 1, \dots, I$

j workstation index, $j = 1, \dots, J$

m production model index, $m = 1, \dots, M$

k chromosome index, $k = 1, \dots, popSize$

r_{ik} priority index of task i at chromosome k, $i = 1, \dots, I, k = 1, \dots, popSize$

2) Parameters:

c cycle time

θ maximum load time; θ =0.8c

q_m production quantity of product model m;

I_j vector that represents the task index of workstation j; $I_j = \{\text{vector of } i \,|x_{ij} = 1\}$

I_m vector that represents the task index of model m; $I_m = \{\text{vector of } i \,|z_{im} = 1\}$

t_i operation time of task i; $t_i \sim N(\mu_i, \sigma_i^2)$

T_j workload of workstation j; $T_j = \sum_{i \in I_j} t_i$, $T_j \sim N(\mu_T, \sigma_T^2)$

μ_T mean of T_j; $\mu_T = \sum_{i \in I_j} \mu_i$

σ_T^2 variance of T_j; $\sigma_i^2 = \sum_{i \in I_j} \sigma_i^2$

$f_{T_j}(x)$ probability density function of T_j; $f_{T_j}(x) =$

$\frac{1}{\sqrt{2\pi}\sigma_T}\exp\left(-\frac{(x-\mu_T)^2}{2{\sigma_T}^2}\right)$

$F_{T_j}(x)$ cumulative distribution function of T_j; $F_{T_j}(x) = P(T_j \le x) = \int_{-\infty}^{x} f_{T_j}(x)dx$

η line efficiency; $\eta = \frac{1}{J\times c}\sum_{j=1}^{J} T_j = \frac{1}{J\times c}\sum_{i=1}^{I} t_i$

μ_η mean of η; $\mu_\eta = \frac{1}{J\times c}\sum_{i=1}^{I}\mu_i$

σ_η^2 variance of η; $\sigma_\eta^2 = \frac{1}{(J\times c)^2}\sum_{i=1}^{I}\sigma_i^2$

P_j work overload probability of workstation j; $P_j = P\{T_j > c\} = 1 - P\{T_j \le c\} = 1 - F_{T_j}(c)$

P_{Max} maximum of P_j; $P_{Max} = max\{P_j | j \in J\}$

V_k fitness value of chromosome k, $V_k = \mu_\eta - \sigma_\eta^2 - P_{Max}$

$pre(i)$ set of predecessors of task i

$suc(i)$ set of successors of task i

3) Decision Variables:

$$x_{ij} = \begin{cases} 1, if\ task\ i\ is\ assigned\ to\ workstation\ j \\ 0, otherwise \end{cases}$$

$$z_{im} = \begin{cases} 1, if\ task\ i\ is\ an\ element\ of\ model\ m \\ 0, otherwise \end{cases}$$

2.1.2. Formulations

1) Objective functions:

$$\text{Max}\ \ \mu_\eta \qquad (1)$$

<Maximize mean of line efficiency>

$$\text{Min}\ \ \sigma_\eta^2 \qquad (2)$$

<Minimize variance of line efficiency>

$$\text{Min}\ \ P_{Max} \qquad (3)$$

<Minimize the maximum probability of work overload times>

2) Subject to:

$$x_{ij} = 0\ or\ 1\ \ i = 1, \dots, I; j = 1, \dots J \qquad (4)$$

$$\sum_{j=1}^{J} x_{ij} = 1\ i = 1, \dots, I$$

<Each task must be assigned to one and only one workstation>

$$z_{im} = 0 \text{ or } 1\ i = 1, \dots, I; m = 1, \dots M \qquad (5)$$

$$\sum_{m=1}^{M} z_{im} = 1\ i = 1, \dots, I$$

<Each task must belong to one and only one product model>

$$\sum_{j=1}^{J} j(x_{pj} + x_{ij} \le 0) \quad \forall p \in pre(i) \quad (6: \text{Straight Line})$$

<All the predecessors for task i are assigned to the same or to an earlier workstation>

$$\sum_{j=1}^{J} j(x_{pj} + x_{ij} \le 0) \text{ or } \sum_{j=1}^{J} j(x_{sj} + x_{ij} \le 0)$$

$$\forall p \in pre(i), \forall s \in suc(i) \quad (7: \text{U-shaped Line})$$

<All the predecessors or all successors for task i are assigned to the same or an earlier workstation>

2.2. Task Assignment

The mixed-model assembly line studied in this paper consists of a set of workstations which are arranged along a straight/U-shaped conveyor that automatically moves at a constant speed. Different products with similar characteristics are launched onto the conveyor according to the product sequence at a fixed rate. Tasks are moved through the workstations sequentially and processed into finished products after leaving the last workstation. During the period of manufacturing, there are several different products processed at different workstations. After a lapse of time called cycle time, each product enters the next workstation and the worker returns back to the upstream boundary of the workstation to manufacture the next product.

2.2.1. Problem Description

For load balancing, it concerns the assignment of a set of tasks to different workstations regarding some special objectives, such as minimizing the number of workstations for a given cycle time (Type 1), minimizing the cycle time for a given number of workstations (Type 2), maximizing the line efficiency by minimizing both the cycle time and the number of workstations (Type E), or optimizing a certain objective for a given combination of cycle time and number of workstations (Type F). Since the objective of this paper is to maximize line efficiency and minimize workload variation simultaneously with given task properties and workstation properties, the type of this problem will be Type F. Examples for both straight and U-shaped assembly line load balancing problem are used to demonstrate the assignment.

(1) Line properties: quantity of workstations J = 4; cycle time c = 15; maximum workload θ = 0.8c = 12.

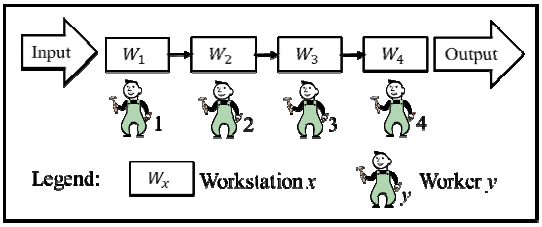

Figure 1. Straight Assembly Line

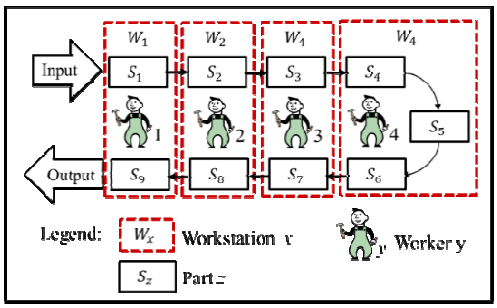

Figure 2. U-shaped Assembly Line

As shown in Figure 1 and Figure 2, the layout of U-shaped lines is significantly different from straight line. U-shaped lines allow the forward and backward task assignment. For instance, the first task and the last task must be placed at the last workstation, which is impossible for a straight line. Besides, there are two kinds of workstation in U-lines: 1) Crossover workstation, where tasks are able to be allocated to both the front and back part of workstations and operators can work at both sides. 2) Regular workstation, where the difference between front and back part of the workstations does not exist. As for the U-shaped line in Figure 2, workstation 1, 2, 3 are crossover workstations while workstation 4 is regular workstation. Furthermore, not all the parts of a workstation need to be assigned with tasks. The parts assignment depends on the product sequence as shown in Figure 3 illustrated in the following head.

(2) Task properties: task quantity I = 11; task time $t_i \sim N(\mu_i, \sigma_i^2)$; one PBC r_{ik} $(i = 1, \ldots, I, k = 1)$; task precedence diagram; quantity of products M = 3.

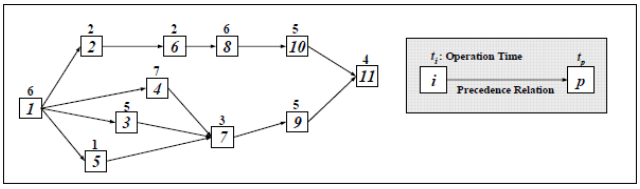

Figure 3. *Task Precedence Diagram*

Because of the technical requirements, each product has its own precedence relationships among tasks called precedence diagram. Generally, these diagrams of different products can be combined into a single precedence diagram. For instance, Figure 3 illustrates the precedence diagrams of 3 kinds of products, in which each node represents a task and each arrow connecting two different nodes indicates their precedence relationship.

Since the difference between straight line and U-shaped line is not negligible, the sequence of assignment for both lines is quite different: For straight line, task 1 must be assigned to the first workstation and task 11 must be assigned to the last workstation, the sequence of searching available tasks starts with task 1 and ends with task 11(From first to last); For U-shaped line, task 1 and task 11 must be assigned to the first workstation, the sequence of searching available tasks starts with task 1 and task 11 (From two edges to middle).

Table 1. *Task Time*

i	1	2	3	4	5	6	7	8	9	10	11	Total
μ_i	6	2	5	7	1	2	3	6	5	5	4	46
σ_i^2	2	1	3	4	2	3	2	5	1	2	3	2

Different from deterministic mathematical model in other researches, task times are assumed to be stochastic in this literature. As described in Table 1, time of task 1 is subject to normal distribution of N (6, 2). Since determining the workload of workstation with random task times is extremely hard, this paper presumes that the mean of task time is the actual task time during the procedure of assignment.

Table 2. *Task Priority & Product ID (A Chromosome)*

i	1	2	3	4	5	6	7	8	9	10	11
r_{i1}	7	2	2	3	10	2	4	2	5	3	8	3	9	1	6	3	11	1	1	2	3	1

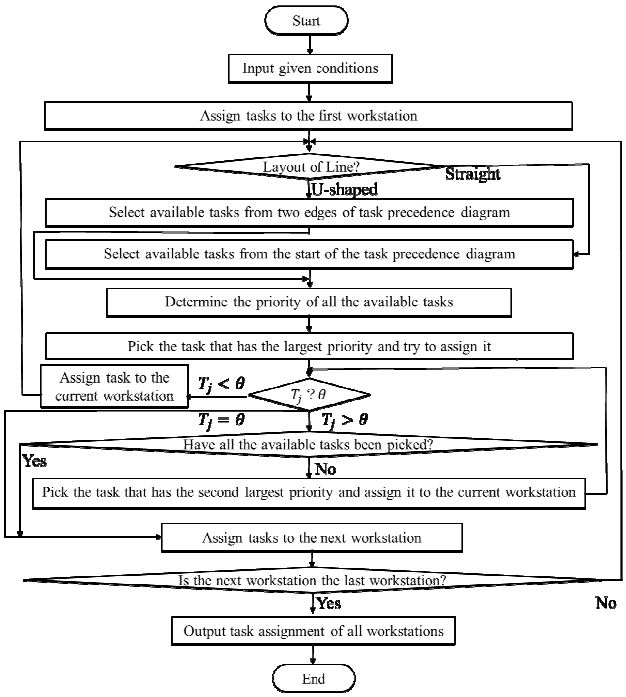

Figure 4. *Regular Structure for Task Assignment*

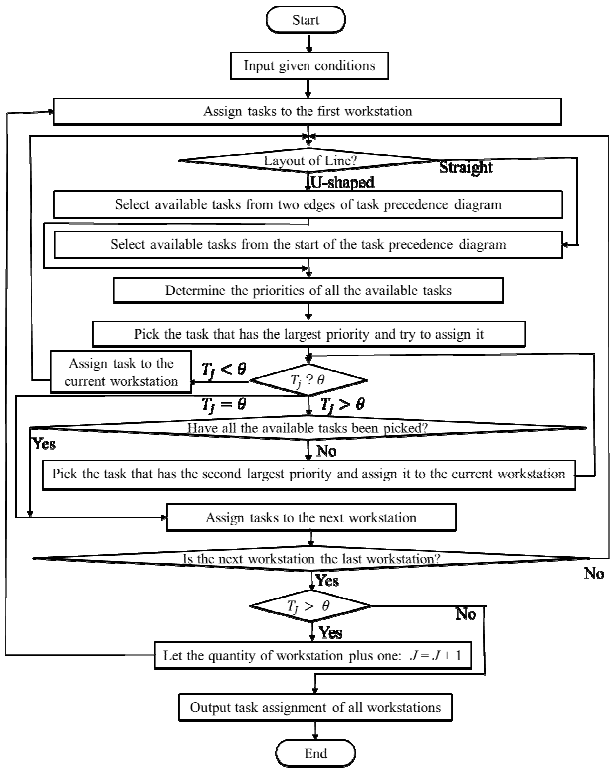

Figure 5. *Ameliorative Structure for Task Assignment*

Each task has a priority value which is randomly generated as shown in Table 2, which represents a chromosome of GA. These priorities are different from each other because the same priorities probably lead to a tie of task assignment.

Besides, product sequence is determined by the priority number of tasks: When the conditions of priorities (r_{i1} = {7,2,10,5,8,9,11,6,1,3}) and quantity of products (M = 3) are given, the ID of product is able to be deducted by calculating the function of $\{mod(r_{ik}, M) + 1\}$. For example, $\{r_{11} = 7, M = 3\} \rightarrow$ task 1 represents product 2 (7 mod 3 + 1 = 2). In this analogy, the product sequence of this chromosome is {2,3,2,2,3,3,1,3,1,2,1} and the quantities for each product are $\{q_1 = 3,\ q_2 = 4,\ q_3 = 4\}$.

2.2.2. Assignment Procedure

Two kinds of assignment procedure are proposed as shown in Figure 4 and Figure 5: 1) Regular structure for MOGA and 2) Hierarchical ameliorative structure for ASMOGA. This ameliorative structure is originally developed by Katayama.

The assignment sequence for workstations is from the first workstation to the last workstation. For straight line, the first task is obliged to be assigned to the first workstation while both the first and the last task are obliged to be assigned at the first workstation of U-shaped line. After assigning first task to first workstation (or first and last task to first workstation), available tasks are selected to be candidates according to the task precedence diagram. Task with the largest priority is about to be assigned before other available tasks. This select-assign process does not stop until the load of this workstation exceeds the maximum workload θ. When this excess occurs, the task that has the second largest priority is selected to replace the task that has the most priority. The rest assignment of workstations can be done in the same manner. At last, the workload of final workstation has an inevitable probability of exceeding the maximum workload. To cope with this excess, this ameliorative structure lets $J = J + 1$ (if $T_J > \theta$), then reassigns the tasks until $T_J \leq \theta$. And this procedure is also the difference between the regular structure and the ameliorative structure.

The results of assignment for the examples of Chapter 2.2.1 are proposed as shown in Figure 6 and Figure 7. This assignment is based on both regular structure and ameliorative structure.

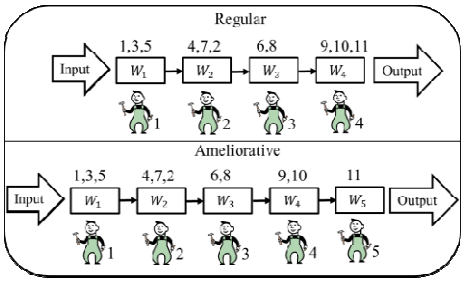

Figure 6. *Regular & Ameliorative Structure for Straight Line Assignment*

For the example of straight line, the assignment results for regular structure are different from those for ameliorative structure. The workload of last workstation in the regular assignment actually exceeds the maximum workload time. Hence, the amount of workstation is about to be 5 according to the ameliorative assignment structure.

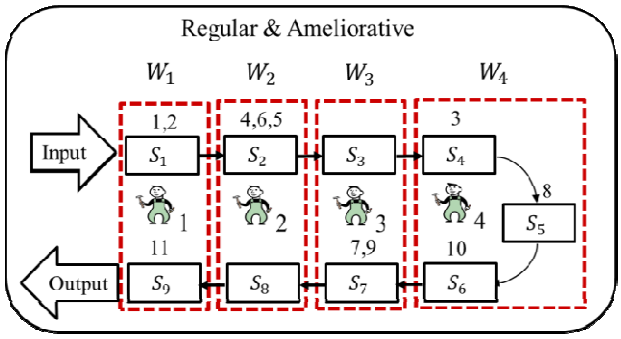

Figure 7. *Regular & Ameliorative Structure for U-shaped Line Assignment*

For the example of U-shaped line, the assignment results for both structures are same. Tasks are assigned to all the parts on workstations 1 and 4 while workstation 2 and 3 do not have all the parts assigned.

3. Genetic Algorithm Design

3.1. The Procedure of Genetic Algorithm

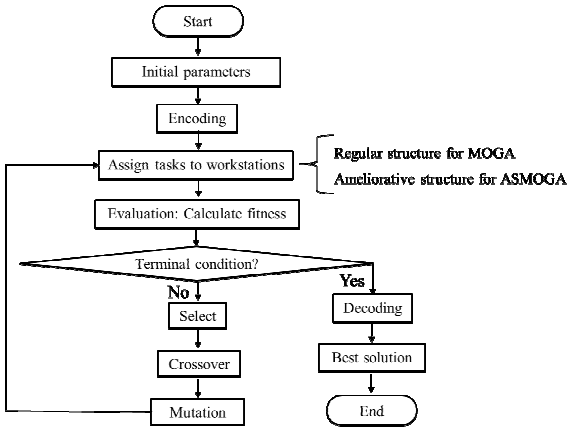

Figure 8. *Process of Genetic Algorithm*

The multi-objective problem formalized in the previous section cannot be easily solved by traditional mathematical techniques. This forces researchers to employ faster and more effective algorithms such as genetic algorithm. GA, which is differing from conventional search techniques, starts with an initial set of random solutions called "population". Each individual in the population is called a "chromosome", representing a solution to the problem at hand. A chromosome is a string of symbols; it is usually, but not necessarily, a binary bit string. The chromosomes "evolve" through successive iterations, called "generations". During each generation, the chromosomes are evaluated, using the measures of "fitness". To create next generation, new chromosomes called "offspring" are formed by selecting current chromosomes which are called "parents" according to the fitness values. Chromosomes have higher fitness are more likely to be selected. Consequently, new selected offspring are

reformed by either merging two parents using a "crossover" operator or modifying a current parent using a "mutation" operator under a given probability. After several of generations, the algorithm converges to the best chromosome, which hopefully represents the optimum or sub-optimum solution of the problem.

This section proposes a MOGA with PBC and an ASMOGA with PBC to deal with the mixed-model straight/U-shaped assembly load balancing problem. As shown in Figure 6, the process for both algorithms are same, except that the task assignment is distinguished from regular structure and ameliorative structure.

3.2. Encoding and Decoding

3.2.1. Encoding

Encode the parameters of problem into chromosomes is a key issue for the genetic algorithm. Gen *et al.* (1997) developed priority-based GA. This method is proposed to handle the difficulty of how to produce an efficient encoding that satisfies all the constraints of actual world. Recall that a gene contains two kinds of information: 1) The locus that represents the position of a gene located within the structure of chromosome; 2) The allele that represents the value taken by the gene. In this research, the initial generation are formed by randomly reproducing chromosomes which have both locus and allele. The locus is used to denote the task ID, and the allele is used to denote the priority and product ID, as shown in Table 3. Table 3 states a generation consists of 10 chromosomes. Within each chromosome, there are 11 priority values for each task. Furthermore, the product ID is conceived by the method mentioned at Chapter 2.2.1.

Table 3. *Population*

i	**1**		**2**		**3**		**4**		**5**		**6**		**7**		**8**		**9**		**10**		**11**	
r_{i1}	7	2	2	3	10	2	4	2	5	3	8	3	9	1	11	3	6	1	1	2	3	1
r_{i2}	2	3	7	2	11	3	5	3	4	2	9	1	8	3	10	2	1	2	3	1	6	1
r_{i3}	6	1	5	3	4	2	2	3	1	2	3	1	10	2	11	3	8	3	7	2	9	1
r_{i4}	10	2	7	2	2	3	4	2	5	3	11	3	9	1	1	2	10	1	8	3	3	1
r_{i5}	3	1	9	1	4	2	10	2	5	3	8	3	7	2	4	3	6	1	5	3	1	2
r_{i6}	5	3	11	3	10	2	4	2	3	1	2	3	1	2	9	1	8	3	7	2	6	1
r_{i7}	4	2	6	1	10	2	2	3	5	3	3	1	9	1	7	2	11	3	8	3	1	2
r_{i8}	9	1	2	3	5	3	7	2	3	1	6	1	1	2	11	3	4	2	10	2	8	3
r_{i9}	1	2	8	3	3	1	10	2	6	1	7	2	5	3	4	2	2	3	9	1	11	3
r_{i10}	8	3	2	3	10	2	11	3	9	1	3	1	5	3	4	2	1	2	6	1	7	2

3.2.2. Task Assignment

In address to calculate the fitness value of chromosome, a known task assignment is necessary. Like mentioned at Chapter 2.2.2, assignment results for each chromosome of each generation is able to be achieved by the assignment procedure.

3.2.3. Decoding

Decoding, a reverse process of encoding, is used to convert chromosomes into understandable solutions. After the terminal condition is satisfied, a chromosome with the best fitness is decoded into the best solution. The decoding method of this paper is converting the priority-based chromosome into the information of actual task assignment. This information consists of the status of workstations which are supposed to have the optimal line balance and product sequence. After obtaining the information of task assignment among workstations, the parameters of mathematical model defined in the Chapter 2.1 can be easily figured out. At last, the problems of line balancing and product sequencing are considered to be solved by outputting the final solution

3.3. Evaluation

Fitness evaluation is used to calculate and check the value of the objectives. In this case, the study considers the standards of evaluation function as all the three objective functions simultaneously.

The first fitness standard (f_1) maximizes the mean of line efficiency:

$$f_1(\mu_\eta) = \mu_\eta = \frac{1}{J\times c}\sum_{j=1}^{J} T_j = \frac{1}{J\times c}\sum_{i=1}^{I} t_i \quad (8)$$

The second fitness standard (f_2) is to minimize the mean of line efficiency:

$$f_1(\sigma_\eta^2) = \sigma_\eta^2 = \frac{1}{(J\times c)^2}\sum_{i=1}^{I} \sigma_i^2 \quad (9)$$

The third fitness standard (f_3) is to minimize the maximum value of work overload probability of workstation *j*:

$$f_3(P_{Max}) = P_{Max} = max\left\{P_j = 1 - \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}\sigma_T} \exp\left(-\frac{(c-\mu_T)^2}{2\sigma^2} dc \,|j \in J\right\}\right. \quad (10)$$

The evaluation function consists of the three factors:

$$F(V_k) = V_k = f_1(\mu_\eta) - f_2(\sigma_\eta^2) - f_3(P_{Max}) = \mu_\eta - \sigma_\eta^2 - P_{Max} \quad (11)$$

The above $F(V_k)$ is the objective function combining μ_η, σ_η^2 and P_{Max} of the k-th chromosome. Since $f_1(\mu_\eta)$ is a maximization function, $f_1(\sigma_\eta^2)$ and $f_3(P_{Max})$ are both minimization functions, the evaluation function converts these functions into maximization functions. By this analogy, all the fitness of individual chromosome can be evaluated.

3.4. Selection

When the search of GA proceeds, the population undergoes evolutional change according to fitness, and relatively good chromosomes are survived while relatively bad solutions are died in order that the offspring composed of good solution are reproduced for offspring. To distinguish solutions, a principle of Darwinian natural selection is necessary. Generally, selection provides the driving force to the evolution.

This paper uses roulette wheel selection, a method to reproduce a new generation that is proportional to the fitness of each individual. The basic idea of this selection is to determine the cumulative selection probabilities for each chromosome by fitness. A roulette wheel is formed by these probabilities. After spinning the wheel population-sized times, a same-sized generation is reproduced. The procedure of selection can be described as five steps:

Step 1: Evaluate the fitness value of the chromosome v_k of current population

$$F(V_k) = V_k\ ,\ k = 1, \dots, popSize \quad (12)$$

Step 2: Calculate the total fitness for the current population:

$$Total = \sum_{k=1}^{popSize} F(V_k),\ k = 1, \dots, popSize \quad (13)$$

Step 3: Calculate the selection probability p_k for chromosome v_k:

$$p_k = \frac{F(V_k)}{Total},\ k = 1, \dots, popSize \quad (14)$$

Step 4: Calculate the cumulative selection probability q_k for chromosome v_k:

$$q_k = \sum_{n=1}^{k} p_k, n = 1, \dots, k \quad (15)$$

Step 5: Generate a random number $r \in [0,1]$, and select chromosomes

1) if $r \le q_1$, then select the first chromosome v_1; $k = 1, \dots, popSize$.

2) else, select the kth chromosome v_k when $q_{k-1} < r \le q_k$; $k = 1, \dots, popSize$.

3.5. Crossover Operator

A cross over operator called Weight Mapping Crossover (WMX) is proposed to diversify the chromosome as shown in Figure 6. This crossover operator combines the features of two parent whose corresponding random value is less than crossover rate P_c.

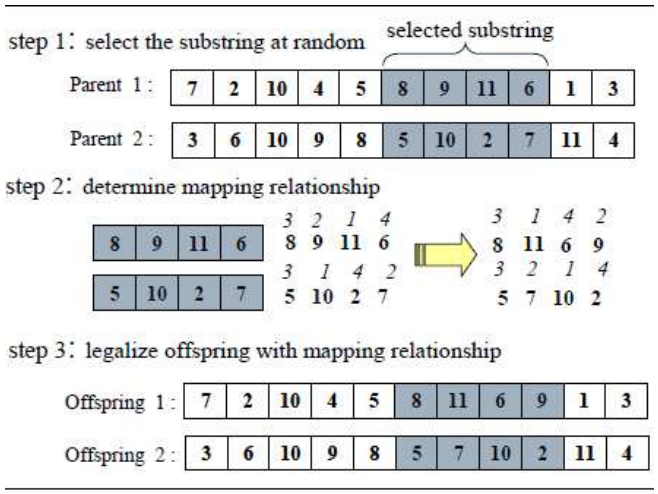

Figure 9. *Two Point-based Weight Mapping Crossover Operator*

3.4. Mutation Operator

Swap Mutation used in this algorithm is described as Figure 7. Two positions are randomly selected and their contents are swapped. This mutation operator arbitrarily alters two components of a selected chromosome and increases the variability of the population. Each chromosome undergoes a random change when the corresponding random rate is less than mutation rate P_m.

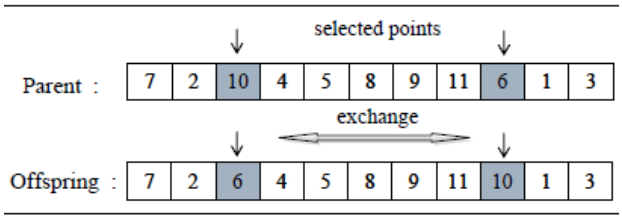

Figure 10. *Swap Mutation Operator*

4. Numerical Experiment

In this numerical experiment, the test problems are described using the well-known benchmark problems of Thomopoulos, Kim and Arcus to compare MOGA with ASMOGA. Besides, a comparison between straight lines and U-lines is conducted to confirm the outperformance of U-lines. The following parameters are applied to GA throughout the simulations:

Population size: $popSize = 10$
Maximum generation: $maxGen = 100$
Crossover probability: $P_c = 0.25$
Mutation probability: $P_m = 0.25$

Terminating condition: Reach the last generation defined at start

The algorithm is coded in C# and the experiments are implemented on a Core i7 3.00GHz PC. To illustrate how ASMOGA of U-lines improves the performance of load balancing by using an ameliorative structure on a set of

numerical examples, two tables and a figure are listed.

As shown in Table 4 and Table 5, the workstations of line are supposed to be increased according to whether the workload of last workstation is bigger than the cycle time. Also, the product sequence for each problem is obtained by the function mentioned at Chapter 2.2.1. At the last column of these tables, the quantities for individual product is enumerated. Furthermore, Figure 8 illustrates the best results for both MOGA and ASMOGA (Jackson 11 tasks). Based on these results, the theory that the proposed ASMOGA can improve the load balancing can be confirmed.

Next, another comparison is conducted to describe the advantages of U-shaped line as shown in Table 5. U-shaped line is able to obtain improvements at line efficiency and workload variance concurrently.

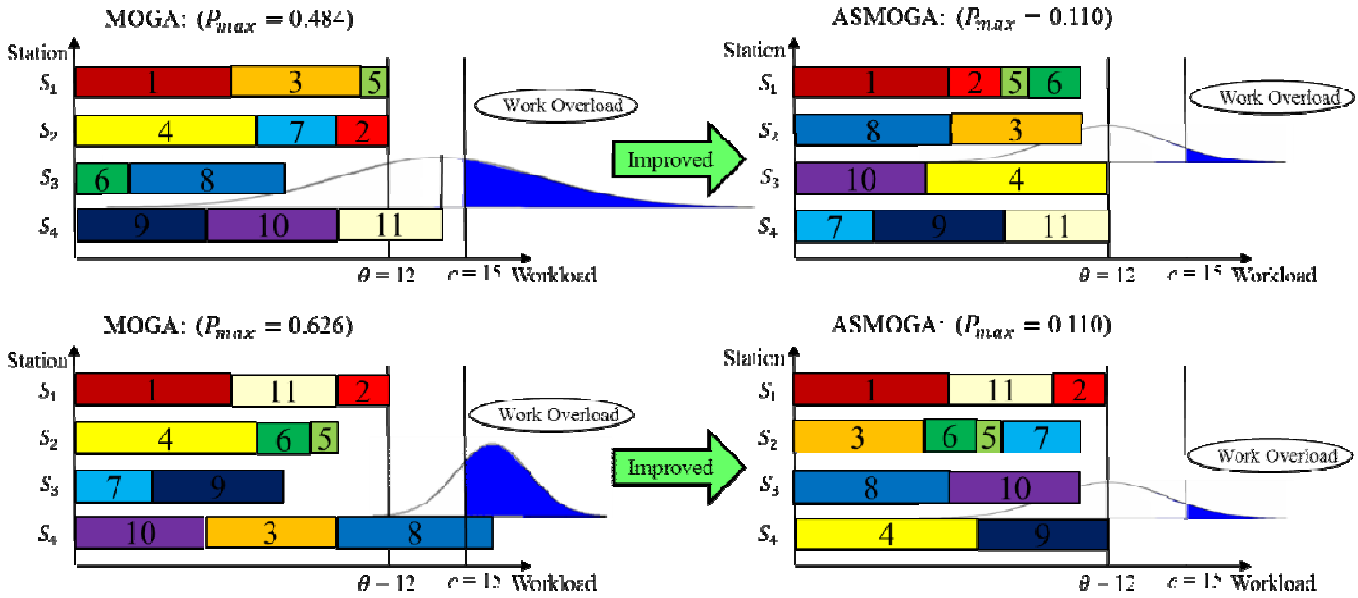

Figure 11. *Best Results from the MOGA and ASMOGA*

Table 4. *Result of Straight Line*

Problem	I	M	c	J	J'	MOGA			ASMOGA			$q_1, q_2, \ldots, q_M$
						μ_η	σ_η^2	P_{Max}	μ_η	σ_η^2	P_{Max}	
			3	3	4	73.2%	0.022	27.1%	76.6%	0.013	11.0%	6,7,6
Thomopoulos	19	3	3	3	4	69.5%	0.090	18.8%	70.7%	0.009	6.5%	6,7,6
			4	3	4	67.9%	0.108	23.5%	72.0%	0.010	12.8%	6,7,6
			138	6	7	70.8%	0.054	26.2%	75.0%	0.031	18.9%	15,16,15,15
Kim	61	4	205	6	7	73.3%	0.049	30.1%	78.2%	0.029	23.1&	15,16,15,15
			324	12	14	68.8%	0.102	27.6%	77.7%	0.047	14.5%	15,16,15,15
			27	12	15	66.6%	0.129	42.5%	71.7%	0.103	28.2%	22,23,22,22,22
Arcus	111	5	33	15	18	69.7%	0.113	57.3%	70.6%	0.097	25.7%	22,23,22,22,22
			34	27	33	71.5%	0.098	66.7%	73.4%	0.084	20.5%	22,23,22,22,22

Table 5. *Results of U-shaped Line*

Problem	I	M	c	J	J'	MOGA			ASMOGA			$q_1, q_2, \ldots, q_M$
						μ_η	σ_η^2	P_{Max}	μ_η	σ_η^2	P_{Max}	
			3	3	4	74.5%	0.017	11.0%	79.0%	0.011	7.8%	6,7,6
Thomopoulos	19	3	3	3	4	73.8%	0.018	6.5%	75.0%	0.010	8.4%	6,7,6
			4	3	4	77,8%	0.021	12.8%	82.3%	0.015	10.1%	6,7,6
			138	6	7	76.9%	0.047	18.9%	81.1%	0.028	17.7%	15,16,15,15
Kim	61	4	205	6	7	73.7%	0.036	23.1&	82.5%	0.023	25.9&	15,16,15,15
			324	12	13	81.1%	0.047	14.5%	85.5%	0.035	16.4%	15,16,15,15
			27	12	16	75.3%	0.114	28.2%	77.6%	0.087	36.3%	22,23,22,22,22
Arcus	111	5	33	15	19	66.6%	0.086	25.7%	69.6%	0.093	24.7%	22,23,22,22,22
			34	27	35	79.7%	0.102	20.5%	81.3%	0.075	18.8%	22,23,22,22,22

Table 6. *Straight Line vs U-shaped Line from ASMOGA*

Problem	I	M	c	J	J'	Straight			U-shaped			$q_1, q_2, \cdots, q_M$
						μ_η	σ_η^2	P_{Max}	μ_η	σ_η^2	P_{Max}	
			3	3	4	76.6%	0.013	11.0%	79.0%	0.011	7.8%	6,7,6
Thomopoulos	19	3	3	3	4	70.7%	0.009	6.5%	75.0%	0.010	8.4%	6,7,6
			4	3	4	72.0%	0.010	12.8%	82.3%	0.015	10.1%	6,7,6
			138	6	7	75.0%	0.031	18.9%	81.1%	0.028	17.7%	15,16,15,15
Kim	61	4	205	6	7	78.2%	0.029	23.1&	82.5%	0.023	25.9&	15,16,15,15
			324	12	13	77.7%	0.047	14.5%	85.5%	0.035	16.4%	15,16,15,15
			27	12	16	71.7%	0.103	28.2%	77.6%	0.087	36.3%	22,23,22,22,22
Arcus	111	5	33	15	19	70.6%	0.097	25.7%	69.6%	0.093	24.7%	22,23,22,22,22
			34	27	35	73.4%	0.084	20.5%	81.3%	0.075	18.8%	22,23,22,22,22

5. Conclusion

This paper studies the load balancing of mixed-model straight/U-shaped assembly line problem. This problem is one of the most classic researches of production management system. The objective of the problem is to maximize the line efficiency and to minimize the work overload imbalance simultaneously for a given combination of cycle time and number of workstations. To solve this problem, the variable notations and the mathematical formulations are employed to obtain a model of problem. Then a multi- objective genetic algorithm with priority-based chromosome is developed and structured in terms of a hierarchical ameliorative design to find a near-optimal solution. To evaluate the performance of this algorithm, two sets of experiments are conducted respectively. One is comparison of MOGA with ASMOGA, and another is the comparison of straight line with U-line. As the result of numerical experiments, it was revealed that the proposed approaches can solve these multi-objective problems more quickly than conventional heuristic methods.

For future researches, I wish this proposed method could make a certain contribution toward more research areas such as logistics, SCM and further scheduling problems.

Acknowledgements

This research was supported by my research advisor Professor Hiroshi Katayama and Professor Katayama's former PhD student Dr. Reakook Hwang. I would like to acknowledge and to thank them for giving me the opportunity to research on assembly line projects and for their constant help and support during the whole realization of this paper.

References

[1] Chen R., Lu K. and Yu S., "A hybrid genetic algorithm approach on multi-objective of assembly planning problem [J]", *Engineering Applications of Artificial Intelligence*, Vol. 15 No. 5, pp. 447-457, 2002.

[2] Gen, M. and Cheng, R., *Genetic algorithm & engineering design*, New York: Wiley, 1997.

[3] Hwang, R. K. and Katayama, H., "A Multi-decision Genetic Approach for Workload Balancing with Mixed-Model U-shaped Assembly Line Systems", *International Journal of Production Research*, Vol. 47, No. 14, pp. 3797-3822, 2009.

[4] Hwang, R. K., *A Study on Load Balancing Problem Solving by Genetic Algorithm -Case Analyses on Assembly Line and Multiprocessor Systems-*, Tokyo: Waseda University Doctoral Dissertation, 2009.

[5] Hwang, R. K. and Katayama, H., "Integrated procedure of balancing and sequencing for mixed-model assembly lines: a multi-objective evolutionary approach", *International Journal of Production Research*, Vol. 47, No. 21, pp. 6417-6441, 2010..

[6] Hackman, S. T., Magazine, M. J. and Wee, T. S., "Fast, Effective Algorithms for Simple Assembly Line Balancing Problems", *Operations Research*, Vol. 37 No. 6, pp. 916-924, 1996.

[7] Kara. Y., Ozcan, U., and Peker, A., "Balancing and Sequencing mixed-model just-in-time U-lines with multiple objectvies", *The International Journal of Advanced Manufacturing Technology*, Vol. 32, No. 11-12, pp.1218-1231., 2007.

[8] Katayama, H., "An integrated management procedure of multi-item mixed-line production system-its hierarchical structure and performance evaluation", *International Journal of Production Research*, Vol. 36, No. 10, pp. 2633–2651, 1998.

[9] Kim, Y. K., Kim, J. Y. and Kim, Y., "An endosymbiotic evolutionary algorithm for the integration of balancing and sequencing in mixed-model U-lines", *European Journal of Operational Research*, Vol. 168, No. 3, pp. 838-852, 2006.

[10] Miltenburg, J., "Balancing and scheduling mixed-model U-shaped production lines", *International Journal of Flexible Manufacturing Systems*, Vol. 14, No. 2, pp. 119–151, 2002.

[11] Scholl, A., *Balancing and Sequencing of assembly lines 2nd ed.*, Hedelberg, Germany: Physisca Press, 1999.

[12] Ponnambalam S. G., Aravindan P. and Naidu G. M., "Multi-objective genetic algorithm for solving assembly line balancing problem", *International Journal of Advanced Manufacturing Technology*, Vol. 16.No. 5, pp. 341-352, 2000.