VOL. 11, NO. 10, MAY 2016 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

METHODOLOGY FOR SOLVING TWO-SIDED ASSEMBLY LINE BALANCING IN SPREADSHEET

Salleh Ahmad Bareduan and Salem Abdulsalam Elteriki
Department of Manufacturing and Industrial Engineering, University Tun Hussein Onn, Malaysia
E-Mail: salematriki@yahoo.com

ABSTRACT

In this paper, modification of simple assembly line balancing spreadsheet was done to accommodate the methodology for solving two-sided assembly line balancing problem. In the two sided assembly line balancing problem, tasks are divided into three main categories; depending on sides position either right side (R), left side (L) or can be positioned at any sides (E). In addition, random task selection is added to conventional priority rules to present the effect of multi-solutions on two-sided assembly line balancing. Tests conducted to evaluate the performance of the spreadsheet methodology has shown that the introduction of the random priority rule has generated better performance results for the two-sided assembly line balancing problem.

Keywords: one side assembly line balancing, two sided assembly line balancing, spread- sheets, random rule.

INTRODUCTION

Assembly line is widely used in various production systems to produce high-volume products, such as vehicles, buses and trucks. The goal of introducing an assembly line into a plant is to maximize efficiency and keep costs efficient. The problem related to optimally distributing assembly tasks among all workstations with respect to certain objectives, is called an assembly line balancing problem (ALBP) (Backer and Scholl, 2006).

Assembly line can be one-sided or two-sided. One-sided assembly lines are most widely studied in ALBP while less attention is paid to two-sided assembly line balancing problems (2-SALBP). The 2-SALBP is an extension of simple ALBP when different assembly tasks are carried out on the same product, in parallel, on both left and right-hand sides of the line. Compared to simple assembly line, two-sided assembly lines can shorten the line length, lower the cost of tools and fixtures, and reduce material handling and operator movement (Bartholdi, 1993). Therefore, the 2-SALBP has prominent research significance and practical value.

In spite of these preface benefits, some complex restrictions exist when balancing two-sided assembly lines, such as operation direction constraints, positional, positive–negative zoning constraints, and synchronous task constraints. Hence, the 2-SALBP is much more complex and the development of new balancing techniques is needed.

The design and balancing of two-sided assembly lines was first studied by Bartholdi (1993) using a balancing algorithm based on the first-fit rule (FFR). Kim *et al*. (2000) applied a Genetic Algorithm (GA) to solve 2-SALBP with positional constraints. Wu *et al*. (2007) and Hu *et al*. (2010) introduced a branch and bound algorithm to minimize the line length of a two-sided assembly line. Hu *et al*. (2008) proposed a station-oriented enumeration algorithm that was integrated with the Hoffmann heuristic to develop a system to solve 2-SALBP. Baykasoglu and Dereli (2006) made the first attempt to show how an Ant Colony Heuristic (ACH) could be applied to solve 2-SALBP with zoning constraints. Özcan and Toklu (2008) presented a Tabu Search (TS) algorithm for 2-SALBP and considered a line efficiency and smoothness index as performance criteria. Özcan and Toklu (2009) developed a pre-emptive goal programme for precise goals and a fuzzy goal programme for imprecise goals to minimize the number of work stations.

Özcan and Toklu (2009) investigated a Mixed Integer Programming (MIP) model and proposed a Simulated Annealing (SA) algorithm for the mixed-model 2-SALBP with zoning, positional, and synchronism constraints. Simaria and Vilarinho (2009) presented an Ant Colony Optimization (ACO) algorithm to address the mixed-model 2-SALBP, and built a balancing solution that considered precedence, zoning, capacity, and side and synchronism constraints. Ozbakir and Tapkan (2011) presented the first Bee Algorithm (BA) application to 2-SALBP and employed fuzzy mathematical programming. Ozcan (2010) was the first to consider 2-SALBP with stochastic task times and presented a chance-constrained, piecewise-linear, mixed integer program (CPMIP). Tapkan *et al*. (2012) considered 2-SALBP more realistically by employing positional, zoning, and synchronous task constraints and proposed a fuzzy approach. They introduced a mathematical programming model for fully constrained 2-SALBP in order to describe the problem formally. Due to the complexity of the problem, swarm intelligence based bee algorithm and Artificial Bee Colony (ABC) algorithms were implemented to evaluate the performance.

Ragsdale and Brown (2004) introduced a simple efficient approach to implement the line balancing problem in spread-sheets by array formula to provide an interesting and easily understood technique. Weiss (2013) combined the array formula approach with an interesting simple precedence coding system to present an efficient spreadsheet for performing assembly line balancing using priority rules method. He presented task coding to show completed task by subtract 1 from the task has had its precedencies met (code=0) and to ensure the task will not

VOL. 11, NO. 10, MAY 2016 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

be scheduled a second time. The tasks do not complete in one iteration take 1 off next time.

Referring to the literature review, no published paper has dealt with solution methodology for two-sided assembly line balancing problems in spreadsheet using a simple coding system. The organization of this paper is as follows: the next section presents modelling and modifying of spreadsheet and Section 3 illustrates selecting tasks using random method. Section 4 discusses the results.

Modification of the spreadsheet model

The proposed spreadsheet was established on a small size two-sided assembly line balancing problem P12 (twelve task problem) shown in Figure-1.

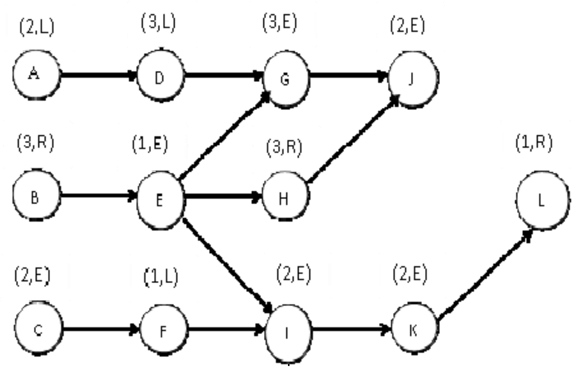

Figure-1. Example of a two-sided line balancing problem.

Referring to Figure-1, the tasks processing times are indicated by the numbers in parenthesis above the task nodes and they are in minutes. The cycle time is set at 8 minutes. In this study, two-sided assembly line approach was used in which tasks are assigned on both sides. Weiss (2013) illustrated three steps in each iteration of the solution executions for single sided ALBP solution using spreadsheet. In this paper, specifically for the 2-SALBP it is necessary to add a 4th Step to select tasks and assignment for both sides in each iteration. Therefore, lookup criterion needed adequate operation direction of tasks in order to function for solving the 2-SALBP.

Looking up criterion for the side direction

A side direction column related to task operation direction was added in Figure-2 under the column heading of position constraint; where R is the right side, L indicates the left side, and E represents either R or L side. For lookup criterion, the side direction was used as a condition to the distinct operation direction tasks, in order to assign tasks to the right or left-hand side in balancing the line.

Left side. Row 57 was added in Figure-3 to demonstrate the left side workstations in two sided assembly line balancing. For selecting a task to be assigned to the first station on the left side, the following formula was used:

= IF(AND(D46=0,D49=0),” ”,IF(D46>=D49,D47,D50)

Firstly, the lookup criterion checked for the L side row 46 and looked up the criterion for E side of row 49. If the two cells equalled zero, then no task assignment in the first iteration in the work station, if not compare between two lookup criterion for L side and E side and assign the task that had bigger lookup criterion value. The second iteration used the following formula:

=IF(AND(D57<>” “,D57=D75),D57,
IF(AND(E46=0,E49=0),” “, IF (E46 >= E49, E47, E50))

If there was a task in the first iteration and the L side was not completed yet, the same task will be continued on second iteration. If not, check for possible new task assignment from the second iteration lookup criterion for the L side and E side. The bigger lookup criterion value from either L or E will be chosen as the assigned task in the second iteration.

Right side. Row 69 was added in Figure 3 to execute the right side workstation assignment in two sided assembly line balancing. To select the task and assign to workstation 2 on the right, we used the following formula:

=IF(AND(D49=0,D52=0,D64=0,” ”,
IF(AND(D49<>0,D50<>D57,D49>D64), D50,
IF(AND(D53<>” ”,D52>D64),D53,
IF(D65<>” ”,D65,” “))

First, check the lookup criterion values for E and R sides. If there are no available tasks to assign, then the task assignment cell will be empty. If there are values bigger than zero in the lookup criterion for E sides and the tasks was not already assigned to the L side and the lookup criterion for E side is bigger than the lookup criterion for the R side, then assign the E side task at the second workstation. If not, check for the task in the lookup criterion for the R side and assign it to the second workstation. Otherwise no task will be assigned to the first iteration in the second workstation. The second iteration uses the following formula:

=IF(AND(D69<>” “,D69=D75),D69,
IF(AND(E49=0,E46=0,E52=0),” ”,
IF(AND(E49<>0,E50<>,E49>E64),E50,
IF(AND(E53<>” “,E52>E64)E53, IF(E65<>” “,E65,” “)

Either side. Row 50 was created to allow a chance that tasks with E side directions are assigned to the left or right side using the following formula:

VOL. 11, NO. 10, MAY 2016 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

=IF(D49=0," ",VLOOKUP(D49,$A$6:$B$17,2,FALSE))

The lookup criterion at cell D49 was referred. If there was no E task available to represent E side, no E task will be selected as assignment candidate. Otherwise, the selected E task will be placed at cell D50 as assignment candidate.

Row 52 was added to identify the second E task candidate and to avoid repeatedly assigning the same task to the right side. To identify the second E task candidate, the following formula was used:

=IF(D52=0," ",VLOOKUP (D52,$A$6:$B$17,2,FALSE))

The lookup criterion for cell D52 was checked. If it was equal to zero, ignore any task and keep cell D53 empty; otherwise, look at the first column of the lookup criterion, match the number to the task in the second column and put the task in cell D53. This was all done in the first iteration. Similarly, E tasks in the second iteration are selected as candidate depending on the highest lookup criterion.

	B	C	D	E	F	G	H	I	J	K	L	M
1	Data	OK										
2												
3				Priority rule/ Random		0	Priority Rule					Random
4		Cycle time	6				1	2	3	4	5	6
5	Lookup Criterion	Task	Time	e prdcssor s	Tie Prevention Code	Position Constrain t	Longest Operation Time	Most Following Tasks	Ranked Positiona l Weight	Shortest Operation Time	Least Following Tasks	Random Selection Tasks
6	7.99999	A	2.00	--	0.00001	L	2.00	3	10.00	0.50	0.25	8
7	0.99998	B	3	--	0.00002	R	3.00	7	17.00	0.33	0.13	1
8	9.99997	C	2	--	0.00003	E	2.00	4	9.00	0.50	0.20	10
9	9.99996	D	3	A	0.00004	L	3.00	2	8.00	0.33	0.33	10
10	7.99995	E	1	B	0.00005	E	1.00	6	14.00	1.00	0.14	8
11	3.99994	F	1	C	0.00006	L	1.00	3	6.00	1.00	0.25	4
12	3.99993	G	3	D,E	0.00007	E	3.00	1	5.00	0.33	0.50	4
13	3.99992	H	3	E	0.00008	R	3.00	1	5.00	0.33	0.50	4
14	1.99991	I	2	E,F	0.00009	E	2.00	2	5.00	0.50	0.33	2
15	5.99990	J	2	H,G	0.0001	E	2.00	0	2.00	0.50	1.00	6
16	4.99989	K	2	I	0.00011	E	2.00	1	3.00	0.50	0.50	5
17	8.99988	L	1	K	0.00012	R	1.00	0	1.00	1.00	1.00	9
18			25.00									

	B / C	E	F	G	H	I	J	K	L	M	N	O	P	Q
19	Iterations (left to right beginning in column D)													
20														
21	Task Code (-1= Completed, 0 = Ready, N>0 = number of unmet precedences)													
22	Task	Iteration 1	Iteration 2	Iteration 3	Iteration 4	Iteration 5	Iteration 6	Iteration 7	Iteration 8	Iteration 9	Iteration 10	Iteration 11	Iteration 12	
23	A	0	0	-1	-2	-2	-2	-2	-2	-2	-2	-2	-2	
24	B	0	-1	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	
25	C	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	
26	D	1	1	1	0	0	0	-1	-1	-1	-1	-1	-1	
27	E	1	1	0	-1	-2	-2	-2	-2	-2	-2	-2	-2	
28	F	1	0	0	0	-1	-2	-2	-2	-2	-2	-2	-2	
29	G	2	2	2	2	1	1	0	-1	-2	-3	-3	-3	
30	H	1	1	1	1	0	0	-1	-2	-2	-2	-2	-2	
31	I	2	2	2	2	1	0	0	0	-1	-1	-1	-1	
32	J	2	2	2	2	2	2	2	1	1	0	-1	-1	
33	K	1	1	1	1	1	1	1	1	0	0	-1	-2	
34	L	1	1	1	1	1	1	1	1	1	1	1	0	
35														
36	Select the next task													
37	Step 1:Time available	6	4.00	3.00	2.00	2.00	1.00	3.00	3.00	1.00	0.00	4.00	4.00	3.00
38	Step 1: Number of feasible tasks (overall)	3	2	2	1	0	0	2	1	0	0	0	1	
39	Step 2: (Possibly adjusted) available time	6.00	4.00	3.00	2.00	2.00	6.00	3.00	3.00	1.00	6.00	4.00	4.00	
40	Step 2: Station Number	1	1	1	1	1	3	3	3	3	5	5	5	

Figure-2. Spread-sheet model for adding direction constraints.

	B	E	F	G	H	I	J	K	L	M	N	O	P
41													
42	Assinging tasks into sided												
43	Step 1: Number of feasible tasks (overall)	3	2	1	1	1	0	0	2	1	0	0	1
44	Step 2: (Possibly adjusted) available time	6.00	4.00	3.00	3.00	2.00	1.00	6.00	4.00	3.00	2.00	6.00	3.00
45	Step 2: Station Number	1	1	1	1	1	1	3	3	3	3	5	5
46	Step 3: Lookup criterion (for L)	0.99999	7.99994	0.99999	0.00000	0.00000	0.00000	1.99996	1.99996	0.00000	0.00000	0.00000	0.00000
47	Step 3: Activity name (for L)	A	F	A				D	D				
48	Step 3: Activity time (for L)	2.00	1.00	2.00				3.00	3.00				
49	Step 3: Lookup criterion (for E)	5.99997	0.00000	0.00000	1.99995	6.99991	0.00000	5.99989	0.00000	0.00000	0.00000	8.99993	8.99990
50	Step 3: Activity name (for E)	C			E	I		K				G	J
51	Step 3: Activity time (for E)	2.00	0.00	0.00	1.00	2.00	0.00	2.00	0.00	0.00	0.00	3.00	2.00
52	Step 3: Lookup criterion (for E)	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
53	Step 3: Activity name (for E)												
54	Step 3: Activity time (for E)		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
55	Step 4: Task selection (Left Side)												
56	Station Number	1.00	1.00	1.00	1.00	1.00	1.00	3.00	3.00	3.00	3.00	5.00	5.00
57	Activity name	C	F	A	A	A		K	D	D	D	G	J
58	Activity time/remaining time	2.00	1.00	2.00	2.00	1.00	0.00	2.00	3.00	2.00	1.00	3.00	2.00
59													
60													
61													
62													
63													
64	Step 3: Lookup criterion (for R)	0.99998	0.00000	0.00000	0.00000	0.00000	0.00000	5.99992	9.99988	9.99988	0.00000	0.00000	0.00000
65	Step 3: Activity name (for R)	B						H	L	L			
66	Step 3: Activity time (for R)	3.00						3.00	1.00	1.00			
67	Step 4: Task selection (Right Side)												
68	Station Number	2.00	2.00	2.00	2.00	2.00	2.00	4.00	4.00	4.00	4.00	6.00	6.00
69	Activity name	B	B	B	E	I	I	H	H	L			
70	Activity time/remaining time	3.00	1.00	0.00	1.00	2.00	1.00	3.00	1.00	1.00	0.00	0.00	0.00
71													
72	Station completion time clock	2.00	3.00	3.00	4.00	5.00	6.00	8.00	9.00	10.00	11.00	14.00	16.00
73	Completed Task	C	F	B	E	A	I	K	H	L	D	G	J
74													
75	Incomplete Task	B	B	A	A	I		H	D	D			
76	Additional time required	1.00	0.00	2.00	1.00	1.00	0.00	1.00	2.00	1.00	0.00	0.00	0.00
77													
78													

Figure-3. Spread-sheet model for adding left and right sides.

Assigning tasks on two sides

As shown in Figure-3, for all direction tasks assigned to the right or left side, the excel formula for array was used. A side direction formula was added to lookup criterion to ensure that only tasks related to direction would be chosen in the rows. The example formula for L Lookup Criterion is as the following:

D46={=MAX(IF($F$6:$F$17="L",1,0)*IF(D23:D34=0, $A$6:$A$17,0))}.

This formula identifies the highest lookup values only for L tasks which fulfilled the precedence constraint. The precedence constraint checking at cells D23 to D34 follows the method suggested by Weiss (2013). The similar formula is used to identify R and E tasks candidates at all iterations. The R and E tasks identification was added, as an additional feature compared to the simple assembly line balancing spreadsheet method.

Multiple priority rules

The multiple priority rules by Weiss (2013) were used to decide which task was assigned to a workstation when multiple tasks were fit for an assignment. The one-sided ALBP methodology recommended five rules as shown in Figure-2 identified as Priority Rule 1 to 5.

Random solution

A random rule was added to the multiple priority rules in Weiss (2013) spreadsheet to retain the system capability of using the existing priority rules as well as generating multiple solutions with random rule. This is shown in Figure-2 under the column heading of Random Selection Tasks. The formula to generate the random priority numbers is:

=RANDOMBETWEEN (1, 10).

The solution change randomly for both sides when number 6 was entered in cell G3. To continue generating different random solutions the user needs to enter a number in any cell in spreadsheet.

The cells from M6 to M17 generate random numbers from 1 to 10, so look up criteria will also change depending on the random numbers. The selection of task was influenced by the order of look up criteria when the task is available to be assigned. Cell B6 has the following formula:

=IF(OFFSET(G6,0,$G$3)=0,0.1-F6,OFFSET (G6,0,$G$3)-F6).

To avoid negative lookup criterion value, the statement '0.1 – F6' is used in the above formula, if the priority value of a task equalled zero. Changing the value at cell G3 will result to a new solution corresponding to the selected priority rules. Priority rules from 1 to 5 will produce only one solution for each rule applied to the 2-SALBP. On the other hand, random priority rule will generate different solution when the priority values change following the generated random values.

Discussion of results

The problem in Figure-1 was used to generate line balancing solutions using the cycle time of 8 minutes. The solutions for priority rule 1 and random priority rule are tabulated in Table-1. The solution for the random priority rule was selected from the best solution obtained after conducting a few trials of random number generations. The summary results for all the priority rules are shown in Table-2. To test the capability of the random priority rule to generate better solution than the priority rule 1 to 5, the same 2-SALBP example was solved again using the spreadsheet method with cycle time of 6 minutes. The result summary is shown in Table-3.

Table-1. Example of 2-SALBP solutions with cycle time 8 minutes.

	Priority Rule 1		**Random Priority Rule**	
Left Side	Workstation1	Workstation2	Workstation1	Workstation2
	ADEF	GIK	ADG	JFIK
Right Side	Workstation3	Workstation4	Workstation3	Workstation4
	BC	HJL	BEH	CL
No. of workstations	4		4	
Efficiency	78%		78%	

Table-2. Summary of 2-SALBP solutions with cycle time 8 minutes.

	Rule 1	**Rule 2**	**Rule 3**	**Rule 4**	**Rule 5**	**Random**
No. of workstations	4	4	4	5	4	4
Efficiency	78%	78%	78%	63%	78%	78%

Table-3. Summary of 2-SALBP solutions with cycle time 6 minutes.

	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Random
No. of workstations	6	6	6	6	6	5
Efficiency	69%	69%	69%	69%	69%	83%

From Table 3, it is noted that the best solution is from the random priority rule with a total of five workstations and 83% efficiency. It can be concluded that the random priority rule introduced in the spreadsheet method has the capability to produce better solution than the conventional priority rule used by Weiss (2013). This is mainly because the random priority rule can generate many different solutions as the random numbers changed. As a result the best solution can be selected from many different alternative solutions.

CONCLUSIONS

This paper introduced an effective approach for modelling and solving two-sided assembly line balancing problems in spreadsheet using operation direction. Some modifications were made to original spreadsheet for the simple assembly line to accommodate two-sided assembly line balancing. A random priority rule was introduced as an additional feature to the existing priority rules to allow flexibility in generating as many possible alternative solutions. The best solution from all the alternative solutions is chosen as the final solution. This flexibility makes the two sided assembly line balancing solving methodology more efficient. This model was applied in a spread-sheet to make it easier than other complicated models, like integer programming. For extension of this work, heuristic algorithms are suggested to be applied in the spreadsheets to support the application of the two sided assembly line balancing solution methodology in future work.

ACKNOWLEDGEMENTS

This publication is partially supported by Fundamental Research Grant Scheme of Malaysia and Post Graduate Studies Centre of Universiti Tun Hussein Onn Malaysia.

REFERENCES

[1] Bartholdi, J. J. (1993). Balancing two-sided assembly lines: A case study. International Journal of Production Research, 31, 2447–2461.

[2] Baykasogˇlu, A., & Dereli, T. (2008). Two-sided assembly line balancing using an ant colony based heuristics. International Journal of Advanced Manufacturing Technology, 36, 582–588.

[3] Becker C, Scholl A (2006). A survey on problems and methods in generalized assembly line balancing. European Journal of Operation Research. 168(3):694-715.

[4] Hu, X., Wu, E., & Jin, Y. (2008). A station-oriented enumerative algorithm for two-sided assembly line balancing. European Journal of Operational Research, 186(1), 435–440.

[5] Howard J. Weiss (2013)"Implementing Line Balancing Heuristics in Spreadsheets". INFORMS Transactions on Education. 13(2), pp. 114-125.

[6] Kim, Y. K., Kim, Y., & Kim, Y. J. (2000). Two-sided assembly line balancing: A genetic algorithm approach. Production, Planning and Control, 11, 44–53.

[7] Kim, Y. K., Song, W. S., & Kim, J. H. (2009). A mathematical model and a genetic algorithm for two-sided assembly line balancing. Computers and Operations Research, 36, 853–865.

[8] Lale Ozbakir, Pınar Tapkan (2011) "Bee colony intelligence in zone constrained two-sided assembly line balancing problem". Expert Systems with Applications Vol.38, pp. 11947–11957.

[9] Ozcan, U., & Toklu, B. (2009a). A tabu search algorithm for two-sided assembly line balancing. International Journal of Advanced Manufacturing Technology, 43(7–8), 822.

[10] Ozcan, U., & Toklu, B. (2009b). Multiple-criteria decision-making in two-sided assembly line balancing: A goal programming and a fuzzy goal programming models. Computers and Operations Research, 36, 1955–1965.

[11] Ozcan, U., & Toklu, B. (2009c). Balancing of mixed-model two-sided assembly lines. Computers and Industrial Engineering, 57, 217–227.

[12] Pinar Tapkan, Lale Ozbakir, Adil Baykasoglu (2012) "Modeling and solving constrained two-sided assembly line balancing problem via bee algorithms". Applied Soft Computing Vol.12, pp.3343–3355. Advanced Manufacturing Technology 011; 53:363–78.

[13] P. Tapkan, A. Baykasoglu, L. Ozbakir, , Bees algorithm for fuzzy multi-objective two-sided fuzzy assembly line balancing problem.I.B. Turksen (Eds.), Proceedings of 1st International Fuzzy Systems Symposium, TOBB Economy and Technology University, Ankara, Turkey, 2011, pp. 203–210.

ARPN Journal of Engineering and Applied Sciences

[14] Ragsdale,C. T., E. C. Brown. 2004. On modelling line balancing problems in spreadsheets. INFORMS Transactions Education. 4(2) 45-48.

[15] Simaria, A. S., & Vilarinho, P. M. (2009). 2-ANTBAL: An ant colony optimization algorithm for balancing two-sided assembly lines. Computers and Industrial Engineering, 56(2), 489–506.

[16] Wu, E. F., Jin, Y., Bao, J. S., & Hu, X. F. (2008). A branch-and-bound algorithm for two-sided assembly line balancing. International Journal of Advanced Manufacturing Technology, 39(9–10), 1009–1015.