Int. J. Adv. Res. Sci. Technol. Volume 3, Issue1,2014, pp.01-07# ICV 5.14

IJARST

International Journal of Advanced Research in Science and Technology

journal homepage: www.ijarst.com

ISSN 2319 – 1783 (Print)

ISSN 2320 – 1126 (Online)

Simulated Annealing Algorithm for U- Shaped Line Balancing Problem

K .G .Durga Prasad [1]*, K. Venkata Subbaiah[2],G. Venu Gopal Rao[3]andG.Padmavath[4]

[1]Department of Mechanical Engineering, School of Engineering, G.V.P.Technical campus, G.V.P.College for Degree & P.G.Courses, Visakhapatnam, Andhra Pradesh,India.

[2]Department of Mechanical Engineering, Andhra University, Visakhapatnam, Andhra Pradesh, India.

[3]Department of Mechanical Engineering, Andhra University, Visakhapatnam, Andhra Pradesh, India.

[4]Department of ECE, Govt. Polytechnic, Amudalavalasa, Srikakulam District, Andhra Pradesh, India.

****Corresponding Author's Email: dr.kgdp@ gmail.com***

A R T I C L E I N F O

Article history:

Received	13 Feb. 2014
Accepted	28 Feb. 2014
Available online	14 Mar. 2014

Keywords:

U-shaped Line Balancing,
Simulated Annealing.

A B S T R A C T

In this paper, Simulated Annealing (SA) algorithm is proposed for solving U-shaped line balancing problem (ULB). In a ULB problem the tasks are assigned to workstations either after all its predecessors or all of its successors, have been assigned to workstations. The objectives considered for the optimisation are to minimise the idle time of a U-shaped line and to minimise the workload deviation among workstations or a combination of both. This paper illustrates SA algorithm for some problems considered from literature and then compared with those of the literature.

Introduction:

A product or an assembly line may be quoted as the best example of production layout used in mass production, which consists of number of work elements known as tasks. These tasks are sequenced among a set of workstations without violating the precedence relationship among the tasks. The product remains at each workstation for a fixed time units called the cycle time. The problem of choosing this sequence of tasks is either to minimise the number of workstations for a given production rate (Type – I problem) or maximise the production rate for a given number of workstations (Type – II problem). Type –I line balancing problems are frequently encountered when a high production rate is required whereas Type –II problems are often encountered when there is a limitation of available working space [1]. The type-I problem is considered in the present work. Salveson was the first to address the assembly line balancing problem [2]. Kilbridge and Wester [3], Ignall [4] provided an excellent review on line balancing. Talbot et al. [5] reviewed specifically the heuristic approaches used to solve line balancing problem. With the increasing competitive pressure and due to the inflexibility of traditional line, the lines are needed to be made more flexible to meet the increasing demand. A common approach to increase the flexibility of a traditional line is to change from a straight line to a U-line. This not only increases the flexibility of the line but also minimises the raw material and work in process inventories. Miltenburg and Wijngaard [6] derived a new problem from traditional line balancing problem, which was arranged in U-shaped lines instead of straight lines. In a traditional line balancing problem, for a given workstation, the set of possible assignable tasks is confirmed to those tasks whose predecessors have already been assigned to workstations, where as in the U-shaped line problem the set of assignable tasks is determined by all those tasks whose predecessors or successors have already been assigned. This makes the U-shaped line balancing problem more complex algorithmically. They developed a dynamic programming procedure for U-shaped lines, which was due to the computational costs limited 11 task problems. For larger problems they adopted heuristics developed for traditional assembly line.

In this paper Simulated Annealing technique is employed to address the Type I U-shaped line balancing problem. The objectives considered for optimization are minimizing the idle time of a U-shaped line and to minimize the work load deviation among work stations or a combination of both. The remaining of this paper is organized as follows. In the section 2, the concept of simulated annealing is discussed. The section 3 presents the proposed methodology.

A numerical illustration is presented in section 4 to demonstrate the application of simulated annealing to address the U-shaped assembly line balancing problem. Finally conclusions are made in section 5.

Simulated Annealing:

Simulated Annealing was first introduced in 1983 by Kirkpatrick et al. as an intriguing technique for optimising functions of many variables [7]. Simulated Annealing is a heuristic strategy that provides a means for optimisation of NP (Non-deterministic Polynomial)-complete problems. At the heart of the method of Simulated Annealing stays an analogy with thermodynamics, specifically with the way that liquids freeze and crystallize, or metals cool and anneal [8]. The basic ideas of SA, which are applicable to optimisation problems with continuous N-dimensional control spaces, e.g., finding the global minimum of some function f (x) in the presence of many local minima, where x is an N-dimensional vector. The elements required by the Metropolis procedure are as follows: the initial value of f, the objective function. The system state is the point x. The control parameter T, temperature with an annealing schedule by which it is gradually reduced at a constant rate, called the cooling rate (CR). And there must be a generator of random changes in the configuration i.e., a procedure for taking a random step from x to x+Δx. This gives the neighbouring solution. If, thus obtained objective function is superior to that of the current solution, the neighbouring solution becomes a 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. The distinctive feature of SA 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 inferior 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. The control parameter T, temperature is determined by generating about 100 feasible points randomly, then the maximum and minimum of the function among the generated points are evaluated. If f represents the function, then the temperature T is found by the difference between the maximum of f and the minimum of *f*. The initial temperature must be sufficiently high, resulting in a high degree of randomness in the initial stages of search. And the degree of randomness decreases with the decrease in temperature. Adil and Nabil [9] applied Simulated Annealing based procedure for Dynamic Layout Problem. McMullen and Frazier [10] presented a SA based technique to address the assembly line balancing problem for multiple objective problems when paralleling of workstations is permitted. They demonstrated how SA could be used to obtain line-balancing solutions when one or more objectives are important. Hong and Cho [11] presented a Simulated Annealing approach for solving dual-criteria single-model and deterministic assembly line balancing problems. The first objective they considered was to minimize the idle time of an assembly line for a given cycle time, while the second objective was to minimize the workload deviation among workstations.

Methodology:

The Type I U-shaped assembly line balancing problem [5] can be stated as:

Given a set of tasks

F = {k|k = 1, 2…n},

A set of precedence constraints

P = {(x, y) |task x must be completed before task y},

A set of task times

T = {t (k) |k = 1, 2… n} and

a cycle time C.

To find a collection of subsets of F,

(S1, S2,…, SN)

Where Si = {k|task k is done at station i}

The conditions to be satisfied are:

$$\bigcup_{i=1}^{N} S_i = F \qquad (1)$$

$$S_i \bigcap_{i \neq j} S_j = \varphi \qquad (2)$$

$$\sum_{k \in S_i} t(k) \leq C,\ i = 1, 2, ..., N. \qquad (3)$$

For each task y:
if $(x,y) \in P$, $x \in S_i$, $y \in S_j$, then $i \leq j$, for all x;
or
if $(y,z) \in P$, $y \in S_j$, $z \in S_k$, then $k \leq j$, for all z. (4)

The condition (1) ensures that all tasks are assigned to a station. As a result of condition (2), each task is assigned only once. That is the simple U-shaped Line Balancing problem does not permit tasks to be shared between stations. Condition (3) ensures that the work content of any station does not exceed the cycle time. Condition (4) ensures that the precedence constraints are not violated on the U-line.

A function f to evaluate the optimal solution that were considered were

$$\text{Minimise } f_1 = NC - \sum_{i=1}^{N} \sum_{k \in S_i} t(k) \qquad (5)$$

Minimise $f_2 = \sum_{i=1}^{N}\left(\sum_{k\in Si} t(k) - C\right)^2$ (6)

Minimise $f_3 = af_1 + bf_2$ (7)

Where f_1 minimizes total idle time, f_2 minimizes mean squared idle time, and f_3 is a combination of f_1 and f_2 where a and b are arbitrary constants each between 0 and 1, and $a+b=1$.

Simulated Annealing Solution:

This section proposes SA algorithm to a U-shaped assembly line balancing problem. A random sequence of the tasks is generated (without violating the precedence constraints) and the tasks from generated sequence are grouped into workstations shown in figure1. This forms the initial solution for the Simulated Annealing algorithm. Initially, the SA algorithm shown in figure 2 is given a feasible solution with the energy in equations (5, 6 or 7) at a high temperature. At iteration step *k*, let the current solution and its energy be denoted by $WS_0$and E_0, respectively. At each iteration step, a number of modified solutions are heuristically found from moving a task to another workstation, or from interchanging the task with another one of other workstation. Among the current solution and the modified ones, a solution is probabilistically chosen as the new solution to start the next iteration step. The procedures for finding modified solutions are summarized in the following steps.

***Step 1*:**

Initialise 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 criterion (T_{min}). The initial feasible solution for the problem is generated using procedure 1 and chooses an objective function for optimisation. 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 transfer or a trade. A transfer is first considered, but if a transfer is not possible, then a trade is considered. Randomly select a task P_α and its workstation ws_i. Also randomly select a workstation ws_j ($i\neq j$) into which the task P_α can be moved with the precedence constraint satisfied. One of the following options is then taken.

Option (a): Transfer

For the random workstation ws_j determine its workload τ_j, and move the $P_\alpha$with its process time $\tau_\alpha$into the ws_j. The new workload τ'_j at the ws_j then becomes $\tau' = \tau_j+\tau_\alpha$. If the τ'_jdoes not exceed the cycle time, that is $\tau'_j \leq C$, a modified solution is found. Otherwise, perform trade.

Option (b): Trade

A task $P_\alpha$in the ws_i can be interchanged with a task P_β having the processing time τ_β in the ws_j provided that not only the modified solution resulting from task interchange satisfies precedence constraints, but also the workloads at both work stations do not exceed the cycle time;$(\tau_i - \tau_\alpha + \tau_\beta) \leq C$ at ws_i, and $(\tau_j + \tau_\alpha - \tau_\beta) \leq C$ at the ws_j.

***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 $\delta E = E_t - E_c$.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.

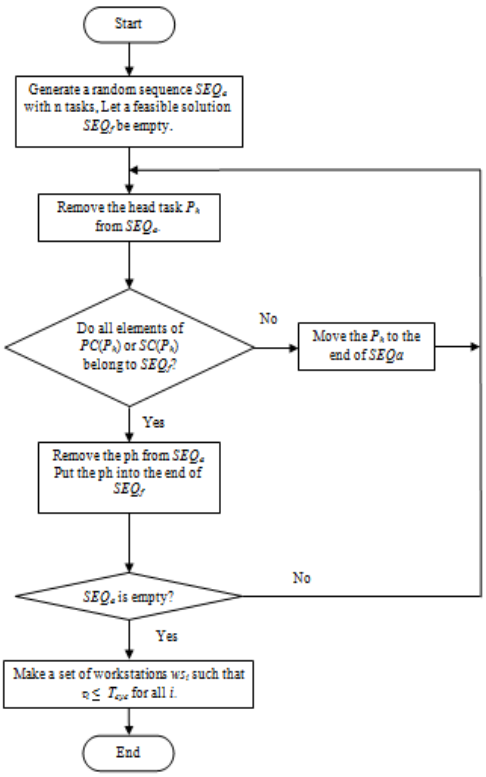

Figure:1. Flow chart for generating initial solution

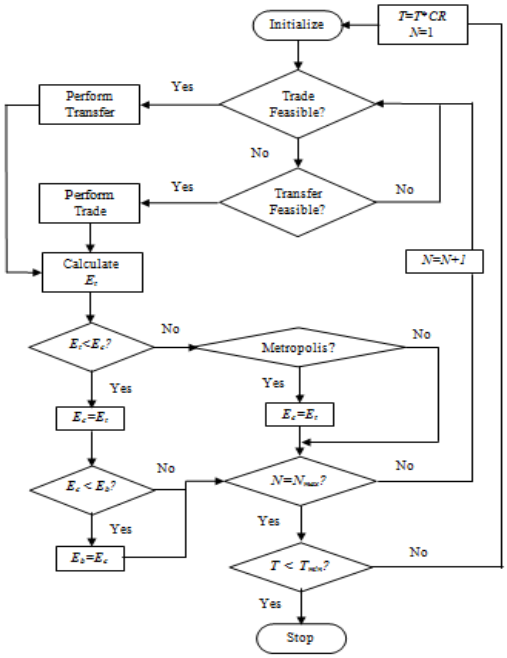

Figure: 2. Flow chart for simulated annealing solution

***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 [12].

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

Where - δE 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 if $Ran < P(a)$, then the test solution is accepted as the current solution along with its associated layout and objective function value. 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 *CR. If the new value of T is less than the stopping criteria (T_{min}), then proceed to step 8. Otherwise, re-initialise 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.The application of the algorithm is illustrated with the help of a numerical example in the next section.

Numerical Illustration:

A simple example of how Simulated Annealing can be used to address the U-Shaped Assembly Line Balancing Problem is shown to lend support to the present methodology. Consider the 11-task problem by Jackson [13]. Considering a cycle time of 9 per unit, an arbitrary sequence of assembly tasks and the respective workstations is initially generated that forms the initial feasible solution. The initial sequence generated arbitrarily is (1) (3 11)(2)(4)(9 6)(10 5)(8 7). The characters in the parenthesis represent the tasks in each workstation. The tasks and the respective work stations are shown in table1. The objective function used for this example is dedicated to the minimization of mean squared idle time.

Table1: The allocation of tasks and the respective workstations

Work station	Tasks
1	1
2	3 11
3	2
4	4
5	9 6
6	10 5
7	8 7

To determine the initial temperature, 100 random feasible solutions were generated and the initial temperature was calculated as discussed in section 2. Using Simulated Annealing, search parameters of an initial temperature of 76.77, a cooling rate of 95% and a single iteration at each level of the temperature and all the iterations are shown 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 i.e., trade or transfer executed. The first line of the third column is the cost of the current solution (Ec) and the second line is the cost of the test solution (Et). The fourth column of the table states the decision made regarding acceptance of the test solution.

If acceptance is made via the Metropolis criterion, the associated probability of acceptance is provided. When T is 76.77 (the first row of the table), the test solution obtained by trading task 8 from workstation 7 to workstation 5 with task 9, is accepted as current solution because the trade results in an improved value of the objective function. When T is 69.28, the current solution is replaced by the test solution (obtained by transferring task 7 from workstation 6 to workstation 5) due to the fact that the Metropolis criterion was met. The Metropolis criterion was met because a random number was generated that was 0.01, and the criterion suggests that an inferior solution should be accepted with a probability of 0.01. These 'neighbouring' solutions, along with their associated costs and decision issued for several iterations are 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 as well as the difference between Ec and Et. For this example, the solution (1 2) (9 11) (4) (10 7) (3 5) (8 6) is the one which results in the lowest cost found 2.667 (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. The set of problems considered for this study were from the literature. These set of problems were solved by Miltenburg and Wijngaard [6] with different cycle times using dynamic programming procedure and heuristic methods. These set of problems were applied to SA heuristic and obtained satisfactory results. The results for these problem sets are tabulated in Table 3.

The first column represents the problem source and the number of tasks that are present. The second column represents various cycle times that were used for the considered problem. And the third column shows the minimum number of workstations for the given cycle time (This is obtained by the relation $\sum t(k)/C$). The column N_{DP} [6] shows the results obtained from the dynamic programming procedure and the column N_{HEUR} [6] represents the number of workstations obtained from heuristic procedures. The column GA represents the number of work stations obtained from GA approach [14]. Our results of SA are shown in columns SA_1, SA_2 and SA_3 representing the objective functions f_1, f_2 and f_3 respectively.

5. Conclusions

This paper considers a SA solution for Type I U-shaped line balancing problem. SA is algorithmically simple than compared to other methods (traditional and non-traditional methods). Moreover the SA used only one procedure for generating the initial solution with which sets of satisfactory results were obtained. On the other hand GA used six different procedures for the generation of the initial solution. The results prove that SA to be computationally competitive method for U-shaped line balancing problem.

References:

1. Mc Mullen, 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(3), 177-190.
2. Lapierre, S.D., Ruiz, A. and Soriano, P., 2006, Balancing assembly lines with tabu search. European Journal of Operational Research, 168(3), 826-837.
3. Kilbridge, M.D., and Wester, L., 1962, A Review Of Analytical Systems Of Line Balancing. Operations Research, 10(5), 626-638.
4. Ignall, E., 1965, A Review of Assembly Line Balancing. Journal of Industrial Engineering, 16(4), 244-254.
5. Talbot, F.B., Patterson, J.H., and Gehrlein, W.V., 1986, A Comparative Evaluation of Heuristic Line Balancing Techniques. Management Science, 32(4), 430-454.
6. Miltenburg, G.J., Wijngaard, J., 1994, The U-Line Balancing Problem. Management Science, 40(10), 1378-1388.
7. Bertsimas, D and Nohadani, O, 2010, Robust optimization with simulated annealing. Journal of Global Optimization, 48(2), 323-334.
8. Orosz, C. Sajo, P. Farkas, L. Nagy, L, 2002, Radio access optimization for point-multipoint systems based on homogeneous simulated annealing., Proceedings of The 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, Portugal, (5), 2214-2217.
9. Adil, B and Nabil N.Z.G, 2001, A Simulated Annealing For Dynamic Layout Problem. Computers & Operations Research, 28, 1403-1426.
10. Mc Mullen, P.R. and Frazier, G.V, 1998, Using Simulated Annealing To Solve A Multi-objective Assembly Line Balancing Problem With Parallel Workstations. International Journal of Production Research, 36(10), 2717-2741.
11. Hong D.S., Cho H.S, 2001, Dual Criteria Assembly Line Balancing For Single-Model And Deterministic Problems Using Simulated Annealing. Intelligent Automation And Soft Computing, 7(1), 60-73.
12. Mc Mullen, P.R. and Frazier, G.V., 1998, Using simulated annealing to solve a multi-objective assembly line balancing problem with parallel work stations. International Journal of Production Research, 36 (10), 2717-2741.
13. J.R. Jackson, "A Computing Procedure For A Line Balancing Problem", Management Science, 1956, No.2, 261-271.
14. Debora A. Ajenblit, Roger L. Wainwright, 1998, Applying Genetic Algorithms To The U-Shaped Assembly Line Balancing Problem. IEEE, 96-101.

Table: 2. Sample iterations of SAalgorithm

T	Solutions (current and test) and description of action	Cost	Decision
76.77	(1) (3 11) (2) (4) (9 6) (10 5) (8 7) (1) (3 11) (2) (4) (8 6) (10 5) (9 7) 8 is traded from WS-7 to WS-5 with 9.	10.714 10.429	Accept test solution via improvement.
72.93	(1) (3 11) (2) (4) (8 6) (10 5) (9 7) (1) (3 11) (4 2) (8 6) (10 5) (9 7) 4 is transferred from WS-4 to WS-3.	10.429 3.333	Accept test solution via improvement.
69.28	(1) (3 11) (4 2) (8 6) (10 5) (9 7) (1) (3 11) (4 2) (8 6) (10 5 7) (9) 7 is transferred from WS-6 to WS-5.	3.333 4.333	Accept test solution via Metropolis criterion (Ran=0.01)
65.82	(1) (3 11) (4 2) (8 6) (10 5 7) (9) (1) (3 11) (4 2) (9 6) (10 5 7) (8) 8 is traded from WS-4 to WS-6 with 9.	4.333 3.667	Accept test solution via improvement.
62.53	(1) (3 11) (4 2) (9 6) (10 5 7) (8) (1) (3 11) (4 2) (8 6) (10 5 7) (9) 9 is traded from WS-4 to WS-6 with 8.	3.667 4.333	Accept test solution via Metropolis criterion (Ran=0.139)
59.40	(1) (3 11) (4 2) (8 6) (10 5 7) (9) (1) (3 11) (4 2) (8 6) (9 5 7) (10) 9 is traded from WS-6 to WS-5 with10.	4.333 4.333	Accept test solution via Metropolis criterion (Ran=0.029)
56.43	(1) (3 11) (4 2) (8 6) (9 5 7) (10) (1) (9 11) (4 2) (8 6) (3 5 7) (10) 9 is traded from WS-5 to WS-2 with 3.	4.333 4.333	Accept test solution via Metropolis criterion (Ran=0.212)
53.61	(1) (9 11) (4 2) (8 6) (3 5 7) (10) (1) (9 11) (4 2) (8 6) (10 5 7) (3) 3 is traded from WS-5 to WS-6 with10.	4.333 4.333	Accept test solution via Metropolis criterion (Ran=0.214)
50.93	(1) (9 11) (4 2) (8 6) (10 5 7) (3) (1) (9 11) (4 2) (3 6) (10 5 7) (8) 8 is traded from WS-4 to WS-6 with 3.	4.333 3.667	Accept test solution via improvement.
48.38	(1) (9 11) (4 2) (3 6) (10 5 7) (8) (1) (9 11) (4 2) (3) (10 5 7) (8 6) 6 is transferred from WS-4 to WS-6.	3.667 4.333	Accept test solution via Metropolis criterion (Ran=0.051)
45.97	(1) (9 11) (4 2) (3) (10 5 7) (8 6) (1) (9 11) (4 2) (10) (3 5 7) (8 6) 3 is traded from WS-4 to WS-5 with10.	4.333 4.333	Accept test solution via Metropolis criterion (Ran=0.141)
43.67	(1) (9 11) (4 2) (10) (3 5 7) (8 6) (1) (9 11) (4 2) (10 7) (3 5) (8 6) 7 is transferred from WS-5 to WS-4.	4.333 3.333	Accept test solution via improvement.
41.48	(1) (9 11) (4 2) (10 7) (3 5) (8 6) (1 2) (9 11) (4) (10 7) (3 5) (8 6) 2 is transferred from WS-3 to WS-1.	3.333 2.667	Accept test solution via improvement.
39.41	(1 2) (9 11) (4) (10 7) (3 5) (8 6) (1) (9 11) (4 2) (10 7) (3 5) (8 6) 2 is transferred from WS-1 to WS-3.	2.667 3.333	Accept test solution via Metropolis criterion (Ran=0.921)

Table: 3. Comparison of the computational results

Problem	Cycle time	Ideal (N)	N_{DP}	N_{HEUR}	N_{GA}	N_{SA1}	N_{SA2}	N_{SA3}
Merten 7	6	5	6	6	6	6	6	6
	7	5	5	5	5	5	5	5
	8	4	5	5	5	5	5	5
	10	3	3	3	3	3	3	3
	15	2	2	2	2	2	2	2
	18	2	2	2	2	2	2	2
Jaeschke 9	6	7	8	8	8	8	8	8
	7	6	7	7	7	7	7	7
	8	5	6	6	6	6	6	6
	10	4	4	4	4	4	4	4
	18	3	3	3	3	3	3	3
Jackson 11	7	7	7	8	7	7	7	7
	9	6	6	6	6	6	6	6
	10	5	5	5	5	5	5	5
	13	4	4	4	4	4	4	4
	14	4	4	4	4	4	4	4
	21	3	3	3	3	3	3	3
Mitchel 21	14	8	----	8	8	8	8	8
	15	7	----	8	8	8	8	8
	21	5	----	6	5	5	5	5
Heskiaoff 28	138	8	----	8	8	8	8	8
	205	5	----	6	6	6	6	6
	216	5	----	5	5	5	5	5
	256	4	----	4	4	4	4	4
	324	4	----	4	4	4	4	4
Sawyer 30	25	13	----	15	14	14	14	14
	27	12	----	14	13	13	13	13
	30	11	----	12	12	12	12	12
	36	9	----	10	10	10	10	10
	54	6	----	7	7	7	7	7
	75	5	----	5	5	5	5	5
Kilbridge & Webster 45	79	7	----	8	8	8	8	8
	92	6	----	7	7	7	6	7
	110	6	----	6	6	6	6	6