Jena Research Papers in Business and Economics

The Multiproduct Parallel Assembly Lines Balancing Problem: Model and Optimization Procedure

Armin Scholl, Nils Boysen

13/2008

Jenaer Schriften zur Wirtschaftswissenschaft

Working and Discussion Paper Series
School of Economics and Business Administration
Friedrich-Schiller-University Jena

ISSN 1864-3108

Publisher:

Wirtschaftswissenschaftliche Fakultät
Friedrich-Schiller-Universität Jena
Carl-Zeiß-Str. 3, D-07743 Jena

www.jbe.uni-jena.de

Editor:

Prof. Dr. Hans-Walter Lorenz
h.w.lorenz@wiwi.uni-jena.de

Prof. Dr. Armin Scholl
armin.scholl@wiwi.uni-jena.de

www.jbe.uni-jena.de

The Multiproduct Parallel Assembly Lines Balancing Problem: Model and Optimization Procedure

Armin Scholl

Chair of Decision Analysis and Management Science, Friedrich-Schiller-University Jena,
Carl-Zeiß-Straße 3, D-07743 Jena, Germany,
e-Mail: armin.scholl@wiwi.uni-jena.de, phone: ++49 3641 943170 (corresponding author)

Nils Boysen

Chair of Operations Management, Friedrich-Schiller-University Jena,
Carl-Zeiß-Straße 3, D-07743 Jena, Germany,
e-Mail: nils.boysen@uni-jena.de

Abstract

A production system which consists of a number of parallel assembly lines is considered. On each line a certain product is manufactured observing a common cycle time. By arranging the lines in a favourable manner, it is possible to increase efficiency of the production system by combining stations of neighbouring lines when balancing them. The objective is to minimize the number of operators required. This problem is called Multiproduct Parallel Assembly Lines Balancing Problem (MPALBP) and has previously been considered by Gökçen, Agpak, and Benzer (*Internat. J. Product. Economics 103, 600-609*). In the paper on hand, we give a detailed problem description and model the problem as a binary linear program. Furthermore, an exact solution approach based on an extension of the well-known branch and bound procedure SALOME is proposed. Computational experiments show that this procedure clearly outperforms other approaches as it is able to solve small- to medium-sized problem instances to optimality and provides good heuristic solutions for large-sized problems.

Keywords

assembly line balancing; parallel assembly lines; combinatorial optimization; branch-and-bound

1. Introduction

Assembly lines are flow-oriented production systems used for industrial production of high-quantity standardized commodities. An assembly line consists of *(work) stations* arranged along a conveyor belt or a similar material handling equipment. The workpieces (jobs) are consecutively launched down the line and are moved from station to station. At each station, certain operations are repeatedly performed regarding the *cycle time* (maximum time available for each workcycle). The decision problem of optimally partitioning (balancing) the assembly work among the stations with respect to some objective is known as *Assembly Line Balancing Problem (ALBP*; cf. Baybars 1986).

The majority of the literature on ALBP deals with isolated assembly lines producing a single product or a mix of product variants. For recent surveys and problem classifications see Scholl and Becker (2006), Becker and Scholl (2006), Boysen et al. (2007, 2008).

Here, we consider an ALBP with interconnected parallel assembly lines first introduced by Gökçen et al. (2006). We call this problem **Multiproduct Parallel Assembly Lines Balancing Problem (MPALBP)**, because it consists of two connected subproblems: (1) Multiple products have to be assigned to parallel lines and (2) the lines have to be balanced. The connection of both subproblems is given by the option to combine adjoining stations of neighbouring lines.

The remainder of the paper is organized as follows: In Section 2, problem MPALBP is defined more precisely, while a zero-one programming formulation is given in Section 3. Section 4 is devoted to describing an exact solution procedure for MPALBP which is based on an extension of the well-established procedure SALOME for SALBP-1 (see Scholl and Klein 1997, 1999). Computational experiments indicating good performance of the procedure are reported in Section 5. A summary and statements on future research issues in Section 6 conclude the paper.

2. Problem statement

MPALBP assumes a factory workfloor with a given number $\overline{H}$ of serial assembly lines arranged side-by-side as depicted in Figure 1. In each line, a number of consecutive stations can be installed, where $\overline{K}$ denotes the overall maximal number of stations per line which is, e.g., given due to space restrictions. Each line has to manufacture a separate product observing a joint cycle time c, i.e., all lines have the same production rate. The production process of every product is modelled by a precedence graph with tasks represented by nodes, task times by node weights and precedence relationships by arcs.

Traditionally, each line is balanced separately by assigning tasks to stations observing cycle time and precedence relations such that the number of installed stations is minimized. The corresponding problem is called *simple ALBP of type 1* (SALBP-1, cf. Baybars 1986, Scholl and Becker 2006).

By the way of contrast, MPALBP allows for improving efficiency by installing *split workplaces* for adjacent lines. This means that a single worker operates on two directly opposite stations of neighbouring lines within the same cycle by splitting the available (cycle) time between operations performed on a workpiece at a station k at line h and operations performed on a workpiece (of another product) at station k of line h+1.

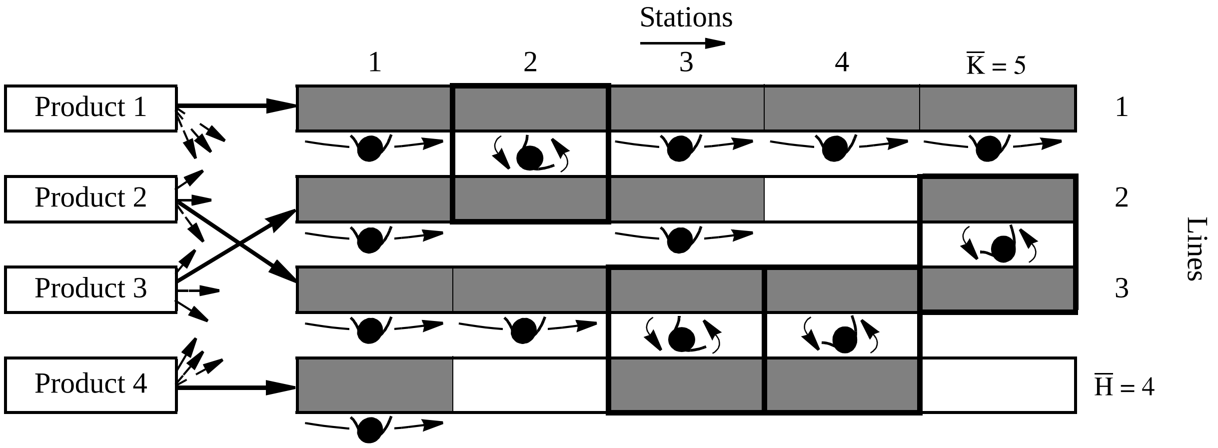

Figure 1. Products, lines, stations, workplaces and operators

In Figure 1, $\overline{H} = 4$ lines are arranged side-by-side each of which manufactures a certain product. Here, product 1 is assembled on line 1, product 2 on line 3, product 3 on line 2, and product 4 on line 4. Each line consists of (at most) $\overline{K} = 5$ stations arranged in a serial manner. In most of these stations a *normal workplace* is installed which covers a single station of a single line and is equipped by one worker who performs a set of operations (workload) on a single workpiece per cycle (symbol). At *empty stations* no workplace is installed (symbolized by non-filled stations) but space and material handling is required. *Split workplaces* cover two stations at the same stage k of two neighbouring lines (symbolized by a framed pair of stations and by the icon for a joint operator turing around to serve both stations). For example, the operator of the split workplace serving station 2 of line 1 and station 2 of line 2 has to perform some operations on the current workpiece of product 1, then has to turn around to line 2 and there he must accomplish some other tasks for the current workpiece of product 3. At the end of the cycle, he turns back to line 1 and the same procedure is repeated.

By installing split workplaces, more efficient assembly processes render possible and workforce might be reduced as is demonstrated by means of an example.

Example: We consider a problem instance with two products to be manufactured on two parallel lines. The precedence graphs for both products are depicted in Figure 2. While product 1 requires 10 tasks, product 2, which is a less demandig variant of product 1, requires only 8 tasks. The joint cycle time of both lines is $c = 10$. W.l.o.g., product 1 is assigned to line 1 and product 2 to line 2.

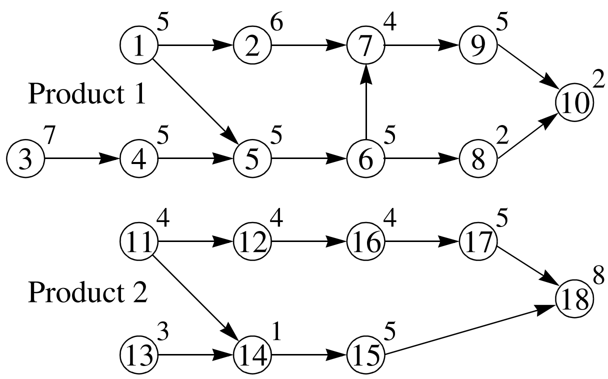

Figure 2. Precedence graphs of two products

When both lines are balanced independently (SALBP-1), the optimal balance for line 1 requires 5 stations (= workplaces = operators). The stations are loaded as follows: $S_1 = \{3\}$, $S_2 = \{1,4\}$, $S_3 = \{5,6\}$, $S_4 = \{2,7\}$, and $S_5 = \{8,9,10\}$.

The optimal balance for line 2 with 4 stations (workplaces, operators) consists of the station loads $S_1 = \{11,12\}$, $S_2 = \{13,14,16\}$, $S_3 = \{15,17\}$, and $S_4 = \{18\}$.

The total number of workplaces (= operators) required can be reduced from 9 to 8 by installing split workplaces as presented in the multi-line balance of Figure 3. For example, consider the pair of the first stations of either line which are linked by a split workplace. In each cycle, the operator first performs task 3 at a unit of product 1 on line 1 which takes 7 time units (70% of the cycle time). Afterwards, he turns around to line 2 (this changeover is assumed to consume zero time due to small distances) and performs task 13 at a unit of product 2 for 3 time units (remaining 30% of the cycle time). Similarly, each of the station pairs 3 to 7 is linked by a split workplace and served by a single operator. Due to an enhanced degree of freedom in combining tasks, no idle time remains in any station. At station 2 of line 2, a normal workplace is installed, i.e., the assigned operator exlusively performs tasks of product 1 and, thus, must not changeover to another line. The adjacent station 2 of line 2 has no work assigned, it is an empty station which only requires space and needs material handling to pass through the workpieces.

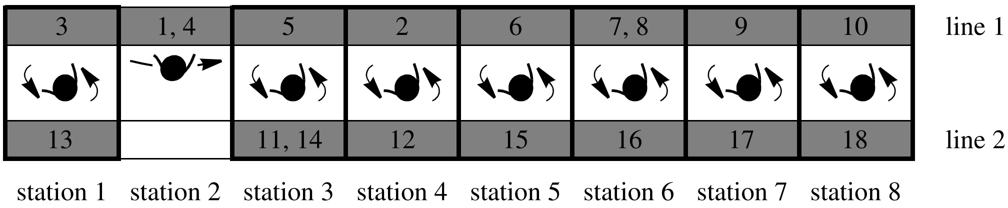

Figure 3. Solution with joint and single stations

Another multi-line balance with the same number of 8 workplaces and operators, respectively, is given in Figure 4. It contains only four split workplaces (stations 1, 4, 6, 7). In station pair 2, two operators work in parallel, one on line 1, the other on line 2 (separate normal workplaces). Concerning stations 3 and 5, a (normal) workplace is only installed on line 1, each equipped by an operator, while stations 3 and 5 of line 2 are empty.

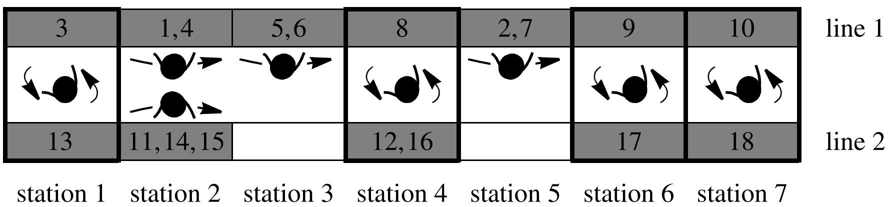

Figure 4. Another solution with joint and single stations

The examples reveal an important difference between terms usually seen as synonyms in ALB literature, i.e., stations and their loads, workplaces and operators. These terms need to be distinguished in MPALBP (as already done when describing the examples):

In MPALBP, a *station* is only a segment of a line which requires a certain space and is connected to neighbouring stations of the same line by a material handling equipment like a conveyor belt. Work can only be performed at a *workplace* which can be installed at a single station (*normal* workplace) or at two stations of neighbouring lines (*split* workplace). To keep changeover times at a minimum, only directly opposite stations (having the same station index k) should be linked by a split workplace. The workloads assigned to a workplace are performed by *operators* and accompanying equipments. MPALBLP assumes that each workplace is manned by one operator.

To summarize, multi-line balances with split workplaces have the potential advantage of increasing the efficiency of work by reducing the number of workplaces and, thus, the workforce required. This might come at cost of additional requirements in space, because split workplaces cover two stations and, moreover, empty stations might be necessary. Thus, such a parallel assembly system is usually not suited in cases where workpieces and stations are large as, e.g., in the automotive industry, but might increase productivity when workpieces and material handling systems are small and less costly. Saving space is further improved by intermittent and/or asynchronous transportation of workpieces. Examples are given in the electronics industry, e.g., production of mobile phones or laptops.

As a consequence of this discussion, the objective should not simply read as "minimize the number of stations" as stated by Gökçen et al. (2006) but "minimize the number of workplaces (operators)". Furthermore, it has to be stated that this objective and, thus, problem MPALBP is only useful if cost of employed workforce is considerably higher than cost caused by additional space requirements.

Though space requirements per station are assumed to be moderate, it is efficient and necessary to keep line lengths (measured in numbers of stations per line) short and empty stations at a minimum. This is achieved by fulfilling the property of an active multi-line balance which is defined by transferring the notion of active schedules from (project) scheduling theory (cf. Sprecher et al. 1995, Klein 2000):

Definition: A feasible multi-line balance is called *active* if no (normal or split) workplace can be shifted to the left, i.e., can get a lower station index k, without violating precedence relations and without shifting another workplace to the right.

In an active multi-line balance, each workplace gets the smallest station index possible such that no empty stations are located between workplaces if this is not necessary due to constraints. Both multi-line balances in Figure 3 and 4 are active.

Summarizing statement of the basic problem: The MPALBP is to find an active multi-line balance with minimal number of workplaces. The multi-line balance consists of (1) assigning each product to a line of the multi-line assembly production system and (2) assigning the tasks of the product-specific precedence graphs to normal workplaces of the corresponding line or to split workplaces of directly adjoining lines respecting the cycle time constraint and the precedence relations. The number of workplaces built in this manner equals the size of the workforce employed.

Possible extensions:

- If space is a rather critical issue, a second-order objective might be utilized to favour more compact and, thus, space-saving line layouts as operationalized by modification 1 of the model in Section 3 such that the balance of Figure 4 is preferred to the one of Figure 3 as it requires only 7 stations per line instead of 8 ones, not to mention the lower number of split workplaces each of which uses two stations and requires turning around or even walking between adjacent lines.
- If distances between lines are considerable, non-zero walking/changeover times might additionally be considered for split workplaces.

3. Mathematical program for MPALBP

In the following, we give a formal statement of the problem as a mathematical program. In contrast to Gökçen et al. (2006, section 4), we do not restrict the model to the balancing part of MPALBP, but also integrate the product-line-assignment decisions.[1] Additionally, the new property of *active* multi-line balances is guaranteed.

Notation:

P	set of products (with index p and number of products $\overline{P}$: $P = \{1,\dots,\overline{P}\}$)
J_p	set of tasks for product $p \in P$ (with index j and number of tasks $\overline{J_p}$: $J_p = \{1,\dots,\overline{J_p}\}$) each task is uniquely identified as the pair (p,j) of product p and task $j \in J_p$ [2]
t_{pj}	operation time of task (p,j); w.l.o.g. assumed to have integral values
A_p	set of precedence relations between tasks (p,i) and (p,j) with $p \in P$ and $i,j \in J_p$
P^*_{pj}	set of direct and indirect predecessors of task (p,j)
F^*_{pj}	set of direct and indirect followers (successors) of task (p,j)
c	joint cycle time
H	set of lines (with index h and number of lines $\overline{H} = \overline{P}$: $H = \{1,\dots,\overline{H}\}$)
K	set of stations (with index k and maximal number $\overline{K}$ of stations per line: $K = \{1,\dots,\overline{K}\}$) each station and a workplace installed there are identified as the unique pair (h,k) of line $h \in H$ and station $k \in K$

Variables:

$$x_{pjhk} = \begin{cases} 1 & \text{if task (p,j) is assigned to workplace (h,k)} \\ 0 & \text{otherwise} \end{cases} \quad \text{for } p \in P,\ j \in J_p,\ h \in H,\ k \in K$$

$$y_{ph} = \begin{cases} 1 & \text{if product p is assigned to line h} \\ 0 & \text{otherwise} \end{cases} \quad \text{for } p \in P \text{ and } h \in H$$

$$w_{pqh} = \begin{cases} 1 & \text{if products p and q are assigned to lines h and h+1} \\ 0 & \text{otherwise} \end{cases}$$
$$\text{for } p,q \in P \text{ with } p \neq q \text{ and } h \in H - \{\overline{H}\}$$

$$z_{hk} = \begin{cases} 1 & \text{if a workplace is installed at station (h,k)} \\ 0 & \text{otherwise} \end{cases} \quad \text{for } h \in H \text{ and } k \in K$$

To simplify modeling, a split workplace linking a station pair (h,k) and $(h+1,k)$ is defined to be located at (h,k), i.e., $z_{hk} = 1$ and $z_{(h+1)k} = 0$, though it serves both lines.

---

1 Though omitted in the model, the product-line-assignments are part of procedure proposed by Gökçen et al. (2006, section 3.2).

2 An alternative way of modelling tasks would consist of numbering all tasks of all products consecutively as is the case with our example of Figure 2 such that only the index j would be required. However, the pairs (p,j) facilitate to access the tasks j belonging to a certain product p. In our example, the pair notation specifies the tasks (1,1),(1,2),...,(1,10) and (2,1),(2,2),...,(2,8) instead of 1,...,10 and 11,...,18.

Model:

The **objective function** (1) minimizes the number of workplaces (operators)

$$\text{Minimize } K(X,Y,Z) = \sum_{h \in H} \sum_{k \in K} z_{hk} \tag{1}$$

subject to the following restrictions:

Product-line assignment: Each product p is assigned to exactly one line h by (2), each line h gets exactly one product p via (3). The variables w_{pqh} indicating products p and q assigned to neighbouring lines h and h+1 are derived from the product-line assignments by (4) to (6) which set w_{pqh} to 1 if and only if $y_{ph} = 1$ and $y_{q(h+1)} = 1$.

$$\sum_{h \in H} y_{ph} = 1 \qquad \text{for } p \in P \tag{2}$$

$$\sum_{p \in P} y_{ph} = 1 \qquad \text{for } h \in H \tag{3}$$

$$w_{pqh} \le y_{ph} \qquad \text{for } p,q \in P \text{ with } p \neq q \text{ and } h \in H - \{\overline{H}\} \tag{4}$$

$$w_{pqh} \le y_{q(h+1)} \qquad \text{for } p,q \in P \text{ with } p \neq q \text{ and } h \in H - \{\overline{H}\} \tag{5}$$

$$w_{pqh} \ge y_{ph} + y_{q(h+1)} - 1 \qquad \text{for } p,q \in P \text{ with } p \neq q \text{ and } h \in H - \{\overline{H}\} \tag{6}$$

Task-station-assignment: Each task j of each product p is assigned to exactly one workplace at a station k of some line h:

$$\sum_{h \in H} \sum_{k \in K} x_{pjhk} = 1 \qquad \text{for } p \in P \text{ and } j \in J_p \tag{7}$$

Cycle time restrictions: The total time of the workload (sum of operation times of tasks assigned) must not exceed the cycle time in any workplace at a station k. If station k is empty ($z_{hk} = 0$), no assignment can take place.

$$\sum_{p \in P} \sum_{j \in J_p} t_{pj} \cdot x_{pjhk} \le c \cdot z_{hk} \qquad \text{for } h \in H,\ k \in K \tag{8}$$

Precedence relations: A task j must not be assigned to a workplace at an earlier station than its predecessor i.

$$\sum_{h \in H} \sum_{k \in K} k \cdot x_{pihk} \le \sum_{h \in H} \sum_{k \in K} k \cdot x_{pjhk} \qquad \text{for } p \in P \text{ and } (i,j) \in A_p \tag{9}$$

Relating tasks and lines: Tasks (p,j) must be assigned to that line h where product p is manufactured (index h with $y_{ph} = 1$) or to line $h-1$ as expressed in restrictions (10). In the latter case, a split workplace linking the stations k of line $h-1$ and h is installed at $(h-1,k)$ such that no additional workplace can be installed at (h,k) as formulated in (11). In case of line 1, no (downward) linkage of stations is possible such that (10) reduces to (12). Restrictions (13) ensure that a normal workplace, the workload of which only contains tasks of a single product q, is installed at the line that is chosen to manufacture q. Only if at least one task of product p (assembled on line h) is assigned to a workplace at (h, k), tasks of product q (assembled on h+1) can also be assigned.

$$\sum_{k \in K} (x_{pj(h-1)k} + x_{pjhk}) \geq y_{ph} \qquad \text{for } p \in P,\ j \in J_p,\ h \in H \backslash \{1\} \tag{10}$$

$$x_{pj(h-1)k} + y_{ph} + z_{hk} \leq 2 \qquad \text{for } p \in P,\ j \in J_p,\ h \in H \backslash \{1\},\ k \in K \tag{11}$$

$$\sum_{k \in K} x_{pj1k} \geq y_{p1} \qquad \text{for } p \in P,\ j \in J_p \tag{12}$$

$$x_{qihk} \leq \sum_{j \in J_p} x_{pjhk} + (1 - w_{pqh}) \qquad \text{for } p, q \in P \text{ with } p \neq q,\ h \in H - \{\overline{H}\},\ k \in K,\ i \in J_q \tag{13}$$

Variable definition: The binary variables are defined as follows.

$$x_{pjhk} \in \{0, 1\} \qquad \text{for all } p \in P,\ j \in J_p,\ h \in H \text{ and } k \in K \tag{14}$$

$$y_{ph} \in \{0, 1\} \qquad \text{for all } p \in P \text{ and } h \in H \tag{15}$$

$$w_{pqh} = \{0, 1\} \qquad \text{for } p, q \in P \text{ with } p \neq q \text{ and } h \in H - \{\overline{H}\} \tag{16}$$

$$z_{hk} \in \{0, 1\} \qquad \text{for all } h \in H \text{ and } k \in K \tag{17}$$

Modification 1: As is the case with the original model of Gökçen et al. (2006, section 4), our model (1)-(17) does not assure that the resulting multi-line balance is active. Though this property could be easily obtained by a left-shifting procedure (as described in Section 4.1) applied to the optimal solution of the model, we describe a modified objective, because it can be used to ensure that the multi-line balance is *active* and, moreover, that it is as *compact* (space-saving) as possible:

$$\text{Minimize } K(X, Y, Z) = \sum_{h \in H} \sum_{k \in K} z_{hk} + \varepsilon \cdot \left(\sum_{h \in H} \sum_{k \in K} k \cdot z_{hk} \right) \tag{18}$$

The second term considers the sum of all the station index values of installed workplaces. If the constant ε is set to a sufficiently small value,[3] this term does not influence the primary objective of minimizing the number of workplaces but guarantees that each workplace is installed as early as possible without unnecessary empty stations in-between.

Note that replacing (1) by $\sum_h \sum_k k \cdot z_{hk}$, the sum of index values of all installed workplaces, would generate an active multi-line balance which is, however, not necessarily optimal with re-

3 A possible value is $\varepsilon = 1/(\overline{H} \cdot \overline{K} \cdot (\overline{K}+1)/2) = 2/(\overline{H} \cdot \overline{K}^2 + \overline{H} \cdot \overline{K})$ which reflects that at $\overline{H}$ lines at most $\overline{K}$ stations with maximal index sum $\overline{K} \cdot (\overline{K}+1)/2$ (arithmetic progression) can be installed. This guarantees that the second term of the objective function never gets a value greater than 1 such that taking a further workplace will never be compensated by left-shifting other workplaces.

spect to the number of workplaces. In our example, the index sum objective would favour the product-individual line balances with 5 stations at line 1 (for product 1) and 4 stations at line 2 (for product 2). This solution has 9 workplaces and index sum 25. The multi-line balance of Figure 3 contains only 8 workplaces but causes an index sum of $(8 \cdot 9)/2 = 36$. The more compact balance of Figure 4 also contains 8 workplaces at an index sum of $1+2 \cdot 2+3+4+5+6+7 = 30$ and is optimal with respect to the modified objective (18).

Modification 2: If the distances between the lines are considerable, the changeover times can be included in the model. Let d_h be the time for walking from a station k of line h to station k of line h+1 or in the opposite direction, then an additional variable is required:

$$v_{hk} = \begin{cases} 1 & \text{if a split workplace is installed at station (h,k)} \\ 0 & \text{otherwise} \end{cases} \qquad \text{for } h \in H \backslash \{\overline{H}\} \text{ and } k \in K$$

The additional restrictions (19) ensure that v_{hk} is set to 1 if at least one task j of the product q assembled at line h+1 is assigned to workplace (h, k), because this indicates a split workplace.

$$v_{hk} \geq x_{qjhk} + y_{q(h+1)} - 1 \qquad \text{for } q \in P,\ j \in J_q,\ h \in H \backslash \{\overline{H}\},\ k \in K \tag{19}$$

For all lines but the last, the cycle time restriction has to be modified to reflect that the way between line h and h+1 must be walked twice in each cycle:

$$\sum_{p \in P} \sum_{j \in J_p} t_{pj} \cdot x_{pjhk} + 2 \cdot d_h \cdot v_{hk} \leq c \cdot z_{hk} \qquad \text{for } h \in H \backslash \{\overline{H}\},\ k \in K \tag{20}$$

Further restrictions, such as $v_{hk} \leq z_{hk}$, which ensure that v_{hk} is set to zero if no split workplace is installed at station (h, k), are not required, because the model will set these variables to zero for normal workplaces if necessary (high load time near to c).

Modification 3: The number of variables can be reduced by computing earliest and latest stations based on the *relative task times* $\tau_{pj} = t_{pj} / c$ of the tasks p, j with $p \in P$ and $j \in J_p$, i.e., the portion of the cycle time required by the tasks. The earliest station E_{pj} and the latest station L_{pj}, respectively, to which a task (p, j) can be assigned feasibly if at most $\overline{K}$ stations are available per line, is computed as follows (cf. Saltzman and Baybars 1987):

$$E_{pj} := \left\lceil \tau_{pj} + \sum_{q \in P^*_{pj}} \tau_{pq} \right\rceil; \quad L_{pj} := \overline{K} + 1 - \left\lceil \tau_{pj} + \sum_{q \in F^*_{pj}} \tau_{pq} \right\rceil \qquad \text{for } p \in P,\ j \in J_p \tag{21}$$

Thus, task (p, j) can only be assigned to one of the stations out of the set of feasible stations $FS_{pj} = \{E_{pj}, E_{pj} + 1, \ldots, L_{pj}\}$. The station sets can be used to reduce the number of variables such that the variables x_{pjhk} have only to be defined for $k \in FS_{pj}$ instead of $k \in K$.

Modification 4: The variables y_{ph} could be removed from the model by using the relationships $y_{ph} = \sum_{q \in P - \{p\}} w_{pqh}$ for $p \in P$, $h \in H - \{\overline{H}\}$ and $y_{p\overline{H}} = 1 - \sum_{h \in H - \{\overline{H}\}} y_{ph}$ for $p \in P$.

In restrictions (2), (3), and (10) to (12), the variables y_{ph} would be replaced accordingly, while the restrictions (4) to (6) and (15) could be removed. Since comprehension is complicated by this reduction, we formulated the model with the help of y_{ph}.

4. An exact solution approach

We propose an exact solution procedure which combines (1) the branch-and-bound procedure ABSALOM for ARALBP-1, an extension of SALBP-1 which considers assignment restrictions, proposed by Scholl et al. (2008a), and (2) a search method enumerating on the different assignments of products to lines.

4.1. Solving MPALBP with fixed product-line assignment

If each product is assigned to a line (by setting y_{hk} such that restrictions (2) and (3) are fulfilled), the remaining MPALBP instance can be transformed to an instance of ARALBP-1. To ease the presentation, we assume that products are renumbered such that the product assigned to line h gets number h for all $h \in H$. Then, tasks of product $h \in H - \{\overline{H}\}$ and h+1 can be combined in a workplace at line h while tasks of non-adjacent products h and h+q with $q \geq 2$ must not be combined. This is prevented by defining task incompatibilities $((h,i),(h+q,j))$ for all pairs of tasks $i \in J_h$ and $j \in J_{h+q}$ with $h \in \{1,\ldots,\overline{H}-2\}$ and $q \in \{2,\ldots,\overline{H}-h\}$. Additionally, a *joint precedence graph* is built by simply combining all single precedence graphs for all products. In order to get unique task numbers the overall graph is renumbered in a topological manner. Because the single precedence graphs are unconnected components of the joint graph, each graph can be given a set of consecutive numbers as in Figure 2 (cf. footnote 2). The resulting instance can be seen as a serial single-product balancing problem of type ARALBP-1 (= SALBP-1 + task incompatibilities).

This ARALBP-instance (*A-instance* for short) is solved by applying ABSALOM, a branch-and-bound procedure for ARALBP-1 proposed by Scholl et al. (2008a). It is an extension of the well-known procedure SALOME for SALBP-1 (cf. Scholl and Klein 1997, Scholl and Klein 1999). We do not describe these procedures but refer to the original sources, because short descriptions seem not to be helpful whereas comprehensive explanations would repeat published knowledge in an ineffective manner.

Let $S_1, S_2, \ldots, S_t, \ldots, S_m$ be the (optimal) serial solution of the A-instance with m stations and $L = (L_{hk})$ the matrix of workplace loads for lines $h \in H$ and stations $k \in K$. Furthermore, consider that a load S_t can only contain tasks of adjacent products (lines) h and h+1. As a consequence, the serial solution can easily be transformed to a corresponding MPALBP solution with the same number m of installed workplaces by the following *distribution procedure*:

Start: Initialize each line $h \in H$ by setting $q_h := 0$ and $L_{hk} := \emptyset$ for $k \in K$.

Iteration: For all $t = 1, \ldots, m$ do

> Let h be the smallest index of a product of which at least one task is contained in S_t.
>
> Set $q_h := q_h + 1$ and install a workplace at station (h, q_h) with load $L_{hq_h} := S_t$.
>
> If the load L_{hq_h} also contains one or more tasks of product h+1, then mark the workplace at station (h, q_h) as a split one linking (h, q_h) and $(h+1, q_h)$, and set $q_{h+1} := q_h$.

Example: Solving the A-instance defined by Figure 2 might result in the balance $\{3,13\}$, $\{1,4\}$, $\{11,14,15\}$, $\{5,6\}$, $\{8,12,16\}$, $\{2,7\}$, $\{9,17\}$, $\{10,18\}$. Applying the distribution procedure leads to the active multi-line balance depicted in Figure 4.

In contrast to this example, the distribution procedure leads to a multi-line balance with minimal number m of workplaces which might not be active.[4] This property can be easily obtained by an additional *left-shifting procedure* which works from the left to the right, i.e., station-by-station. For each workplace at station (h,k) it is examined if it can be shifted to the left, i.e., to a still empty station (h,q) with $q<k$ without violating precedence constraints. If several left-shifts are feasible the one with smallest q is realized. For left-shifting a split workplace, a pair (h,q) and $(h+1,q)$ of empty stations is required.

Remark: If the final multi-line balance should additionally fulfill the compactness property described by the modified objective function (18), the fathoming rules of ABSALOM have to be modified such that the second-order objective is considered as well when evaluating a line balance. This requires to consider all (not only one) solutions with minimal number of stations by merely fathoming nodes of the branch-and-bound tree whose lower bound is *greater* (not equal) than the current global upper bound. To each of the unfathomed solution candidates the distribution and left-shifting procedures are to be applied to find the most compact multi-line balancing with minimal number of workplaces. In most cases this will take much more computation time and should only be used if necessary.

4.2. A search procedure

Assigning $\overline{P}$ products to the same number of lines can be done by enumerating all permutations of the products and assigning them to the lines $h = 1,\ldots,\overline{H} = \overline{P}$ in the sequence defined by the permutation. However, due to the symmetry of the system, inverted sequences, in fact, lead to the same constellations such that $\bar{I} = \overline{P}!/2$ product-line assignments are to be examined.

Though this combinatorial variety seems to be intractable, it should be mentioned that no more than 4 to 5 lines are expected to be arranged in parallel under real-world conditions. In case of 4 parallel lines, only 12 different product-line assignments, and in case of 5 lines, 60 constellations are to be considered.

Basic approach

A a very simple approach for solving MPALBP consists of enumerating all relevant product-line assignments for each of which an individual A-instance is defined and solved by ABSALOM as described in Section 4.1.[5] Let $I = \{1,\ldots,\bar{I}\}$ be the set of different A-instances and $K^*(i)$ the

4 Concerning the relationship to scheduling theory, the resulting multi-line balance is *semi-active* (cf. Sprecher et al. 1995), i.e., no workplace can be shifted to the left without changing the order of workplaces at any line. An active multi-line balance might require to change this order by using empty stations if this is precedence-feasible and does not require to shift other workplaces to the right.

5 Such a straight-forward approach is utilized by Gökçen et al. (2006, section 3.2) who apply their heuristic procedure to each product-line-assignment.

minimal number of workplaces for A-instance $i \in I$, then the MPALBP optimum requires $K^* = \min\{K^*(i) \mid i \in I\}$ workplaces.

This approach might fail to find an optimal solution, because SALBP-1 and also its generalization ARALBP-1 and, of course, MPALBP are NP-hard optimization problems (cf. Wee and Magazine 1982, Baybars 1986). Therefore, the search procedure should get more intelligence. We will describe such an extended search approach in the following. It is based on similar procedures for other extensions of SALBP (cf. Scholl et al. 2008b, 2008c), where even many more (SALBP-) instances are to be considered. In order to keep the paper short, we only shortly describe the theoretical basics and features of the procedure which can be taken over from these former approaches and concentrate on aspects to be modified.

The performance of the search procedure heavily depends on lower and upper bounds on the minimal number of workplaces, because these bounds allow for sorting the A-instances and, even better, discarding A-instances if it can be proven that they do not lead to the optimal solution. Sorting has two effects, i.e., (1) improving bounds early such that more A-instances can be discarded before they have been solved and (2) having found good solutions when the procedure is terminated due to a limit of computation time.

Lower and upper bounds

A *global lower bound* on the number of workplaces for MPALBP can be computed by transforming the entire problem instance into a SALBP-instance as described in Section 4.1 (without imposing any incompatibility constraints). The most simple bound argument for SALBP-1 is $LB1 = \lceil t_{sum} / c \rceil$, where t_{sum} denotes the sum of all task times in the overall SALBP-instance. This bound has been proposed for MPALBP by Gökçen et al. (2006, section 5). Besides this simple bound, the complete arsenal of further bounding arguments for SALBP-1 can be applied (cf. Scholl 1999, ch. 2.2.2.1; Scholl and Becker 2006). The maximum of several bound values computed serves as an initial global lower bound LB for MPALBP.

A *global upper bound* UB on the number of workplaces is given by any, more precisely, the best known feasible solution to MPALBP which can, e.g., be found by applying a priority rule based construction heuristic such as described by Gökçen et al. (2006, section 3) prior to the search procedure. However, since each (improved) feasible solution found within the search process also defines a global UB, an initial application of a heuristic showed to be not necessary in preliminary computational tests.

For each A-instance $i \in I$, a *local lower bound* LB(i) can be computed by applying bound arguments for ARALBP-1 (cf. Scholl et al. 2008a) which are, in fact, slight modifications of SALBP-1 bounds. Similarly, a *local upper bound* UB(i) is derived from applying some heuristic procedure to the A-instance i or, even better, immediately obtained when applying ABSALOM (see rule BR4 below).

Bounding rules: The bounds can be used to accelerate the search process described below in several manners:

- BR1: Each A-instance i with $LB(i) \geq UB$ is discarded, because its optimal solution cannot provide an overall improvement.

- BR2: Whenever a feasible solution with objective value $UB(i) < UB$ is obtained for any A-instance i, the global upper bound is improved by setting $UB := UB(i)$.
- BR3: The complete procedure is immediately terminated whenever $LB \geq UB$ is obtained, because the incumbent solution is optimal, i.e., $K^* = UB$.
- BR4: When ABSALOM is applied to an instance i, it almost immediately finds a first feasible solution and, thus, a first upper bound UB(i) due to the depth-first search performed. Furthermore, it successively strengthens the lower bound LB(i) due to the local lower bound method contained. In many cases, ABSALOM can terminate before finding the optimal objective value $K^*(i)$ of instance i. This is possible whenever $LB(i) \geq UB$ holds.

Advanced search procedure

Using the bounding rules and the power of ABSALOM as discussed before, a versatile search procedure can be constructed which is essentially identical to the one proposed by Scholl et al. (2008a):

(1) Initialize the search by computing the initial global bounds LB and UB.

(2) Generate the A-instances $i \in I$ in a systematic (lexicographic) manner. For each just generated A-instance i, apply ABSALOM for a very short time limit TL1 but at least until it has found a first feasible solution. This provides a lower bound LB(i) and an upper bound UB(i). If instance i is not discarded by applying the bounding rules BR1 to BR4, it is stored in a list L, together with the current bound values LB(i) and UB(i).

(3) Adjust the list L through BR1. If the remaining L is empty, then terminate the complete procedure. Otherwise, sort the list in non-decreasing order of the lower bound values LB(i).

(4) Work through the list instance-per-instance, let i be the current A-instance. If $LB < LB(i)$, then increase LB to LB(i), because i has the smallest lower bound value of all remaining A-instances due to the sorting of the list. If BR3 does not hold, apply ABSALOM to i starting with the stored LB(i) and UB(i). Apply BR2 and BR3 and remove i from L.

Result: The incumbent solution with the minimal number $K^* := UB$ of workplaces is transformed to an optimal MPALBP-solution by applying the distribution and left-shifting procedures. If the search procedure has stopped due to a time limit, it has only found a feasible (incumbent) solution with UB workplaces and a lower bound LB which allows for judging solution quality.

5. Computational experiment

In a computational experiment, we analyse the performance of our mathematical model solved by the standard solver XPressMP (MOD for short) and our solution approach based on ABSALOM (ABS for short). These procedures are compared to the heuristic of Gökçen et al. (2006, section 3) which is referred to as GÖK.

The experiment is based on the data set constructed by Gökçen et al. (2006, section 5). It consists of 95 MPALBP-instances with two products and lines, respectively, based on 14 precedence graphs with 7 to 111 tasks. In each case, the original precedence graph is used for product

1, while the precedence graph for product 2 is derived from the original one by randomly deleting 1 to 4 tasks together with their precedence relations. The data set can be downloaded from the "homepage for assembly line optimization research" *www.assembly-line-balancing.de* (select "Standard Problems" and then "MPALBP").

The experiments were run on a personal computer with an Intel Pentium IV processor of 3.0 GHz clock speed and 1.5 GByte of RAM. For each MPALBP-instance a time limit of 500 seconds was imposed for each of both procedures MOD and ABS. Unfortunately, Gökçen et al. (2006) do not report computation times for their procedure GÖK. Since they only specify the number of workplaces, we also evaluate solutions with respect to this first-order objective.

Table 1 collects the results, where K(proc) denotes the number of workplaces in the solutions computed by the procedures with $\text{proc} \in \{\text{GÖK}, \text{MOD}, \text{ABS}\}$. The 4th column specifies the optimal MPALBP objective function value K* (minimal number of workplaces) or the best known lower bound LB, if K* is still unknown. The 7th and 9th columns specify that MOD or ABS were able to find the optimum or denote the lower bound which is valid at the termination of either procedure. K*(SEP) defines the optimal number of workplaces when both lines are balanced separately (SEP). In one case, the SEP optimum is yet unknown and an interval is given. The last column reports the number of workplaces saved by installing joint stations, for some instances this number is not yet known exactly and an interval is given.

Precedence graph	#tasks: line 1 line 2	cycle time c	K* or best LB	K(GÖK)	K(MOD)	MOD opt or best LB	K(ABS)	ABS opt or best LB	K*(SEP)	K*(SEP)−K*
Mertens	7 6	9	7	7	7	opt	7	opt	7	0
		11	5	5	5	opt	5	opt	6	1
		13	5	5	5	opt	5	opt	5	0
		17	4	4	4	opt	4	opt	4	0
Jaeschke	9 8	9	8	8	8	opt	8	opt	9	1
		11	7	7	7	opt	7	opt	8	1
		13	6	6	6	opt	6	opt	6	0
		15	5	5	5	opt	5	opt	6	1
		17	4	4	4	opt	4	opt	5	1
Jackson	11 10	8	13	13	13	opt	13	opt	13	0
		10	9	9	9	opt	9	opt	10	1
		13	7	7	7	opt	7	opt	8	1
		15	6	6	6	opt	6	opt	7	1
		19	5	5	5	opt	5	opt	6	1
Roszieg	25 24	14	18	18	20	18	18	opt	20	2
		16	16	16	16	16	16	opt	16	0
		17	15	15	16	15	15	opt	16	1
		22	12	12	12	opt	12	opt	12	0
		30	9	9	9	opt	9	opt	10	1
Sawyer	30 28	25	26	26	28	26	26	opt	27	1
		27	24	25	26	24	24	opt	25	1
		30	22	22	23	22	22	opt	23	1
		36	18	18	19	18	18	opt	19	1
		41	16	16	16	opt	16	opt	16	0
		54	12	12	12	12	12	opt	13	1
Kil-bridge	45 43	57	20	20	20	opt	20	opt	20	0
		79	14	14	15	14	14	opt	14	0
		92	12	12	12	12	12	opt	12	0
		110	10	10	10	10	10	opt	11	1
		138	8	8	8	opt	8	opt	8	0
		184	6	6	6	opt	6	opt	6	0
Hahn	53 51	2004	14	14	17	14	14	opt	15	1
		2338	12	12	15	12	12	opt	13	1
		2806	10	10	11	10	10	opt	11	1
		3507	8	8	9	8	8	opt	9	1
		4676	6	6	7	6	6	opt	7	1

Table 1. Results for the MPALBP data set

Precedence graph	#tasks: line 1 line 2	cycle time c	K* or best LB	K(GÖK)	K(MOD)	MOD opt or best LB	K(ABS)	ABS opt or best LB	K*(SEP)	K*(SEP)−K*
Tonge	70 66	160	44	45	46	43	44	opt	45	1
		168	41	43	45	41	42	41	43	1-2
		207	34	34	35	34	34	opt	35	1
		234	30	30	32	30	30	opt	31	1
		270	26	26	27	26	26	opt	27	1
		293	24	24	25	24	24	opt	25	1
Wee-Mag	75 71	28	123	123	123	105	123	opt	123	0
		29	123	123	123	102	123	opt	123	0
		31	121	121	122	95	121	opt	122	1
		33	119	119	120	89	119	opt	120	1
		34	119	119	120	87	119	opt	120	1
		41	116	116	116	72	116	opt	116	0
		42	107	107	108	70	107	opt	108	1
		43	98	98	98	69	98	opt	98	0
		49	60	62	63	60	62	60	61-63	0-3
		54	60	60	61	55	60	opt	61	1
Arcus1	83 79	3786	39	40	42	39	40	39	40	0-1
		3985	37	38	40	37	37	opt	38	1
		4206	35	36	38	35	36	35	36	0-1
		4454	33	34	35	33	34	33	34	0-1
		4732	31	32	33	31	32	31	32	0-1
		5853	25	26	26	25	25	opt	26	1
		6842	22	22	23	22	22	opt	23	1
		7571	20	20	21	20	20	opt	21	1
		8412	18	18	19	18	18	opt	19	1
		10816	14	14	14	14	14	opt	15	1
Lutz2	89 85	11	86	90	97	86	88	86	94	6-8
		12	79	82	88	79	79	opt	85	6
		13	73	75	81	73	73	opt	77	4
		14	68	71	79	68	69	68	71	2-3
		15	63	64	68	63	63	opt	65	2
		16	59	60	66	59	59	opt	60	1
		17	56	56	60	56	56	opt	57	1
		19	50	50	53	50	50	opt	51	1
		20	47	48	51	47	47	opt	48	1
Lutz3	89 85	75	43	45	46	43	44	43	45	1-2
		79	41	43	45	41	41	opt	43	2
		83	39	40	43	39	39	opt	41	2
		87	37	38	40	37	37	opt	39	2
		92	35	36	37	35	35	opt	37	2
Mukherje	94 90	176	47	48	49	47	48	47	49	1-2
		183	45	46	47	45	45	opt	47	2
		192	43	44	44	43	43	opt	44	1
		201	41	42	44	41	41	opt	42	1
		211	39	40	40	39	39	opt	40	1
		222	37	38	40	37	38	37	38	0-1
		234	35	36	37	35	36	35	37	1-2
		248	33	34	35	33	33	opt	35	2
		263	31	32	33	31	32	31	33	1-2
		281	29	30	31	29	30	29	31	1-2
		301	27	28	29	27	28	27	29	1-2
		324	26	26	27	26	26	opt	27	1
		351	24	24	25	24	24	opt	25	1
Arcus2	111 107	5785	52	55	54	52	53	52	54	1-2
		6016	50	53	52	50	51	50	52	1-2
		6267	48	50	50	48	49	48	49	0-1
		6540	46	48	50	46	47	46	47	0-1
		6837	44	46	48	44	45	44	45	0-1
		7162	42	44	44	42	43	42	43	0-1

Table 1. Results for the MPALBP data set

Table 1 reveals that ABS clearly outperforms GÖK, while MOD is the worst approach in the experiment. This is confirmed by a summary of the results given in Table 2. While ABS solves 74 of 95 instances to proven optimality, GÖK finds optimal solutions in 56 cases, 45 of which can be proven to be optimal, because K(GÖK) equals the lower bound LB1 used by GÖK. The average relative deviation from optimality provided by ABS is considerably lower than the value of GÖK. In particular, improved solutions are found for large-sized instances.

	GÖK	MOD	ABS
# opt. solutions found	56	29	74
# opt. solutions proven	45	20	74
average relative deviation from opt.	1.39%	4.97%	0.57%

Table 2. Summary of the results

Solving the mathematical model (MOD) is only useful for instances with a total of at most 40 to 50 tasks. In case of larger instances, very considerable deviations from optimality might occur (the maximum relative deviation occurred is 25%). This is due to the fact that the initial LP-relaxation to be solved in the root node of the branch-and-bound tree to compute the first global lower bound might take several hundreds of seconds.

The last two columns of Table 1 show that the option of installing joint stations frequently allows for reducing the number of workplaces (i.e. operators) by 1 up to 8. A reduction is realized in 68 (or even 69, if K*(SEP)>K* for instance Wee-Mag with c=49; cf. Table 1) out of 95 instances. The average relative reduction in workforce, i.e., capacity requirement, amounts to about 4.1%. The maximal relative saving is even 20%. Such reductions make the concept of split workplaces in parallel lines an interesting option to increase productivity.

6. Conclusions and future research

In this paper, we analyzed and modelled the multiproduct parallel assembly lines balancing problem (MPALBP) originally introduced by Gökçen et al. (2006). Furthermore, a competitive exact solution procedure was developed and evaluated in a computational experiment.

The analysis revealed that the original objective function is not sufficient as it might lead to inefficiencies. This is overcome by defining the property of active multi-line balances and introducing an additional second-order objective which results in a compact multi-line layout. The model formulated in this paper extends the original one as it also includes the assignment of products to lines.

The developed solution procedure is based on transforming the problem into a single-model assembly line balancing problem with task incompatibilities (ARALBP-1) which can be solved with a known branch-and-bound procedure (ABSALOM). If more than two lines (products) are considered, each possible product-line assignment requires to define and solve such an instance. A search procedure is proposed that finds the best overall solution.

The computational experiment shows that the new procedure finds promising results as it solves small- and medium-sized instances to optimality and finds good heuristic solutions for large-sized instances. Solving the problem by a standard MIP solver seems to be no useful alternative.

As already mentioned by Gökçen et al. (2006), future research should generalize the problem MPALBP with respect to diverging cycle times and additional constraints such as obstacles hindering from linking stations by split workplaces. Moreover, the concepts of parallel lines, parallel stations, parallel tasks and parallel workplaces (cf. Becker and Scholl 2006) should be compared and, possibly, integrated in the same (real-world and model) setting, because those concepts are alternatives with respect to increasing efficiency by introducing some type of parallelism.

References

Baybars, I. 1986. A survey of exact algorithms for the simple assembly line balancing problem. *Management Sci.* **32** 909-932.

Becker, C., A. Scholl. 2006. A survey on problems and methods in generalized assembly line balancing. *European J. Oper. Res.* **168** 694-715.

Boysen, N., M. Fliedner, A. Scholl. 2007. A classification of assembly line balancing problems. *European J. Oper. Res.* **183** 674-693.

Boysen, N., M. Fliedner, A. Scholl. 2008. Assembly line balancing: Which model to use when? *Internat. J. Product. Economics* **111** 509-528.

Gökçen, H., K. Agpak, R. Benzer. 2006. Balancing of parallel assembly lines. *Internat. J. Product. Economics* **103** 600-609.

Klein, R. (2000): Scheduling of resource-constrained projects. Kluwer, Boston.

Saltzman, M.J., I. Baybars. 1987. A two-process implicit enumeration algorithm for the simple assembly line balancing problem. *European J. Oper. Res.* **32** 118-129.

Scholl, A. 1999. *Balancing and sequencing assembly lines*. 2nd ed. Physica, Heidelberg.

Scholl, A., C. Becker. 2006. State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. *European J. Oper. Res.* **168** 666-693.

Scholl, A., M. Fliedner, N. Boysen. 2008a. ABSALOM: Balancing assembly lines with assignment restrictions. *Jena Research Papers in Business and Economics (JBE)* 2/2008, FSU Jena.

Scholl, A., N. Boysen, M. Fliedner. 2008b. The sequence-dependent assembly line balancing problem. *OR Spectrum* **30** 579-609.

Scholl, A., N. Boysen, M. Fliedner. 2008c. Optimally solving the alternative subgraphs assembly line balancing problem. *Jena Research Papers in Business and Economics (JBE)* 5/2008, FSU Jena.

Scholl, A., R. Klein. 1997. SALOME: A bidirectional branch and bound procedure for assembly line balancing. *Informs J. Computing* **9** 319-334.

Scholl, A., R. Klein. 1999. Balancing assembly lines effectively - A computational comparison. *European J. Oper. Res.* **114** 50-58.

Sprecher, A.; R. Kolisch, and A. Drexl (1995): Semi-Active, active and nondelay schedules for the resource-constrained project scheduling problem. *European J. Oper. Res.* **80**, 94-102.

Wee, T.S., M.J. Magazine. 1982. Assembly line balancing as generalized bin packing. *Oper. Res. Letters* **1/2** 56–58.