<http://www.diva-portal.org>

Postprint

This is the accepted version of a paper published in *Journal of Modelling in Management*. This paper has been peer-reviewed but does not include the final publisher proof-corrections or journal pagination.

Citation for the original published paper (version of record):

Fathi, M., Fontes, D B., Urenda Moris, M., Ghobakhloo, M. (2018)
Assembly line balancing problem: a comparative evaluation of heuristics and a
computational assessment of objectives
Journal of Modelling in Management, 13(2): 455-474
<https://doi.org/10.1108/JM2-03-2017-0027>

Access to the published version may require subscription.

N.B. When citing this work, cite the original published paper.

<https://creativecommons.org/licenses/by-nc/4.0/>

Permanent link to this version:

<http://urn.kb.se/resolve?urn=urn:nbn:se:his:diva-15107>

Assembly line balancing problem: a comparative evaluation of heuristics and a computational assessment of objectives

Journal:	<i>Journal of Modelling in Management</i>
Manuscript ID	Draft
Manuscript Type:	Original Article
Keywords:	Assembly line balancing, SALBP-1, Number of workstations, Workload smoothness, Heuristics

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

Assembly line balancing problem: a comparative evaluation of heuristics and a computational assessment of objectives

Abstract

Purpose – The purpose of this study is to firstly investigate the efficiency of the most commonly used performance measures for minimizing the Number of Workstations (NWs) in approaches addressing Simple Assembly Line Balancing Problem (SALBP) for both straight and U-shaped line. Secondly, this study aims to provide a comparative evaluation of 20 constructive heuristics to find solutions to the SALBP-1.

Design/methodology/approach – 200 problems are solved by 20 different constructive heuristics for both straight and U-shaped assembly line. Moreover, several comparisons have been made to evaluate the performance of constructive heuristics.

Findings – Minimizing the Smoothness Index (SI) is not necessarily equivalent to minimizing the NWs, therefore, it should not be used as the fitness function in approaches addressing the SALBP-1. Line efficiency (LE) and the idle time (IT) are indeed reliable performance measures for minimizing the NWs. The most promising heuristics for straight and U-shaped line configurations for SALBP-1 are also ranked and introduced.

Practical implications – Results are expected to help scholars and industrial practitioners to better design effective solution methods for having a most balance assembly line. This study will further help with choosing the most proper heuristic with regard to the problem specifications and line configuration.

Originality/value – There is little to no research assessing the efficiency of the common objectives for SALBP-1. This study is among the first to prove that minimizing the workload smoothness is not equivalent to minimizing the NWs in SALBP-1 studies. This work is also one of the first attempts for evaluating the constructive heuristics for both straight and U-shaped line configurations.

Paper type – Research paper

Keywords: Assembly line balancing, SALBP-1, Number of workstations, Workload smoothness, Heuristics

1. Introduction

The assembly line balancing problem aims at assigning assembly operations to a set of workstations in order to optimize some performance measures, while satisfying technological, operational, and organizational constraints [1]. According to the literature (e.g.,[2–5]), assembly line balancing problems (ALBPs) are categorized into two main groups based on their

assumptions and limitations, namely simple assembly line balancing problems (SALBPs) and generalized assembly line balancing problems (GALBPs).

The SALBP is one of the most well studied problems within the area of line balancing and the majority of the reported research on ALBPs has been targeted at SALBP [6]. An important aspect that differentiates SALBPs is the performance measure, usually referred to as objective, considered. Following the literature (e.g., [7–9]), the SALBPs are classified into two main groups based on their objective, namely: SALBP-1, which aims to minimize the number of workstations for a known, fixed and common (to all workstations) cycle time; and SALBP-2, which aims to minimize the cycle time while the number of workstations is fixed. According to the literature (e.g., [10–12]), most of research on SALBPs has been concentrated on improving solution procedures for SALBP-1.

Considering the shape of the line, the two common configurations of assembly lines are straight and U-shaped [13]. In principal, in a straight line, assembly tasks can only be assigned to operators from one end of the line (front or back) while tasks from both ends of the line can be handled by the same operator in a U-shaped line. A schematic view of straight and U-shaped line is depicted in Figure 1.

Figure 1: Straight and U-shaped assembly line configurations.

The SALBP has been shown to be NP-hard and therefore, optimal solutions to realistic sized problems cannot be found in a reasonable amount of time [14, 15]. Thus, most of the research regarding this problem has been on heuristic methods [16]. Comprehensive discussions on heuristic methods for SALBPs can be found in Fathi and Gobakhloo [12] and Battaïa and Dolgui [17].

The effectiveness and efficiency of heuristic algorithms depend, among other factors, on the fitness function used. The fitness function has a major role since, in addition to being used to identify the best solution found, it is also used to guide search for good quality solutions as it points the algorithm towards an area of the feasible region with promising solutions. Therefore, the fitness function must be closely correlated with the objective of the original problem in order to avoid losing the best found solution, due to miscalculation of the solution quality, and to avoid late convergence and consequently higher computational

times. In addition, the computation of the fitness function must be easy and fast due to the iterative nature of heuristic methods.

One of the often used fitness functions for SALBP-1 is the workload smoothness index (SI) [12]. In this paper we show that the minimization of the SI, using its common formula provided in Equation (12), may not satisfy the objective of the SALBP-1, i.e., the resulting solution may not minimize the number of workstations, regardless of the line configuration considered (straight and U-shaped). Therefore, the SI should not be used as the fitness function in the studies targeted at minimizing the number of workstations since, as shown in Section 3, solutions with worst SI may have a better NWs, and vice-versa.

Many different heuristic have been proposed overtime for the SALBP-1. Nevertheless, a comprehensive computational comparative study on their performance has never been done. In here, we shed some light on the choice and performance of 20 constructive heuristics considering 100 benchmark problem instances and two different line configurations (straight and U-shaped). We chose to test constructive heuristics as they are a very useful type of heuristics, since they can play several roles in solutions approaches: as a stand-alone method, given the ability of finding feasible solutions; as an initial solution generator, and as a foundation for complex heuristics. We also use these heuristics to compare the number of workstations used and the workload smoothness index of the solutions obtained. Thus providing empirical evidence on the inadequacy of using the smooth index as a fitness function to solve this problem.

The remainder of this paper is organized as follows. Section 2 describes the SALBP-1 in detail and provides the corresponding mathematical formulation, both for straight and U-shaped line configurations. The validity of selecting the workload smoothness as the fitness function in SALBP-1 studies is investigated in Section 3 by resorting to 100 benchmark problem instances. Section 4 reports the comparative evaluation of 20 heuristics for SALBP-1 and Section 5 presents the conclusions drawn, as well as some future research directions.

2. Problem description and formulation

The SALBP-1, first proposed by Bowman [18] and Urban [19], aims to minimize the number of workstations given a set of assembly tasks $i(i = 1, \dots, N)$ with known precedence relationships, and a set of undistinguishable workstations $j(j = 1, \dots, M_{max})$.

The general assumptions of the straight and U-shaped SALBP-1 are presented below; followed by the notation used in the mathematical programming models.

Assumptions

The assembly line is a paced line producing a single model (mass-production of one homogeneous product);

The cycle time is known, fixed, and common to all workstations;

The assembly tasks are connected by precedence constraints;

An assembly task cannot be divided among several workstations;

Only one task can be performed in a workstation at a given time;

The time to perform a task does not depend on the assigned workstation;
 A task can be performed in any workstation (workstations are equally equipped);
 The processing time of each assembly task is known and deterministic.

Table 1: Notation used in the SALBP-1 models.

i, r, k	Task
j	Workstation
C	Cycle time
N	Number of tasks
t_i	Processing time of task i
P	Set of precedence relations; for example, $(r, k) \in P$ if and only if task r is an immediate predecessor of task k
M_{max}	Upper bound on the number of workstations
y_j	1, if any task is assigned to station j ; 0, otherwise
z_{ij}	1, if task i is assigned to station j in backward direction; 0, otherwise
x_{ij}	1, if task i is assigned to station j in forward direction; 0, otherwise

Mathematical model

The mathematical model of SALBP-1 for straight line is as follows.

$$\text{Minimize } \sum_{j=1}^{M_{max}} y_j \quad (1)$$

$$\sum_{j=1}^{M_{max}} x_{ij} = 1, \quad \forall i = 1, 2, \dots, N, \quad (2)$$

$$\sum_{i=1}^N t_i x_{ij} = C y_j, \quad \forall j = 1, 2, \dots, M_{max}, \quad (3)$$

$$\sum_{j=1}^{M_{max}} j(x_{rj} - x_{kj}) \leq 0, \quad \forall (r, k) \in P, \quad (4)$$

$$x_{ij}, y_j \in \{0, 1\}, \quad \forall i = 1, 2, \dots, N; \forall j = 1, 2, \dots, M_{max}. \quad (5)$$

The objective in this problem is to minimize the total number of used workstations, which here is enforced by assigning the last task to the earliest possible station as in Equation (1). Constraints (2) ensure that each task will be assigned to exactly one workstation. Constraints (3) control the workstations' workload, by guaranteeing the total processing time of the tasks assigned to each used workstation to be at most the given cycle time. Constraints (4) impose the precedence relations amongst the assembly tasks. Finally, the domain of the variables are defined in Equation (5).

The mathematical model of SALBP-1 for U-shaped line is as follows.

$$\text{Minimize } \sum_{j=1}^{M_{max}} y_j \quad (6)$$

$$\sum_{j=1}^{M_{max}} (x_{ij} + z_{ij}) = 1, \quad \forall i = 1, 2, \dots, N, \quad (7)$$

$$\sum_{i=1}^N t_i (x_{ij} + z_{ij}) = C y_j, \quad \forall j = 1, 2, \dots, M_{max}, \quad (8)$$

$$\sum_{j=1}^{M_{max}} j (M_{max} - j + 1) (x_{rj} + x_{kj}) \geq 0, \quad \forall (r, k) \in P, \quad (9)$$

$$\sum_{j=1}^{M_{max}} j (M_{max} - j + 1) (z_{kj} + z_{rj}) \geq 0, \quad \forall (r, k) \in P, \quad (10)$$

$$x_{ij}, z_{ij}, y_j \in \{0, 1\}, \quad \forall i = 1, 2, \dots, N; \forall j = 1, 2, \dots, M_{max}. \quad (11)$$

As before the objective is to minimize the number of used workstations, Equation (1) is here renumbered as Equation (6). Constraints (7) ensure that every task is performed exactly once, either in the backward or forward direction. Similarly to constraints (3), constraints (8) ensure that the total processing time of the assigned tasks to each used workstation does not exceed the given cycle time. Constraints (9) and (10) ensure precedence constraints, more specifically, constraints (9) ensure that forward direction task assignment occurs after all predecessors of each specific task have been assigned, while constraints (10) require all successors to be assigned before a backward direction assignment is made. Finally, variables' domain is defined in Equation (11).

The objective of SALBP-1 is, by definition, the minimization of the number of workstations. Although, the number of workstations can easily be defined and modelled as a simple counter or binary variable, usually more complex fitness functions are used in research studies where a heuristic algorithm is presented. It is obvious that, the efficiency of the heuristic algorithms vastly depends on the fitness function used, as it plays a significant role in evaluating the generated solutions and in guiding the algorithm towards good solutions. Therefore, the fitness function must be closely correlated with the objective of the original problem in order to avoid late convergence and consequently higher computational times. Furthermore, optimizing the fitness function should imply optimizing the original objective function, since otherwise good or even optimal solutions may go unnoticed and therefore lost.

One of the most commonly used objective functions in SALBPs is the minimization of the smoothness index (SI), which allows for evenly distributing the workload amongst the

workstations [12, 20]. The importance and advantages of such objective have been vastly discussed in the literature (e.g., [21–24]) and are out of the scope of this work. The most common formula to compute the SI is given in Equation (12).

$$SI = \sqrt{\frac{\sum_{j=1}^M (T_{max} - T_j)^2}{M}} \quad (12)$$

where M is the number of workstations, T_{max} is the maximum workstation time $\left(\max \left\{ \sum_{i \in j} t_j; j = 1, 2, \dots, M \right\}\right)$, and T_j is the time of j^{th} workstation $\left(\sum_{i \in j} t_j; j = 1, 2, \dots, M\right)$.

In the next section we will investigate, through a set of computational experiments, whether SI satisfies the objective of SALBP-1 (minimizing the NWs) as it is being used in such studies; see for example the recent studies by Baykasoğlu [25] and Baykasoğlu and Özbakir [26].

3. Assessment of objectives to minimize the workstations

As far as we are aware, there has been no discussion in the literature regarding the applicability of the smoothness index as the optimization criterion to solve the SALBP-1. Given its wide spread use as such and the fact that it is neither equivalent nor implies the optimization of the number of workstations used, here we carry out a set of experiments to infer on the validity of doing so. The computational experiments involve both straight and U-shape assembly line configurations. For each of these configurations a set of 100 benchmark problem instances, which can be found at the assembly line optimization research homepage [27], is considered. It should be noticed that many of these problems come from industrial applications, namely Bartholdi, Gunther, Hahn, Heskiaoff, Lutz, Mukherjee, Tonge and Warnecke [28].

All instances are solved by 20 different constructive heuristics that have been collected from previous studies (e.g., [29–31]), see Table 2. All the heuristics were coded in MATLAB 2014a and run on a personal computer with a 2.6 GHz Intel Core i5 CPU and 16 GB of RAM.

The notation used to present the heuristics in Table 2, which has not been introduced before, is as follows:

IP_i : immediate predecessors of tasks i ;

P_i : set of all predecessors of tasks i ;

IS_i : immediate successors of tasks i ;

S_i : set of all successors of tasks i ;

UB_i : workstation upper bound for task i obtained as $N + 1 - \left\lceil \frac{(t_i + \sum_{r \in S_i} t_r)}{C} \right\rceil^+$, i.e., the last workstation until which task i has to be performed such that all its successors can still be performed;

LB_i : workstation lower bound for task i that is calculated as $\left\lceil \frac{(t_i + \sum_{r \in P_i} t_r)}{C} \right\rceil^+$, i.e., the first work station in which task i can be performed as all its predecessor have to be already performed.

Table 2: List of the constructive heuristics used.

Number	Notation	Definition
1	MaxTime	Longest processing time, t_i
2	MinTime	Shortest processing time, t_i
3	MaxIS	Maximum number of immediate successors, $ IS_i $
4	RP	Random priority, random i
5	MinTaskNo	Smallest task number, i
6	MaxRPW	Maximum ranked positional weight, $t_i + \sum_{r \in S_i} t_r$
7	Max (Time/UB)	Greatest (processing time divided by the upper bound), $\frac{t_i}{UB_i}$
8	MinLB	Smallest lower bound, $\left\lceil \frac{(t_i + \sum_{r \in P_i} t_r)}{C} \right\rceil^+$
9	MinSL	Minimum slack, $UB_i - LB_i$
10	MaxIP	Maximum number of immediate predecessors, $ IP_i $
11	MinUB	Smallest upper bound, $N + 1 - \left\lceil \frac{(t_i + \sum_{r \in S_i} t_r)}{C} \right\rceil^+$
12	MaxS	Greatest number of successors, $ S_i $
13	MaxARPW	Greatest average rank positional weight, $\frac{(t_i + \sum_{r \in S_i} t_r)}{(S_i + 1)}$
14	Min (UB/S)	Smallest (upper bound divided by the number of successors), $\frac{UB_i}{(S_i + 1)}$
15	Min (S/TaskSL)	Minimum (number of successors divided by task slack), $\frac{ S_i }{(UB_i - LB_i)}$
16	MaxPWS	Maximum positional weight of successor tasks, $\sum_{r \in S_i} t_r$
17	MaxTimeIS	Maximum task time of immediate successor tasks, $ IS_i $
18	MinS	Minimum total number of successor tasks, $ S_i $
19	MaxTimeS	Maximum total time of successor tasks, $T S_i $
20	MaxP	Maximum total number of predecessor tasks, $ P_i $

For illustration purposes and since it would be impossible to report full details for all the 20 solutions of the 200 instances solved, in here we provided an example where full details for two solutions for each assembly line configuration are reported. To this end we have considered the Sawyer instance, a well-known benchmark problem instance. The relative precedence network for the Sawyer’s problem is shown in Figure 2, where the number above a node represents the task processing time, in seconds, and the one within the node represents the task number. The cycle time is 41 seconds.

Table 3 reports two feasible solutions (solution 1 and solution 2) for each of the two assembly line configurations considered (straight and U-shape). The last row of Table 3 provides the SI value for each solution, computed as given by Equation (12). The first column identifies the workstations by associating a number to each one of them. The second and third columns provide two different feasible solutions for the straight line configuration; while the fourth and fifth columns provide the two feasible solutions for the U-shape line configuration. Each column (except the first) identifies the tasks to be performed in the

Figure 2: Sawyer's problem instance: precedence network.

corresponding workstation (column 1). Regarding the straight line configuration, solution 1 (column two) requires the use of 10 workstations and has a SI value of 10.601; while solution 2 (column three) requires 9 workstations and has a SI value of 11.522. Thus, according to the SI values, solution 1 would be better (10.601 is smaller than 11.522). However, since in the SALBP-1 the NWs is to be minimized, solution 2 is actually better as it requires only 9 workstations. A similar situation is exemplified for the U-shape line, since solution 1 has a SI value smaller than solution 2 (columns four and five, respectively); however, solution 2 only requires 9 workstations while solution 1 requires 10. Again, solution 2 is better, although having a larger SI value.

Table 3: Assigned tasks to workstations on straight and U-shape lines.

Workstations number	Straight line		U-shape line	
	Solution 1	Solution 2	Solution 1	Solution 2
1	1,2,3,5,10	3,16,1,10	30,10,2,28,1,5,6	3,19,10
2	4,6,7,16	4,2,12,13,5,17	4,7,8,9	11,16,18,2
3	11,12,13,17	11,14,18	11,29	12,1,4,28
4	14,18,8,9	19,20,24	12,13,14,15	13,14,20,24
5	15,19	21,25	19,18,17	21,25,5
6	20,24,25	15,22,23	3,16	15,22,23,17
7	21,22	6,7,8,9,26	20,24,25	6,7,8,30,9
8	23,26	27,29,30	26,21	29,27
9	27,28	28	22,23	26
10	29,30	—	27	—
SI	10.601	11.522	9.969	11.030

Each of the 200 instances considered has been solved by each of the 20 heuristics described

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

in Table 2. We report on the SI value and the NWs required for each of the solutions found. Here, in Tables 3 and 4, we only report two different solutions for the instances considered; nevertheless results for all problem instances obtained by each heuristic can be found in <http://goo.gl/5zzJTL>.

Tables 4 and 5 refer, respectively, to the straight line and U-shaped assembly line configurations and report both on the SI value and the NWs used. In each table we provide two solutions for the problem instances considered: solution 1 (given in columns 3-4 and 9-10) has a smaller SI value, while solution 2 (given in columns 5-6 and 11-12) requires a smaller NWs. Therefore, considering the SI as the main fitness function solution 1 would be chosen which is inconsistent with the SALBP-1 objective.

Further evidence is provided by comparing the SI results obtained by the different heuristics for the same problem instance. To avoid massive number of pairwise comparisons, we compare each heuristic results with the ones obtained by the heuristics with a higher identification number, that is the results obtained by heuristic h are compared with the results obtained by heuristics $h + 1, h + 2, \dots, H$. Therefore, a total of $\sum_{h=1}^H (h - 1)$ pairwise comparisons are performed, where, H is the total number of heuristics.

A comparison is considered to involve conflictive pairs, whenever a solution involving a lower SI value but higher NWs is preferred to a solution that although having a higher SI is indeed better as it has lower NWs. In Figures 3 and 4 we depict the number of times a conflictive pairwise comparison occurs for each problem instance for straight and U-shaped assembly lines, respectively.

As it can be seen in Figures 3 and 4, for each problem instance solved there are, on average, more than 11 cases in which the “best” solution is wrongly chosen. This happens both for straight and U-shaped line configurations. Therefore, in all these cases an inferior solution would be chosen, although a better one was found. Furthermore, although in some cases the phenomenon is not too frequent in some others it happens up to 90 times for the straight line configuration and 64 times for U-shaped line configuration, which corresponds to almost 50% and 35% of the comparisons, respectively.

Given the results obtained, it is safe to conclude that the SI (see Equation (12)) is not a good choice for the fitness function as its inaccurate evaluation of solutions will, likely lead to a poor choice of solutions. This in turn will not only lead to late convergence and consequently higher computational times, but also to the choice of a solution that is not the best one found. Therefore, we recommend the use of a different (and valid) fitness function. Among the different objectives available in the literature for SALBP-1, it can be easily proven that the line efficiency (LE) and the idle time (IT), which can be calculated, respectively, as given by Equations (13) and (14) are valid alternatives for NWs. Unlike the SI, proving the equivalency of LE and IT to the NWs is not a difficult task and does not need computational exploration. It is obvious that in the area of SALBP-1, where the cycle time is given and processing time of tasks are known with certainty, any changes in the values of Equations (13) and (14) can only be caused by the “number of workstations”, since everything else is constant. Therefore, it can be stated with certainty that minimizing the number of workstations in SALBPs of type-1 is equivalent to minimize the idle time or maximize the line efficiency.

Table 4: Solutions for the straight line configuration.

Problem	Cycle time	Solution 1		Solution 2		Problem	Cycle time	Solution 1		Solution 2	
		NWs	SI	NWs	SI			NWs	SI	NWs	SI
Arcus2	5755	29	867,862	28	1033,461	WeeMag	35	61	9,104	60	10,752
	5785	29	889,899	28	1097,889		41	60	14,650	59	16,370
	6016	28	900,591	27	1014,214		42	60	15,699	58	17,115
	6540	26	991,418	25	1137,606		49	33	5,228	32	7,112
	6837	25	1004,463	24	1097,046		50	33	6,562	32	7,646
	7162	24	1385,143	23	1515,575		52	33	8,393	32	9,170
	7520	22	924,132	21	1188,827	Barthol2	87	52	7,467	51	8,037
	7916	21	1234,790	20	1498,703		89	52	10,756	51	11,610
	8356	22	1779,336	21	1919,477		91	50	8,471	49	9,305
	8847	19	1276,417	18	1651,457		95	48	9,647	47	11,067
Arcus1	9400	19	2053,123	18	2323,971		97	47	10,163	46	13,103
	10027	17	1496,067	16	1817,392		99	48	14,898	47	15,046
	10743	16	1927,561	15	2306,204		101	46	13,114	45	13,807
	11378	15	2098,387	14	2211,087		104	44	11,954	43	12,180
	11570	15	2197,221	14	2821,288		106	44	13,281	43	14,558
	3786	23	705,213	22	757,927		109	42	13,569	41	15,050
	4454	20	785,054	19	822,348		112	42	14,713	41	16,337
	Lutz3	75	17,429	26	19,791		121	38	16,085	37	18,429
		79	11,690	23	13,951		125	37	16,213	36	21,044
		83	14,739	22	15,900		129	36	18,536	35	21,428
Barthol1		87	15,582	21	19,475		133	35	19,828	34	20,575
		92	12,304	19	14,502		137	33	11,940	32	12,539
		97	18,753	19	21,600		142	32	14,409	31	14,690
		118	18,107	15	22,9928		152	32	28,000	31	29,987
		403	92,419	15	97,401		157	29	17,956	28	19,710
		470	112,509	13	121,021		170	27	23,069	26	24,004
	Kilbrid Tonge	62	13,757	10	16,583	Lutz2	15	37	2,760	36	2,833
		160	26,811	24	28,928		18	30	2,330	29	2,722
		168	30,692	23	33,318		20	28	3,375	27	3,404
		176	31,792	22	32,913		21	27	3,610	26	3,695
Warnecke		185	35,926	21	36,555	Mukherjee	176	28	37,444	27	37,592
		195	45,071	21	48,535		183	26	27,878	25	28,136
		220	48,751	18	51,804		201	24	31,040	23	38,722
		251	45,097	15	53,902		222	21	33,565	20	37,873
		270	51,533	14	63,687		234	21	47,223	20	47,810
		293	57,711	13	70,869		281	18	61,128	17	64,231
		320	75,989	12	76,452	Rosenberg	18	9	4,496	8	5,111
	Sawyer	54	13,812	35	14,420		27	15	5,633	14	5,781
		56	12,760	33	13,620		30	14	7,540	13	7,625
		58	10,933	30	11,457		33	13	8,682	12	10,400
WeeMag		62	10,605	29	11,965		41	10	10,601	9	11,522
		65	16,386	29	16,418		47	9	8,006	8	13,143
		68	13,211	26	13,376	Michel	15	10	4,969	9	5,754
		82	17,444	22	19,533	Lutz1	2020	9	454,073	8	460,832
		86	17,254	20	17,541	Buxey	30	14	7,483	13	8,057
		92	18,253	19	21,111		33	13	8,673	12	9,814
		111	23,141	16	26,483	Heskiaoff	138	10	34,111	9	36,624
	Mertens	30	6,338	63	6,952		8	6	1,354	5	1,414
		31	6,904	62	7,5462		10	6	1,354	5	2,569
		33	8,526	61	9,111	Gunther	81	8	24,085	7	28,415

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

Table 5: Solutions for the U-shaped line configuration.

Problem	Cycle time	Solution 1		Solution 2		Problem	Cycle time	Solution 1		Solution 2	
		NWs	SI	NWs	SI			NWs	SI	NWs	SI
Arcus2	5785	28	633,315	27	876,754	Warnecke	111	17	19,846	16	23,361
	6016	27	697,933	26	995,851	WeeMag	31	63	6,781	62	7,7115
	6267	26	698,570	25	976,165		32	63	7,673	61	8,195
	6540	25	669,417	24	966,391		40	62	14,307	60	15,941
	6837	24	795,840	23	1061,722		41	62	14,307	59	16,634
	7162	23	1028,547	22	1150,053		54	32	8,708	31	9,310
	8847	19	1269,022	18	1810,323	Hahn	2004	9	580,260	8	663,836
	9400	18	1675,933	17	1986,974		2338	8	798,734	7	811,828
	10743	17	2354,341	15	2723,262	Barthol2	85	53	7,180	52	11,214
	11378	15	1929,730	14	2205,274		93	48	6,429	47	10,280
	11570	15	2101,473	14	3074,314		95	47	7,326	46	10,725
Arcus1	3985	22	687,569	21	789,094		97	46	6,500	45	10,975
	4454	20	785,054	19	904,545		104	44	10,485	43	15,003
	5048	17	693,701	16	894,809		106	43	10,188	42	14,237
	5408	16	768,244	15	937,570		112	40	8,351	39	11,967
	6309	14	1055,114	13	1307,023		118	39	13,208	38	17,697
	6842	13	1086,770	12	1485,723		129	35	14,149	34	17,156
Lutz3	6883	13	1418,948	12	1572,319		133	34	9,591	33	16,581
	83	22	10,647	21	15,226		142	32	17,306	31	19,058
	87	21	11,328	20	14,010		163	28	17,010	27	22,414
	92	20	13,928	19	19,128		170	27	23,958	26	26,621
	97	19	13,080	18	19,442	Lutz2	15	35	1,740	34	2,319
	110	17	17,756	16	23,138		17	31	1,951	30	2,822
	118	16	23,659	15	28,696		18	30	2,183	29	2,281
	127	15	21,228	14	25,472		19	28	2,211	27	2,993
	137	14	23,271	13	33,890		20	26	1,829	25	2,236
	434	15	81,822	14	114,665		21	25	2,000	24	2,188
	470	14	111,319	13	128,427	Mukherjee	176	26	21,819	25	25,458
Kilbrid	56	12	12,138	11	16,113		183	25	19,866	24	23,954
	57	12	12,138	11	16,138		192	24	22,005	23	29,281
	62	11	13,892	10	18,390		201	23	25,048	22	34,500
	79	9	22,085	8	26,537		211	22	29,339	21	39,165
	160	24	16,668	23	25,395		234	20	32,556	19	45,737
	176	22	21,660	21	25,777		248	19	42,514	18	48,756
Tonge	185	23	37,290	22	39,031		281	17	47,617	16	60,404
	195	20	27,736	19	34,614		301	16	48,172	15	71,582
	207	19	35,480	18	40,7049		30	13	6,838	12	7,359
	234	18	42,542	17	44,692		33	12	4,222	11	7,982
	251	16	46,300	15	57,926		41	10	9,969	9	11,030
	293	14	56,255	13	68,987	Buxey	30	13	6,189	12	6,879
	320	13	52,778	12	82,088		33	12	6,582	11	8,836
	54	33	8,177	32	9,303		36	11	7,077	10	7,823
	58	30	8,209	29	10,214		47	9	7,141	8	13,351
	60	30	9,791	29	10,221	Heskiaoff	138	10	35,318	9	38,231
	68	26	9,687	25	10,877	Mertens	7	6	1,354	5	1,549
	71	25	10,019	24	11,210		8	6	1,354	5	2,645
Warnecke	74	24	12,179	23	14,154	Gunther	44	13	8,142	12	9,945
	78	23	13,689	22	15,765		54	11	12,037	10	14,110
	86	20	10,747	19	11,410		61	10	9,596	9	18,726
	92	19	12,303	18	13,666		69	9	18,511	8	20,949

Figure 3: Number of conflictive pairs found for each problem in straight line configuration.

$$IT = M \times C - \sum_{i=1}^N t_i \quad (13)$$

$$LE = \frac{\sum_{i=1}^N t_i}{M \times C} \quad (14)$$

It should be noticed, however, that the SI may be an important performance measure and therefore interesting to use as a complementary objective, for example for breaking ties where more than one solution with the same number of workstations exists.

4. Comparative evaluation of heuristics

The majority of constructive heuristics for ALBP have been proposed for SALBP [31]. Constructive heuristics are very popular and can be used in three different ways, namely: as a stand-alone method capable of finding feasible solutions very quickly; as an initial solution generator for algorithms evolving or changing solutions (e.g., metaheuristics, local search, etc.); and imbedded as a part of more sophisticated heuristic/metaheuristic methods. The general assignment procedure of constructive heuristics for SALBP-1 is presented as algorithm 1.

Given the importance and vast use of constructive heuristics, it is crucial to know, or at least to have some idea, of the heuristics performance regarding to each problem type and objective. To the best of our knowledge, no comprehensive evaluation on the performance of the available constructive heuristics for SALBP-1 has been reported in the literature. In

Figure 4: Number of conflictive pairs found for each problem in U-shaped line configuration.

this section, the efficiency and performance of the 20 heuristics given in Table 2, at solving SALBP-1 for both straight and U-shaped assembly line are evaluated through a series of computational tests. The comparison is performed based on the results obtained for the 200 instances solved. Each heuristic is compared to the others with respect to two objectives namely, minimizing the number of workstations (NWs) and minimizing the smoothness index (SI). The results of comparisons are plotted in the charts of Figures 5 to 10.

Figure 5 shows the cumulative number of times, i.e., number of problem instances, each heuristic found the lowest NWs amongst the 20 heuristics, both for straight and U-shaped line configurations. In many cases, several heuristics have been able to find the same “best” solution. From the graph it can be seen that the “maximum ranked positional weight”

Algorithm 1 General assignment procedure of constructive heuristics for SALBP-1.

- Input:** Cycle time (C), precedence relationships between tasks, and tasks' time
1. Open a workstation, $CT = C$.
 2. Determine the list of candidate tasks, LCT . Task i is a member of LCT if $t_i \leq CT$ and has no immediate predecessors.
 3. If the list of candidate tasks is not empty, assign task i to the current workstation based on the priority defined by the heuristic; otherwise, go to step 5.
 4. Update the workstation cycle time ($CT = C - t_i$), remove task i from the list of precedence tasks for all the tasks that have task i as an immediate predecessor and, go to step 2.
 5. If all the tasks are assigned, terminate the process. Otherwise, go to step 1.

Figure 5: Number of problem instances for which each heuristic found the best NWs, amongst the 20 heuristic solutions obtained.

(heuristic 6) is the best performing heuristic for the straight line configuration, as it has found the smallest, or at least as small, NWs in 80 out of 100 instances solved. This heuristic is closely followed by heuristic 7, i.e., “greatest (processing time divided by the upper bound)”, which had such performance for 78 instances. It is still worth mentioning heuristic 1, the “longest processing time”, as it has done so for 71 instances. Heuristics number 7 and 1 stand out since they have achieved the “best” value in 94 and 92 instances for U-shaped line, respectively. This number comes down to 75 for the third best performing heuristic, heuristic 6. It is interesting to notice that the three top performing heuristics are the same for both line configurations, although not exactly in the same order.

Another important aspect regarding the heuristics’ robustness is how good or bad it is when not performing well. To analyse this, all the heuristics are ranked from 1 to the end in each problem instance solved, considering the NWs found. It is obvious that there could be more than one heuristic with the same rank in each problem instance. In Figure 6 we plotted for each heuristic the average of its ranks, out of the 100 instances solved, for each line configuration. Since we are minimizing the NWs, the closer to 1 the rank is, the better the heuristic performance.

As it can be seen in Figure 6, the top 3 heuristics remain the same. Moreover, the ranking among them is also the same, namely heuristics 6, 7, and 1 for the straight line configuration and 7, 1, and 6 for the U-shaped line configuration.

Regarding the NWs, for each heuristic we have also compared the NWs found with the optimal one, see Figure 7. The optimum NWs can be found at the assembly line optimization research homepage [27]. This comparison is not a relative comparison but

Figure 6: The average of the ranks of heuristics in finding NWs, out of the 100 instances solved, for both line configurations.

an absolute comparison. As it can be seen in Figure 7, the top two heuristics remain unchallenged, although in the U-shaped case they swapped positions. However, the third performing heuristic, for both configurations is different. For the straight line configuration, the heuristic based on “greatest number of successors” (number 12) is now third, while heuristic number 1 became fourth. In regards to the U-shaped configuration the third best heuristic is now the one based on the “minimum slack” (number 9) and number 6 became fourth. It is interesting to notice that, in all the three comparisons made for NWs the worst performing heuristic is heuristic number 2 (“shortest processing time”). It is also worth mentioning that heuristics are more successful at finding an optimum for straight line instances than for U-shaped ones. An optimum was found in 43 out of 100 problem instances solved for straight line while, this number came down to 26 for U-shaped line.

According to Figures 5 to 7, it is safe to say that heuristic 6 and 7 are the recommended ones for straight line instances, as they have consistently perform better at finding the “best” NWs. For U-shaped instances the recommended ones are heuristics 7 and 1.

Similar statistics have been collected regarding the heuristics performance in finding the best SI. Figure 8 shows the number of times each heuristic was able to match the best SI found amongst the 20 heuristics. The best performing heuristics for straight line were heuristics 6 (“maximum ranked positional weight”), 7 (“greatest (processing time divided by the upper bound)”), and 9 (“minimum slack”) and 13 (“greatest average rank positional weight”). For the U-shaped line there was almost always more than one heuristic with the same performance ranking: heuristics 5 (“smallest task number”) and 13 (“greatest average rank positional weight”), heuristics 7 (“greatest (processing time divided by the

Figure 7: Number of times each heuristic was able to find the optimum NWs, considering 100 problem instances both for straight line and U-shaped line configurations.

upper bound)”) and 9 (“minimum slack”), and heuristics 1 (“longest processing time”) and 17 (“maximum task time of immediate successor tasks”). It is noteworthy that, although heuristic 5 obtained very good results for U-shaped line, it has not been able to find even one good solution for straight line.

Just like the case of NWs, we have averaged the ranks of the heuristics in finding the SI. From Figure 9, it can be observed that the best results for the straight line configuration are due to heuristics 9 (“minimum slack”), 16 (“maximum positional weight of successor tasks”), and 11 (“smallest upper bound”); while for U-shaped line are due to heuristics 13 (“greatest average rank positional weight”), 5 (“smallest task number”) and 16 (“maximum positional weight of successor tasks”), and 9 (“minimum slack”).

Since no optimum SI value is available, we don’t know how close the solutions obtained are to optimality. From the observation of Figures 8 and 9, it can be concluded that the performance of the heuristics regarding SI is not as consistent as it was for NWs. Therefore, a conclusion regarding the best heuristics for achieving good SI values seem to be difficult. However, heuristic 9 is in the top three for both indicators, regarding straight line. Regarding U-shaped line configuration the ranking of the top three heuristics is 13, 5, and 9.

One last comparison of the heuristics’ performance is obtained by finding how many times each heuristic found simultaneously the best value for both NWs and SI (i.e., found a non-dominated solution), see Figure 10. As it can be seen, the best results for straight line were obtained by heuristic 6 (“maximum ranked positional weight”), 7 (“greatest (processing time divided by the upper bound)”), and 13 (“greatest average rank positional weight”). Regarding U-shaped line, the best heuristics are 5 (“smallest task number”) and 13 (“great-

Figure 8: Number of problem instances for which each heuristic found the best SI, amongst the 20 heuristic solutions obtained.

Figure 9: The average of the ranks of heuristics in finding the SI, out of the 100 instances solved, for both line configurations.

Figure 10: Number of times each heuristic found a non-dominated solution, considering simultaneously the NWs and the SI.

est average rank positional weight”), 7 (“greatest (processing time divided by the upper bound)”), and 17 (“maximum task time of immediate successor tasks”).

5. Conclusion and future work directions

This study empirically investigates the adequacy of using the smoothness index (SI) as the fitness function to solve the simple assembly line balancing problem of type-1 (SALBP-1). Since minimizing the SI is neither equivalent to minimizing the NWs, nor implies it we recommend not using the SI as the fitness function. Otherwise, it may happen that (i) the best solution found is lost, as the method misidentifies the solution with the best SI as the best solution and not the one with the best NWs; (ii) it may guide the search away from best solutions and thus lead to late convergence of the algorithm; and (iii) computational requirements become larger. Due to the above explanations, the selection of the SI as the fitness function when solving the SALBP-1 is not recommended. This has been illustrated by solving a set of 100 benchmark problem instances, each for straight and U-shaped line configurations.

Researchers should use instead the idle time, which should be minimized, or the line efficiency, which should be maximized, since both are equivalent to minimizing the number of workstations in SALBP-1.

Additionally, a comparative evaluation of 20 constructive heuristics is presented, again both for straight and U-shaped line configurations. All the heuristics were compared using two objective functions namely, NWs (which is the SALBP-1 objective function) and SI.

The heuristics were compared regarding the number of times each was able to find the best solution amongst the 20 heuristic solutions obtained, and the average of the ranks of heuristics, out of the 100 instances solved. Regarding the first objective (NWs) we were also able to compare the solution obtained by each heuristic with an optimal solution, since one is known. This study allowed for the conclusion that for SALBP-1, i.e., minimizing NWs, heuristics 6 “maximum ranked positional weight” and 7 “greatest (processing time divided by the upper bound) undoubtedly produce better results for the straight line configuration and heuristics 7 “greatest (processing time divided by the upper bound) and 1 “longest processing time” are the recommended ones for U-shaped line configuration. The conclusions regarding the minimization of SI are more difficult and weak; nevertheless it seems that the best performing heuristics are 9 “minimum slack” for straight line configuration and 13 “greatest average rank positional weight” and 5 “smallest task number” for U-shaped line configuration.

A final comparison was performed by simultaneously considering both objectives and looking at the number of non-dominated solutions each heuristic was able to find. The results obtained showed that the best heuristics are different for straight and U-shaped line configurations; however two of the heuristics namely, 7 “greatest (processing time divided by the upper bound)” and 13 “greatest average rank positional weight”, can provide a good solution to both line configurations.

The computational time of the heuristics, although slightly different, it is quite small (merely a few seconds) for all heuristics.

Similar studies should be performed for other ALBP problems with considering different line configurations. Moreover, such studies may be useful to evaluate the efficiency of heuristics in other scheduling problems, e.g., job shop scheduling, considering these or other constructive heuristics.

Conflict of interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] R. Benzer, H. Gökçen, T. Cetinyokus, and H. Cerçioğlu, “A network model for parallel line balancing problem,” *Mathematical Problems in Engineering*, vol. 2007, 2007.

[2] T. Koltai, V. Tatay, and N. Kalló, “Application of the results of simple assembly line balancing models in practice: the case of a bicycle manufacturer,” *International Journal of Computer Integrated Manufacturing*, vol. 27, no. 9, pp. 887–898, 2014.

[3] C. Becker and A. Scholl, “A survey on problems and methods in generalized assembly line balancing,” *European journal of operational research*, vol. 168, no. 3, pp. 694–715, 2006.

[4] N. Boysen, M. Fliedner, and A. Scholl, “A classification of assembly line balancing problems,” *European journal of operational research*, vol. 183, no. 2, pp. 674–693, 2007.

[5] M. Alavidoost, M. Tarimoradi, and M. F. Zarandi, “Fuzzy adaptive genetic algorithm for multi-objective assembly line balancing problems,” *Applied Soft Computing*, vol. 34, pp. 655–677, 2015.

- [6] A. Scholl and C. Becker, "State-of-the-art exact and heuristic solution procedures for simple assembly line balancing," *European Journal of Operational Research*, vol. 168, no. 3, pp. 666–693, 2006.
- [7] M. Ritt and A. M. Costa, "Improved integer programming models for simple assembly line balancing and related problems," *International Transactions in Operational Research*, 2015.
- [8] R. Esmailbeigi, B. Naderi, and P. Charkhgard, "The type e simple assembly line balancing problem: A mixed integer linear programming formulation," *Computers & Operations Research*, vol. 64, pp. 168–177, 2015.
- [9] M. Ariffin, M. K. Anuar, M. Fathi, and N. Ismail, "A new heuristic method to solve straight assembly line balancing problem," *Pertanika Journal of Science & Technology*, vol. 20, no. 2, pp. 355–369, 2012.
- [10] J. Bautista and J. Pereira, "A dynamic programming based heuristic for the assembly line balancing problem," *European Journal of Operational Research*, vol. 194, no. 3, pp. 787–794, 2009.
- [11] N.-C. Wei and I.-M. Chao, "A solution procedure for type e simple assembly line balancing problem," *Computers & Industrial Engineering*, vol. 61, no. 3, pp. 824–830, 2011.
- [12] M. Fathi and M. Ghobakhloo, "A technical comment on "a review on assembly sequence planning and assembly line balancing optimisation using soft computing approaches"," *The International Journal of Advanced Manufacturing Technology*, vol. 71, no. 9-12, pp. 2033–2042, 2014.
- [13] J. Nilakantan and S. Ponnambalam, "Robotic u-shaped assembly line balancing using particle swarm optimization," *Engineering Optimization*, vol. 48, no. 2, pp. 231–252, 2016.
- [14] R. Pitakaso, "Differential evolution algorithm for simple assembly line balancing type 1 (salbp-1)," *Journal of Industrial and Production Engineering*, vol. 32, no. 2, pp. 104–114, 2015.
- [15] T. Pape, "Heuristics and lower bounds for the simple assembly line balancing problem type 1: Overview, computational tests and improvements," *European Journal of Operational Research*, vol. 240, no. 1, pp. 32–42, 2015.
- [16] M. Fathi, M. Ariffin, and N. Ismail, "A note on "a multi-objective genetic algorithm for solving assembly line balancing problem"," *The International Journal of Advanced Manufacturing Technology*, vol. 50, no. 5-8, pp. 771–773, 2010.
- [17] O. Battaia and A. Dolgui, "A taxonomy of line balancing problems and their solution approaches," *International Journal of Production Economics*, vol. 142, no. 2, pp. 259–277, 2013.
- [18] E. H. Bowman, "Assembly-line balancing by linear programming," *Operations Research*, vol. 8, no. 3, pp. 385–389, 1960.
- [19] T. L. Urban, "Note. optimal balancing of u-shaped assembly lines," *Management Science*, vol. 44, no. 5, pp. 738–741, 1998.
- [20] X. Hu, Y. Zhang, N. Zeng, and D. Wang, "A novel assembly line balancing method based on pso algorithm," *Mathematical Problems in Engineering*, vol. 2014, 2014.
- [21] T. Kellegöz, "Balancing lexicographic multi-objective assembly lines with multi-manned stations," *Mathematical Problems in Engineering*, vol. 2016, 2016.
- [22] A. Mozdgir, I. Mahdavi, I. S. Badeleh, and M. Solimanpur, "Using the taguchi method to optimize the differential evolution algorithm parameters for minimizing the workload smoothness index in simple assembly line balancing," *Mathematical and Computer Modelling*, vol. 57, no. 1, pp. 137–151, 2013.
- [23] A. Nourmohammadi and M. Zandieh, "Assembly line balancing by a new multi-objective differential evolution algorithm based on topsis," *International Journal of Production Research*, vol. 49, no. 10, pp. 2833–2855, 2011.
- [24] A. C. Nearchou, "Maximizing production rate and workload smoothing in assembly lines using particle swarm optimization," *International Journal of Production Economics*, vol. 129, no. 2, pp. 242–250, 2011.
- [25] A. Baykasoglu, "Multi-rule multi-objective simulated annealing algorithm for straight and u type assembly line balancing problems," *Journal of Intelligent Manufacturing*, vol. 17, no. 2, pp. 217–232, 2006.
- [26] A. Baykasoglu and L. Özbakır, "Discovering task assignment rules for assembly line balancing via genetic programming," *The International Journal of Advanced Manufacturing Technology*, vol. 76, no. 1-4, pp. 417–434, 2015.

[27] A. Scholl, N. Boysen, M. Fliedner, and R. Klein, "Homepage for assembly line optimization research," 2007.

[28] A. Scholl *et al.*, "Data of assembly line balancing problems," tech. rep., Darmstadt Technical University, Department of Business Administration, Economics and Law, Institute for Business Studies (BWL), 1995.

[29] M. Fathi, M. J. Álvarez, and V. Rodríguez, "A new heuristic-based bi-objective simulated annealing method for u-shaped assembly line balancing," *European Journal of Industrial Engineering*, vol. 10, no. 2, pp. 145–169, 2016.

[30] M. C. O. Moreira, M. Ritt, A. M. Costa, and A. A. Chaves, "Simple heuristics for the assembly line worker assignment and balancing problem," *Journal of heuristics*, vol. 18, no. 3, pp. 505–524, 2012.

[31] T. Kellegöz and B. Toklu, "A priority rule-based constructive heuristic and an improvement method for balancing assembly lines with parallel multi-manned workstations," *International Journal of Production Research*, vol. 53, no. 3, pp. 736–756, 2015.