Division of Research
School of Business Administration

July 1987

ENHANCEMENTS TO HOFFMANN'S ASSEMBLY LINE BALANCING PROCEDURE

Working Paper #517

Ram Rachamadugu
University of Michigan

Brian Talbot
University of Michigan

Abstract

In this note we comment on a few aspects of Hoffmann's procedure for Type I Assembly Line Balancing problems. First, a textual inaccuracy that appeared in the original article is pointed out. We next suggest modifications to Hoffmann's procedure to take into consideration compatibility and incompatibility constraints. Finally, we examine the effects of task numbering on the performance of Hoffmann's procedure. Based on our computational studies, we suggest that the LPT (maximum duration) priority sorting heuristic be used as a preprocessor in Hoffmann's procedure and its variants.

Enhancements to Hoffmann's Assembly Line Balancing Procedure

Hoffmann's procedure [2] is one of the most effective heuristic procedures available for assembly line balancing. Talbot et al., [3] verified in their computational study that variants of Hoffmann's procedure yielded the best results among the heuristic procedures tested by them. The purpose of this note is to provide clarifications on the procedure developed by Hoffmann [2] and to suggest methods for enhancing its performance. Further, we show how Hoffmann's procedure can easily be modified to take into account compatibility relationships among tasks. This note assumes that the reader is familiar with the original paper by Hoffmann [2].

CORRECTIONS

Hoffmann mentions that only the immediate precedence is indicated in the precedence matrix. "If 1 >> 3 >> 4 a one (1) is *not* entered in row 1, column 4" [2, p. 553].

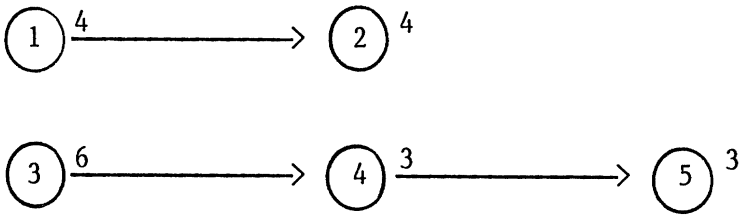
Remark 1: Hoffmann's procedure will work correctly even if the precedence matrix represents more than minimal (or immediate) precedence.

Since in every combination we subtract the row of the task selected, the Hoffmann procedure works the same even if transitive relationships are shown in the precedence matrix. Further, it may be noted that the matrix still remains upper triangular. Invoking Remark 1 can help in reducing the effort involved in preprocessing the data before using the Hoffmann procedure.

Further, Step 8A [1, p. 554] reads: "Go to Step 10." This is in error. Since at this step we have not yet completely enumerated all possible combinations for the current station, we cannot proceed to the next station. Step 10

involves opening a new station. So Step 8A (p. 554) should read "Go to Step 9." We presume that this is a typographical error since the code provided by Hoffmann in the appendix of his paper is correctly written.

A limitation of the original version of the procedure suggested by Hoffmann is a failure to terminate the search for a new combination once a maximal feasible assignment with total processing time equal to cycle time has been found for a given station. As an illustration, consider the example given in Figure 1.



Key: Assume the cycle time = 8. Task indices are shown within the circles and task times are shown outside the circles.

Precedence Diagram for ALB Problem
Figure 1

For the example above, it is clear that tasks 1 and 2 constitute the best combination for station 1. Their total task time exactly equals the cycle time. Hoffmann's code, however, further generates a combination consisting of only task 3 and then decides to use 1 and 2 as the selection for station 1. Gehrlein et al., ([1], page 1067) recognized this limitation of Hoffmann procedure and modified Hoffmann's procedure in their experiments accordingly. Computational effort can be saved by making the following modifications to Hoffmann code (our modifications are italicized).

Step 4. If the result is *zero or* greater than zero, go to Step 5.

4a. If the result is negative, go to Step 6.

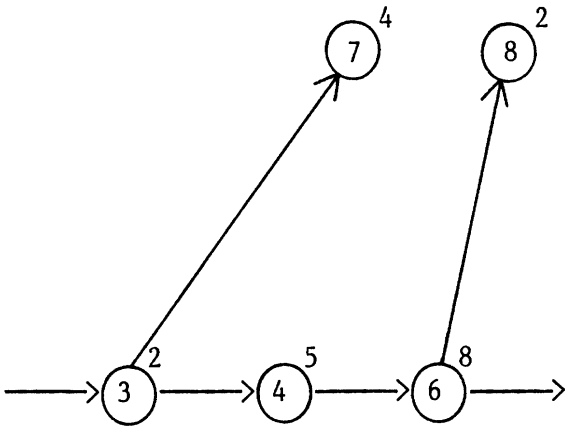
5. Substract from the Code Number the row corresponding to the element selected and use this result as a new Code Number. If the result is greater than zero, go to Step 6.

   If the result is equal to zero, the combination generated in this step is the one having maximum elemental time for this station. Go to Step 10.

Further, it is stated on page 556 [2] that *'this technique and the Helgeson-Birnie technique [4] differ only in that the latter sums the times for all succeeding elements while this matrix method is equivalent to selection on the basis of the total number of succeeding elements [6].'* It can be seen that, in general, this is not valid. We show this using the same example shown in Figure 1. Hoffmann's procedure selects tasks 1 and 2 first. However, if task selection is based on the total number of succeeding elements, we would choose task 3 first. Hoffmann's claim will be valid only if the tasks are numbered such that a task with the larger total number of succeeding elements has a lower index than a task with a smaller number of total succeeding elements. We further pursue the effect of task numbering on the computational performance of Hoffmann's procedure in a later section of this note.

COMPATIBILITY CONSTRAINTS

A practical aspect of assembly line balancing that often needs to be taken into consideration is the fact that some pairs of tasks have to be performed at the same station (Gehrlein et al., [1]). This can easily be accomplished with any solution method by modifying the precedence diagram. As an illustration, consider the set of tasks shown in Figure 2. Figure 3 represents an 'equivalent' precedence diagram which takes into consideration this requirement.



Tasks 3 and 6 must be performed at the same station

Figure 2

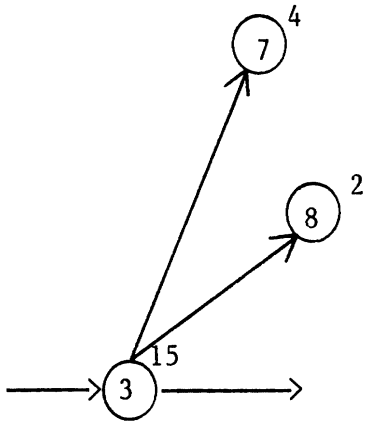


Figure 3

INCOMPATIBILITY CONSTRAINTS

Yet another practical issue in assembly line balancing is incompatibility among tasks. For example, safety or quality considerations may forbid two tasks from being performed together at a workstation. A quality illustration we recently dealt with was for manual assembly of electronic components. Quality considerations warranted avoiding assignment of tasks involving use of identical looking components at the same station. Such incompatibilities among tasks can easily be incorporated into Hoffmann's procedure as described below.

It may be noted that the precedence matrix is upper triangular. We use the lower triangular matrix to represent the incompatibility constraints as follows:

$p(j, i) = -1$ if i and j are incompatible

0 otherwise

where $i < j$

P(i) is a row vector corresponding to task i in the precedence matrix. Define for each task i, two row vectors, M(i) and N(i), each with n elements, as follows:

M(i) = (0, 0..... 0, p(i + 1, i), p(i + 2, i)..... p(n,i)).

N(i) = (p(i, 1), p(i, 2)..... p(i, i - 1), 0, 0.....).

Note that M(i) merely represents the set of tasks which are incompatible with task i and which have indices greater than i. Similarly N(i) represents the set of tasks which are incompatible with task i and which have indices smaller than i. (N(1) and M(n) are zero row vectors). M(i) and N(i) are constructed whenever necessary by looking up the precedence matrix.

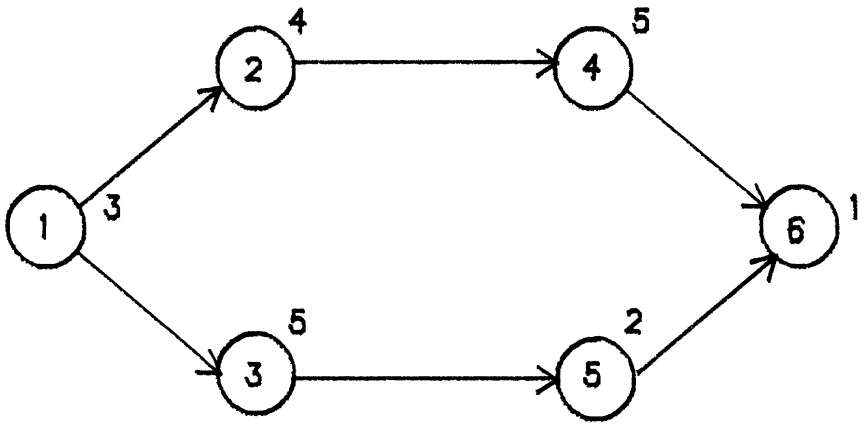
Additional necessary changes to Hoffmann's code so that incompatibilities are taken into account are as follows.

Step 5: Suppose that the element selected is i. Subtract from the Code Number the row corresponding to i and also M(i). Use this result as a new Code Number. Go to Step 6.

Step 10: Replace the first Code Number with the last Code Number corresponding to the previous result. Add to the Code Number $\sum_{i \in s} (M(i) + N(i))$ where s represents the set of tasks assigned at the previous station.

Note that incompatibilities may exist among tasks (Step 5). Hence, whenever a new station is opened, adjustments must be made in the Code Number used so that the first Code Number for the new station represents the precedence relationships among the unassigned tasks. This warrants the changes in Step 10.

An illustration of the modified procedure with an example is shown in Figure 4. Calculations are shown in Table 1.



Key: Cycle time = 9

Tasks 1 and 2 are incompatible
Tasks 3 and 5 are incompatible.

ALB Example Problem with Task Incompatibilities
Figure 2

Table 1

Calculations for ALB Example Problem with Task Incompatibilities

Task		1	2	3	4	5	6	Element Selected	Element Time	Station Time Remaining Before Selection	Station Time Remaining After Selection	Elements Contained in Combination After Selection	Comments
Task	1	D	1	1	0	0	0						
	2	-1	D	0	1	0	0						
	3	0	0	D	0	1	0						
	4	0	0	0	D	0	1						
	5	0	0	-1	0	D	1						
	6	0	0	0	0	0	D						
Combination Column↓													↓Comments
	K_1	0 ↑	1	1	1	1	2	1	3	9	6	1	
		D +0	1 -1	1 0	0 0	0 0	0 0						Subtract (P(1)+N(1))
	K_2	-D	1	0 ↑	1	1	2	3	5	6	1	1,3	
		0 0	0 0	D 0	0 0	1 -1	0 0						Subtract (P(3)+N(3)
	$*K_3$	-D	1	-D	1	1	2						
	K_2	-D	1	0	1	1	2						

* indicates the latest combination generated at each station in step 8.

——— in the combination column indicates the generation of a maximal feasible combination.

═══ in the combination column indicated the completion of assignment at a station.

Table 1 (con't)

K_1	0	1	1	1	1	2						
K_3	-D	1	-D	1	1	2						
	0	-1	0	0	0	0						Add (M(1)+N(1))
	0	0	0	0	-1	0						Add (M(3)+N(3))
K_3	-D	0 ↑	-D	1	0	2	2	4	9	5	2	
	-1 0	D 0	0 0	1 0	0 0	0 0						Subtract (P(2)+N(2))
K_4	-D+1	-D	-D	0 ↑	0	2	4	5	5	0	2,4	
	0 0	0 0	0 0	D 0	0 0	1 0						Subtract (P(4)+N(4))
*K_5	-D+1	-D	-D	-D	0 ↑	1	5	2	0		2	
K_4	-D+1	-D	-D	0	0 ↑	2	5	2	5	3	2,5	
	0 0	0 0	-1 0	0 0	D 0	1 0						Subtract (P(5)+N(5))
K_5	-D+1	-D	-D+1	0	-D	1			3			
K_4	-D+1	-D	-D	0	0	2						
K_3	-D	0	-D	1	0 ↑	2	5	2	9	7	5	
	0 0	0 0	-1 0	0 0	D 0	1 0						Subtract (P(5)+N(5))
K_4	-D	0	-D+1	1	-D	1			7			

Table 1 (con't)

K_3	-D	0	-D	1	0	2						
K_5	-D+1	-D	-D	-D	0	1						
	-1	0	0	0	0	0						Add (M(2)+N(2))
	0	0	0	0	0	0						Add (M(4)+N(4))
K_5	-D	-D	-D	-D	0 ↑	1	5	2	9	7	5	
	0 0	0 0	-1 0	0 0	D 0	1 0						Subtract (P(5)+N(5))
K_6	-D	-D	-D+1	-D	-D	0 ↑	6	1	7	6	5,6	
	0 0	0 0	0 0	0 0	0 0	D 0						Subtract (P(6)+N(6))
$*K_7$	-D	-D	-D+1	-D	-D	-D			6		5,6	

EFFECTS OF TASK NUMBERING

In this section, we investigate the effects of task numbering schemes on the Hoffmann procedure and a variant of the Hoffmann procedure proposed by Gehrlein and Patterson [1].

It is well known that task indexing schemes can affect the performance of heuristic procedures. But, prior researchers have not looked at the effects of task indexing on the Hoffmann procedure. A reason for this might be that in his procedure a feasible combination of tasks from all unassigned tasks is chosen which has the largest work content. If this combination is unique, then there are no alternate combinations from which we need to choose. Thus, one may expect that task indexing procedures are unlikely to affect the quality

(number of stations) of solution obtained from the Hoffmann procedure. However, if there are alternate feasible combinations of tasks which have same maximal work content, then the quality of solution might vary depending on the task indexing scheme.

Gehrlein and Patterson [1] proposed a variant of Hoffmann's procedure to allocate the idle time evenly among workstations and also to reduce the computational time. The procedure works as follows: terminate the search for a new combination of tasks to be assigned at a station once a set of feasible tasks is found such that the following relation holds:

$$0 \leq C - \sum_{i \in f} t_i \leq \theta \left\{C - \frac{\Sigma t_i}{N}\right\}$$

where C is cycle time, f is the feasible set of tasks chosen from unassigned tasks, N is the minimum number of stations required and θ is a parameter. By and large, this modification does reduce the computational time and often the reduction is achieved with no increase in the number of stations required. Based on prior experience, Talbot, Patterson and Gehrlein [3] used 0.5, 1.0 and 2.0 for θ in their computational studies.

The reader may note that for the variant proposed by Gehrlein and Patterson, the task indexing scheme may affect the quality of solution obtained. For example, it is easy to see that if the SPT (Shortest Processing Time) rule is used in indexing the tasks (i.e., tasks are ordered according to increasing task time after precedence restrictions have been met) early stations will tend to be packed with smaller tasks, and hence solution quality may deteriorate for later stations. Using similar reasoning it may be argued that LPT the (Longest Processing Time) rule indexing should have beneficial effects.

In order to investigate these effects, we tested LPT and SPT indexing schemes on 128 balances. These were obtained using data sets from the

literature [3] and creating two problems for each data set (one original data set and the other obtained using the reversed precedence network). We then compared the relative performance of these task indexing schemes:

i) LPT (With ties broken using maximum number of immediate followers)

ii) SPT (With ties broken using maximum number of immediate followers)

iii) The indexing scheme as found in the original literature source. (These numbering schemes were apparently arbitrary, since they were not specified by their authors.)

iv) Random indexing.

Results comparing the performance of LPT and SPT with the original tasks indexing scheme are shown in Table 2.

Table 2

Results Comparing LPT, SPT and Literature Numbering Schemes for Literature Problems

	Parameter value (θ)											
	2.0			1.0			0.5			0.0		
	Better	Worse	Net	Better	Worse	Net	Better	Worse	Net	Better	Worse	Net
LPT	17	7	10	15	6	9	14	6	8	15	9	6
SPT	3	5	-2	4	6	-2	5	6	-1	6	6	0

Key: Better and worse indicate the number of balances where the solution obtained was better or worse than the solution obtained using the original indexing scheme. The total number of balances is 128. A parameter value of 0.0 corresponds to the original Hoffmann procedure.

It is clear from Table 2 that using LPT improves the quality of solution in an average sense. Although in all cases, the difference in the solution obtained was no more than one station. Further, note that the parameter value 0 corresponds to Hoffmann's original procedure. In that version each feasible

combination is examined at each station. Though SPT did not have a detrimental effect on the original Hoffmann procedure ($\theta = 0$), it did perform poorly. LPT does improve the performance of the modified Hoffmann procedure. Even in the original version of Hoffmann code, LPT overall attained better balances when compared to the original task indexing scheme.

Table 3

Results Comparing LPT, SPT and Random Numbering Schemes for Literature Problems

	Parameter value (θ)											
	2.0			1.0			0.5			0.0		
	Better	Worse	Net	Better	Worse	Net	Better	Worse	Net	Better	Worse	Net
LPT	13	6	7	11	8	3	8	6	2	11	6	5
SPT	8	11	-3	6	13	-7	5	13	-8	9	11	-2

In order to eliminate potential biases involved in the task indexing schemes used by various authors in the literature data set, we reindexed the tasks randomly (subject to precedence) and then compared the performance of LPT and SPT against this random indexing scheme. These results are shown in Table 3. Here again, it can be seen that LPT performs better than random indexing and SPT performs worse.

We further tested the effects of task indexing schemes on a set of 1320 problems used by Talbot, Patterson and Gehrlein. Each problem was solved in both forward and reverse directions for a total of 2640 balances (indexing was done first for each direction).

Table 4

Results Comparing LPT, SPT and Random Numbering Schemes for 2640 Balances

	Parameter value (θ)											
	2.0			1.0			0.5			0.0		
	Better	Worse	Net	Better	Worse	Net	Better	Worse	Net	Better	Worse	Net
LPT	597	372	225	518	362	156	517	368	149	516	369	147
SPT	406	407	-1	383	365	18	382	369	13	381	370	11

Since in the original problem set tasks were indexed randomly subject to precedence restrictions, no attempt was made to randomly index them again. The performance of LPT and SPT are shown in Table 4 for this data set. It is clear that LPT again performs better than the original (i.e., random) indexing.

The improved solution performance comes at a very small computational price. The largest time required for presorting the tasks was 11 milliseconds which was for Arcus 111 task problems, and there appears to be no additional computational cost associated with the Hoffmann procedure itself as a result of presorting. All tests were run on an IBM 3090-400 computer without any optimization (VS FORTRAN 77 Compiler with OPT(0)).

CONCLUSION

In this note we describe how practical considerations such as compatibility constraints and incompatibility constraints can be incorporated into Hoffmann's procedure. Further we pointed out a few inaccuracies in the original paper. However, it should be noted that the code provided by Hoffmann [2] is valid. Also, we mentioned how the computational performance of Hoffmann's

procedure and its variants are influenced by task indexing schemes. Based on our computational studies, it appears that the use of LPT can improve the results obtained using Hoffmann type procedures.

References

1. Gehrlein, W. V., and J. H. Patterson, "Sequencing for Assembly Lines with Integer Task Times," Management Science, Vol. 21, No. 9, May 1975, pp. 1064-1070.

2. Hoffmann, T. R., "Assembly Line Balancing with a Precedence Matrix," Management Science, Vol. 9, No. 4, July 1963, pp. 551-562.

3. Talbot, F. B., J. H. Patterson, and W. V. Gehrlein," A Comparative Evaluation of Heuristic Line Balancing Techniques," Management Science, Vol. 32, No. 4, April 1986, pp. 430-454.