World Applied Sciences Journal 13 (8): 1916-1926, 2011
ISSN 1818-4952

Multi-Objective Simulated Annealing Algorithms for Scheduling Just-In-Time Assembly Lines

Ghorbanali Mohammadi

Department of Industrial Engineering, College of Engineering,
Shahid Bahonar University of Kerman, P.O. Box 76175-133, Kerman, 7618891167, Iran

Abstract: New approaches to sequencing mixed-model manufacturing systems are present. These approaches have attracted considerable attention due to its potential in dealing with difficult optimization problems. This paper presents Multi-Objective Simulated Annealing Algorithms (MOSAA) approaches to Just-In-Time (JIT) sequencing problem where workload-smoothing (WL) and number of set-ups (St) are to be optimized simultaneously. Mixed-model assembly lines are types of production lines where varieties of product models similar in product characteristics are assembled. Moreover, this type of problem is NP-hard.Two annealing methods are proposed to solve the multi-objective problem and find an efficient frontier of all design configurations. The performances of the two methods are tested on several problems from the literature. Experimentation demonstrates the relative desirable performance of the presented methodology.

Key words: Scheduling · Just-in-time · Mixed-model assembly line · Sequencing · Simulated Annealing

INTRODUCTION

In mixed-model assembly line production systems, management would like to sequence different products without having an excessive number of set-ups or changeovers to reduce costs. A good sequence should have an acceptable level of intermixing of products and at the same time an acceptable number of required set-ups[1].

For JIT systems, Monden [1] define two goals for the final assembly sequencing problem; *Goal 1*: even distribution of workload (total assembly time on each workstation on the line) and *Goal 2*: keeping a constant rate of usage for each part used on the final assembly line. *Goal 1* recognises that models may not have the same operation time at each one workstation on the line and seeks to smooth out the workload on the final assembly line to reduce line inefficiencies such as idleness, work deficiency, utility work and work congestion [for more detail 2,3,4]. *Goal 2*, on the other hand, requires that the quantity of each part used by the mixed-model final assembly line per unit time be kept as constant as possible. Miltenburg [5] has formulated the problem as a nonlinear integer-programming problem having the objective of minimizing the total variation of actual production rates from the desired production rates. This is referred to as "levelling" the schedule. Inman and Bulfin [6] Give an efficient Earliest Due Date (EDD) algorithm for an objective function that is mathematically different, but intuitively similar to the objective function of Miltenburg [7]. Kubiak and Sethi [8] developed an optimization algorithm for the problem and its extension that runs in polynomial time in the total demand for all products produced on the line over a given time horizon. They show that the objective function may actually be represented by using penalties for deviation from most even, perhaps unrealizable, distribution of production during a specified time horizon and if these penalty functions are non-negative and convex and then the problem can be reduced to an assignment problem. Determining an optimal schedule for most of the industrial applications is a very difficult combinatorial problem [9]. In a JIT production system, the objective for *Goal 2* should be to sequence models on the mixed-model final assembly line in such a way that usage variation and workload smoothing with required set-up cost of each model at each level of the manufacturing process is kept as constant as possible. Finding production sequences

Corresponding Author: Ghorbanali Mohammadi, Department of Industrial Engineering, College of Engineering, Shahid Bahonar University of Kerman, P.O. Box 76175-133, Kerman, 7618891167, Iran.
Tel: +98-341-2112861, Fax: +98-341-2112861, E-mail: gmohammadi@mail.uk.ac.ir.

with desirable levels of both numbers of set-ups and production rates variation is NP-hard as pointed out by [10, 11]. In this environment, the sequencing problem becomes a multi-objective problem.

In this paper, two objectives are considered: minimizing the optimization of workload-smoothing (*WL*); and minimization of set-up costs (*St*).

A Multiple Objective Sequencing Problem in Jit Production Lines Workload-smoothing Problem: Controlling a mixed-model assembly manufacturing facility, operating under a just in time (JIT) production control system, is by setting a production schedule for the last process in the facility, which is usually a mixed-model final assembly line. The workload-smoothing problem is to smooth the workload on the final assembly line to reduce the chance of production delays and stoppages.

M different products with demand $d_1, d_2, \ldots d_m$ are assembled in the mixed-model final assembly line. The line consists of (*S*) number of stations between which products move until production is completed. The available production time at each station on the assembly line is fixed. Let $P_{m,s}$ be the processing time of model (*m*)on workstation *s* where *s* = 1,2,...,*S* stations and ($t_{m,s}$)be the total processing time required by all requirements of model (*m*) on workstation *s*, $t_{m,s} = d_m P_{m,s}$. The workload-smoothing problem is minimizing the deviation of actual workload from the ideal workload. The following model is modified from smooth production loads model presented by [12].

$$\text{Minimize} \quad WL = \sum_{k=1}^{D_T}\sum_{m=1}^{M}\sum_{s=1}^{S}\left(p_{m,s}x_{m,k} - \gamma k\left(\frac{t_{m,s}}{T_p}\right)\right)^2$$

Subject to:

$$\sum_{m=1}^{M} x_{m,k} = 1, k = 1,\ldots D_T \qquad (1)$$

$$0 \le x_{m,k} - x_{m,k-1} \le 1 \text{ for m=1,..., } M, k = 1,..., D_T \qquad (2)$$

$$x_{k,m} \text{ is a non-negative integer, } \forall_m, \forall k \qquad (3)$$

Set-Up Costs: Costs of operations performed on every product, but in different choices are affected by sequencing Mixed-Model assembly lines. For these types of operations, set-up costs may be incurred each time the operation switches from one choice to another. Therefore, an incentive exists to sequentially assemble products having the same choice of an operation.

The second objective function is to minimize the number of required set-up costs (*St*) in a production sequence. Many assembly operations often require sequence-dependent set-ups. For instance, an automotive body station need set-ups when the door types are changed. Similarly, an engine mounting station requires a set-up when the engine types are changed. Burns and Daganzo [13] addressed sequence-dependent set-ups cost and production capacity in determining an assembly line job sequence. Mathematical formulation is as follows:

$$\text{Min} \sum_{m=11}^{M}\sum_{k=1}^{D_T}\sum_{s=1}^{S}\sum_{r=1}^{M} X_{s,m,r}C_{s,m,r}$$
$$\sum_{m=1}^{M}\sum_{r=1}^{M} x_{m,k.r} = 1 \qquad \forall k \qquad (4)$$

$$\sum_{k=1}^{D_T}\sum_{r=1}^{M} x_{m,k,r} = d_m \quad \forall m$$
$$x_{m,k,r} = 0 \quad or \quad 1 \quad \forall m,k,r \qquad (5)$$

Where $C_{s,m,r}$ is the set-up costs required when model type is changed from *m* to *r* and $x_{s,m,r}$ is 1 if model type *k* and *r* are assigned, respectively, at the *k*th and (*k*+1) st position in a sequence; otherwise, $X_{s,m,r}$ is 0. In this work, it is assumed the set-up time is sequence-dependent. Equation (4) is a set of position constraints indicating that every position in a sequence is occupied by exactly one product. Equation (5) imposes the restriction that all the demands should be satisfied.

Efficient Frontier Approach: Multiobjective optimization problems require separate techniques, which are very different to the standard optimization techniques for single objective optimization. It is very clear that if there are two objectives to be optimized, it might be possible to find a solution, which is the best with respect to the first objective and other solution, which is the best with respect to the second objective.

It is convenient to classify all potential solutions to the multiobjective optimization problem into *dominated* solutions and *non-dominated* (*efficient*) solutions. As solution *x* is dominated if there exits a feasible solution *y* not worse than *x* on all coordinates, i.e. for all objectives

$$f_i(i = 1,..., k)$$

$$f_i(x) \ge f_i(y) \text{ for all } 1 \ge i \ge k.$$

If a solution is not dominated by any other feasible solution, we call it non-dominated (or efficient) solution. All efficient solutions might be of some interest; ideally, the system should report back the set of all efficiency optimal points. In other disciplines, the term efficient is also known as *Pareto optimality*, *admissibility*, or *non-inferiority*. We call the set of all efficient points the efficient set.

General Simulated Annealing Algorithm: The principal idea used in simulated annealing was first proposed by [14]. Kirkpatrick *et al.* [16,17] applied the concept of annealing to combinatorial optimization problem. This concept is based on an analogy between the physical annealing process of solids and solution process of combinatorial optimization problems.

The physical annealing process which obtains low energy states of a solid in a heat bath may be modelled as follows [17]: In each step an atom is given a small random displacement and the resulting change in the energy of the system, ΔE, is computed. If $\Delta E \leq 0$, then the displacement is accepted and the configuration with the displaced atom is used as the starting point of the next step. If $\Delta E > 0$, the case is treated probabilistically, the probability that the configuration is accepted is determined using Equation [18], where T is the temperature and $k_b$is Boltzmann's constant.

$$P(\Delta E) = e^{-\frac{\Delta E}{k_b T}} \tag{6}$$

A random number uniformly distributed on the interval [0,1] is selected and compared with $P(\Delta E)$. If the random number is less than $P(\Delta E)$, then the new configuration is accepted and used to start the next step. Otherwise, the configuration is rejected. The process is continued until an "*equilibrium*" state is achieved, then the temperature is lowered according to the annealing schedule. This procedure is repeated until the system *freezes*. At each temperature, the annealing schedule must allow the simulation to proceed sufficiently long for the system to reach steady state condition (equilibrium point). Simulated annealing for the model is as follow:

Initial Temperature: The number of iterations during the annealing process partly depends on the initial temperature. The procedures for setting the initial temperature may be broadly classified into two types. Most schemes determined the initial temperature as fixed numbers prior to execution of the annealing process. Golden and Skiscim, [19] and Bukard and Rendl, [20] set high initial temperatures, while Wilhelm and Ward, (1987) starts at a low temperature.

In Connolly's method, the initial and final temperatures were determined by information obtained in trials prior to the annealing process. In these trials, a certain number of random moves are performed to record the resulting changes in the objective function. From the results, the minimum value $\Delta f_{\min}$ and the maximum value $\Delta f_{\min}$ for the changes in the objective function are calculated for these exchanges. Using these values, the initial temperature, T_0 and the final temperature, T_F, are set according to equations (7) and (8), respectively:

$$T_0 = \Delta f_{\min} + \frac{1}{10}\left(\Delta f_{\max} - \Delta f_{\min}\right), \tag{7}$$

$$T_F = \Delta f_{\min} \tag{8}$$

Temperature Tuning (Cooling): One of the major issues related to the annealing schedule is how to cool the temperature during the annealing process. Each annealing schemehasitsownindividual function. For example, [20,21] calculate the next temperature, T_{i+1} using equation [19], where the parameter a is usually set close to one. Golden and Skiscim, [19] used Equation [22], which reduces the temperature by 1/25 of the initial temperature at each stage.

$$T_{i+1} = aT_i \tag{9}$$

$$T_{i+1} = T_i - \frac{T_0}{25} \tag{10}$$

Another method employs information obtained from trails prior to the execution of annealing process. In Connolly, [23] during these trails the initial temperature T_0 and final temperature T_f are determined. The next temperature is calculated by Equation (11). In Equation (15), M is calculated on the basis of moves examined by Equation (12). In this equation parameter β usually has a small value and therefore the temperature reduction proceeds slowly. $N$is the problem size.

$$T_{i+1} = \frac{T_i}{1+\beta T_i}, \quad \beta = \frac{T_0 - T_f}{MT_0T_f} \tag{11}$$

$$M = \frac{N(N-1)}{2} \tag{12}$$

Lundy and Mees, [24] used equation (13) for cooling temperature. They used the maximum iteration number N as a stopping condition.

$$T_{i+1} = \frac{T_i}{1+\beta T_i}, \quad \beta = \frac{T_0 - T_f}{T_0 T_f (N-1)} \tag{13}$$

Where β is a constant, this decreasing function is rewritten as:

$$T_{i+1} = \frac{T_i}{1+i\beta T_i}, \quad i = 1,2,...N-1 \tag{14}$$

It is clear when the temperature is too high; a lot of poor uphill moves are accepted. Conversely, when the temperature is too low the probability of falling into a local minimum is very high. Kirkpatrick and Wester [17] mentioned that between these two extremes there is a critical band of the temperatures in the whole annealing process where very slow cooling is required. Equation (11) and the procedures for deterring the initial temperature used in Connolly, [23] were designed based on this idea.

Equilibrium Test: Each SA scheme has its own means for testing the Equilibrium State, testing whether the annealing process should preceded to the next temperature. Most existing schemes test the Equilibrium State according to prescribed criteria independent of the problem characteristics.

For instance in Golden and Skiscim, [19] and Wilhelm and Ward, [21] a test as to whether the Equilibrium State has been reached is conducted after certain duration at each temperature. In both studies, this duration called an "*epoch*". In Golden and Skiscim, [19], it is represented as the *a priori* specified number of attempted moves including non-accepted moves, while in [21] it is defined as *a priori* specified number of accepted moves only. In both methods, after the execution of each epoch, the value of the objective function is calculated and the equilibrium test is conducted based on the current and previous values of the objective function. If the system reaches the Equilibrium State, then the temperature is lowered. Otherwise, exchanges are repeated during the next epoch at the same temperature and at the end of that epoch the equilibrium test is conducted again. Golden and Skiscim, [19] as shown in equation (15) find different procedures for determining the equilibrium state. If the mean value of the objective function from the most recent epoch, j, at temperature T_i which is defined as l_j^i in this equation, is sufficiently close to any of the mean values at previous epochs, i.e. $l \in \{l_o^i,...,l_{j-1}^i\}$, then the systems is assumed to be at equilibrium.

On the other hand, Wilhelm and Ward [21] uses Equation (16) as the equilibrium test, where $\bar{f}_e$ is the mean value of the objective function during the most recent epoch at temperature T_i and $\bar{f}_e$ is the grand mean of the objective function for all preceding epochs at temperature T_i.

$$\left[l_j^i - l\right] \le \varepsilon \tag{15}$$

$$\frac{\bar{f}_e - \bar{f}_e'}{\bar{f}_e'} \le \varepsilon \tag{16}$$

Kirkpatrick *et al.* [16] use the number of accepted and rejected moves as an equilibrium criterion.

Termination Criterion (Frozen Test): In general SA algorithms, the stopping criterion is specified in advance. This criterion usually depends upon the number of iterations for which an appropriate number of rejected or attempted "transitions" have taken place. It is shown that the stopping criterion has a great effect on the performance and CPU time of a SA algorithm [25].

In the proposed algorithm, the frozen state of the system is reached when either the total number of pairwise exchanges (m) is greater than maximum number of pairwise exchanges (M), or the temperature reaches the final temperature T_f.

Simulated Annealing and Neighborhood Search: In this research we investigate a pairwise exchange mechanism, where swapping two members of the sequence searches the neighbourhood. The following is an example and algorithm of how a current solution is modified the simulated annealing search.

Before Modification: ABABABCBABAAB
After Modification: CBABABABABAAB

The objective of this study is to create an annealing scheme that leads to superior performance with fewer iterations and therefore less computational time.

Proposed Algorithms: The complete proposed algorithms for the annealing process procedures mentioned above are described here.

Notation

T_0 = Initial temperature
T_{min} = Minimum value
T_{max} = Maximum value
T_f = Final temperature
T_i = Temperature at i stage
E = Value of the function
M = Number of exchange
N = Position size
a_0 = Initial function

Methods A

Step 0: Determine an initial solution, which is selected from a population of 200000 randomly generated solutions T_0 and calculateand $T_f$by equations $T_0 = \Delta f_{\min} + \frac{1}{10}(\Delta f_{\max} - \Delta f_{\min}), \quad T_f = \Delta f_{\min}$, respectively.

The temperature T is initialised to T_0 and the iteration counter m and equilibrium counter (r), are set to 0.

STEP 1: Compute the objective function, select efficiency frontier and randomly select starting point, set the temporarysolution $a^* = a^0$ and the temporary function $E = f(a_0)$. In executing SA, the temperature tuning β and M are calculated by $T_{i+1} = \frac{T_i}{1+\beta T_i}, \quad \beta = \frac{T_0 - T_f}{MT_0T_f}, \quad M = \frac{N(N-1)}{2}$, respectively.

STEP 2: Generate a feasible neighborhood search by "pairwise exchange". For pairwise exchanging, two unique products are randomly selected and exchanged. This new sequence, which obtained after exchange is referred to as the present solution and its objective value, is determined (f_p).

STEP 3: Evaluate the value of the objective function after pairwise exchange: $\Delta f = f_p - f_c$. If Δf is less than or equal zero go to step 5; otherwise go to Step 4.

STEP 4: Exchange acceptance process: (a) If $\Delta f \le 0$, then go to STEP 4b; otherwise go to STEP 4d. (b) Accept the pairwise exchange and increment the iteration counter m = m+1 and go to step 5. (c) If $\Delta f \ge 0$, then go to STEP 4d, otherwise go to step 5. (d) Compute (Metropolis) $P(\Delta f) = e^{\frac{\Delta f}{k_b T}}$ and select a uniform distribution with the range [0, 1]. If the random number less than $P(\Delta f)$, then go to STEP 4b; otherwise return to STEP 2.

STEP 5: Equilibrium test process: (a) If the value of objective function after exchange (E) is less than the best value found so far (f_p) go to step 5b; otherwise go to step 5c. (b) Change the temporary solution and if m < e go to step 5c; otherwise go to Step 2. (C) If $\frac{\left|\bar{f}_e - \bar{f}_g\right|}{\bar{f}_g} \le \varepsilon$, go to step 6; otherwise go to step 2.

STEP 6: If m, the number of pairwise exchange examined, is greater than or equal M, than Go to step 7; otherwise change the temperature according to equation (9), increment the equilibrium counter r =r + 1 and go to step 2

STEP 7: Stop.

Methods B: This method is similar to the previous one except for steps 0, 1 and 2, which become as follows:

STEP 0: Determine an initial solution, which is selected from a population of 200000 randomly generated solutions and calculate T_0 and T_f by equations (7) and (8), respectively. The temperature T is initialised to T_0 and the iteration counter m set to 0.

STEP 1: Compute the objective function, which is the initial objective function $f(a^0)$, set the temporary solution $a^* = a^0$ nd the temporary function $E = f(a_0)$. In executing SA, the cooling parameter on temperature β and M are calculated by equation (11) and (12), respectively.

STEP 2: If m, equal or less than the number of pairwise exchange go to step 7; otherwise return to step 2.

STEP 7: STOP.

Numerical Experiments

Test Method: Numerical experiments were conducted to evaluate the proposed methods. Several test problems from the literature [26] were used; see appendix A.

Table 1: Computational Results

Prob. Name	Structure	Corr.	MethodA	CPU (Minute)	Method B	CPU Min.	Set-ups	Sequence
	Simple	Low	74.99	0.4333	75	0.05	9	AAAAADAAAAAACABAAEAA
	Simple	High	10.54	0.45	10.54	0.05	9	AAABAAADAAACAEAAAAAA
A	Moderate	Low	49.85	0.4333	49.58	0.05	9	AAABAAADAAACAEAAAAAA
	Moderate	High	8.7	0.45	8.7	0.05	9	AAABAAADAAACAEAAAAAA
	Complex	Low	7.21	0.4661	7.21	0.05	9	AAAAADAAAAAACABAAEAA
	Complex	High	7.21	0.45	7.21	0.05	9	AAABAAADAAACAAAAAAEA
	Simple	Low	75.2	0.4333	75.22	0.05	11	AEACAAAAAAABADAABAAA
	Simple	High	10.54	0.4	10.22	0.05	11	AEACADAAAAABAAAABAAA
B	Moderate	Low	50.06	0.4333	50.06	0.05	11	AACAADAAABAAAABAAEAA
	Moderate	High	8.7	0.4667	8.7	0.05	11	AACAADAAABAAAABAAEAA
	Complex	Low	7.21	0.3333	7.21	0.067	11	AEACAAAAAAABADAABAAA
	Complex	High	7.21	0.4333	7.21	0.067	11	AACAADAAABAAAABAAEAA
	Simple	Low	75.58	0.4167	75.6	0.05	14	ABCAAAEABAABAAADABAA
	Simple	High	10.54	0.4167	10.54	0.05	14	AACABABAABAEADABAAAA
C	Moderate	Low	50.42	0.45	50.42	0.05	14	AABAADAABACBAABAAEAA
	Moderate	High	8.71	0.4667	8.71	0.05	15	AACABABAABAEAAABADAA
	Complex	Low	7.21	0.3167	7.21	0.067	15	AABAADAABACABABAAEAA
	Complex	High	7.21	0.4333	7.21	0.067	15	AACABABAABAEAAABADAA
	Simple	Low	75.96	0.35	75.99	0.067	8	EAAAAABBBBAAAAADDCBC
	Simple	High	10.54	0.4167	10.54	0.067	7	EAAAAABBBBAAAAADDBCC
D	Moderate	Low	50.83	0.45	50.83	0.05	8	EAAAAABBBBAAAAADDCBC
	Moderate	High	8.73	0.4	8.73	0.05	8	EAAAAABBBBAAAAADDCBC
	Complex	Low	7.21	0.45	7.21	0.05	8	EAAAAABBBBAAAAADDCBC
	Complex	High	7.21	0.4333	7.21	0.05	8	EAAAAABBBBAAAAADDCBC
	Simple	Low	76.24	0.4333	72.58	0.067	19	AEADBABADBABCBAACBAB
	Simple	High	10.54	0.4500	10.05	0.067	19	AEADBABADABBCBAACBAB
E	Moderate	Low	51.08	0.4333	51.08	0.067	19	AEADBABADBABCBAACBAB
	Moderate	High	8.73	0.4167	8.72	0.067	7	AAAAECAADDBBBBBBBCAA
	Complex	Low	7.21	0.4333	7.21	0.067	19	BADABAEABBACBABDABCA
	Complex	High	7.21		7.21	0.067	19	AEADBABADBABCBAACBAB
	Simple	Low	76.35	0.4333	76.38	0.05	9	BBDAAAACCCCBBBBCEDAA
	Simple	High	10.54	0.4333	10.54	0.05	9	BBDAAAACCCCBBBBCEDAA
F	Moderate	Low	51.22	0.4	51.22	0.067	9	BBDAAAACCCCBBBBCEDAA
	Moderate	High	8.75	0.3667	8.75	0.067	9	BBDAAAACCCCBBBBCEDAA
	Complex	Low	7.21	0.4	7.21	0.067	9	BBDAAAACCCCBBBBCEDAA
	Complex	High	7.21	0.4	7.21	0.067	9	BBDAAAACCCCBBBBCEDAA
	Simple	Low	305.83	0.45	305.92	0.05	9	AACCCCEEBBBDDDBAAACB
	Simple	High	42.15	0.45	42.15	0.05	9	AACCCCEEBBBDDDBAAACB
G	Moderate	Low	205.29	0.4333	205.29	0.067	9	AACCCCEEBBBDDDBAAACB
	Moderate	High	34.99	0.45	34.99	0.067	9	AACCCCEEBBBDDDBAAACB
	Complex	Low	28.89	0.4167	28.89	0.05	9	AACCCCEEBBBDDDBAAACB
	Complex	High	28.89	0.4	28.89	0.05	9	AACCCCEEBBBDDDBAAACB
	Simple	Low	688.34	0.4	688.5	0.05	11	BACAADCCCDDDEEEBBAAB
	Simple	High	94.81	0.45	94.81	0.05	11	BACAADCCCDDDEEEBBAAB
H	Moderate	Low	462.15	0.4	462.15	0.05	11	BACAADCCCDDDEEEBBAAB
	Moderate	High	78.72	0.4333	78.72	0.05	11	BACAADCCCDDDEEEBBAAB
	Complex	Low	65.07	0.2	65.07	0.067	11	BACAADCCCDDDEEEBBAAB
	Complex	High	65.07	0.3167	67.07	0.05	11	BACAADCCCDDDEEEBBAAB
	Simple	Low	1225.71	0.4333	1225.98	0.05	10	BBBBEDDADEEECAAADCCC
	Simple	High	168.51	0.4	168.51	0.05	10	BBBBEDDADEEECAAADCCC
I	Moderate	Low	823.13	0.4167	462.15	0.067	10	BBBBEDDADEEECAAADCCC
	Moderate	High	139.95	0.4333	78.72	0.067	10	BBBBEDDADEEECAAADCCC
	Complex	Low	115.71	0.4167	115.71	0.05	10	BBBBEDDADEEECAAADCCC
	Complex	High	115.71	0.45	115.71	0.05	10	BBBBEDDADEEECAAADCCC

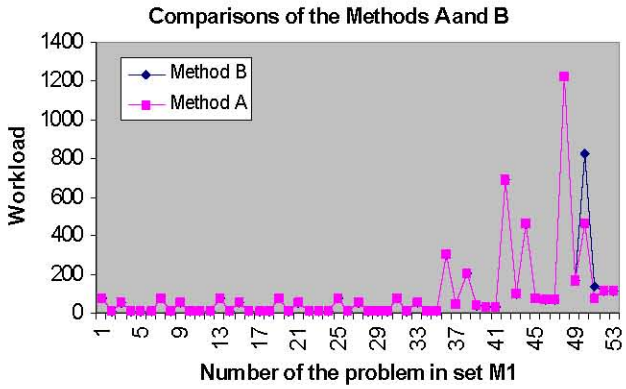

Fig. 2: Comparisons of Methods A and B

Table 2: Result summary.

Problem Set M1		
Problem name	Average MSD	Average difference
A	0.55	0.0
B	0.59	0.0
C	0.75	0.0
D	0.49	0.10
E	0.53	0.0006
F	0.57	0.0577
G	0.63	0.11
H	0.69	0.02
I	0.80	0.0
Problem Set M2		
Problem name	Average MSD	Average difference
A	1.37	0.0
B	1.52	0.0179
C	1.54	0.02
D	1.58	0.0
E	1.42	0.0
F	1.52	0.0
G	1.61	0.0
H	1.65	0.0
I	1.65	0.0

These problem sets, M1, M2 M3 were solved by proposed methods to find the best objective function and the best method is compared with the existing solutions. The algorithms are coded in Visual C++ and run on a Pentium 3 compatible PC. During the experimentation, CPU time was taken in minutes.

Previous studies indicated that such factors as setting the parameters and computational time have a great impact on the performance of a search algorithm [27,28].

Concerning parameter settings, Connolly's scheme was employed in our method B. Method A, the tuning parameter for temperature a was set at 0.97 from the results of preliminary experiments.

Computational Results: To evaluate the problem of minimizing deviations of actual workload from the ideal workload, a problem set M1 from appendix A and the assembly times required at each station for each of the three model structures are used to test each method.

The results of these tests are broken by the methods and are shown in Table 1. The results of our computational testing in Table 1 and Figure 2 have shown the Method B finds the better results in terms of CPU times for this problem set.

The following average measures over nine problems set are calculated for each solution approach (see Table 1):

Mean Squared Deviation (MSD) $= \sum_{k=1}^{D_T} \sum_{m=1}^{M} \sum_{s=1}^{S} \left(p_{m,s} x_{m,k} - \gamma k \left(\frac{t_{m,s}}{T_p} \right) \right)^2$, % difference in the objective function, CPU time,

CONCLUSIONS

In this research, a new simulated annealing algorithm is developed to obtain optimal solutions to multiple objective sequencing problems in mixed-model assembly lines. We have considered two objectives: workload smoothing and minimizing total set-up costs. These are important for an efficient operation of mixed-model assembly lines. Methods are work efficiently and find good solutions in a very short time, even when the size of the problem is too large. Mathematical formulations for the two objectives are provided.

Based on the above concepts, we proposed two methods, which differed only in cooling schedule and hence affected in computational time. In the first method, method B, information obtained during trial prior to the annealing process is utilized and the system appropriate for relatively small size problem.

On the other hand, the second method, method A uses a simpler procedure for temperature tuning, i.e. it determines the next temperature according to a prescribed cooling function. This function used in the present study cools faster and thereby contributes to saving computational time.

In order to evaluate the proposed methods, we conducted a number of numerical examples using the standard problems of Sumichrast, *et al.* [28]. Based on the results, method B was shown to be obtaining higher quality solution than method A and in most cases similar to optimal solutions for all productions size. This method works well for lager problems.

Finally, although the simulated annealing algorithms proposed here is for the mixed-model assembly production line in a JIT system, it is possible to apply these methods to other combinatorial optimization problems by finding rules or structure based on the characteristics of the problem. The algorithm developed provides higher quality solutions with increasing efficiency in comparisons with other heuristics methods available in the literature. We can concluded from the results of this study that it is worthwhile to take the workload-smoothing goal into account especially in cases where the assembly line is not long and/or variability in the processing times of the models is high.

ACKNOWLEDGMENTS

The author wishes to express his gratitude to Dr. Patrick R. McMullen, Wake Forest University, USA for providing results of the benchmark algorithms.

Appendix

Problem Set M1, Total production $D_T = 20$, n=5

Problem name	d_1	d_2	d_3	d_4	d_5
A	16	1	1	1	1
B	15	2	1	1	1
C	13	4	1	1	1
D	10	5	2	2	1
E	8	7	2	2	1
F	6	6	5	2	1
G	5	5	5	3	2
H	5	4	4	4	3
I	4	4	4	4	4

Problem Set M2, Total production $D_T = 20$, n=10

Problem name	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}
A	11	1	1	1	1	1	1	1	1	1
B	10	2	1	1	1	1	1	1	1	1
C	9	3	1	1	1	1	1	1	1	1
D	8	4	1	1	1	1	1	1	1	1
E	7	5	1	1	1	1	1	1	1	1
F	6	5	2	1	1	1	1	1	1	1
G	5	5	3	1	1	1	1	1	1	1
H	4	4	4	2	1	1	1	1	1	1
I	2	2	2	2	2	2	2	2	2	2

Problem SetM3, Total production $D_T = 20$,,n=15

Problem name	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{11}	d_{12}	d_{13}	d_{14}	d_{15}
A	40	40	8	1	1	1	1	1	1	1	1	1	1	1	1
B	35	35	10	5	5	1	1	1	1	1	1	1	1	1	1
C	30	30	15	10	5	1	1	1	1	1	1	1	1	1	1
D	25	25	20	15	5	1	1	1	1	1	1	1	1	1	1
E	20	20	20	10	10	1	1	1	1	1	1	1	1	1	1
F	20	20	15	15	10	6	6	1	1	1	1	1	1	1	1
G	15	15	15	10	10	10	10	5	4	1	1	1	1	1	1
H	15	15	10	10	10	10	10	10	4	1	1	1	1	1	1
I	7	7	7	7	7	7	7	7	7	7	6	6	6	6	6

Assembly time required by station and model: Simple structure. Low Correlation

	Station									
Model	1	2	3	4	5	6	7	8	9	10
1	6.41	0	0	0	1.28	3.85	7.69	1.28	3.85	0
2	5.13	6.41	2.56	2.56	1.28	7.69	8.97	2.56	3.85	6.41
3	2.56	1.28	6.41	2.56	1.28	0	7.69	6.41	2.56	7.69
4	1.28	2.56	1.28	4.91	1.28	0	5.13	7.69	7.69	7.69
5	7.69	1.28	7.69	0	3.85	1.28	7.69	6.14	3.85	8.97

Assembly time required by station and model: Simple structure. High Correlation

	Station									
Model	1	2	3	4	5	6	7	8	9	10
1	6.67	3.33	3.33	3.33	3.33	6.67	3.33	3.33	3.33	3.33
2	3.33	6.67	3.33	3.33	3.33	3.33	6.67	3.33	3.33	3.33
3	3.33	3.33	6.67	3.33	3.33	3.33	3.33	6.67	3.33	3.33
4	3.33	3.33	3.33	6.67	3.33	3.33	3.33	3.33	6.67	3.33
5	3.33	3.33	3.33	3.33	6.67	3.33	3.33	3.33	3.33	6.67

Assembly time required by station and model: Moderate structure. Low Correlation

	Station									
Model	1	2	3	4	5	6	7	8	9	10
1	2.47	5.56	1.23	0.62	2.47	4.78	0.93	2.78	6.17	2.78
2	1.39	2.78	3.55	4.46	4.63	2.16	5.25	2.16	0.77	1.23
3	6.95	2.16	3.40	2.78	5.56	3.86	4.48	3.24	1.54	2.93
4	4.94	2.47	2.16	2.93	3.24	4.63	2.31	3.86	4.94	6.95
5	3.24	3.40	6.77	3.09	1.85	6.48	4.17	3.86	5.09	3.86

Assembly time required by station and model: Moderate structure. High Correlation

	Station									
Model	1	2	3	4	5	6	7	8	9	10
1	2.35	4.06	2.35	1.50	1.50	4.49	2.35	4.06	4.92	1.50
2	1.93	2.35	2.78	4.06	3.21	3.21	4.06	1.50	1.93	1.50
3	5.34	2.35	2.35	1.50	4.92	3.37	5.34	1.93	1.50	2.78
4	4.06	1.50	3.21	1.93	4.49	3.21	3.37	2.78	4.92	5.34
5	3.37	4.06	1.93	2.35	2.35	4.92	3.37	2.78	3.37	2.78

Assembly time required by station and model: Complex structure. Low Correlation

	Station									
Model	1	2	3	4	5	6	7	8	9	10
1	4.49	0	1.23	1.23	1.23	2.45	7.36	1.23	3.68	0
2	4.49	26.14	2.45	2.45	1.23	8.59	7.36	2.45	3.68	6.14
3	2.45	2.45	4.91	2.45	2.45	1.23	8.59	4.91	2.45	7.36
4	3.68	2.45	2.45	4.91	2.45	1.23	6.14	7.36	7.36	7.36
5	7.36	1.23	7.36	1.23	2.45	2.45	7.36	6.14	3.68	7.36

Assembly time required by station and model: Complex structure. High Correlation

	Station									
Model	1	2	3	4	5	6	7	8	9	10
1	2.35	4.06	2.35	1.50	1.50	4.49	2.35	4.06	4.92	1.50
2	1.93	2.35	2.78	4.06	3.21	3.21	4.06	1.50	1.93	1.50
3	5.34	2.35	2.35	1.50	4.92	3.37	5.34	1.93	1.50	2.78
4	4.06	1.50	3.21	1.93	4.49	3.21	3.37	2.78	4.92	5.34
5	3.37	4.06	1.93	2.35	2.35	4.92	3.37	2.78	3.37	2.78

REFERENCES

1. Monden, Y., 1983. Toyota production system" (Norcross, GA: Institute of Industrial Engineers Press.
2. Bolat, A. and C. Yano, 1992. Scheduling algorithms to minimize utility work at a single station on a paced assembly line. Production Planning and Control., 3: 393-405.
3. Okamura, K. and H. Yamashina, 1979. A heuristic algorithm for the assembly line model-mix sequencing problem to minimize the risk of stopping the Conveyor. International J. Production Res., 17: 233-247.
4. Xiaobo, Z. and K. Ohno, 1994. A sequencing problem for mixed-model assembly line in a just-in-time production system. Computers and Industrial Engineering, 27: 71-74.
5. Miltenburg, G.J., G. Steiner and S. Yeomans, 1990. A dynamic programming algorithm for scheduling mixed-model JIT production systems. Mathematical and Computer Modelling, 13:57-66.
6. Inman, R.R. and R.L. Bulfin, 1991. " Sequencing just-in-time mixed-model assembly lines. Management Sci., 37: 901-904.
7. Miltenburg, G.J., 1989. Level schedules for mixed-model assembly lines in Just-In-time production system. Management Sci., 35: 192-207.
8. Kubiak, W. and S. Sethi, 1991. Level schedules for mixed-model assembly lines in just-in-time production systems. Management Sci., 37: 121- 122.
9. Steiner, G. and S. Yeomans, 1993. Level Schedules for mixed model just-in-timeprocesses, Management Sci., 39: 728-735.
10. McMullen, P.R., 2001. A kohonen self organizing map approach to addressing of Production Economics, 72: 59-71.
11. Mohammadi, G. and M. Ozbayrak, 2006, Scheduling mixed-model final Assembly Lines in JIT manufacturing. International J. Computer Integrated Manufacturing, 19: 377-382.
12. Meral, S. and T. Krokmazel, 2000. Bicriteria sequencing methods for the Mixed-mode assembly line in JIT production systems, European J. Operational Res., 131: 188-207.
13. Burns, L.D. and C.F. Daganzo, 1987. Assembly line job sequencing principle.INT. J. Production Res., 25: 71-99.
14. Metropolis,N.,A.Rosenbluth,M.Rosenbluth, A. Teller and E. Teller, 1953. Equation of State Calculations by Fast Computing Machines. J. Chemical Physics, 21: 1087-1092.
15. Kirkpatrick, S., C.D. Gelatt, Jr. and M.P. Vecchi, 1983, Optimization by Simulated Annealing Sci., 220: 671-680.
16. Kirkpatrick, S., C.D. Gelatt, JR. and M.P. Vecchi, 1982, Optimization by Simulated Annealing", IBM Computer Science/Engineering Technology Report, IBM Thomas J. Watson Research Centre, Yorktown Heights, NY.,
17. Kilbridge, M.D. and L. Wester, 1963. The assembly line model-mix sequencing problem. Proc. of the third int. conf. op. Res. English universities press (Paris: Dunod Editor).
18. Cheh, K.M., J.B. Goldberg and R.G. Askin, 1991. A note on the effect of neighbourhood structure in simulated annealing. Computers Operation Res., 14: 537-547.

19. Golden, B. and C. Skiscim, 1986, Using simulated annealing to solve routing and location problems. Nav. Res. Q., 33: 261-279.
20. Burkard, R.E. and F. Rendle, 1989. A thermodynamically motivated simulation procedure for combinatorial optimization problems. European J. Operational Res., 17: 169-174.
21. Wilhelm, M.R. and T.L. Ward, 1987. Solving quadratic assignment problem by simulated annealing. IEE Transactions, 1: 107-119.
22. Hall, R., 1983. Zero inventories. Dow Jones-Irwin, Homewood, IL.
23. Connolly, D.T., 1990. An improved Annealing scheme for the QAP, European J. Operational Res., 46: 93-100.
24. Lundy, M. and A. Mees, 1986. Convergence of annealingalgorithm. Mathematical Programming, 34: 11-124.
25. Kouvelis, P. and W. Chiand, 1992. A simulated annealing procedure for single row layout problems in flexible manufacturing systems. International J. Production Res., 30: 717-732.
26. Sumichrast, R.T. and R.S. Russel, 1990. Evaluating mixed-model assembly line sequencing heuristics for just-in-time production systems. J. Operations Manage., 9: 371-390.
27. Jain, A.S. and S. Meeran, 1999. Deterministic job-shop scheduling: Past, present and Future. European J. Operational Res., 113: 390-434.
28. Nowicki, E. and C. Smutnicki, 1996. A fast tabu search algorithm for the permutation flow shop problem. European J. Operational Res., 91: 160-175.