

# CHAPTER 1

---

## Introduction

This chapter provides an introduction to the book. Section 1.1 presents an overview of current disassembly issues. Section 1.2 describes the motivation of the text. The scope and objectives of the book are then given in Sec. 1.3 while the notations used throughout the text are detailed in Sec. 1.4. Finally, Sec. 1.5 presents an outline of the book.

---

### 1.1 Overview

Manufacturers are increasingly recycling and remanufacturing their postconsumer products due to new, more rigid environmental legislation, increased public awareness, and extended manufacturer responsibility. In addition, the economic attractiveness of reusing products, subassemblies, or parts instead of disposing of them has helped to further energize this effort. *Recycling* is a process performed to retrieve the material content of used and nonfunctioning products. *Remanufacturing* on the other hand, is an industrial process in which worn-out products are restored to like-new conditions. Thus, remanufacturing provides the quality standards of new products with used parts.

*Product recovery* seeks to obtain materials and parts from old or outdated products through recycling and remanufacturing in order to minimize the amount of waste sent to landfills. This includes the reuse of parts and products. There are many attributes of a product that enhance product recovery; examples include ease of disassembly, modularity, type and compatibility of materials used, material identification markings, and efficient cross-industrial reuse of common parts/materials. The first crucial step of product recovery is disassembly.

*Disassembly* is defined as the methodical extraction of valuable parts/subassemblies and materials from discarded products through a series of operations. After disassembly, reusable parts/subassemblies are cleaned, refurbished, tested, and directed to the part/subassembly inventory for use in remanufacturing operations. The recyclable materials can be sold to raw-material suppliers, while the residuals are sent to landfills. Disassembly is a process that interacts with all phases of product recovery including *before life* (the period of design and life cycle analysis), the *useful period* (the time when the

## 4 Disassembly Background

product is actually being manufactured or is in use), and *end-of-life* (the period in which a product completes its useful life and is ready for further processing, recovery, or disposal).

Disassembly has gained a great deal of attention in the literature due to its role in product recovery. A disassembly system faces many unique challenges; for example, it has significant inventory problems because of the disparity between the demands for certain parts or subassemblies and their yield from disassembly. The flow process is also different. As opposed to the normal “convergent” flow in regular assembly environment, in disassembly the flow process is “divergent” (a single product is broken down into many subassemblies and parts). There is also a high degree of uncertainty in the structure and the quality of the returned products. The condition of the products received is usually unknown and the reliability of the components is suspect. In addition, some parts of the product may cause pollution or may be hazardous. These parts may require special handling that can also influence the utilization of the disassembly workstations. For example, an automobile slated for disassembly contains a variety of parts that are dangerous to remove and/or present a hazard to the environment, such as the battery, airbags, fuel, and oil. Various demand sources may also lead to complications in disassembly-line balancing. The reusability of parts creates a demand for them; however, the demands and availability of the reusable parts are significantly less predictable than what is found in the assembly process. Most products contain parts that are installed (and must be removed) in different attitudes, from different areas of the main structure, or in different directions. Since any required directional change increases the setup time for the disassembly process, it is desirable to minimize the number of directional changes in the chosen disassembly sequence. Finally, disassembly-line balancing is critical in minimizing the use of valuable resources (such as time and money) invested in disassembly and maximizing the level of automation of the disassembly process and the quality of the parts (or materials) recovered.

This part of the book (Part I) provides a background to many of the issues and much of the current research, as well as an introduction to the problems and possible solutions. Included in this is a review of assembly lines, an introduction to disassembly lines, a survey of current research, graphical representations of products, an overview of computational complexity, and a description of combinatorial optimization searches.

In Part II of this book, the DISASSEMBLY LINE BALANCING PROBLEM (DLBP) is addressed using combinatorial optimization methodologies. While exhaustive search consistently provides the optimal solution, its exponential time complexity quickly reduces its practicality. Combinatorial optimization techniques are instrumental in obtaining optimal or near-optimal solutions to problems with

intractably large solution spaces. *Combinatorial optimization* is an emerging field that combines techniques from applied mathematics, operations research, and computer science to solve optimization problems over discrete structures. Some of these techniques include greedy algorithms, integer and linear programming, branch-and-bound, divide-and-conquer, dynamic programming, local optimization, simulated annealing, genetic algorithms, and approximation algorithms.

The seven techniques selected for application in this text seek to provide a *feasible* disassembly sequence (i.e., one in which no *precedence constraints* are violated since some tasks cannot be performed until their predecessor tasks have been completed), minimize the number of workstations, minimize the total idle time, and minimize the variation in idle times between workstations, while attempting to remove hazardous and high-demand product components as early as possible and remove parts with similar part removal directions together. Four data sets are used as case studies. These instances are used with all of the solution techniques to illustrate implementation of the methodologies, measure performance, and enable comparisons.

In Part III of this book, other problems related to the disassembly line are explored. This part visits the bulk of the remaining disassembly-related areas. A background—much of it from traditional assembly line and production theory—is provided and then disassembly-specific issues and research are detailed. These research areas include product planning, facility and line layout, sequencing and scheduling, inventory, just-in-time, revenue, and unbalanced lines.

---

## 1.2 Motivation

End-of-life processing for products is becoming increasingly desirable due to consumer preference, government regulation, and corporate financial interests. Obtaining components and materials that have some value while minimizing the amount of waste sent to landfills and reducing the amount of processed toxins introduced into the environment are all compelling reasons that disassembly is of such importance and has garnered so much global interest.

Disassembly has unique characteristics. While possessing similarities to assembly, it is not the reverse of the assembly process (Brennan et al., 1994); therefore, new and efficient approaches and methodologies are needed to effectively perform disassembly-line operations. The difficulty in obtaining efficient disassembly-line sequence solutions stems from the fact that a solution sequence consists of a permutation of numbers. This permutation would contain as many elements as there are parts in the product. As such, the observation can be made that the DISASSEMBLY LINE BALANCING PROBLEM would appear to be

NP-hard and the decision version would appear to be NP-complete. Also of interest, the DISASSEMBLY LINE BALANCING PROBLEM is a recent problem, first formally described in this century (Güngör and Gupta, 2002).

The importance of the disassembly line's role in end-of-life processing, the contemporary nature of the problem, and its NP-complete characteristics makes the DISASSEMBLY LINE BALANCING PROBLEM—and the disassembly line in general—interesting and relevant.

---

### 1.3 Scope and Objectives

This book introduces the disassembly line (Part I) and details its primary focus—disassembly-line balancing (Part II)—and then considers other disassembly-line problems that do not directly address balancing. Various techniques are explored that involve multiple objectives to address disassembly-line balancing. Since the decision version of the DISASSEMBLY LINE BALANCING PROBLEM is NP-complete, this book considers a rigorous combinatorial optimization treatment of the problem. As part of this, the problem is mathematically defined and proven to be NP-complete (as well as unary NP-complete and NP-hard), quantitative and qualitative evaluation criteria are developed, and four problem instances are introduced. Next, seven different techniques from the realm of combinatorial optimization are employed to solve the four instances. The seven methodologies are then compared to each other.

The first methodology used is exhaustive search, which here employs a depth-first search using a recursive backtracking procedure to visit all permutations of an instance's  $n$  parts to consistently obtain the optimal solution sequence. The exhaustive search algorithm is presented for obtaining the optimal solution to small instances of the DLBP. While always optimal, it is limited in the size of the instance it can solve since the time to solve an instance grows exponentially with the size of the instance. The second technique used is a genetic algorithm (GA), a metaheuristic that recombines and mutates the best solutions over many generations. The genetic algorithm considered here involves either a randomly generated or a hot-started initial population with crossover, mutation, and fitness competition performed over many generations. The third technique used is ant colony optimization (ACO), another metaheuristic. The ant colony optimization metaheuristic applied here is an ant system algorithm known as the ant-cycle model that is enhanced for the DLBP. Ant colony optimization uses software agents referred to as ants that grow a solution from first to last part under greedy decision-making rules. Successful ants add pheromone (in proportion to the quality of their solution) to their paths, all paths slowly evaporate, and the process repeats for a given number of cycles. The fourth technique used

is a deterministic first-fit-decreasing greedy algorithm, similar to that developed for the BIN-PACKING problem. Two 2-phase hybrids are considered next. The first hybrid process consists of the greedy sorting algorithm followed by a hill-climbing local search. The problem-specific hill-climbing algorithm only considers swapping parts that are in adjacent workstations. A second deterministic hybrid process is then shown. It consists of the same greedy sorting algorithm followed by a 2-optimal (2-opt) local search. The 2-opt search is modified for the DLBP by exchanging parts instead of the arcs connecting them. The final technique used is an uninformed, modified British Museum-type search that moves through the search space similarly to exhaustive search but only samples the space, methodically visiting equally spaced solutions in a deterministic manner. Influenced by the hunter-killer search tactics of military helicopters, this general-purpose heuristic algorithm easily lends itself to the DLBP. All of the techniques deliver optimal or near-optimal solutions while preserving the precedence relationships among the components.

Next (Part III), other wide-ranging disassembly-line problems are introduced. These problems encompass the areas of product design, facility location and line layout, sequencing and scheduling, removed-part inventory, just-in-time, revenue, and intentionally or unintentionally unbalanced lines.

---

## 1.4 Format and Notation

The following general guidelines are used throughout the book. Using the format of Garey and Johnson (1979), names of problems, including NP-complete problems, are capitalized when referring to the problem but not when referring to a methodology (e.g., “the LINEAR PROGRAMMING problem” is a different use than “modeling a problem using linear programming”) or when the formal name of the problem is not appropriate. Since the search methodologies in this book are used for problems other than the DLBP, the versions used in this book make use of a title case format and are often prefaced by “DLBP,” while generic references to the methodologies are made in lowercase (e.g., “DLBP Exhaustive Search” and “Exhaustive Search” refer to the problem-specific methodologies developed for use in this book while “exhaustive search” refers to any type or implementation of exhaustive search). The first time a proper mathematical or scientific term is defined (or if it is not defined, the first time it is used in the chapter that primarily references it, or lacking this, the first time it is used in the book) it is italicized. This is primarily the case when mathematical terms have a specific meaning but make use of a word common to the English language, an example being use of the word “language” in Chap. 6. Italics are also used to highlight some proper names. Most of the mathematical conventions are as found in Cormen

et al. (2001), Rosen (1999), or Garey and Johnson (1979). Finally, part removal times may also refer to *virtual parts* (also referred to as *virtual components*; Lambert and Gupta, 2005); that is, a task performed that is required (or desired or possible) and takes a finite amount of time but does not result in the immediate removal of any part. Therefore, the terms “task time” and “part removal time” are both used here with the understanding that the two are potentially distinct. Unless otherwise specified, work element, job, part, component, subassembly, and task may be used interchangeably throughout this text.

#### 1.4.1 General and Disassembly-Line Balancing Notation

The following notation is used in the remainder of the book:

|                                  |  |
|----------------------------------|--|
| $\langle 1, 2, \dots, n \rangle$ | ordered $n$ -tuple   |
| $\{1, 2, \dots, n\}$             | set (using the formal definition, i.e., list of $n$ distinct items)  |
| $(1, 2, \dots, n)$               | list of $n$ items  |
| $(p, q)$                         | arc (i.e., direct edge) $pq$   |
| $[p, q]$                         | edge $pq$  |
| A-B                              | subassembly made up of part A and part B   |
| $\leq_p$                         | polynomial time reduction or polynomial transformation; read as: “can be converted to,” “is easier than,” “is a subset of,” or “is a smaller problem than” |
| $\prec$                          | partial ordering, that is, $x$ precedes $y$ is written $x \prec y$   |
| $ X $                            | cardinality of the set $X$   |
| $\lceil x \rceil$                | ceiling function of $x$ ; assigns the smallest integer $\geq x$ , for example, $\lceil 1.3 \rceil = 2$   |
| $\max(x, y)$                     | maximum of $x$ and $y$   |
| $\min(x, y)$                     | minimum of $x$ and $y$   |
| $\forall$                        | “for all,” “for every”   |
| $\in$                            | “an element of,” “is in”   |
| $\exists$                        | “there exists,” “there exists at least one,” “for some”  |
| $\propto$                        | “in proportion to”   |
| $\cap$                           | intersection   |
| $\cup$                           | union  |
| $\subseteq$                      | subset   |
| $\wedge$                         | conjunction (logical AND)  |
| $\vee$                           | disjunction (logical OR)   |
| $!$                              | factorial  |
| :                                | “such that,” used primarily to avoid confusion with the vertical bars used to define cardinality   |

|                         |   |
|-------------------------|---|
|                         | “such that,” used primarily to avoid confusion with the pseudo-code equal sign “:=” also used as a triplet separator in scheduling theory and to represent absolute value in the efficacy index |
| →                       | “maps to”   |
| ↔                       | “if and only if”  |
| $\alpha$                | weight of existing pheromone (trail) in path selection; also refers to the machine environment in scheduling theory   |
| $\beta$                 | weight of the edges in path selection; also refers to the processing characteristics and constraints in scheduling theory   |
| $\varepsilon$           | empty string  |
| $\gamma$                | objective to be minimized in scheduling theory  |
| $\eta_{p,q}(t)$         | visibility value of edge (arc for the DLBP) $pq$ at time $t$  |
| $\rho$                  | variable such that $1 - \rho$ represents the pheromone evaporation rate   |
| $\tau_{p,q}(\text{NC})$ | amount of trail on edge $pq$ (arc for the DLBP) during cycle NC   |
| $\psi_k$                | $k$ th element’s skip measure (i.e., for the solution’s third element, visit every second possible task for $\psi_3 = 2$ )  |
| $\Delta\psi_k$          | $k$ th element’s delta skip measure; difference between problem size $n$ and skip size $\psi_k$ (i.e., for $\Delta\psi = 10$ and $n = 80$ , $\psi = 70$ )                                       |
| $O$                     | “big-oh,” $g(x)$ is $O(h(x))$ whenever $\exists y :  g(x)  \leq y \cdot  h(x)  \forall x \geq z$  |
| $\Pi$                   | product; also refers to a decision problem (consisting of a set $D$ of decision instances and a subset $Y \subseteq D$ of yes-instances) in complexity theory                                   |
| $\Theta$                | “big-theta,” $g(x)$ is $\Theta(h(x))$ whenever $g(x)$ is $O(h(x))$ and $g(x)$ is $\Omega(h(x))$   |
| $\Sigma$                | summation; also refers to an alphabet (a finite set of symbols, e.g., $\Sigma = \{0, 1\}$ ) in complexity theory  |
| $\Sigma^*$              | set containing all possible strings from $\Sigma$   |
| $\Omega$                | “big-omega,” $g(x)$ is $\Omega(h(x))$ whenever $\exists y :  g(x)  \geq y \cdot  h(x)  \forall x \geq z$  |
| $a$                     | MULTIPROCESSOR SCHEDULING problem task variable; also refers to a function variable in complexity theory  |
| $A$                     | MULTIPROCESSOR SCHEDULING problem task set  |
| $b$                     | function variable in complexity theory  |

## 10 Disassembly Background

|                    |   |
|--------------------|---|
| $B$                | MULTIPROCESSOR SCHEDULING problem deadline bound  |
| $BST_k$            | identification of $k$ th part in temporary best solution sequence during adjacent element hill-climbing (AEHC) and 2-Opt  |
| $c$                | initial amount of pheromone on all of the paths at time $t = 0$ ; also refers to a cost function that maps feasible points to the real numbers in complexity theory                 |
| $C_k$              | completion time of task $k$ in a flow shop in scheduling theory   |
| $C_{\max}$         | completion time of the last task to be completed in the flow shop in scheduling theory  |
| CT                 | cycle time; maximum time available at each workstation  |
| $d_k$              | demand; quantity of part $k$ requested  |
| $D$                | demand rating for a given solution sequence; also demand bound for the decision version of DLBP; also refers to the set of all instances of a decision problem in complexity theory |
| $D^*$              | optimal demand rating for a given instance; also used to refer to the set of solutions optimal in $D$   |
| $D_{\text{lower}}$ | lower demand bound for a given instance   |
| $D_{\text{upper}}$ | upper demand bound for a given instance   |
| DP                 | set of demanded parts   |
| $e$                | encoding function in complexity theory  |
| $E[x]$             | expected value of $x$   |
| $EI_x$             | efficacy index of measure $x$ ; generates values between 0 and 100 percent  |
| $f$                | a feasible solution in complexity theory  |
| $F$                | measure of balance for a given solution sequence; also refers to the finite set of feasible points in complexity theory   |
| $F^*$              | optimal measure of balance for a given instance; also used to refer to the set of solutions optimal in $F$  |
| $F_{\text{lower}}$ | lower measure of balance bound for a given instance   |
| $F_{\text{upper}}$ | upper measure of balance bound for a given instance   |
| $F_{t,r}$          | measure of balance of ant $r$ 's sequence at time $t$   |
| $Fm$               | flow shop with $m$ machines   |
| FS                 | feasible sequence binary value; FS = 1 if feasible, 0 otherwise   |
| $g(x)$             | function in complexity theory   |

|                             |  |
|-----------------------------|--|
| $h_k$                       | binary value; 1 if part $k$ is hazardous, else 0   |
| $h(x)$                      | function in complexity theory  |
| $H$                         | hazard rating for a solution; also hazard bound for the decision version of DLBP   |
| $H^*$                       | optimal hazard rating for a given instance; also used to refer to the set of solutions optimal in $H$                    |
| $H_{\text{lower}}$          | lower hazard bound for a given instance  |
| $H_{\text{upper}}$          | upper hazard bound for a given instance  |
| HP                          | set of hazardous parts   |
| $i$                         | counter variable   |
| $I$                         | total idle time for a given solution sequence; also refers to an instance of a problem in complexity theory              |
| $I^*$                       | optimum idle time for a given instance   |
| $I_j$                       | total idle time of workstation $j$   |
| $I_{\text{lower}}$          | lower idle time bound for a given instance   |
| $I_{\text{upper}}$          | upper idle time bound for a given instance   |
| $\text{ISS}_k$              | binary value; 1 if part $k$ is in the solution sequence, else 0  |
| $j$                         | workstation count (1, ..., NWS)  |
| $k$                         | counter variable (typically $k \in \{1, 2, \dots, n\}$ and identifies a part or refers to a sequence position)           |
| $l(a)$                      | MULTIPROCESSOR SCHEDULING problem task length  |
| $L$                         | Language in complexity theory, that is, any set of strings over $\Sigma$ (e.g., $L = \{1, 10, 100, 010, 1001, \dots\}$ ) |
| $L_r$                       | ACO delta-trail divisor value; set equal to $F_{n,r}$ for the DLBP   |
| $m$                         | number of processors in the MULTIPROCESSOR SCHEDULING problem; also number of ants; also number of machines              |
| $n$                         | number of parts for removal  |
| $N$                         | number of chromosomes (population)   |
| $\mathbb{N}$                | set of natural numbers, that is, $\{0, 1, 2, \dots\}$  |
| $\text{NC}_{\text{max}}$    | maximum number of cycles for ACO   |
| $\text{NPW}_j$              | number of parts in workstation $j$   |
| NWS                         | number of workstations required for a given solution sequence  |
| $\text{NWS}^*$              | optimal (minimum) number of workstations for $n$ parts   |
| $\text{NWS}_{\text{lower}}$ | lower bound on the minimum possible number of workstations for $n$ parts   |

## 12 Disassembly Background

|                      |   |
|----------------------|---|
| $NWS_{\text{upper}}$ | upper bound on the maximum possible number of workstations for $n$ parts  |
| $p$                  | counter variable; also edge/arc variable providing node/vertex identification   |
| $p(x)$               | polynomial function in complexity theory  |
| $P$                  | set of $n$ part removal tasks   |
| $Pr_{p,q}^r(t)$      | probability of ant $r$ taking an edge $[p, q]$ [arc $(p, q)$ for the DLBP] at time $t$ during cycle NC  |
| PRT                  | set of part removal times   |
| $PRT_k$              | part removal time required for part $k$ ( $PRT_k$ is onto mapped to PRT though not necessarily one-to-one since multiple parts may have equal part removal times) |
| $PS_k$               | identification of $k$ th part in a solution sequence, that is, for solution $\langle 3, 1, 2 \rangle$ , $PS_2 = 1$  |
| $PSG_k$              | identification of $k$ th part in the solution sequence after application of the Greedy algorithm  |
| $PSS_k$              | identification of $k$ th part in solution sequence after sorting  |
| $PST_k$              | identification of $k$ th part in solution sequence after AEHC and 2-Opt   |
| $q$                  | counter variable; also edge/arc variable (node/vertex identification)   |
| $Q$                  | amount of pheromone added if a path is selected   |
| $r$                  | set of unique part removal directions; also ant counter   |
| $r_k$                | integer value corresponding to part $k$ 's part removal direction   |
| $R$                  | direction rating for a given solution sequence; also direction bound for the decision version of DLBP   |
| $R^*$                | optimal direction rating for a given instance; also used to refer to the set of solutions optimal in $R$  |
| $R_k$                | binary value; 0 if part $k$ can be removed in the same direction as part $(k + 1)$ , else 1   |
| $R_{\text{lower}}$   | lower direction bound for a given instance  |
| $R_m$                | mutation rate   |
| $R_{\text{upper}}$   | upper direction bound for a given instance  |
| $R_x$                | crossover rate  |
| $\mathbf{R}^z$       | set of real numbers in $z$ dimensions   |
| $s$                  | MULTIPROCESSOR SCHEDULING task size   |
| $S$                  | solution structure in complexity theory   |
| $ST_j$               | station time; total processing time requirement in workstation $j$  |

|           |   |
|-----------|---|
| $t$       | time within a cycle for ACO; ranges from 0 to $n$                               |
| $T(n)$    | function describing an algorithm's running time on a computer processor         |
| $TMP_k$   | identification of $k$ th part in temporary sequence during AEHC and 2-Opt       |
| $V$       | maximum range for a workstation's idle time                                     |
| $x$       | general variable; also refers to a part removal direction                       |
| $X_{k,j}$ | binary decision variable; 1 if part $k$ is assigned to workstation $j$ , else 0 |
| $y$       | general variable; also refers to a part removal direction                       |
| $Y$       | set of all "yes" instances in complexity theory                                 |
| $z$       | general variable; also refers to a part removal direction                       |
| $Z$       | set of integers, that is, $\{\dots, -2, -1, 0, 1, 2, \dots\}$                   |
| $Z^+$     | set of positive integers, that is, $\{1, 2, \dots\}$                            |
| $Z_p$     | $p$ th objective to minimize or maximize  |

### 1.4.2 Assembly-Line Balancing Notation

Part I also makes use of the following notation:

|       |  |
|-------|--|
| DQ    | total daily quantity required          |
| DT    | available daily production time        |
| $U$   | utilization rate                       |
| $T_x$ | artificial task $x$                    |
| TT    | total time (the sum of the task times) |

### 1.4.3 Disassembly-to-Order Notation

In Part III, disassembly-to-order (DTO) notation includes the following:

|          |   |
|----------|---|
| $CEP_g$  | end-of-life product cost (acquisition, transportation, disassembly, and part cleaning, inspection, sorting, etc.) for DTO product $g$ |
| $CNP_k$  | new-part acquisition cost for DTO part $k$  |
| $CPD_k$  | disposal cost for DTO part $k$  |
| $g$      | DTO product identification  |
| $G$      | total number of DTO products  |
| $k$      | DTO part identification   |
| $na$     | total number of DTO parts for all products  |
| $n_g$    | number of DTO parts in end-of-life product $g$  |
| $Pr(sc)$ | probability of scenario $sc$ occurring  |
| SC       | set of DTO recourse model scenarios   |

## 14 Disassembly Background

|                 |   |
|-----------------|---|
| $w$             | number of possible outcomes in the DTO recourse model   |
| $X$             | decision variable   |
| $X1_g$          | decision variable indicating the amount of end-of-life product $g$ to acquire for disassembly                                       |
| $X2_k$          | decision variable indicating the amount of new part $k$ to acquire in order to meet the overall demand                              |
| $X2_{k, sc}$    | decision variable indicating the amount of new part $k$ to acquire in order to meet the overall demand in scenario $sc$             |
| $X3_k$          | decision variable indicating the number of excess removed parts $k$ that need to be disposed of                                     |
| $X3_{k, sc}$    | decision variable indicating the number of excess removed parts $k$ that need to be disposed of in scenario $sc$                    |
| $YP_{g, k}$     | yield of part $k$ from product $g$ (the amount of part $k$ obtained after disassembly of end-of-life product $g$ )                  |
| $YP_{g, k, sc}$ | yield of part $k$ from product $g$ (the amount of part $k$ obtained after disassembly of end-of-life product $g$ ) in scenario $sc$ |

### 1.4.4 Disassembly Just-in-Time Notation

In Part III, just-in-time (JIT) notation includes:

|              |  |
|--------------|--|
| $a$          | kanban container capacity for JIT  |
| $ARS_x$      | arrival rate of subassembly $x$ from an external source                                      |
| $C(n)$       | total number of possible combinations of a set of $n$ parts                                  |
| $D_e$        | JIT expected demand per unit time (e.g., per day)  |
| $DRP_x$      | demand rate for part $x$   |
| $FRP_x$      | furnish rate of part $x$   |
| $FRS_x$      | furnish rate of subassembly $x$  |
| $K$          | number of kanbans for JIT  |
| $L$          | JIT production lead time (equal to the production time plus waiting time plus movement time) |
| $LRP_{x, z}$ | removal rate of part $x$ at workstation $z$  |
| $LRS_y$      | removal rate of subassembly $y$  |
| $NPK_x$      | number of part kanbans for part $x$ at a given point on the disassembly line                 |
| $NSK_x$      | number of subassembly kanbans for part $x$ at a given point on the disassembly line          |
| $RRP_x$      | request rate of part $x$   |
| $RRS_x$      | request rate of subassembly $x$  |

|       |   |
|-------|---|
| $W$   | JIT buffer stock (equal to zero ideally; often 10 percent of $D_e L$ is used) |
| $x$   | part, part kanban, or workstation identification in the multikanban system    |
| $y$   | part, part kanban, or workstation identification in the multikanban system    |
| $y_z$ | subassembly or subassembly kanban identification in the multikanban system    |
| $z$   | part, part kanban, or workstation identification in the multikanban system    |

### 1.4.5 Disassembly Revenue Notation

Also in Part III, revenue notation includes the following (the artificial task notation  $T_x$  from Part I is not repeated here):

|               |  |
|---------------|--|
| $AP_{kt}$     | set of AND predecessors of task $kt$                                     |
| $AL$          | total number of artificial tasks   |
| $C_{kt}$      | cost of each disassembly task $kt$ undertaken                            |
| $CT_{upper}$  | decision-maker determined upper bound on the cycle time                  |
| $d_{kp}$      | part $kp$ 's demand  |
| $kp$          | identifies any of the $np$ parts   |
| $kt$          | identifies any of the $nt$ tasks   |
| $np$          | number of parts  |
| $nr_{kt, kp}$ | total count of each part $kp$ removed by task $kt$                       |
| $nt$          | number of tasks  |
| $OP_{kt}$     | set of OR predecessors of task $kt$                                      |
| $OS_{kt}$     | set of OR successors of task $kt$  |
| $P^-$         | set of parts having a negative revenue                                   |
| $P^+$         | set of parts having a positive revenue                                   |
| $q_{kp}$      | number of demanded part $kp$ that is removed                             |
| $REV_{kp}$    | revenue per unit demanded  |
| $S$           | cost per unit time of keeping one station open                           |
| $u_j$         | decision variable  |
| $x_{kt, j}$   | binary variable equal to one if task $kt$ is assigned to workstation $j$ |

### 1.4.6 Queueing Theory Notation

Finally, in Part III additional queueing notation includes:

|             |   |
|-------------|---|
| $\lambda$   | mean number of arrivals per time period |
| $\lambda_e$ | effective arrival rate                  |

|               |   |
|---------------|---|
| $\mu$         | service rate; the mean number of units that can be served per time period                                 |
| $\rho$        | traffic intensity; also known as the utilization factor   |
| $\sigma^2$    | variance  |
| $k$           | state of the queueing system; that is, number of customers in the waiting line plus the service facility  |
| $L$           | length of the queueing system (which is given by the expected number of customers in the queueing system) |
| $L_q$         | length of the queue itself  |
| $n$           | maximum number of customers in the system   |
| $\text{Pr}_k$ | probability of being in state $k$   |
| $s$           | number of servers   |
| $t$           | time  |
| $W$           | expected time in the queueing system  |
| $W_q$         | expected waiting time in line   |

---

## 1.5 Outline

Organized in three parts and 29 chapters, this book is primarily concerned with the complete disassembly of products on a *paced* disassembly line for component/material recovery purposes. It investigates the qualitative and quantitative aspects of the multicriteria solution sequences generated using the various combinatorial optimization techniques.

Chapter 2 provides a background in assembly lines, assembly-line balancing, and assembly-line modeling in order to provide a foundation prior to pursuing the disassembly line.

Chapter 3 gives an introduction to the disassembly line, detailing the many disassembly-specific considerations and providing some fundamental definitions.

Chapter 4 presents a survey of the research into environmentally conscious manufacturing and product recovery, assembly and manufacturing, disassembly, and various optimization and search techniques.

Chapter 5 provides some of the variety of product representations used in the disassembly field, including a graph-theory- and electrical-schematic-based arc and vertex representation for instances of the DLBP, which is then used throughout the book.

Chapter 6 gives an overview of complexity theory including the concepts of NP-completeness, unary NP-completeness, and NP-hardness, along with a listing of some of the specialized solution methodologies to address these problem classes. Also in this chapter, the computer processor hardware, software, language, software engineering, and analysis considerations are documented.

Part II begins with Chap. 7. This chapter presents the problem statement and research objectives of this book.

Chapter 8 provides a detailed description of the DISASSEMBLY LINE BALANCING PROBLEM including all objectives, mathematical formulae, and theoretical bounds.

Chapter 9 builds on the overview of complexity theory given in Chap. 6 and provides the proofs that the DISASSEMBLY LINE BALANCING PROBLEM is NP-complete, unary NP-complete, and NP-hard, necessitating specialized solution methodologies including those from the field of combinatorial optimization.

In Chap. 10, each of the combinatorial optimization searches used in Part II is introduced.

Chapter 11 introduces the four case-study problem instances. These include the personal computer instance, the 10-part instance, the cellular telephone instance, and the group of known-optimal instances. The personal computer and 10-part problem instances are slightly modified case studies from the literature, while the cellular telephone instance is a 25-part experimentally determined instance from an actual electronic consumer product. The known-optimal instances are a variable-dimension data set with known-optimal measures in all of the criteria under evaluation in this book. This chapter also includes a thorough statistical analysis of the number and percentage of optimal solutions with instance size using the known-optimal data set.

Chapter 12 introduces an analytical methodology for the qualitative and quantitative study of DLBP results. Also, the multicriteria decision-making format used in the remainder of the text is detailed. The concept of normalization is discussed and formulae for the efficacy indices are listed. Finally, simulation as an analysis tool is discussed.

Chapters 13 through 19 present the combinatorial optimization methodologies of exhaustive search, the genetic algorithm, ant colony optimization, the greedy algorithm, the greedy/hill-climbing hybrid, the greedy/2-opt hybrid, and the uninformed deterministic search heuristic, respectively. These chapters also include the numerical results of the methodologies' application to each of the four instances.

Chapter 20 compares all of the methodologies using the same qualitative and quantitative processes as used in Chaps. 13 through 19. These include graphical depictions of performance and calculated efficacy indices with growth in instance size, using the known-optimal instances.

Chapter 21 concludes Part II with a discussion of disassembly-line balancing extensions, additional issues, solution methodologies, probabilistic considerations, and areas for future research.

Chapter 22 introduces Part III where disassembly-line problems that are not directly related to balancing are reviewed.

Chapter 23 demonstrates application of disassembly considerations to the planning phase of a product, including the use of the

Chap. 9 mathematical formulae for use as indices in measuring the efficiency of future disassembly on multiple, competing product designs.

Chapter 24 provides a detailed background into facility design considerations and the field's associated definitions, then reviews traditional location and layout problems, and finally reuses the Chap. 9 formulae metrics for comparing different line designs.

In Chap. 25, classical sequencing and scheduling as used in production analysis is reviewed and applied using disassembly data.

Inventory theory is discussed in Chap. 26, and deterministic and stochastic disassembly-to-order systems are described.

Chapter 27 extends the disassembly-related inventory theory to encompass just-in-time.

In Chap. 28, a model is provided in which the revenue generated from disassembly is the primary consideration.

Chapter 29 provides a short summary of queueing theory as part of a disassembly application of the assembly-line concept of unbalancing lines.

Finally, the Appendix provides the acronyms that are referred to throughout the book.