Profit oriented disassembly line balancing

F. TEVHIDE ALTEKIN[1], LEVENT KANDILLER[2] and NUR EVIN OZDEMIREL[2,*]

[1]*Graduate School of Management, Sabancı University, Orhanlı, Tuzla, 34956, Istanbul, Turkey*
E-mail: altekin@sabanciuniv.edu

[2]*Industrial Engineering Department, Middle East Technical University, 06531, Ankara, Turkey*
E-mail: kandil@ie.metu.edu.tr or nurevin@ie.metu.edu.tr

We study the profit oriented partial disassembly line balancing problem, which simultaneously determines (i) the parts whose demand is to be fulfilled to generate revenue, (ii) the tasks that will release the selected parts under task and station costs, (iii) the number of stations that will be opened, (iv) the cycle time, and (v) the balance of the disassembly line, i.e. the feasible assignment of selected disassembly tasks to stations such that various types of precedence relations among the tasks are satisfied. We characterize the precedence relation types in disassembly and provide a mixed integer programming formulation of the problem. We propose a linear programming based lower and upper bounding scheme. Computational results show that our approach provides near optimal solutions for small problems and is capable of solving larger problems with up to 320 disassembly tasks in reasonable time.

***Corresponding author**, E-mail: nurevin@ie.metu.edu.tr, Phone: +90 (312) 210 47 88, Fax: +90 (312) 210 12 68

1. Introduction

Today manufacturers of products such as automobiles and electrical and electronic equipment are involved in product and material recovery activities so as to exploit economic gains, to conform to governmental legislations, and to respond to environmental concerns. The worldwide market for electronic waste had a volume of $7.2 billion in 2004, and the volume is expected to increase to $11 billion in 2009 (LaCoursiere, 2005). In North America recovery activities are carried out mainly due to economic attractiveness and a typical example is automobile recycling where recovery activities are performed as long as it is profitable to do so (Spicer and Johnson, 2004). Legislation examples include Japan's law encompassing TV sets, refrigerators, air conditioners and washing machines, and European Parliament's end-of-life vehicles directive and waste of electrical and electronic equipment (WEEE) directive. Strikingly, the stringent obligations outlined in the WEEE directive are binding not only for the European manufacturers but also for others that wish to sell their products within the EU member states (European Parliament and Council of European Union, 2003).

The recovery activities consist of collecting the discarded products, their reprocessing and redistribution. Thus manufacturers are designing their closed loop supply chains by extending their forward production systems to incorporate "reverse production systems that include remanufacturing and demanufacturing activities networked by reverse logistics systems" (Realff *et al.*, 2004).

Demanufacturing entails disassembly operations to retrieve subassemblies or parts for subsequent recovery options such as recycling, remanufacturing, refurbishing and reuse. The majority of today's demanufacturing facilities consist of a single station (Das and Caudill, 1999) where disassembly is carried out manually to retrieve highest value components or easily recyclable bulk parts. Based on a 1999 study, it is reported that (i) higher than 58%

reuse and recycling rates are achieved for products such as washing machines, computers, telephones, kettles and refrigerators; (ii) these activities are profitable; and (iii) increasing the current reuse and recycling rates to the target levels set by the WEEE Directive is expected to increase costs (Commission of the European Communities, 2000).

Wiendahl *et al*. (1998) state the possibility of improving the efficiency of a single station disassembly system up to 70% by interlinking of multiple stations. Benefits of high volume disassembly lines include achieving economies of scale and division of labor, greater degree of disassembly, and retrieval of a wider range of parts and materials (Das and Caudill, 1999). Thus, disassembly lines are proposed as one of the settings in which demanufacturing activities can be performed efficiently, because they can yield high productivity rates and are suitable for automation (Wiendahl *et al*., 1998; Das and Caudill, 1999; Güngör and Gupta, 2001a and 2002). Although the majority of today's demanufacturing facilities consist of a single station, there are numerous disassembly lines in Europe, Japan and USA that are dedicated to product families such as refrigerators, TV sets, washing machines, air conditioners and automobiles.

Similar to an assembly line, a disassembly line is made up of an ordered sequence of *stations* often connected by some mechanical material handling equipment. There is a *station cost* associated with opening and operating a station. The disassembly process consists of a set of *disassembly tasks* some of which result in removal of parts or materials that might have an associated demand. Performing a disassembly task may require certain equipment resulting in a *task cost*. The time required to perform a disassembly task is called the *task time*. Technological and physical conditions define *precedence relations* among disassembly tasks. While assembly tasks are typically characterized by *AND precedence* relations (Güngör and Gupta, 2001a), additional precedence relations are found in disassembly, such as *OR precedence* and *OR successor* relations. The additional precedence relations are mostly due to

physical restrictions or processing alternatives.

At each station, the assigned set of disassembly tasks is performed within the *cycle time*, which is common to all stations in paced lines. *Revenue* is obtained for per unit demand satisfaction of each part removed from the discarded product. Revenue can be negative implying there is a *disposal cost* associated with that part.

While all tasks must be completed to produce the end product in an assembly line, the disassembly process does not have to be *complete*. Realization of complete disassembly might be hindered by technical constraints such as irreversible connections and economic constraints such as the revenues obtained from recovered parts being lower than the disassembly costs (Lambert, 2002). In the presence of task costs and part revenues, discarded products should be disassembled to the extend it is profitable to do so. Hence disassembly is typically *partial*. Unlike assembly, in disassembly, several parts and subassemblies can be demanded in different quantities, implying different disassembly rates. As not all demand has to be satisfied, determination of disassembly rate and the corresponding cycle time is not straightforward.

The basic *disassembly line balancing problem* (DLBP) can be stated as the assignment of disassembly tasks to an ordered sequence of stations such that various forms of disassembly precedence relations are satisfied and some measure of effectiveness is optimized. *Profit oriented* DLBP aims at maximizing the profit while simultaneously deciding on (i) the *parts* whose demand will be fulfilled to generate revenue, (ii) the *tasks* that will release the selected parts under task and station costs, (iii) the *number of stations* that will be opened, (iv) the *cycle time*, and (v) the *balance of the disassembly line.*

Our purpose is to formulate and solve the profit oriented DLBP under partial disassembly. We characterize and formulate various forms of precedence relations specific to disassembly and provide a mixed integer programming formulation of the problem. In our

formulation both the number of stations and the cycle time are decision variables. This makes the problem computationally intractable even when the set of tasks to be performed is known as is in assembly line balancing problem (ALBP). The addition of part and task selection under partial disassembly further increases the complexity of the problem. The nature of disassembly requires line designs with flexibility in mind, allowing restructuring on a continuous basis as the input flow of discarded products and the demand and revenue for released parts change. In such an environment, the problem is not to make a single shot decision for a complex problem, but to make it repeatedly. Therefore, we aim at finding high quality solutions through fast heuristic solution procedures.

In the next section, we give an overview of the DLBP literature. In Section 3 the disassembly precedence relations are characterized. Our assumptions and the mixed integer mathematical formulation are provided in Section 4. Section 5 is dedicated to the linear programming based lower and the upper bounding schemes. The design and analysis of the experiment to evaluate the performance of the proposed bounding schemes are given in Section 6. Finally, the main conclusions of this study and extensions for future research are summarized in Section 7.

2. Literature review

Due to its critical role in recovery of products and materials, disassembly has recently become an active research area. The work published on disassembly planning and scheduling has been recently reviewed by Güngör and Gupta (1999), Lee *et al.* (2001), Lambert (2003) and Lambert and Gupta (2005). In the literature, most of the studies discuss the differences in physical and operational characteristics of disassembly and assembly, even though approaching disassembly as the reverse of assembly may sound reasonable.

There are a number of papers that study disassembly lines but with an emphasis on

issues other than DLBP. These studies include Penev and de Ron (1994), Wiendahl *et al.* (1998), Das and Caudill (1999), Tang *et al.* (2001) and Ketzenberg *et al.* (2004).

There are very few studies that provide solution procedures for DLBP. Güngör and Gupta (2002) study DLBP under complete disassembly. Their objective is to minimize the number of stations while meeting all the demand for recovered parts. They propose a priority rule based and station oriented heuristic solution procedure. They incorporate into the priority function station idle times, accessibility and demand level of parts, hazardous content, and changes in disassembly direction. They illustrate the proposed heuristic on an eight-task and eight-part personal computer disassembly example.

Although published earlier, Güngör and Gupta (2001a) incorporate task failures into DLBP. If a task cannot be performed because of some defect, some or all of the succeeding tasks may be disabled. This may result in various complications in the flow of used products like early leaving, self-skipping, skipping, disappearing and revisiting. The problem is to assign tasks to stations such that the effect of failures is minimized. They also modify the ALBP models they use to incorporate AND/OR precedence relations.

McGovern and Gupta (2003) present a greedy heuristic procedure for multi-objective DLBP under complete disassembly. They propose a two-phased solution where the first step constructs a feasible solution with (near) minimum number of stations using a "first-fit decreasing algorithm". The second step aims at balancing the workload of the stations and uses a 2-opt exchange local search algorithm to improve the initial solution.

Ranky *et al.* (2003) propose a dynamic disassembly line balancing algorithm that aims at smoothing the load of the shop floor. They extend the COMSOAL algorithm (Arcus, 1966) used for ALBP to obtain a balanced schedule in which all precedence relations are met and flow time of the discarded product is minimized.

None of these studies consider profit oriented nature of disassembly, allow partial

disassembly, provide a mathematical formulation, or conduct an extensive computational analysis to evaluate the performance of the proposed solution procedure.

Despite the differences between DLBP and the generalized ALBP, some ALBP studies include common features with our study. These include solving cost oriented ALBP (Amen, 2000; Buckin and Tzur, 2000), solving profit oriented ALBP (Rosenblatt and Carlson, 1985; Martin, 1994), balancing the cycle time versus the number of stations (Deckro, 1989), and solving the assembly line design problem with processing alternatives (Pinto *et al.*, 1983). We refer interested reader to Becker and Scholl (2006) for an extensive overview of the generalized ALBP literature.

As a consequence of our review, the following observations are noteworthy for our study. Amen (2000) shows that balancing stations maximally might lead to missing a cost-oriented optimal solution. Similarly, Rosenblatt and Carlson (1985) illustrate that a solution that maximizes efficiency does not necessarily maximize the profit. Deckro (1989) states that the installation and operating costs as well as the profits obtained from a line mainly depend on the number of stations and cycle time. Pinto *et al.* (1983) point out the importance of jointly considering the processing alternatives and task assignment decisions in order to minimize the total labor and fixed costs over the expected life of the production line.

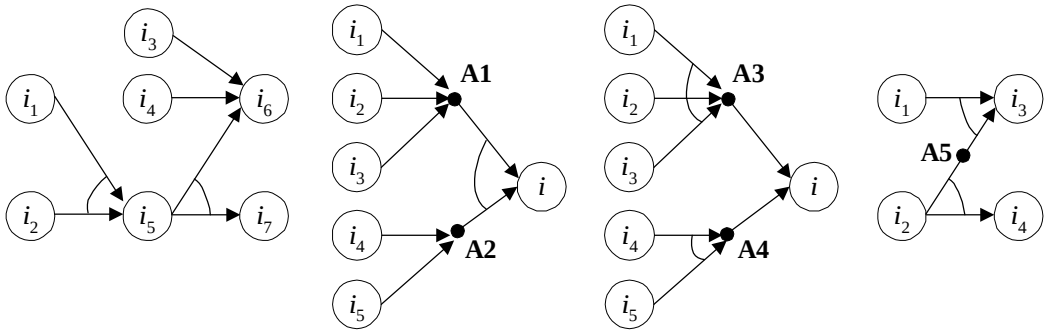
3. Precedence relations

Gottipolu and Gosh (1997) review the precedence relation representation schemes used in the assembly process planning literature while Lambert (2003) provides an overview of the same schemes for disassembly. They include precedence diagrams, state transition diagrams, liaison graphs and AND/OR graphs. The ALBP literature traditionally sticks to precedence diagrams in which tasks are represented by nodes connected by unidirected arcs depicting the binary precedence relations.

We also use the precedence diagrams. In addition to the AND, OR and complex AND/OR precedence relations introduced by Güngör and Gupta (2001b), we define a new relation type, namely the OR successor. To represent the additional relation types, we incorporate AND/OR graph representation of Homem de Mello and Sanderson (1990) and disassembly graph approach of Lambert (1999). In doing this, we transform their product structure based representation to a task based precedence diagram.

Our representation includes AND precedence, OR precedence and OR successor as the three main precedence relation types. More complex relations can be defined using these.

Tasks i_1 through i_m are (immediate) *AND predecessors* of task i, if all of them must be finished before starting task i. The same tasks are *OR predecessors* of task i, if at least one of them must be finished before starting task i. For example, in Fig. 1a tasks i_3, i_4 and i_5 are AND predecessors of task i_6, whereas tasks i_1 and i_2 are OR predecessors of task i_5. Tasks i_1 through i_m are *OR successors* of task i, if at most one of them can start after task i is finished, e.g. tasks i_6 and i_7 are OR successors of task i_5 in Fig. 1a. This precedence relation arises from the fact that, although there may be multiple ways of dividing a parent subassembly into child subassemblies, only one of them can be realized.



a. Main precedence relation types | b. AND within OR precedence | c. OR within AND precedence | d. OR successors within OR precedence

Fig. 1. Precedence relations in DLBP.

More complex precedence relation types, namely AND within OR, OR within AND, and OR precedence with OR successors, can be represented by using the three main types and

inserting dummy tasks (A1-A5 are dummy tasks in Fig. 1b through 1d). Dummy tasks have zero task time and cost and do not release parts. Note that complex AND/OR precedence relation of Güngör and Gupta (2001b) is a special case of OR within AND precedence.

4. Problem formulation

Our assumptions, some of which are adopted from Güngör and Gupta (2001a and 2002), are as follows: (i) a single discarded product is partially disassembled on a paced line; (ii) supply of discarded products is infinite; (iii) disassembly precedence relations include those given in Section 3; (iv) task times and costs are deterministic and independent of the station at which tasks are performed; (v) a disassembly task cannot be split among two or more stations; (vi) a disassembly task may result in removal of one or more parts (or subassemblies); (vii) demand quantities and revenues for parts are deterministic (demand is zero for parts with negative revenue); (viii) a station cost is incurred for opening and operating a station per unit time; and (ix) an upper limit on the cycle time is provided by the decision maker in relation with the desired disassembly rate.

We next give the notation used in our mixed integer programming formulation.

n	Number of actual disassembly tasks;
DMY	Index set of all dummy tasks, DMY = {0, A1, A2, A3,..., D}. Task 0 is the first dummy task which AND precedes all tasks, task D is the last dummy task such that all tasks are its OR predecessors, and task Ay is the y^{th} dummy task, y = 1, 2, …;
N	Total number of tasks (including dummy tasks);
i	Disassembly task index, i = 0, 1, 2,..., n, A1, A2,..., D;
j	Part index, j = 1, 2,..., J;
k	Station index, k = 1, 2,..., K (K = n when no upper bound is provided);
t_i	Task time of disassembly task i;
c_i	Cost of disassembly task i;
PAND(i)	Index set of AND predecessors of task i;
POR(i)	Index set of OR predecessors of task i;

SOR(i) Index set of OR successors of task i;

$m_{i,j}$ Number of units of part j released by task i. $m_{i,j} \geq 1$ if task i releases part j for potential sale. $m_{i,j} = -1$ if part j is in fact a subassembly which might be further disassembled (used up) by subsequent task i;

d_j Demand for part j;

r_j Revenue realized for fulfilling per unit demand of part j;

R^+ Index set of parts with nonnegative revenue;

R^- Index set of parts with negative revenue;

CT_{U} Upper limit on cycle time;

S Station cost for keeping a station open per unit time;

We use the following decision variables in our formulation.

CT Cycle time;

q_j Number of units of revenue generating part j released;

$$x_{i,k} = \begin{cases} 1 & \text{if task } i \text{ is assigned to station } k,\ i = 0,\ 1, ..., \mathrm{D} \text{ and } k = 1, 2, ..., K; \\ 0 & \text{otherwise;} \end{cases}$$

$$u_k = \begin{cases} 0 & \text{when } x_{\mathrm{D},k} = 0; \\ CT & \text{when } x_{\mathrm{D},k} = 1; \end{cases}$$

The mathematical programming model (PC) can now be formulated as follows.

$$\text{PC:} \quad \max \quad \sum_{j=1}^{J} r_j\, q_j - \sum_{i=0}^{\mathrm{D}} \sum_{k=1}^{K} c_i\, x_{i,k} - S \sum_{k=1}^{K} k\, u_k, \tag{1}$$

subject to

$$x_{i,k} \leq \sum_{h=1}^{k} x_{l,h} \qquad \forall k,\ i,\ l \in \mathrm{PAND}(i), \tag{2}$$

$$x_{i,k} \leq \sum_{h=1}^{k} \sum_{l \in \mathrm{POR}(i)} x_{l,h} \qquad \forall k,\ i, \tag{3}$$

$$\sum_{k=1}^{K} \sum_{l \in \mathrm{SOR}(i)} x_{l,k} \leq \sum_{k=1}^{K} x_{i,k} \qquad \forall i, \tag{4}$$

$$\sum_{l \in \mathrm{SOR}(i)} x_{l,k} \leq \sum_{h=1}^{k} x_{i,h} \qquad \forall k,\ i, \tag{5}$$

$$\sum_{k=1}^{K} x_{i,k} \leq 1 \qquad \forall i, \tag{6}$$

$$\sum_{i=0}^{\mathrm{D}} t_i \; x_{i,k} \le CT \qquad \forall k, \tag{7}$$

$$q_j \le d_j \qquad \forall j \in R^+, \tag{8}$$

$$q_j \le \sum_{i=0}^{\mathrm{D}} \sum_{k=0}^{K} m_{i,j} \; x_{i,k} \qquad \forall j \in R^+, \tag{9}$$

$$q_j \ge \sum_{i=0}^{\mathrm{D}} \sum_{k=0}^{K} m_{i,j} \; x_{i,k} \qquad \forall j \in R^-, \tag{10}$$

$$\sum_{k=1}^{K} k \; x_{i,k} \le \sum_{k=1}^{K} k \; x_{\mathrm{D},k} \qquad \forall i, \tag{11}$$

$$x_{i,k} \le \sum_{i=0}^{\mathrm{D}} t_i \; x_{i,k} \qquad \forall k, \; i \in \mathrm{DMY}, \tag{12}$$

$$u_k \le (\sum_{i=0}^{\mathrm{D}} t_i) \; x_{\mathrm{D},k} \qquad \forall k, \tag{13}$$

$$\sum_{k=1}^{K} u_k \ge CT, \tag{14}$$

$$CT \le CT_{\mathrm{U}}, \tag{15}$$

$$CT, \; u_k, \; q_j \ge 0 \qquad \forall k, \; j, \tag{16}$$

$$x_{i,k} \in \{0, \; 1\} \qquad \forall k, \; i. \tag{17}$$

The objective function (1) maximizes profit per cycle that is the difference between total part revenues and the total of task and station costs. Constraint sets (2), (3) and (4-5) handle the AND precedence, OR precedence and OR successor relations. Set (2) allows assignment of task *i* to station *k* only if all of its AND predecessors are already assigned to stations 1 through *k*. Set (3) does not allow assignment of task *i* to station *k* unless at least one of its OR predecessors is assigned to one of the stations 1 through *k*. Set (4) permits assignment of at most one of the OR successors of task *i* (to some station) if task *i* is performed. Set (5) allows assignment of the OR successors of task *i* to station *k*, if task *i* is assigned to one of the stations 1 through *k*. Constraint set (6) indicates that a task can be assigned to at most one station. It takes the form of equality for "must do" tasks. The cycle time constraint set (7) enforces the work content of each station to remain within the cycle

time. Constraint sets (8) and (9) determine the number of units of a positive revenue generating part as the minimum of its demand and the total number of units released. These constraints are needed since a discarded product may contain multiple units of the same part. Similarly constraint set (10) evaluates the number of released units for a part with negative revenue. Constraint set (11) assures that no task is assigned to the stations following the station to which dummy task D is assigned, thereby determining the number of stations opened. Constraint set (12) ensures assignment of dummy tasks to stations to which some actual tasks are assigned. These two sets can be perceived as technical constraints. Constraint sets (13) and (14) determine the total available time on the line, which would normally be found by multiplying two decision variables, namely the number of stations opened and the cycle time. Decision variable u_k and these two constraints are used for linearizing this term, which is used in calculating the total station cost component of the objective function. Constraint (15) ensures that the cycle time decision variable is less than or equal to its upper limit determined by the decision maker.

5. Linear programming based bounding schemes

5.1. *Upper bound on profit*

Solving the linear programming relaxation of PC (PC-LP) yields a weak upper bound on the profit. This is expected because PC-LP assigns tasks fractionally to multiple stations, which may give rise to the following consequences: (i) all K stations are used with zero idle time, thus cycle time is minimized; (ii) among the u_k variables associated with the stations to which task D is assigned, the one with the smallest (instead of the largest) index takes the value of CT to minimize the station opening cost component; this leads to undercosting of station opening in the objective function; (iii) sum of the fractional assignments of a task exceeds that of each of its AND predecessors (or all of its OR predecessors).

Strengthening the PC-LP solution is possible by rendering such infeasible solutions with consequences (ii) and (iii) through the introduction of two "logical inequalities".

The first logical inequality (LI-1) below assures that the total time used over all stations does not exceed the total time available on the opened stations.

$$\sum_{k=1}^{K}\sum_{i=0}^{D} t_i \; x_{i,k} \le \sum_{k=1}^{K} k \; u_k \qquad (17)$$

Two forms of the second logical inequality (LI-2) are proposed for AND and OR precedence relations. Constraint (18) guarantees that the total fractional assignment of task *i* does not exceed the total fractional assignment of each of its AND predecessors. Similarly, constraint (19) assures that the total fractional assignment of task *i* is less than or equal to the total fractional assignment of all of its OR predecessors.

$$\sum_{k=1}^{K} x_{i,k} \le \sum_{k=1}^{K} x_{l,k} \quad \forall i \text{ and } l \in \text{PAND}(i) \qquad (18)$$

$$\sum_{k=1}^{K} x_{i,k} \le \sum_{k=1}^{K} \sum_{l \in \text{POR}(i)} x_{l,k} \quad \forall i \qquad (19)$$

It is possible to show that (17), (18) and (19) are stronger than the respective surrogate constraints that would be obtained by summing up (7), (2) and (3) over all *K* stations.

We call the PC-LP with additional constraints sets (17), (18) and (19) the ***strengthened LP relaxation*** (PC-SLP) and use its objective function value as an upper bound.

5.2. *Lower bound on profit*

Our lower bounding scheme starts with a construction heuristic that takes the total fractional task assignments from the upper bound solution of PC-SLP and converts them to feasible (binary) assignments. Then, a two step improvement heuristic re-evaluates the task selection and assignment to maximize the profit.

Construction Heuristic

Our construction heuristic starts by assuming all tasks with a positive fractional

assignment in the upper bound solution are "selected" and proceeds with a station oriented assignment procedure (SOAP). SOAP prioritizes the selected tasks using each of the following numerical scores.

$NX_1(i)$ = The sum of task times for task i and all of its successors, i.e. positional weight (Helgeson and Birnie, 1961);

$NX_2(i)$ = The number of immediate successors of task i (Mastor, 1970);

$NX_3(i)$ = The number of all successors (Tonge, 1965);

$NX_4(i)$ = The sum of revenues generated by parts released by task i and all of its successors;

$NX_5(i)$ = The sum of task times over all tasks - $NX_1(i)$.

Given a cycle time value, we repeat the task assignment independently using each numerical score. In each iteration of the procedure, among the assignable (i.e. precedence feasible and fitable to the current station) tasks, the one with the highest numerical score is selected and assigned to the current station. When there are no more tasks assignable to the current station, a new station is opened.

The cycle time is bounded from below by the minimum task time of the selected tasks (t_{min}) and from above by a prespecified upper limit. We repeat SOAP for all possible integral values of the cycle time within these bounds. Among the resulting line balances, the one yielding the highest profit is selected. If the highest profit is negative then the best solution is taken as no disassembly with a profit of zero.

Our assignment procedure has the following differences from ALBP procedures in handling of the precedence relations: (i) precedence feasibility check includes OR precedence; (ii) in calculating numerical scores for a task having OR successors, we take the average over all OR successors; (iii) as soon as one of the OR successors of a task is assigned, all the remaining OR successors of that task are dropped from further consideration.

The time complexity of SOAP for a single cycle time value is $O(N^3 \log N)$ when all numerical score computations are considered. The overall complexity of the construction heuristic is $O((CT_U - t_{min}+1)N^3 \log N)$ and thus is pseudo polynomial.

Improvement Heuristic

We apply a two step improvement heuristic to the initial solution obtained by the construction heuristic. If the initial set of tasks selected by the PC-SLP solution contains tasks with fractional (less than 1.0) total assignments, a *task deletion heuristic* is used in the first step. We eliminate tasks that were fractionally assigned in the upper bound solution and became redundant after other fractional tasks are fully assigned by the construction heuristic. In each iteration we attempt to delete one of the selected tasks and re-execute SOAP. If a higher profit value is attained the task is deleted permanently. We repeat the entire process twice, forward starting with task 0 and backward starting with task D, and choose the highest profit solution.

In the second step a *task insertion heuristic* is applied on the solution of task deletion. In our lower bounding scheme, task selection and assignment decisions are not made simultaneously and the idle time cost does not affect the task selection. The task insertion heuristic has a potential of increasing the profit by decreasing station idle times. Unselected tasks are considered for insertion first in forward direction and then in backward direction. For each insertion possibility SOAP is re-executed in an attempt to attain higher profit.

A heuristic upper bound on the number of stations (K_{UB})

In solving PC and PC-SLP, the upper bound on the number of stations that can be opened (K) is ordinarily set to the number of actual tasks (n). This is a very loose bound and, for large problem instances, a tighter upper bound on the number of stations is needed. The idea here is to allow assignment of as many tasks as possible as long as they release revenue generating parts. To facilitate this we multiply the total cost component in the objective with a small value e (we still need this term to prevent assignment of multiple OR predecessors unnecessarily). We also set $K = 1$. The resulting objective function is presented in (20).

PK-UB: max $\sum_{j=1}^{J} r_j \, q_j - e \sum_{i=0}^{D} (c_i + S \, t_i) \, x_{i,1}$ (20)

subject to

$q_j \geq 0 \quad \forall j$

(2) - (6), (8) - (12), (16)

A set of tasks is selected by solving the linear programming relaxation of PK-UB. Then, all tasks with positive fractional assignments are assigned to stations using SOAP. In applying the assignment procedure, the cycle time is set to the maximum task time of the selected tasks to ensure that the assignment yields the maximum number of stations. Note that the K_{UB} found in this manner may be smaller than the true upper bound. We resort to this heuristic upper bound only for large problem instances.

6. Computational analysis

6.1. *Problem generation*

The computational analysis conducted is based on ten problems. Five of these problems are originally created during the conduct of this study. The sixth problem uses the precedence relation and part release structure of a personal computer example (Güngör and Gupta, 2002). Precedence relations of the seventh and ninth problems are taken from the disassembly process planning literature (Lambert, 1999) and are transformed to equivalent task based precedence diagrams. In these problems there is only demand for end parts. Eighth and tenth problems are similar to these two, but we assume that there is also demand for all subassemblies released at intermediate stages. The number of actual tasks in these ten problems varies between 8 and 30 while the number of parts released is between 4 and 29. In Table 1 each problem is represented by a code that denotes the author(s) who defined the problem (first three letters), the number of actual tasks and the number of parts (or subassemblies) that have demand. We also include in the table the total number of AND

precedence, OR precedence and OR successor relations, the number of continuous and binary variables and constraints in PC formulation with $K = n$.

Table 1. Basic features of the problems

Problem no	Problem code	Actual tasks	Total tasks	Total parts	AND prec.	OR prec.	OR succ.	Cont. var.	Bin. var.	Con-straints
1	AKO8T6P	8	11	6	9	3	0	15	88	173
2	AKO20T4P-A	20	22	4	15	6	1	25	440	576
3	AKO20T4P-B	20	22	4	15	5	1	25	440	556
4	AKO20T4P-C	20	22	4	15	5	1	25	440	556
5	AKO30T12P	30	39	12	22	9	5	43	1170	1520
6	GUN8T8P	8	10	8	10	2	0	17	80	167
7	LAM20T10P	20	25	10	5	8	5	31	500	578
8	LAM20T24P	20	25	24	5	8	5	45	500	606
9	LAM30T10P	30	49	10	16	11	10	41	1470	1871
10	LAM30T29P	30	49	29	16	11	10	60	1470	1909

Each of these ten problems is conceived as a problem category. Ten instances are created for each problem category by randomly generating the task times and costs, part demands and revenues. The base station cost per unit time is set to unity. However, eight different levels of the station cost that range from a quarter of the base level to twice of the base level are used in solving each instance.

The task times for an instance are generated from discrete uniform distribution between one and twenty. To create a tradeoff with the base station cost due to task times, the task costs are generated from discrete uniform distribution where the lower limit is one and the upper limit is found assuming the mean is equal to the average of the generated task times.

We further assume that the total revenue over all parts is equal to the total costs. Hence, in generating the discrete uniform part revenues we determine the limits using the average and variance of the total cost figures. We allow a 5% probability of having negative revenue. Finally, we use discrete uniform distribution to generate low demand (between zero and ten) and high demand (between ten and 100, with 5% chance of no demand) for parts.

We need an upper limit on the cycle time, which we assume is provided by the decision maker. To determine this value in our computational analysis we stick to Güngör and Gupta's

(2001a) cycle time determination approach, which assumes all demand is to be met within the planning horizon. In implementing their approach we take the planning horizon as 400 (4000) time units in the low (high) demand case. For sensitivity analysis purposes, after solving each instance for low demand, we recalculated the upper limit on cycle time considering the demand of only those parts that are released in the optimal solution. If this limit was different from the original one, we solved PC again with the new limit to see if the solution changed. Such change was observed (the cycle time limit was binding) in only 12 of 800 instances.

6.2. *Computational results*

All of our routines are coded in Visual Studio C 6.0. Callable libraries of ILOG CPLEX 9.0 are invoked to solve PC, PC-LP and PC-SLP throughout the computational analysis.

We first compare PC-LP, PC-LP with LI-1, PC-LP with LI-2, and PC-SLP having both logical inequalities. Over all 800 cases of low and high demand levels, LI-1 alone results in a 35% average reduction in upper bound over PC-LP. The same figure is 28% when LI-2 is used alone. When both are used simultaneously a 66% average reduction is achieved over the PC-LP solution, which means that our logical inequalities have been effective.

We also examine the number of times the best solution is found by each numerical score over all 800 instances of both demand levels. The first numerical score (positional weight) yields the best solution in more than 79% of all cases with positive profit. The first three ALBP numerical scores, which rely on task times and precedence relations, show close performance. This is expected since tasks with a large number of successors might also have a large sum of task times for its successors. The fourth one follows the first three with 73% success rate. The fifth score is in a sense the complement of the first one and apparently creates variety. When we consider the number of times the best solution is *uniquely* found, the fifth score is the most effective; it finds the best solution in 13% of all cases where the other four numerical scores fail to do so.

We next evaluate the performance of our bounds. In solving PC to optimality we first set K to the number of actual tasks (n) and allow MIP solver of ILOG CPLEX 9.0 to search for the optimum for six hours. If the optimum is not found within this time limit, we let K take the value of the heuristic upper bound (K_{UB}) and re-run MIP solver for another six hours.

The average and maximum deviations from the optimal of the linear programming relaxation solutions (PC-LP), the upper bound (UB) obtained by PC-SLP, and the lower bound (LB) are summarized for each problem category in Tables 2 and 3. We also report the number of times the optimal solution is found. PC-LP indeed yields a very loose upper bound on profit, with over 400% average deviation from the optimal. Average deviation of UB, on the other hand, is 9% and 10% for the low and high demand data sets, respectively. This empirically illustrates that the two logical inequalities used in PC-SLP have strengthened the linear programming relaxation considerably. UB's overall maximum deviation is 500% and 750% for low and high demand levels. These maximum deviations correspond to instances where the optimal profit is 1 and UB is found as 6 and 8.5, respectively. UB finds the optimal profit in more than 35% of the instances in both data sets, but the solution may be infeasible.

Table 2. Solution quality of bounds for low demand data set

	PC-LP			UB (PC-SLP)			LB		
PROBLEM	AVG[+]	MAX[+]	NOS[*]	AVG[+]	MAX[+]	NOS[*]	AVG[+]	MAX[+]	NOS[*]
AKO8T6P	454	4635	0	3.75	56.82	53	0.19	5.49	76
AKO20T4P-A	455	22493	0	15.46	500.00	9	0.43	3.13	55
AKO20T4P-B	454	11792	0	24.51	205.26	9	3.14	71.05	69
AKO20T4P-C	202	627	0	15.55	56.44	14	3.21	100.00	50
AKO30T12P	598	6473	0	4.39	50.00	13	5.36	54.67	14
GUN8T8P	725	6053	0	2.54	42.86	63	0.08	5.00	79
LAM20T10P	185	547	0	1.19	16.44	50	0.19	2.94	67
LAM20T24P	346	3708	0	8.55	341.62	53	0.07	4.48	79
LAM30T10P	247	458	0	1.04	4.04	16	0.07	1.04	70
LAM30T29P	621	9066	0	15.84	446.40	29	4.34	100.00	68
OVERALL	422	22493	0	8.86	500.00	309	1.70	100.00	627

[*] NOS: Number of optimal solutions found out of 80 cases (10 problem instances x 8 station cost levels).
[+] AVG and MAX: Average and maximum percentage deviations from the optimal over cases with nonzero optimal objective function value (out of 80 cases).

Table 3. Solution quality of bounds for high demand data set

	PC-LP			UB (PC-SLP)			LB		
PROBLEM	AVG[+]	MAX[+]	NOS[*]	AVG[+]	MAX[+]	NOS[*]	AVG[+]	MAX[+]	NOS[*]
AKO8T6P	441	6255	0	5.55	183.33	44	1.57	37.06	73
AKO20T4P-A	920	31147	0	26.09	750.00	4	2.63	100.00	49
AKO20T4P-B	443	9149	0	20.14	205.26	14	3.06	100.00	70
AKO20T4P-C	203	1247	0	28.85	108.23	13	5.33	100.00	49
AKO30T12P	362	2062	0	2.82	17.14	18	3.16	22.86	10
GUN8T8P	761	7775	0	3.50	75.00	59	1.22	30.77	74
LAM20T10P	188	777	0	1.59	27.91	39	0.10	2.80	71
LAM20T24P	356	5323	0	9.50	341.62	46	0.50	11.11	75
LAM30T10P	288	642	0	0.98	4.04	16	0.07	1.04	70
LAM30T29P	441	6013	0	7.62	160.00	28	2.17	100.00	69
OVERALL	436	31147	0	10.37	750.00	281	1.95	100.00	610

[*] NOS: Number of optimal solutions found out of 80 cases (10 problem instances x 8 station cost levels).
[+] AVG and MAX: Average and maximum percentage deviations from the optimal over cases with nonzero optimal objective function value (out of 80 cases).

LB finds the (feasible) optimal solution in 627 instances (78%) for low demand level and 610 instances (76%) for high demand level. The average deviation from the optimal is less than 2% for both demand levels, when the optimal profit is positive. When the 90 and 79 instances with zero optimal objective function value are included for low and high demand, average deviations drop to 1.5% and 1.8%. As for the contribution of the heuristics, LB is found by the construction heuristic in 1423 of the 1600 instances. The task deletion step of our improvement heuristic yields the LB in 168 of the remaining instances, and the task insertion step terminates with success only in 9 cases. The construction heuristic on the average deviates 12% from the LB, and the task deletion heuristic 0.3%.

LB's performance deteriorates as the number of stations in the solution increases and the ratio of the number of parts to the number of tasks decreases. For example, the number of stations opened for instances of AKO30T12P varies between 1 and 8. Thus, the detriment of not balancing the line optimally emerges more significantly as higher station costs are incurred. On the other hand, in problems AKO20T4P-A, B and C, a low ratio of the number of parts to the number of tasks leads to worse UBs, which in turn affects LB solution quality.

Figure 2 illustrates the average percentage deviations of UB and LB from the optimal

across eight station cost levels for low demand (LD) and high demand (HD) data sets. Deviation of LB is always within 5%. The performance of both bounds deteriorates with increasing station cost, as expected. This is mainly because the performance of UB deteriorates as the station cost increases, as PC-SLP does not incur idle time cost for stations.

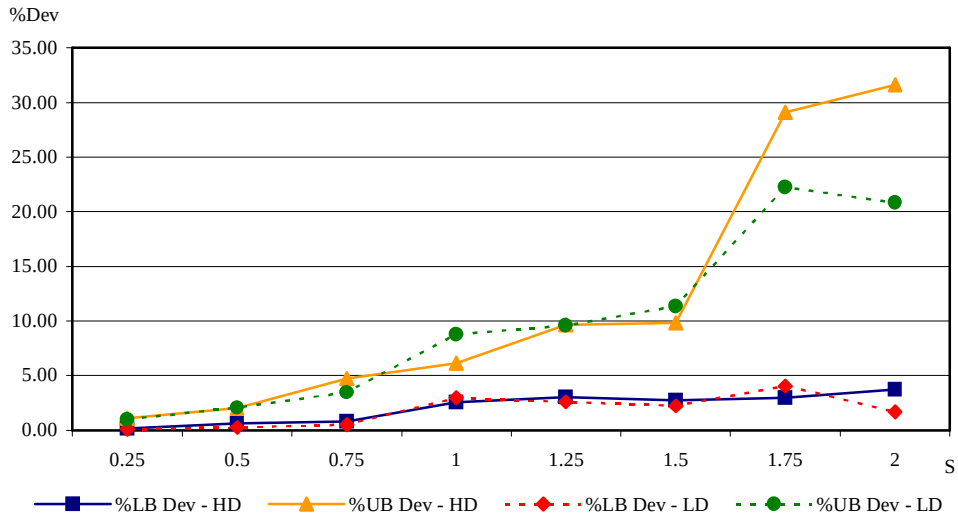


Fig. 2. Average percentage deviation of bounds from the optimal.

Tables 4 and 5 give the average and standard deviation of the CPU times to find the solutions over 80 instances of each problem category. These computations have been made on a Toshiba Satellite Notebook with a Celeron 2800 MHz processor and 728 MB RAM. According to these tables, PC-SLP finds the UB in less than three seconds. Our LB scheme runs in less than four seconds including the UB time, which is part of the solution procedure.

When the overall performance of LB is analyzed, it is seen that our heuristic procedure yields near optimal solutions for small problems in very short time. Next, we attempt to find the largest problem size that can be solved using the proposed procedure.

Table 4. CPU solution times (in seconds) for low demand data set

	PC (MIP)		PC-LP		UB (PC-SLP)		LB***	
PROBLEM*	AVG	STD	AVG	STD	AVG	STD	AVG	STD
AKO8T6P (80)	1.00	0.27	0.03	0.01	0.03	0.01	0.05	0.02
AKO20T4P-A (74)	3173.26	7071.69	0.27	0.06	0.22	0.13	0.32	0.07
AKO20T4P-B (68)	2820.72	5223.73	0.24	0.04	0.22	0.21	0.30	0.08
AKO20T4P-C (72)	2714.32	6276.39	0.29	0.13	0.18	0.06	0.31	0.10
AKO30T12P**	2387.22	3769.91	3.01	1.04	1.31	0.36	1.85	0.42
GUN8T8P (80)	0.80	0.19	0.03	0.01	0.02	0.01	0.04	0.02
LAM20T10P (80)	1076.56	3699.63	0.40	0.07	0.29	0.22	0.42	0.06
LAM20T24P (80)	37.91	32.98	0.27	0.08	0.24	0.08	0.42	0.11
LAM30T10P**	115.10	144.97	3.62	1.25	2.30	0.69	3.06	0.80
LAM30T29P (75)	5240.09	9058.62	2.53	1.08	2.33	1.56	2.90	1.50

* With two exceptions, optimum is first sought for six hours using $K = n$ and then with $K = K_{UB}$. The figures in parenthesis indicate the number of instances out of 80 solved with $K = n$.
** Optimum is directly sought using $K = K_{UB}$.
*** LB times include UB times.

Table 5. CPU solution times (in seconds) for high demand data set

	PC (MIP)		PC-LP		UB (PC-SLP)		LB***	
PROBLEM*	AVG	STD	AVG	STD	AVG	STD	AVG	STD
AKO8T6P (80)	0.96	0.25	0.03	0.01	0.03	0.01	0.05	0.01
AKO20T4P-A (74)	994.95	3692.26	0.27	0.06	0.21	0.12	0.31	0.06
AKO20T4P-B (68)	1044.88	3368.02	0.25	0.04	0.20	0.05	0.31	0.09
AKO20T4P-C (72)	1648.75	4035.45	0.29	0.12	0.18	0.06	0.30	0.09
AKO30T12P**	2053.52	3727.58	3.02	1.08	1.39	0.37	1.74	0.39
GUN8T8P (80)	0.76	0.20	0.03	0.01	0.02	0.01	0.04	0.01
LAM20T10P (80)	560.29	3212.07	0.42	0.10	0.26	0.06	0.42	0.06
LAM20T24P (80)	432.19	3218.63	0.29	0.17	0.26	0.12	0.40	0.10
LAM30T10P**	67.82	77.30	3.69	1.39	2.32	0.67	2.96	0.72
LAM30T29P (75)	5935.30	9472.69	2.18	0.75	2.00	1.30	2.59	1.28

* With two exceptions, optimum is first sought for six hours using $K = n$ and then with $K = K_{UB}$. The figures in parenthesis indicate the number of instances out of 80 solved with $K = n$.
** Optimum is directly sought using $K = K_{UB}$.
*** LB times include UB times.

Due to unavailability of large test problems in DLBP literature, we merge three of our test problems, namely LAM30T29P, AKO20T4P-B, and AKO30T12P from the low demand data set, to obtain a larger problem with 80 actual tasks. A precedence diagram is created to connect these three problems in parallel. Similarly, we obtain a problem with 160 actual tasks by putting two 80-task problems together. Finally, the size of the problem is doubled once more to obtain a 320-task problem. We stop increasing the problem size at this point since this problem is larger than the largest common ALBP test problem with 297 tasks (Scholl, 1999). We let $K = K_{UB}$ and $CT_U = 40$ in solving PC-SLP formulation.

Table 6. Summary of results for large problems

Instance	S	K_{UB}	LB CT	LB K	LB profit	UB profit	%UB-LB gap	LB CPU (sec.)	UB CPU (sec.)
Large Problem 1 n = 80 N = 112	0.25	19	37	9	586.75	590.25	0.59	21.27	20.66
	0.50	19	40	8	507.00	510.50	0.69	50.50	47.91
	0.75	19	40	8	427.00	432.25	1.21	9.75	9.02
	1.00	19	40	8	347.00	354.00	1.98	2.06	1.45
	1.25	19	37	8	266.00	276.00	3.62	20.41	19.75
	1.50	19	37	8	192.00	204.00	5.88	1.77	1.17
	1.75	18	36	7	116.00	140.50	17.44	1.56	1.05
	2.00	17	37	4	66.00	87.00	24.14	3.86	2.11
Large Problem 2 n = 160 N = 222	0.25	39	37	19	1318.25	1322.00	0.28	11.45	8.78
	0.50	39	37	19	1142.50	1150.00	0.65	14.75	8.94
	0.75	39	39	18	964.50	979.50	1.53	12.75	8.95
	1.00	38	36	17	795.00	816.00	2.57	13.06	6.86
	1.25	38	39	15	644.75	670.75	3.88	17.88	7.47
	1.50	38	38	15	502.00	538.17	6.72	18.84	6.36
	1.75	37	40	10	391.00	416.81	6.19	20.81	6.47
	2.00	36	40	10	291.00	321.00	9.35	14.80	6.09
Large Problem 3 n = 320 N = 442	0.25	80	30	49	2573.50	2585.00	0.44	145.91	111.78
	0.50	78	31	46	2214.00	2233.00	0.85	133.91	109.08
	0.75	79	37	34	1865.50	1889.88	1.29	245.02	198.24
	1.00	78	35	32	1567.00	1591.00	1.51	131.00	95.99
	1.25	76	35	31	1286.75	1320.25	2.54	586.39	540.22
	1.50	78	40	24	1010.00	1066.17	5.27	159.11	85.44
	1.75	75	39	22	776.50	833.31	6.82	161.25	76.47
	2.00	75	40	19	563.00	637.00	11.62	131.83	70.86

In Table 6 we summarize the results for eight station cost levels. Although the overall average gap between UB and LB is less than 4%, we observe a similar monotonic increase as the station opening cost increases. LB CPU times again include the UB times.

7. Conclusions

We provided a mixed integer programming formulation for the profit oriented partial DLBP. We strengthened the LP relaxation of the formulation to obtain an upper bound on profit. An initial feasible solution was constructed based on this upper bound solution and improved with neighborhood search heuristics. We also conducted a computational analysis to evaluate the performance of the proposed bounding schemes.

Our approach found the optimum in 77% of the 1600 instances solved. The average deviation from the optimum was less than 2% and these solutions were obtained within 1

second of CPU time on the average. A DLBP problem instance with 320 disassembly tasks was solved in 10 minutes. Our upper bounding scheme alone found the optimum in 35% of the 1600 instances with an average deviation from the optimum of 10%.

Possible further research directions related with the proposed solution procedure include devising a new basis for determining an upper limit on the cycle time, tightening the upper bounds via other relaxations, and developing optimum seeking solution procedures. Although choosing an effective branching strategy would be a challenge, the tight upper and lower bounds we propose could be employed in approaches such as branch and bound, branch and cut and beam search. The issue would be to sequence the branching decisions including choice of the cycle time, part selection, task selection, and task assignment.

It is possible to extend DLBP to cover mixed model DLBP with profit maximization by combining multiple precedence diagrams. Stochastic version of the DLBP can be considered as disassembly tasks are known to have highly variable task times. Uncertainties in the quality of incoming discarded products and in the flow of parts due to task failures can also be incorporated. DLBP can be formulated and solved for unpaced lines as well.

It is possible to use the proposed precedence diagram representation (with OR predecessors and OR successors) and solution procedure in ALBP. If, in addition, subassemblies can be purchased or sold, then the profit oriented, cost oriented, Type-I, Type-II, and Type-E ALBP problems can be formulated and solved by adapting the proposed PC formulation and solution procedure.

Finally, systems that integrate assembly and disassembly, as in maintenance or renovation of tanks and aircraft, can be studied. This can be done sequentially (start assembly after disassembly is finished), or assembly and disassembly can be conducted simultaneously as in hybrid manufacturing/remanufacturing systems.

References

Amen, M. (2000) Heuristic methods for cost-oriented assembly line balancing: A survey. *International Journal of Production Economics*, **68**, 1-14.

Arcus, A. L. (1966) COMSOAL: A computer method for assembly line balancing with more than one worker in each station. *International Journal of Production Research*, **4**, 259-277.

Becker, C. and Scholl, A. (2006) A survey on problems and methods in generalized assembly line balancing. *European Journal of Operational Research*, **163**, 694-715.

Buckhin, J. and Tzur, M. (2000) Design of flexible assembly line to minimize equipment cost. *IIE Transactions*, **32**, 585-598.

Das, S. K. and Caudill, R. (1999) The design of high volume disassembly lines, in Demanufacturing of Electronic Equipment for Reuse and Recycling Information Exchange Meeting Archives, 25-26 October 1999.

Deckro, R. F. (1989) Balancing cycle time and workstations. *IIE Transactions*, **21**, 106-111.

Commission of the European Communities (2000) Draft proposal for a European Parliament and Council directive on waste electrical and electronic equipment, Brussels, Belgium.

European Parliament and Council of European Union (2003) Directive 2002/96/EC of the European Parliament and of the Council of 27 January 2003 on waste electrical and electronic equipment. Official Journal of the European Union, 13.2.2003, L 37/24.

Gottipolu, R. B. and Ghosh, K. (1997) Representation and selection of assembly sequences in computer-aided assembly process planning. *International Journal of Production Research*, **35**, 3447-3465.

Güngör, A. and Gupta, S. M. (1999) Issues in environmentally conscious manufacturing and product recovery. *Computers & Industrial Engineering*, **36**, 811-853.

Güngör, A. and Gupta, S. M. (2001a) A solution approach to the disassembly line balancing problem in the presence of task failures. *International Journal of Production Research*, **39**, 1427-1467.

Güngör, A. and Gupta, S. M. (2001b) Disassembly sequence plan generation using a branch and bound algorithm. *International Journal of Production Research*, **39**, 481-509.

Güngör, A. and Gupta, S. M. (2002) Disassembly line in product recovery. *International Journal of Production Research*, **40**, 2569-2589.

Helgeson, W. B. and Birnie, D. P. (1961) Assembly line balancing using the ranked positional weight technique. *Journal of Industrial Engineering*, **12**, 394-398.

Homem de Mello, L. S. and Sanderson, A. C. (1990) And/or graph representation of assembly plans. *IEEE Transactions on Robotics and Automation*, **6**, 188-199.

Ketzenberg, M. E., Souza, G. C. and Guide, V. D. R. Jr. (2004) Mixed assembly and disassembly operations for remanufacturing. *Production & Operations Management*, **12**, 320-336.

LaCoursiere, C. (2005) Electronic Waste Recovery Business, Report No: E-128, Business Communications Company, Connecticut.

Lambert, A. J. D. (1999) Linear programming in disassembly/clustering sequence generation.

Computers & Industrial Engineering, **36**, 723-738.

Lambert, A. J. D. (2002) Determining optimum disassembly sequences in electronic equipment. *Computers & Industrial Engineering*, **43**, 553-575.

Lambert, A. J. D. (2003) Disassembly sequencing: A survey. *International Journal of Production Research*, **41**, 3721-3759.

Lambert, A. J. D. and Gupta, S. M. (2005) *Disassembly modeling for assembly, maintenance, reuse and recycling*, CRC Press, Florida.

Lee, D-H., Kang, J-G. and Xirouchakis, P. (2001) Disassembly planning and scheduling: review and further research, in *Proceedings of Institute of Mechanical Engineers Part B*, **215**, pp. 695-710.

Martin, G. E. (1994) Optimal design of production lines. *International Journal of Production Research*, **32**, 989-1000.

Mastor, A. A. (1970) An experimental investigation and comparative evaluation of production line balancing techniques. *Management Science*, **16**, 728-746.

McGovern, S. and Gupta, S. M. (2003) 2-opt heuristic for disassembly line balancing problem, in *Proceedings of the SPIE International Conference on Environmentally Conscious Manufacturing III*, Providence, Rhode Island, pp. 71-84.

Penev, K. D. and de Ron, A. J. (1994) Development of disassembly line for refrigerators. *Industrial Engineering*, **11**, 50-53.

Pinto, P. A., Dannenbring, D. G. and Khumawala, B. M. (1983) Assembly line balancing with processing alternatives: An application. *Management Science*, **29**, 817-830.

Ranky, R. J., Subramanyam, M., Caudill, R. J., Limaye, K. and Alli, N. (2003) Dynamic scheduling and line balancing methods, and software tools for lean and reconfigurable disassembly cells and lines, in *Proceedings of the IEEE International Symposium on Electronics and Environment*, pp. 234-239.

Realff, M. J., Ammons, J. C. and Newton, D. J. (2004) Robust reverse production system design for carpet recycling. *IIE Transactions*, **36**, 767-776.

Rosenblatt, M. J. and Carlson, R. C. (1985) Designing a production line to maximize profit. *IIE Transactions*, **17**, 117-122.

Scholl, A. (1999) *Balancing and sequencing of assembly lines*, Physica-Verlag, Heidelberg.

Spicer, A. J. and Johnson, M. R. (2004) Third-party demanufacturing as a solution for extended producer responsibility. *Journal of Cleaner Production*, **12**, 37-45.

Tang, Y., Zhou, M. C. and Caudill, R. J. (2001) A systematic approach to disassembly line design, in *Proceedings of the IEEE International Symposium on Electronics and Environment*, pp. 173-178.

Tonge, F. M. (1965) Assembly line balancing using probabilistic combinations of heuristics. *Management Science*, **11**, 727-735.

Wiendahl, H.-P., Lorenz, B. and Brückner, S. (1998) The necessity for flexible and modular structures in hybrid disassembly processes. *Production Engineering*, **5**, 71-76.