ELSEVIER

Int. J. Production Economics 64 (2000) 187–195

international journal of
production
economics

www.elsevier.com/locate/dsw

An exact method for cost-oriented assembly line balancing

Matthias Amen*

Universität Bern, Institut für Unternehmensrechnung und Controlling, Engehaldenstrasse 4, CH-3012 Bern, Switzerland

Abstract

After a characterization of the cost-oriented assembly line balancing problem it will be shown that by loading the stations maximally the cost-oriented optimum can be missed. Instead of loading the stations maximally the criterion "two-stations-rule" has to be used. For generating optimal solutions an exact backtracking method is introduced in which the enumeration process is limited by modified and new bounding rules. Results of an experimental investigation show that the new method finds optimal solutions for small and medium-sized problem instances in acceptable time.

Keywords: Assembly line balancing; Cost-oriented production planning; Exact methods

1. The cost-oriented assembly line balancing problem

This paper deals with the cost-oriented balancing of a single model assembly line where a vast quantity of one single product (model) has to be assembled. In assembly line production systems the work to be done is organized as follows: The assembly line consists of a number of consecutive stations. The product units move with a constant transportation speed through the line. The production rate is determined by the cycle time which is the time period between the entering of two consecutive product units in a station. The cycle time limits the duration of the work content for a single product unit in a station of the assembly line. In planning an assembly line production system the total work content to be performed in the assembly line production system has to be divided among the stations.

Therefore the total work content first has to be split up into single economically indivisible tasks, which means that a further division of labour would cause additional micro-tasks (work-elements) and an increase in the total work content. Among the single tasks there may exist precedence relations. A set of tasks to be performed in the same station is called an operation. The assembly line balancing problem consists in an aggregation of the tasks to operations restricted by the technological precedence relations among the tasks and the quantitative capacity per product unit in each station which is determined by the cycle time. For a given cycle time the objective in time-oriented assembly line balancing is to minimize the number of stations. The cost-oriented assembly line balancing problem is a generalization of the time-oriented

* Tel.: +41-31-631-85-01; fax: +41-31-631-37-80.
E-mail address: amen@iuc.unibe.ch (M. Amen)

0925-5273/00/$ - see front matter
PII: S0925-5273(99)00057-2

assembly line balancing problem [[1], pp. 65–122; [2]]. The objective in cost-oriented assembly line balancing consists in minimizing the total costs per product unit.

Usually, final assembly of products is very labour intensive. The effective wage rate of a station is determined by the most difficult task to be performed in the considered station. In other words: as the wage rates increase with the point values the wage rate of a station depends on the maximal point value of the tasks aggregated to the stations operation and is equal to the maximal wage rate of these tasks. As in each station not only the duration of the operation but the total cycle time has to be compensated to the worker the total labour cost per product unit consists in the sum of the wage rates of the stations multiplied by the cycle time. Further it is assumed that the cost of capital depends on the total line length and that all stations have the same dimension. All other costs (e.g. material cost) are assumed to be independent of the division of labour and the line length.

For the purpose of a precise description the problem will be presented formally. As the problem will not be solved by standard methods implemented in commercial software packages there is no need to give an integer programming formulation. Note that the proposed method doesn't even require a mathematical problem description.

List of symbols used in the mathematical problem description

I number of tasks, dimensionless
h, i index for the tasks, $h, i \in \{1, \ldots, I\}$
M number of stations, dimensionless
m index for the stations, $m \in \{1, \ldots, M\}$
m_i station, to which task i is assigned to, $i = 1(1)I$
I^{s}_m set of tasks assigned to station m, $m = 1(1)M$
c cycle time, TU/PU
k total costs per product unit, MU/PU
k^{s}_m cost rate of station m, $m = 1(1)M$, MU/TU
k^{sc} cost of capital per station, MU/PU
k^{sw}_m wage rate of station m, $m = 1(1)M$, MU/TU
k^{t}_i cost rate of task i, $i = 1(1)I$, MU/TU
k^{tw}_i wage rate of task i, $i = 1(1)I$, MU/TU
d^{t}_i duration of task i, $i = 1(1)I$, TU/PU
d^{s}_m duration of the operation in station m, $m = 1(1)M$, TU/PU
V_i set of tasks which are immediate predecessors of task i, $i = 1(1)I$

(*Note*: Dimensions: TU: time units; PU: product units; MU: monetary units.
Labels: c: cost of capital; s: station; t: task; w: wage rate).

As the objective is to minimize the total costs per product unit first the calculation of this value has to be considered. The costs per product unit can be stated as

$$k = \sum_{m=1}^{M} ck^{\mathrm{sw}}_m + Mk^{\mathrm{sc}} \quad \text{where } k^{\mathrm{sw}}_m = \max\{k^{\mathrm{tw}}_i \mid i \in I^{\mathrm{s}}_m\}.$$

This calculation will be simplified in the following way: Rewriting the cost of capital Mk^{sc} as $Mc(k^{\mathrm{sc}}/c)$ and considering that the cost of capital per time unit (k^{sc}/c) and the wage rate of a task k^{tw}_i can then be combined to the total cost rate of a task $k^{\mathrm{t}}_i := k^{\mathrm{tw}}_i + (k^{\mathrm{sc}}/c)$, the total cost rate of a station amounts to $k^{\mathrm{s}}_m = \max\{k^{\mathrm{t}}_i \mid i \in I^{\mathrm{s}}_m\}$ which is the same as $k^{\mathrm{s}}_m := k^{\mathrm{sw}}_m + (k^{\mathrm{sc}}/c)$. With this formulations the total costs per product unit are given by

$$k = \sum_{m=1}^{M} ck^{\mathrm{s}}_m \quad \text{where } k^{\mathrm{s}}_m = \max\{k^{\mathrm{t}}_i \mid i \in I^{\mathrm{s}}_m\}.$$

While assigning tasks to stations the following restrictions have to be fulfilled:

Precedence relations:

$$m_h \leqslant m_i, \quad \forall h \in V_i,\ i = 1(1)I. \tag{1}$$

Capacity restrictions:

$$\sum_{i \in I^{\mathrm{s}}_m} d^{\mathrm{t}}_i = d^{\mathrm{s}}_m \leqslant c, \quad m = 1(1)M. \tag{2}$$

Occurrence restrictions:

$$\bigcup_{m=1}^{M} I^{\mathrm{s}}_m = \{1, \ldots, I\} \quad \text{and} \quad I^{\mathrm{s}}_{\bar{m}} \cap I^{\mathrm{s}}_{\hat{m}} = \emptyset, \tag{3a}$$

$$\text{for all } \bar{m}, \hat{m} \in \{1, \ldots, M\} \text{ with } \bar{m} \neq m. \tag{3b}$$

Restrictions (1) require that the stations are numbered in ascending order. These restrictions ensure

that the station performing task i must not precede any station performing a technologically immediate preceding task $h \in V_i$. In restrictions (2) it is stated that the sum of the durations of the tasks aggregated to an operation must not exceed the cycle time. Restrictions (3a) and (b) ensure that each task is assigned exactly once.

2. Applicability of dominance criteria known from time-oriented assembly line balancing

After describing the problem of cost-oriented assembly line balancing two basic dominance criteria which are used in time-oriented assembly line balancing are considered.

2.1. Maximally loaded station rule

Heuristic and exact methods for time-oriented assembly line balancing usually fill stations with tasks successively. A station is said to be maximally loaded if there are no more tasks assignable to it [[3], p. 266]. A task is called assignable to a station if all its predecessor tasks are performed in earlier stations or in the considered station at the latest and if its operation time does not exceed the remaining idle time of the station under current consideration. The following example shows that a cost-oriented optimal solution can be missed if the stations are loaded maximally (see Fig. 1 and Table 1).

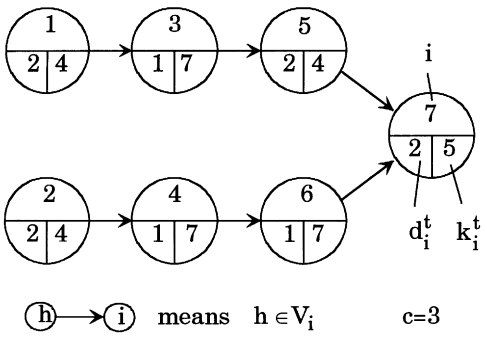

Fig. 1. Precedence graph for the example.

The time-oriented optimal solution cited in Table 1 is one of the three possible time-oriented optimal solutions with the minimal number of stations $M^{\text{time}} = 4$ which can be obtained when stations are loaded maximally. Each of these solutions will cause costs per product unit of $k^{\text{time}} = 78$ MU/PU. As shown in Table 1 it is possible to generate solutions with lower costs per product unit if stations are not loaded maximally. The cost-oriented optimal solution cited in the table is one of the two cost-oriented optimal solutions with minimal costs per product unit $k^{\text{cost}} = 72 < 78$ MU/PU but with a higher number of stations $M^{\text{cost}} = 5 > 4$. In both cost-oriented optimal solutions there are stations which are not loaded maximally. From this simple example two important conclusions can be drawn:

- There can exist a conflict between the time-oriented objective to minimize the number of stations and the cost-oriented objective to minimize the total costs per product unit.
- Not every dominance criterion used in time-oriented assembly line balancing is valid if a solution with minimal costs per product unit is required. As the example shows the well-known maximally loaded station rule is not valid for the generation of a cost-oriented optimal solution.

Table 1
Optimal solutions to the example

Station	Time-oriented optimal solution		Cost-oriented optimal solution	
m	I_m^s	k_m^s	I_m^s	k_m^s
1	{1, 3}	7	{1}	4
2	{2, 4}	7	{2}	4
3	{5, 6}	7	{3, 4, 6}	7
4	{7}	5	{5}	4
5	—	—	{7}	5
$\sum_{m=1}^{M} k_m^s$		26		24
k		$3 \cdot 26 = 78$		$3 \cdot 24 = 72$
M		4		5

2.2. Two-stations rule

A dominance criterion which is much weaker than the maximally loaded station rule is the two-stations rule: The two-stations rule is fulfilled if the sum of the durations of the operations in two consecutive stations m and $m+1$ exceed the cycle time c: $d^s_m + d^s_{m+1} > c, m = 1(1)M-1$. It is obvious that the two-stations rule is fulfilled automatically if the stations are loaded maximally. Therefore the two-stations rule is not required explicitly in solution methods for time-oriented assembly line balancing. If the two-stations rule is not fulfilled the concerning stations can be combined to one. As the sum of the costs per product unit in these stations is $c(k^s_m + k^s_{m+1})$ before combining and $c \max\{k^s_m, k^s_{m+1}\}$ after combining these stations to one, costs can be saved in the latter case. The cost reduction amounts to $\Delta k = c \min\{k^s_m, k^s_{m+1}\}$ per product unit. Therefore the two-stations rule is applicable when an optimal solution for the cost-oriented assembly line balancing problem is required.

3. An exact backtracking method for the cost-oriented assembly line balancing problem

In this section an exact backtracking method for solving the cost-oriented assembly line balancing problem is introduced. Besides the new exact method presented in this section only one simple approach is reported in the literature, that gains for an cost-oriented optimal solution. Applying only one dominance criterion (local lower bound for the costs per product unit (see Section 3.2)) the reported approach is just demonstrated on a single example and not described in general [[1], pp. 95–105].

For the backtracking method presented in this paper it is assumed that the tasks are numbered topologically, i.e. if $h \in V_i$, then $h < i$ holds. The enumeration steps of the backtracking method [[4], p. 518; [5], pp. 150–151) consist of two forward and two backtrack steps and a variety of dominance criteria. The dominance criteria prevent the search from exhaustive enumeration. The total enumeration process of the backtracking method and the application of the dominance criteria are illustrated in the flow chart of Fig. 2 (see next page). The bold lined rhombs symbolize the use of the dominance criteria and the bold lined rectangles symbolize the forward and backtrack steps. The outgoing arrows of the rhombs symbolizing the dominance criteria are named operative and inoperative, respectively, according to whether the enumeration process is limited or not. The enumeration process and the dominance criteria will be discussed briefly in the next two sections.

3.1. Enumeration process

Apart from the dominance criteria which are considered in the next section the general enumeration process can be characterized as follows: In each step of the enumeration process only the current station is taken into consideration (station-oriented approach). In a forward step S, a new station is established, in a forward step T, a task is assigned to the station under current consideration. In a backtrack step S, the enumeration process goes back from the current station to the immediately preceding station. In a backtrack step T, a task is removed from the current station.

The enumeration process is based on the topological numbering of the tasks. Let i^{last} be the last task which has been assigned to or which has been removed from the current station. In a forward step T, among the assignable tasks only the one with the lowest number exceeding i^{last} (this set of tasks is named "I_assign,last" in Fig. 2) has to be chosen for assignment. In a backtrack step T, the highest numbered task has to be chosen to remove from the current station. By making use of the topological numbering of tasks and organizing the enumeration process in the way mentioned above multiple enumerations of identical solutions are avoided [[6], pp. 11,17,18]. If no dominance criteria is operative after a forward step T, the enumeration process continues with a forward step S. Generally after each forward step S a single forward step T follows. A backtrack step T is necessary if an additional task should be assigned to a station but there is no assignable task with a higher number than i^{last}. If no backtrack step T is possible (because the station is empty) a backtrack step S has to be taken. If this is not possible (we are already in the first station) then the enumeration

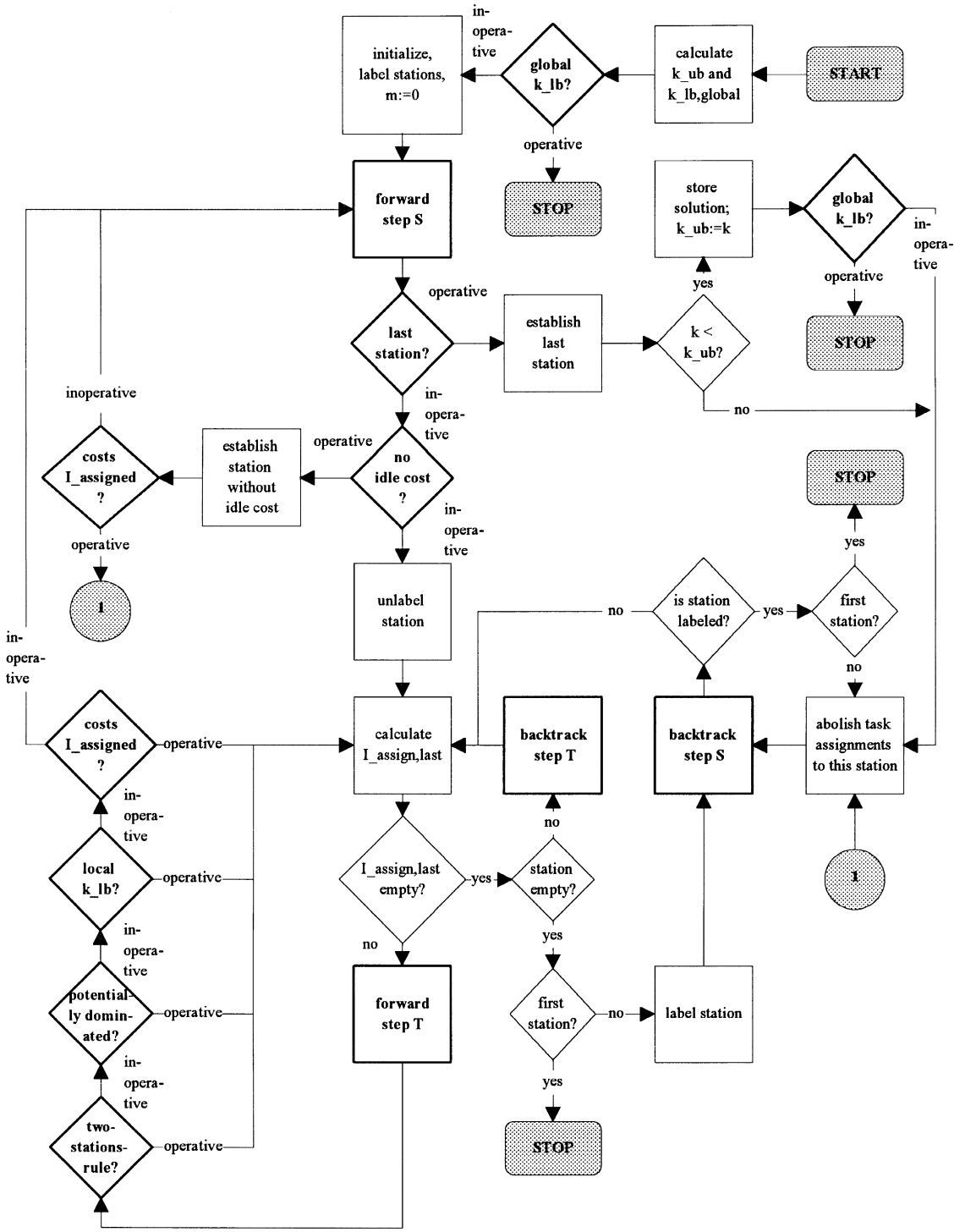

Fig. 2. Flow chart of the exact backtracking method.

process stops because all feasible assignments of tasks to stations have been generated explicitly or implicitly. Otherwise after each backtrack step T or S, it has to be proved whether an additional task can be assigned to the station or not.

3.2. Dominance criteria

The general enumeration process is controlled by the dominance criteria which are proved at different stages of the enumeration scheme. These dominance criteria will be discussed briefly in this section.

(1) *Global lower bound for the costs per product unit* (rhomb "global k_lb?"): Before starting the enumeration process a global lower bound for the costs per product unit $k^{\mathrm{lb,global}}$ [MU/PU] has to be calculated. If a solution of the assembly line balancing problem with costs per product unit $k = k^{\mathrm{lb,global}}$ is generated no more improvement is possible. In this case the enumeration process stops because of the optimality of the solution found yet. The calculation of the global lower bound is illustrated in Fig. 3.

It is calculated by assigning the tasks in order of nonincreasing cost rates to $M^{\mathrm{lb},1} = \lceil d^{\mathrm{p}}/c \rceil$ stations (with d^{p} [TU/PU] duration of total work content of one product unit) ignoring the precedence relations among the tasks and the indivisibility of tasks [[1], pp. 97–100] and subsequently summing up the total cost per product unit in these stations. A further improvement is possible by calculating an alternative lower bound for the number of stations $M^{\mathrm{lb},2}$ which in some cases exceeds the lower bound $M^{\mathrm{lb},1}$ [[7], p. 20; [8], p. 246]. Since the cost rate of

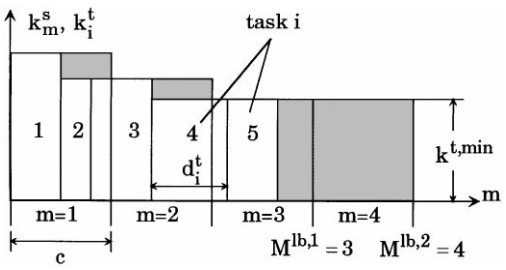

Fig. 3. Calculation of the global lower bound for the costs per product unit.

every station will be at least $k^{\mathrm{t,min}}$, which is the minimal cost rate of all tasks, an improvement of the global lower bound for the costs per product unit can be effectuated. If $M^{\mathrm{lb},2} > M^{\mathrm{lb},1}$ holds then $(M^{\mathrm{lb},2} - M^{\mathrm{lb},1})ck^{\mathrm{t,min}}$ is added to the lower bound which is obtained by the fictitious assignment process described above. As an initial feasible solution is needed to obtain a first upper bound for the costs per product unit ("k_ub" in Fig. 2) in order to work as the best solution known so far, it may happen that the costs per product unit of the heuristically generated solution are as high as the global lower bound. In this case the enumeration process stops because the initial solution is proved to be optimal.

(2) *Last station* (rhomb "last station?"): After a station is established in a forward step S the sum of the durations of the tasks, that are not assigned yet, d^{rest} [TU/PU] is compared with the cycle time c. If the sum of the durations of the unassigned tasks is not longer than the cycle time the considered station is the last one. Therefore all unassigned tasks can be assigned in one single step to this station. After that the total costs per product unit of the complete solution are calculated and compared to those of the best solution known so far which serve as an upper bound for the costs per product unit ("k_ub" in Fig. 2). If the costs per product unit of the current solution are lower then the current solution becomes the best-known solution to be stored and its costs become the new upper bound. After each improvement of the best-known solution the dominance criterion "global lower bound for the costs per product unit" has to be proved. If the enumeration process does not stop because no improvement was possible or in the case of an improvement because the dominance criterion 'global lower bound for the costs per product unit' was inoperative the whole task assignment to this station will be abolished and a backtrack step S will follow.

(3) *Establish station without idle cost* (rhomb "no idle cost?"): After a station is established and it is found that this is not the last one it will be proved whether it is possible to fill the station with assignable tasks in a way that no idle cost exists. Idle cost are caused by idle time and/or cost rate differences between the tasks assigned to the considered station. A station load with no idle cost dominates all

other possible task assignments to the just established station. If this dominance criterion is operative and a station load is generated the dominance criterion 'costs of all assigned tasks' has to be proven afterwards. If the dominance criterion "costs of all assigned tasks" is operative all task assignments to this station have to be abolished and a backtrack step S follows. To identify stations which are filled with tasks by applying this dominance criterion initially the stations are labelled. The label will be removed from a station if this dominance criterion is inoperative. If a labelled station is reached in an enumeration step the station is either empty (just established station in a forward step S) or the station is filled without idle cost because of this dominance criterion. If the enumeration process reaches a labelled station from a backtrack step S all task assignments to the labelled station have to be abolished and a further backtrack step S follows because within the partial solution determined by the task assignments from the first station to the station foregoing the current station all other task assignments to the current station cannot dominate the one without idle cost. If the labelled station is the first station any further backtrack step S is possible and therefore the enumeration process stops.

(4) *Two-stations rule* (rhomb "two-stations rule?"): Regardless of other dominance criteria a new station cannot be established if the sum of the durations of the operations in the current and the foregoing station does not exceed the cycle time. It is necessary to fill the current station with more tasks before establishing a new station otherwise the current and the foregoing station can be combined to one station (see Section 2.2).

(5) *Potential dominance* (rhomb "potentially dominated?"): Regardless of other dominance criteria a new station cannot be established if there exists an assignable task with a cost rate not higher than the calculated cost rate of the station under current consideration. If such a task exists the current partial solution determined by the task assignments from the first to the current station is potentially dominated by an alternative partial solution which consists of the same tasks assignments from the first station to the station foregoing the current station and an extension of the current station load by this task.

(6) *Local lower bound for the costs per product unit* (rhomb "local k_lb?"): Regardless of other dominance criteria a new station cannot be established if the sum of the calculated costs per product unit from the first to the current station and a lower bound for the costs per product unit of the currently unassigned tasks is not lower than the costs per product unit of the best solution known. The local lower bound for the costs per product unit of the unassigned tasks is calculated in an analogous manner as the global lower bound for the costs per product unit, but only $\lceil d^{\mathrm{rest}}/c \rceil$ stations are considered.

(7) *Costs of all assigned tasks* (rhomb "costs I_assigned?"): Regardless of other dominance criteria a new station cannot be established if the costs per product unit of the current partial solution (i.e. the set of all assigned tasks) are not lower than those of the same set of tasks allocated to stations in a different manner earlier in the enumeration process. For this dominance criteria all enumerated sets of all assigned tasks with the corresponding costs per product unit have to be stored. Therefore a binary tree structure is implemented with the help of dynamic variables. Hence only explicitly enumerated sets of all assigned tasks have to be stored [9]. If this dominance criteria is proved at last in the order of the dominance criteria (4)–(7) before a new station is established the storage requirements will be low because sets of assigned tasks which have been found earlier to be dominated by other tasks assignments are prevented from storage.

4. Results of an experimental investigation

For an evaluation of this exact backtracking method test problems were generated randomly using the following parameters:

- number of tasks I [-]: 50, 75, 100
- order strength OS [-]: 0.7, 0.8, 0.9
- cycle time to maximal task duration $c/d^{\mathrm{t,max}}$ [-]: 2, 3
- maximal to minimal task duration $d^{\mathrm{t,max}}/d^{\mathrm{t,min}}$ [-]: 5, 10
- maximal to minimal task cost rate $k^{\mathrm{t,max}}/k^{\mathrm{t,min}}$ [-]: 1.25, 1.50, 2.00

Table 2
Run time and percentage of solutions proved optimal

Number of tasks I Dimensionless	Maximal to minimal task cost rate $\frac{k^{t,\max}}{k^{t,\min}}$ Dimensionless	Order strength OS					
		0.7		0.8		0.9	
		Average run time (s)	Solutions proved optimal (%)	Average run time (s)	Solutions proved optimal (%)	Average run time (s)	Solutions proved optimal (%)
50	1.25	9.74	100.0	4.36	100.0	1.05	100.0
	1.50	35.26	100.0	8.32	100.0	1.86	100.0
	2.00	75.34	100.0	17.52	100.0	3.07	100.0
75	1.25	211.18	99.5	71.87	100.0	11.80	100.0
	1.50	1201.69	91.0	260.87	100.0	22.47	100.0
	2.00	2560.62	53.5	620.46	99.0	40.13	100.0
100	1.25	1078.42	82.0	796.23	96.5	92.31	100.0
	1.50	2953.62	31.5	2357.43	64.5	184.00	100.0
	2.00	3595.23	1.0	3417.45	16.5	298.66	100.0

Two hundred test problems per field. Parameters $c/d^{t,\max}$ and $d^{t,\max}/d^{t,\min}$ take all levels considered.

The order strength OS describes the number of existing direct and indirect precedence relations divided by the theoretical maximal number of direct and indirect precedence relations and therefore ranges form 0 to 1 [[10], pp. 56–59]. In each of the resulting 108 test fields ($2^2 \times 3^3$ experimental design) 50 problems were generated randomly. This totals to 5400 test problems. A heuristic version of this exact backtracking method was used to generate a feasible starting solution to obtain an initial upper bound for the costs per product unit. The backtracking method was implemented in BORLAND PASCAL 7.0. The run times of the program were standardized on a 133 MHz Pentium PC. The run time for solving each problem was restricted by a limit of 3600 seconds.

In evaluating an exact method the run time of an efficient implementation is the main criterion. In Table 2 the average run times and the percentage of solutions proven optimal for the combinations of the parameters number of tasks, maximal to minimal task cost rate, and order strength are listed.

It appeared that the parameters cycle time to maximal task duration and maximal to minimal task duration have minor influence on the run time. Therefore in Table 2 the data are differentiated only by the other parameters. As the data show for 50 tasks and for an order strength of 0.9 the exact backtracking method was able to find optimal solutions for each test problem within the given run time limit regardless of the values of other parameters. As a result the number of tasks is the most important factor determining the run time of the method. The number of tasks determines the possible permutations of tasks during the enumeration process. The order strength restricted the number of possible permutations. Therefore, besides the number of tasks, this is the most important factor but one to influence the run time. As the relation of the maximal to the minimal task cost rate decreases the problem instances become more similar to those of time-oriented assembly line balancing. The problem instances thus are easier to solve, and consequently the run times decrease for every combination of the other parameters.

Further the storage requirements of the binary tree to handle the dominance criterion "costs of all assigned tasks" were not very high for the considered problem instances. For all of the 5400 test

problems a maximal storage of 2,625,300 byte was needed for the binary tree. The total storage requirements for all other variables amounts only to 12,760 byte for a given maximal problem size of 100 tasks.

5. Conclusion

The presented exact backtracking method is the first exact method for solving the cost-oriented assembly line balancing problem described in general and using a variety of dominance criteria. As the experimental investigation demonstrates the new method works very well for small and medium-sized problem instances as it finds optimal solutions within an acceptable run time. The storage requirements for the realization of the dominance criterion "costs of all assigned tasks" are not too high and those for other variables are neglectably low. As a consequence it can be supposed that the storage capacity of today's computers might become a limiting factor only for problem instances with much more tasks.

References

[1] R. Steffen, Produktionsplanung bei Fließbandfertigung, Gabler, Wiesbaden, 1977.
[2] O. Rosenberg, H. Ziegler, A comparison of heuristic algorithms for cost-oriented assembly line balancing, Zeitschrift für Operations Research 36 (1992) 477–495.
[3] J.R. Jackson, A computing procedure for a line balancing problem, Management Science 2 (1956) 261–271.
[4] S.W. Golomb, L.D. Baumert, Backtrack-programming, Journal of the Association for Computing Machinery 12 (1965) 516–524.
[5] H. Müller-Merbach, Ein Verfahren zur Lösung von Reihenfolgeproblemen der industriellen Fertigung, Zeitschrift für wirtschaftliche Fertigung 61 (1966) 147–152.
[6] Th.R. Hoffmann, Permutations and precedence matrices with automatic computer applications, Ph.D. Thesis, University of Wisconsin, 1959.
[7] M.E. Salveson, The assembly line balancing problem, Journal of Industrial Engineering 6 (1955) 18–25.
[8] R.V. Johnson, Optimally balancing large assembly lines with 'FABLE', Management Science 34 (1988) 240–253.
[9] F.J. Nourie, E.R. Venta, Finding optimal line balances with OptPack, Operations Research Letters 10 (1991) 165–171.
[10] A.A. Mastor, An experimental investigation and comparative evaluation of production line balancing techniques, Ph.D. Thesis, University of California at Los Angeles, 1966.