The Journal of Applied Business Research - Fall 1987

THE PACKING HEURISTIC FOR ASSEMBLY LINE BALANCING PROBLEMS

Francis J. Nourie, Loyola University
Enrique R. Venta, Loyola University

Abstract

This paper describes the packing principle for assembly line balancing problems, develops a simple heuristic method based on the principle, and explores the implications of the principle for other methods. Computational results show that the proposed method compares favorably with two well-known heuristic methods and that the principle can be applied to improve these two methods with small additional computational effort.

INTRODUCTION

The assembly line balancing problem consists of assigning n tasks, each with performance time t_i, among work stations, each with time capacity C so that the number of work stations is minimized, without violating any precedence relationships between tasks. A good discussion of this problem can be found in the review of Ignall [2].

A new idea, the packing principle, is described here and is used to obtain a fast and simple heuristic method. A computer implementation of this method has been developed and computational results compare favorably with those of two well-known heuristic methods, the ranked positional weight method (RPW) of Helgeson and Birnie [1] and the column method (KW) of Kilbridge and Wester [3]. The packing principle has also been applied as an addendum to these methods, and better solutions were found with small additional computational effort.

Although optimal solution seeking methods are not part of this study, the packing principle has been adapted to develop an exact method in [4]. Preliminary computational results with the algorithm of [4] compare favorably with those reported in Thangavelu and Shetty [5], a zero-one integer programming method which appears to be the most efficient of known optimal solution seeking methods.

THE PACKING PRINCIPLE

The packing principle is a logical idea which apparently has been overlooked in previous work. A station is said to be packed when no additional assignments of tasks to the station are possible. A station is said to be closed when it is not considered for additional assignments of tasks. The packing principle is that no station is closed unless it is packed.

In other heuristic methods, it is possible to violate this principle and with some methods this can occur quite often. For example, the RPW method assigns tasks exclusively in descending order of positional weight. This means that if a task cannot be assigned to the current station, the station will be closed without ensuring that it is packed. A similar situation occurs with the KW method.

This principle leads directly to the algorithm presented in detail in the

next section. But the idea can also be applied to improve existing heuristic methods. This modification has been implemented for both the RPW and KW methods.

To modify the RPW method, the packing principle is applied to the RPW-ordered tasks. Thus, if task i does not fit into the current station, instead of closing the station, the remaining tasks will be considered for possible assignment to this station. Only when stations 1 through j-1 are packed will assignment to station j be permitted.

The modification of the KW method is not as direct. The packing principle is invoked when a new column is considered. Before tasks from column k are assigned to station j, an attempt is made to assign each task in the column to stations 1 through j-1, successively. Note that stations are always available for the possible assignment of additional tasks (stations are never closed). Since at the time considered, each task is assigned to the earliest possible station, an assignment obtained in this way results in each station being packed.

To summarize, the packing principle, whatever its specific implementation, results in the assignment of tasks to spaces (remaining in stations) that would otherwise have gone unused. The extra computational effort that this entails appears to be small.

THE PACKING ALGORITHM

The implementation of the packing principle as a stand-alone heuristic method is given below. Here:

- i is the index for tasks,
- j is the index for stations,
- A is the set of currently assigned tasks,
- a is the number of currently assigned tasks,
- s is the remaining time capacity of the current station,
- C is the time capacity of each station,
- P_i is the set of immediate predecessors of task i,
- t_i is the performance time of task i,
- n is the number of tasks, and
- x_i is the station to which task i is assigned.

The algorithm is as follows:

0. Set j = 1, A = 0, and a = 0.
1. Set i = 1 and s = C.
2. If i is not a member of A and P_i is a subset of A and

   $t_i \leq s$, set $A = AU\{i\}$,

   $a = a + 1$, $x_i = j$,

   and $s = s - t_i$.
3. Set $i = i + 1$. If $i \leq n$, go to step 2.
4. If $a = n$, stop. (All tasks are assigned.)
5. Set $j = j + 1$. If $j \leq n$, go to step 1.

Otherwise, stop. (Infeasible problem.)

In steps 2 and 3, each task is checked successively for possible assignment to the current station. A task is assigned to the current station if its predecessors are already assigned and if its performance time does not exceed the station's remaining time capacity. When it is not possible to find an assignable task, one of two conditions will occur: (1) if all tasks have been assigned, the algorithm stops and a solution has been found (step 4); (2) if unassigned tasks remain, the current station is closed (because it is packed) and assignments are made to the next

station (step 5). A problem is infeasible when either $t_i > C$ for some i and/or there is an inconsistency in the precedence ordering. These conditions are detected when the index of the current station exceeds n (step 5).

EXAMPLE

The example in Figure 1 has been formulated to illustrate the packing principle in several situations. When the packing algorithm is applied the solution is:

Station	Tasks
I	1,2,3,6
II	4,5,8,9
III	7,10,11

Note how task 6 is packed into station I when tasks 4 and 5 cannot be assigned in the remaining time capacity. The same situation occurs with tasks 8 and 9 in station II as task 7 is by-passed.

The importance of the packing idea is seen when the example is solved by the RPW method. This method sorts the tasks according to their rank-positional weight yielding the order (1,2,3-,5,6,4,7,8,9,10,11). When the tasks are assigned in this order, the solution is

Station	Tasks
I	1,2,3
II	5,6,4
III	7,8,9
IV	10,11.

Notice that the fourth task to be assigned, task 5, cannot be assigned in the remaining time capacity of station I. This leads to the premature closing of station I and a solution which requires four stations.

When the packing principle is added to the RPW method, tasks are considered in the same order as above, but tasks that cannot be assigned (performance times too large) to the current station are by-passed. Thus, tasks 1,2,3 are assigned as above, tasks 5 and 4 are by-passed, and task 6 is assigned to station I. Station I is now packed and assignments can be made to station II. A similar situation occurs when assigning tasks to this station. Here, task 7 is by-passed, allowing the assignment of tasks 8 and 9. The solution is:

Station	Tasks
I	1,2,3,6
II	5,4,8,9
III	7,10,11

Note a solution in one less stations has been achieved.

The same effect is observed when the example is solved by the KW method. In this method tasks are grouped into columns, representing precedence groups. The columns are considered sequentially. The tasks within a particular column are considered simultaneously and assigned optimally. This method yields the following solution:

Station	Tasks
I	1,2,3
II	4,5,6
III	7,8,9
IV	10,11

To obtain the above solution, column 1 is considered and all its tasks are assigned to station I. Since there is remaining time capacity in station I, tasks in column 2 are considered for assignment in that station. Task 3 forms an optimal subset for assignment to station I, which is now closed (but not packed). Tasks 4 and 5 are assigned to station II, leaving enough time capacity for task 6 of column 3. Tasks 7, 8 and 9 are assigned to station III from columns 3, 4 and 5, respectively. Finally, tasks 10 and 11 are assigned to station 4. When the packing modification is employed, tasks 1, 2 and 3 are assigned as above, but station I is not closed. Tasks 4 and 5 are

assigned to station II, without closing it. When the next column is considered (column 3), task 6 is "packed" into station I. Task 7 is assigned to station III, but tasks 8 and 9 (the next columns considered) are "packed" into station II. The resulting solution:

Station	Tasks
I	1,2,3,6
II	4,5,8,9
III	7,10,11

requires one fewer station than that obtained using the KW method alone.

COMPUTATIONAL RESULTS

The algorithm and the additions described above have been implemented on the IBM 3081D on a set of 20 problems from the literature as described in [5] and 80 randomly generated problems of 3 different numbers of tasks: 50 tasks (Random 50), 75 tasks (Random 75), and 100 tasks (Random 100). The randomly generated problems were obtained by varying the flexibility ratio (the percentage of zeros in the half-matrix of precedence relationships), and by randomly generating the time capacity of stations and the performance times of the tasks.

The computational results are shown in Tables 1 and 2. Table 1 compares the packing method to the KW and RPW methods. It shows that the quality of solutions obtained by the packing method is comparable to that of the KW method and significantly better than that of the RPW method. The computational times are much faster, ranging from 37.5% to 59% of the KW time, and from 1% to 11% of the RPW time.

Table 2 explores the effect of using the packing principle as an addendum to the KW and RPW methods. The improvement in the quality of the solutions obtained ranges from minor (KW small-sized problems) to impressive (RPW large-sized problems). The additional computation time is small, ranging from .012 sec. to .045 sec. for the KW addendum and from .031 sec. to .062 sec. for the RPW addendum.

FINAL COMMENTS

Two main conclusions can be drawn from this work. First, the packing algorithm is a viable heuristic method in itself. The method is also simple to explain and understand and is quite suitable for hand computation, even for large problems.

Second, the packing principle appears to be a useful addendum to other assembly line balancing heuristic methods since the additional computational cost is small and there is a potential for improved solutions, especially for larger problems.

Further research is currently being carried out by the authors on a generalization of the packing principle that yields optimal solutions [4]. All problems reported here have been solved by this method with encouraging results, particularly with respect to the reduction in time for the solution of large problems. Complete results are forthcoming.

Table 1. Computational Results for Packing Algorithm.

		Literature Problems	Random Problems 50	75	100
Packing	A	12	16	12	7
	B	7	12	11	12
	C	1	1	3	5
	D	0	0	0	1
	Time (secs.)	.035	.070	.100	.150
KW	A	11	18	13	6
	B	9	10	10	14
	C	0	1	2	5
	D	0	0	1	0
	Time (secs.)	.060	.130	.250	.400
RPW	A	5	5	1	0
	B	10	12	13	7
	C	4	5	4	8
	D	1	7	8	10
	Time (secs.)	.332	1.760	6.180	15.890

A. Number of problems in which optimal solution was found.
B. Number of problems in which solution found had 1 more station than optimal solution.
C. Number of problems in which solution found had 2 more stations than optimal solution.
D. Number of problems in which solution found had 3 or more stations than optimal solution.

Note: Time reported in seconds on an IBM 3081U.

Table 2. Computational Results for Packing Addendum.

		Literature Problems	Random Problems 50	75	100
KW + Packing	A	20	27	22	16
	B	0	2	4	9
	C	0	0	0	0
	D	0	0	0	0
	Add. Time	.012	.015	.030	.045
RPW + Packing	A	10	8	6	4
	B	5	13	10	7
	C	4	7	7	9
	D	1	1	3	5
	Add. Time	.031	.044	.062	.031

A. Number of problems showing no improvement.
B. Number of problems showing an improvement of 1 station.
C. Number of problems showing an improvement of 2 stations.
D. Number of problems showing an improvement of 3 or more stations.

Note: Additional time reported in seconds on an IBM 3081U.

REFERENCES

[1] Helgeson, W.P. and Birnie, D.P., "Assembly line balancing using the ranked positional weight technique," *Journal of Industrial Engineering* 12, 6, 394-8 (November-December, 1961).

[2] Ignall, Edward J., "A review of assembly line balancing," *Journal of Industrial Engineering* 16, 4, 244-54 (July-August, 1965).

[3] Kilbridge, M.D. and Wester, L., "A heuristic method of assembly line balancing," *Journal of Industrial Engineering* 12, 4, 292-8 (July-August 1961).

[4] Nourie, F.J. and Venta, E.R., "The packing principle for assembly line balancing: A method that yields optimal solutions, "Working Paper, Loyola University of Chicago, (1987).

[5] Thangavelu, S.R. and Shetty, C.M., "Assembly line balancing by zero-one integer programming," *AIIE TRANSACTIONS* 3, 1, 61-8 (March 1971).

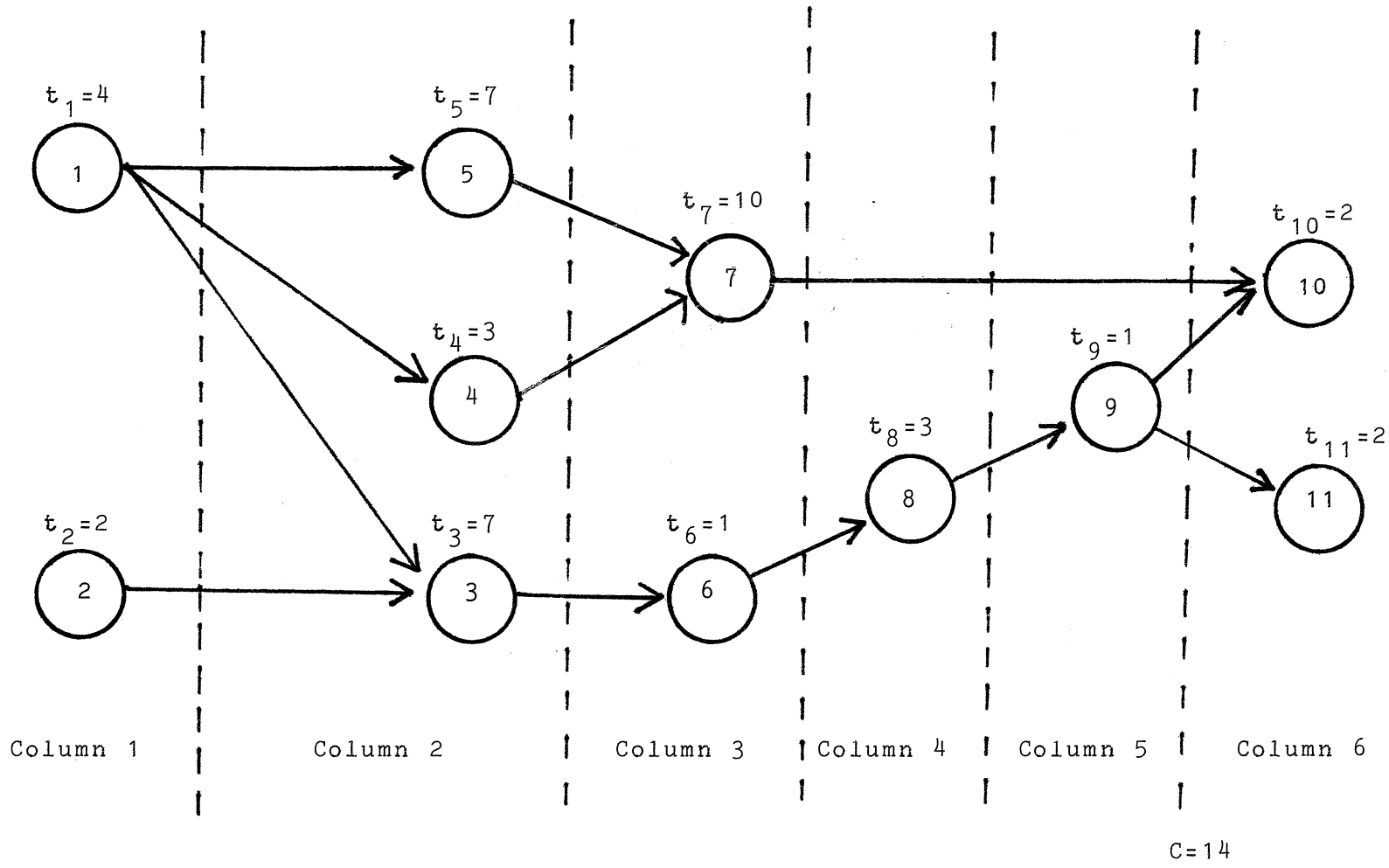

Figure 1. Display of Example Data.