Circuits and Systems, 2016, 7, 907-931
Published Online May 2016 in SciRes. http://www.scirp.org/journal/cs
http://dx.doi.org/10.4236/cs.2016.76078

Solving Level Scheduling in Mixed Model Assembly Line by Simulated Annealing Method

Senthilkumar Ramalingam[1], Ramkumar Anna Subramanian[2]

[1]Maharani Polytechnic College, Dharapuram, India
[2]Maharaja Prithvi Engineering College, Coimbatore, India
Email: senthilkumarbpa@gmail.com

Received 23 March 2016; accepted 13 May 2016; published 18 May 2016

Abstract

This paper presents an application of the simulated annealing algorithm to solve level schedules in mixed model assembly line. Solving production sequences with both number of setups and material usage rates to the minimum rate will optimize the level schedule. Miltenburg algorithm (1989) is first used to get seed sequence to optimize further. For this the utility time of the line and setup time requirement on each station is considered. This seed sequence is optimized by simulated annealing. This investigation helps to understand the importance of utility in the assembly line. Up to 15 product sequences are taken and constructed by using randomizing method and find the objective function value for this. For a sequence optimization, a meta-heuristic seems much more promising to guide the search into feasible regions of the solution space. Simulated annealing is a stochastic local search meta-heuristic, which bases the acceptance of a modified neighboring solution on a probabilistic scheme inspired by thermal processes for obtaining low-energy states in heat baths. Experimental results show that the simulated annealing approach is favorable and competitive compared to Miltenburg's constructive algorithm for the problems set considered. It is proposed to found 16,985 solutions, the time taken for computation is 23.47 to 130.35, and the simulated annealing improves 49.33% than Miltenberg.

Keywords

Mixed Model Assembly Line, Level Schedule, Sequence, Just-in-Time Manufacturing, Simulated Annealing

1. Introduction

Mixed Model Assembly Line (MMAL) sequencing is a problem of determining a sequence of the product mod-

How to cite this paper: Ramalingam, S. and Subramanian, R.A. (2016) Solving Level Scheduling in Mixed Model Assembly Line by Simulated Annealing Method. *Circuits and Systems*, **7**, 907-931. http://dx.doi.org/10.4236/cs.2016.76078

els whereby a major emphasis is placed on maximizing the line utilization. MMAL is a type of production line, in which variety of product models is assembled and many industries use MMALs for diversified small-lot productions. In mixed model assembly, industrial scheduling is the most important concept, *i.e.*, correct sequence is necessary of effective utilization of assembly lines. It addresses two key problems: i) Level scheduling and ii) Line balancing problem. The balancing problem is the reasonable distribution of the operation units and it is a long time decision problem. The scheduling is short time decision making problem. Level scheduling problem in a mixed model assembly line is a famous approach for resulting short term sequence to facilitate a just-in-time supply. In the past, research on sequencing problem in MMAL are considered with the objectives of optimizing minimum cycle time, constant rate usage of parts, minimum variable parts usage, minimum number of workstation and minimum total work over load time by using various methods like genetic algorithm, particle swarm optimization, mathematical models and many heuristic procedures. In a Just-in-Time (JIT) production system, only the necessary products at the necessary time, in the necessary quantity are manufactured and stock on hand is held to minimum. Assembly is the process of collecting the various parts from raw material and putting together to form a product. The assembly line is classified into single model assembly, multi model assembly, mixed model assembly. In the single, multi and mixed models, lines are shown in **Figure 1**.

The single model assembly line has been used to produce single model only. In this production line, large quantity of products can be produced without changing the setup. In multi model assembly line, similar products are manufactured in one or more assembly lines. In mixed model assembly line, two or more products are produced in the same assembly line.

In [1] it shows that the production sequence of introducing variety of product models to the mixed model assembly line is different due to different objective goal of controlling the line. The problem of scheduling the sequence of products to be assembled by a line is:

Leveling the load on each station on the line,

Keeping constant rate of usage of every parts used by the line.

The customers need the different models as per their necessity. According to the customers demand the need to produce the different model is must. In mixed model assembly line, the different models are produced as similar product characteristics are assembled. The two main objectives are line balancing and scheduling. The line balancing is leveling the work load to each work station is uniform. The operation times at each station are not the same. The certain work station operation time is exceeding to cycle time. The assembly line is adjusted by this cycle time without line stoppage. However, the successive scheduling creates delays and it leads to line stoppage, so it is essential to minimize the line stoppage.

The scheduling is much more important than line balancing. The quantity of each part used by the mixed model assembly line per unit time should be kept as constant as possible and always there will be little variation between the actual production and desirable production. To implement effective utilization of the mixed model assembly line the following objective functions are to be solved.

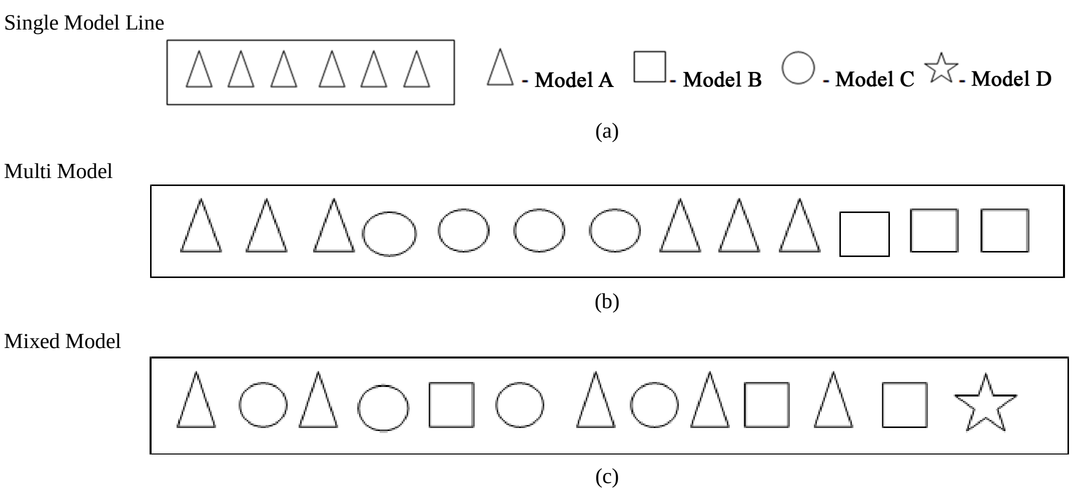

Figure 1. (a)-(c) Single, multi and mixed model lines.

Determination of line cycle times;
Determination of the number and sequence of station on the line;
Line balancing;
Determination of sequencing scheduling for producing different products on the line.

In mixed model assembly line, it requires production sequence to solve the following objectives;
Determination of cycle time;
Determination of work in process;
Determination of effective utilization;
Determination of setup time;
Determination of make span.

Determination of cycle time is the maximum time spent at any one work station. The amount of time is available at each work station to complete all assigned work. Work in process is the arrange number of units in the production system at any given time as the result of the production sequence. Effective utilization is the measure of the ability to keep the schedule level of evenly intermixed by keeping the raw materials for different products of arriving at the system as constant as possible. Setup is required when the different products are produced in the same assembly line. Makeup is the length of production run in the production sequence.

Our objective functions are to minimize the both usage rate and required setups. Setup is required when the two consecutive products in the production sequence are different. The usage rate is a measure of the ability arriving to keep the schedule level on evenly intermixed by keeping the raw materials for different products arriving at the system rate as constant as possible. Generally, the setup and utility are inversely proportional functions. When the utility is low the setup is high. When the utility is high the setup is more. So it has to be balanced between the setup and utility. The weightage of setup and utility has been used, for that the composite objective function value is required.

As an enormous number of possible production sequence, it is difficult to find the optimal solution by using the traditional optimization methods like branch and bound algorithm, goal chasing algorithm, liner and non-linear programming methods and dynamic programming algorithm. It is taking more computation time and also more complexity. Recent trend in solving the optimization problems is heuristic. The heuristic method is used to solve many big problems using simple formula. The algorithms are used to address this type of multiple objective sequencing problems. Many heuristic methods are used to solve the mixed model assembly line problem. The heuristic methods are Genetic Algorithm (GA), Ant-Colony Algorithm (ACA), Particle Swarm Optimization (PSO), Simulated Annealing (SA) and Tabu Search (TS).

It is proposed that the sequence obtained from Miltenburg algorithm is considered as seed. From this seed, the sequence has altered randomly. The utility and setup has been taken as objective function value to optimize the level schedules in JIT production sequence. For this objective function value, different weightages give utility and setup as $E = W_u U + W_s s$. The weights Ws and Wu used for the objective function values are to emphasize the importance of the setup and utility. The weights are determines as W_s = c/number of setups from initial solution, $W_u = c$/number of utility from initial solution, c = constant = 1000. Four types of heuristics are used as per varying the importance of setup and utility.

The objective function value for different weightage has been found from heuristic 1 to 4. The simulated annealing algorithm has utilized to find the optimum function value and optimum sequence. In Section 2, the literature survey of mixed model assembly line and simulated annealing has been presented. In Section 3, the detailed description of the mixed model assembly line model and Miltenburg algorithm has been discussed. In Section 4, the simulated annealing procedure and algorithm have been discussed. Similarly in Section 5 and Section 6, the numerical example and experimental results are presented. Section 7 describes about the computational time and the conclusion is presented in Section 8.

2. Literature Survey

The Toyota production system discusses the leveling and balancing schedule. It shows that the sequence of models in mixed model assembly line is different due to different level of load and usage of parts [1]. The best production schedule by algorithm1 is discussed in [2], but this algorithm is not feasible for the same number of product. The feasible algorithm 2, algorithm 3 are discussed in [2], it also presents heuristics 1, 2. Finally the optimal scheduling algorithm with low variation by heuristics 2 has discussed. An optimal sequence of units that minimizes total line stoppage is discussed in [3]. In [3] the branch and bound method to derive the lower and upper bounds of the total line stoppage time and idle time has been proposed. In [4], the technique utilized is Tabu search to find a sequence when minimization of both material usage rates and setup are of concern, this technique is applied to several problems and resulting sequences are simulated to determine production performance measures of production make span, average work in progress of inventory level. The genetic algorithm provide formidable solution to the multiproduct JIT production sequencing problem with setup and compare favorable to those found using the search techniques of Tabu search and simulated annealing [5] [6]. In [7], an application of the relatively new approach of Ant Colony Optimization (ACO) of address a production sequencing problem, when two objectives are preset, simulated the artificial intelligence agents of virtual ants to obtain desirable solution to a manufacturing logistic problem is discussed.

The two stage variation in mixed model sequencing problem reduces the part variation which is discussed in [8]. A transformed two stage heuristic using product level for reducing the par-level variation in sequencing mixed model assembly line is provided. The performance of genetic algorithm for sequencing problem in mixed model assembly line is investigated in [9]. The results of evaluation indicate that the genetic algorithm that uses the parents-stratum niche cubicle performs better than genetic algorithm with other selection mechanisms. In [10], a multi objective GA for MMAL on JIT assembly line problem where variation of production rates and number of setups are to be optimized minimization of the production rates variation and setup are discussed. The research in JIT sequencing and a Pseudo-polynomial binary search for a feasible B-bounded sequence obtained through perfect matching in bipartite graph solves the single-level min-max absolute-deviation problem are reviewed in [11].

In [12] the major planning approaches in mixed model car assembly sequencing, level scheduling and provides a hierarchical classification scheme to systematically record the efforts in each field has been discussed. It also gives the structure of the vast field of assembly line balancing according to characteristic practical settings and highlights relevant model extension which is required to reflect real world problems. In [12], the reviews on important problem setting alternative buffer configurations, resulting decisions problems are described. The assembly line balancing besides the advantage of genetic algorithm and soft computing and hybrid systems increases the multi objective assembly line problems are studied in [13]. In [14], it presents an integer programming formulation for sequencing problem in mixed model assembly lines where number of temporarily hired utility works and setup are to be optimized simultaneously through a cost function. In [15], it has been proposed to balance the product variety and manufacturing complexity by relative complexity method and find the best set of product variants to be offered while balancing market share and complexity. The multi objective ant colony optimization algorithm for smooth production has been discussed in [16]. The objective is to have minimum number of stations for given cycle time.

The particle swarm optimization algorithm with negative knowledge to solve multi-objective two sided MMAL problems is discussed in [17]. The knowledge of poor solutions is also utility to avoid the pairs of adjacent tasks appearing in the poor solutions from being selected as part or new solutions in the next generation. In [18] it has been solve the balancing and sequencing problem in MMAL to minimize total utility by new mixed integer linear programming model is developed to provide the exact solutions of the problem with standard time. A new hybrid algorithm which executes ant colony optimizations in combination with genetic algorithm (ACO-GA) for MMALBP-1 (mixed model assembly line balancing problem) such as parallel workstations, zoning constraints, and sequence dependent setup times between tasks has been presented in [19]. The multiple colony hybrid bees algorithm for mixed model assembly line balancing problem for low, medium, high variability of setup times and compared with single colony algorithm in terms of computation time and solution quality is discussed in [20].

From the above literature survey, Mondon find the sequence for mixed model assembly, Mitenberg develop it and find feasible algorithm for low variation parts, we use his algorithm as seed, McMullan derive five types of heustics and compare genetic algorithm, tabu, and ant colony algorithm, and from this we use heustics weightages. We study the various types mixed model assembly application, problems and solutions.

3. Mixed Model Assembly Line

The mixed model assembly line may vary from product to product, when large lots of parts assembly, the scheduling is difficult. The usage rate is high or low which depends upon the product assembly. The just in time system works for constant rate of usage for all parts. The small lot of sequence of products is minimizing the variation in the usage of each part that it can achieve a constant rate of part usage by considering only the demand rates for the products.

Minimizing setup is also important in the production line. A set up is required each time two consecutive items in the production sequence are different. An objective function value for the production sequence is utility and setup, then determinate a composite measure of utility and setup. The main aim is to minimize the utility and setup which is combination natural problem. The weightage of each utility and setup has introduced for various weightages applied to utility and setup and find the optimum weightage of function value and sequence.

3.1. Notations

N products with demand $d_1, d_2, \cdots, d_n$. Totally $D_T = \sum_{i=1}^{n} d_i$ units are to be produced. $r_1 = \frac{d_i}{D_T}$, is the proposition of product "*I*" demand to the total demand.

The objective is to schedule the assembly line that the proportion of product "*i*" produced to the total production is close to r_1 as possible.

Let $S_{i,k}$, $i = 1, 2, \cdots, n$, $k = 1, 2, \cdots, D_T$, where $S_{i,k}$ is either 0 or 1 be a production schedule.

If $S_{i,k} = 1$ then product *i* will be produced during stage *k*.

$\sum_{i=1}^{n} S_{i,k} = 1$, for all *k*, because only one product can be produced during each stage.

Let, $x_{i,k} = \sum_{i=1}^{k} S_{i,k}$ be the total production of product *i* over stages 1 to *k*.

Clearly, $x_{i,k}$ is a non-negative integer and $\sum_{i=1}^{n} x_{i,k} = k$, for all *k*. The objective might be one of the following,

$$\text{Minimize} \sum_{k=1}^{D_T} \sum_{i=1}^{n} \left(x_{i,k} - r_i \right)^2 \tag{1}$$

$$\sum_{i=1}^{n} x_{i,k} = k,\ k = 1, 2, \cdots, D_T \tag{2}$$

$$x_{i,k} = k r_i \tag{3}$$

The objective function is equal to zero and constraints are satisfied.

$$\sum_{i=1}^{n} x_{i,k} = \sum_{i=1}^{n} k r_i = k \sum_{i=1}^{n} r_i = k \tag{4}$$

3.2. Miltenberg Algorithm

The flowchart of miltenberg's algorithm is shown in **Figure 2** and it finds the nearest point *M* to point *X*, where $\sum_{i=1}^{n} m_i = \sum_{i=1}^{n} x_i = k$

1. Calculate $k = \sum_{i=1}^{n} x_i$
2. Find the nearest non-negative integer m_i to each coordinate x_i, that is, Find m_i, so that $\left| m_i - x_i \right| \le \frac{1}{2}$, $i = 1, 2, \cdots, n$.
3. Calculate $k_m = \sum_{i=1}^{n} m_i$

a) If $k - k_m = 0$ stop. The nearest integer point is $M = \left(m_1, m_2, \cdots, m_n \right)$

b) If $k - k_m > 0$ go to step 5

c) If $k - k_m < 0$ go to step 6

4. Find the coordinate x_i with the smallest $m_i - x_i$ increment the value of this m_i; $m_i \rightarrow m_i + 1$ Go to step 3
5. Find the coordinate x_i with largest $m_i - x_i$ decrement the value of this m_i; $m_i \rightarrow m_i - 1$Go to step 3

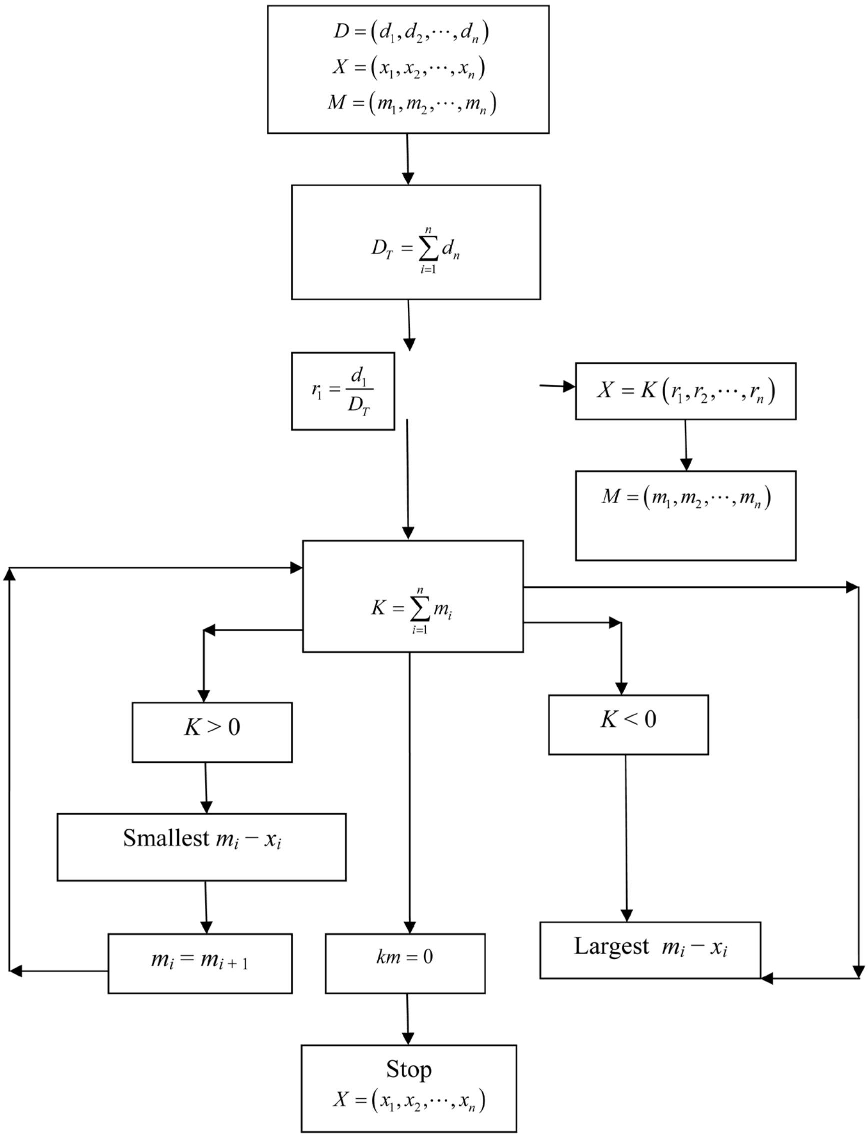

Figure 2. Flowchart of Miltenberg's algorithm.

3.3. Objective Function

3.3.1. Minimizing the Setups

The number of setup

$$S=\sum_{k=1}^{D_T} S_k \tag{5}$$

where, k = Index of the position in the sequence if the product in position k is different from product in position $k-1$, then setup is require and $S_k = 1$, 0 otherwise it is assumed here that initial setup is required regardless in sequence. It should be noted that the setup time are assumed to sequence independent, so that the machine does not depend on which other product preceded it on that machine.

3.3.2. Minimizing the Utility

$$U=\sum_{k=1}^{D_T}\sum_{i=1}^{n}\left(x_{i,k}-k\cdot\frac{d_i}{D_T}\right)^2 \tag{6}$$

While keeping the usage of materials as constant as smooth as possible is of extreme importance when different products are to be made on an assembling line, this usage rate of material is especially sensitive to the production sequence. Because the material usage rate is sensitive to the production sequence, considerable effect has gone into development or techniques intended to minimize this material usage rate.

3.4. Composite Objective Function Value

An objective function value of the production sequence is then determined with composite measure of utility and setup, where W_u is the weight placed upon the usage rate and W_s is the weight place upon the number of setups. The composite functions is

$$\text{Min } E = W_s S + W_u U \tag{7}$$

Sequencing Heuristics Used

In [5], it derives the different heuristics according to different weightage of utility and setup. The different heuristic are used to obtain the different objective function are

Heuristic 1

$$\text{Min } E = S \tag{8}$$

This heuristic sequences the products in such a way that the required number of set ups is minimized. It does not require minimum setup, it does not require heuristics, it get from inspection.

Heuristic 2

$$\text{Min } E = U \tag{9}$$

This heuristic minimizes the material usage rate and it is addressed in [21], which is simplification of Miltenburgs sequence in [2].

Heuristic 3

$$\text{Min } E = 14.2755S + 1U \tag{10}$$

This heuristic sequence produces in such a way that composite function of both utility and set up is minimized. The coefficients used for this objective function come from sampling in such a way that both utility and setups are gives equal contribution. The coefficient is derived in [4].

Heuristic 4

$$\text{Min } E = 3 \times 14.2755S + 1S \tag{11}$$

In this heuristic, the number of setup is 3 times minimizing, so that the utility and setups are minimizing. The importance of setup time is three times more than utility but the utility importance is still considered.

Heuristic 5

$$\text{Min } E = 14.2755S + 3U \tag{12}$$

In this the minimizing utility is 3 times as minimizing the number of setup. The utility and setup is consider but the importance of utility is three time than setup.

For first two objectives, it does not require to minimize, because it comes directly from its own minimizes the number of require setups and utility rates. But other objectives to be minimize by using simulated annealing with respect of the weights W_u, W_s, it reflect the level of importance of setup and utility. Four objective functions were evaluated and for varying weightages can be placed to lower the number of setups and usage rates. The used four types of heuristics are $E = U + S$, $E_1 = U + 3S$, $E_2 = U + 3 \times 14.27S$, $E_3 = 3U + S$.

3.5. Heuristic Methods and Proposed Algorithm

A Heuristic is simply a rule of thumb, hopefully will find a good answer. Heuristic are typically used to solve complex, large, non-liner non-convex multivariate combinational optimization problems that rate difficult to solve to optimality. Many heuristic methods are simulated annealing, Genetic algorithm, particle swarm optimization, ant colony algorithm and Tabu search.

A genetic algorithm is a search strategy. To implement GA, a representation of the parameters in the problem to be searched is developed first, several initial GA solutions are formed to make an initial populations of several so called chromosomes, and then the GA operations selection, recombination and mutations are employed to improve the search repetitively as measured by a fitness or evaluation function. Particle optimization is developed from the behavior of bird and fish while searching for food, the birds are either scattered or go together before they locate the place where they can find the food while searching food they go from one place to another place where good food resource available. This information is transmitted and good information is equal most optimist solution. Ant colony optimization is the behavioral simulation of social insects such as Bees, ant, wasps. ACO simulate the collective forging habits of ants venturing out for food. A chemical substance deposited by ants as they travel, pheromone provides ants with ability to communicate with each other. Ants move randomly when they encounter a pheromone trail, they decide whether or not to follow it. The probability that an ant choose one path over another is governed by the amount of pheromone on potential path of interest. Tabu search setup that it utilities a short term memory component of previous solutions which presents cycling, which can be in turn result in being trapped at local optima thereby preventing finding an optimal solution. Tabu search takes initial solutions and makes changes to this solution during the iterative process. As changes are made, they are recorded on a Tabu list which is simply a listing of the recent changes or moves. It a move under consideration appears on the list, the move is forbidden unless its objective function value satisfies what known as aspiration criteria. The basic procedure is repeated until user specified stopping criteria are met. Any efficient optimization algorithm must use these techniques to find global maximum by exploration, new and unknown search space to make use of knowledge found at point previously visited. These two requirements are full filled by simulated annealing algorithm. SA can deal with highly now-liner models, chaotic and noisy data and many constraints. It is robust and general technique. The simulated annealing algorithm is better than local search methods in flexibility and ability to approve global optimality. The algorithm is quite versatile since it does not rely on any restrictive properties of the model. The **Figure 3** shows the structure of SA algorithm.

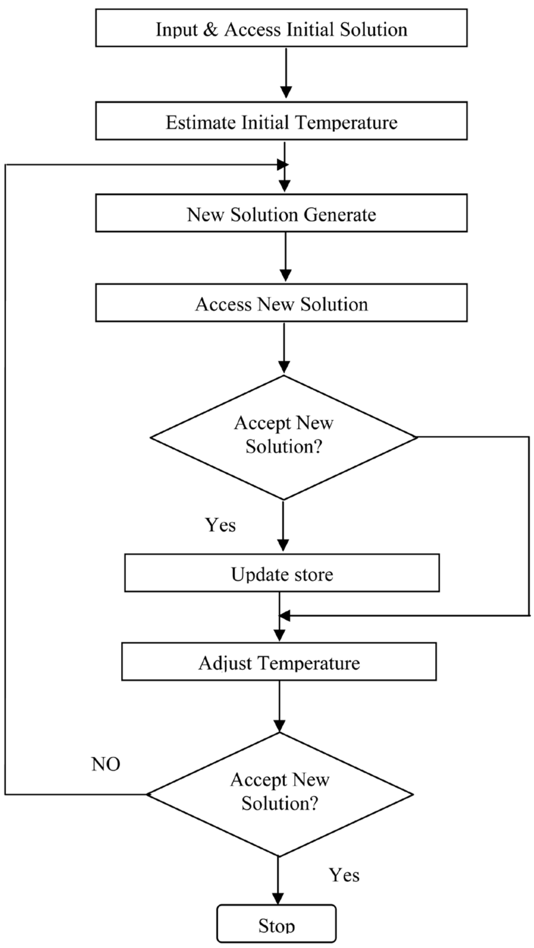

Figure 3. Structure of SA algorithm.

4. Simulated Annealing

Simulated annealing is a random search technique which exploits an analogy between the ways in which a Meta cools into minimum energy crystalline structure. The simulated annealing method is for obtaining good solutions to difficult optimization problems. In the early 80's, the concept of annealing has been discussed in [22]. In simulated annealing, first melt the solid by increasing the temperature then slowly cool it so it crystallizes into perfect lattice. Solution considered as states of the physical system, objective function as energy and control parameter as temperature.

In simulated annealing, the initial state of a thermodynamic system energy E, temperature T and the change in energy ΔE is completed. If the change in energy is negative, the new state is accepted. If the change in energy is positive is accepted with $\mathrm{e}^{-\frac{\Delta E}{T}}$, then the processes is then repeated sufficient times to give good sampling statistics of current temperature and then temperature is decremented until the final temperature or number of iteration or sufficient computational time is attain. In this, the parameter selection is very important. The parameters are initial temperature, final temperature and the number of iterations. If high initial temperature is chosen, it takes number of iterations for convergence. If a small initial temperature, the search is not adequate to the search space before finding the time optimum. The advantage of simulated annealing is the ability to move from local optimization thus the ability to find the global optimum is not related to the initial condition. The disadvantage of simulated annealing is the subjective nature of choosing the configuration parameters.

4.1. Annealing Procedure

4.1.1. Initial Temperature

In [25], the initial and final temperature were determined by information obtained the trial to annealing process. In this trial, a certain number of random moves were performed to record the changes in results in the objective function. From this result, the minimum temperature is given by,

$$T_0 = \Delta E_{\min} + \frac{1}{10}\left(\Delta E_{\max} - \Delta E_{\min}\right) \tag{13}$$

$$T_0 = \Delta E_{\min} \tag{14}$$

4.1.2. Decrementing Temperature

One of the major issues is related to the annealing schedule to cool the temperature during the annealing process. Various methods are used to reduce the temperature as shown in **Table 1**.

The Connolly method is selected because it is clear that when the temperature is too high, a lot of uphill moves are accepted, when the temperature is too low, the probability is falling into a local minimum. The temperature should be between these two extreme that the temperature is high but the cooling is low. The Connolly was designed based on this idea; hence this method has been adopted.

In Connolly method, during these trials the initial temperature T_0 and final temperature T_f are determined and M refers to the number of pair wise exchanges examined.

$$T_{i+1} = \frac{T_i}{1+\beta T_i}, \quad \beta = \frac{T_0 - T_f}{M T_0 T_f} \tag{15}$$

$$M = \frac{n(n-1)}{2} \tag{16}$$

Table 1. Methods to reduce the temperature during annealing process.

Wilhelm & War method [23]	Golden & Skiscin method [24]	Connolly method [25]	Vilarinho & Simaria method [26]
$T_{i+1} = \alpha T_i$, $0 < \alpha < 1$	$T_{i+1} = T_i - \frac{T_0}{25}$, $i = 0, \cdots, 25$	$T_{i+1} = \frac{T_i}{1+\beta T_i}$, $\beta = \frac{T_0 - T_f}{M T_0 T_f}$	$T_{k+1} = T_k - T$, T_{k+1} = Temperature of next range, T_k = Initial temperature, T = reduction of temperature

In this equation parameter β usually has a small value and there after the temperature reduction proceeds slowly and n is the number of demand. The algorithm perform depends upon the cooling rate than individual temperature for better result, reduction rate should be slower in middle temperature range.

4.1.3. Random Number Generation

A significant component of an SA code is the random number generator, which is used both for generating random changes in the control variables and for the increase acceptance test. It is important, particularly when tacking large scale problems requiring thousands of iterations, so that the random numbers generator used have good spectral properties. The Microsoft excel VBA procedure method is inbuilt into the program for find the random number generation.

4.1.4. Number of Iteration

A constant number of iteration at each temperature is generally employed. Another method is only one iteration at each temperature but to decrease the temperature very slowly. The iterations at each temperature is proportional to $n = t/(t + 1)$. An alternative is to dynamically change the number of iterations as the algorithm progress. We use the number of iteration is total number product.

4.1.5. Stopping Criteria

A given total number of iterations have been completed or fixed amount of execution time, the stopping criteria can either be a suitably low temperature or when the system is "frozen" at the current temperature (*i.e.* no better or worse moves are being accepted). Once the final temperature has been attained, the process will stop.

4.1.6. Parameter Set

The initial and final temperatures were determined by information obtained in trail to the annealing process. In this trail certain number of random moves was performed to record the resulting changes in the objective function. From this result, the minimum value of ΔE min and maximum value of ΔE max are to be final. The initial temperature is set as

$$T_0 = \Delta E_{\min} + \frac{1}{10}\left(\Delta E_{\max} - \Delta E_{\min}\right) \tag{17}$$

and the final temperature $T_0 = \Delta E_{\min}$

Another annealing schedule is how to cool the temperature during the annealing process

$$T_{i+1} = \frac{T_i}{1+\beta T_i}, \quad \beta = \frac{T_0 - T_f}{M T_0 T_f}, \quad M_0 = \frac{n(n-1)}{2} \tag{18}$$

where, M refers to the number of pair wise exchange examined,

T_{i+1} = next temperature to be set, n = no of demand.

4.2. Simulated Annealing Algorithm

Choose initial temperature, temperature reduction factor and final temperature;

Select the objective function;

Select the number of iteration;

Find the initial energy state (E_0);

Find the randomizing, select the another energy state (E_1);

Find the difference between the two energy states $\Delta E = E_1 - E_0$;

Check whether $\Delta E < 0$. If yes store the energy and find the randomly energy state. If no generate randomly $X \in U(0, 1)$ Check the whether $X < \mathrm{e}^{-\frac{\Delta E}{T}}$;

If yes store the energy state otherwise go for iteration. The above function is repeating until the all the iteration and reduce the temperature according to reduction factor. Continue the above procedure until reach up to final temperature.

***Simulated* Annealing Pseudocode**

Generate initial sequence by Miltenberg algorithm and calculate objective function E. The flowchart for simulated algorithm and proposed algorithm are shown in **Figure 4** and **Figure 5**.

Select an initial solution E_0 and E_1, $E_1 = E$;
Select an initial temperature $T_i > 0$;
Select a temperature reduction function T_{i+1}
Select an final temperature T_f;
Maximum iteration count Max IT;
Repeat
Set iteration count $IT = 0$;
$IT = IT + 1$;
Randomly generate sequence by VBA method and calculate E_1;
set $\Delta E = f(E_1) - f(E_0)$;

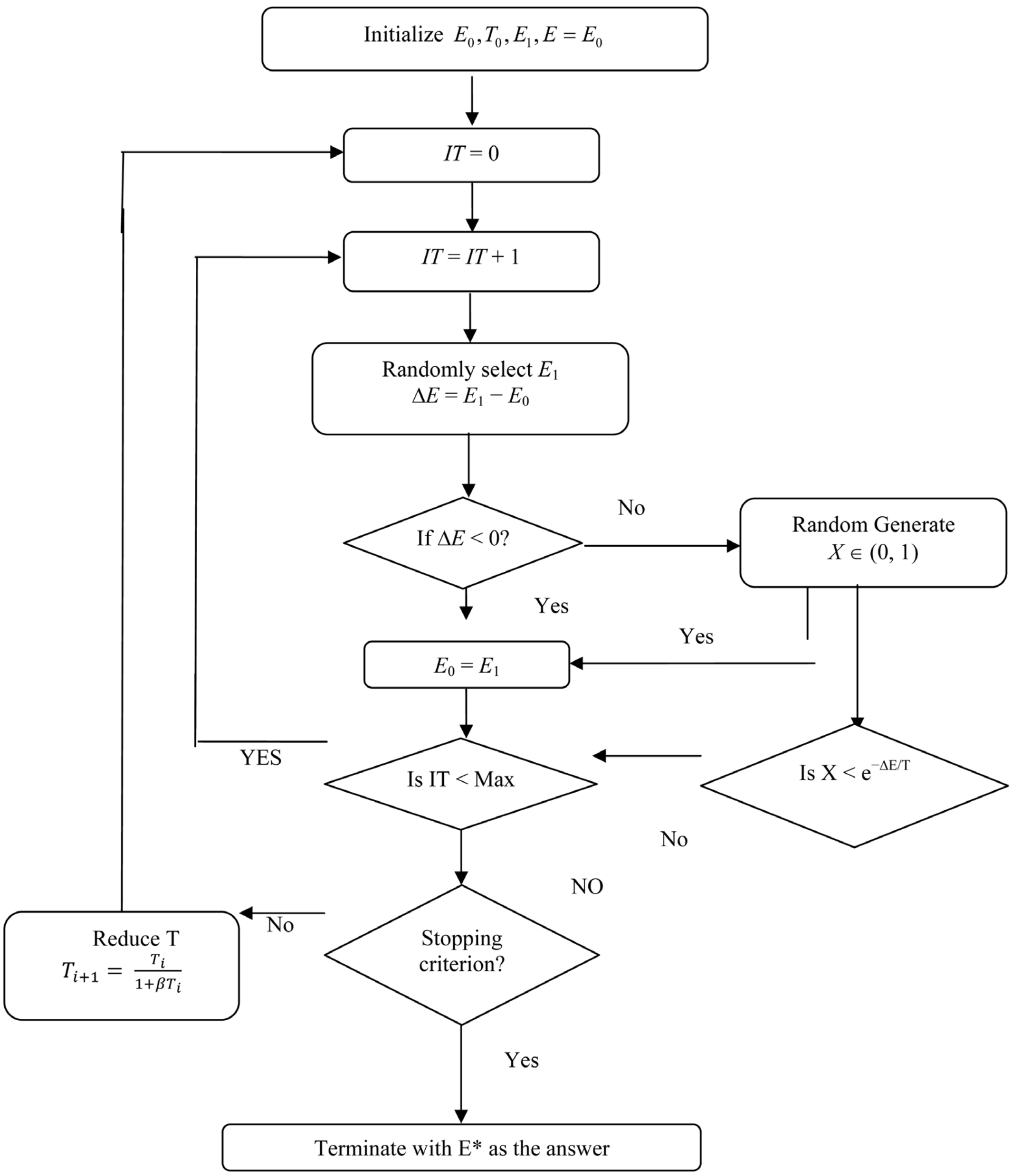

Figure 4. Flowchart of simulated algorithm.

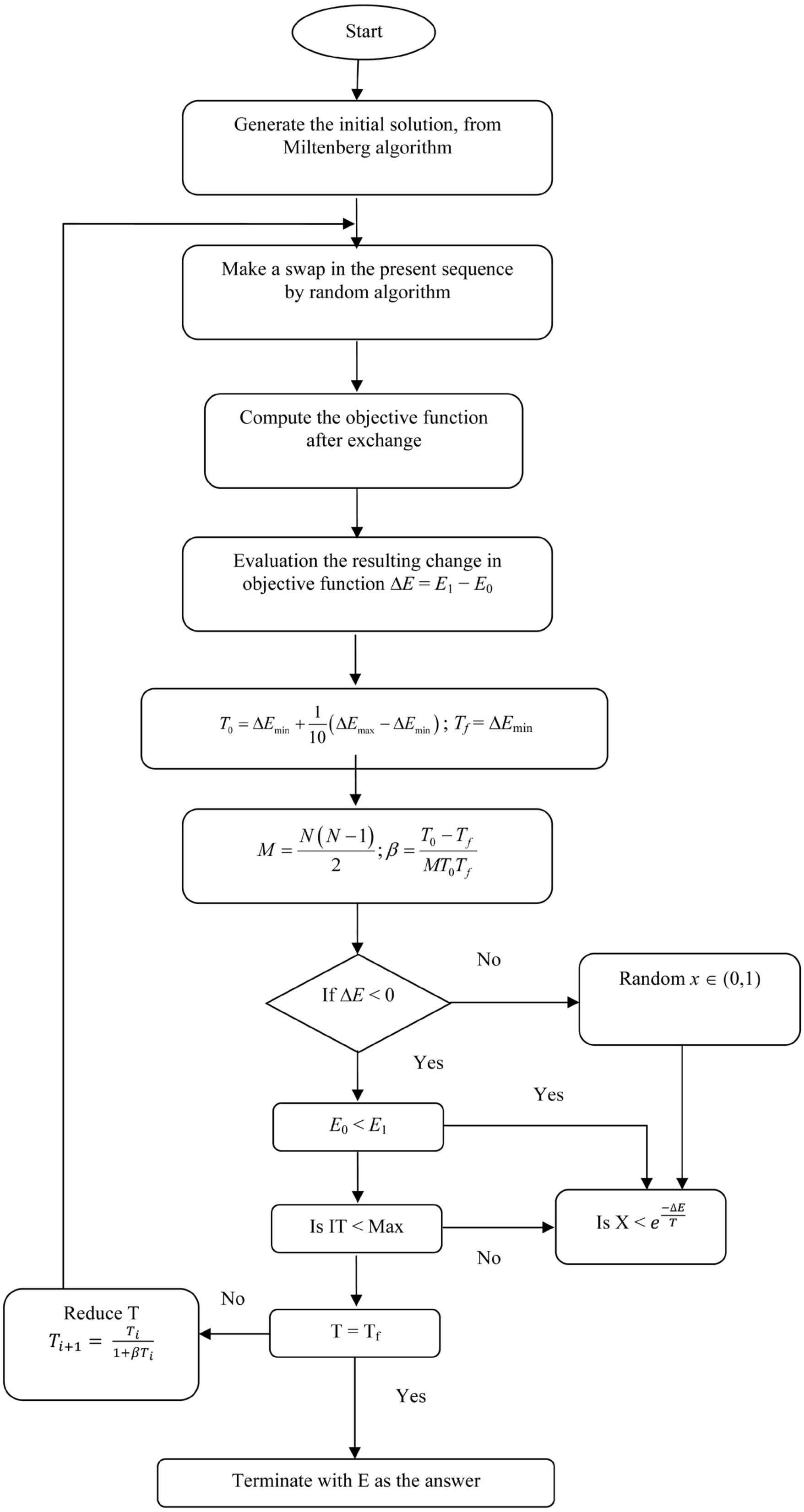

Figure 5. Flowchart of proposed algorithm.

If $\Delta E < 0$;
then set $E_0 = E_1$ (downhill move) else;
generate random X uniformly in the range [0, 1];
If $x < \exp(-\Delta E/T)$;
then set $E_0 = E$(uphill move) ;
If $f(E_0) < f(E)$ then $E_1 = E_0$.
Until IT = Max;
Set $T_i = (T_{i+1})$;
Until stopping condition becomes true.
Output E0 as an approximation to the optimal solution.

5. Numerical Example

5.1. Miltenberg Algorithm

In Table 2, a test problem of 5 types of products A, B, C, D and E are to be produced and their demands of each type requires 2. The following steps will illustrate the working of Miltenburg algorithm.

$D_T = A + B + C + D + E$
$D_T = 2+2+2+2+2 = 10$
$d_1 = 2, d_2 = 2, d_3 = 2, d_4 = 2\ d_5 = 2$
$r_1 = r_2 = r_3 = r_4 = r_5 = 2/10$
K = No of stages = 2 + 2 + 2 + 2 + 2 = 10
M = (M1, M2 ……M10) X = (x1, x2 ……x10)
At stage K = 1
X1 = K (r1, r2, r3, r4, r5)
X1 = 1(2/10, 2/10, 2/10, 2/10, 2/10)
X1 = (0, 0, 0, 0, 0)
M1 = (0, 0, 0, 0, 0)
Km = 0 + 0 + 1 = 1, k-km = 1 − 0 = 1 > 0 go to step 5
Find the smallest coordinate of M
Select $m_1 \rightarrow m_1 + 1 = 0 + 1 = 1$
X1 = (1, 0, 0, 0, 0)
Km = 0 + 0 + 1 = 1, k-km = 1 − 1 = 0 stop
Schedule product-A
At stage K = 2
X2 = 2(2/10, 2/10, 2/10, 2/10, 2/10)
X2 = (0, 0, 0, 0, 0)
M2 = (0, 0, 0, 0, 0)
Km = 0 + 0 + 1 = 1 k-km = 1 − 0 = 1 >0 go to step 5
Find the smallest coordinate of M
Select $m_1 \rightarrow m_1 + 1 = 0 + 1 = 1$
X2 = (1, 0, 0, 0, 0)
Km = 0 + 0 + 1 = 1 k-km = 2 − 1 = 1 > 0 go to step 5
Find the smallest coordinate of M

Table 2. Test problem.

A	2
B	2
C	2
D	2
E	2

Select $m_2 \rightarrow m_2 + 1 = 0 + 1 = 1$
X2 = (1, 1, 0, 0, 0)
Km = 0 + 1 + 1 = 2, k-km = 2 − 2 = 0 stop
Schedule product-B
At stage K = 3
X3 = 3(2/10, 2/10, 2/10, 2/10, 2/10)
X3 = (1, 1, 1, 1, 1)
M3 = (1, 1, 1, 1, 1)
Km = 1+ 1+1+1+1 = 5 k-km = 3 − 5 = −2 < 0 go to step 6
Find the Largest coordinate of M_3
Select $m_5 \rightarrow m_5$ -1 = 1 − 1 = 0
X3 = (1, 1, 1, 1, 0)
Km = 4, k-km = 3 − 4 = −1 < 0 go to step 6
Find the Largest coordinate of M_3
Select $m_4 \rightarrow m_4 - 1 = 1 - 1 = 0$
X6 = (1, 1, 1, 0, 0)
Km = 3, k-km = 3 − 3 = 0 stop
Schedule product-C
At stage K = 4
X4 = 4(2/10, 2/10, 2/10, 2/10, 2/10)
X4 = (1, 1, 1, 1, 1)
M4 = (1, 1, 1, 1, 1)
Km = 1 + 1 + 1 + 1 + 1 = 5, k-km = 4 − 5 = −1 < 0 go to step 6
Find the Largest coordinate of M_4
Select $m_5 \rightarrow m_5$ -1 = 1 − 1 = 0
X4 = (1, 1, 1, 1, 0)
Km = 4, k-km = 4 − 4 = 0 Stop
Schedule product - D
At stage K = 5
X5 = 5(2/10, 2/10, 2/10, 2/10, 2/10)
X5 = (1, 1, 1, 1, 1)
M5 = (1, 1, 1, 1, 1)
Km = 5, k-km = 5 − 5 = 0 Stop
Schedule product – E
At stage K = 6
X6 = K (r1, r2, r3, r4, r5)
X6 = 6(2/10, 2/10, 2/10, 2/10, 2/10)
X6 = (1, 1, 1, 1, 1)
M6 = (1, 1, 1, 1, 1)
Km = 5, k-km = 6 − 5 = 1 > 0 go to step 5
Find the smallest coordinate of M
Select $m_1 \rightarrow m_1 + 1 = 0 + 1 = 1$
X6 = (2, 1, 1, 1, 1)
Km = 6, k-km = 6 − 6 = 0 stop
Schedule product-A
At stage K = 7
X7 = K (r1, r2, r3, r4, r5)
X7 = 7(2/10, 2/10, 2/10, 2/10, 2/10)
X7 = (1, 1, 1, 1, 1)
M7 = (1, 1, 1, 1, 1)
Km = 5, k-km = 7 − 5 = 2 > 0 go to step 5
Find the smallest coordinate of M
Select $m_1 \rightarrow m_1 +1 = 1+ 1 = 2$

X7 = (2, 1, 1, 1, 1)
Km = 8, k-km = 7 − 6 = 1 2 > 0 go to step 5
Find the smallest coordinate of M
Select $m_2 \rightarrow m_2 + 1 = 1 + 1 = 2$
X7 = (2, 2, 1, 1, 1)
Km = 7, k-km = 7 − 7 = 0 Stop
Schedule product-B
At stage K = 8
X8 = 8(2/10, 2/10, 2/10, 2/10, 2/10)
X8 = (2, 2, 2, 2, 2)
M8 = (2, 2, 2, 2, 2)
Km = 10, k-km = 8-10 = −2 < 0 go to step 6
Find the Largest coordinate of M_4
Select $m_5 \rightarrow m_5 - 1 = 2 - 1 = 1$
X8 = (2, 2, 2, 2, 1)
Km = 9, k-km = 8-9 = −1 < 0 go to step 6
Find the Largest coordinate of M_4
Select $m_4 \rightarrow m_4 - 1 = 2 - 1 = 1$
X8 = (2, 2, 2, 1, 1)
Km = 8, k-km = 8 − 8 = −0 Stop
Schedule product-C
At stage K = 9
X9 = 9(2/10, 2/10, 2/10, 2/10, 2/10)
X9 = (2, 2, 2, 2, 2)
M9 = (2, 2, 2, 2, 2)
Km = 10 k-km = 9-10 = −1 < 0 go to step 6
Find the Largest coordinate of M_5
Select $m_5 \rightarrow m_5$ -1 = 2 − 1 = 1
X9 = (2, 2, 2, 2, 1)
Km = 9, k-km = 9 − 9 = −0 Stop
Schedule product-D
At stage K = 10
X10 = 10(2/10, 2/10, 2/10, 2/10, 2/10)
X10 = (2, 2, 2, 2, 2)
M10 = (2, 2, 2, 2, 2)
Km = 10, k-km = 10 − 10 = 0 Stop
Schedule product-E
Final sequence is A, B, C, D, E, A, B, C, D, E
Set up: The number of setup require is 9

Utility = $U = \sum_{k=1}^{D_T} \sum_{i=1}^{n} \left(x_{i,k} - k \cdot \frac{d_i}{D_T} \right)^2$

$(1 - 2/10)^2 + (0 - 2/10)^2 + (0 - 2/10)^2 + (0 - 2/10)^2 + (0 - 2/10)^2 + (1 - 4/10)^2 + (1 - 4/10)^2 + (0 - 4/10)^2 + (0 - 4/10)^2 + (0 - 4/10)^2 + (1 - 6/10)^2 + (1 - 6/10)^2 + (1 - 6/10)^2 + (0 - 6/10)^2 + (0 - 6/10)^2 + (1 - 8/10)^2 + (1 - 8/10)^2 + (1 - 8/10)^2 + (1 - 8/10)^2 + (0 - 8/10)^2 + (1 - 10/10)^2 + (1 - 10/10)^2 + (1 - 10/10)^2 + (1 - 10/10)^2 + (1 - 10/10)^2 + (2 - 12/10)^2 + (1 - 12/10)^2 + (1 - 12/10)^2 + (1 - 12/10) + (1 - 12/10)^2 + (2 - 14/10)^2 + (2 - 14/10)^2 + (1 - 14/10)^2 + (1 - 14/10)^2 + (1 - 14/10)^2 + (2 - 16/10)^2\ (2 - 16/10)^2 + (2 - 16/10)^2 + (1 - 16/10)^2 + (1 - 16/10)^2 + (2 - 18/10)^2 + (2 - 18/10)^2 + (2 - 18/10)^2 + (2 - 18/10)^2 + (1 - 18/10)^2 + (2 - 20/10)^2 + (2 - 20/10)^2 + (2 - 20/10)^2 + (2 - 20/10)^2 + (2 - 20/10)^2$

$U = 7.99$

$E = W_u U + W_s S \quad = (1 \times 7.99 + 1 \times 9) = 16.99\ (W_u = 1, W_s = 1),$

$E = 17$

The obtained Miltenburg algorithm sequence is A, B, C, D, E, A, B, C, D, E. Now this seed sequence will generate the following five sequences randomly by using method of VBA Microsoft excel for the position of product to be produced like sequence-1 as A, C, E, B, D, C, D, A, E, B, sequence-2 as E, A, C, D, A, B, B, D, C, E, sequence-3 as B, C, C, B, A, E, A, D, D, E, sequence-4 as B, E, A, B, D, A, C, D, C, E, and sequence-5 as D, C, C, B, A, E, A, B, D. For initial trial, five random sequences as said above are generated and the corresponding objective function values are calculated as $E_1 = 17$, $E_2 = 18$, $E_3 = 29$, $E_4 = 23$, $E_5 = 21$.

5.2. SA Algorithm

Step 1: *Find the initial temperature*

From initial trial, take $E_{\max}$, $E_{\min}$

$$T_0 = \Delta E_{\min} + \frac{1}{10}\left(\Delta E_{\max} - \Delta E_{\min}\right) = 17 + 1/10\left(29.00 - 17\right)$$

$T_0 = 18.200$

Step 2: Final temperature, $T_{final} = \Delta E_{\min} = 17.00$

Step 3: Temperature change $T_{i+1} = \dfrac{T_i}{1+\beta T_i}$

$$M = n\left(n-1\right)/2 = 10\left(10-1\right)/2 = 45$$

$$\beta = \frac{T_0 - T_f}{M T_0 T_f} \quad T_f = \frac{18.200 - 17.001}{45 \times 18.200 \times 17.001} = 2.4 \times 10^{-3}$$

$T_1 = 18.2000$, $T_2 = 18.1964$, $T_3 = 18.1928$

$T_4 = 18.1892$ as like $T_5 = 18.18$ to T_n

Step 4:

Consider initial sequence obtained from Miltenburg algorithm and the initial energy state is its objective function value (E_0) = 17. Now randomly generate sequences by VBA Microsoft excel method and calculate corresponding energy state by its objective function value

$E_1 = 19$, $E_2 = 17$, $E_3 = 20$

$E_4 = 26$, $E_5 = 19$, $E_6 = 32$, $\cdots$, E_n

$\Delta E = E_1 - E_0 = 19 - 17 = 2$

Step 5

Check whether $\Delta E < 0$, $2 < 0$, No, generate random number $X = 0.90$

Find $e^{-\Delta E/T} = e^{-2/18.20} = 0.89$

Check $X < e^{-\Delta E/T}$, 0.90 < 0.89 No, the energy state is rejected.

Go to step 4, generate another sequence by random method

New sequence is 3425123145 and objective function value $E_2 = 17$,

$\Delta E = E_2 - E_0 = 17 - 17 = 0 < 0$, yes, store the energy state, $E_0 = E_2$

Go to step 4, generate another sequence by random method

New sequence is 5341132425 and objective function value $E_3 = 20$,

$\Delta E = E_2 - E_0 = 19.99 - 17 = 2.99 < 0$, NO, find $e^{-\frac{\Delta E}{T}} = e^{-\frac{2.99}{18.20}} = 0.84$

Generate random number $X = 0.46$, Check 0.46 < 0.84 then accept the energy state reset $E = E_2$ then go to next iteration, up to 10 iteration, repeat procedure then reduce the temperature from 18.22 to 18.19 and continue the process. Up to reach of final temperature 17.00, the number of sequence generated is 3594. Finally, the obtained sequence is E, D, C, B, A, D, A, C, B, E and objective value is 15.99 by simulated annealing, which is minimum compared with Miltenburg algorithm. The comparison of numerical results sequence is shown in **Table 3**.

6. Experimental Design and Discussion of Results

The three problem sets are taken from [7] for analysis. The problem set 1 contains 7 types of problems with total demand is 10 each. The obtained results are tabulated in **Table 4**. The solution obtained is 2524 to 4104. The

Table 3. Comparison numerical results sequence.

Stage	1	2	3	4	5	6	7	8	9	10
Product sequence miltenberg	A	B	C	D	E	A	B	C	D	E
Product sequence random	A	C	E	B	D	C	D	A	E	B
Product sequence Simulated annealing	E	D	C	B	A	D	A	C	B	E

Table 4. Solution for problem set 1.

	Miltenburg algorithm					Simulated annealing algorithm				
Problem Set	Scheme	u	s	z	Sequence	u	s	z	Sequence	RPT
A (22222)	W	7.99	9	17	1234512345	7.99	8	15.99	3254114235	5.94
	W1	7.99	9	136.47	1234512345	14.79	7	85.10	3324451215	37.64
	W2	7.99	9	393.43	1234512345	28.02	4	199.30	4433112255	49.35
	W3	7.99	9	152.47	1234512345	14	6	127.65	4522113345	16.27
B (32221)	W	5.69	9	14.69	1234152341	5.69	9	14.69	1243513241	0
	W1	5.69	9	134.17	1234152341	29.52	4	86.60	5223344111	35.45
	W2	5.69	9	391.13	1234152341	30.92	4	202.20	3352244111	48.30
	W3	5.69	9	145.57	1234152341	13.31	6	125.55	1354422311	13.75
C (33211)	W	5.4	9	14.39	1234125312	6.2	8	14.20	1322154312	1.32
	W1	5.4	9	133.07	1234125312	14.95	5	86.30	5244331112	35.14
	W2	5.4	9	434.08	1234125312	24.95	5	239.05	5244331112	44.92
	W3	5.4	9	144.67	1234125312	10.81	6	118.05	1223354112	18.40
D (42211)	W	4.9	9	13.9	1231451231	5.69	8	13.69	1235114321	1.51
	W1	4.9	9	133.37	1231451231	13.72	5	85.07	1142233511	36.21
	W2	4.9	9	390.33	1231451231	30.92	4	202.2	5422331111	48.19
	W3	4.9	9	143.179	1231451231	13.70	4	112.47	1143322511	21.14
E (43111)	W	5.8	9	14.80	1231241521	6.2	8	14.20	2113251421	4.05
	W1	5.8	9	134.27	1231241521	11.72	5	84.37	3115222411	37
	W2	5.8	9	391.23	1231241521	30.22	4	201.5	3452221111	48.49
	W3	5.8	9	145.87	1231241521	13.20	5	110.97	5112233411	23.92
F (52111)	W	5.4	9	14.40	1231241521	6.6	7	13.60	1251143211	5.55
	W1	5.4	9	133.87	1231241521	11.02	5	82.37	4111225311	38.4
	W2	5.4	9	390.80	1231241521	30.42	4	201.7	4352211111	48.38
	W3	5.4	9	144.67	1231241521	11	5	104.37	4111225311	27.85
G (61111)	W	7.00	7	14.00	1121341151	8	5	13.0	4521311111	70.1
	W1	7.00	7	99.92	1121341151	8.99	5	80.35	2111534111	24.8
	W2	7.00	7	306.78	1121341151	27.02	4	27.02	4523111111	35.3
	W3	7.00	7	120.377	1121341151	9	5	98.37	3111145211	18.64

Sequence (A-1, B-2, C-3, D-4, E-5).

problem set 2 contains 9 types of problems with total demand is 12 each and the results are tabulated in **Table 5**. The solution obtained is 4300 to 5680. The problem set 3 contains 9 types of problems with total demand is 15 each and the results are in **Table 6**. The solution obtained is 8524 to 11104.

Table 5. Solution for problem set 2.

		Miltenberg algorithm				Simulated algorithm				
Problem set	**Scheme**	**u**	**s**	**z**	**Sequence**	**u**	**s**	**z**	**Sequence**	**RPT**
A (81111)	W	7.94	8	15.94	112131141511	9.11	6	15.11	111421113511	5.2
	W1	7.94	8	122.10	112131141511	13.11	5	84.46	311111142511	30.82
	W2	7.94	8	350.42	112131141511	13.06	5	227.16	311111425111	35.17
	W3	7.94	8	138.01	112131141511	13.11	5	110.69	311111124511	19.79
B (72111)	W	6.77	9	15.77	121311451121	6.11	9	15.11	112114135121	0
	W1	6.77	9	135.20	121311451121	8.61	7	108.50	211134511121	19.74
	W2	6.77	9	392.15	121311451121	26.61	6	283.53	243511111121	27.6
	W3	6.77	9	148.76	121311451121	8.61	7	125.72	211113451121	15.48
C (63111)	W	6.66	11	17.66	121312415121	8.50	8	16.5	211152341121	6.56
	W1	6.66	11	163.63	121312415121	11.83	7	111.72	311122541121	31.72
	W2	6.66	11	477.68	121312415121	31.49	6	288.41	532241111121	39.62
	w3	6.66	11	177.00	121312415121	11.84	7	135.41	311122541121	23.49
D (62211)	W	6.30	10	16.30	121314511231	6.3	9	15.30	132115411231	6.13
	W1	6.30	10	149.00	121314511231	18.80	6	104.42	351111142231	29.91
	W2	6.30	10	434.50	121314511231	18.80	6	275.72	341111152231	36.54
	W3	6.30	10	161.61	1221314511231	3.38	9	138.57	121431115231	14.24
F (53211)	w	5.88	11	16.88	123141251321	6.38	10	16.38	213152411321	2.96
	W1	5.88	11	162.85	123141251321	20.05	6	105.67	522111143321	35.11
	W2	5.88	11	476.90	123141251321	20.05	6	276.97	522111143321	41.92
	W3	5.88	11	174.63	123141251321	8.72	8	140.32	311224115321	19.64
G (52221)	W	6.86	10	16.86	123141251321	6.86	10	16.86	123411521341	0
	W1	6.86	10	149.56	123411521341	20.69	6	106.31	451111223341	43.25
	W2	6.86	10	435.06	123411521341	20.69	6	277.61	451111223341	36.19
	W3	6.86	10	163.31	123411521341	14.86	7	144.49	431111225341	11.52
H (43221)	W	5.80	11	16.80	123412513421	5.80	11	16.80	124312513421	0
	W1	5.80	11	162.77	123412513421	21.80	6	107.42	221113354421	34.00
	W2	5.80	11	476.82	123412513421	21.80	6	278.72	221113354421	41.54
	W3	5.80	11	174.41	123412513421	20.97	6	148.554	522111334421	14.82
J (44211)	W	6.19	11	17.19	1234124512312	6.86	10	16.86	132215421312	1.91
	W1	6.19	11	163.16	1234124512312	23.19	6	108.81	511122243312	33.31
	W2	6.19	11	477.21	1234124512312	23.19	6	280.11	422211153312	41.30
	W3	6.19	11	175.553	1234124512312	10.69	8	146.2433	311222451312	16.69
K (33222)	W	7.08	11	18.08	123451234512	7.08	11	18.08	123451234512	0
	W1	7.08	11	164.05	123451234512	22.58	6	108.08	221133445512	34.04
	W2	7.08	11	478.10	123451234512	25.24	6	282.16	224411335512	40.98
	W3	7.08	11	178.22	123451234512	12.58	8	151.910	542211534512	14.76

Sequence (A-1, B-2, C-3, D-4, E-5).

Table 6. Solution for problem set 3.

Problem set	Miltenberg algorithm					Simulated algorithm				RPT
	scheme	u	s	z	sequence	u	s	z	sequence	
A (10211)	W	8.35	10	18.35	112131141511211	9.55	8	17.55	112411113511211	4.35
	W1	8.35	10	151.05	112131141511211	12.35	7	112.24	211111143511211	25.71
	W2	8.35	10	436.455	112131141511211	13.35	7	312.02	211111134511211	28.51
	W3	8.35	10	167.79	112131141511211	17.11	6	136.97	211111143511211	18.366
B (11111)	W	10.22	8	18.22	111211314111511	10.88	7	17.88	111211143111511	1.86
	W1	10.22	8	124.38	111211314111511	92.88	6	98.50	432111111111511	20.80
	W2	10.22	8	352.72	111211314111511	32.90	5	246.95	311111241111511	29.98
	W3	10.22	8	144.85	111211314111511	12.83	6	124.30	211114321151121	14.18
C (93111)	W	8.26	12	20.26	121131412151121	10.26	9	19.26	211114321151121	4.9
	W1	8.26	12	179.50	121131412151121	13.86	8	128.02	111223411151121	28.67
	W2	8.26	12	522.02	121131412151121	38.00	11	356.90	111212134151121	31.63
	W3	8.26	12	196.03	121131412151121	9.10	9	155.75	311411221151121	20.54
D (75111)	W	8.62	14	22.62	121231412512121	9.288	12	21.288	211241123512121	5.88
	W1	8.62	14	208.40	121231412512121	22.88	9	151.31	222111134512121	27.52
	W2	8.62	14	608.00	121231412512121	22.91	9	408.20	222111134512121	32.86
	W3	8.62	14	225.68	121231412512121	12.89	10	181.39	112221143512121	19.625
E (73221)	W	7.34	14	21.37	123141512134121	8.31	12	20.31	211431521134121	4.96
	W1	7.34	14	207.15	123141512134121	26.71	9	155.14	341112251134121	25.10225
	W2	7.34	14	606.75	123141512134121	29.13	9	414.42	345221111134121	31.69
	W3	7.34	14	221.92	123141512134121	12.44	10	180.04	411132251134121	18.87
F (63221)	W	7.55	14	21.55	123141523141231	7.81	13	20.82	132411532141231	3.38
	W1	7.55	14	207.33	123141523141231	14.75	1 0	157.45	411332251141231	24.71.07
	W2	7.55	14	606.89	123141523141231	33.95	9	419.24	453322111141231	30.82
	W3	7.55	14	207.37	123141523141231	5.06	10	157.88	114223351141231	23.86
G (53331)	W	8.35	14	22.35	123415213412341	8.35	14	22.35	123415213412341	0
	W1	8.35	14	208.13	123415213412341	28.35	9	156.78	331112254412341	24.672
	W2	8.35	14	607.79	123415213412341	28.41	9	413.7	221113354412341	31.92
	W3	8.35	14	224.88	123415213412341	13.96	11	198.86	411332251412341	11.59
H (43331)	W	8.35	14	22.35	123451234152341	8.35	14	22.35	124351243152341	0
	W1	8.35	14	208.13	123451234152341	30.88	9	159.31	224453311152341	23.45
	W2	8.35	14	607.69	123451234152341	28.22	9	413.51	522334411152341	31.95
	W3	8.35	14	224.84	123451234152341	18.35	10	197.76	153344221152341	12.04
J (33333)	W	11.9	14	25.99	123451234512345	12.49	12	24.99	133252441512345	3.8
	W1	11.9	14	211.77	123451234512345	32	9	160.43	441133225512345	24.24
	W2	11.9	14	611.34	123451234512345	32	9	417.29	114433225512345	31.74
	W3	11.9	14	235.77	123451234512345	18	11	210.97	154422331512345	10.5151

Sequence (A-1, B-2, C-3, D-4, E-5).

The Miltenburg algorithm, Random generation algorithm and simulated annealing algorithm are coded in Microsoft Excel in macro and executed on an Intel processor at 2 GB under windows XP using 250 MB of RAM. The problem set 1 has solved with 7 types problem for all problems to all demand of product is equal but demand of each product is different for each problem.

The 4 types of heuristics have been solved. The Heuristics 1 has minimum weightage for both setups a usage rate. Heuristics 2 have a composite value of 14.755 for setup. The coefficient used from the sampling, so the setup and material usage made equal contribution to the objective function. Heuristics 3 is weighted three times important of setup than minimum usage rate. Heuristics 4 is three times of usage rate than setup. So the importance of usage rate is three times than minimum setup. The all the problems with heuristics 1 to 4 are solved by Miltenburg algorithm and simulated annealing algorithm. The two methods are compared by RPT.

As shown in **Table 7**, in Heuristics1 (W_u = 1, W_s = 1) the weightages of setup and utility are 1, denote as w. In Heuristics 2 (W_u = 1, Ws = 14.27) the weightages of setup is 14.27 and utility are 1, denote as w1. Heuristics3 (W_u = 1, W_s = 42.81) the weightages of setup is 42.81 and utility are 1, denote as w2. Heuristics 4 (W_u = 3, W_s = 14.27) the weightages of setup is 14.27 and utility are 3 denote, as w3. All the results obtained by heuristics 1 to 4 are tabulated. For all the problem sets, the solutions obtained by Simulated Annealing heuristics shows minimum compared with Miltenburg Algorithm.

The number of setup is equal or small reduction. The comparison results of heuristics 1 to 4 are tabulated in **Table 8**. Heuristics 1 show RPT of 0% to 7.01% and number or setup is low in SA compare with Miltenburg. In SA number or setup is requires is 4 to 8 where as in Miltenburg are 7 to 9. The objective function in SA is 13 to 14.69 where as in Miltenburg are 13.90 to 17. In this number or setup and usage rates is balance in SA compare with Miltenburg.

In heuristics 2 shows RPT of 24 to 38.47% and number or setup is low in SA compare with Miltenburg. In SA number or setup is requires is 4 to 7 where as in Miltenburg are 9. The objective function in SA is 85 to 80.35 where as in Miltenburg are 136.47 to 106.89. In this number or setup and usage rates is balance in SA compare with Miltenburg.

In heuristics 3 the RPT is 49.35 to 35.45 Miltenburg the number or setup is 9. The objective function in SA is 202.2 to 198.30 whereas in Miltenburg are 391.23 to 306.7. So the setup and usage rate is much balance in simulated annealing but in heuristics 4, the RPT is 27.85 to 13.75 the objective function value in SA is 127.65 to 98.67 but in Miltenburg is 152.47 to 104.37.

In problem set 1, heuristic 3 shows high RPT. But the utility is more and setup is less. In heuristic 1 and 4 shows low utility and setup is high. In heuristic 2, balancing the setup and utility as well as the RPT is high. The Problem type F (5 2 1 1 1) shows utility 11.02 and setup 5, objective function value is 82.37 RPT is 38.4. The

Table 7. Heuristic weightages.

Heuristic number	W_S	W_U
1	1	1
2	14.27	1
3	3 * 14.27 = 42.81	1
4	14.27	3

Table 8. Comparison results of Miltenburg and simulated annealing for heuristic 1 to 4.

Heuristics	Problem set 1			Problem set 2			Problem set 3		
	Miltenburg algorithm	SA	RPT	Miltenburg algorithm	SA	RPT	Miltenburg algorithm	SA	RPT
Heuristic 1	13.90	13.00	7.01	15.77	15.11	6.13	18.22	17.55	5.88
Heuristic 2	99.92	80.35	38.40	122.10	84.46	43.25	124.38	98.50	28.67
Heuristic 3	306.78	239.05	49.35	350.42	227.16	41.92	352.72	246.3	32.86
Heuristic 4	120.37	98.37	27.85	138.01	110.69	23.49	144.85	128.02	23.86

optimum sequence is D, A, A, A, B, B, E, C. In problem set 2 the heuristic 3 shows high RPT. But the utility is more and setup is less. In heuristic 1 and 4 show low utility and setup is high and RPT is low. But in heuristic 2 shows the RPT is moderate as well as balancing the utility and setup.

In problem type F (5 2 2 2 1) shows the utility 20.69 and setup 6 and objective function value is 106.31 and RPT is 43.25. So the obtained optimum sequence is D, E, A, A, A, A, B, B, C, C, D, A. In problem set 3 the heuristic 3 shows high RPT of 32.86, but the utility is more and setup is less. In heuristic 1 and 4 show low utility and setup is high and RPT is low. But in heuristic 2 shows the RPT is moderate as well as balancing the utility and setup. In problem type D (93111) shows the utility13.86 and setup 8 and objective function value is and RPT is 28.67. So the obtained optimum sequence is A, A, A, B, B, C, D, A, A, A, E, A, A, B, A.

So it has been concluded that from the above problem sets analysis, RPT is high in heuristic3 for the problem set1. The simulated annealing solution is 49.35% improved then Miltenberg algorithm. The utility and setups are most minimized. The RPT is low in heustics 1 for problem set 3. The simulated annealing solution is only 5.88% improved then Miltenberg algorithm solution. From the above results, we conclude that setup is three times more important than utility.

7. Computational Time

The solutions for all the three problem sets are found and CPU time taken from each heuristic is presented. The details are presented in **Tables 9-12** and the problem sets are mentioned in Annexure. In the computational problems, finding the CPU time is important because CPU time should be less. In the problem sets, the average cpu time is 70.66.and solutions is 5662.

In problem set 1, the average CPU time is 23.47 and solution is 2947. For problem set 2 average CPU time is 58.16 and solution is 4820. For problem set 3 average CPU time is 130.35 and solution is 9218.

The comparison of CPU time is shown in **Figure 6**. Among the all weightages, W3 have less CPU time of 66.61, no of solution is 5229 and w2 take CPU time of 66.97 for 5342 solution, and w1 take CPU time of 81.12 for 6288 solutions.

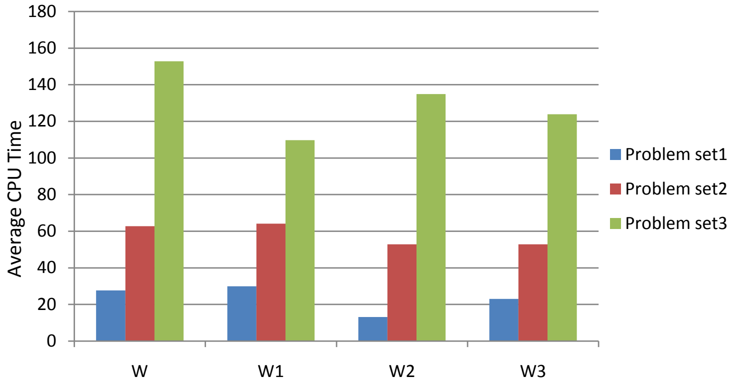

Figure 6. Comparison of CPU time.

Table 9. Computational time for problem set 1.

Problem	W		W1		W2		W3	
	No. of solution	CPU time (secs)	No. of solution	CPU time (secs)	No. of solution	CPU time (secs)	No. of solution	CPU time (secs)
B (22222)	3594	20.10	2494	13.36	2734	19.10	2624	20.10
C (32221)	4154	24.20	2784	14.36	3054	20.10	2534	14.10
D (33211)	4104	30.24	3216	43.30	3293	15.20	2524	12.52
E (42211)	3914	37.14	2604	35.00	2744	10.16	2544	33.10
F (43111)	3174	42.10	2454	34.40	2494	09.39	2687	37.55
G (52111)	2884	15.10	2764	35.10	2804	09.13	2664	21.16
H (61111)	3354	25.10	3034	34.20	2914	09.01	2364	23.10

Table 10. Computational time for problem set 2.

Problem	W		W1		W2		W3	
	No. of solution	CPU time (secs)	No. of solution	CPU time (secs)	No. of solution	CPU time (secs)	No. of solution	CPU time (secs)
B(81111)	4420	32	4432	58	4456	50	4372	50
C(72111)	6556	89	4468	61	4780	55	4648	55
D(63111)	5680	49	4756	70	4650	54	4768	54
E(62211)	5200	90	4540	61	4200	53	4564	53
F(53211)	5092	38	4672	56	4300	40	4564	40
G(52221)	5836	70	5140	78	5080	45	4360	45
H(43211)	5680	65	4884	65	5128	60	4492	60
I(44211)	5560	58	5284	68	5668	64	4600	64
J(33222)	5776	74	4144	60	4252	55	4329	55

Table 11. Computational time for problem set 3.

Problem	W		W1		W2		W3	
	No. of solution	CPU time (secs)	No. of solution	CPU time (secs)	No. of solution	CPU time (secs)	No. of solution	CPU time (secs)
B (102111)	7500	92	9525	116	10264	85	8614	125
C (111111)	10744	88	9600	117	9994	160	9034	132
D (93111)	10579	180	8944	129	9154	140	8494	115
E (75111)	9829	181	8674	87	9574	152	8404	110
F (73221)	9285	170	8954	76	9484	144	8524	120
G (63221)	11104	195	8615	118	8884	123	8569	130
H (53331)	10984	180	9229	126	9574	150	8629	142
I (43331)	10789	170	9159	115	7872	117	8449	124
J (33333)	8600	120	8610	104	8929	143	8705	117

Table 12. Total average number of solutions and CPU time.

Problem set	W		W1		W2		W3		Cumulative average of all heuristics	
	Average no of solutions	Average CPU time	Average no of solutions	Average CPU time	Average no of solutions	Average CPU time	Average no of solutions	Average CPU time	Average no of solutions	Average CPU time
Problem set 1	3597	27.71	2764	29.96	2862	13.15	2563	23.09	2947	23.47
Problem set 2	5333	62.77	4702	64.11	4723	52.88	4521	52.88	4820	58.16
Problem set 3	9934	152.88	9034	109.79	9303	134.88	8602	123.88	9218	130.35

8. Conclusion

In this paper, the various heuristic methods based on simulated annealing have been studied for solving production sequence in level schedule optimization problem. The Miltenburg algorithm has been used to find the sequence for scheduling. The sequence has been modified to obtain another sequence and to get utility and setup estimations by giving different weightages. These weights are then adjusted to obtain the desired value. According to different weightages, the utility and number of setups are obtained for heuristics 1 to 4. These selective heuristics procedures are applied for both Miltenburg algorithm and simulated annealing algorithm. The proposed algorithm based on simulated annealing technique has been applied to find production sequences when objective function values of setup and usage rates are desired as minimum. The results obtained are found to be useful to take good managerial decisions on production sequences. Three problem sets up to 15 numbers of products are solved and the results obtained for all the heuristics and results are compared to obtain balanced setup and utility and minimize the objective function value.

References

[1] Monden, Y. (1983) Toyota Production System. Institute of Industrial Engineers Press, Norcross.

[2] Miltenburg, J. (1989) Level Schedules for Mixed-Model Assembly Lines in Just-in-Time Production System. *Management Science*, **35**, 192-207. http://dx.doi.org/10.1287/mnsc.35.2.192

[3] Zhao, X.P. and Ohno, K. (1994) A Sequencing Problem for a Mixed-Model Assembly Line in a JIT Production System. *Computers & Industrial Engineering*, **27**, 71-74. http://dx.doi.org/10.1016/0360-8352(94)90240-2

[4] McMullen, P.R. (1998) JIT Sequencing for Mixed-Model Assembling Lines with Setups Using Tabu Search. *Production Planning & Control*, **19**, 504-510. http://dx.doi.org/10.1080/095372898233984

[5] McMullen, P.R. and Taransewich, P. (2000) Using Genetic Algorithm to Solve the Multi-Product JIT Sequencing Problem with Set-Ups. *International Journal of Production Research*, **38**, 2653-2670. http://dx.doi.org/10.1080/002075400411411

[6] McMullen, P.R. and Frazier, G.V. (2000) A Simulated Annealing Approach to Mixed-Model Sequencing with Multiple Objectives on a Just-in-Time Line. *Institution of Industrial Engineering Transactions*, **32**, 679-686. http://dx.doi.org/10.1080/07408170008967426

[7] McMullen, P.R. (2001) An Ant Colony Optimization Approach to Addressing a JIT Sequencing Problem with Multiple Objectives. *Artificial Intelligence in Engineering*, **15**, 309-317. http://dx.doi.org/10.1016/S0954-1810(01)00004-8

[8] Ding, Y. and Zhu, J. (2000) A Transformed Two-Stage Method for Reducing the Part-Usage Variation and a Comparison of the Product-Level and Part-Level Solutions in Sequencing Mixed-Model Assembly Line. *European Journal of Operational Research*, **127**, 203-216. http://dx.doi.org/10.1016/S0377-2217(99)00322-7

[9] Ponnambalam, S.G., Aravindan, P. and Subba Rao, M. (2003) Genetic Algorithms for Sequencing Problems in Mixed Model Assembly Lines. *Journal of Computers and Industrial Engineering*, **45**, 669-690. http://dx.doi.org/10.1016/j.cie.2003.09.001

[10] Mansouri, A. (2005) A Multi-Objective Genetic Algorithm for Mixed-Model Sequencing on JIT Assembly Lines. *European Journal of Operational Research*, **167**, 696-716. http://dx.doi.org/10.1016/j.ejor.2004.07.016

[11] Dhamala, T.N. and Kubiak, W. (2005) A Brief Survey of Just-in-Time Sequencing of Mixed-Model System. *International Journal of Operations Research*, **2**, 38-47.

[12] Boyson, N. and Scholle, A. (2012) Resequencing of Mixed-Model Assembly Lines: Survey and Research Agenda. *European Journal of Operational Research*, **216**, 594-604. http://dx.doi.org/10.1016/j.ejor.2011.08.009

[13] Matondang, M.Z. and Jambak, M.I. (2010) Soft Computing in Optimizing Assembly Line Balancing. *Journal of Computer Science*, **6**, 141-162. http://dx.doi.org/10.3844/jcssp.2010.141.162

[14] Goard, V. and Jennet, J. (2010) Optimal Sequencing of Mixed Models with Sequence-Dependent Setups and Utility Workers on an Assembly Line. *International Journal of Production Economics*, **123**, 290-300. http://dx.doi.org/10.1016/j.ijpe.2009.09.001

[15] Wang, H., Zhu, X.W., Wang, H., Hu, S.J., Lin, Z.Q. and Chen, G.L. (2011) Multi-Objective Optimization for Product Variety and Manufacturing Complexity in Mixed-Model Assembly Systems. *Journal of Manufacturing Systems*, **30**, 16-27. http://dx.doi.org/10.1016/j.jmsy.2011.03.002

[16] Yagmahan, M.M.A.L. (2011) Mixed-Model Assembly Line Balancing Using a Multi-Objective Ant Colony Optimization Approach. *Expert Systems with Applications*, **38**, 12453-12461. http://dx.doi.org/10.1016/j.eswa.2011.04.026

[17] Chutima, P. and Chimklai, P. (2012) Multi-Objective Two-Sided Mixed-Model Assembly Line Balancing Using Particle Swarm Optimization with Negative Knowledge. *Journal Computers and Industrial Engineering*, **62**, 39-55. http://dx.doi.org/10.1016/j.cie.2011.08.015

[18] Mosadegh, M., Zandign, M. and Fatemi Ghomi, S.M.T. (2012) Simultaneous Solving of Balancing and Sequencing Problems with Station-Dependent Assembly Times for Mixed-Model Assembly Lines. *Applied Soft Computing*, **12**, 1359-1370. http://dx.doi.org/10.1016/j.asoc.2011.11.027

[19] Akpinar, S., Bayhan, M. and Baykasoglu, A. (2013) Hybridizing ant Colony Optimization via Genetic Algorithm for Mixed-Model Assembly Line Balancing Problem with Sequence Dependent Setup Time between Tasks. *Applied Soft Computing*, **13**, 574-589. http://dx.doi.org/10.1016/j.asoc.2012.07.024

[20] Akpinar, S. and Baykasoglu, A. (2014) Modeling and Solving Mixed-Model Assembly Line Balancing Problem with Setups. Part II: A Multiple Colony Hybrid Bees Algorithm. *Journal of Manufacturing Systems*, **33**, 445-461. http://dx.doi.org/10.1016/j.jmsy.2014.04.001

[21] Ding, F.Y. and Cheng, L.P. (1993) An Effective Mixed-Model Assembly Line Sequencing Heuristic for Just-in-Time Production Systems. *Journal of Operations Management*, **11**, 45-50. http://dx.doi.org/10.1016/0272-6963(93)90032-K

[22] Kirkpatrick, S.C.D., Gelatt Jr., C.D. and Vecchi, M.P. (1983) Optimization by Simulated Annealing. *Science*, **220**, 671-680. http://dx.doi.org/10.1126/science.220.4598.671

[23] Wilhelm, M.R. and Thomas, L.W. (1987) Solving Quadratic Assignment Problems by "Simulated Annealing". *Institution of Industrial Engineering Transactions*, **19**, 107-119. http://dx.doi.org/10.1080/07408178708975376

[24] Golden, B.L. and Skiscim, C.C. (1986) Using Simulated Annealing to Solve Routing and Location Problems. *Naval Research Logistics Quarterly*, **33**, 261-279. http://dx.doi.org/10.1002/nav.3800330209

[25] Connolly, D.T. (1990) An Improved Annealing Scheme for the QAP. *European Journal of Operational Research*, **46**, 93-100. http://dx.doi.org/10.1016/0377-2217(90)90301-Q

[26] Vilarinho, P.M. and Simaria, A.S. (2002) A Two-Stage Heuristic Method for Balancing Method for Balancing Mixed-Model Assembly Line with Parallel Workstations. *International Journal of Production Research*, **40**, 1405-1420. http://dx.doi.org/10.1080/00207540110116273

Annexure

Problem Set 1

B	6	1	1	1	1
C	5	2	1	1	1
D	4	2	2	1	1
E	4	3	1	1	1
F	3	3	2	1	1
G	3	2	2	2	2
H	2	2	2	2	2

Problem Set 2

B	8	1	1	1	1
C	7	2	1	1	1
D	6	3	1	1	1
E	6	2	2	1	1
F	5	3	2	1	1
G	5	2	2	2	1
H	4	3	2	2	1
I	4	4	2	1	1
J	3	3	2	2	2

Problem Set 3

B	11	1	1	1	1
C	10	2	1	1	1
D	9	3	1	1	1
E	7	5	1	2	1
F	7	3	2	2	1
G	6	3	3	2	1
H	5	3	3	3	1
I	4	3	3	3	2
J	3	3	3	3	3