

Using simulated annealing to solve a multiobjective assembly line balancing problem with parallel workstations

P. R. McMULLEN^{†*} and G. V. FRAZIER[‡]

This research presents a Simulated Annealing based technique to address the assembly line balancing problem for multiple objective problems when paralleling of workstations is permitted. The Simulated Annealing methodology is used for 23 line balancing strategies across seven problems. The resulting performance of each solution was studied through a simulation experiment. Many of the problems consisted of multiple products, which were sequenced in a mixed model fashion, task times were assumed to be stochastic, and parallel workstations were permitted. Two primary performance objectives were of most interest: total cost (labour and equipment) per part, and the degree to which the desired cycle time was achieved. Other traditional line balancing and production performance measures were also collected. This paper demonstrates how Simulated Annealing can be used to obtain line balancing solutions when one or more objectives are important. The experimental results showed that Simulated Annealing approaches yielded significantly better solutions on cycle time performance but average solutions on cost performance. When cycle time performance and total unit cost are weighted equally, performance rankings showed that Simulated Annealing approaches still showed better mean performance than the other approaches.

1. Introduction and literature review

1.1. Background

Assembly line balancing is an area of research that has received relatively little attention in recent years. With the number of simplifying assumptions that most traditional line balancing approaches make, it is unsurprising that production managers today are often reluctant to use these old approaches. Modern production environments are often fast paced and flexible, and an increasing number of companies are adopting JIT techniques. In these new environments, many traditional line balancing approaches may lead to poor solutions. The motivation for this research was to investigate the problem of balancing a production or assembly line in a realistic, modern manufacturing environment.

Several common assumptions for line balancing problems were relaxed in this research to reflect a production environment that might exist in a successful company today. First, the assumption of deterministic task times was relaxed. Since people commonly perform many tasks in a production line, task times are assumed to be somewhat stochastic to reflect reality better. Second, the assumption of a single objective was relaxed. Because individual task times are uncertain, the resulting cycle

Received March 1998.

[†] University of Maine, The Maine Business School, Orono, ME 04469-5723, USA.

[‡] The University of Texas at Arlington, College of Business Administration, Department of Information Systems and Management Sciences, Arlington, TX 76019, USA.

* To whom correspondence should be addressed.

time is uncertain. Two primary objectives were then emphasized in the study, the extent to which the desired cycle time was achieved and the total design cost of the production line (equipment cost and labour cost).

A third common assumption that was relaxed was the limit of one worker per workstation. As competitive pressures push companies to shorten lead times and increase efficiencies, higher output levels of production lines are desired (shorter cycle times). As the cycle time decreases, it may become shorter than the longest task duration at some point. The most effective solution is usually to duplicate the workstation so that two or more workers are performing the same set of tasks in parallel. The use of parallel workstations also provides greater flexibility in designing the production line. When parallel workstations are allowed, the additional cost of equipment then becomes important. Managers are interested in minimizing the total design cost of the line (both labour cost and equipment cost).

Finally, the common assumption of a single product type was relaxed. As manufacturing flexibility becomes more important, multiple product production lines are more common. Both single product lines and multiple product lines are considered in this study. Further, since a common JIT technique is mixed-model product sequencing, this approach was used here. It should be noted that the relaxation of these assumptions is intended to reflect a more realistic manufacturing environment.

1.2. *Simulated annealing*

This study evaluated many line balancing strategies under this new environment, both existing and new strategies. One new approach involves Simulated Annealing to obtain good line balancing solutions. Simulated Annealing is a search technique that is useful for solving combinatorial optimization problems. It attempts to avoid being trapped at local optima in its search for the global optima. Simulated Annealing gets its name from the physical annealing of solids—heating a solid to a very high temperature and then cooling it at a slow rate, spending a relatively large amount of time near the freezing point of the solid. When this heating and subsequent slow cooling occur, the particles within the solid rearrange themselves. The purpose of annealing is for the particles to arrange themselves in such a way that the solid possesses some desired attribute, such as high strength or surface hardness.

Simulated Annealing has been adapted as a search-based technique to solve combinatorial optimization problems. It has become a popular tool for solving problems where mathematical programming formulations become intractable. Analogous to its use with the physical annealing of solids, the combinatorial optimization problem solution undergoes a series of changes while looking for an improved solution (according to some objective function). As Simulated Annealing starts, an initial solution is generated and used as the first current solution. A control parameter, T , is specified which is analogous to the annealing temperature. This ‘temperature’ is systematically decreased, or ‘cooled’ according to a cooling rate, CR . As the temperature drops, neighbouring solutions to the current solution are found. If the objective function value is superior to that of the current solution, the neighbouring solution becomes the new current solution. If the neighbouring solution provides an objective function value inferior to that of the current solution, the neighbouring solution may still become the current solution if a certain acceptance criterion is met. A distinctive feature of Simulated Annealing is that inferior solutions are sometimes accepted as the current solution to try and prevent getting trapped at local optima. Through the occasional acceptance of infer-

ior solutions which meet the acceptance criteria, the search moves to a different location on the continuum of feasible solutions in an effort to reach the global optimum. The process of finding neighbouring solutions and accepting these as current solutions if acceptance criteria are met is repeated according to the cooling pattern until some stopping criteria is met. Kirkpatrick *et al.* (1983), Eglese (1990) and Goldberg (1989) provide further basic descriptions of Simulated Annealing, in addition to informative examples.

Suresh and Sahu (1994) developed one Simulated Annealing approach for line balancing. They considered the problem of stochastic task durations but did not allow parallel workstations or consider multiple products. Also, they only considered single objective problems. For one set of problems they minimized a smoothness index used by Moodie and Young (1965). For a second comparison the authors minimized the probability of line stopping, used by Reeve (1971). Suresh and Sahu found that Simulated Annealing performed at least as well as the approaches by Moodie and Young and by Reeve. They also mentioned future plans to incorporate multiple objectives into their Simulated Annealing approach.

1.3. Description of research

Aside from the primary objectives of cycle time performance (percentage of desired cycle time actually achieved) and design cost performance (sum of total labour cost and total equipment cost), several other performance measures were also examined in this study. Traditional line balancing performance measures and production performance measures included average work in process inventory level (WIP), average flow time per unit, average throughput, average utilization of system resources, percentage of parts at each workstation completed within the desired cycle time, and average unit labour cost. While the authors consider these measures to be of secondary interest but not primary design objectives, some companies might wish to optimize one or more of these measures instead. In order to evaluate line balancing approaches using these performance measures a production simulation experiment was performed.

To summarize, this paper presents a Simulated Annealing approach to solve assembly line balancing problems and then describes a simulation experiment comparing 23 line balancing strategies or approaches. The type of line balancing problem addressed consists of stochastic task times, multiple product types scheduled in mixed-model fashion, the allowance of parallel workstations, and the use of multiple objectives. The two primary objectives were the extent to which the desired average cycle time was achieved, and the total design cost of labour and equipment. Performance of the different rules was also evaluated using several other measures as well. The following sections provide background information on line balancing with parallel workstations and on Simulated Annealing, describe the research methodology of this study, present the experimental design and results, and offer conclusions based on the research results.

2. Assembly line balancing and parallel workstations

Assembly line balancing is the practice of placing definable units of work (referred to as tasks) into cohesive groups (referred to as work centres) in accordance with some goal. This goal typically takes on one of two forms. The first is minimization of workers required, known as a Type I problem, and the second is minimization of the amount of time elapsing between the completion of two consecutive jobs

(cycle time), known as a Type II problem. The research presented here is closest to the Type I problem since the desired cycle time is assumed to be specified by management.

Regardless of the type, assembly line balancing problems have some general constraints that apply here:

- a task will not be assigned to a work centre until all of its predecessors have been assigned;
- a task can be assigned to only one work centre;
- all workers on the assembly line possess the same level of skill;
- all tasks are independent of each other;
- changeover times between different products are negligible.

Specific to this research is the subject of paralleling of tasks within work centres. The vast majority of assembly line balancing literature addresses the case where only one worker is permitted to occupy a work centre (non-paralleling). With this scenario, tasks can be placed into a work centre provided that the total time required by the tasks does not exceed the specified cycle time. Figure 1 shows an example of non-paralleling. In this example, tasks 3, 5, 6 and 9 will be performed by a single worker in work centre 2, and the total duration of all four tasks will be less than the specified cycle time. (To clarify the language used from here on, each worker has a workstation where he or she can perform one or more tasks. Each task requires a different piece of equipment. Each work centre consists of either one workstation for the case of non-paralleling or multiple parallel workstations.) Since four tasks are performed by a single worker, it is assumed that four pieces of equipment are required.

When the sum of task durations in a work centre exceeds the specified cycle time, either one or more tasks must be removed from the work centre, or else duplicate workstations (and workers) can be included in the work centre. This latter alternative is referred to as paralleling, where the same task set is assigned to each workstation within the work centre. The amount of paralleling necessary is dependent upon the ratio of total task durations within the work centre to the cycle time. Figure 2 shows an example of three parallel workstations within a work centre. The same four tasks occupy the work centre as in figures 1–3, 5, 6 and 9. In this case, however, three workers each perform all four tasks in parallel. Also, since there are three workers in this work centre and four tasks for each worker, it is assumed that 12 pieces of equipment are required. For additional discussions of paralleling of workstations within work centres see Pinto *et al.* (1975, 1978, 1981).

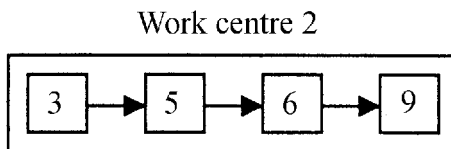


Figure 1. Examples of non-paralleling.

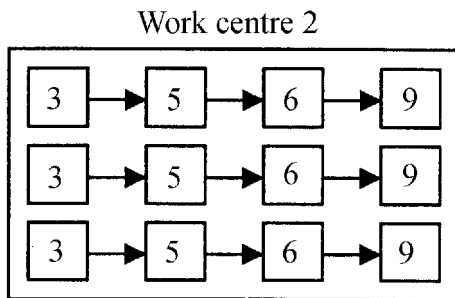


Figure 2. Graphical example of paralleling.

3. Methodology

3.1. Assembly line balancing and mixed-model problems

The general assembly line balancing heuristic used here to generate initial solutions is based upon Gaither's Incremental Utilization Heuristic (1996), with modifications (McMullen, 1995). Gaither's text has detailed examples. It should be noted that any line balancing heuristic could be used to obtain an initial feasible solution. In the event of mixed-model problems, composite durations for each task are computed using weighted averages. The weights used are proportional to the contribution to product-mix of products involved in the mixed-model problem. In the event of single-product models, the given task durations are used directly for composite task durations and weighted averages are not needed. Regardless of whether mixed-models or single-product models are used, the composite task durations are used as the task durations. The line balancing is then performed according to the Gaither heuristic.

3.2. Objective functions

During the iterative Simulated Annealing process, the user specifies an objective function—a goal, so to speak. A variety of objective functions were investigated to use with the Simulated Annealing approach. A list of variables is summarized first, followed by descriptions of the objective functions and a description of the Simulated Annealing steps.

- E_i value of objective function i (chosen objective),
- r total number of work centres,
- t_i duration of task i (which is assigned to work centre j)
- $\hat{\sigma}_i$ estimated standard deviation of task i ,
- $\hat{\sigma}_j$ estimated standard deviation of work centre j ,
- C desired cycle time (as specified by management),
- cv coefficient of variation,
- w_j number of workers required in work centre j ,
- \hat{w}_j integer-adjusted workers required in work centre j ,
- q_j number of tasks in work centre j ,
- m_j number of pieces of equipment required in work centre j ,
- L labour cost per worker in \$/year,
- Q equipment cost per piece in \$/year,
- p_j probability of lateness in work centre j .

Prior to the presentation of the objective functions, it should be noted that objective functions 1, 2 and 3 address a single objective, while objective functions 4, 5 and 6 address multiple objectives—both design cost and lateness.

3.2.1. Minimize design cost (objective function 1)

The objective function presented below is to minimize the total cost associated with the assembly line balancing solution. Specifically, this objective is concerned with minimizing the sum of costs associated with both labour requirement and equipment requirement. The objective function is

$$\text{Min: } E_1 \sum_{j=1}^r (\hat{w}_j L + m_j Q). \quad (1)$$

The number of workers required in work centre j is the ratio of assigned work (the sum of task durations) in work centre j divided by the specified cycle time. This value is rounded upward to the nearest integer value if the ratio is not an integer. If the ratio takes on an integer value, w_j is not adjusted to \hat{w}_j . The logic for this is that ‘whole’ workers must be assigned to a work centre. Mathematically, this is

$$w_j = \left\lceil \frac{1}{C} \left(\sum_{i=1, i \in j}^n t_i \right) \right\rceil, \quad (2)$$

and

$$\hat{w}_j = 1 + \text{int}(w_j). \quad (3)$$

The number of machines (or pieces of equipment) required for work centre j is the product of the number of workers in work centre j and the number of tasks in work centre j . Hence

$$m_j = \hat{w}_j q_j. \quad (4)$$

For example, figure 2 shows that three workers are required for work centre 2 ($\hat{w}_2 = 3$), and four tasks are assigned to the work centre ($q_2 = 4$). This means that 12 machines are required for this work centre ($m_2 = 3 \times 4 = 12$).

3.2.2. Minimize smoothness index (objective function 2)

The intent of the objective function presented below is to minimize the ‘smoothness’ across all work centres. This objective function is basically the paralleling equivalent to the ‘smoothness index’ as presented by Moodie and Young (1965) and also used by Suresh and Sahu (1994), which did not permit for paralleling of tasks. This ‘smoothness index’ is intended to distribute work into the work centres as evenly as possible. The ‘smoothness index’ for the paralleling case is as follows:

$$\text{Min: } E_2 = \sqrt{\sum_{j=1}^r (\hat{w}_j - w_j)^2}. \quad (5)$$

3.2.3. Minimize probability of lateness (objective function 3)

When the stochastic nature of task durations is considered, there is a possibility that the actual task duration could exceed its expected duration, which implies the possibility of lateness (Silverman and Carter 1984, 1986). This particular objective

function is to minimize lateness across all work centres. Consider a group of tasks that have been assigned to work centre j . In this study a fixed coefficient of variation is assumed. The estimated standard deviation for a task is computed from the expected task duration and coefficient of variation, cv , as

$$\hat{\sigma}_i = cv(t_i). \quad (6)$$

Based upon the expected total duration of all tasks within work centre j , the expected number of workers is \hat{w}_j and the actual number of workers used is w_j . The expected number of workers, w_j , can be thought of as the expected value of a normally distributed random variable with a standard deviation of:

$$\hat{\sigma}_j = \sqrt{\sum_{i=1, j \notin i}^r \left(\frac{\hat{\sigma}_i}{C}\right)^2}. \quad (7)$$

If the actual amount of work assigned exceeds the capacity of the number of workers assigned, lateness will occur. This probability of lateness can be estimated by integrating the normal distribution function as follows:

$$p_j = \frac{1}{\sqrt{2\pi}} \int_Y^{\infty} e^{-.5z^2} dz, \quad (8)$$

where Y is the normalized critical value:

$$Y = \frac{\hat{w}_j - w_j}{\hat{\sigma}_j}. \quad (9)$$

With all of the supporting parameters presented, the objective to minimize lateness across all work centres can now be defined:

$$\text{Min: } E_3 = \prod_{j=1}^r p_j. \quad (10)$$

The objective function here attempts to generate layouts successful in completing jobs in a timely fashion. The objective function, E_3 , is a proxy for lateness of the entire assembly line, obtained by multiplying the lateness measure of all work centres involved in the layout of interest.

It should be noted that whenever the minimize lateness criteria is used, normal rules of integration cannot be used to approximate p_j . As a result, numerical integration is used—the trapezoidal rule with 500 subintervals.

3.2.4. Minimize composite function (objective functions 4, 5, and 6)

As is usually the case when multiple objectives are important, some trade-off among the objectives can be expected. For example, a solution which attempts to minimize cost, E_1 , would probably result in a solution which performs poorly with regard to system lateness, E_3 . Because of such trade-offs, three composite functions are presented which attempt to minimize the weighted sum of two previous objective functions. The first composite objective function, E_4 , is

$$\text{Min: } E_4 = E_1 + 1.67E_3. \quad (11)$$

The normalizing weights were determined by sampling many solutions for different problems in terms of E_1 and E_3 , so that each component of this objective function

would make an equal contribution, on average, to the value of E_4 . For each of the seven problems, 50 Simulated Annealing based solutions were sampled and average values of E_1 and E_3 computed. Coefficients were then determined such that for the average size problem both E_1 and E_3 make similar contributions to the value of E_4 . Both coefficients were then divided by the E_1 coefficient, so that the E_1 coefficient is unity—this procedure resulted in the E_3 coefficient being 1.67.

The second composite objective function, E_5 , changes the weighting of E_3 so that the probability of lateness makes three times the contribution to E_5 on average as does design cost. The objective function is

$$\text{Min: } E_5 = E_1 + 3 \times 1.67E_3. \quad (12)$$

The last composite objective function, E_6 , does the same as the second, except that the weights are reversed so that design cost makes three times the contribution to E_6 as does the probability of lateness. The objective function is

$$\text{Min: } E_6 = 3 \times E_1 + 1.67E_3. \quad (13)$$

The choice of relative weights used for E_4 , E_5 and E_6 is admittedly arbitrary. The purpose here is to show that a decision-maker can select any set of relative weights desired for different objectives to reflect best the relative importance of each objective. The authors are not suggesting that a particular set of weights is best, but rather are demonstrating that different weighting schemes for multiple objectives can be easily incorporated into the Simulated Annealing line balancing heuristic.

Also, objective function E_2 was not included in any composite objective functions because preliminary results showed E_1 and E_3 to have better overall performance than E_2 . E_2 was included in the experiment only because Moodie and Young (1965) and Suresh and Sahu (1994) used a similar 'smoothness index' in their research, and the authors here wanted to evaluate the performance of such an objective function.

3.3. Simulated annealing heuristic

The Simulated Annealing heuristic consists of eight algorithmic steps described below. Following the description is a flowchart that graphically shows the logic of this approach.

- Step 1.* Initialize the model by specifying a cycle time (C), a control parameter (T), a cooling rate (CR), a number of iterations for each level of T (N_{\max}), and a stopping criteria (T_{\min}). Also generate an initial feasible solution for the problem, and choose an objective function for optimization. This initial solution becomes the first 'current' solution and the first 'best' solution used for the search technique. Calculate the objective function value for this initial solution. The objective function value for the current solution will be referred to as E_c , and the objective function value for the best solution will be referred to as E_b .
- Step 2.* From the current solution, generate a feasible neighbouring solution. This is done via a trade or a transfer. A trade is first considered, but if a trade is not possible, then a transfer is considered. Regardless of the strategy used to generate a neighbouring solution, crew-size, equipment requirement, and other relevant measures must be re-calculated to determine the result of the trade in terms of the objective function. One of the following options is then taken.

Option (a): Trade. Trade tasks between two adjacent work centres. A work centre is selected randomly, and it is determined whether or not the final task within this work centre (referred to as task X) can be exchanged with the first task in the next work centre (referred to as task Y). This determination is made by examining the precedence relationship between the two tasks in question. If the precedence relationship shows that task Y is permitted to precede task X, the trade is considered feasible. Otherwise, the trade is not feasible and option (b) is explored. Figures 3 and 4 show an example of a trade. Suppose work centre 6 is randomly selected. It is determined that task 22 can precede task 20, so the trade is executed. Figure 4 shows the results of the trade. This example assumes that the same number of workers are still required in each work centre—five workers and 16 pieces of equipment are still required, but the smoothness index and probability of lateness will most likely have changed as the result of the trade. Proceed to step 3.

Option (b): Transfer. Transfer a task from one work centre to another, and update the number of workers and pieces of equipment required as a result of the transfer. A work centre is selected at random, and the last task in this work centre is transferred to the next work centre as the first task. This will be referred to as a 'regular' transfer. As an example, refer to the scenario in figure 3. Work centre 6 has been randomly selected for a transfer. Task 20 will be transferred to work centre 7. Figure 5 shows the result of this type of transfer. In terms of the objective function, five workers are still required.

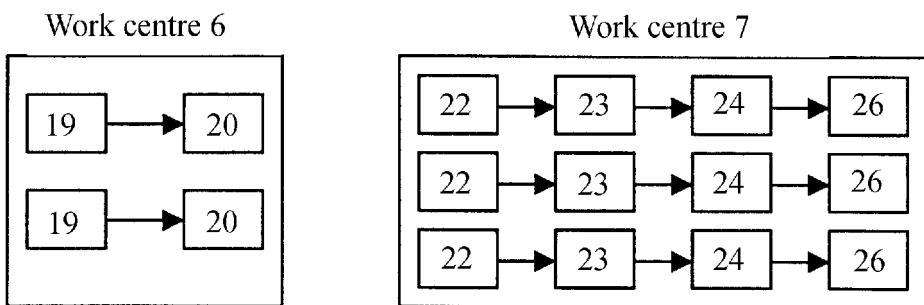


Figure 3. Work centres 6 and 7 prior to a trade.

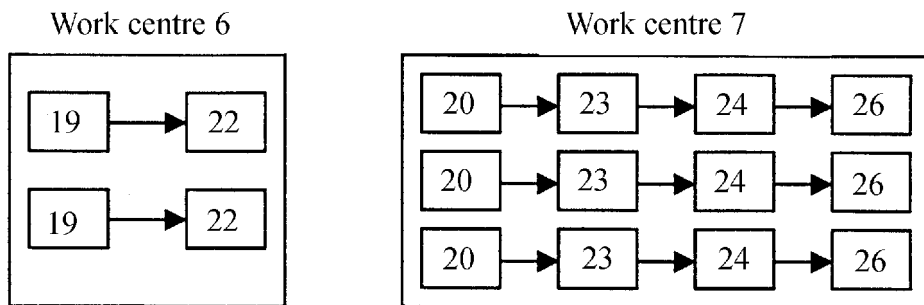
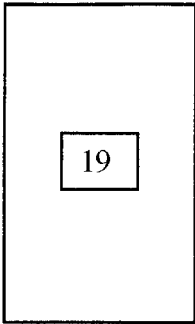


Figure 4. Work centres 6 and 7 after a trade.

Work centre 6



Work centre 7

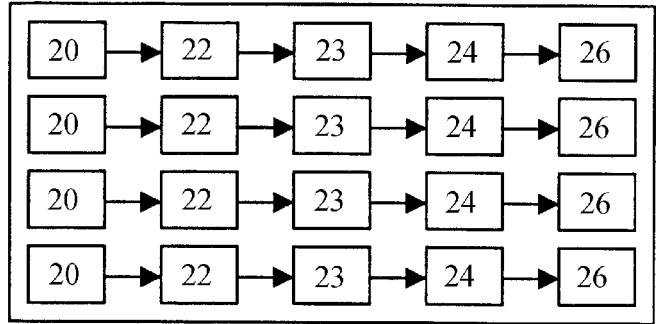


Figure 5. Graphical example of work centres 6 and 7 after a regular transfer.

The transfer results in five additional required pieces of equipment, which will likely result in changes in the objective function value ($21 - 16 = 5$).

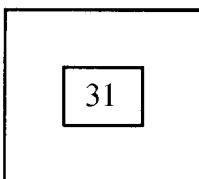
A regular transfer is not always possible. Consider the situation shown in figure 6. If work centre 9 is randomly selected for a transfer, the lone task in the work centre would be transferred to work centre 10, leaving work centre 9 empty.

Instead of leaving an empty work centre, both work centres will be 'fused' into one. This is referred to as a 'compression' transfer, which results in the same number of workers. Figure 7 shows the result of work centres 9 and 10 undergoing a compression transfer. For this example, the objective function would reflect the same number of workers and five more pieces of equipment ($12 - 7 = 5$) as a result of the compression transfer.

When the final work centre of a line balancing solution is randomly selected for a transfer, special measures must also be taken. Figure 8 shows an example of the final work centre for a line balancing solution. In this situation, transferring the final task out of the work centre (Task 40) would result in the creation of a new work centre.

This is referred to as an 'expansion' transfer—the final task in the final work centre becomes the lone task in a new work centre. Figure 9 shows the result for this example. The expansion transfer in this example results in a

Work centre 9



Work centre 10

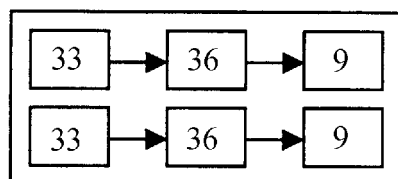


Figure 6. Graphical example of work centres 9 and 10 prior to a transfer.

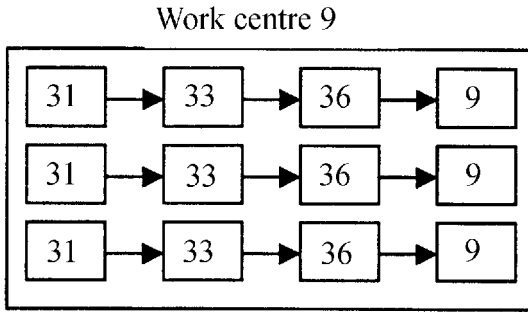


Figure 7. Example of a compression transfer.

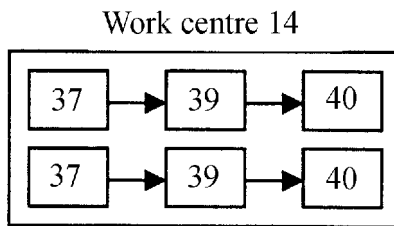


Figure 8. Final work centre prior to transfer.

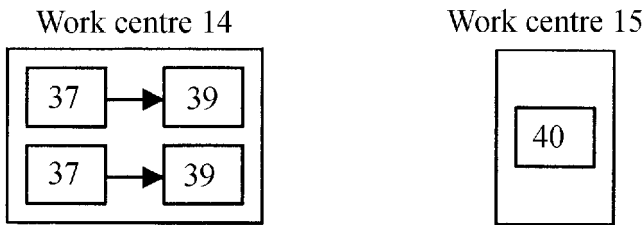


Figure 9. Result of an expansion transfer.

solution with one additional worker and one less piece of equipment. Whichever type of transfer is performed, proceed to step 3. (In the case where the last work cell is selected for a transfer and that cell only has one task, a compression transfer would not be attempted and the algorithm would proceed to step 6.)

If a trade or transfer has been executed, use this neighbouring solution (which will be referred to as a test solution) to calculate the value of the objective function. This objective function value is referred to as E_t —the objective function value of the test solution. Proceed to step 3.

Step 3. Calculate the difference between the objective function values of the test solution and the current solution. This difference will be referred to as the 'energy change,' δE , and is calculated from the formula

$$\delta E = E_t - E_c. \quad (14)$$

If the value of the energy change is negative ($E_t < E_c$)—the test solution provides a lower objective function value than the current solution—then the test solution is accepted as the new current solution along with its associated layout and objective function value. In this case, proceed to step 4. Otherwise, proceed to step 5.

Step 4. If the objective function value of the new current solution (E_c) is less than that of the best solution (E_b), then the best solution is replaced by the new current solution. Regardless of whether or not the best solution is replaced with the current solution, proceed to step 6.

Step 5. Generate the Metropolis criterion for accepting a test solution with an objective function inferior to that of the current solution. This criterion provides the following probability of an inferior test solution being accepted as the current solution

$$P(a) = \exp(-\delta E/T). \quad (15)$$

Where $-\delta E$ is the negative difference in the energy change and T is the current temperature (or control parameter). Next, a uniformly distributed random number (Ran) from the interval $(0, 1)$ is generated. If the value of Ran is less than the probability of the inferior test solution being accepted as the current solution, hence

$$\text{if } Ran < (a), \text{ then} \quad (16)$$

the test solution is accepted as the current solution along with its associated layout and objective function value (Metropolis *et al.* 1953). Otherwise, the current solution remains unchanged. Regardless of the action taken, proceed to step 6. It should again be emphasized that the reason an inferior solution is occasionally accepted as the current solution is to find other locations on the continuum of feasible solutions and avoid being trapped at local optima.

Step 6. If the current iteration number (N) is equal to the maximum number of iterations (N_{max}) for the current level of the control parameter (T), then proceed to step 7. Otherwise, increment the iteration number (N) by 1 and return to step 2.

Step 7. Adjust the cooling temperature by using the following relationship

$$T = T \times CR. \quad (17)$$

If the new value of T is less than the stopping criteria (T_{min}), then proceed to step 8. Otherwise, re-initialize the current iteration number (N) to 1, and return to step 2.

Step 8. The Simulated Annealing heuristic is complete. The best solution is that corresponding to E_b .

Figure 10 summarizes the logic of this heuristic in a flow chart (logical nodes have lines that are bolder than procedural nodes).

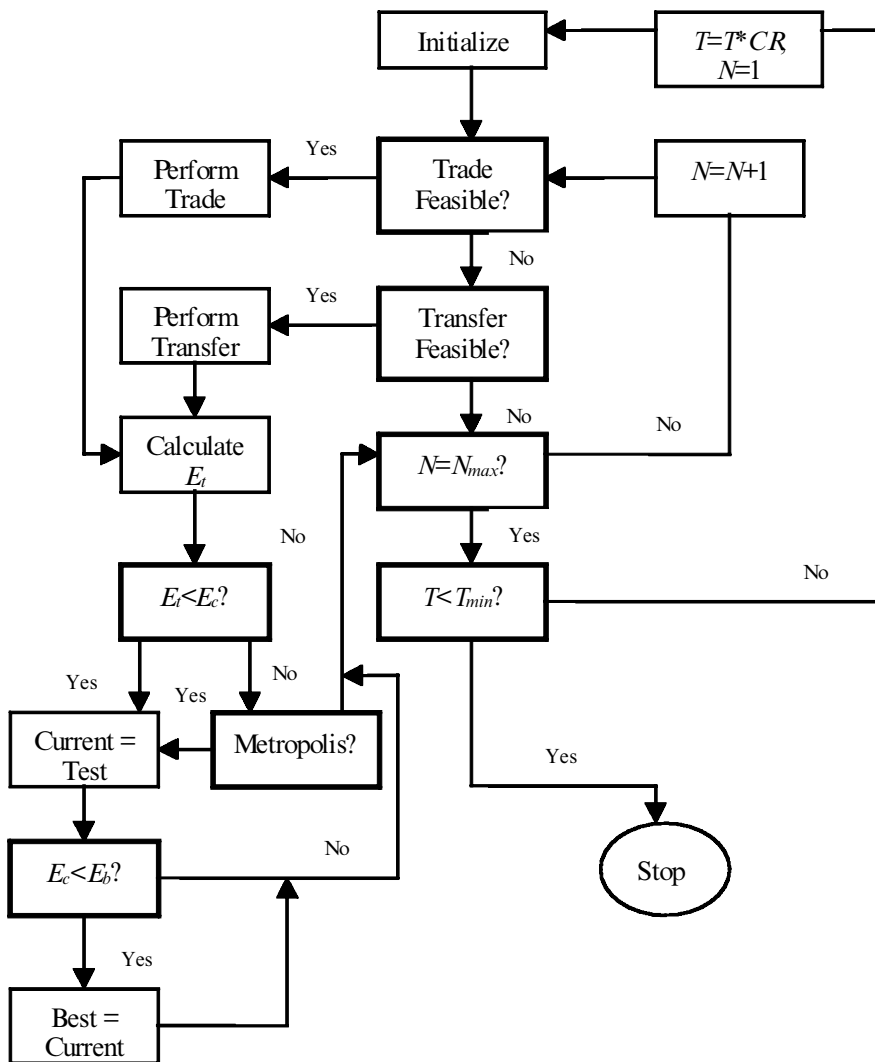


Figure 10. Flow-chart of Simulated Annealing heuristic.

3.4. Simple example of Simulated Annealing

A simple example of how Simulated Annealing can be used to address the Assembly Line Balancing Problem is shown to lend support to the presented methodology. Consider the 11-task problem by Mariotti (1970). The precedence relationships and task durations are shown in figure 11.

Given a cycle time of 0.225 minutes per unit, an arbitrarily generated initial solution is provided in table 1.

Equations (3) and (4) provide values for the number of workers and equipment units respectively. In more compact notation, this solution can be expressed as follows:

$$(AB)_{3,6}(CDEF)_{4,16}(GHI)_{3,9}(JK)_{3,6}.$$

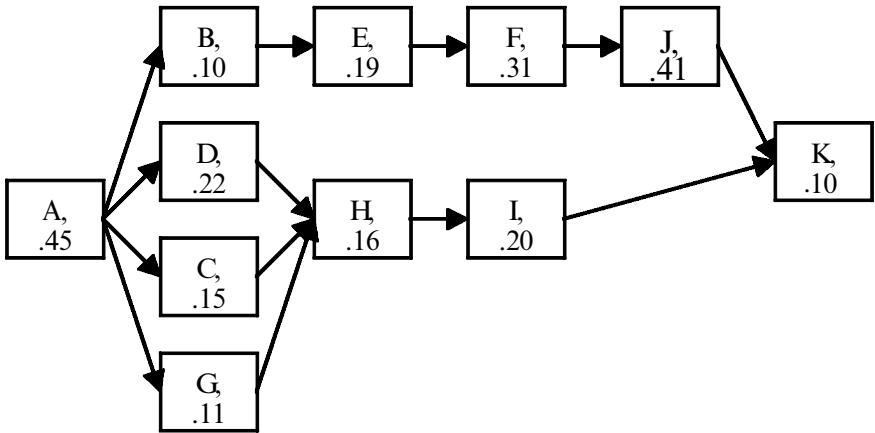


Figure 11. Precedence relationships for the example problem.

Work centre	Tasks	Workers	Equipment units
1	AB	3	6
2	CDEF	4	16
3	GHI	3	9
4	JK	3	6
Total		13	37

Table 1. An initial solution for the assembly line balancing problem.

The characters in parentheses represent the tasks in each work centre and the numbers following each set of parentheses represent the number of workers and equipment units in each respective work centre. If the annual cost of a worker is assumed to be \$30 000 per year and the annual cost of an equipment unit is \$3000, the total annual cost is \$501 000. The objective function used in this example is dedicated to the minimization of design cost (section 3.2.1).

Using Simulated Annealing search parameters of an initial temperature of 50 000, a cooling rate of 95%, and a single iteration at each level of the temperature ($T_1 = 50\,000$, $CR = 0.95$ and $N_{\max} = 1$), the first few 'steps' using the heuristic as described in section 3.3 are presented in table 2.

The first column of this table provides the annealing temperature values. The first line of the second column is the current solution, while the second line is the test solution. The third line of this column describes the type of action taken—for this particular example, only transfers were executed (section 3.3, step 2b). Trades were not executed because a small problem of this type has very few opportunities for feasible trades with respect to precedence relationships. The first line of the third column is the cost of the current solution (E_c) and the second line is the cost of the test solution (E_t). Both values of cost are in 1000s. The fourth column of the table states the decision made regarding acceptance of the test solution. If acceptance is made via the Metropolis criterion (section 3.3, steps 3–5), the associated probability of acceptance is provided.

T	Solutions (current and test) and description of action	Cost (000s)	Decision
50 000	(AB) _{3,6} (CDEF) _{4,16} (GHI) _{3,9} (JK) _{3,6}	501	Accept test solution via improvement
	(A) _{2,2} (BCDEF) _{4,20} (GHI) _{3,9} (JK) _{3,6} B is transferred to work centre 2	471	
47 500	(A) _{2,2} (BCDEF) _{4,20} (GHI) _{3,9} (JK) _{3,6}	471	Accept test solution via Metropolis criterion (P(A) = 0.5664)
	(A) _{2,2} (BCDE) _{3,12} (FGHI) _{4,16} (JK) _{3,6} F is transferred to work centre 3	498	
45 125	(A) _{2,2} (BCDE) _{3,12} (FGHI) _{4,16} (JK) _{3,6}	498	Accept test solution via improvement
	(A) _{2,2} (BCDE) _{3,12} (FGH) _{3,9} (IJK) _{4,12} I is transferred to work centre 4	465	
42 869	(A) _{2,2} (BCDE) _{3,12} (FGH) _{3,9} (IJK) _{4,12}	465	Accept test solution via improvement
	(A) _{2,2} (BCDE) _{3,12} (FG) _{2,4} (HIJK) _{4,16} H is transferred to work centre 4	432	
40 725	(A) _{2,2} (BCDE) _{3,12} (FG) _{2,4} (HIJK) _{4,16}	432	Accept test solution via Metropolis criterion (P(A) = 0.5971)
	(A) _{2,2} (BCDE) _{3,12} (FG) _{2,4} (HIJ) _{4,12} (K) _{1,1} K transferred to new work centre 5	453	
38 689	(A) _{2,2} (BCDE) _{3,12} (FG) _{2,4} (HIJ) _{4,12} (K) _{1,1}	453	Accept test solution via improvement
	(A) _{2,2} (BCDE) _{3,12} (FG) _{2,4} (HI) _{2,4} (JK) _{3,6} J is transferred to work centre 5	444	
36 755	(A) _{2,2} (BCDE) _{3,12} (FG) _{2,4} (HI) _{2,4} (JK) _{3,6}	444	Accept test solution via improvement
	(A) _{2,2} (BCDE) _{3,12} (FG) _{2,4} (HI) _{2,4} (J) _{2,2} (K) _{1,1} K transferred to new work centre	435	
34 917	(A) _{2,2} (BCDE) _{3,12} (FG) _{2,4} (HI) _{2,4} (J) _{2,2} (K) _{1,1}	435	Accept test solution via Metropolis criterion (P(A) = 0.9135)
	(A) _{2,2} (BCDE) _{3,12} (FG) _{2,4} (H) _{1,1} (IJ) _{3,6} (K) _{1,1} I is transferred to work centre 5	438	

Table 2. Simulated Annealing iterations of the example problem.

When T is 50 000 (the first row of the table), the test solution obtained by moving task B to work centre 2 is accepted as current because the transfer results in an improved value of the objective function. When T is 47 500, the current solution is replaced by the test solution (obtained by transferring task F to work centre 3) due to the fact that the Metropolis criterion was met. The Metropolis criterion was met because a random number was generated that was in the interval (0, 0.5664), and the criterion suggests that an inferior solution should be accepted with a probability of 0.5664. The finding of these ‘neighbouring’ solutions, along with their associated costs and decision issues continues for several iterations as shown in table 2.

For this simple example, all instances result in acceptance of the test solutions—regardless of whether acceptance comes via improvement or the Metropolis criterion. This, of course, will not be the case for all problems. It is important to keep in mind that the acceptance of test solutions having objective function values inferior to that of their corresponding current solutions depends on the value of T and well as the difference between E_c and E_t . For this example, the solution (A)_{2,2}(BCDE)_{3,12}(FG)_{2,4}(HIJK)_{4,16} is the one which results in the lowest cost found—\$432 000 (the shaded region in table 2). It is important to note that this

'best' solution was found after accepting a relatively inferior solution via the Metropolis criterion, which is the basic strategy used with Simulated Annealing to avoid being trapped at local optima.

4. Design of experiment

An experiment was designed to compare the Simulated Annealing solution with solutions attained from other line balancing heuristics. Seven different line balancing problems (11, 21, 25, 29, 40, 45, and 74 tasks) were each solved by 23 different heuristics (six of these use Simulated Annealing), yielding 161 different assembly line layouts for analysis. Details of the seven problems are given in table 3.

For each of the seven problems, the desired cycle time was specified to be 10 minutes between completed units. The durations for the tasks were created via a random number generator. For each task, there was a 75% probability the task duration would be uniformly distributed between 2 and 10 minutes, and a 25% probability the task duration would be uniformly distributed between 10 and 15 minutes. The per-person labour cost (L) was assumed to be \$30 000 per year and the per-machine equipment (Q) cost was assumed to be \$3000 per year. The precedence diagram for the 11 task problem is from Mariotti (1970), for the 21 task problem is from Tonge (1965), for the 29 task problem is from Buxey (1974), and for the 45 task problem is from Kilbridge and Wester (1961). The 25, 40 and 74 task problems were generated for this research. For the 29, 40, 45 and 74-task problems, mixed-models are present. The product weights for these problems are presented in table 3 and represent the portion each product type contributes to the total product mix.

4.1. Mixed-model problems and production sequencing

Note that the last four of the seven problems described in table 3 have multiple products made simultaneously during the production run. This is known as a mixed-model problem and reflects a practical issue pertaining to the design of assembly lines in JIT systems. The following paragraph describes the sequencing approach used for the mixed-model problems during the production simulations.

Consider the 74-task problem as an example. Four different products are to be made simultaneously. One possible way to sequence them is according to Sequence A:

Sequence A: 1-1-1-1-2-2-3-4.

Tasks	Different products	Product-mix weights	Simulated build-up time (min)	Iterations for each value of $T(N_{\max})$
11	1	$w_1 = 1$	500	10
21	1	$w_1 = 1$	1000	25
25	1	$w_1 = 1$	1000	25
29	2	$w_1 = 2, w_2 = 1$	3000	50
40	3	$w_1 = 3, w_2 = 2, w_3 = 1$	3000	50
45	2	$w_1 = 2, w_2 = 1$	3000	50
74	4	$w_1 = 4, w_2 = 2, w_3 = 1, w_4 = 1$	3000	75

Table 3. Description of seven different line balancing problems.

With this sequence, the products are not being introduced into the production system at a steady rate, as would be desirable in many JIT systems. Another approach is Sequence B:

Sequence B: 1-2-1-3-1-2-1-4.

This sequence introduces products into the production system at a steadier rate than Sequence A. Since this research assumes product changeovers are negligible (Sequence B requires more changeovers than A), Sequence B would be preferable to A due to its better compatibility with JIT systems. The methodology for this mixed-model sequencing is presented by Ding and Cheng (1993), and is utilized for all production sequencing here.

4.2. *Output performance measures*

The production performance for the 161 solution layouts was simulated using SLAMSYSTEM v4.6 and FORTRAN v5.1 user-written inserts. The production performance measures of interest were average WIP level, average flow-time, system throughput, system utilization, on-time completion, average unit labour cost, and cycle time performance. Each simulation run was for a set period of time rather than for a set number of completed units. For this reason, total design cost per unit was examined as well as absolute total design cost.

On-time completion is a measure of the layout's ability to move units through the individual work centres within a certain time frame, which in this case is the cycle time. The measure is expressed in units of percentage of units completed on-time and is referred to hereafter as POT. High values of POT are desired. Cycle time performance is the ratio of desired cycle time to actual cycle time achieved. This output is a measure of the layout's ability to move units through the system and is referred to hereafter as CTR. Like POT, higher levels of CTR are preferred.

For all simulated production runs, statistics were reset after steady-state conditions were attained—build-up times are offered in table 3. This was done to avoid any bias which may be associated with system start-up. The run was then continued for 3000 simulated minutes and relevant statistics were collected. For each of the 161 layouts, 25 repetitions were made so that reasonable estimates of outputs could be attained. This resulted in a database of 4025 records. Also, for each of the 161 layouts, a common random number seed was not used.

The main factor for this experiment was the heuristic used to generate the production layouts. Twenty-three different heuristics were used to generate production layouts. Six of these resulted from the Simulated Annealing based objective functions, and the other 17 are described in the next section.

4.3. *Line balancing heuristics used in the experiment*

Many of the heuristics used in this study are a derivation of Gaither's Incremental Utilization Heuristic (1996). Generally speaking, for each iteration of placing a task into a work centre according to precedence relationships, a list is constructed of all tasks which are eligible for immediate assignment to a work centre. From this list a task is selected based upon some rule, or heuristic. The chosen task is not added to a work centre unless its addition results in an increase in the work centre's utilization—hence the name 'incremental utilization.' Table 4 provides a brief description of all heuristics used in the experiment.

Label	Heuristic	Incremental utilization
1	Select task providing maximum incremental utilization	Used
2	Select task randomly (Acrucus 1966)	Used
3	Select task with longest duration	Used
4	Select task with shortest duration	Used
5	Select task providing minimum incremental utilization	Used
6	Select task providing minimum probability for lateness within work centre	Used
7	Select task best composite of 5 and 6	Used
8	Select task according to lexicographic attributes	Used
9	Single Mega work-centre	Not used
10	Individual work centre for each task	Not used
11	Select task having fewest followers	Used
12	Select task having fewest immediate followers	Used
13	Select task which is first to become available	Used
14	Select task which is last to become available	Used
15	Select task having most followers	Used
16	Select task having most immediate followers	Used
17	Select task with highest Ranked Positional Weight (Helgeson and Birnie 1961)	Used
18	Simulated Annealing: Minimize Design Cost—E ₁	Not used
19	Simulated Annealing: Minimize Smoothness Index—E ₂	Not used
20	Simulated Annealing: Minimize Overall System Lateness—E ₃	Not used
21	Simulated Annealing: Minimize Composite Function—E ₄	Not used
22	Simulated Annealing: Minimize Composite Function—E ₅	Not used
23	Simulated Annealing: Minimize Composite Function—E ₆	Not used

For detailed descriptions of the first 17 heuristics, refer to McMullen (1995), McMullen and Frazier (1997) and Baybars (1986).

4.4. *Research questions*

To guide the investigation of whether the Simulated Annealing solutions provide desirable layouts in terms of both the resources required and the production performance measures, two research questions were addressed.

- (1) Do the 23 heuristics have an overall effect on the performance measures?
- (2) If so, which of the 23 heuristics are most favourable in terms of both design cost and production performance?

4.5. *Computational experience*

The Simulated Annealing procedure for all problems was performed using an initial value of 10 000 and stopping criteria of 1000 for T , and a cooling rate (CR) of 0.95. Table 3 shows how many iterations were used at each value of T for each of the seven problems.

The Simulated Annealing was performed using Microsoft Visual Basic for DOS on a Pentium-75 Processor. Each solution required as little as one minute for the 11 task problems and as long as 20 minutes for the 74 task problems.

5. Experimental results

After running the simulations a correlation matrix of all the performance measures was obtained. The measures of average WIP level, average flow time, and average system throughput were each highly correlated with the cycle time ratio (CTR—correlation coefficients of - 0.966, - 0.965 and 1.000, respectively). As a result, average WIP level, average flow time and average system throughput were deleted from further direct analysis and left to be explained by CTR. Because of this relationship, high levels of WIP and flow time will obviously result in the layout's inability to move units through the system in a timely fashion, and vice versa. In addition to design cost, then, four output performance measures from the simulations remained for further analysis: average unit labour cost (ULC), percentage of units completed within the desired cycle time (POT), average system utilization (UTIL) and cycle time ratio (CTR).

To address the first research question a multivariate analysis of variance was performed. The results suggested that the different heuristics did have an overall multivariate effect on the production performance measures and the design cost. Wilks' $\lambda = 0.0501$, with an associated $F = 149.987$ and $p < 0.0001$. As a follow-up, table 5 provides univariate and discriminant analysis statistics for the production performance measures and the implementation cost. For the discriminant analysis, five functions were identified, but only four were determined to be statistically significant.

The discriminant function coefficients provide information as to which response variables are sensitive to the 23 different heuristics. Discriminant function coefficients unique from zero suggest sensitivity. The number of functions interpreted is dependent on the number of response variables interpreted. For more information on the discriminant analysis, refer to Tabachnick and Fidell (1989). From inspection of table 5, the results suggest that the production performance measures and the implementation cost are all sensitive to the selected heuristic, with the production performance measures of POT and Utilization being most sensitive to the selected heuristic.

Given that the choice of heuristic has a significant effect on the solution, an investigation ensued to determine which heuristics provided better solutions. Table 6 shows the mean values by heuristic for each of the five performance measures. Table 6 also shows mean values for total design cost per unit completed, or unit total cost (UTC). For each of the seven problem sizes, standard deviations were computed for each heuristic for each performance measure. Since these were all reasonably consistent across heuristics, they are not presented here.

Output measure	Func. 1	Func. 2	Func. 3	Func. 4	Univariate F	$p <$
ULC	- 0.7995	- 1.2025	- 1.6186	0.6869	5.95	0.0001
POT	0.3151	0.6936	- 1.2685	- 0.9385	154.42	0.0001
Utilization	- 2.8665	0.5000	- 1.4939	- 0.0411	161.47	0.0001
CTR	2.2406	- 0.2282	0.7671	1.6446	50.08	0.0001
Design Cost	1.4425	1.9025	1.4053	- 0.2472	54.86	0.0001

Table 5. Standardized discriminant coefficients and univariate F s for performance measures.

Heuristic	ULC	POT	UTIL	CTR	Cost (1000s)	UTC
1	153.6	0.711	0.758	0.853	1104	4576
2	139.9	0.706	0.802	0.878	1153	4542
3	158.7	0.762	0.763	0.835	1145	5018
4	137.1	0.713	0.827	0.882	1196	4666
5	147.4	0.763	0.780	0.857	1336	5540
6	136.0	0.765	0.819	0.900	1323	5072
7	135.2	0.778	0.822	0.908	1326	4997
8	139.5	0.771	0.814	0.892	1182	4617
9	113.2	0.812	0.971	0.934	3555	15110
10	204.1	0.920	0.555	0.930	1447	5338
11	148.4	0.780	0.776	0.878	1172	4684
12	144.4	0.763	0.782	0.881	1188	4663
13	149.3	0.791	0.762	0.874	1191	4732
14	151.1	0.800	0.755	0.857	1176	4782
15	154.2	0.800	0.748	0.865	1215	4950
16	150.8	0.812	0.770	0.870	1160	4703
17	150.0	0.786	0.768	0.878	1179	4731
18	162.2	0.732	0.732	0.810	1085	4841
19	129.6	0.755	0.860	0.917	1655	6267
20	139.2	0.943	0.790	0.991	1492	5056
21	138.0	0.917	0.796	0.978	1383	4795
22	137.6	0.919	0.797	0.979	1394	4818
23	137.4	0.908	0.796	0.972	1355	4739

Table 6. Mean performance measure values by rule.

5.1. Interpretation of results

Upon examination of table 6, it is clear that a trade-off generally exists between achieving superior cycle time performance and superior design cost performance. For example, heuristics 1 and 2 provided the lowest unit total design cost (UTC) but provided relatively poor cycle time performance (CTR). Interestingly, heuristic 2 randomly selects a task from the eligible list of tasks. As another example, heuristic 18 provided the very best absolute design cost of 1 085 000 but provided the very worst cycle time performance.

To investigate further any differences between mean values of cycle time performance and mean values of unit total cost performance, analyses of variance were performed. The resulting *F*-statistic was predictably significant at the 0.0001 level, but this result is not very meaningful due to the very large sample size (4025). A Tukey's multiple comparison test was then performed for PCTA and UTC, which did yield meaningful results. For CTR, rules 20, 21, 22 and 23 (all Simulated Annealing based rules) were all found to be significantly better than the remaining rules at the 0.05 level. For UTC, none of the better rules were found to be statistically different than the majority of other rules at the 0.05 level.

To investigate which rules perform better at both design cost and cycle time performance simultaneously, rankings of rules within problem sizes were examined. Although admittedly some information is lost in using non-parametric analyses, the use of average rankings can provide useful insights into the performance patterns of the heuristics. The difference in magnitudes across performance measures and the presence of outlying mean values makes it difficult to compare the multiobjective

performance using absolute values. The use of rankings provides a degree of consistency across performance measures.

To compute average rankings, the mean performance values for each rule for each of the seven problem sizes was calculated from the output of the 25 simulation repetitions. Then for each problem size the 23 rules were ranked separately for each performance measure, with the best rule receiving the lowest rank value. Finally, the ranked value for each rule was averaged across the seven problem sizes to obtain the average rankings shown in table 7.

Of primary interest in table 7 is the last column, which shows the average of the values in the CTR column and UTC column. The rank values in the right column represent an equal weighting of cycle time performance and unit design cost performance. The fact that none of the values in the last column is less than 7 suggests that none of the rules demonstrated truly superior performance on both measures. Four of the rules showed an average ranking of less than 10. Three of these rules (21, 22 and 23) are Simulated Annealing based heuristics. The fourth (rule 2) used a random selection strategy. The authors are at a loss in trying to explain why rule 2 performed so well on design cost.

Results for the performance measures of unit labour cost (ULC), percent of on-time completion (POT), and system utilization (UTIL) have not been discussed much. These results have been included in the above tables for the interested reader since they are traditional line balancing performance measures. However, because cycle time performance (CTR) and design cost performance (UTC) are

Rule	ULC	POT	UTIL	CTR	COST	UTC	CTR&UTC
1	17.5	16.8	17.0	19.1	4.5	7.1	13.1
2	9.4	18.6	11.3	14.5	6.1	4.7	9.6
3	13.3	13.9	14.7	20.1	5.3	9.3	14.7
4	7.9	19.4	7.3	15.7	7.6	8.4	12.1
5	10.7	15.7	10.3	13.7	15.8	16.4	15.1
6	7.9	15.8	8.4	11.6	15.0	15.9	13.8
7	7.6	13.4	7.8	11.2	15.0	13.6	12.4
8	10.6	14.7	10.0	13.9	8.2	7.9	10.9
9	1.0	11.6	1.0	5.9	23.0	23.0	14.4
10	22.7	3.7	22.7	8.1	20.4	19.9	14.0
11	15.0	12.5	14.2	14.1	8.6	9.1	11.6
12	13.6	14.7	13.1	13.4	8.4	9.0	11.2
13	16.0	12.4	16.6	15.4	7.7	9.6	12.5
14	14.4	11.5	14.9	13.2	8.6	9.4	11.3
15	15.4	13.1	15.3	14.1	10.6	13.4	13.8
16	13.8	10.4	13.2	13.4	8.1	9.6	11.5
17	14.8	12.4	14.2	14.5	9.4	10.6	12.6
18	14.1	16.8	14.0	18.8	1.9	8.4	13.6
19	3.9	16.3	2.9	10.1	17.4	17.3	13.7
20	13.1	2.0	13.0	3.6	20.4	17.4	10.5
21	11.5	3.4	11.7	4.3	18.4	12.4	8.3
22	11.2	3.2	11.4	3.7	18.8	13.2	8.5
23	10.7	3.8	11.0	3.5	16.7	10.4	7.0

Table 7. Average performance rankings by rule.

considered by the authors to be of much greater importance in modern manufacturing environments, the emphasis of the above analyses was focused on these measures.

5.2. Effects of sequencing

Another experiment was conducted to investigate the effect(s) that the sequencing of the mixed-model problems had on the relevant performance measures. Table 8 shows two different sequences used for each of the problems involving mixed-models. As stated earlier, the JIT sequence was used for the mixed-model problems in this research (Ding and Cheng 1993) whose results are discussed in the previous section. These results were then compared to more of a non-JIT sequence—a production sequence where the different products were introduced to the production system in a ‘lumpier’ fashion when compared to the steady release of products of the JIT sequence. Each integer value in the presented sequences is simply a label of the product to be processed by the system. The labels are consistent with the product-mixes as presented in table 3, and regardless of the sequence used, the product mixes are the same across each problem.

These product sequences are repeated throughout the simulated production runs, and the same general attributes of the original simulation model apply here.

The experiment showed that the chosen sequence does have an overall multivariate effect on the performance measures of interest (Wilks $\lambda = 0.8256$, $F = 237.31$, $p < 0.0001$). Table 9 shows the results of the experiment in terms of the means of the performance measures that are dependent upon the sequence.

Table 9 shows that at the 0.05 level of significance, only the percentage of on-time completion is not sensitive to the sequencing decision. The other performance measures are sensitive to the sequencing decision and they favour the JIT sequencing. The reason for this is that when compared to the non-JIT sequencing policy, the JIT sequence utilizes the steady release of differing jobs. When this occurs, the weighted averages used for task durations for the mixed-model problems (obtained from the line balancing procedure and used for the layout for the simulated production run) are more compatible with the task durations of the jobs moving through the system.

Tasks	JIT (mixed-model) sequence	Non-JIT sequence
29	121, 121, 121	111111222
40	1213121, 1213121, 1213121	111111111222222333
45	121, 121, 121	111111222
74	12131214, 12131214, 12131214	111111111111222222333444

Table 8. JIT and non-JIT sequences used.

Sequence	JIT-sequence	Non-JIT sequence	Univariate F	p -value
ULC	145.80	146.90	4.54	0.0332
POT	0.7551	0.7572	0.76	0.3840
UTIL	0.7433	0.6865	459.06	< 0.0001
CTR	0.8379	0.8317	4.52	0.0335

Table 9. Effect of sequence on relevant performance measures.

When the jobs are moving through the system according to the non-JIT sequence, their task durations are less compatible with the task durations used to design the layout (obtained from the weighted averages used for line balancing). This relative lack of compatibility results in more system 'clutter' when compared with the JIT sequence, and thereby inferior performance measures are observed.

6. Conclusions

In a manufacturing environment of high volume production, small batch sizes, mixed model sequencing, and schedule reliability, several assumptions of traditional line balancing approaches need to be relaxed. Since the problem of how best to allocate tasks across work centres must still be resolved, existing line balancing approaches should be re-evaluated. Also, since competitive emphasis is shifting from primarily cost performance to both cost and cycle time performance, new line balancing approaches should be developed to help achieve these goals. This study examined the performance of 23 existing and new line balancing strategies or approaches under a modern manufacturing environment. By using a simulation experiment, the resulting production performance of the approaches was evaluated.

Because of the inexpensive availability of computing power and the combinatorial nature of line balancing problems, Simulated Annealing was examined as a line balancing approach. Several search objectives were evaluated with Simulated Annealing. Combinations of different search objectives were also evaluated to investigate their impact on multiple performance objectives. On cycle time performance, several Simulated Annealing approaches performed significantly better than the other approaches, based on average performance measures and rankings. On design cost per unit produced, most of the Simulated Annealing approaches performed about average for all the heuristics.

Most importantly, this paper has shown how Simulated Annealing can be used to find improved solutions for balancing a production line where task times are stochastic, multiple products are produced in a mixed-model fashion, parallel workstations are allowed, and cycle time performance as well as total labour and equipment cost are all important. With appropriate search objectives, Simulated Annealing demonstrated a strong potential for yielding good solutions if cycle time performance is of primary concern. When unit total cost (labour and equipment) and cycle time performance were of equal importance, Simulated Annealing with composite search objectives still resulted in better performance than other approaches, based on average rankings. For a particular shop environment, several Simulated Annealing algorithmic parameters might be 'fine-tuned' for even better performance.

As is common with many multiple objective problems, there is an inherent trade-off between the objectives of cycle time performance and design cost performance. Conceptually, a company could improve cycle time performance by hiring more workers and purchasing more equipment. This research was unable to identify line balancing strategies that consistently excelled at both cycle time performance and design cost performance. If cycle time performance is of primary concern, then Simulated Annealing with search objectives 3, 4, 5 or 6 appears to provide the best solutions. If design cost is of primary concern then some of the traditional line balancing approaches appear to provide better solutions. If both performance objectives are of equal concern, then Simulated Annealing with composite search objectives appears most promising.

One avenue for further research would be to use Simulated Annealing in addition to traditional line balancing approaches. If design cost is the most important performance objective, then a few of the better traditional heuristics could be used to generate solutions. Each of these solutions could then be used as the initial solution for Simulated Annealing, and using a cost-focused search objective (Peterson 1993, Leu *et al.* 1994). Conducting a research study to evaluate the use of different line balancing heuristics to obtain initial solutions for Simulated Annealing when different performance objectives are important would contribute to a better understanding of the potential of this heuristic.

References

- ARCUS, A. L., 1966, COMSOAL: a computer method of sequencing operations for assembly lines. *International Journal of Production Research*, **4**, 259–277.
- BAYBARS, I., 1986, A survey of exact algorithms for the simple assembly line balancing problem. *Management Science*, **32**, 909–932.
- BUXEY, G. M., 1974, Assembly line balancing with multiple stations. *Management Science*, **20**, 1010–1021.
- DING, F. and CHENG, L., 1993, An effective mixed-model assembly line sequencing heuristic for just-in-time production systems. *Journal of Operations Management*, **11**, 45–50.
- EGLISE, R. W., 1990, Simulated annealing: a tool for operational research. *European Journal of Operational Research*, **46**, 271–271.
- GAITHER, N., 1996, *Production and Operations Management* (Belmont, CA: Duxbury).
- GOLDBERG, D. E., 1989, *Genetic Algorithms in Search, Optimization and Machine Learning* (Reading, MA: Addison-Wesley).
- HELGESON, W. B. and BIRNIE, D. P., 1961, Assembly line balancing using the ranked positional weight technique. *The Journal of Industrial Engineering*, **12**, 394–398.
- KILBRIDGE, M. D. and WESTER, L., 1961, A heuristic method of line balancing. *The Journal of Industrial Engineering*, **12**, 292–298.
- KIRKPATRICK, S., GELATT, C. D. and VECCHI, M. P., 1983, Optimization by simulated annealing. *Science*, **220**, 671–679.
- LEU, Y., MATHESON, L. A. and REES, L. P., 1994, Assembly line balancing using genetic algorithms with heuristic-generated initial populations and multiple evaluation criteria. *Decision Sciences*, **25**, 581–606.
- MARIOTTI, J., 1970, Four approaches to manual assembly line balancing. *Industrial Engineering*, **21**, 35–40.
- McMULLEN, P. R., 1995, A simulation approach to solving the type *i* assembly line balancing problem for mixed models with stochastic task durations. Doctoral dissertation, University of Oregon.
- McMULLEN, P. R. and FRAZIER, G. V., 1997, A heuristic for solving mixed-model line balancing problems with stochastic task durations and parallel stations. *International Journal of Production Economics*, **51**, 177–190.
- METROPOLIS, N., ROSENBLUTH, A., ROSENBLUTH, N., TELLER, A. and TELLER, E., 1953, Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, **21**, 1087–1092.
- MOODIE, C. L. and YOUNG, H. H., 1965, A heuristic method of assembly line balancing for assumptions of constant or variable work element times. *Journal of Industrial Engineering*, **16**, 23–29.
- PETERSON, C., 1993, A Tabu search procedure for the simple assembly line balancing problem. *Proceedings of the 1993 Decision Sciences Conference*, Washington, DC, pp. 1502–1504.
- PINTO, P., DANNENBRING, D. G. and KHUMAWALA, B. M., 1975, A branch and bound algorithm for assembly line balancing and paralleling. *International Journal of Production Research*, **13**, 183–196.
- PINTO, P., DANNENBRING, D. G. and KHUMAWALA, B. M., 1978, A heuristic network procedure for the assembly line balancing problem. *Naval Research Logistics Quarterly*, **25**, 299–307.

- PINTO, P., DANNENBRING, D. G. and KHUMAWALA, B. M., 1981, Branch and bound and heuristic procedures for assembly line balancing with paralleling of stations. *International Journal of Production Research*, **19**, 565–576.
- REEVE, N. R., 1971, Balancing continuous assembly lines. Doctoral dissertation, State University of New York at Buffalo.
- SILVERMAN, F. N. and CARTER, J. C., 1984, A cost effective approach to stochastic line balancing with off-line repairs. *Journal of Operations Management*, **6**, 145–157.
- SILVERMAN, F. N. and CARTER, J. C., 1986, A cost methodology for stochastic line balancing with intermittent line stoppages. *Management Science*, **32**, 455–463.
- SURESH, G. and SAHU, S., 1994, Stochastic assembly line balancing using simulated annealing. *International Journal of Production Research*, **32**, 1801–1810.
- TABACHNICK, B. G. and FIDELL, L. S., 1989, *Using Multivariate Statistics* (New York: Harper Collins).
- TONGE, F. M., 1965, Assembly line balancing using probabilistic combinations of heuristics. *Management Science*, **11**, 727–735.