Spring 5-31-2001

Modeling, design and scheduling of computer integrated manufacturing and demanufacturing systems

Ying Tang
New Jersey Institute of Technology

ABSTRACT

MODELING, DESIGN, AND SCHEDULING OF COMPUTER INTEGRATED MANUFACTURING AND DEMANUFACTURING SYSTEMS

by
Ying Tang

This doctoral dissertation work aims to provide a discrete-event system-based methodology for design, implementation, and operation of flexible and agile manufacturing and demanufacturing systems. After a review of the current academic and industrial activities in these fields, a Virtual Production Lines (VPLs) design methodology is proposed to facilitate a Manufacturing Execution System integrated with a shop floor system. A case study on a back-end semiconductor line is performed to demonstrate that the proposed methodology is effective to increase system throughput and decrease tardiness. An adaptive algorithm is proposed to deal with the machine failure and maintenance. To minimize the environmental impacts caused by end-of-life or faulty products, this research addresses the fundamental design and implementation issues of an integrated flexible demanufacturing system (IFDS). In virtue of the success of the VPL design and differences between disassembly and assembly, a systematic approach is developed for disassembly line design. This thesis presents a novel disassembly planning and demanufacturing scheduling method for such a system. Case studies on the disassembly of personal computers are performed illustrating how the proposed approaches work.

MODELING, DESIGN, AND SCHEDULING OF COMPUTER INTEGRATED MANUFACTURING AND DEMANUFACTURING SYSTEMS

by
Ying Tang

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
In Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Electrical Engineering

Department of Electrical and Computer Engineering

May 2001

APPROVAL PAGE

MODELING, DESIGN, AND SCHEDULING OF COMPUTER INTEGRATED MANUFACTURING AND DEMANUFACTURING SYSTEMS

Ying Tang

Dr. MengChu Zhou, Dissertation Advisor — Date
Professor of Electrical and Computer Engineering, NJIT

Dr. Reggie J. Caudill, Dissertation Co-Advisor — Date
Professor of Industrial and Manufacturing Engineering, NJIT

Dr. John D. Carpinelli, Committee Member — Date
Associate Professor of ECE & CIS, NJIT

Dr. Edwin Hou, Committee Member — Date
Associate Professor of Electrical and Computer Engineering, NJIT

Dr. Robin Qiu, Committee Member — Date
Sr. Computer Scientist, GHR Systems, PA

BIOGRAPHICAL SKETCH

Author: Ying Tang

Degree: Doctor of Philosophy

Date: May 2001

Undergraduate and Graduate Education:

- Doctor of Philosophy in Electrical Engineering
  New Jersey Institute of Technology, Newark, NJ, 2001

- Master of Engineering in Electrical Engineering
  Northeastern University, Shengyang, P. R. China, 1998

- Bachelor of Engineering in Electrical Engineering
  Northeastern University, Shengyang, P. R. China, 1996

Major: Electrical Engineering

Presentations and Publications:

Tang, Y., Zhou, M. C. and Caudill, R., "An Integrated Approach to Disassembly Planning and Demanufacturing Operation," *IEEE Transaction on Robotics & Automation* (accepted), April 2001.

Tang, Y., Zhou, M. C., Zussman, E. and Caudill, R., "Disassembly modeling, planning, and application", submitted to *Journal of Manufacturing Systems* (in review), revised version, Jan., 2001.

Tang, Y., Zhou, M. C and R. Qiu, "Virtual Production Line Design for Back-end Semiconductor Manufacturing System," submitted to *IEEE Transaction on Semiconductor Manufacturing* (in review), Jan. 2001

Presentations and Publications:

Tang, Y., Zhou, M. C., Zussman, E. and Caudill, R., "Disassembly modeling, planning, and application: a review," *Proceedings of IEEE International Conference on Robotics & Automation*, San Francisco, CA, April 22-28, 2000, pp. 2197-2202.

Tang, Y., Zhou, M. C. and Caudill, R., "An Integrated Approach to Disassembly and Demanufacturing Operation," *Proceedings of IEEE International Conference on Electronics & the Environment*, San Francisco, CA, May 8-10, 2000, pp. 354-359.

Tang, Y., Zhou, M. C. and Qiu, R., "Design of Virtual Production Lines in Back-end Semiconductor Manufacturing Systems," *Proceedings of IEEE International Conference on System, Man & Cybernetics*, Nashville, TN, Oct. 8-11, 2000, pp. 1733-1738 (The Best Student Paper Award).

Tang, Y. and Zhou, M. C., "Design of Reconfigurable Semiconductor Manufacturing Systems with Maintenance and Failure," to appear in *Proceedings of IEEE International Conference on Robotics & Automation*, Seoul, Korea, May 21-26, 2001.

Tang, Y., Zhou, M. C. and Caudill, R., "A systematic approach to disassembly line design," to appear in *Proceedings of IEEE International Conference on Electronics & the Environment,* Denver, CO, May 7-9, 2001.

Caudill, R., Zhou, M. C., Hu, J. J., Tang, Y., and Limaye, K., "Demanufacturing System Simulation and Modeling," in *Lifecycle Engineering: A Handbook for Mechanical Engineers*, Chapter 17, pp. 405-428, 2001.

To my beloved family and Gou Gou

ACKNOWLEDGEMENT

I would like to express my genuine appreciation to my advisor, Professor MengChu Zhou, and co-advisor, Professor Reggie J. Caudill, for their guidance, support, and friendship through this research.

Special thanks are due to Professor John D. Carpinelli, Professor Edwin Hou, and Dr. Robin Qiu to serve as my doctoral committee.

I would also acknowledge the support of the New Jersey Commission on Science and Technology and such industrial firms as AT&T, IBM, Lucent Technology and Panasonic etc. through the Multi-lifecycle Engineering Research Center (MERC) in NJIT.

This is a perfect opportunity to thank all of these fellows in the Discrete Event System Laboratory and the Multi-lifecycle Engineering Research Center for their assistance, collaboration and friendship in the past three years. Among them are Jianlin Gao, Congzhe Zhang, Meimei Gao, Lori E. Nanton, Lorietta Barnes, and Yanchun Luo.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF ABBREVIATIONS

DFX:	design for X
DL:	disassembly line
DP:	disassembly path
DP_d:	disassembly precedence matrices
DPN:	disassembly Petri nets
DPP:	disassembly process planning
DS:	disassembly sequence
EOL:	end-of-life
E-R	entity-relationship
IFDS:	Integrated flexible demanufacturing system
LMDB:	lot manager database
MES:	manufacturing execution systems
MFC:	microsoft fundation class
VC:	visual C++
PN:	Petri nets
PPN:	product Petri nets
SPN:	scheduling petri nets
SQL:	structured query language
VPL:	virtual production line
WP:	wave propagation
WPN:	workstation Petri nets

LIST OF SYMBOLS

$\hat{\theta}$: The point estimate of output variable Y based on n observations

$\hat{V}(\hat{\theta})$: standard error of the point estimate $\hat{\theta}$

$\Lambda(p)$: delay time associated with resource place p in *SPN,* where p represents a product, assembly or part

$\pi(p)$: EOL value function assigned to place p in *PPN*

$\lambda(t)$: a conservative time function associated with transition t in *PPN,* where t represents a disassembly operation

$\rho(t)$: probability value assigned to transition t in *DPN,* meaning the success rate of the operation represented by t

$\delta(t)$: a decision value associated with transition t in *DPN/PPN*

$\Gamma(t)$: processing time associated with transition t in *SPN*

$\sigma(w)$: the cost value for the time delay caused by workstation w

ς: a firing vector of the transition

π^B_{ijw}: the probabilities of the busy condition of the ith machine in the jth workstation of the wth VPL

π^F_{ijw}: the probabilities of the failure condition of the ith machine in the jth workstation of the wth VPL

π^I_{ijw}: the probabilities of the idle condition of the ith machine in the jth workstation of the wth VPL

ξ_i: the priority of the ith input inventory

ξ_{ik}: the priority of the ith machine in the kth class of machine pool (M_k)

LIST OF SYMBOLS
(Continued)

η_i: the total disassembly time for the ith job

τ_{ijw}: the ith machine/worker's processing time in the jth workcell of the wth VPL/DL

ι_{ijw}: the maintenance rate of the ith machine in the jth workstation of the wth VPL

α_{ijw}: the failure rate of the ith machine in the jth workcell of the wth VPL

β_{ijw}: the percentage time of the ith machine/worker dedicated to the jth workstation of the wth VPL

υ_{ijw}: the speed of the ith machine in the jth workstation of the wth VPL

γ_{ik}: the execution time of the disassembly process for the ith job in the kth workstation

ϑ_j: Number of machines assigned to the jth workstation

ε_{jw}: number of machines and workers in the jth workstation of the wth VPL/DL

ϖ_{jw}: maintenance periodicity of machines in the jth workstation of the wth VPL

ϕ_k: number of idle machines/workers of the kth class

$\psi_w(t)$: earliness of the wth VPL/DL at time t

$\zeta_w(t)$: slack time of the wth VPL/DL at time t

$\varphi_w(t)$: tardiness of the wth VPL/DL at time t

$\overline{a}_w$: the maximal real-time speed of workstations in the wth VPL/DL

$\underline{a}_w$: the minimal real-time speed of workstations in the wth VPL/DL

$|Y|$: the cardinality of set Y

LIST OF SYMBOLS
(Continued)

A: incidence matrix

a_{jw}: average speed of the jth workstation in the wth VPL/DL

a_w: the desired speed of the wth VPL/DL

B: exchange rate between cost and time delay

c: a cost vector corresponding to all disassembly tasks or transitions

C_i: set of final parts after the disassembly of each product in the ith input inventory

c_{ijw}: the ith machine/worker's capacity in the jth workstation of the wth VPL/DL

c_k: the cost of the disassembly activity k or its corresponding transition

$END(w)$: completion time of the wth work-order/demand

$d(p)$: disassembly value of a place p in PPN

$d(t)$: disassembly value of a transition t in PPN

D: set of VPLs/DLs with tardiness

$Date(t)$: check time of the wth work-order/demand

D_{ijw}: the transition rate matrix for the ith machine in the jth workstation of the wth VPL

d_{ik}: the mean failure rate of the ith machine in the kth class

$Due(w)$: due time of the wth work-order/demand

D_w: set of parts in the wth demand

E: set of VPLs/DLs with earliness except that it means a node-adjacency matrix in Chapter 2

$EST(w)$: earliest start time of the wth work-order/demand

$f(DP)$: the value gain in a DP

LIST OF SYMBOLS
(Continued)

f: machine utilization

F: a fastener matrix

F_i: the value gain in the disassembly path for the products in the ith input inventory

$g_{\alpha/2,(n-1)}$ the 100(1-α)% percentage point of a t-distributed with (n-1) degrees of freedom

g: system throughput

G: directed/undirected graph

$h(t)$: cost value function associated with transition t in *PPN*

H: the set of workstations a product goes through

I: input function that defines the set of directed arcs from places to transitions in PN

I_k: the kth class of idle machine pools

J: a set of jobs

k: machine/worker class index

K_{jw}: the allowed maximum failure rate of machines in the jth workstation of the wth VPL before its removal from the system for repair

M: marking vector in *PN*

m_{ik}: the mass of input

M_k: the kth class of machine pools

m_{ok}: the mass of output

m_w: the number of types of parts in the wth DL

N_{jw}: total number of the jth type of parts in the wth DL

LIST OF SYMBOLS
(Continued)

$n_w(t)$: number of finished magazines in VPL or product in DL in the wth work-order/demand at time t

N_w: total number of magazines in VPL or product in DL in the wth work-order/demand

O: output function that defines the set of directed arcs from places to transitions in PN

O_i: the number of parts in the ith output inventory

O_{ijkz}: the jth task of the ith job being performed by the kth machine in the zth workstation

P: a finite set of places in *PN*

p_{ik}: specific price of input

p_{ok}: specific price of output

Q: A set of places standing for product, subassemblies or components in *PPN*

q: a place in *PPN*

$r(w)$: time delay function associated with a workstation-available place w

R: set of resource places in *SPN*

r_{ijw}: the repair rate of the ith machine in the jth workstation of the wth VPL

R_k: the kth class of repair machine pools

r_k: the revenue of activity k

S: set of status places in *SPN*

$ST(w)$: actual start time of the wth work-order/demand

LIST OF SYMBOLS
(Continued)

t:	a transition in *PN*
T:	a finite set of transitions in *PN*
T^b:	the set of the transitions whose firings begin the operations in a *WPN*
T^e:	the set of the transitions whose firings end operations in a *WPN*
T_w:	production time for the wth work-order/demand
u:	the number of VPLs/DLs in the system
U:	set of edges
v:	bottlenck workstation index
V:	set of verticies
V^2:	sample variance of output variable Y based on n obseavations
w	VPL/DL index
W:	available machine pools in *PPN*
W^a:	available machine pools in *WPN*
W^b:	busy machine pools in *WPN*
x:	VPL index
x_{ijw}	the maintenance (repair) time for the ith machine in the jth workstation of the wth VPL
X_w:	number of workstations in the wth VPL/DL
Y	output variable
y:	Workstation index
Z:	input inventory set for products received VPL without getting immediate repair

CHAPTER 1

INTRODUCTION

1.1 Background

1.1.1 Automation for Manufacturing

Manufacturing is a vital source of wealth in every industrialized nation [Rembold *et al.*, 1993]. Technically, it is the application of physical and chemical processes to alter the geometry, properties, and/or appearance of a given starting material to make parts or products [Groover, 1996]. From the economic point of view, manufacturing adds value to the material by changing its shape or properties or by combining it with other materials that may have been similarly altered.

Over the past decades, a widely diverse array of developments and inventions laid the groundwork for modern-day automation. Economic and social considerations also encourage the adoption of advanced industrial automation, where the design and production operations are integrated into a completely computerized manufacturing system [Ouellette *et al.*, 1983].

Several factors act together to make production automation a viable and attractive alternative to manual methods of manufacture [Groover, 1980]:

(1) Increased productivity: Automation of manufacturing operations holds the promise of increasing the productivity of labor. Without automation, the practical and realistic output is directly limited by the capabilities of the manufacturing personnel. High production rates are achieved with advanced technology.

(2) High cost of labor: Automation has frequently been used to replace humans in tasks that are monotonous, physically tiring, dangerous or harmful to health. As a result, higher investment in automated equipment has become economically justifiable to replace manual operations.

(3) Labor shortage: In many advanced nations there has been a general shortage of labor. Labor shortage also stimulates the development of automation as a substitute for labor.

(4) Safety: Automation can meet social needs by permitting machines to perform unpleasant jobs, where a human operator would be exposed to hazardous conditions or toxic fumes etc.

(5) High cost of energy: The high cost of energy in manufacturing results in the need for greater efficiency in using the energy. Automated manufacturing processes may be operated at the optimum conditions for energy conservation. This is especially meaningful considering human beings' requirements for a sustainable economy.

(6) High cost of raw materials: Automated manufacturing operations can reduce scrap losses and maximize efficient use of the feedstock and raw materials.

(7) Improved product quality: Automated operations not only produce parts at faster rates than their manual counterparts do, but also they produce parts that are consistently of a high standard of quality.

(8) Reduced manufacturing lead-time: Automation allows a manufacturer to reduce the time between customer order and product delivery.

(9) Reduction of in-process inventory: Holding large inventories of work-in-process represents a significant cost to a manufacturer because it ties up capital. Automation

makes it possible to reduce work-in-process to a minimum by reducing the time a workpiece spends in the factory.

With the invention of computers, it was quickly realized that computers had an enormous potential for becoming the focal point in future automation endeavors. Conventional automation was based primarily on sophisticated mechanical machinery controlled by cams and levers or electrical switching gear [Rembold *et al.*, 1985]. In general, the equipment was conceived to perform fixed manufacturing assignments, so the degree of automation that can be achieved with these tools was rather limited. The original automation endeavors were directed primarily toward the improvement of the machining capabilities of the manufacturing equipment, where computers play an ever-increasing role.

Within the last few years, competitive pressure has forced many companies to adapt a plant quickly to changing market conditions. This problem led to the concept of computer-integrated manufacturing (CIM). CIM comprises product design, production planning, production control, production processes, quality control, production equipment, and plant facilities [Rembold *et al.*, 1985]. A conceived CIM system can be built well through the following developments [Rembold *et al.*, 1993]:

- The development of universal models of a product, manufacturing system and manufacturing operations;
- The development of simulation systems for all manufacturing processes and configuration means to set up a desired manufacturing environment;
- The investigation of new management information systems to support the decision-making functions of management to control and operate a plant; and

- The exploitation of artificial intelligence (AI) methods for manufacturing.

1.1.2 Environmental Problems and Responses

During the industrial revolution, environmental issues were not well addressed when products were designed, manufactured and consumed. This gives rise to environmental problems, the first of which is a deteriorating natural resource base [Reijnders, 1995]. Today's information society requires thousands of different products that ultimately result in billions of tons of materials discarded, most of which end up in landfills [Gungor and Gupta, 1998]. Rapid technological advances, especially in the area of information technology, make more and more otherwise functioning-well products obsolete in an accelerated pace. Many types of pollution are increasing. In practice over-exploitation of renewable resources, like the massive burning of woods and soil erosion, contributes substantially to pollution, giving rise to, among other things, greenhouse warming and eutrophication of surface wasters [Reijnders, 1995].

As a consequence of both fast depletion of the raw materials and an increasing amount of different forms of waste, customers have become aware of their environment and the potential problems that can be created by neglecting it. Therefore, they have started to show more interest in "green products" and taken their environmentally responsible use and consumption of the products. This has become an incentive for manufacturers to design and market environmentally friendly products to gain advantage in the marketing platform against their competitors.

The manufacturers and consumers are also forced by government laws and legislation to pay more attention to the environmental issues. In many countries, the environmental protection laws, regulations, and tax implications are already in effect

[Crognale, 1996]. For example, a German legislation mandated that as of January 1, 1994, manufacturers and retailers must take back and salvage products at the end of their lives and must design new ones with recycling in mind [Owen, 1993]. The European community has passed laws prohibiting the disposal more than 15% of an automotive product by the year 2002 and this percentage drops to 5% in the year 2015 [Nasr, 1997]. Japan proposed take-back legislation for electronic products, which was enacted in 1998 and comes into force on April 1st 2001 [Furuhjelm *et al.*, 2000].

1.2 Motivation

As shown in Section 1.1, with the fast development of computers and information technologies, computer-integrated manufacturing was coined to provide a new concept and direction to grow automatic manufacturing [Harrington, 1973]. As industry progresses, more products are made available for consumers. At the same time, more and more diversified models are appearing to suit individual tastes. This environment places a premium on agile manufacturing methods, which allow new manufacturing systems to be designed and planned within shorter and shorter lead-times and have quick reaction to a new market situation. In addition, product lifecycles are becoming shorter as a result of rapid technological advancement and international competition [Nof *et al.*, 1997]. This accelerating trend requires the redesign and re-planning of manufacturing systems more frequently and within shorter lead-times. The flexibility of manufacturing systems can take a number of forms, including [O'Grady, 1986]:

(1) Volume flexibility: the ability to handle changes in the production volume of a part;

(2) Re-routing flexibility: the ability to have a number of routes through the system for each part in order to enable, for example, machine breakdowns to be dealt with; and

(3) Part flexibility: the ability to handle a wide variety of parts including the ability quickly to adopt the system to handle a new part.

Due to the mammoth needs of the semiconductor market, semiconductor-manufacturing systems have been given a special attention. Since it is a business of high investment, high technology, and fierce competition, there is a significant amount of risk involved in the industry [Liao *et al.*, 1996]. The current manufacturing systems in semiconductor industries are designed based on a clustering of all of the equipment on the shop floor into predefined physical production lines, where each production line is capable of fabricating a group of products. If the part-mix for a production line changes, the system has to accommodate the change by making relevant adjustments in the production line [Qiu and Wysk, 1999]. Moreover, there exist high uncertainties in operations due to machine failures and fluctuation of yield rates [Liao *et al.*, 1996]. Thus, agility of production lines is a fundamental requirement for semiconductor manufacturers to stay competitive, which is a part of our focus in this research.

During the industrial revolution, environmental issues were not well addressed when products were designed, manufactured and consumed. Over the last few years, industries, governments and academia have paid much more attention on environmentally conscious manufacturing and product recovery because of the escalating deterioration of the environment. Some of efforts focus on the design stage of products, and is referred to as Design for 'X' (DFX) where X stands for a design under consideration such as Manufacturability, Testability, Installability, Compliance, Reliability, and Disassembly.

[Gatenby and Foo, 1990]. Since the biggest damage to the environment often occurs when products complete their useful life, understanding and developing techniques for end-of-life (EOL) management of the products by means of product/material recovery are extremely crucial [Thierry *et al.*, 1995]. Such companies as IBM, Sony, Phillips, Panasonic, and Lucent Technologies have made their significant research and technology effort, and made EOL product recovery economically profitable and environmentally responsible. As an integral part of a product life cycle, the demanufacturing, a process to disassemble products and then reuse, remanufacture, reengineer, or dispose of them, has been introduced [Shyamsundar *et al.*, 1997]. Disassembly gains much attention in demanufacturing processes. It serves to extract hazardous substances out of the systems, to reutilize valuable raw materials and components in products, and to minimize the amount of waste that must be disposed of in special purpose landfills. Disassembly modeling, process planning and system design are all considered in our work.

1.3 Objectives

The goal of this research is to provide a discrete-event system-based methodology for design, implementation, and operation of flexible and agile manufacturing and demanfacturing systems. Specific objectives are to:

(1) Provide formal and mathematical models for products, manufacturing and demanufacturing processes problems;

(2) Provide systematic and efficient methodologies to solve these problems;

(3) Perform case studies for semiconductor manufacturing systems; and

(4) Perform case studies for demanufacturing of obsolete consumer electronic products.

1.4 Organization

This dissertation is organized as follows. Chapter 2 makes a literature review for the current research issues in automated semiconductor manufacturing and demanufacturing systems. Chapter 3 develops a virtual production line (VPL) design methodology for back-end semiconductor manufacturing systems. In order to deal with the uncertainties in operations due to machine maintenance and failures, an adaptive methodology for the reconfiguration of VPLs is addressed in Chapter 4.

To obtain the better tradeoff between the benefit of disassembly processes and the resource requirement, an integrated approach to disassembly planning and demanufacturing operation is presented in Chapter 5. Important concepts of the methodology are defined and a case study on the disassembly of a batch of obsolete personal computers is provided to demonstrate the significance of this method. Based on the concept of virtual production line design in manufacturing systems and the similarity between manufacturing and demanufacturing, a solution methodology for the disassembly line-balancing is proposed and demonstrated in Chapter 6. Finally, the conclusions and some future research directions are presented in Chapter 7.

CHAPTER 2

LITERATURE REVIEW

As mentioned in Chapter 1, due to the growing demands from the markets and customers, the basic ideas of automatic operation have merged with more and more new concepts and technologies for manufacturing and demanufacturing systems today. Significant research effort is being carried out in industries and academia addressing various problems in these areas. This chapter reviews the major issues in these fields.

2.1 Issues in Manufacturing Line Design

Increasing global competition has made many business leaders and policy makers turn their attention to such critical issues as productivity and quality [Zhou and Venkatesh, 1998a]. One of the frequently prescribed remedies for the problem of decreased productivity and declining quality is the automation of factories. More specifically, technologies such as computer integrated manufacturing, robotics, and flexible manufacturing systems are the focal points of much research and exploration.

In order to meet customers' preferences and changing market demands, how to improve the flexibility and agility of manufacturing systems becomes extremely crucial. This involves two important issues: line balancing and resource planning. Bartholdi and Eisenstein (1996) introduced the concept of "operation of bucket brigades" for manual lines, used for example in the automotive industry. Rekiek *et al.* (1998) proposed the balance for ordering to treat line balancing and order variants simultaneously. Ham

(1996) presented a model for determining the optimal routing that maintains workload balance for flexible flow lines. A recent survey of balancing and resource planning for assembly lines was presented in [Rekiek *et al.*, 1999]. The University of Michigan Engineering Research Center has been developing a science base for a new generation of manufacturing systems-Reconfigurable Manufacturing Systems. This system is the one designed at the outset for rapid change in its structure, as well as its hardware and software components, in order to accommodate rapid adjustment of production capacity and functionality needed in response to new market demands [Koren *et al.*, 1998]. The National Research Council (1998) identified Reconfigurable Manufacturing Systems as the number one priority technology in manufacturing for the Year 2020.

Although the history of the semiconductor industry is not long compared with other manufacturing industries, semiconductor manufacturing has become one of the most competitive fields [Kim *et al.*, 1998]. It requires extremely high capital investment. Moreover, in the risky environment of semiconductor manufacturing systems, it is quite complicated and difficult for the production planning and shop-floor control because of reentrant product flows, diverse types of equipment, complex production processes and unpredictable yield and equipment downtime [Uzsoy, *et al.*, 1992].

2.1.1 Production Planning and Scheduling

Wein (1998) has pointed out that scheduling has a significant impact on the performance of semiconductor wafer fabrication. Liao *et al.* (1996) presented the development of a daily scheduling tool for a research and development pilot line of semiconductor wafer fabrication using Lagrangian relaxation and network flow techniques. Golovin (1986) discussed issues in production planning and scheduling in the semiconductor industry and

pointed out the difficulty of selecting an appropriate objective function. A hierarchical approach such as that suggested by Bitran *et al.* (1981, 1982) was advocated. Their approach is to solve an aggregate production planning problem on a rolling horizon basis, implementing only the first period decisions that are then disaggregated into decisions for the shorter-term operational scheduling problems such as what particular products to run and what order to run them in.

A planning system developed by Siemens that has been implemented in a development wafer fab was described in [Hadavi, 1994]. Their system explicitly addresses the dynamic nature of the production environment by localizing rescheduling as much as possible and making minimal resource commitments to maintain maximum flexibility. Order due dates and workcenter capacities are both embedded into constraint sets that correspond to particular time windows. When a new order arrives, a feasibility analysis module examines the resources required by the order and determines whether or not it is possible to produce the order. Then, the order is passed to a detailed scheduling module that works down the hierarchy of time windows to schedule the order into specific quarters, months, days and shifts.

Leachman (1986) gave a corporate-level production planning model for the semiconductor industry. The manufacturing process is divided into the stages of fab, probe, assembly and test, linked by inventories. The model includes multiple facilities, and treats entire production processes in each plant as integral entities. Computerized routines create the input files of an aggregate planning model and generate the linear programming formulation. The solution to this linear program yields a production plan at the process level of detail, which is validated by management. If it is invalid, the input

data is revised and the process repeated until an acceptable plan is generated. Once an aggregate plan has been obtained, it is disaggregated by solving a number of linear programs to divide the volume of production planned for each product family over the individual products. The output from the model is a capacity-feasible weekly start schedule for the various facilities in a company.

Based on work initiated by Kimemia and Gershwin (1983), Bai *et al.* (1990) proposed a hierarchical production planning and scheduling system for a semiconductor wafer fab. The key issue of their methodology is classifying the events that take place in a wafer fab according to the frequency of their occurrence and whether or not they are controllable. At each level, decisions are made in a way that satisfies the capacity constraints that are appropriate to that level and that meets objectives determined at a higher level.

2.1.2 Shop-Floor Control

Due to their complexity, the most common approach to shop-floor control problems in practice has been the use of dispatching rules [Uzsoy *et al.*, 1994]. Kim *et al.* (1998) considered multiple products with different due dates and different process flows. New dispatching rules for the objective of minimizing the mean tardiness of the orders were developed. Glassey and Resende (1988) pointed out that due to the extensive use of computer-aided manufacturing systems dispatching and lot release decisions can be made based on global information. They also developed a dispatching rule to complement the Starvation Avoidance input regulation policy. To minimize the average queue in front of the bottleneck, the method uses dynamically updated estimates of lead times and queue sizes, and gives higher priority to lots that are expected to encounter a shorter queue at

their next visit there. Focusing on the effect of lot sizes on cycle time performance, an order release policy for a semiconductor packaging line was proposed [Chandra and Gupta, 1992]. They also suggested a pull type synchronization mechanism to link the assembly and burn-in operations. The effects of these procedures on system behavior were analyzed using a simulation model and managerial implications of the results were discussed.

Adams *et al.* (1988) proposed the shifting bottleneck procedure for a job-shop scheduling problem. The entire approach can be outlined as follows: (a) divide the facility into a number of workcenters that consist of individual machines, set of parallel identical machines, or batch processing machines; (b) use a disjunctive graph to represent the problem; (c) schedule each workcenter and rank the workcenters in order of criticality; (d) capture the interactions between the workcenters already scheduled and not yet scheduled; and (e) resequence workcenters that have already scheduled using the new information obtained in Step (d). If all workcenters are scheduled, stop. Else, go to Step (c). The approximation approach is efficient to solve the minimum makespan problem of job shop scheduling and capable of handling re-entrant product flows, sequence-dependent setup times and different types of machines.

Ikura and Gimple (1986) provided an algorithm to determine whether there is a schedule in which all jobs are completed by their due dates for the cases where jobs arriving later have later due dates and all jobs have identical processing times. Ahmadi *et al.* (1992) considered the problems of minimizing mean flow time and makespan in flowshops containing batch and unit-capacity machines, assuming that all jobs have identical processing times on the batch machine, and provided polynomial-time

algorithms as the solutions to these problems. Lee *et al.* (1992) considered tardiness problems and presented polynomial-time algorithms for minimizing the number of tardy jobs on a single batch processing machine under such assumptions as constant processing time, agreeable release times and due dates.

Based on optimal control policies for two-machine flowshops, Lou and Kager (1989) proposed a two-boundary control methodology in which the fab is modeled as a virtual flowshop and each processing step is represented by workstation. This method is found to be more robust to changes in demand and to maintain low working-in-process and production surplus.

2.1.3 Manufacturing Execution Systems (MES)

A salient point emerging from the above review is that the relationship between production planning and shop floor control has been almost ignored. However, production-planning decisions that set goals for the shop affect the effectiveness of shop floor control policies with each other [Uzsoy, *et al.*, 1994]. Manufacturing Execution Systems (MES) fill the gap between production planning and shop floor control and have drawn increasing attention from the manufacturing industry for its success in reducing cycle time, lowing inventory, and increasing on-time delivery. MES includes a rich set of products [Scott, 1996] for:

- Factory and process modeling;
- Specification management;
- Work in process tracking;
- Resource management;
- Quality monitoring and control;

- Plant management decision support;
- Distribution of manufacturing instruction and equipment recipes for operators and machines;
- Real-time dispatching;
- Shop floor data collection;
- Production reporting, performance analysis and query tools;
- Corporate-wide planning;
- Cost analysis; and
- Final capacity scheduling.

Use the distributed object-oriented technique, Cheng *et al.* (1998) proposed a systematic approach to develop a computer-integrated MES framework that is open, modularized, distributed, configurable, interoperable, collaborative, and maintainable. Each of components of the MES framework is developed by inheriting a proper design pattern that is considered as the basic designs for the architecture, framework messages, and interfaces of this component to interoperate and collaborate with the other components.

Kane (1996) presented an Advanced Manufacturing Systems, to reconstruct manufacturing process flows in Xilinx, Inc. Programmed logic encoded in the application as part of business rules, lot characteristic, and flow input requirements allows the system to dynamically associate lots to process flows while taking advantage of real time changes in process flow designs and the current accumulation of lots characteristics. The generic design of the higher level flows using subroutine calls to lower levels preserves the reusability of a large body of flow designs.

Baliga (1998) pointed out that the winning strategy for semiconductor manufacturing companies was to implement highly responsive and efficient production lines using advanced management, computer, and control technologies. A synthesis of the advantages of MES was addressed in [Scott, 1996].

2.2 Collection Issues

With the rapid development of the industrial society, environmental issues received more and more attention in the recent years. The collection of the used items and/or their packages is one of the major issues in a product recovery environment [Livingstone and Sparks, 1994; Rembert, 1997]. Collection decisions involve location selection of collection centers (where retired products are collected and stored prior to distribution to demanufacturing facility); layout design of collection centers (including material handling and storage); and transportation (designing the transportation networks to bring used products from many origins to a single collection center) [Gungor and Gupta, 1999a]. Flow of used products back into the product demanufacturing facility is known as reverse distribution [Kooi *et al.*, 1996]. The high degree of uncertainty inherent in the collection activities makes it more complicated. The life cycle of distribution items and their environmental, financial and operational effects are analyzed in [Flapper, 1996, 1995; Lambert and Splinter, 1996]. Some researchers considered returnable distribution items in their studies [Goh and Varaprasad, 1986; Kelle and Silver, 1989; Kroon and Vrijens, 1995].

2.3 Disassembly Process Modeling

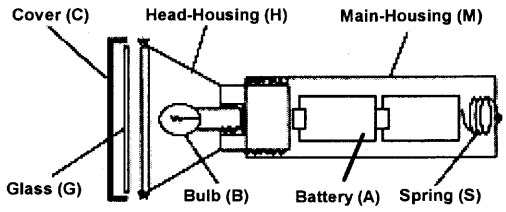
In a product disassembly process, choosing the representation of disassembly sequences is an important decision not only in creating a disassembly sequence planner but also in designing an intelligent control system for a disassembly system. The objective is to efficiently represent all feasible and complete disassembly sequences with correct precedence relations. Recently, a number of modeling strategies have been proposed. They can be categorized into four types, Component-Fastener Graph [Zhang, H. C. and Kuo, T. C., 1996, 1997], directed graph [Homem de Mello and Sanderson, 1990], AND/OR graph [Homem de Mello and Sanderson, 1990, 1991], and Disassembly Petri Net (DPN) [Cao and Sanderson, 1995, 1998; Moore *et al.*, 1998a, 1998b; Suzuki *et al.*, 1993; Zussman and Zhou, 1998-2000]. This work uses a handlight shown in Fig. 2.1 (a) as an example to illustrate these representations and concepts. To compare them, $|Y|$ is used to denote the cardinality of set Y, and n to the number of components in a product.

2.3.1 Connection Graph/Component-Fastener Graph

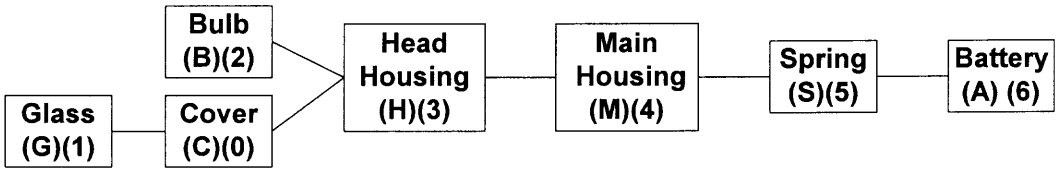
The component-fastener graph is an undirected graph, which can be generated according to the information from CAD packages. In such a graph $G=(V, U)$, vertices $V=\{v_1, v_2, \ldots v_n\}$ represent components; and edges $U=\{u_1, u_2, \ldots u_m\}$ denote a group of assembly relationships among components where m is the number of edges. It is clear that the upper bound for $|V|$ is n. A node-adjacency matrix E and a fastener matrix F are obtained as follows:

$E = (E_{ij})_{n\times n}$, where $E_{ij}=1$ if component i is connected with component j; otherwise 0.

$F = (F_{ij})_{n\times n}$, where $F_{ij}=k$ if component i is connected with component j by k fasteners; otherwise 0.



(a)



(b)

Fig. 2.1: (a) A handlight and (b) Its connection graph

With such a graph representation, the problem of identifying the optimal disassembly sequence is easily transformed into a graph search problem, which is addressed in Section II. The component-fastener graph for a handlight is essentially identical to the connection graph in Fig. 2.1(b). Its E and F matrices are shown below. Note that the node-adjacency matrix E is redundant.

$$E = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \qquad F = \begin{bmatrix} 0 & 2 & 0 & 2 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

2.3.2 Direct Graph

A directed graph can be used to represent the set of all disassembly sequences, and formally defined as G=(V, U). The nodes in V correspond to the possible states of the disassembly process; the edges in U corresponding to disassembly tasks are ordered pairs of nodes. The worst case to configure this graph happens when every part is connected to every other part. Thus, the size of this graph is [Homem de Mello and Sanderson, 1990]:

$$|V| = \text{partitions (n)} = \sum_{i=0}^{n-1} \text{partitions (n-1-i)} \binom{n-1}{i} \text{ where partitions (0) = 1}$$

Fig. 2.2 shows the directed graph of feasible disassembly sequences for the product shown in Fig. 2.1 and |V| = 20.

2.3.3 AND/OR Graph

The AND/OR graph of feasible disassembly sequences is also a directed graph, which is represented by G=(V, U), where each node in V can be a product, part or subassembly, and hyperarcs in U represent the set of feasible disassembly tasks. A node i representing a product or subassembly can have k ($k \geq 1$) disassembly methods, forming an OR-relation; if a method disassembles i into u ($u>2$) nodes (parts and/or subassemblies), u arcs link node i to those u nodes, forming an AND-relation. The maximum number of nodes in this graph is $|V| = 2^n-1$ when every component is connected with every other component. Figure 2.3 shows the AND/OR graph for a handlight. Figures. 2.2-2.3 give a simple proof that the AND/OR graph needs less storage space for nodes and edges than the directed graph. In fact, Fig. 2.2 has 20 nodes and 33 edges while Fig. 2.3 has 15 nodes and 20 edges. This advantage of AND/OR graphs becomes greater as the number of parts increases [Homem de Mello and Sanderson, 1990].

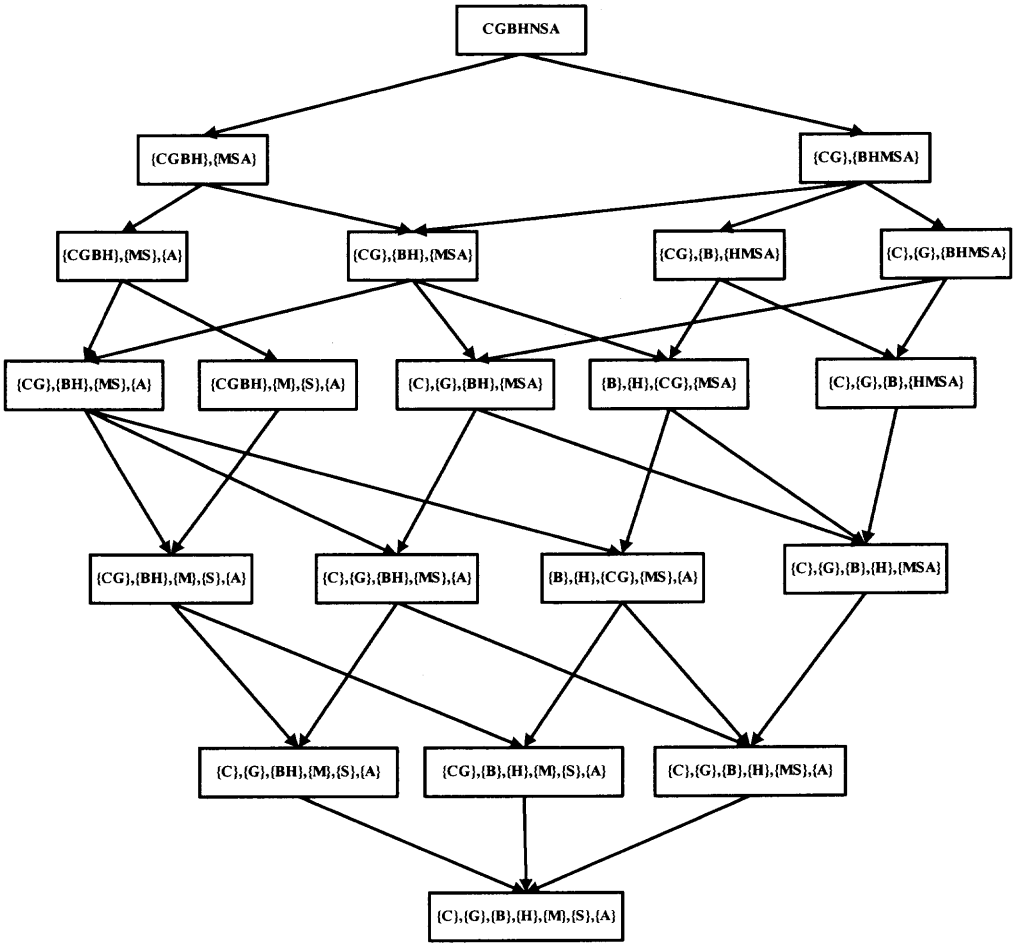


Fig. 2.2: Directed Graph of feasible disassembly sequences for a handlight

Based on the previous work [Homem de Mello and Sanderson, 1990, 1991], Cao and Sanderson proposed the AND/OR net, which extends the AND/OR graph representation to incorporate system mechanisms and devices [Cao and Sanderson, 1998].

2.3.4 Disassembly Petri Net

For the purpose of the analysis of the network, it is convenient to represent each disassembly process and system resources respectively. Bipartite multi-folded

Disassembly Petri Nets (DPN) have such advantages. The below definition is due to [Zussman and Zhou, 1998].

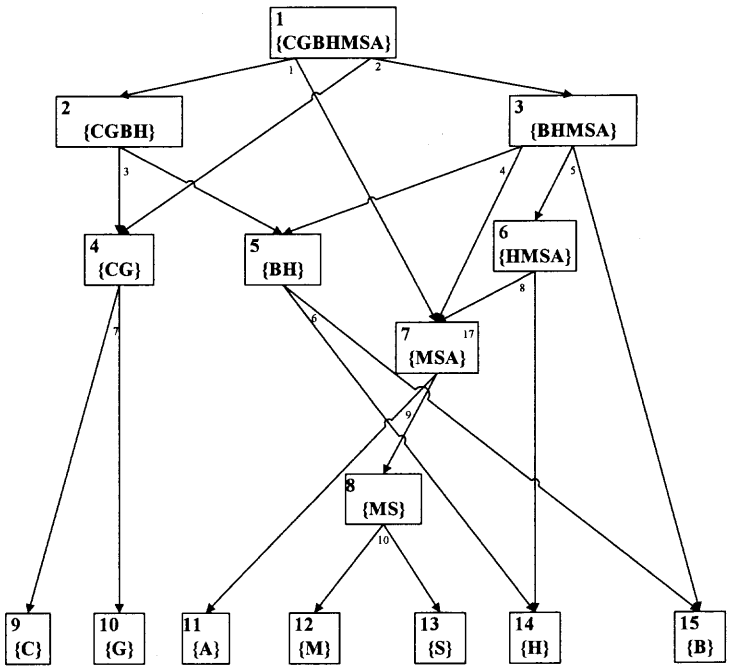


Fig. 2.3: The AND/OR graph for a handlight

DPN=(P, T, I, O, m_0, π, τ, δ, ρ) where:

P-places. There is a place called a product or root denoted by p_1 with no input arc, a set of places called subassembly, a set of places called leaves denoted by P', each of which has no output arcs.

T-transitions, each of which represents disassembly and has at most one input arc and at least two output arcs.

I: $P \times T \rightarrow \{0, 1\}$ is an input function that defines the set of directed arcs from P to T.

O: $P \times T \rightarrow \{0, 1\}$ is an input function representing directed arcs from T to P.

m_0: $P \rightarrow \{0, 1, 2, \ldots\}$is an initial marking, $m_0(p_1)=1$ and $m_0(p)=0, \forall p \in P-\{p_1\}$.

π-a value/cost function assigned to a place (end-of-life value or cost of the product, part or subassemblies a place represents).

h-a cost function assigned to a transition.

δ-a decision value associated with a transition.

ρ-[0,1] probability value associated with a transition representing the success rate of its corresponding disassembly operation.

Figure 2.4 gives an example DPN for a handlight. Compared to Fig. 2.3, it is clear that the DPN is a variant of an AND/OR graph. A token game and association of pre-firing and post-firing values with transitions facilitate adaptive dynamic disassembly planning.

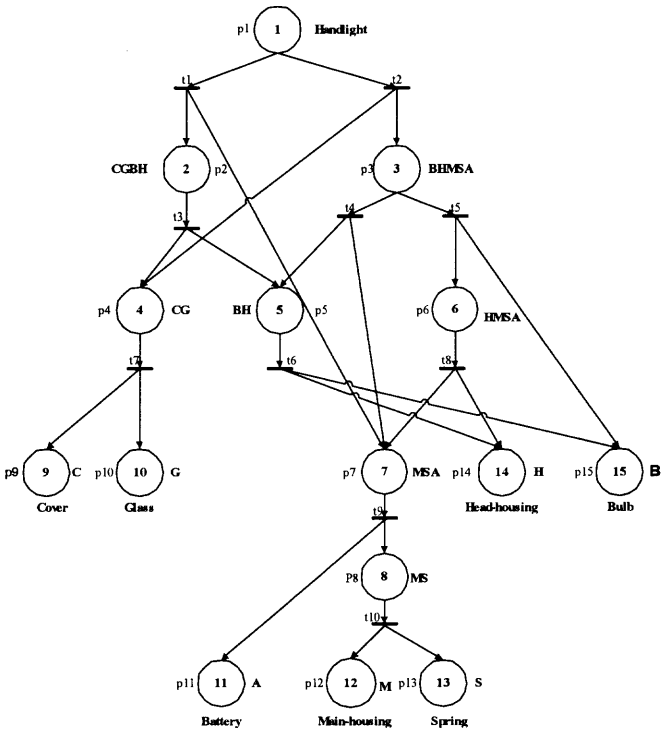


Fig. 2.4: Disassembly Petri Net for a handlight

Cao and Sanderson (1998) presented a method for mapping an AND/OR net to a Petri net. They also proposed Fuzzy Petri nets by introducing fuzzy members and

complex fuzzy reasoning functions into the ordinary Petri nets to facilitate the search for correct operation sequences [Cao and Sanderson, 1995].

Through analysis of these four methodologies, the comparison results for their advantages and disadvantages are concluded in Table 2.1. It is clear that each method has its special application, and a representation framework itself is not sufficient for automatic disassembly because an intelligent system must be able to supervise these plans and derive new and vital information for further disassembly plans and operations.

2.4 Disassembly Process Planning (DPP)

Disassembly process planning (DPP) finds a sequence of disassembly tasks that begins with a product to be disassembled and terminates with all of the parts of interest disconnected. Since a sequence can drastically affect the efficiency of a disassembly process, for instant, some sequences require less change of tools and disassembly time, and include simpler and more reliable operations than others, disassembly planning is an important research issue. In the earliest studies, some researchers treated the disassembly problem as an assembly one to derive the best strategies in review of disassembly as the logic reverse of a feasible assembly regardless of whether the reverse operation itself is feasible or not [Cao and Sanderson, 1995, 1998; Homem de Mello and Sanderson, 1991; Suzuki *et al.*, 1993]. Even though the similiar approach may be used for the disassembly planning, the changes of products during the utilization phase can result in great uncertainty about product structures and conditions, which sometimes disables such approaches valid for assembly planning making disassembly automation more difficult.

Table 2.1 The comparison of methodologies for modeling disassembly

Models	Advantages			Disadvantages		Ref.[+]
Component-fastener graph	1.Be retrieved directly from the CAD package. 2.Be illustrative to show the fastener information. 3.Use graph search algorithm to get optimal DS.			Be incapable of modeling disassembly tasks		[116] [117]
Directed graph	Contain both disassembly and assembly tasks where all disassembly tasks are reversible.	1. Represent the set of all disassembly sequences.		Require component and fastener information to construct the graph.	Require large space for nodes and edges	[49]
AND/OR graph		1.Encompass all possible disassembly sequences with a reduced number of nodes and edges. 2.Explicitly shows the possibility of simultaneous execution of disassembly tasks. 3.Use the heuristic algorithm to get optimal DS.			Difficult to integrate with resource modeling	[48] [49] [64][*] [75][*]
Disassembly PN-1 [Suzuki et al.]		1.Represent each subassembly and each task respectively. 2.It is easy to integrate with resources to form a comprehensive model for process planning and control. 3.Explicitly shows the possibility of simultaneous execution of disassembly tasks.	Use LP to derive the optimal plan.		No end-of-life value incorporated	[15] [16] [100]
Disassembly PN-2 [Zussman et al.]			Consider the end-of-life value and easy to implement adaptive planning.		Lacking LP formulation and solution	[122] [123] [124]

[*] The disassembly graph in [64] and the product tree in [75] are based on AND/OR graph
[+] In this column, the paper numbers correspond to the numbers in the REFERENCE List to save space.

Thus, there is a growing need to develop new methodologies to specifically address the disassembly characteristics. This includes two important decisions: (1) how far to disassemble a certain product; and (2) how to find an optimal or near-optimal DS.

2.4.1 Reverse Assembly Approach

Homem de Mello and Sanderson (1991) transformed the problem of generating assembly sequences into that of generating disassembly sequences and presented a correct and complete algorithm as a solution. The basic idea underlying the approach is to enumerate the decompositions of an assembly and to select those decompositions which are feasible and correspond to a hyperarc in the AND/OR graph connecting a node to two nodes. The decompositions are obtained by enumerating the cut-sets of the assembly's graph of connection.

Using the handlight example shown in Fig. 2.1, two algorithms *Get-Feasible-Decomposition* and *Generate-And-Or-Graph* are examined [Homem de Mello and Sanderson, 1991b]. First, *Get-Feasible-Decomposition* computes the graph of connections shown in Fig. 2.1(b) and all its cut-sets indicated in Fig. 2.5. The analysis of those cut-sets indicates the feasible decompositions. The first cut-set does not yield a feasible decomposition since it is impossible to extract the glass before the cover is disassembled. The second cut-set yields a feasible decomposition since it is feasible to separate the subassembly consisting of glass and cover, and the subassembly made up of the other five parts. Similarly, the fourth cut-set yields feasible decomposition while the others not. Then, with such information, the AND/OR graph of disassembly sequences can be obtained as shown in Fig. 2.3 by using algorithm *Generate-And-Or-Graph*, which

takes the relational model of an assembly, and returns AND/OR graph representation of all disassembly sequences for that assembly.

Homem de Mello and Sanderson (1991) also presented a heuristic-search algorithm to derive the optimal disassembly process plans by weighting hyperarcs in proportional to the difficulty of their corresponding operations. It is clear that each plan corresponds to a tree in the AND/OR graph. Thus to get the best disassembly plan is just to compute the total cost of each tree from a node, recursively, as

- Zero, if the node has no leaving hyperarcs; or
- The sum of the weights of the hyperarc leaving the node and the cost of the trees from the successor nodes.

Then, the less the cost of a tree, the better the plan corresponding to such a tree. For this evaluation function, the search for the best DS can be conducted using algorithms such as AO* [Nilsson, 1980]. The amount of computation involved in disassembly planning depends not only on the number of parts and how they are interconnected, but also on the structure of the AND/OR graph. At the worst case, such search algorithms have the exponential computational complexity in terms of n.

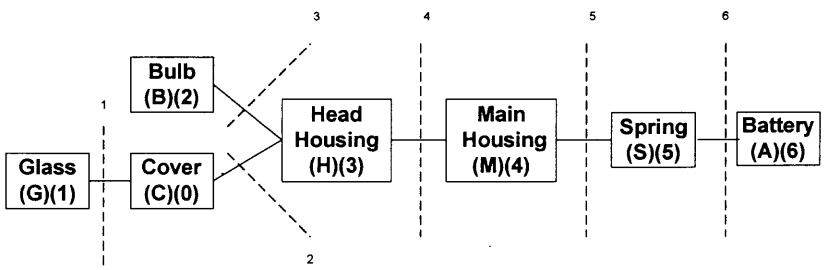


Fig. 2.5: The cut-sets of the graph of connections for the handlight

Suzuki *et al.* (1993) assumed assembly operations and disassembly operations are invertible each other. Then, they modeled the disassembly network using Petri net and

proved that Linear Programming technique can be used to find the optimal task sequences for DPN as follows:

Minimize: $c^T\varsigma$

subject to: $A\varsigma = M - M_0$

where c is a cost vector corresponding to each disassembly task or a transition, ς is a transition vector, A is the incidence matrix of a DPN, M is a final marking and M_0 is an initial marking. The (0, 1)-solution-constrain can be removed owing to the graph structural properties of DPN. This illustration made it possible for us to use more powerful algorithms (e.g., simplex algorithm and interior point algorithm) than AO^* to find an optimal DS. Note that interior or point algorithm runs at polynomial time with respect to the number of variables (i.e., the number of transitions in this case). Since the number of transitions is exponential to n, the algorithm still has the exponential complexity in terms of n.

2.4.2 Disassembly Leveling

While complete disassembly may provide the best way of minimizing the damage to the environment, some studies has concluded that with the current recycling techniques and market prices, complete disassembly is not always profitable since the cost of disassembly may be more than the market and environment benefit. Thus, disassembly-leveling problem becomes an important issue. It decides a disassembly level to which the product of interest is disassembled to keep profitability and environmental features of the process at a desired level. In the literature, qualitative and quantitative knowledge of the disassembly processes are combined with economic factors to obtain the good tradeoff

between the benefit of a disassembly process and the resource requirement [Gungor and Gupta, 1998].

Lambert (1997) discussed the determination of disassembly level and sequence for a product. A criterion for the optimum disassembly is the maximum revenue of the activities. The revenue r_k of activity k is defined as:

$$r_k = \sum_o (p_{ok} \cdot m_{ok}) - (p_{ik} \cdot m_{ik}) - c_k$$

where p_{ok} and p_{ik} represent the specific prices of output and input respectively; m_{ok} and m_{ik} are their respective mass, and c_k is the cost of the disassembly activity k. For this purpose, the author proposed an action-oriented graphical approach:

a) Model a disassembly graph for a product based on the AND/OR graph;

b) Construct a process graph by moving from the right (single component) to the left (assembly) in the disassembly graph; and

c) Calculate the revenue for each action using the rules based on the type of its corresponding node.

If an action has a negative or zero revenue, it will not be performed because of losses. Thus, the optimal disassembly sequence is a series of actions with the highest revenue. It is clear that this method is essentially similar to the heuristic-search algorithm illustrated before. The most complicated step in this approach is to model a disassembly graph. Since the size of the AND/OR graph is an exponential function of n, the computational complexity for this method is exponential.

To decide whether or not a product to be disassembled and how far to disassemble, Meacham *et al.* (1999) presented a product tree that similar to AND/OR graph and associated each node in this tree with a revenue and a marginal benefit. Although the

revenue definition is different from that of Lambert (1997), their basic ideas are similar, that is, to calculate the revenue and marginal benefit for each node and compare them to decide whether it is worthwhile to be disassembled. Based on these insights, an algorithm *MAXREV* was proposed. Even though the worst computational complexity of this algorithm is polynomial in term of the number of nodes in the product tree, the number of nodes in a tree grows exponentially with n.

Considering the variations of product conditions, which in turn will cause operational failure in disassembly and different end-of-life values embedded in a product, subassembly, and part, Zussman *et al.* (1995) proposed a method to model a disassembly system. The method provides predictive/reactive decision support using DPN linked to a Bayesian network. Zussman and Zhou (1999, 2000) also derived several planning algorithms and design, and implemented a DPN-based disassembly process planner and robotic system. The key point of such algorithms is to assign different priority to different disassembly alternatives, and the system selects a transition with the highest priority. Such priority is decided not only based on the revenue and cost data used in other methods (e.g., *MAXREV* and [Lambert, 1997]), but also disassembly operational failure rates and performance tolerance. The worst-case complexity for the optimal planning is exponential.

2.4.3 Optimal Disassembly Sequences

One of the objectives of DPP is to find an optimal or near-optimal DS that obtains the best cost/benefit ratio for disassembly. To fulfill this task, liaison analysis is of importance. Researchers in this field have incorporated various parameters influencing the complexity of the disassembly into geometrical, logical and dimensional measures. In

this section, a handlight example is used to illustrate some important methodologies in details.

A. Zhang and Guo's Method

Zhang and Kuo (1996, 1997) proposed a graph-based method to find the possible disassembly sequences from the CAD system directly. It starts with identifying cut-vertices (the connection component in an assembly), bi-connected graph (subassembly), pendent vertices (single component) in a component-fastener graph through a graph search algorithm *Depth First Search*. Take the handlight as an example. First, perform *Depth First Search* of the graph shown in Fig. 2.1. Choose any vertex $v \in V$ as the starting point (e.g., vertex 3), a new graph shown in Fig. 2.6 is obtained. Second, separate the cut vertices, bi-connected graph, pendent vertices and subgraphs into several graphs also shown in Fig. 2.6 by comparing the dfs[v] and low[v] of each vertex and following the rules shown in Table 2.2. Note that dfs[v] is the number assigned by the search algorithm; low[v] is the smallest dfs number of any vertex reachable from v using tree edges and one back edge. Then, analyze the disassembly precedence matrices DP_d retrieved from CAD packages, and reduce the structure of the feasible module that can be disassembled from the assembly by the following rule:

$$DP_d = \begin{bmatrix} D_{1,1} & D_{1,2} \cdots D_{1,n} \\ D_{2,1} & D_{2,2} \cdots D_{2,n} \\ \vdots & \vdots \quad \vdots \\ D_{n,1} & D_{n,2} \cdots D_{n,n} \end{bmatrix} \xrightarrow{ruleA} DP_d = \begin{bmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,k} \\ A_{2,1} & A_{2,2} & \cdots & A_{2,k} \\ \vdots & \vdots & & \vdots \\ A_{k,1} & A_{k,2} & \cdots & A_{k,k} \end{bmatrix}$$

where

d is one of directions ±x, ±y, and ±z;

$D_{u,v} = 1$ if component v needs to be removed before u, otherwise 0;

$k = p + q + r + s$ and $m_1 + m_2 + \ldots + m_p + q + r + s = n$ assuming there are p subassemblies, q cut vertices, r pendant vertices, and s bi-connected graph in the graph G_c, and there are m_i vertices in each subassembly;

$$A_{u,v} = \begin{cases} 1, & \text{if } u \neq v \text{ and } v \text{ needs to be removed before the u, where u amd v are either a single vertex or a subassembly} \\ 0, & \text{otherwise} \end{cases}$$

Rule A: $D_{u,v} = 0$ if $u, v \in S_i$, S_i represents a subassembly ($i = 1, 2, \ldots$, and p).

Rule B: The vertex/subassembly u can be removed from the assembly if

$$DP_u^d = \sum_{v=1}^{k} A_{u,v} = 0$$

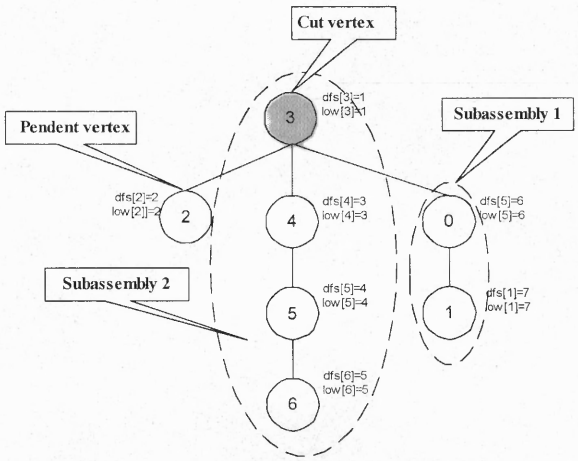


Fig. 2.6: Depth first tree with subassemblies' identification for the graph in Fig. 2.1

In this example, there is one cut vertex 3, one pendant vertex 2, and two subassemblies $SB_1=\{0,1\}$ and $SB_2=\{4,5,6\}$. When the total edge number is examined for each subassembly that connects with the cut vertex, the cut vertex is grouped with the one of subassemblies that has the largest number connected with it. In this example, cut vertex 3 is grouped with SB_2. The disassembly precedence matrix of the handlight just

has two directions, which is denoted as DP_{+x} and DP_{-x} as shown in Fig. 2.7. From these matrices, it is known that SB_1 can be first disassembled from -x direction. After that, the row and column vectors of SB_1 are set to 0. Thus vertex 2 can be removed from -x direction, and finally SB_2. Finally, the disassembly tree is obtained in Fig. 2.8 using object-oriented techniques, where the object vertex consists of the component information, the object edge has the fastener information, and the object fastener refers to the disassembly metdhod. Undoubtedly, the most complex step in this method is to construct depth first tree. At the worst case, this method has exponential computational complexity.

Table 2.2 Rules in the Component-Fastener graph

Rule 1	The root of a graph is a cut vertex if and only if it has more than one child.
Rule 2	A vertex v other than the root of a graph is a cut vertex if and only if v has a child x such that low[v] ≥dfs[v].
Rule 3	A vertex v is a pendant vertex if and only if its total edge number in E except the edge that is connected with the cut vertex is 0, assuming the cut vertex's row and column vectors are 0's in E.

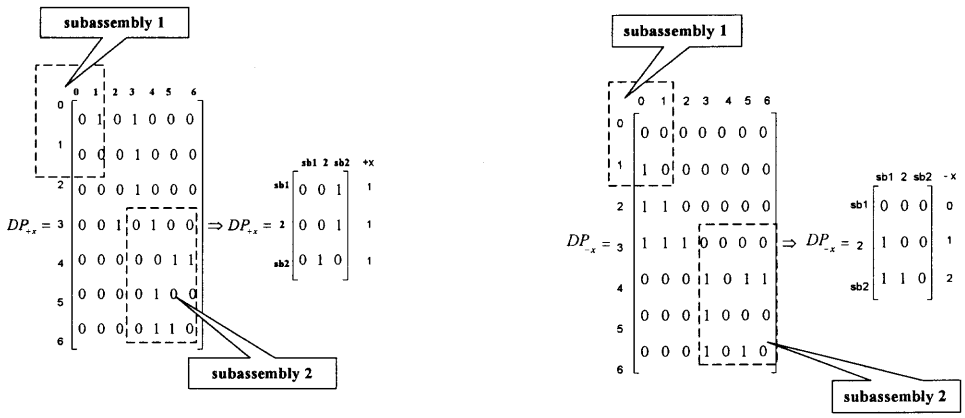


Fig. 2.7: Disassembly precedence matrix for the handlight

Based on the analysis of geometric precedence relationship (the same as DP_d illustrated before), Gungor and Gupta proposed a methodology to address a more practical problem, i.e., which component or subassembly should be disassembled first when more than one available after simplifying the precedence matrix [Gungor and Gupta, 1998b]. The key work in this method is to assign priority to each component or subassembly by considering such factors as the time of tool change, disassembly direction change, delaying the disassembly of a part with hazardous contents, and delaying the disassembly of a part with high market value. The component or subassembly is disassembled from the high to low priority and the precedence matrix is updated until the whole product is recovered.

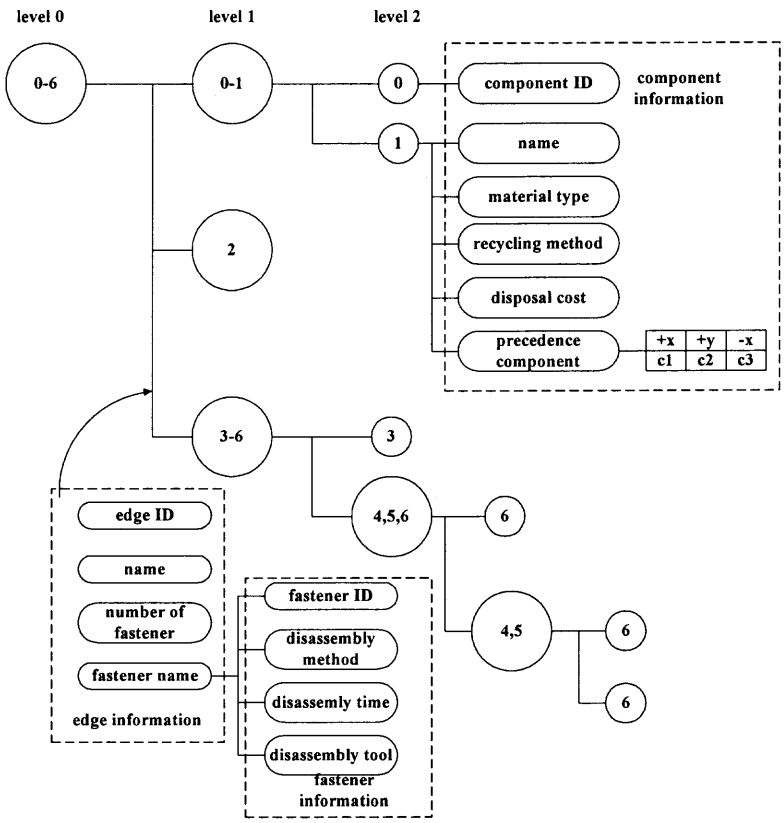


Fig. 2.8: Final disassembly tree

B. Zussman and Zhou's Method

To incorporate the uncertainty caused by different product conditions and performance of the disassembly resources, Zussman and Zhou (1998) proposed a DPN-based methodology for design and implementation of adaptive disassembly systems, where cost values h(t), decision values $\delta(t)$ and probability values $\rho(t)$ are associated with transitions and End-Of-Life value $\pi(p)$ with places. This method has three steps: (1) calculate the remanufacturing value of places d(p) and path_value of transitions d(t) based on $\pi(p)$ and h(t) in a bottom-up way; (2) divide the transitions into three groups ($T^-_{(p)} = \{ t \in T_p$ and $d(t) < 0\}$, $T^0_{(p)} = \{ t \in T_p$ and $d(t) = 0\}$, and $T^+_{(p)} = \{ t \in T_p$ and $d(t) > 0\}$ where $T_p = \{ p^\bullet, d(t) > \pi(p)\}$) and calculate the decision value for each transition as following:

- For $t \in T^-_{(p)}$, $\delta(t) = i$, where $d(t)/\rho(t)$ is the ith smallest value in $\{d(t)/\rho(t), t \in T^-_{(p)}\}$ and $\delta^0_{(p)} = \underset{t \in T^-_{(p)}}{Max}\, \delta(t)$;
- For $t \in T^0_{(p)}$, $\delta(t) = \delta^0_{(p)} + i$, where $\rho(t)$ is the ith smallest value in $\{\rho(t), t \in T^0_{(p)}\}$ and $\delta^+_{(p)} = \underset{t \in T^0_{(p)}}{Max}\, \delta(t)$;
- For $t \in T^+_{(p)}$, $\delta(t) = \delta^+_{(p)} + i$, where $d(t) * \rho(t)$ is the ith smallest value in $\{d(t) * \rho(t), t \in T^+_{(p)}\}$.

(3) choose the transition with the highest decision value in a top-down way.

Assuming the relevant information about the DPN for a handlight in Fig. 2.4 is collected as π = (-1.40, -0.90, -0.80, -0.40, -0.70, -0.20, -0.60, -0.50, 0.20, 0.10, 0.40, 0.20, 0.30, -0.4, 0.1) and h = (0.013, 0.079, 0.068, 0.062, 0.057, 0.089, 0.025, 0.067, 0.071, 0.01), two cases are compared through this methodology (ignoring Step 1). First

case is to set $\rho = (1, 1, \ldots, 1)$, which means all disassembly operations are successful initially. Then, the decision values are computed as $\delta(t_{1\text{-}10}) = (1, 2, 1, 2, 1, 1, 1, 1, 1, 1)$. Finally, the optimal DS is obtained as $\{t_1, t_3, t_9, t_6, t_7, t_{10}\}$. When the success rates are updated to $\rho(t_{1\text{-}10}) = (0.5, 0.8, 0.9, 0.8, 0.7, 0.88, 0.95, 0.75, 0.85, 0.71)$, the decision values become $\delta(t_{1\text{-}10}) = (2, 1, 1, 2, 1, 1, 1, 1, 1, 1)$. Due to the low success rate of disassembly operation t_1, the optimal DS switches to operation t_2. Then, the final sequence for the disassembly operations is changed to $\{t_2, t_5, t_7, t_8, t_9, t_{10}\}$.

C. Wave Propagation for Selected Disassembly

Disassembly of a selected component from an assembly often occurs during product recycling, reuse and maintenance. Considering this, Srinivasan and Gadh (1998a) proposed a "wave propagation" (WP) method to determine disassembly sequence with minimum component removals (OS) for a given assembly (A) and a selected component (C_x) to be disassembled. It can be categorized into the following two steps:

(1) Analyze the accessibility, disassemblability and removal influence of each component in A (their definitions shown in Table 2.3) and create a disassembly wave with C_x as a wave source, which is represented by a removal influence graph, in which nodes correspond to $\{C_i\}$ in the disassembly wave and arcs correspond to the removal influence between the mating components. For example, $C_i \rightarrow C_j$ indicates C_i is disassemblable after removing C_j; and

(2) Perform τ wave propagation, τ -WP for short, i.e., begins with C_x and keeps going with time until a boundary component is evaluated. Fig. 2.9 illustrated the use of a WP approach to a handlight, assuming $C_x=\{4\}$.

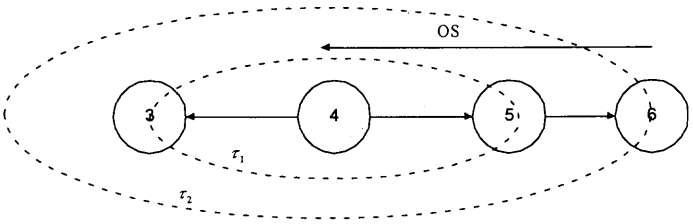


Fig. 2.9: Removal influence graph at t = 1 and t = 2 for a handlight with $C_x = \{4\}$

The worst case complexity for this method is O(nM) where M is the number of mating faces. Moreover, it is extended to incorporate the disassembly of multiple components from an assembly with a modified disassembility procedure [Srinivasan and Gadh, 1998b].

Table 2.3 Definition in WP methodology

	Definition
Accessibility	The set of directions with which C_i can move relative to C_j denoted as AC_i^j.
Disassemblability	Δ_i, a binary value, indicates if $C_i \in A$ is removable.
Removal influence	RI_i^j denotes the removal influence of $C_j \in MA_I$ on C_i (MA_I denotes the mating adjacent of C_i). If Δ_i becomes TRUE from FALSE with removal of C_j, then RI_i^j = TRUE; else RI_i^j =FALSE.
τ-WP	For every time step, the single τ wave propagates by one wavefront. For t = 0, $\tau_t = C_x$; for t >1, τ WP from $C_I \in \tau_t$ to $C_j \in MA_i$ exists if Δ_i = FALSE, $C_j \in W$ (set of components in all the wavefronts τ_p (p = 0, t – 1) at t = a) and RI_i^j = TRUE. Then $C_j \in \tau_{t+1}$.

D. Destructive Disassembly

Since a product may have some changes in its structure by its end-users, and/or contain defective parts as a result of being in an accident, which invalidate the precedence relationship, destructive disassembly is needed. Gungor and Gupta (1998a, b) proposed a methodology to develop a framework to deal with such uncertainty in optimal disassembly planning. It involves three steps:

(1) Generating a disassembly precedence matrix DP_d;

(2) Generating an optimal DS; and

(3) Carrying out disassembly with uncertainty.

Using the handlight example given in Fig. 2.1, this methodology can be understood in depth. Assuming that the battery leaks, the spring is affected, and DP_d and optimal DS (DS = (C, G, B, H, A, S, M)) are maintained, the following results are obtained (ignoring Steps 1-2):

Input: $\Phi = \{C, G, B, H, A, S, M\}$, $\Psi = 0$, $\Omega = 0$ and DV = [0, 0, 0, 1, 1, 0]

Step 3-1: ❶ $dv_C, dv_G, dv_B, dv_H = 0$, $\Psi = 0$, $\Omega = 0$ and update DP_d

❷ $dv_A = 1 \rightarrow \Omega = \{A, S, M\}$, $\Psi = \{A, S, M\}$ and update DP_d

Step 3-2: there is no component i in Ψ satisfied $dv_i = 0$, $DP_i^d = 0$. Thus using destructive disassembly, remove A, S, and M.

Lee and Gadh (1995) also developed a destructive disassembly algorithm, which accomplished the disassembly by eliminating one or more constraints through cutting off parts of a component.

E. Other Methods

Other papers consider more geometrical constrains and metrics which quantify the ease of disassemblility of components for disassembly sequencing. Subramani and Dewhurst (1991) proposed a method to identify the local constraints for the removal of a part, considering its disassembly direction and relationship with other parts. They also utilized a "branch and bound" algorithm to find a disassembly sequence that minimizes the total disassembly cost. Dutta and Woo (1991, 1995) generated a sequence of motions for removing components in a three-dimensional assembly one at a time. Especially, they

mapped the boundary representation of a given assembly to a tree called a disassembly tree. Traversing the tree in pre- and post-order yielded a minimal sequence of operations for disassembly and assembly. Beasley and Martin (1993) proposed several additional concepts to ensure the validity of a disassembly motion and developed an algorithm to easily determine the local and global feasibility of parts relying on the mating and coupling information. Based on the removability of parts, Chen *et al.* (1997) also presented an algorithm for parallel disassembly by using a tree representation, which involved the analysis of the directionality of parts and the usage of the mating graph and a data structure. Instead of considering only linear transnational motions of an assembly, rotational motions of an assembly were addressed in [Inaba *et al.*, 1997]. A genetic algorithm based methodology was also proposed to search a posture of a subassembly to avoid collision with another subassembly.

2.5 Disassembly System Design

Disassembly operations have a growing number of applications, which are being extended to almost every industry. BMW, the largest German car manufacturer, has already opened a dismantling plant in Orlando, Florida [Grogan, 1994]. The US Defense Department is now hiring military contractors to dismantle and recycle unwanted weapons [Meier, 1993]. Automatic disassembly techniques are developed and used in the areas of electronics, aerospace, construction, and industrial equipment.

Tzafestas *et al.* (1997) presented the architectural and implementation issues of an autonomous car disassembly robotic system. The use of neural network trained with the back-propagation algorithm makes improvement in the preciseness of system operations.

Kopacek and Kronreif (1996) proposed the concept of a semi-automatic disassembly cell for personal computers. To optimize the design of a cell and the usage of sensors, a five-step strategy has been specified to help the selection of the components of a robotic disassembly cell and the cell layout. They also introduced a computer-aided, semi-automated planning system named "ROBPLAN+".

Jorgensen and Andersen (1996) described a shape recognition system for automatic disassembly of TV-sets. The image processing module and a RAM net architecture were implemented.

Kotera and Sato (1996) introduced an automated disassembly process, which is developed by Mitsubishi Electric and incorporated in integrated recycling plants for refrigerators, air-conditioners, washing machines and televisions.

Zussman and Zhou (2000) presented a robotic disassembly cell for electronic products. A DPN-based modeling and control method is proposed and illustrated through disassembly of a Motorola Car radio.

Lee and Bailey-Vankuren (1997) introduced an automated disassembly system model for describing workcell and used product states collectively. A supervisory control algorithm to reduce processing time while accounting for variations in the product configurations was also proposed and illustrated through a disassembly simulation for a single-use camera.

Gungor and Gupta (1999b) first introduced the concept of disassembly line (DL) balancing as a solution to the disassembly system efficiency. A systematic approach was proposed to solve the disassembly line-balancing problem by considering several important factors unique to disassembly when balancing the disassembly line.

Due to the current research, a variety of industrial robots are used for the automation of disassembly processes, but most of their applications are suitable for simple tasks. The research thrust is to design an integrated intelligent robotic system for disassembly. Table 2.4 lists the constitution of a flexible, multi-functional, and precise robotic system and its functions [Tang *et al.*, 2000].

Table. 2.4 Constitution and function of a robotic system for disassembly

Constitution	Function	
Sensory System	Material Recognition	Obtains knowledge of assembly materials and its working conditions, especially to identify hazardous or defective parts.
	Object Recognition	Recognizes the location of assembly or parts in the system line
	Error Recovery	Recovers erroneous situations during disassembly operations
Data Base	Product Database	Stores product models, constitutions and their characteristics
	Disassembly Method Database	Stores different disassembly procedures including tools' choices and cost of the procedures, and alternative disassembly sequences corresponding to the treatment of a particular faulty condition.
	Expert System	Trains different disassembly sequences according to system status, and make them more reasonable through self-learning.
Control System	Robot Control	Controls the robots' operation, including path planner to extract optimal route for robots' shifting.
	Error Control	Evaluates error conditions and chooses the most appropriate sequences
	Supervisory Control	Communicates with all subsystems and coordinates their operations

2.6 Summary

With scientific/technological developments and social/economic evolutions, manufacturing becomes more complex and challenging. Due to the increasing

environmental pressure, there is much interest in demanufacturing among manufacturer, governments and consumers.

This chapter reviews some of the recent methodology and technology development activities in the manufacturing and demanufacturing areas. The issues about manufacturing systems address how to improve the system agility and flexibility, especially for semiconductor manufacturing systems. Product recovery issues focus on how to minimize the environmental impacts of industrial products at the end of their life.

This research addresses some issues in both areas and provides systematic approaches for manufacturing and demanufacturing system design to be discussed next.

CHAPTER 3

VIRTUAL PRODUCTION LINE DESIGN FOR BACK-END SEMICONDUCTOR MANUFACTURING SYSTEMS

As illustrated in Chapter 2, Manufacturing Execution Systems (MES) fill the gap between office planning and shop floor manufacturing and make it possible to have a real-time facility-wide view of critical processes and product data that are necessary for office decision-making and optimal factory manufacturing executions. Typical MES functions include lot tracking, resource tracking, engineering data collection, and material dispatching. Using current and accurate data, an MES is capable of initiating and guiding shop floor manufacturing activities, responding to and reporting on them as they occur. As technology transitions continue to shorten and global competitions intensify, it is the trend to deploy higher degree automation systems for manufacturers to stay leading-edge and competitive. Thus, flexible and agile manufacturing is of increasing importance. Flexibility signifies a manufacturing system's ability to adjust to customers' preferences and agility means the system's speed in reconfiguring itself to meet changing demands. Both together make it possible for manufacturers to respond instantly to the market.

Due to expensive, highly complex and time-consuming processes, semiconductor-manufacturing systems have been given a special attention. However, the current MES in semiconductor industry are designed based on a clustering of all the equipment on the shop floor into predefined physical production lines, where each production line is capable of fabricating a group of products. It is clear that the scalability of the MES, in terms of keeping the real-time responses with required resolution, to large-scale and fully automatic applications is limited, which shrinks the MES capability and thus trims down

the plant productivity. The mechanisms are lacking in existing MES to acquire real-time, better resolution responses. Thus, agility of production lines becomes extremely crucial for the semiconductor manufacturers. Although Qiu and Wysk (1999) introduced the concept of virtual production lines (VPL) to improve the agility of production lines for the back-end semiconductor manufacturing systems, the development of discrete-event driven VPL for the back-end is yet to be untaken. This chapter addresses the fundamental issues for the discrete-event driven VPL design, considering line balancing, system throughput and due dates.

This chapter is organized as follows: Section 3.1 states the problems in the back-end semiconductor manufacturing system; Sections 3.2 and 3.3 illustrate VPL design and operation algorithms, respectively; a case study is presented in Section 3.4.

3.1 Problem Statement in Back-end Semiconductor Manufacturing System

A typical back-end semiconductor line consists of the following stages:

- saw
- 2/OP-inspection
- tape attach
- die attach
- plasma cleaning
- wire bond
- 3/OP-inspection
- pre-mode bake
- plasma cleaning

- global topping
- mark
- plasma cleaning
- sold ball
- flux clean
- singulation
- final visual
- pack

At each stage, there is one or more machines to handle work-orders with different lot sizes. A lot is typically a cassette with 20 pieces of wafers, each of which is converted into 5 magazines at the die attach stage. Each magazine contains 20 pieces of strips, which are singularized at the sigulation stage. Currently, such a line is predefined to manufacture a certain type of products at a time. When a new product is introduced, the production line cannot produce it until the on-line products are finished. Obviously, the automated manufacturing system is inflexible and slow to make production transitions, thus lacking the agility and reconfigurability. Therefore, to improve the system agility and reconfigurability, this work develops the algorithms implementable in MES to facilitate the VPL design and operations. Considering line balance and system efficiency, MES regularly inspects the status of all VPLs in the system and tries to maximize the use of bottleneck machines when malfunction or exception happens. When a new work-order arrives, it can thus configure a new VPL on the fly based on the system status, new order information, and the slack time of VPLs in the system.

3.2 VPL Design

A VPL is organized as a sequence of workstations, each with one or more machines to handle processes in a stage. Once a new work-order arrives, a VPL is configured based on the system status and work-order information and dedicated to making these products. Once the work-order is finished, its VPL can be retained for another same work-orders; revised for a similar one; or dismissed so that the groups of equipment are reassigned to other VPLs to speed them up.

Based on field engineers' descriptions, this chapter makes the following assumptions:

(1) A pull mechanism [Venkatesh *et al.*, 1996] is used in a back-end semiconductor manufacturing system.

(2) Setup time for changing lot types of production at each machine is incorporated as part of the processing time of each lot. Its cost is negligible.

(3) Lot transportation time and cost from one workstation to another are negligible compared with the processing time and cost.

(4) Lots are not split in the middle of a process

(5) Each workstation is equipped with one or more machines to handle processes in a stage. Machines may have different capacity and speed in a workstation.

(6) Each work-order in a batch follows the same sequence of workstations.

(7) Although each VPL is dedicated to one work-order, some machines in a VPL can be shared by other lines.

(8) A work-order with the earliest due date has the highest priority for the idle machine assignment.

The key point of a VPL design is to maintain balance between production capacity and workload. In the wth VPL, the average speed of the jth workstation a_{jw} is calculated as:

$$a_{jw} = \sum_{i=1}^{\varepsilon_{jw}} \frac{c_{ijw}}{\tau_{ijw}} \times \beta_{ijw} \tag{3.1}$$

where,

c_{ijw}: the ith machine's capacity in the jth workstation of the wth VPL.

τ_{ijw}: the ith machine's processing time of a magazine in the jth workstation of the wth VPL.

β_{ijw}: the percentage time of the ith machine dedicated to the jth workstation of the wth VPL.

Initially, β_{ijw} equals to 1 that means the ith machine is completely assigned and dedicated to the jth workstation of the wth VPL. β_{ijw} may change based on the machine's utilization at the wth VPL operation. The machine utilization is defined as follows:

$$f = \frac{\text{Actual processing time of a machine}}{\text{Total production time}} \tag{3.2}$$

where actual processing time of a machine is cumulative time a machine is processing parts. Total production time is the difference between the completion time of the last order and the starting time of the first order.

The maximal and minimal speeds of workstations in the wth VPL are:

$$\bar{a}_w = Max\ \{a_{jw}, j=1, 2, \ldots, X_w\} \tag{3.3}$$

$$\underline{a}_w = Min\ \{a_{jw}, j=1, 2, \ldots, X_w\} \tag{3.4}$$

It is clear that the dynamically balanced VPL attempts to keep a_{1w}, a_{2w}, ..., and $a_{s_w w}$ as close to each other as possible, which can be expressed in Equation 3.5:

$$Minimize\ \ (\bar{a}_w - \underline{a}_w) \tag{3.5}$$

The ideal case is to maintain $\overline{a}_w = \underline{a}_w$.

To improve the system throughput as well as keep a line balanced, MES first tries to configure a new VPL based on the information of a new work-order and system status. If current system status cannot satisfy the requirement of this new work-order, MES needs to check all VPLs in the system and reallocate resources to make the new VPL configuration feasible. The key point is to keep the line balanced and minimize the number of equipment for a given due date of a VPL in the system. Considering this, the definitions of slack time, tardiness, and earliness of a VPL are given. Then, the whole process for a new VPL design can be presented.

The predicted slack time of the wth VPL at time t is defined as $\zeta_w(t)$:

$$\zeta_w(t) = Due(w) - Date(t) - (N_w - n_w(t))/\ \underline{a}_w \tag{3.6}$$

where $Due(w)$ is the due time of the wth work-order, $Date(t)$ is the present check time, N_w is the size of the wth work-order, and $n_w(t)$ is the number of finished magazines in the wth work-order at time t.

The tardiness of the wth VPL at time t is $\varphi_w(t)$:

$$\varphi_w(t) = max\ \{-\zeta_w(t),\ 0\} \tag{3.7}$$

The earliness of the wth VPL at time t is:

$$\psi_w(t) = max\ \{\zeta_w(t),\ 0\} \tag{3.8}$$

Thus, all VPLs in the system will be categorized into two different sets, earliness set E and tardiness set D, based on their predicted slack time.

Algorithm 3.1:

(1) Set $X = E$, the set of VPLs with earliness, and assume that the jth workstation in a VPL requires the kth class of machines and ϕ_k is the number of available machines of the kth class.

(2) Evaluate the information of the new work-order and calculate the desired speed of the $(u+1)^{th}$ VPL:

$$a_{u+1} = \frac{N_{u+1} + s_{u+1} - 1}{Due(u+1)} \tag{3.9}$$

where s_{u+1} is the number of workstations in the $(u+1)^{th}$ VPL.

(3) Evaluate the system status and calculate the maximum speed of each workstation in the $(u+1)^{th}$ VPL :

$$a_{j(u+1)} = \sum_{i=1}^{\phi_k} \frac{c_{ij(u+1)}}{\tau_{ij(u+1)}} \times \beta_{ij(u+1)} \tag{3.10}$$

(4) Get the minimal speed of the workstations in the $(u+1)^{th}$ VPL according to Eq. 3.4, i.e., $\underline{a}_{(u+1)}$.

(5) If $\underline{a}_{(u+1)} \geq a_{u+1}$, keep this new VPL running with speed $\underline{a}_{(u+1)}$. Then, choose minimum $\varepsilon_{j(u+1)}$ using Eq. 3.1 such that $a_{j(u+1)} \geq \underline{a}_{(u+1)}$, set $\beta_{ij(u+1)} = 1$ that means each machine assigned to this VPL is fully dedicated to it, and update $\phi_k = \phi_k - \varepsilon_{j(u+1)}$, where $\varepsilon_{j(u+1)}$ is the number of machines assigned to the jth workstation of the wth VPL. Then, calculate $\psi_{(u+1)}(t)$ and add this new VPL into E. Go to Step (9).

(6) If $X \neq \Phi$.

a) Select the wth VPL in X, remove it from X, and adjust its $\underline{a}_w$ as follows:

$$Min\ \varepsilon_{jw} \quad j = 1\ to\ X_w \tag{3.11}$$

Subject to:

$$\psi_w \geq 0 \tag{3.12}$$

$$a_w = \frac{N_w - n_w(t)}{Due(w) - Date(w)} \tag{3.13}$$

$$a_{jw} \geq a_w \;\; j = 1 \text{ to } X_w \tag{3.14}$$

where X_w is the number of workstations in the wth VPL.

b) If Step a) succeeds, update ε_{jw}. Exclude the redundant machines from this VPL and add them into idle machine pool ϕ_k;

c) If $\psi_w(t) = 0$, remove it from E;

d) Return to Step (3).

(7) If there is at least one machine available at each stage of the $(u+1)^{th}$ work-order and $\sum_{w=1}^{u}(\psi_w - \varphi_w) - \varphi_{u+1} > 0$, configure this new VPL with a slow rate $\underline{a}_{(u+1)}$, put this new line into D and go to Step (9).

(8) Otherwise if no VPL can be constructed, reject this new order, consider it later and exit.

(9) Calculate the production time for the $(u+1)^{th}$ work-order T_{u+1} and exit:

$$T_{u+1} = (N_{u+1} + s - 1) / \underline{a}_{(u+1)} \tag{3.15}$$

Theorem 3.1 The computational complexity of Algorithm 3.1 is $O(unl)$, where,

$$n = max\ \{X_w\} \text{ and } l = max\ \{\phi_k\}.$$

Proof: Let $m = \underset{j,w}{Max}\{\varepsilon_{jw}\}$, the complexity of each step is analyzed as follows:

(1) Step 1 takes constant time.

(2) Step 2 takes constant time.

(3) At Step 3, there are n workstations and their corresponding machine classes all have at most l idle machines. Thus, Step 2 needs $O(nl)$ time to calculate workstations speed at the worst.

(4) To get the minimal speed of the workstations in Step 4, $O(n)$ is needed.

(5) If the minimal speed of the workstations is much larger than the desired speed, the adjustment of the number of machines to each workstation costs time in Step 5, which is $O(nl)$.

(6) When the current system status cannot satisfy the requirement of this new order, the algorithm needs to update all VPLs in the system and try to spare some machines to handle this new order, which is fulfilled in Step 6. Since $0 \leq w \leq u$, $0 \leq j \leq n$ and $0 \leq i \leq m$, the complexity for Steps 6 is $O(unm)$.

(7) If Step 6 fails, Step 7 checks the total tardiness and earliness, which runs at $O(u)$.

(8) Steps 8 and 9 both take constant time.

Since $\varepsilon_{jw} \leq \phi_k$, $l \geq m$. Therefore, the computational complexity of Algorithm 3.1 is $O(unl)$.

Q. E. D.

3.3 VPL Operation

Because there are some VPLs with tardiness, how to decrease their tardiness and make sure their corresponding work-orders finished on time becomes an important issue. During operations, some malfunctions or exceptions are inevitable, which in turn cause imbalance and bottleneck. Thus, how to reallocate resources in the system to minimize their impacts is extremely critical. For this consideration, a criterion allowing machine

sharing among VPLs is first introduced. Considering the costs caused by production transitions and negative impacts on the present VPLs, only if the processing time percentages of machines in these VPLs are less than or equal to 0.5, can they be shared by other VPLs. At each discrete-event time, new batch arrival, batch completion, machine breakdown, and exception, MES checks all VPLs in the system and classifies them into different sets D and E according to their tardiness or earliness. If $D \neq \Phi$, MES finds the bottleneck workstations for each VPL in the set D. A bottleneck resource in a VPL is defined as such a workstation whose speed is the smallest (i.e., $\underline{a}_w$) and $\underline{a}_w$ is less than the desired speed a_w. If there are enough idle machines that can be allocated for processes in such workstations due to revision or dismiss of some VPLs, MES increases the speed of such bottleneck workstations by assigning a certain number of idle machines. Otherwise, MES adjusts VPLs in the system in order to spare machines' work time for the bottleneck workstations. Algorithm 3.2 fulfills this task.

Algorithm 3.2:

(1). Repeat the following steps for all VPLs:

a) Calculate $\zeta_w(t)$, $\varphi_w(t)$and $\psi_w(t)$;

b) If $\psi_w(t) > 0$, update $E = E \cup \{w\}$; otherwise if $\varphi_w(t) > 0$, update $D = D \cup \{w\}$;

c) Update β_{ijw} as $\beta_{ijw} = f_i$;

(2). If $D \neq \Phi$, do:

a) If $E \neq \Phi$, for each VPL in set E, reallocate resources through Eq. 3.11-14; and update ϕ_k and $\psi_w(t)$. If $\psi_w(t) = 0$, remove it from E;

b) Arrange VPLs in D with an ascending order of earliest due dates. For each VPL in D, do:

I) Find the bottleneck workstation v

II) Assuming the jth workstation in the wth VPL requires the kth class of machines

(i) If $\phi_k \neq 0$, assign ϑ_j machines to the jth workstation to increase its speed:

$$a_{jw} = \sum_{i=1}^{\varepsilon_{jw}+\vartheta_j} \frac{c_{ijw}}{\tau_{ijw}} \times \beta_{ijw} \ , \ \vartheta_j \leq \phi_k \tag{3.16}$$

(ii) Otherwise, check other VPLs in the system (i.e., x^{th} VPL and the yth workstation in the xth VPL requires the kth class of machines). If there is a machine i such that $\beta_{iyx} \leq 0.5$, this machine can be shared by the wth VPL.

$$a_{jw} = a_{jw} + \sum_{\substack{x=1 \\ x \neq w}}^{u} \sum_{i=1}^{\varepsilon_{jyx}} \frac{c_{iyx}}{\tau_{iyx}} \times (1 - \beta_{iyx}) \tag{3.17}$$

If Step II) fails, go to Step V)

III) Calculate $\underline{a}_w$ according to Eq. 3.4

IV) If $|\underline{a}_w - a_w| \leq 10^{-2}$, remove this VPL from D; otherwise go back to Step II)

V) Otherwise, update its tardiness φ_w. If $\varphi_w > 0$, this order will be delayed φ_w time to finish based on the present time forecast.

Theorem 3.2 The computational complexity of Algorithm 3.2 is $O(unl)$, where,

$n = max\ \{X_w\}$ and $l = max\ \{\phi_k\}$.

Proof: Let $m = \underset{j,w}{Max}\{\varepsilon_{jw}\}$, the complexity of each step in Algorithm 3.2 is analyzed as follows:

(1) It takes $O(u)$ to categorize all VPLs into sets E or D in Step 1. It takes $O(unm)$ to update machines utilization. Thus, the overall computation complexity of Step 1 is $O(umn)$.

(2) In Step 2, the computation of updating VPLs in set E takes $O(unm)$. Then, the algorithm uses two methods to minimize the tardiness of a VPL in D, which runs at $O(l)$ and $O(um)$, respectively. Since $l \geq m$, the time complexity of Step 2 is $O(unl)$.

Therefore, the computational complexity of Algorithm 3.2 is $O(unl)$.

Q. E. D.

3.4 Case Study

To fully understand the proposed methodology and algorithms, a practical implementation of the concept of VPLs is studied using a simplified back-end semiconductor line. The implementation method is first presented. Then, the experimental results are discussed.

3.4.1 Implementation Method

This work has developed simulation software as shown in Fig. 3.1, which runs under Window NT/2000 operating system written in Microsoft Function Class and SQL. The software architecture includes the following interacting and cooperating modules:

- Database
- Data Input
- Floor Planner

- Data Output

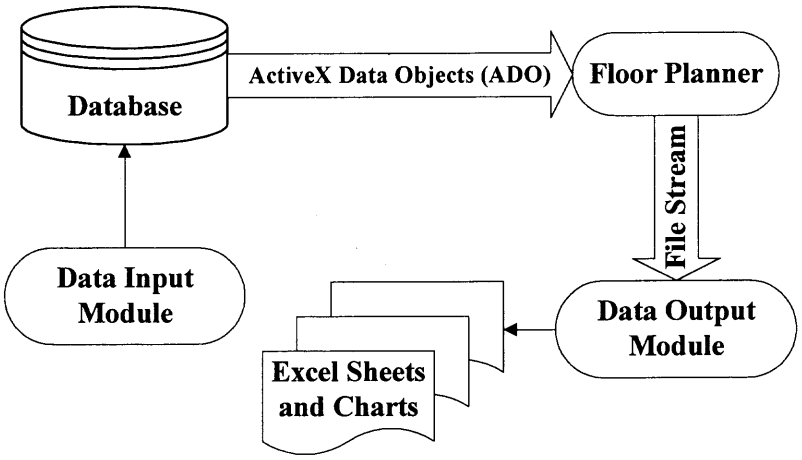


Fig. 3.1: The system software modules

During the implementation of the simulation software, the input data via random number generators is populated first into Database:

- Work-orders enter the system with the fixed due date, earliest start time and different order size. The order size has a uniform distribution between [10000, 300000].
- A work-order is usually made through the following sequential processes: Saw, 2/OP, Tape attach, Die attach, Plasma cleaning, Wire bond, 3/OP, Pre mold bake, Plasma cleaning, Glob topping, Mark, Plasma cleaning, Solder ball, Flux clean, Signulation, Final visual and Pack.
- Machine capacity and processing time are both randomly chosen with a uniform distribution, whose range depends on its corresponding machine class.
- Initially, a type of machine class has a certain number of machines, whose size is randomly decided with a uniform distribution between [80, 200].

- Users can choose the number of work-orders to be simulated using two methods through the interface shown in Fig. 3.2.

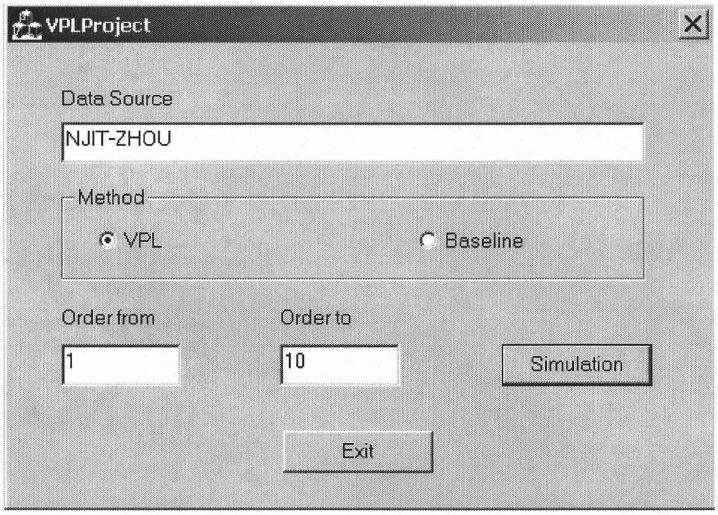


Fig. 3.2: Simulation software interface

A. Database

The database called LMDB (Lot Management Database) is established using SQL sever 7.0. It contains the relevant information of products, work-orders, workstations and machines, which are saved in six tables. Those tables are related to each other through foreign keys. The E-R schema for this database is shown in Fig. 3.3.

B. Data Input Module

Data Input Module consists of seven projects, InputMachieClass, InputMachine, InputMPT, InputProduct, InputWorkCell, InputWorkOrder and VPLDatabase, which are shown in Fig. 3.4. The first six projects are developed in MFC and Open Database Connectivity (ODBC) aimed to input the corresponding data into database LMDB. The VPLDatabase project is created through VC++ Database project Wizard that is shown in Fig. 3.5. It can be used to add, edit and delete data items in database LMDB interactively.

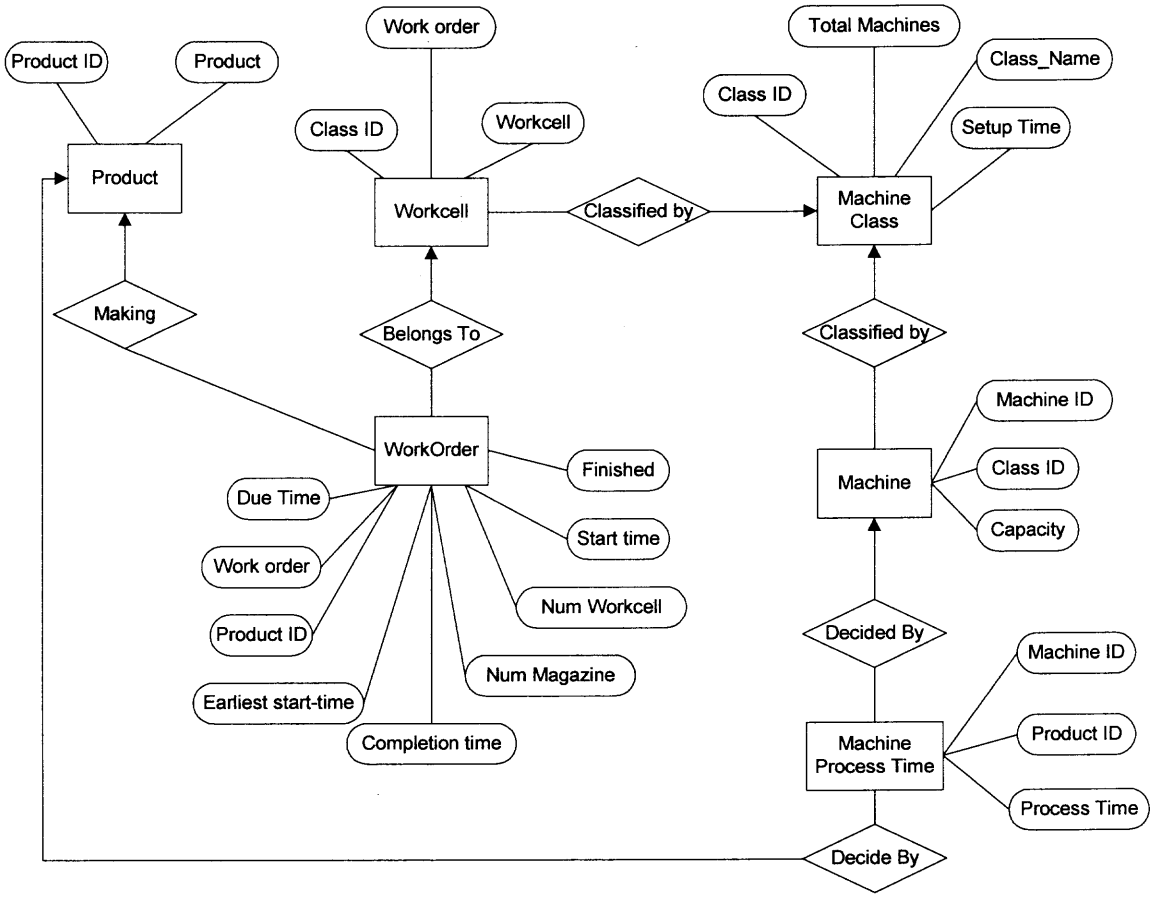


Fig. 3.3: E-R schema diagram of database (LMDB)

C. Simulation Module

Simulation Module is a program created using VC++6.0, which is called as VPLProject. Fig. 3.6 gives the diagram of VPLProject. Five classes are defined to represent the memory map of LMDB and realize the algorithms. They are listed as follows:

(1) BeltaClass: keep β values, utilization, machine classes, capacities and process times of machines.

(2) TrackingRecord: track the results of the VPL simulation and keep them as outputs.

(3) MachineClass: store the data of a machine that has been allocated to a workstation.

(4) MachineArrayClass: store machines that are used by a worksation.

(5) WorkOrderClass: store all data of a work-order and allocate machines for a work-order.

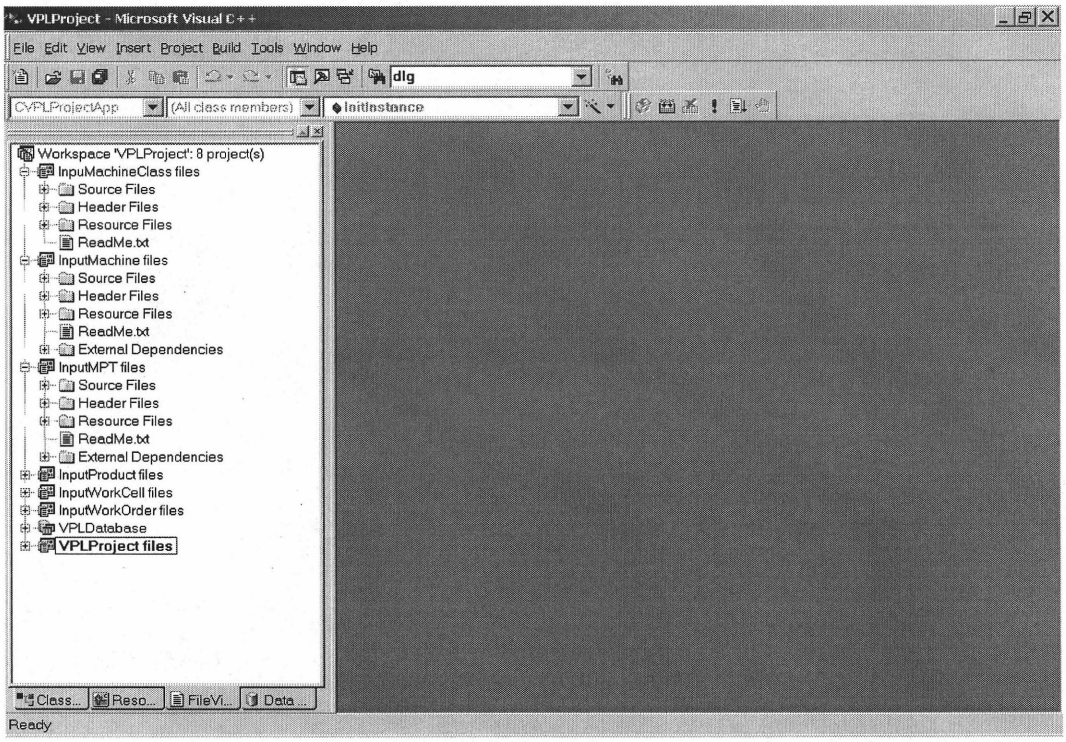


Fig. 3.4: VPL simulation software workspace

D. Data Output Module

This module just simply outputs the results that are stored in all class objects to a file called Result.dat or ResultConventional.dat, which corresponds to different methods that users choose, i.e., the proposed method or baseline method to be discussed next.

3.4.2 Simulation Results

In this simulation software, work-orders are made through the system using two methods. For the baseline method, the system successively processes these work-orders according

to their earliest start time. In other words, at each time, the line is predefined to manufacture certain type of products. In the proposed one, whenever a work-order comes, MES configures a VPL on the fly and in parallel deals with all VPLs in the system to maintain line balance.

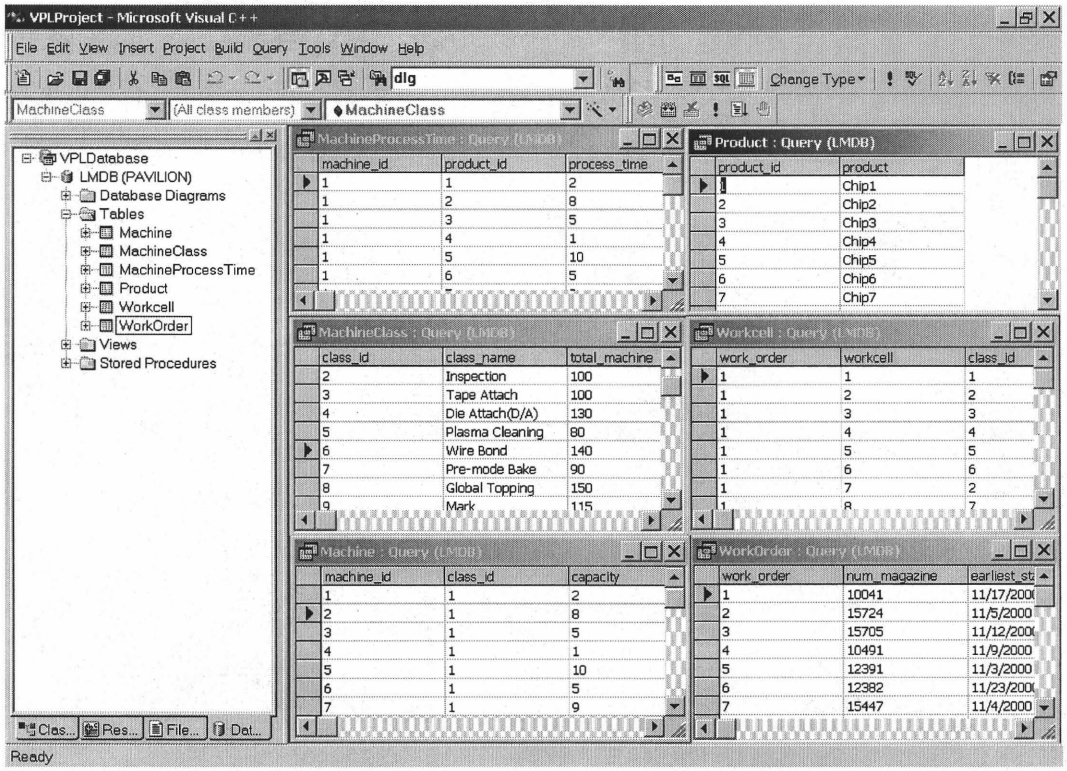


Fig. 3.5: VPLDatabase Project

To show the difference between two cases, the system throughput g is introduced first, which is the total number of all completed magazines divided by the difference between maximum completion time and minimum start time of all the order completed.

$$g = \frac{\sum_{w=1}^{u} N_w}{Max\ \{END(w)\} - Min\ \{ST(w)\}} \tag{3.18}$$

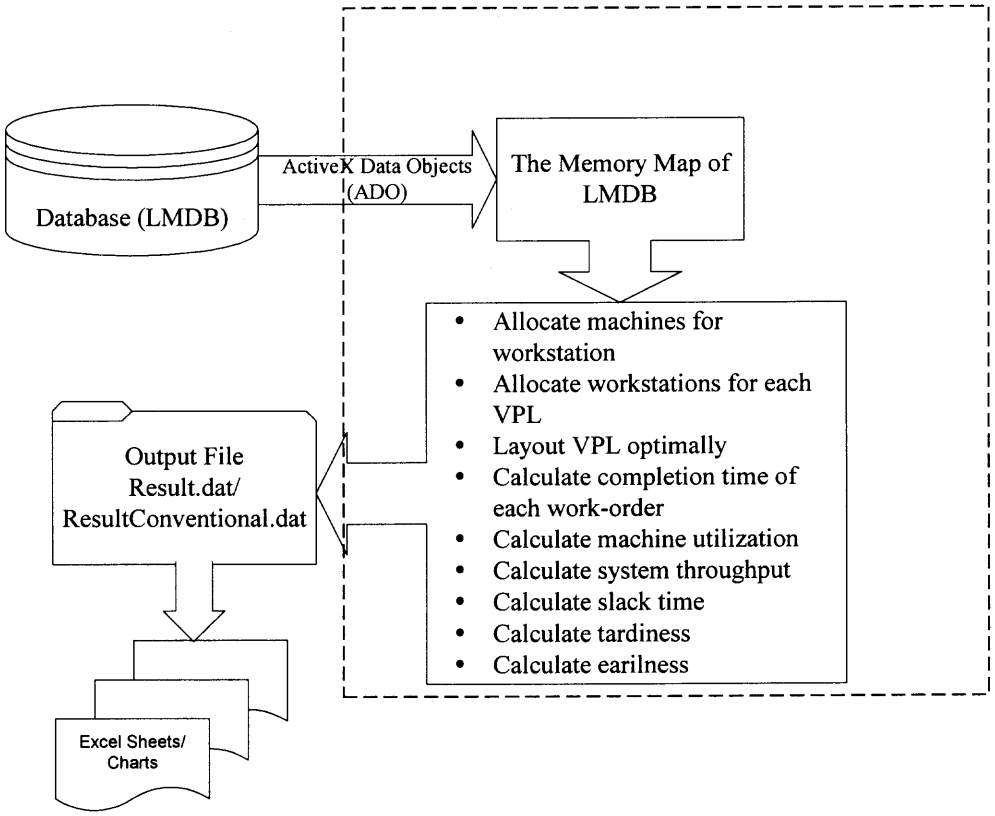


Fig. 3.6: The diagram of VPLProject (in dashed box)

Then, a sample mean, a sample variance, confidence interval estimation and standard error are introduced and a correlated sampling statistical technique is used [Banks *et al.*, 2000]. Finally, two alternatives in real-time procedures are compared through a set of experiments.

Definition 3.1:The point estimator based on n observations $\{Y_1, Y_2, \ldots, Y_n\}$ for a typical output variable, Y, is defined by:

$$\hat{\theta} = \frac{1}{n}\sum_{i=1}^{n} Y_i \tag{3.19}$$

where $\hat{\theta}$ is a sample mean based on a sample of size n.

The sample variance based on n observations $\{Y_1, Y_2, \ldots, Y_n\}$ for a typical output variable, Y, is defined by:

$$V^2 = \frac{1}{n-1}\sum_{i=1}^{n}(Y_i - \hat{\theta})^2 \tag{3.20}$$

The standard error of the point estimator $\hat{\theta}$ is defined as:

$$\hat{V}(\hat{\theta}) = \frac{V}{\sqrt{n}} \tag{3.21}$$

The approximate 100(1-α)% confidence interval for θ is defined as:

$$\hat{\theta} \pm g_{\alpha/2,(n-1)}\hat{V}(\hat{\theta}) \text{ or } \hat{\theta} - g_{\alpha/2,(n-1)}\hat{V}(\hat{\theta}) \le \theta \le \hat{\theta} + g_{\alpha/2,(n-1)}\hat{V}(\hat{\theta}) \tag{3.22}$$

where $g = \frac{\hat{\theta} - \theta}{\hat{V}(\hat{\theta})}$ and $g_{\alpha/2,\,(n-1)}$ is the 100(1-α)% percentage point of a t-distributed with (n-1) degrees of freedom.

In the first set of experiments, as the number of work-orders changes from one to ten, the speed of VPLs, tardiness time and the total earliness time are obtained for and compared between baseline and proposed methods. The results are shown in Figs. 3.7 and 3.8. Table 3.1 lists randomly generated input data.

For the simplicity, only the case when the system deals with three work-orders is considered in Fig. 3.7, and the tardiness time in Fig. 3.8 is represented as a negative number. It is clear that using the proposed method, the MES can concurrently handle multiple work-orders and adjust the speed of these VPLs according to their condition and the system capacity. When the job load is light, there is no difference between the baseline and the proposed methods. With the number of work-orders increasing, system resources become scarce, the proposed method is increasingly better than the baseline one in the tardiness time and earliness time.

Table 3.1 The input data for ten work-orders

Work-order	EST	Due	$N_{w\ (mag.)}$
1	11/3/2000 11:00:00 AM	11/8/2000 8:00:00 PM	157050
2	11/5/2000 6:00:00 AM	11/7/2000 7:00:00 AM	10491
3	11/5/2000 5:00:00 AM	11/11/2000 1:50:00 AM	15705
4	11/7/2000 7:00:00 AM	11/10/2000 7:00:00 PM	68970
5	11/10/2000 7:00:00 AM	11/15/2000 1:00:00 PM	179650
6	11/11/2000 9:00:00 AM	11/12/2000 3:00:00 PM	123820
7	11/12/2000 6:00:00 AM	11/14/2000 8:00:00 PM	15447
8	11/13/2000 11:00:00AM	11/15/2000 2:00:00 AM	19912
9	11/13/2000 3:00:00 AM	11/24/2000 4:00:00 AM	187030
10	11/15/2000 9:00:00 AM	11/20/2000 1:00:00 AM	246647

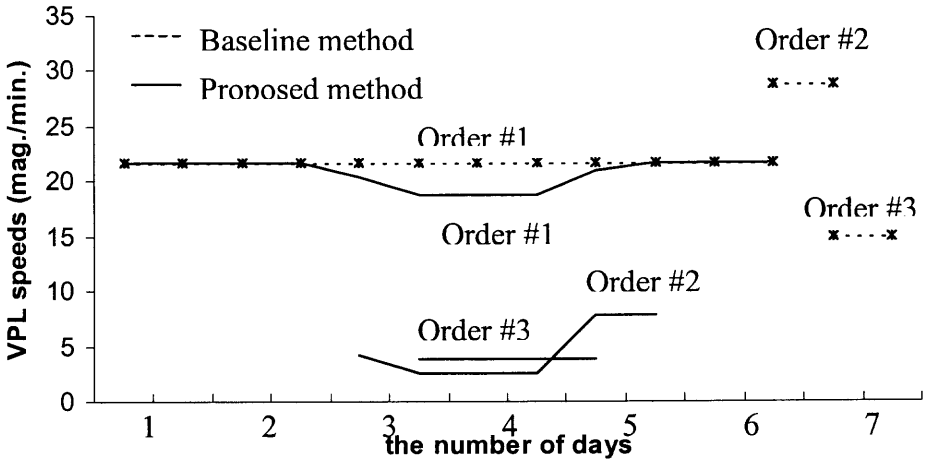


Fig. 3.7: VPL speeds for different cases when the system processes three work-orders

In the second set of experiments, the system throughput is compared between the baseline and the proposed method, where the number of work-orders is fixed to ten and the total size of work-orders increases. Using the data in Table 3.2, 95% confidence intervals for the true mean differences in system throughput are calculated as follows

(The value of $g_{\alpha/2,\,b} = g_{0.0025,\,5} = 2.57$ is obtained from t-distribution table [Banks *et al.*, 2000]):

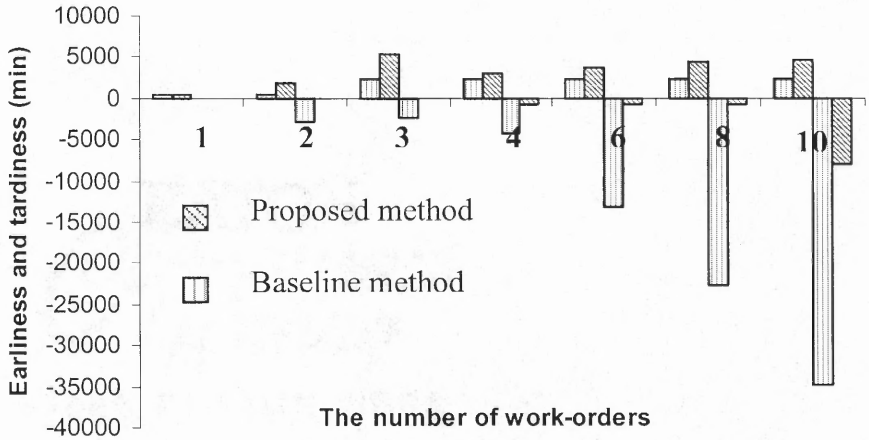


Fig. 3.8: Earliness and tardiness comparison between proposed and baseline methods

Table 3.2 The comparison of the system throughput

Ten work-orders	Total size	System throughput (mag./min)		Observed difference
		Baseline method	Proposed method	
	691045	20.44	24.55	-4.11
	837692	24.66	27.66	-3.00
	891045	26.15	29.16	-3.01
	1047834	27.65	31.90	-4.25
	1180810	29.52	33.80	-4.28
	1250019	30.78	35.56	-4.78
Sample mean		26.53	30.44	-3.91
Sample variance		13.80	16.74	0.54
Standard error				0.30

A 95% confidence interval for system throughput is given by:

$$-3.91 \pm (2.57)0.30$$

or

$$-4.68 \leq \theta_1 - \theta_2 \leq -3.14$$

The 95% confidence interval for system throughput lies completely below zero, which provides strong evidence that $\theta_1 - \theta_2 < 0$ — that is, the proposed method is better than the baseline one because its system throughput is significantly larger statistically.

3.5 Summary

This chapter addresses the algorithmic issues for the VPL design and operation in back-end semiconductor systems. Based on this proposed methodology, a new production line dedicated to a specific product can be configured and formed on the fly by using system status and work-order information. During operation, MES monitors the VPLs in the system and coordinates their speeds according to due dates and line balance. Moreover, the correct estimation of the completion time of a work-order makes it possible to check whether a work-order can be dealt with in the production facility within the delivery date a customer wants before making a commitment for on-time delivery. From the example, the approach is found to be effective in increasing the system throughput and decreasing the tardiness.

CHAPTER 4

DESIGN OF RECONFIGURABLE SEMICONDUCTOR MANUFACRUREING SYSTEMS WITH MAINTENANCE AND FAILURE

Although some promising results for the discrete-event driven VPL design are obtained in Chapter 3, these methodologies were limited when applied to a system subject to machine failure and periodic maintenance. In the recent academic literature, researchers use Petri nets (PN) to model manufacturing system and catch such dynamics [Proth and Xie, 1994; Zhou *et al.*, 1998b; Qiu and Wysk, 1999]. Queuing theories are also adopted for the analysis and design [Connors, *et al.*, 1996; Vinod and Altiok, 1986]. Despite these relatively modest activities, it is strongly emphasized that the variability of VPL is a decisive topic to investigate because of the complexity of the semiconductor manufacturing procedures. This chapter aims to present a method for the adaptive VPL design, considering machines failure and maintenance.

The rest of this chapter is organized as follows: Section 4.1 analyzes VPL with periodic maintenance; Section 4.2 studies VPL with failure and maintenance; Section 4.3 presents the reconfiguration algorithm for the VPL design; and Section 4.4 gives a simple example.

4.1 VPL with Periodic Maintenance

Machines may have different capacity and efficiency in a workstation. Moreover, their idleness due to failure is harmful to the throughput and thus increases the production cost. Therefore, most of them need to be maintained regularly to achieve high availability. It is

assumed that machines may have different maintenance time and the maintenance periodicity of workstations is independent. Thus, the speed and the mean maintenance rate of the ith machine in the jth workstation of the wth VPL are calculated as:

$$\upsilon_{ijw} = \frac{c_{ijw}}{\tau_{ijw}} \tag{4.1}$$

$$\iota_{ijw} = \frac{x_{ijw}}{\varpi_{jw}} \tag{4.2}$$

where c_{ijw} is the ith machine/worker's capacity in the jth workstation of the wth VPL/DL; τ_{ijw} is the maintenance (repair) time for the ith machine in the jth workstation of the wth VPL; x_{ijw} is the ith machine/worker's processing time in the jth workcell of the wth VPL/DL and ϖ_{jw} is maintenance periodicity of machines in the jth workstation of the wth VPL. Then, the average speed of the jth workstation a_{jw} is obtained.

$$a_{jw} = \sum_{i=1}^{\varepsilon_{jw}} \upsilon_{ijw}(1-\iota_{ijw})\beta_{ijw} \tag{4.3}$$

4.2 VPL with Failure and Maintenance

Even though periodical maintenance is a useful mean to prevent machines from failure, some malfunctions and exceptions are still inevitable, which in turn cause line imbalance and decrease the speed of workstations. In this section, each machine with failure is modeled as a state machine as shown in Fig. 4.1. This chapter assumes that machines may break down only when they are busy, and a_{yw} is the rate of magazine entering into the jth machine.

Let π^I_{ijw}, π^B_{ijw} and π^F_{ijw} denote the probabilities of the idle, busy, and failure conditions of the ith machine in the jth workstation of the wth VPL, respectively.

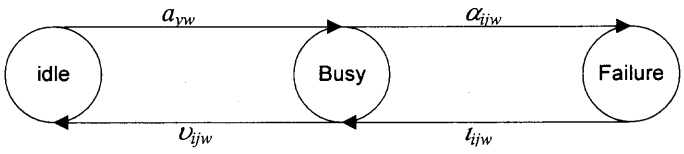


Fig. 4.1: State transition model of the ith machine in the jth workstation of the wth VPL

Then, the steady-state probabilities are obtained by solving the following linear equations [Zhou and Venkatesh, 1998]:

$$\Pi D_{ijw} = 0 \tag{4.4}$$

$$\sum_{t\in\{I,B,F\}} \pi^t_{ijw} = 1 \tag{4.5}$$

where $\pi = (\pi^I_{ijw}, \pi^B_{ijw}, \pi^F_{ijw})$ and D_{ijw} is the transition rate matrix.

$$D_{ijw} = \begin{pmatrix} -a_{yw} & a_{yw} & 0 \\ \upsilon_{ijw} & -(\upsilon_{ijw} + \alpha_{ijw}) & \alpha_{ijw} \\ 0 & r_{ijw} & -r_{ijw} \end{pmatrix}$$

Then solving Equations 4.4 and 4.5 leads to π^I_{ijw}, π^B_{ijw} and π^F_{ijw} as follows:

$\pi^I_{ijw} = \dfrac{r_{ijw}\upsilon_{ijw}}{\Delta}$ $\quad \pi^B_{ijw} = \dfrac{a_{yw}r_{ijw}}{\Delta}$ $\quad \pi^F_{ijw} = \dfrac{a_{yw}\alpha_{ijw}}{\Delta}$ where $\Delta = r_{ijw}\upsilon_{ijw} + a_{yw}\alpha_{ijw} + a_{yw}r_{ijw}$ and $\alpha_{ijw} = d_{hk}$ if the hth machine in M_k serves as the ith machine in the jth workstation of the wth VPL, where d_{hk} is the mean failure rate of the hth machine in the kth class and M_k is the kth class of machine pools.

Due to periodic maintenance, the service speed of a machine is decreased from υ_{ijw} to $\upsilon_{ijw}(1-\iota_{ijw})$ where ι_{ijw} is the maintenance rate. Thus, considering the effects of both failure and periodic maintenance of machines, equation 4.3 becomes:

$$a_{jw} = \sum_{i=1}^{\varepsilon_{jw}} \upsilon_{ijw}(1-\iota_{ijw})\beta_{ijw}\pi_{ijw}^{B} \tag{4.6}$$

4.3 Adaptive Reconfiguration for VPL

From the above analysis, it is found that failure of machines is a critical factor that affects their performance. Thus, to select a machine with higher speed and lower failure rate is very important. For example, assume that machines 1 and 2 both can serve jobs in a certain stage j, υ_{1jw} is larger than υ_{2jw} but the failure rate of machine 1 is higher than that of machine 2. Then, the system may frequently spend resources to perform the operation associated with machine 1 without success, and then perform with machine 2 with success. This will certainly decrease the system throughput. In the other words, if the system can select machine 2 prior to machine 1, the system throughput may be improved. Introducing priority for machines and integrating them into VPL design should enable system to adapt for the best reconfiguration of VPLs.

Based on the past performance of machines, the values of ξ_{hk} assigned to the hth machine in the M_k is decided as: $\xi_{hk} = e$ if $\upsilon_{ijw}(1\text{-}d_{hk})$ is the eth greatest value in $\{\upsilon_{ijw}(1\text{-}d_{hk}), h \in I_k\}$ (assuming the hth machine in M_k is idle and may serve as the ith machine in the jth workstation of the wth VPL). It is clear that the priority of a machine may change depending on the job it serves.

When a new job comes or other events (e.g., breakdown) happen, the system will update the idle machine pools. If the mean failure rate of a machine is higher than a pre-set number K_{jw}, the allowed maximum failure rate (assuming this machine can serve jobs

in the jth workstation of the wth VPL), remove this machine from its corresponding idle machine pool and add it into R_k, where machines get immediate repairs.

At initialization, machines all work in good condition. Due to the statistical data from a real time semiconductor manufacturing system, the mean failure rate of a machine is assumed known. Thus, to configure a VPL for a new work-order, the system orders these machines in I_k in a decreasing order of their priority values according to work-order information. Then, machines are selected from these sorted idle machine pools according to their values from the highest to the lowest. The extended algorithm is as follows:

Algorithm 4.1 (Adaptive reconfiguration):

(1) Set $X = E$, the set of VPLs with earliness, and assume the jth workstation in a VPL requires the kth class of machines.

(2) Evaluate the information of the new work-order and calculate the desire speed of the $(u+1)^{th}$ VPL:

$$a_{u+1} = \frac{N_{u+1} + s_{u+1} - 1}{Due(u+1)} \tag{4.7}$$

(3) Update idle machine pools: if the failure rate of an idle machine in the I_k is larger than $K_{j(u+1)}$, move it into its corresponding repair machine pool R_k; assign priorities to the remained machines in I_k and order them in decreasing priority values

(4) Evaluate the system status and calculate the maximum speed of each workstation in the $(u+1)^{th}$ VPL (assuming the arrival rate of jobs is larger than speed of any machine in a VPL):

i). First, calculate the maximum speed of workstations with maintenance:

$$a_{j(u+1)} = \sum_{i=1}^{\phi_k} \upsilon_{ijw}(1 - \iota_{ijw})\beta_{ijw} \tag{4.8}$$

ii). Second, get the minimal speed of the $(u+1)^{th}$ VPL only considering workstations with maintenance, that is: $\underline{a}_w = Min\ \{a_{jw}, j \in [1, X_w]$ and the jth workstation without failure$\}$

iii). Then, calculate speeds of workstations with failures and maintenance (Note that $\underline{a}_w$ is arrival rates of machines in such workstations):

$$a_{jw} = \sum_{i=1}^{\phi_k} \upsilon_{ijw}(1-\iota_{ijw})\beta_{ijw}\pi_{ijw}^{B} \tag{4.9}$$

(5) Re-get the minimal speed of the $(u+1)^{th}$ VPL according to $\underline{a}_w = Min\ \{a_{jw}, j=1, 2, \ldots, X_w\}$

(6) If $\underline{a}_{(u+1)} \geq a_{u+1}$, keep this new VPL running with speed $\underline{a}_{(u+1)}$. Then, choose the minimum number of machines ($\varepsilon_{j(u+1)}$) using Eq. 3.3 such that $a_{j(u+1)} \geq \underline{a}_{(u+1)}$ and they are the first $\varepsilon_{j(u+1)}$ machines in I_k, then update $\phi_k = \phi_k - \varepsilon_{j(u+1)}$, and calculate $\psi_w(t)$. If $\psi_w(t)>0$, add this new VPL into set E and go to Step (10)

(7) If $X \neq \Phi$, do:

a) Select the wth VPL in the set X, remove it from set X, and adjust its $\underline{a}_w$ as follows:

$$Min\ \varepsilon_{jw} \quad j = 1\ to\ X_w \tag{4.10}$$

Subject to:

$$\psi_w \geq 0 \tag{4.11}$$

$$a_w = \frac{N_w - n_w(t)}{Due(w) - Date(w)} \tag{4.12}$$

$$a_{jw} \geq a_w \quad j = 1\ to\ X_w \tag{4.13}$$

b) If Step a) succeeds, update ε_{jw}. Exclude the redundant machines from this VPL and add them into idle machine pool I_k;

c) If $\psi_w = 0$, remove it from the set E;

d) Return to Step (2)

(8) If there is at least one machine available at each stage of the $(u+1)^{th}$ work-order and $\sum_{w=1}^{u}(\psi_w - \varphi_w) - \varphi_{u+1} > 0$, configure this new VPL with a slow rate $\underline{a}_{(u+1)}$, put this new line into the set D and go to Step (10)

(9) Reject this new order, consider it later and exit.

(10) Calculate the production time for the $(u+1)^{th}$ work-order T_{u+1} and exit:

$$T_{u+1} = (N_{u+1} + s - 1) / \underline{a}_{(u+1)} \tag{4.14}$$

Theorem 4.1. The computational complexity of Algorithm 4.1 is $O(unl)$, where $n = max\ \{X_w\}$ and $l = max\ \{\phi_k\}$.

Proof: Since Algorithm 4.1 is an extension of Algorithm 3.1, the differences are to update idle machine pools and order idle machines based on their priorities in Step 3, and to calculate the maximum speed of workstations in Step 4.

(1) In Step 3, let ω is the number of machine classes, the time complexity of Step 3 is $O(l\omega)$.

(2) In Step 4, all the operations are the same as those in Algorithm 3.1, except that the maintenance rate and probability of busy condition of machines are introduced, where they both take constant time. Thus, the total complexity of Step 4 is $O(umn)$.

Since one type of machines may serve jobs in different workstations, $\omega \leq n$. Because of $l \geq m$, the computational complexity of Algorithm 4.1 is $O(unl)$.

Q. E. D.

Because of tardiness, malfunctions and exceptions during operation, how to reallocate resources to minimize these negative impacts is extremely important. Considering these, an adjustment algorithm is presented based on the method in Chapter 3:

Algorithm 4.2 (Dynamical adjustment)

System adjusts VPLs when tardiness, malfunctions or exceptions happen:

(1) Repeat the following steps for all VPLs:

a) Calculate $\zeta_w(t)$, $\varphi_w(t)$and $\psi_w(t)$;

b) If $\psi_w(t) > 0$, update $E = E \cup \{w\}$; otherwise if $\varphi_w(t) > 0$, update $D = D \cup \{w\}$;

(2) If $D \neq \Phi$, do:

a) If $E \neq \Phi$, for each VPL in set E, Reallocate resources through Equations 4.4-4.7; and update ϕ_k and $\psi_w(t)$. If $\psi_w(t) = 0$, remove it from the set E;

b) Arrange VPLs in D with an ascending order of earliest due dates. For each VPL in D, do:

I) For $j = 1$ *to* X_w (assuming the jth workstation in the wth VPL requires the kth class of machines)

(i) If $\phi_k \neq 0$, update I_k using the same way as Step 3 in Algorithm 4.1, then, assign the first ϑ_j ($\vartheta_j \leq \phi_k$) machines in I_k to the jth workstation to increase its speed.

for ($i = \varepsilon_{jw}$ to $\varepsilon_{jw} + \vartheta_j$) do:

if($\alpha_{ijw} = 0$) $a_{jw} = a_{jw} + \upsilon_{ijw}(1-\iota_{ijw})\beta_{ijw}$

else $a_{jw} = a_{jw} + \upsilon_{ijw}(1-\iota_{ijw})\beta_{ijw}\pi^{\beta}{}_{ijw}$

(ii) Otherwise, check other VPLs in the system (i.e., x^{th} VPL and the vth workstation in the xth VPL requires the kth class of machines). If there is machine i such that its utilization is less than 0.5, this machine can be shared by wth VPL.

for ($x=1$ to u ($x \neq w$))

for ($i=1$ to ε_{vx})

if ($\alpha_{ivx} = 0$ && $\beta_{ivx} < 0.5$) $a_{jw} = a_{jw} + \upsilon_{ivx}(1-\iota_{ivx})(1-\beta_{ivx})$

else if ($\beta_{ivx} < 0.5$) $a_{jw} = a_{jw} + \upsilon_{ivx}(1-\iota_{ivx})(1-\beta_{ivx})\pi^{\beta}_{ivx}$

II) Calculate $\underline{a}_w = Min\ \{a_{jw}, j=1, 2, \ldots, X_w\}$

III) If $|\underline{a}_w - a_w| \leq 10^{-2}$, remove this VPL from D

IV) Otherwise, update its tardiness φ_w. If $\varphi_w > 0$, this order will be delayed φ_w time to finish based on the present forecast.

Theorem 4.2. The computational complexity of Algorithm 4.2 is $O(unl)$, where $n = max\ \{X_w\}$ and $l = max\ \{\phi_k\}$.

Proof:

Since Algorithm 4.2 is an adjustment of Algorithm 3.2, the only difference is that the maintenance rate and probability of busy condition of a machine is introduced in Step 2, where a constant number of operations are performed. Thus, the computational complexity of Algorithm 4.2 is the same as that of Algorithm 3.2, that is, $O(unl)$.

Q. E. D

4.4 An Example

Using the same example in Chapter 3 to demonstrate the above methodology, a simplified back-end semiconductor line is used to run three cases. The first two cases are the same as presented in previous chapter. The adaptive algorithm proposed in this chapter is applied in the third case. Through these cases, the system is already running and two new work-orders need to be made right now. For the simplicity, it is assumed that except in "Saw" and "Wire bond" workstations, machines are reliable. Workstations are equipped with identical machines with different failure rates and machines are fully dedicated to its corresponding workstation.

For these three cases, this chapter considers the following instance: before configuring a VPL for the first new work-order, there are ten machines with a failure rate in I_5, which can process jobs in "Wire bond" workstation. Their failure rates are {0.004, 0.008, 0.01, 0.01, 0.01, 0.008, 0.008, 0.008, 0.04, 0.04}. Based on field engineers' experience, the K_{jw} depends on ϖ_{jw}, the machine maintenance periodicity. In this example, the baseline for the highest failure rate is assumed 0.004 (1/min). On the fifth day of the first new work-order being processed, three machines in the "Saw" workstation break down. The next day after that, eighteen idle machines are added into I_5 since an old work-order was finished. These input data are shown in Tables 4.1 and 4.2.

Table 4.1 The input data for two work-orders

Work-order	EST	Due	$N_{w\,(mag.)}$
1st order	1/1/00	1/16/00	10^5
2nd order	1/3/00	1/13/00	10^4

Table 4.3 lists the computation results, which compare the changes of speeds of production lines, system throughput and utilization of "Wire bond" workstation, when

two work-orders going through the three cases. It is clear to see that in the last two cases, the system can concurrently handle multiple work-orders and adjust the speed of these VPLs according to their conditions and system capacity. Thus, the systems' throughputs increase by 26.7%. In the third case, due to the priority introduced, system can monitor machines' performance and immediately repair machines with a high failure rate, that is $d_{ik} > 0.004$, instead of keeping such machines running until the next periodic maintenance takes place. Thus, the second work-order was finished in advance by one day, and workstation utilization increases by 10.2% and 4.2% than those in the first and second cases, respectively.

Table 4.2 The input data for workstations

	Saw	D/A	Cure	Plasma	Wire bond	Inspection
c_{ijw} (mag.)	1	1	24	2	1	1
τ_{ijw} (min.)	7	6	120	4	30	2.5
χ_{ijw} (hour)	1	1	1	1	1	1
ϖ_{jw} (day)	7	7	7	7	7	7
ϕ_k	50	50	50	40	150	30

4.5 Summary

This chapter addresses the design issues in reconfigurable back-end semiconductor manufacturing systems. A queuing network model is used to analyze a workstation's throughput due to its failure, unplanned and planned maintenance. Because unexpected breakdowns of machines can degrade significantly the system performance, a priority value is introduced for each idle machine. Adaptive algorithms are developed based on this and allow the system to dynamically select machines and adjust the VPL. From the example, the approach is found to be effective in increasing system throughput and machine utilization.

Table 4.3 The computation results

# of days	conventional case ($\underline{a}_w$ 1/min)		2nd case (algorithms in Chapter 3)		3rd case (extended algorithms)	
	1st work-order	2nd work-order	1st work-order	2nd work-order	1st work-order	2nd work-order
1	4.742		4.742		4.659	
2	4.742		4.742		4.659	
3	4.742		4.639	0.123	4.639	0.318
4	4.742		4.639	0.123	4.639	0.318
5	4.429		4.429	0.123	4.429	0.318
6	4.429		4.639	0.719	4.639	0.915
7	4.429		4.639	0.719	4.639	0.915
8	4.429		4.639	0.719	4.639	0.915
9	4.429		4.639	0.719	4.639	0.915
10	4.429		4.639	0.719	4.639	0.915
11	4.429		4.639	0.719	4.639	0.915
12	4.429		4.639	0.719	4.639	0.915
13	4.429		4.639	0.719	4.639	
14	4.429		4.639	0.719	4.639	
15	4.429		4.639		4.639	
16	4.429					
17	4.429					
18		5.358				
19		5.358				
g	5789		7333		7333	
f	90.2% (the 5th workstation)		95.4% (the 5th workstation)		99.4% (the 5th workstation)	

CHAPTER 5

AN INTEGRATED APPROACH TO DISASSEMBLY PLANNING AND DEMANUFACTURING OPERATION

As shown in Chapter 2, there are new forces at work, such as increased awareness of the state of the environment by both customers and producers, recycling regulations and resource conservation needs [Brennan *et al.*, 1994]. End-of-life options for discarded products and material have become an emerging area of engineering research. This concern is being driven by product take-back initiatives in Europe, international certification standards for environmental management, increasingly stringent regulations governing hazardous materials world-wide, requirements for waste stream purity for landfill and incineration, as well as a growing consumer preference for "green" product characteristics. These environmental drivers, however, are offset by the increased complications of modern products in terms of material variety, quantity of components and fasteners, and geometric and functional complexity of these parts. These complexities manifest themselves in the dismantling of assemblies such as automobiles ($\sim 10^4$ parts), consumer appliances ($\sim 10^2$ parts), information technology equipment ($\sim 10^4$ parts), or airframes ($\sim 10^7$ parts) and identify a need for developing a new approach towards modeling and planning of their dismantling and demanufacturing processes.

Demanufacturing is an economical form of recovering subsystems of manufactured goods after consumer use. It can be defined as a process in which obsolete, worn-out or malfunctioning products are brought back to original specifications and conditions, or converted into raw materials and parts for reuse. Central to demanufacturing is the disassembly process. The objective of product disassembly is to maximize the value of

parts and materials recovered for repair and reuse and to minimize the disposal cost and environmental burden. Some methods and techniques are proposed and some tools are developed that are efficient and successful in solving some specific problems. However, very little effort has been focused on demanufacturing from a system perspective. In fact, the allocation of resources in a demanufacturing system is very critical to efficient operations. A unified representation scheme to capture the assembly plan and the control level is formulated in [Thomas *et al.*, 1994]. Due to the differences between disassembly and assembly, the fundamental issues in design and operation of an integrated flexible demanufacturing system (IFDS) remain to be addressed. The EOL value of products and the capacity of systems need to be considered in order to bring realistic industrial applications.

This chapter is organized as follows: Section 5.1 describes the generic model for IFDS. Section 5.2 focuses on disassembly modeling for products and resources using Petri nets. Section 5.3 presents the disassembly process planning. Section 5.4 gives a case study.

5.1 Generic Model for Integrated Flexible Demaufacturing Systems

Demanufacturing is becoming an integral part of a product life cycle. It performs a set of functions (such as inspection and disassembly) to recover values from products and waste streams, and ships these recovered materials and components for reuse, recycling, reengineering and remanufacturing. An IFDS as shown in Fig. 5.1 consists of the following interacting and cooperating units:

- Database

- Collection Unit
- Inspection Unit
- Disassembly Workstation
- Sorting Unit
- Sensory Unit
- Supervision

5.1.1 Database

Database is the information center of an IFDS. It consists of three parts, Product Database that stores product model, constitution and characteristic; Disassembly Method Database that has product disassembly sequences, and treatment methods for products under faulty conditions; and Resource Database that keeps the knowledge of available machines, robots, and human operators.

5.1.2 Collection Unit

Discarded products are transported to the demanufacturer from various sources, which introduce uncontrolled variability in terms of product types and conditions. Different models may need different disassembly facilities, and batches of products of same disassembly families provide faster disassembly and lower setup and material handling time by keeping the same tooling and workstation setup. Considering these, an IFDS performs a preliminary screening at the Collection Unit to separate coming products according to their models and send them to different storage areas. In another word, the group technology concept is used to improve the system efficiency [Ng, 1991; Sedqui *et al.*, 1995].

5.1.3 Inspection Unit

Before going to disassembly, batches of products of the same model are tested for their potential reusability in Primary and Secondary Inspection areas. In the Primary Inspection area, a Product Petri net (PPN) with all feasible disassembly sequences and a Workstation Petri net (WPN) with workstation information are introduced. During the first inspection, the system obtains information of products, such as working condition, model and constitution, and determines the Disassembly Path (DP) with the EOL value based on the real-time machine availability. Products in very poor condition are immediately sent for disposal (e.g., smelting, shredding and landfill). In the Secondary Inspection area, a Scheduling Petri net (SPN) is proposed. With the fixed DP and resource information, the system will continue to optimize the disassembly operations to obtain the maximum operational efficiency of demanufacturing facilities. The concepts and algorithms based on WPN, PPN, and SPN are discussed in Sections 3 and 4.

5.1.4 Disassembly Workstation

According to the DP determined in Primary Inspection, a product is sent to certain disassembly workstations. In an example IFDS, there are four workstations that fulfill different disassembly tasks for a personal computer. Workstation 1 separates case and cables from a computer product. Workstation 2 deals with the disassembly of disk drive, hard drive and CD-ROM. Workstation 3 performs the disassembly of display adapter, audio card and network adapter. Workstation 4 accomplishes the disassembly of power supply units. Due to the complexity of disassembly tasks, workstations are equipped with different machines and robots.

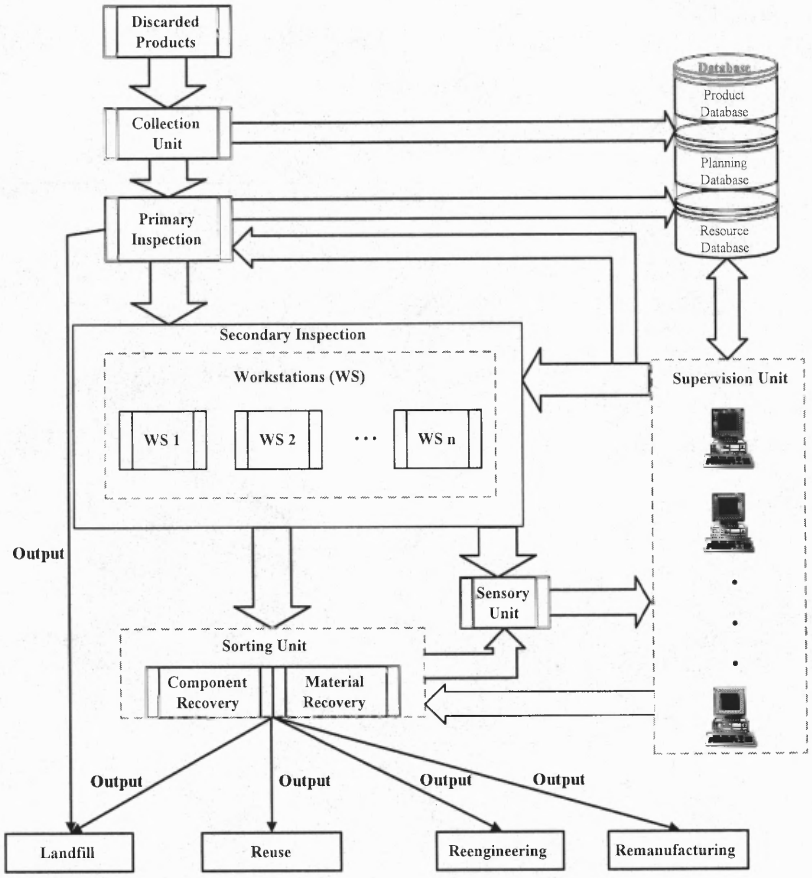


Fig. 5.1: A Generic Model for Integrated Flexible Demanufacturing System (IFDS)

5.1.5 Sorting Unit

The sort operation segregates the disassembled components and materials into the proper commodities. Commodity containers are weighed and information on commodity type and container's weight is sent back to the Sorting controller. After that, different commodities are shipped to different vendors for reuse, remanufacturing, reengineering or disposal.

5.1.6 Sensory Unit

The Sensory Unit monitors the status of machines and buffers, and feedback information to Supervision Unit. It plays an important role in decision making and maintaining excellent system performance.

5.1.7 Supervision Unit

Supervision Unit is the heart of an IFDS, which consists of workstation, robot, sorting and super-controllers. Workstation controller is responsible for controlling the operations in its machines and buffers. Robot controller extracts the optimal route for a robot's movements. They both check the status messages from their responsible units received from Sensory Unit. Once malfunction or exception happens, such as breakdown of a machine and container overflow, it evaluates errors and chooses appropriate tactic to handle them. Sorting Controller checks signals from Sorting Unit, appropriately removes the filled container, and replaces it with a new one. Super-controller just communicates with other controllers and coordinates their operations.

5.2 Modeling Disassembly Processes

One of the key purposes of dealing with discarded products is to maximize recycled resources with the consideration of ecological, environmental and economic factors. Zussman *et al.* (1995) termed the multipurpose goal as "increasing the EOL value". Since each disassembly task is processed at a certain workstation, the realizability of a disassembly task is affected by the availability of machines in its corresponding workstation. Moreover, a workstation has finite machines with different speeds. Best

machine assignment is thus a key to high system throughput. In an IFDS, not only are the EOL value and the capacity of each workstation considered, but also the efficient machine assignment is emphasized. Considering these, this work proposes the following three petri nets models:

(1) A Product Petri net (PPN) with all feasible disassembly sequences and the EOL value of products, subassemblies and components.

(2) A Workstation Petri net (WPN) to model the status of workstations and provide the availability information of workstations.

(3) A Scheduling Petri net (SPN) for local optimal machine scheduling.

Subsection 5.2.1 gives the definition of Petri nets (PNs). Then definitions of WPN, PPN, and SPN are presented in Subsections 5.2.2-5.2.4, respectively.

5.2.1 Petri Nets (PNs)

Petri nets, as a graphical and mathematical tool, provide a uniform environment for modeling, formal analysis, and design of discrete event systems [Zurawski and Zhou, 1994]. A Petri net may be identified as a particular kind of bipartite directed graph populated by three types of objects. They are places, transitions, and directed arcs connecting places to transitions and transitions to places. Pictorially, places are depicted by circles and transitions by bars. A place is an input (output) place to a transition if there exists a directed arc connecting the transition (place) to the place (transition). In order to study dynamic behavior of the modeled system, each place may potentially hold either none or a positive number of tokens, pictured by small solid dots. Formally, a Petri net can be defined as follows:

Definition 5.1 A Petri net is defined as a five-tuple: PN = (P, T, I, O, M)

1. $P = \{p_1, p_2, \ldots, p_n\}$ is a finite set of places.
2. $T = \{t_1, t_2, \ldots, t_s\}$ is a finite set of transitions, $P \cup T \neq \Phi$, $P \cap T = \Phi$
3. $I: P \times T \rightarrow \{0, 1\}$ is an input function that defines the set of directed arcs from P to T, where $I_{ij} = 1$, if p_i is an input place for t_j, otherwise 0.
4. $O: P \times T \rightarrow \{0, 1\}$ is an output function that defines the set of directed arcs from T to P, where $O_{ij} = 1$, if p_i is an output place for t_j, otherwise 0.
5. $M: P \rightarrow \{0, 1, 2, \ldots\}$ is a marking vector whose i^{th} component represents the number of tokens in the ith place. An initial marking is denoted by m_0.

The defined Petri net called ordinary PN can be further extended to represent various aspects of modeled systems. For instance, places may represent status of resources. Time and cost can be introduced into ordinary Petri nets to analyse system performance. This thesis uses and extends the ordinary PN to model workstation status, product disassembly sequences, and scheduling.

5.2.2 Workstation Petri Net (WPN)

Definition 5.2 A Workstation Petri net (WPN) is a PN where:

1. $P = W^a \cup W^b = \{w_1^a, w_2^a, \ldots, w_n^a\} \cup \{w_1^b, w_2^b, \ldots, w_n^b\}$, where the set of places W^a represents the available machine pools and the set of places W^b stands for the busy machine pools in certain workstations.
2. $T = T^b \cup T^e = \{t_1^b, t_2^b, \ldots, t_n^b\} \cup \{t_1^e, t_2^e, \ldots, t_n^e\}$, where the set of transitions T^b (T^e) represents the beginning (end) of operations processed in workstations.

3. I = 2n × 2n identity matrix, where the diagonal identifies that each place in W^a has a directed arcs to its corresponding transition in T^b, and each place in W^b has a directed arcs to its corresponding transition in T^e.

4. $$O = \begin{bmatrix} 0 & 1 & 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 1 & \dots & 0 & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 & 0 \\ & \vdots & & \vdots & & \vdots & \\ 0 & 0 & 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & 0 & 0 & \dots & 1 & 0 \end{bmatrix}$$, which points out that each transition in T^b has a directed arcs to its corresponding place in W^a, and each transition in T^e has a directed arcs to its corresponding place in W^b.

An example is given in Fig. 5.2 where workstations are modeled. The present markings show that both machines are busy at Workstation 1, and idle at Workstations 2-4.

5.2.3 Product Petri Net (PPN)

Definition 5.3 A Product Petri net (PPN) is an eight-tuple extended PN,

PPN = (P, T, I, O, M, π, h, r, λ, δ) where:

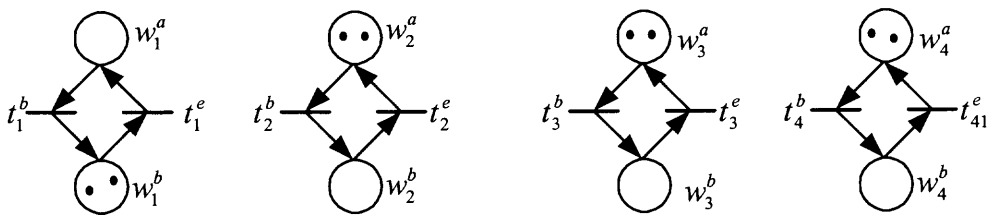


Fig. 5.2: A simple example of WPN with four workstations

- $P = W \cup Q$, where the set of places $W = \{w_1, w_2, \ldots, w_n\}$ represents the available machine pools in certain workstations, whose information comes from the WPN. The set of places Q stands for product, subassemblies or components.

(1) $\exists\, p \in Q$ such that ${}^{\bullet}p = \Phi$: This place is usually denoted by p_1 and named as root or product place.

(2) $\exists\, Q' \subset Q$ such that $Q' \neq \Phi$ and $\forall p \in Q'$, $p^{\bullet} = \Phi$. Q' is called leaves.

(3) $I(p, t) = O(p, t)$, $\forall t \in T$, $\forall p \in W$.

(4) $m_0(p_1) = 1$ and $m_0(p) = 0$, $\forall p \in Q'$.

(5) $m_f(p) = m_0(p)$, $\forall p \in W$, and their values come from the WPN.

- $\pi: Q \rightarrow R$ is an EOL value function assigned to a place, which is defined as $\pi(p) = \max\{\pi_{reuse}(p), \pi_{remanufacturing}(p), \pi_{reengineering}(p), \pi_{landfill}(p)\}$, where $\{\pi_{reuse}(p), \pi_{remanufacturing}(p), \pi_{reengineering}(p), \pi_{landfill}(p)$ are the EOL values when the product, subassembly, or component represented by p is reused, remanufactured, reengineered, or landfilled [Zussman *et al.*, 1995].

- $h: T \rightarrow R$ is a cost value function associated with transitions, which mainly depends on the accessibility and difficulty of disassembly.

- $r: W \rightarrow R^+$ is the time delay function associated with each workstation-available place. $r_i = 0$ when there is at least one machine available in the ith workstation; otherwise, r_i is the waiting time for the first available machine in the ith workstation, in other words, it represents a delay for a corresponding subassembly to wait for a sequence of disassembly processes in the ith workstation.

- $\lambda: T \rightarrow R$ is a conservative time function associated with a transition. It is the maximum time needed to finish the operation a transition represents.

- δ: T $\rightarrow$ N is a decision value associated with a transition. This value is decided before the corresponding disassembly operation, or firing of the transition.

Condition (3) of a PPN assures that the token in a workstation to enable certain transitions is released after their firing. EOL values, costs, decision values and time delays are all used to generate an optimal disassembly path. Fig. 5.3 gives a simple example of PPN for a personal computer disassembly, where letters a to k represent the components of a personal computer as listed in Table 5.1.

Table 5.1 Component List for a PERSONAL COMPUTER

Letter in *PPN*	**Description**
a	Motherboard
b	Disk Drive
c	Hard Drive
d	CD-ROM
e	Display Adapter
f	Audio Card
g	Network Adapter
h	Supply Unit
i	Memory Extension
j	Case
k	Cables

5.2.4 Scheduling Petri Net (SPN)

This chapter makes the following assumptions for an IFDS:

1 All the machines in a workstation have the same functions but may exhibit different speed.

2 Each machine can process at most one task at a time.

3 Each task is non-preemptive, requiring one and only one machine at any time.

4 Setup time is included in process time, and two same operations in sequence need no setup time.

5 The transportation time between workstations is negligible compared with the disassembly time.

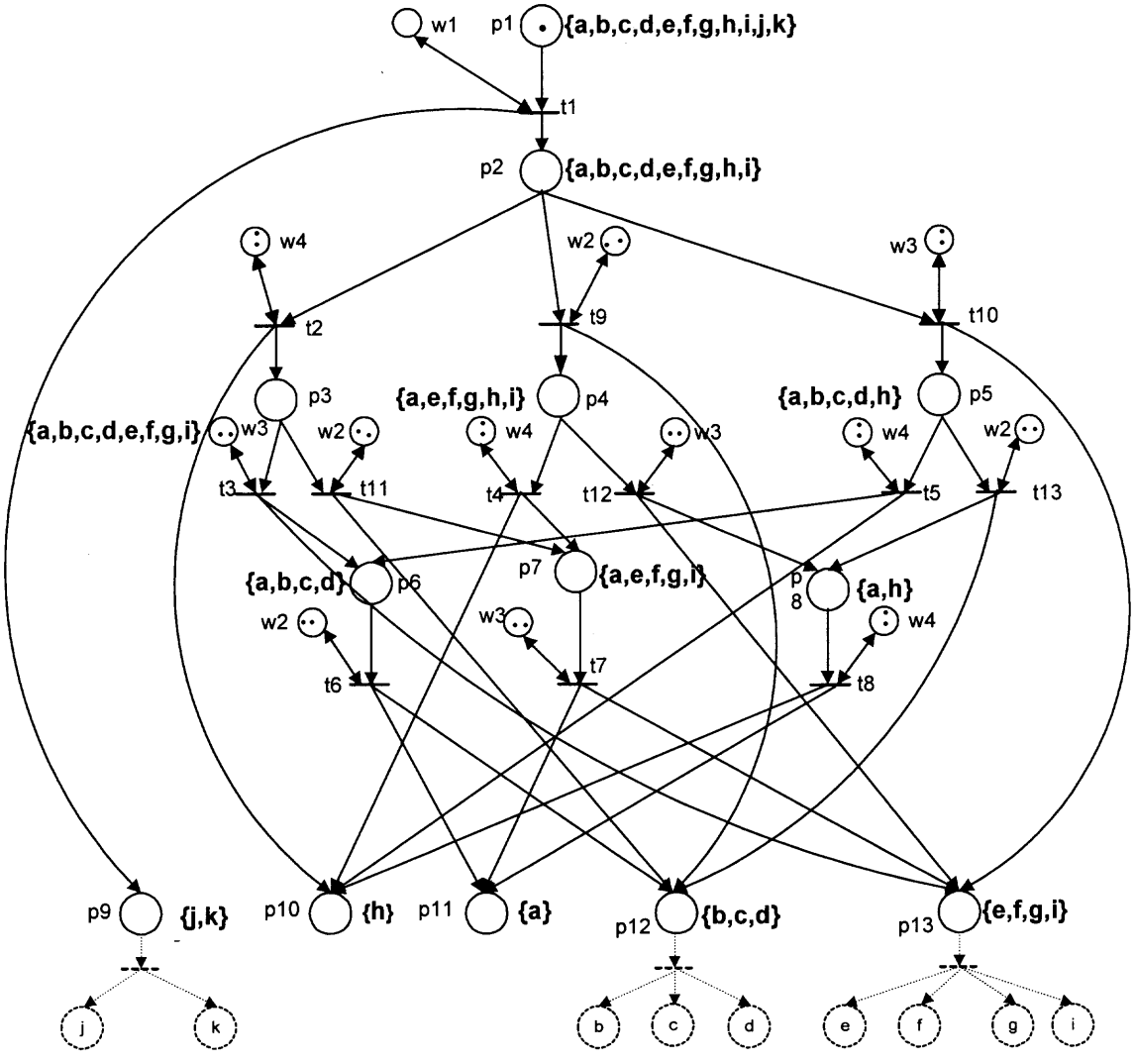


Fig. 5.3: A simple example of PPN for a personal computer

Due to the uncertainty of disassembly, used products with the same model probably require different disassembly methods. Thus, the entire model is decomposed into several subnets according to jobs and workstations, which are stated as follows:

- The disassembly of a discarded product is called a job. n jobs to be processed are denoted by $J = \{J_1, J_2, \ldots, J_n\}$.
- Each job has a sequence of tasks. An operation is to process a task by one machine in a workstation. O_{ijkz} represents the jth task of the ith job being performed by the zth machine in the kth workstation.

A dynamic model for each workstation is denoted by Scheduling Petri net (SPN). SPN_{ik} represents that the ith job being processed in the kth workstation.

Definition 5.4 A Scheduling Petri net SPN_{ik} is defined as a seven-tuple Petri net: $SPN_{ik} = (P, T, I, O, M, \Gamma, \Lambda)$ where:

- $P = R \cup S$ where the set of resource places R represents the machines and the set of status places S corresponds to job status.
  (1) $\exists p \in S, {}^{\bullet}p = \Phi$. This place is usually denoted by $p_{ik}^{initial}$ or p_1, which represents the initial status for the ith job in the kth workstation.
  (2) $\exists p \in S, p^{\bullet} = \Phi$. This place is usually denoted by p_{ik}^{final} corresponding to the final status for the ith job in the kth workstation.
- $\forall t \in T$, there is one and only one resource place $p \in R$ and $I(p, t) = O(p, t)$.
- $\forall t \in T$, there is one and only one pair of status places $p_1, p_2 \in S$ such that $t \in p_1^{\bullet}$, and $t \in {}^{\bullet}p_2$.
- Initial marking $m_0(p) = 1, \forall p \in R \cup \{ p_{ik}^{initial} \}$ and $m_0(p) = 0\ \forall p \in S - \{ p_{ik}^{initial} \}$.
- Final marking $m_f(p) = 1, \forall p \in R \cup \{ p_{ik}^{final} \}$, and $m_f(p) = 0\ \forall p \in S - \{ p_{ik}^{final} \}$.
- $\Gamma: T \rightarrow R^+$ is a processing time function associated with each transition.

- Λ: R $\rightarrow$ R^+ is a delay time function associated with a resource place, which represents the waiting time for the resource to handle a disassembly process of the current subassembly. It is updated according to the sensing and execution results of the corresponding disassembly operation performed by the Supervision Unit.

Fig. 5.4 and Table 5.2 give an example SPN_{31}, which shows the Petri net model for the third job in the first workstation. For convenience, places and transitions are also numbered into sequences as shown on their left sides.

These extended Petri nets models are used to facilitate the disassembly planning and machine scheduling during the Primary and the Secondary Inspection, which are elaborated in the Section 5.3.

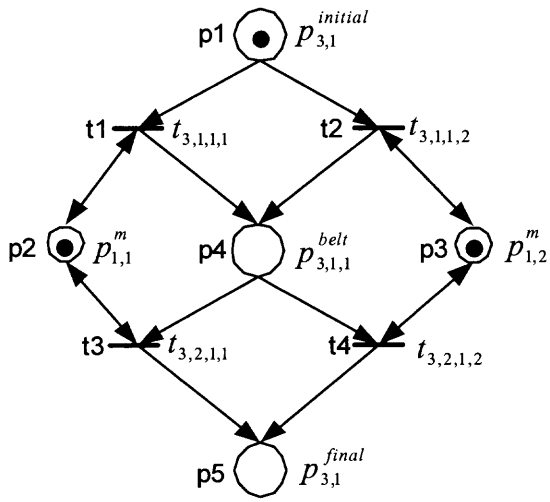


Fig. 5.4: SPN_{31} for a personal computer

5.3 Disassembly Process Planning and Machine Scheduling

Zussman and Zhou (1998) define that the disassembly process plan is to find the order of disassembly operations with the maximal EOL value. In an IFDS, both the EOL value

and resource scheduling are considered, which can be implemented by two-level planning in Primary and Secondary Inspection Units. Fig. 5.5 summaries the overall logic for this approach.

Table 5.2 Explanation for typical places and transitions in *SPN*

	Explanation
$p_{i,k}^{initial}$	the initial status place for the ith job in the kth workstation representing the beginning of disassembly in the workstation.
$p_{i,j}^{m}$	resource place for the jth machine in the ith workstation
$p_{i,k,j}^{belt}$	the intermediate status place of the ith job in the kth workstation after the jth process
$p_{i,k}^{final}$	the final place for the ith job in the kth workstation representing the end of disassembly in the workstation.
$t_{i,j,k,z}$	Representing operation $O_{i,j,k,z}$

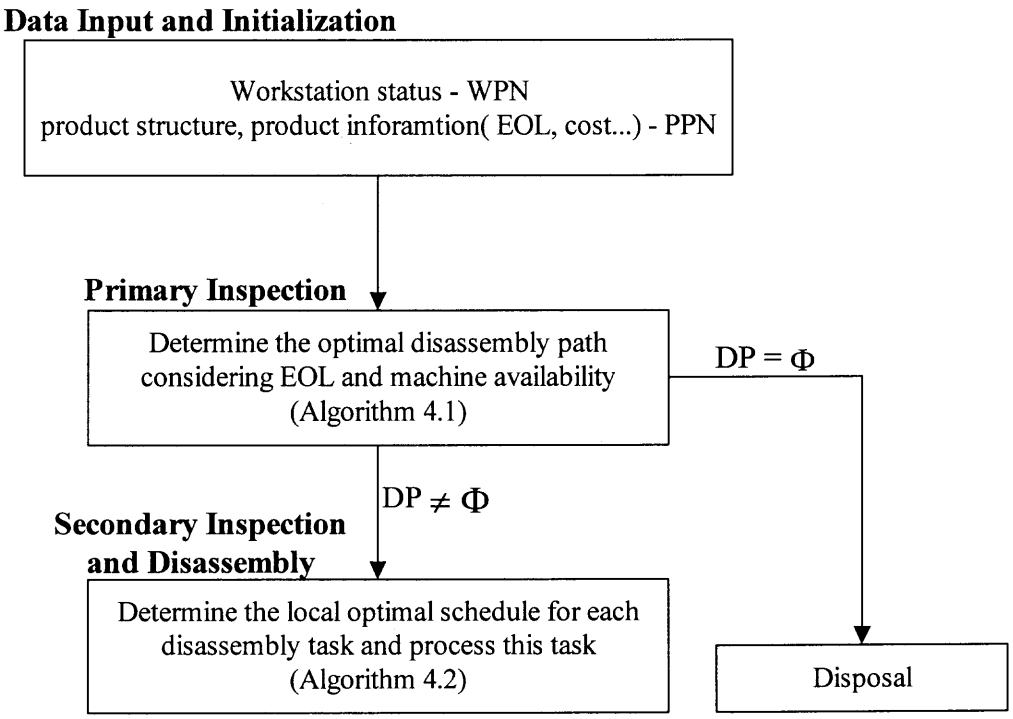


Fig. 5.5: Logic of disassembly planning and machine scheduling

5.3.1 Planning Algorithm

In an IFDS, workstations facilitate the implementation of disassembly tasks. In a PPN, a workstation place is introduced as an input to a transition, which means that a transition cannot fire until there is at least one machine available in the corresponding workstation to handle it. Considering the capacity of each workstation, not only are tokens used to represent the number of machines idle ($M(w_i)$) in the ith workstation, but also a release time r is assigned to each w_i. During the initialization, r is set to zero, which means that there is at least one machine available in each workstation. During the operation, Workstation Controller monitors the status of workstations and tracks their tokens. Once a used product is released into the facility, Workstation Controller checks the current status of WPN. If all machines in the ith workstation are occupied, i.e., all tokens in w_i^a go to w_i^b in WPN, Workstation Controller calculates the release time for the first available machine and informs the Primary Inspection Unit to update the corresponding r. This r-value together with other defined function in Definition 5.3 is used to decide the local optimal plan.

In a PPN, a place $p \in Q$ with multiple output transitions means that there are various disassembly methods available. To choose the best one, disassembly values of a place and a transition have to be defined. Due to the different units for EOL value ($), cost value ($) and time delay (hour), the unit exchange is needed.

Definition 5.5 The cost value for the time delay caused by a workstation w is denoted as $\sigma(w)$, and

$$\sigma(w) = r(w) \times B \tag{5.1}$$

where: r(w) –time delay for workstation w, and

B – exchange rate between cost and time delay, e.g., R = \$20/hour.

Definition 5.6 The disassembly values of place p and transition t are denoted as d(p) and d(t), respectively, and calculated in a recursive manner using the below equations.

(1) $\forall p \in Q', d(p) = \pi(p)$

(2) $\forall t \in T, d(t) = \sum_{q \in t^{\bullet}} d(q) - h(t)$

(3) if $p \in Q - Q', d(p) = \max\{\pi(p), \max_{t \in p^{\bullet}}(d(t))\}$

After calculating all the disassembly values, the EOL value of each place $p \in Q$ is compared with that of its corresponding transitions. $\forall t \in p^{\bullet}$ if $d(t) < \pi(p)$, i.e., t's disassembly value is lower than its input place p's EOL value, t is pruned from a PPN.

With the information of EOL values of a product, subassemblies and components, the cost values of disassembly methods, and costs for time delays of workstations, an optimal disassembly path of a product can be obtained by Algorithm 5.1 shown below where reman-value is due to [Zussman and Zhou, 1998].

Algorithm 5.1 Disassembly Plan

Step 1: Apply Procedure **reman_value** to generate d(x), $x \in Q \cup T$ in a PPN.

Step 2: Set $Z = \{p_1\}$ (assuming p_1 is the root node for every incoming product).

Step 3: Set $DP = \Phi$, $C = \Phi$, and $H = \Phi$ (DP represents final disassembly path, C represents the last components after disassembly and H is the set of workstations the product will go through).

Step 4: While ($Z \neq \Phi$) do:

For each "new" node p in Z:

Apply Procedure **Optimal_Path** to get the best disassembly method at each place, i.e., T(p), $\forall p \in Q\text{-}Q'$;

if $T(p) \neq \Phi$

Select the first transition t in T(p), $DP = DP \cup \{t\}$;

$H = H \cup \{{}^{\bullet}t \cap W\}$;

Update time delay vector r: $r(w_i) = \max\{r(w_i)-\lambda(t), 0\}$, i=1, 2,…, n;

$Z = Z \cup \{t^{\bullet}\}$;

else $C = C \cup \{p\}$

$Z = Z\text{-}\{p\}$;

End

Procedure reman_value (PPN) [Zussman and Zhou, 1998]

For $p \in Q'$, set $d(p) = \pi(p)$;

Set $L = Q\text{-}Q'$;

While $(L \neq \Phi)$ do:

Find a place $p \in L$ such that $\forall q \in (p^{\bullet})^{\bullet}$, $q \notin L$, i.e., d(q) is known.

Calculate:

$$d(t) = \sum_{q \in t^{\bullet}} d(q) - h(t);$$

$$d(p) = \max\{\pi(p), \max_{t \in p^{\bullet}} d(t)\}$$

Let $L = L\text{-}\{p\}$, i.e., remove p from L;

End

Procedure Optimal_Path (p)

Set $T(p) = \Phi$;

if $\pi(p) < d(p)$

$\forall t \in p^{\bullet}$ and $d(t) \geq \pi(p)$, then $T(p) = T(p) \cup \{t\}$;

Order transitions in T(p):

Apply equation 5.1 and transfer time delay vector r into cost value vector σ;

Calculate decision values $\delta(t) = i$ if $d(t)- \sigma(g)$ is the ith smallest in $\{d(t)- \sigma(g), g \in t^{\bullet} \cap W, t \in T(p)\}$;

Order transitions in T(p) in a decreasing order of their decision values.

End

The difference between the proposed algorithm and the original one in [Zussman and Zhou, 1998] is that the proposed one considers machine availability and waiting time needed for the disassembly plan. Thus, it is more suitable for realistic applications. However, since the time delay function r is introduced and dynamically updated when a used product is released into the IFDS facility and does not account the future products entering the IFDS, the DP obtained through Algorithm 5.1 is locally optimal instead of globally optimal as the original one did. Note that the original one [Zussman and Zhou, 1998] only guarantees the optimality for a single product but not a series of products arriving one after another. From the complexity point of view, two algorithms use the heuristic search and run exponentially in terms of number of components in a product. The detailed proof was presented in [Zussman and Zhou, 1998].

5.3.2 Scheduling Algorithm

After a product or subassembly enters a workstation, it goes through a secondary inspection. In this area, an SPN is developed to model machines and use a token to represent the availability of a machine. During an operation, Workstation Controller

always monitors the status of machines and tracks the tokens representing them. Once the station is occupied, a waiting time for this machine is associated with the corresponding token and updated as time passes. Then machines are assigned based on their speeds and availability. Algorithm 5.2 shown below fulfills this task and obtains the execution time to perform the disassembly process for the ith job in the kth workstation denoted as γ_{ik}. The notation η_i is the total execution time for the disassembly of the ith job.

Algorithm 5.2 Scheduling and Execution (SPN_{ik})

Step 1: Set $V = \{p_1\}$ and $\gamma_{ik} = 0$;

Step 2: While ($V \neq \Phi$) do:

Find a $t_1 \in V^{\bullet}$ such that $\Gamma(t_1) + \Lambda(q_1)$, $q_1 \in (t^{\bullet}_1) \cap R$ is the smallest in $\{\Gamma(t) + \Lambda(q), q \in (t^{\bullet}) \cap R, t \in V^{\bullet}\}$, and $\gamma_{ik} = \gamma_{ik} + \Gamma(t_1) + \Lambda(q_1)$;

Update waiting time vector Λ: $\forall p \in R$, $\Lambda(p) = \max\{(\Lambda(p)-\Gamma(t_1)-\Lambda(q_1)), 0\}$;

Set $V = \{p, p \in (t^{\bullet}) \cap S\}$

End

From Algorithm 5.2, the total disassembly time for the ith job is

$$\eta_i = \sum \gamma_{ik} \quad (\text{the } k^{th} \text{ workstation} \in H) \tag{5.2}$$

Theorem 5.1 The computational complexity of Algorithm 5.2 is *O(nm)*, where n and m denote the number of tasks and machines in SPN_{ik}, respectively.

Proof: The complexity of both steps in Algorithm 5.2 is analyzed as follows:

(1) The time complexity for Step 1 is a constant; and

(2) Since each machine may process each task, Step 2 goes through m choices for each task of a job and assigns one of machines to handle this task based on a machine's speed and availability. Thus, the computational complexity for this step is *O(nm)*.

Q. E. D.

The algorithm guarantees that each disassembly task for the ith job is processed as quickly as possible based on the present state. There is no guarantee that it is globally optimal since it cannot take into account the future tasks entering the workstations.

5.4 Case Study

To fully understand the above concepts and algorithms, the disassembly of a batch of obsolete personal computers in an example system is used to demonstrate the disassembly process planning and scheduling in both Primary Inspection and Secondary Inspection Units. Subsection 5.4.1 gives the implementation method; Subsection 5.4.2 presents the simulation results.

5.4.1 Implementation Method

To optimize the EOL value and system throughput for each incoming product, implementation steps are proposed as follows:

1. Construct a WPN given the present workstations' condition;
2. Construct a PPN for an incoming product given the product information and all feasible disassembly choices;
3. Collect and associate all the data with places and transitions in the PPN;
4. Apply Algorithm 4.1 to the Primary Inspection for this product;

5. Construct an SPN for each task of the product being processed;
6. Collect and associate all the data with places and transitions in the SPN; and
7. Integrate Algorithm 4.2 with computers and controllers in the IFDS.

When a discarded product comes to the IFDS facility and waits for its processes, the first four steps are executed. Based on the results of the Primary Inspection, the rest of steps are performed when each task of this used product reaches a workstation in its process plan.

A discarded personal computer consists of eleven main components. Its layout without its case and cables is given in Fig. 5.6, which shows how nine components are connected to each other.

First, check the workstations' condition and model a WPN, where dynamically updated markings represent the number of idle and busy machines.

Second, to construct a PPN for an incoming personal computer, start with the entire product labeled by p_1. Identify all possible ways to disassemble it into the next-level subassemblies or components. For instance, at p_1, only one method is identified and represented by t_1, which means that other components can be disassembled only after opening the case and taking out all cables. This process is repeated for each subassembly until no subassembly left. Then workstation places are added. Finally the complete PPN is obtained as shown in Fig. 5.3. The dotted arrows show the parallel disassemblies in a real-time system. For clarity, workstation places w_i ($i = 1$ to 4) are shown more than once in Fig. 5.3.

Then, the relevant information associated with places and transitions is collected. The detailed data is given in the next subsection.

Based on the result of Primary Inspection, an SPN is modeled for each job. To construct SPN_{ij}, start with the initial status of the ith job in the jth workstation labeled by p_1. Identify all feasible ways to process its disassembly into the next-level subassembly/components. Repeat this procedure until the final status of the ith job in the jth workstation is reached. Taking SPN_{31} as an example, at p_1, there are two machines that can process the disassembly of a case and cables, which are represented by two transitions as shown in Fig. 5.4. The detailed data associated with places and transitions are presented in Subsection 5.4.2.

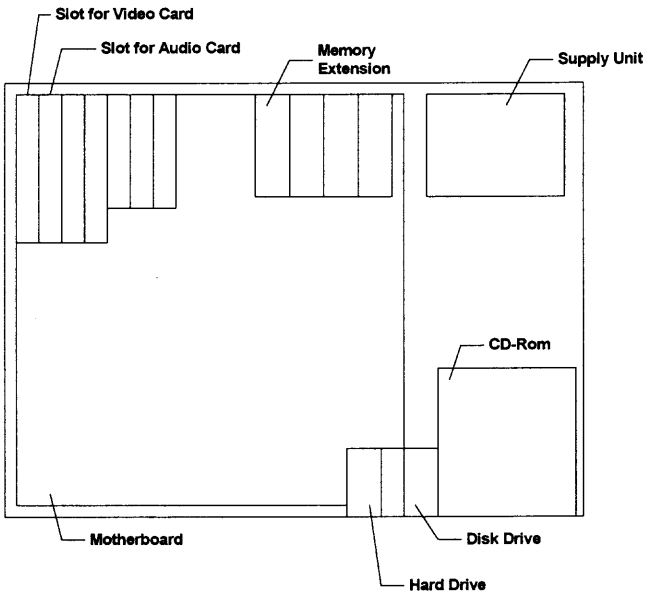


Fig. 5.6: Layout of a personal computer

Finally, the proposed algorithms are implemented into the IFDS. This work has developed a simulation module as shown in Fig. 5.7, which runs under Windows 98/NT operating system written in Visual C++. The software architecture includes the following modules:

- Database
- Process Planner

- Controller

In IFDS, there are various parameters. Some of them are the input data of the Process Planner (i.e., π, h, r); others are calculated during the operations (i.e., δ). In the real-time situation, there are two kinds of input data, one of which is decided according to the previous benchmark experience and the market price (i.e., EOL value π, cost value h); the other is dynamically obtained through sensors (i.e., r). During the implementation of the simulation software, the input data are created as follows and used to both cases:

- Products come into the IFDS with an arrival interval, which has a uniform distribution between [0.01, 1] minute;
- The process time for tasks is randomly chosen, which also has a uniform distribution between [0.1, 5] minutes;
- In a PPN, the EOL value of each place and the cost value of each transition are both randomly chosen with a uniform distribution, which have the range [-2, 1] and [0.001, 0.1], respectively; and
- Users can define the number of workstations, machines and tasks in each workstation. In our example, the IFDS has four workstations in total. Each workstation is equipped with two machines.

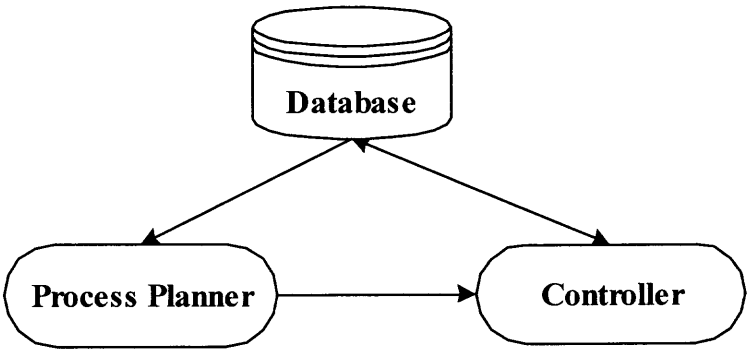


Fig. 5.7: The System Software Module

A. Database

Database provides the relevant information of each product and workstation, including EOL and cost values in PPN, process times in SPN, and idle/busy markings in WPN. The data is created through Data Input project, which has five files to populate their corresponding data into Database. They are Arrival.cpp, Cost.cpp, Eol.cpp, initialization.cpp, and Process.cpp as shown in Fig. 5.8.

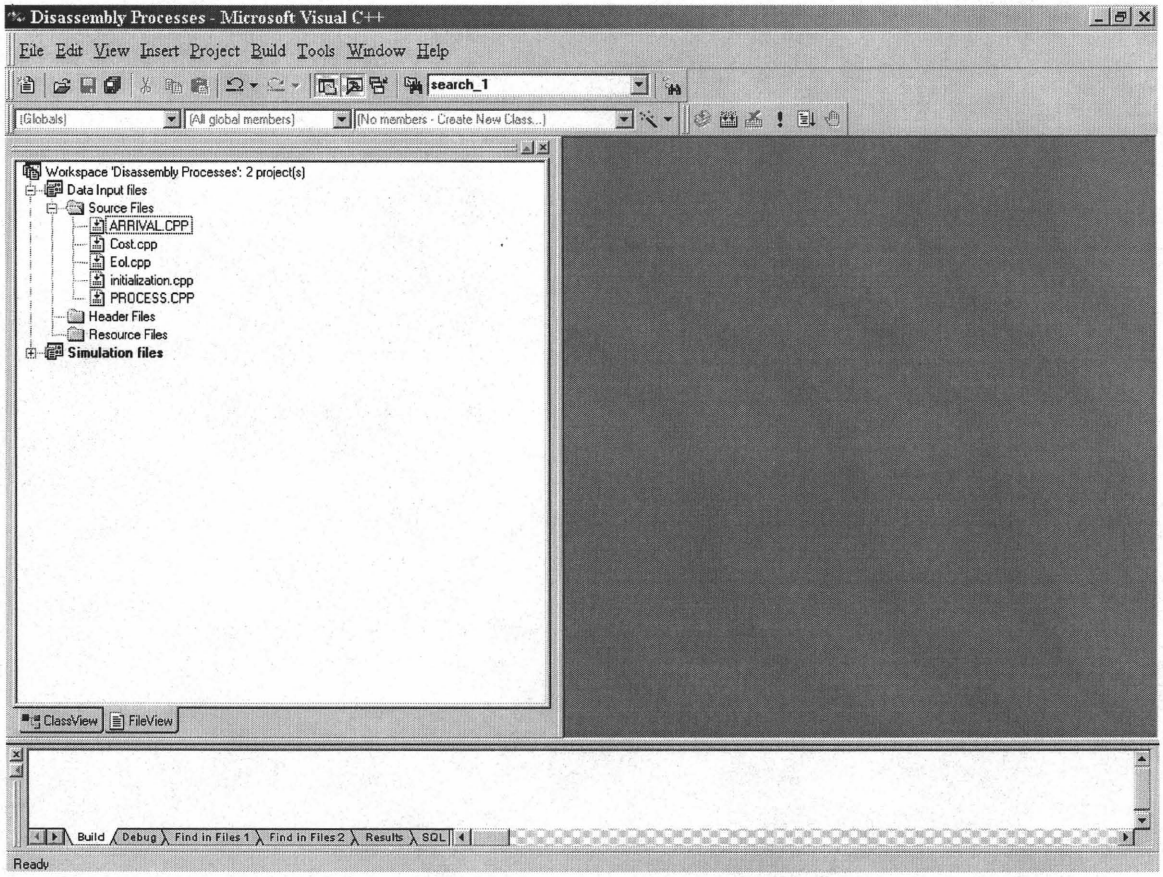


Fig. 5.8: Disassembly Processes Simulation software workspace

B. Process Planner and Controller

Based on the information from Database, Process Planner obtains the local optimal DP,

decides machines assignments, and transfers it to Controller. Then, Controller assigns jobs to machines according to the results from Process Planner, updates workstation information, and feedbacks it to Database. These tasks are fulfilled in Simulation project, which has ten files shown in Fig. 5.9. They are explained as follows:

- InitialValue: retrieve data from Database;
- InitialPetrinets: model PPN for a product;
- Optimization: implement Algorithm 5.1 to obtain the disassembly sequence of a used product;

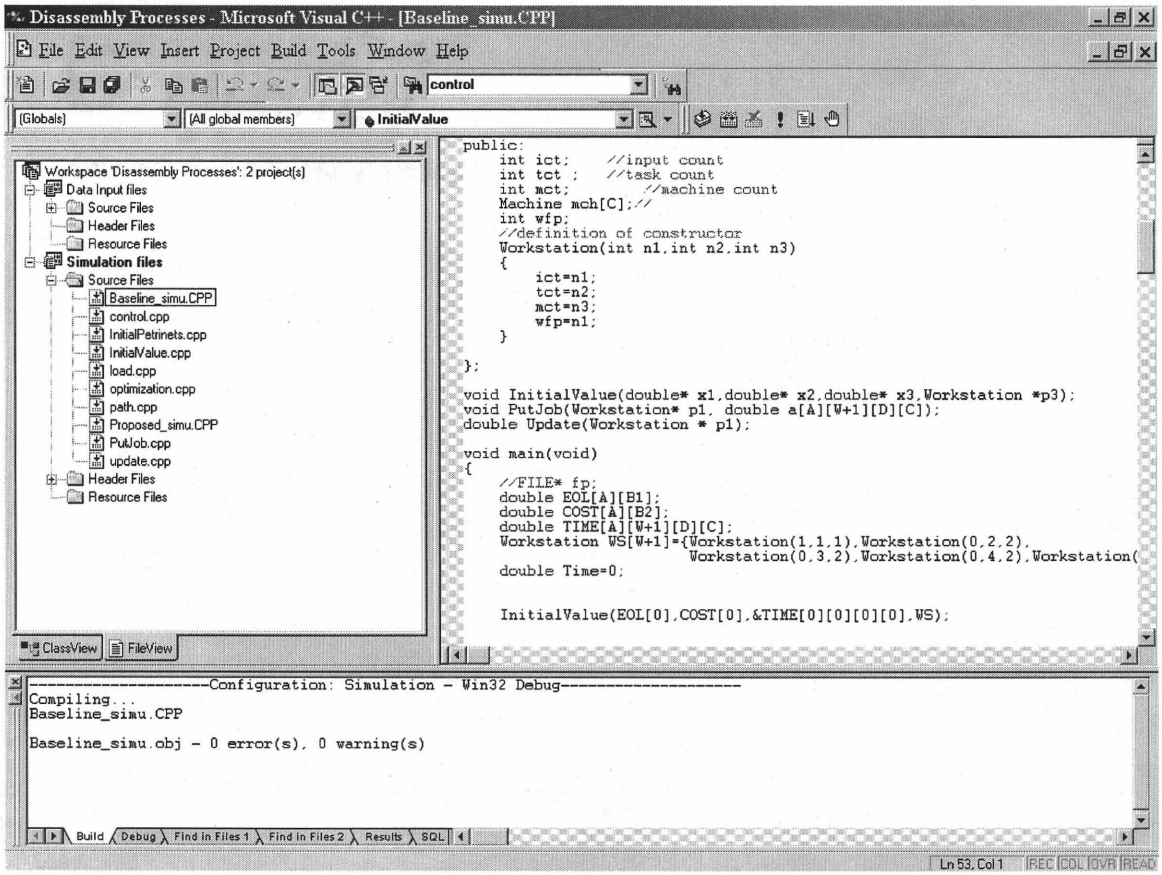


Fig. 5.9: Simulation Project

- PutJob/Load: load jobs into the IFDS/workstations;

- Control: implement Algorithm 5.2 to assign jobs to machines in each workstation; and
- Update: update workstation information.
- Path: check the disassembly path of a used product and mark the finished tasks
- Baseline_simu/Proposed_simu: according to the information user input, monitor the disassembly processes of a batch of products to be simulated.

5.4.2 Simulation Results

In this section, a value gain in the disassembly plan (DP) is defined first. Two cases are considered in the simulation software. In the baseline case, a coming personal computer is completely disassembled through Workstations 1 to 4, no matter what condition it has. The disassembly sequence is fixed. A job immediately takes an idle machine based on their speeds when both are available. In the proposed case, each coming personal computer is first inspected based on its potential diassembility and system condition. Then it is disassembled following its optimal sequence derived by Algorithms 5.1 and 5.2.

To show the statistically significant difference between two cases, the same statistical technique is used as before. The execution of Algorithms 5.1 and 5.2 is presented. Two alternatives in real-time procedures are compared through a set of experiments.

Definition 5.7: The value gain in a disassembly plan DP f(DP) is:

$$f(DP) = \sum_{p \in C} \pi(p) - \sum_{t \in DP} h(t) \tag{5.3}$$

A. Illustration of Disassembly Sequences

In this experiment, J_3 is chosen as an example. The detailed execution of Algorithm 4.1 is presented as follows:

(1) When it comes into the IFDS, its PPN and WPN are modeled and corresponding data are collected from Database:

π = (-1.40, -0.90, -0.80, -0.40, -0.30, 0.10, 0.20, 0.50, 0.70, 0.70, 0.80, 0.90, 0.90)

h = (0.013, 0.079, 0.068, 0.062, 0.057, 0.089, 0.025, 0.067, 0.071, 0.001, 0.026, 0.019, 0.027)

λ = (9.44, 3.82, 12.59, 3.82, 3.82, 16.87, 12.59, 3.82, 16.87, 12.59, 16.87, 12.59, 16.87). This implies that the maximum disassembly times are 9.44, 16.87, 12.59 and 3.82 for the processes in Workstations 1 to 4, respectively. Time unit is minute.

(2) The workstation information is transferred from the WPN:

r = (2.73, 0, 0, 0) where 2.73 indicates the earliest machine in W_1 will be ready after 2.73 minutes.

(3) Calculate the disassembly values of places and transitions using procedure **reman_value**:

d(p) = (3.89, 3.20, 2.54, 2.31, 2.30, 1.61, 1.67, 1.43, 0.70, 0.70, 0.80, 0.90, 0.90).

d(t) = (3.89, 3.17, 2.44, 2.31, 2.25, 1.61, 1.67, 1.43, 3.14, 3.20, 2.54, 2.31, 2.30).

(4) Apply procedure **Optimal_Path** to get the best disassembly method at each place:

$T(p_1) = \{t_1\}$, $T(p_2) = \{t_{10}, t_2, t_9\}$, $T(p_3) = \{t_{11}, t_3\}$, …, $T(p_7) = \{t_7\}$, $T(p_8) = \{t_8\}$

(5) The sequence of operations for J_3 is obtained as $\{t_1, t_{10}, t_{13}, t_8\}$, $U = \{w_1, w_3, w_2, w_4\}$, and f(DP) = \$3.89

Note that the EOL value of the entire product without disassembly is $\pi(p_1)$ = -\$1.4. At the end of the disassembly process, the EOL value of the personal computer increases to \$3.89. Fig. 5.10 gives the disassembly value as a function of the disassembly steps.

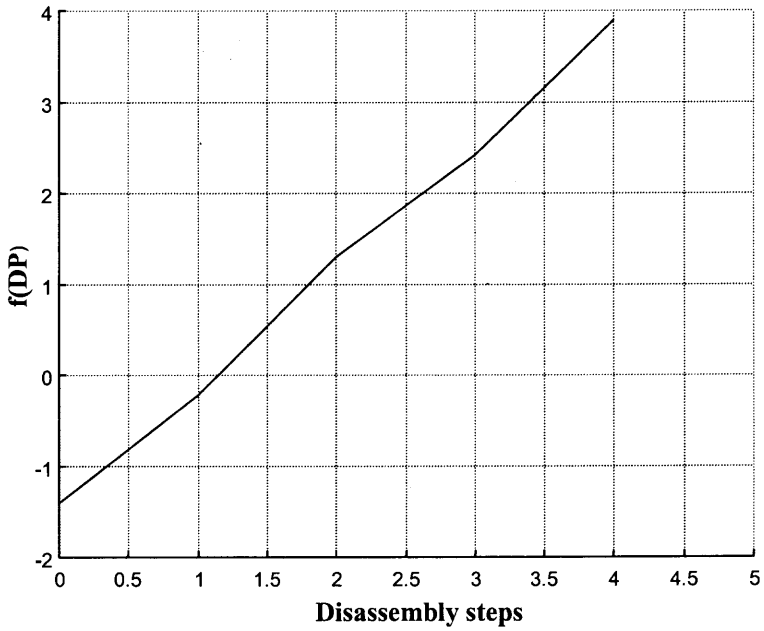


Fig. 5.10: Experimental results of J_3 process plan

Following the decided disassembly sequence, J_3 goes through Workstations 1 and 3, then 2 and 4. Once J_3 enters the ith workstation, SPN_{3i} is modeled. The relevant information associated with places and transitions is collected. Take SPN_{31} as an example, the execution of Algorithm 4.2 is presented as follows:

(1) Collect the corresponding data for SPN_{31}:

Γ = (4.52, 3.46, 4.32, 4.92)

Λ= (1.45, 0) where 1.45 indicates Machine 1 (p_2) will be available after 1.45 minutes.

(2) The local optimal execution plan is decided and the execution time for the third job in the first workstation is obtained: θ_{31} = 7.78 minutes.

B. Comparison Results

In this set of experiments, as the batch size of personal computers changes from 100 to 10000, the average disassembly time and value gain are obtained for and compared between baseline and proposed methods. They are shown in Figs. 5.11 and 5.12. Using the data in Tables 5.3 and 5.4, 95% confidence intervals for the true mean differences in average disassembly time and value gain of products are calculated as follows (The value of $g_{\alpha/2,(n-1)} = g_{0.0025,5} = 2.57$ is obtained from t-distribution table [Banks *et al.*, 2000]):

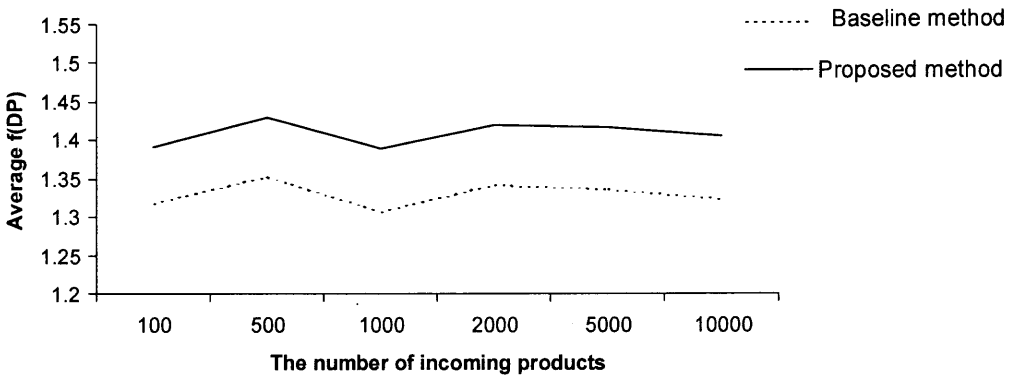


Fig. 5.11: Value gain for 10000 personal computers through the proposed and the baseline methods

A 95% confidence interval for average disassembly time is given by:

$$1.111 \pm (2.57)0.029$$

or

$$1.036 \leq \theta_1 - \theta_2 \leq 1.186$$

A 95% confidence interval for value gain of products is given by:

$$-0.078 \pm (2.57)0.002$$

or

$$-0.083 \leq \theta_1 - \theta_2 \leq -0.073$$

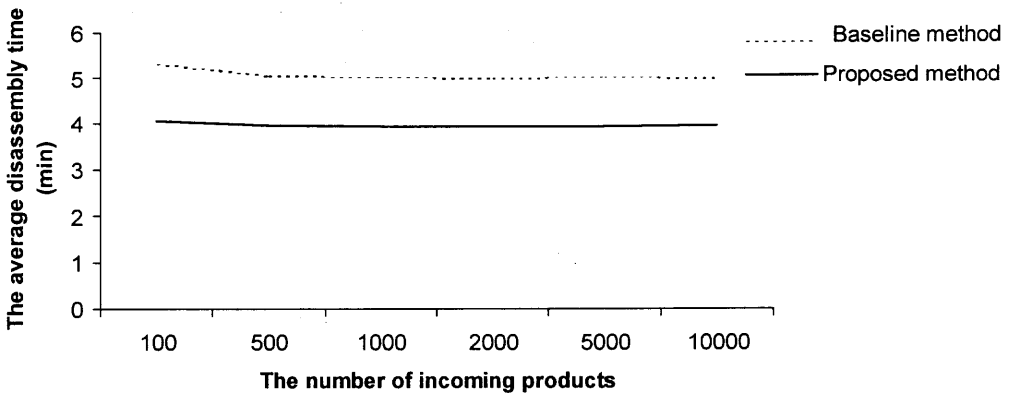


Fig. 5.12: Disassembly time for 10000 personal computers through the proposed and the baseline methods

The 95% confidence interval for average disassembly time lies completely above zero, which provides strong evidence that $\theta_1 - \theta_2 > 0$ — that is, the proposed method is better than the conventional one, because its average disassembly time is smaller. Another convincing evidence that the proposed methodology is better, is the hypothesis $\theta_1 - \theta_2 < 0$ for the value gain of products, which shows that the EOL values of products are increased in the proposed case.

The simulation output analysis represents a significant improvement in terms of system throughput and overall EOL values for the proposed methodology. The experiments with input date that have truncated normal distribution are also tested and the similar results are obtained.

5.5 Summary

Demanufacture of products for component and material recovery is an emerging field of research. This thesis addresses algorithmic issues to design and implementation of the integrated flexible demanufacturing system (IFDS). Three Petri net models (WPN, PPN,

SPN) are designed and implemented in real-time. To deal with the unique features in disassembly planning, the EOL value function and cost value function are introduced to places and transitions in a PPN, respectively. To incorporate the time delay caused by availability of machines in workstations, the delay time function associated with each workstation place is introduced in a PPN. The information of each workstation place comes from a WPN. In a SPN, processing time and delay time functions assigned to transitions are used to deal with machine assignment in each workstation.

Table 5.3 The comparison of the average disassembly time

# of products	Average disassembly time (min)		Observed Difference
	Baseline method	Proposed method	
100	5.332	4.072	1.26
500	5.068	3.954	1.114
1000	5.037	3.945	1.092
2000	5.007	3.932	1.075
5000	5.020	3.949	1.071
10000	5.010	3.958	1.052
Sample mean	5.079	3.968	1.111
Sample variance	0.016	0.006	0.005
Standard error			0.029

Table 5.4 The comparison of the value gain

# of products	f(DP)		Observed Difference
	Baseline method	Proposed method	
100	1.319	1.391	-0.072
500	1.353	1.429	-0.076
1000	1.308	1.389	-0.081
2000	1.342	1.420	-0.078
5000	1.336	1.417	-0.081
10000	1.323	1.405	-0.082
Sample mean	1.330	1.4085	-0.078
Sample variance	0.0008	0.0003	0.00002
Standard error			0.002

This work successfully overcomes a deficiency in [Zussman and Zhou, 1998, 1999] that ignores real-time resource capacity and availability, thereby making the proposed

methodology more applicable to real industrial settings. The main benefit of this methodology is that a reliable and environmentally friendly execution of robotic disassembly task sequences is guaranteed through modeling, planning and demanufacturing control.

CHAPTER 6

A SYSTEMATIC APPROACH TO DISASSEMBLY LINE DESIGN

The increasing importance of a product's relationship and effects on the environment has prompted active research in demanufacturing systems. A disassembly process, in which the obsolete, discarded, and/or faulty products are taken apart, is a critical element of a demanufacturing system. Its optimal design and efficient operation can generate positive financial and environmental impact on parts/subassemblies to be reused, materials to be recycled, and amount of waste to be disposed. Current disassembly processes are generally manual and labor intensive. Their dynamic configuration and real-time operation proves a technically challenging research issue in order to optimize the use of resources (labor, money and time). This is especially true when these processes are subject to resource breakdown, maintenance, performance degrade and other exceptions. In the previous chapter, an integrated approach to disassembly planning and demanufacturing operations is presented. However, the system efficiency issue remains open when it is subject to a changing environment. Although Gungor and Gupta (1999b) proposed an approach to disassembly line (DL) balancing problem, more complicated systems with variant types of incoming used products and multiple demand sources are not considered. Motivated by the success of a virtual production line design illustrated in Chapter 3, this chapter presents a systematic approach to optimal design and efficient operation of demanufacturing systems. This is done by considering the previously mentioned factors and unique characteristics of disassembly, e.g., disassembly

termination goals being not necessarily fixed. An innovative algorithm is then developed to facilitate the disassembly line design and optimization. It can allow one to dynamically configure a large system into many disassembly lines based on system status and demanding sources while guarantees the line balance and efficiency.

The rest of this chapter is organized as follows: Section 6.1 states the problem for the current disassembly system; Section 6.2 presents the methodology for the disassembly line design; and Section 6.3 gives an example to demonstrate this method.

6.1 Problem Statement

Disassembly is a process of systematic removal of desirable constituents from an assembly while ensuring that there is no impairment of the parts due to the process [Brenna, 1994]. Since the changes of products during the utilization phase result in uncertainty about product structure and components' conditions, disassembly automation is extremely difficult. Considering the recycling techniques and market price, it is necessary to obtain the good tradeoff between the benefit of disassembly processes and the resource requirement. However, many current disassembly systems are generally manual and labor intensive. No matter what condition a used product has, it goes through a fixed process sequence to reach a predetermined disassembly level. When a new demand is introduced, the disassembly line cannot process it until the previous demand is met. It is clear that the disassembly process is not always profitable since the cost of disassembly may exceed the market and environmental benefit, and the disassembly system is inflexible to handle varying demands. Therefore, to improve the system agility and benefit, this thesis develops algorithms to facilitate the disassembly line (DL) design

and operations. Considering overall benefit and line balance, the system inspects each incoming product to test its potential disassembility, and configures a DL on the fly based on the system status, the information of its corresponding demand, and the slack time of DLs in the system.

6.2 Disassembly Line Design

A DL is organized as a sequence of workstations, each with one or more machines/workers. Since there is uncontrolled variability in terms of product type and condition for products received, preliminary inspection is necessary to test reusability and obtain the corresponding disassembly path for each product. In the proposed disassembly system, the off-line inspection is processed using the DPN proposed in Chapter 5. The batches of products with the same disassembly path and the final disassembled parts are collected in the same input inventory. Once a new demand comes, a DL is configured and dedicated to disassemble batches of products from the corresponding storage area based on the inspection results, system status, and demand information. During the operation, the by-products are stored into different output inventory. As soon as the demand is satisfied, a DL can be revised or dismissed and then the groups of equipment/resources are reassigned to other DLs to speed them up or form a new DL.

This chapter makes the following assumptions:

- The supply of discarded or faulty products to be disassembled is unlimited.
- The discarded or faulty products are inspected when they enter the system to gain the EOL value of their components and disassembly costs.

- The transportation time and cost from one workstation to the other are negligible compared with processing time and cost.
- One or more machines/workers is assigned to each workstation, which may has different capacity and efficiency to process a task.
- Setup time for changing types of tasks at each machine/worker is incorporated as a part of processing time.

The key point of a DL design is not only to obtain the maximum economic benefits from the disassembly, but also to maintain balance between system capacity and workload.

To gain the maximum economic benefits, the preliminary inspection is first conducted off-line, where the end-of-line (EOL) value and disassembly costs are considered. Then, discarded products to be disassembled are stored into an inventory with the same value gain, which is associated with a disassembly path and defined in Chapter 5. Since the disassembly paths are different, the same type of products may be stored in different inventory, and the exact quantity of parts in them may vary. Thus, it is very important to select an inventory with the higher value gain and the total number of products to be disassembled is smaller. Following this line of the thoughts, priority ξ_i is introduced to the ith input inventory and decided as:

$$\xi_i = x \text{ if } F_i/N_w \text{ is the } x^{th} \text{ greatest value in } \{F_i/N_w,\ N_w = \max_{j-1,2,\ldots m_d} \{(N_{jw}-O_j)/R_{ij}\},\ i \in V\},$$

where

- F_i is the value gain in the disassembly path for the products in the ith input inventory;
- N_w is the total number of products to be disassembled in the wth DL;

- O_i is the number of parts in the ith output inventory;
- R_{ij} is the number of the jth type of final parts for each product in the ith input inventory;
- N_{jw} the total number of the jth type of parts in the wth DL; and
- V is the set of input inventory.

Once a new demand arrives, the system chooses the inventory with the highest priority to supply the products to be disassembled. Then, to improve the system throughput as well as keep a line balanced, the system configures a new DL with the fixed supply (i.e., the ith input inventory) based on the system status. It is clear that the average value gain for this DL is F_i. As stated in Chapter 5, the EOL value function and cost value function are introduced to deal with the unique features in disassembly planning. Thus, the fixed supply not only affords used products to be disassembled, but also decides the disassembly sequence for them. With the inspection results, the configuration of a DL looks similar with that of a VPL. Therefore, Algorithm 3.1 is modified as follows and used for DL design.

Algorithm 6.1:

1. Check each DL in the system, i.e., the wth DL and assume that the supply of products for the wth DL is the xth input inventory. If it has the by-product that is the final part (i.e., the yth part) in the $(u+1)^{th}$ demand, update the information of the $(u+1)^{th}$ demand $N_{y(u+1)}$, that is $N_{y(u+1)} = N_{y(u+1)} - \underline{a}_w * (max\{Due(u+1), Due(w)\} - EST(w))s_w * R_{xy}$. If $N_{(u+1))} = max\{N_{y(u+1)}, y=1, 2, \ldots, m_{(u+1)}\} = 0$, exit Algorithm 6.1.

2. For each inventory in V, calculate its priority. Order the inventory in V with decreasing priority values. Choose the first inventory in V as the supply of products (i.e., the ith input inventory).

3. Set $X = E$, the set of DLs with earliness, and assume that the jth workstation in a DL requires the kth class of machines and ϕ_k is the number of available machines of the kth class.

4. Evaluate the information of the new demand and calculate the desired speed of the $(u+1)^{th}$ DL:

$$a_{u+1} = \frac{N_{u+1} + s_{u+1} - 1}{Due(u+1) - EST(u+1)} \tag{6.1}$$

5. Evaluate the system status and calculate the maximum speed of each workstation in the $(u+1)^{th}$ DL by Eq. 3.10:

$$a_{j(u+1)} = \sum_{i=1}^{\phi_k} \frac{c_{ij(u+1)}}{\tau_{ij(u+1)}} \times \beta_{ij(u+1)}$$

6. Get the minimal speed of the $(u+1)^{th}$ DL according to Equation 3.4:

$$\underline{a}_w = Min\ \{a_{jw}, j=1, 2, \ldots, s_w\}$$

7. If $\underline{a}_{(u+1)} \geq a_{u+1}$, keep this new DL running with speed $\underline{a}_{(u+1)}$. Then, choose minimum $\varepsilon_{j(u+1)}$ using Eq. 3.1 such that $a_{j(u+1)} \geq \underline{a}_{(u+1)}$, update $\phi_k = \phi_k - \varepsilon_{j(u+1)}$, and calculate $\psi_w(t)$. If $\psi_w(t)>0$, add this new DL into set E and go to Step 11.

8. If $X \neq \Phi$, do:

   a) Select the wth DL in the set X, remove it from set X, and adjust its $\underline{a}_w$ through Eq. 3.11-3.14.

b) If Step a) succeeds, update ε_{jw}. Exclude the redundant machines from this DL and add them into idle machine pool ϕ_k;

c) If $\psi_d = 0$, remove it from the set E;

d) Return to Step 5.

9. If there is at least one machine available at each stage of the $(u+1)^{th}$ demand and $\sum_{w=1}^{u}(\psi_w - \varphi_w) - \varphi_{u+1} > 0$, configure this new DL with a slow rate $\underline{a}_{(u+1)}$, put this new line into the set D and go to Step 11.

10. Reject this new order, consider it later and exit Algorithm 6.1.

11. Calculate the disassembly time for the $(u+1)^{th}$ demand T_{u+1} using Eq. 3.15:

$$T_{u+1} = (N_{u+1} + s - 1) / \underline{a}_{(u+1)}$$

12. When the DL is finished, for the jth type of parts where $j \in C_i$ and $j \notin D_{(u+1)}$, update its corresponding output inventory $O_j = O_j + R_{ij} * N_{(u+1)}$ and exit.

It is clear that the computations for updating the demand size and sorting input inventory are trivial compared to that in Step 8. Thus, the time complexity of this algorithm is the same as that of Algorithm 3.1, which is $O(unm)$.

6.3 An Example

To better understand the above concepts and methods, a practical DL implementation of the concept of DL is studied using a simplified disassembly system. In this example, two cases are considered to process three demands, whose input data is listed in Table 6.1. In the baseline case, the system successively processes these demands according to their order date. Each time, the system is fully dedicated to disassemble a certain type of

products to satisfy the current demand. In the proposed case, whenever a demand comes, the system configures a DL on the fly based on the inventory information and system status, and adjusts the speed of workstations to maintain line balance. Finally, the system throughput and average value gains are compared and analyzed through these two cases. For simplicity, this example only considers the disassembly of personal computers. The inspection result, including final disassembled parts, their quantity and value gain, are assumed known and shown in Table 6.2 where letters a to k represent motherboard, disk drive, hard drive, CD-ROM, display adapter, audio card, network adapter, power unit, RAM, case and cables, respectively. Table 6.3 lists the workstation capacity, disassembly time and total idle machines, which are deterministic.

Table 6.1 The input data for three demands

d	EST	Due	Parts	N_{jd}
1	1/1/01	1/13/01	Hard disk	10^3
			Floppy disk	10^6
2	1/3/01	1/14/01	RAM	$2*10^6$
3	1/8/01	1/11/01	Motherboard	10^4

Table 6.2 The input inventory information

V	C_i	F_i	R_{ij}^*
1	{a-k}	1.30	
2	{b-d, h, j, k}	1.42	One RAM missing
3	{e-k}	1.39	

* The number of parts is the same as the original PC while no special mention for R_{ij}
a – motherboard; b – disk drive; c – hard drive; d – CD-ROM; e – display adapter; f – audio card; g – network adapter; h – power unit; i – RAM; j – case; k – cables.

Table 6.3 The input data for workstations

Workstation	1	2	3	4
c_{ijw}	1	1	1	1
$\tau_{ij1\ (min)}$	3	5		3
$\tau_{ij2\ (min)}$	3	5	4	3
$\tau_{ij3\ (min)}$	3		4	3
ϕ_k	90	100	100	90

From Fig 6.1 and Table 6.4, it is clear that using the proposed methods, the disassembly system can concurrently handle multiple demands and adjust the speed of these DLs according to their condition and the system capacity. Since the preliminary inspection is introduced, the system makes a better tradeoff between disassembly cost and market price. Thus, the system throughput and value gain increase by 28.5% and 7.5%, respectively.

Table 6. 4 The comparison of the system throughputs and value gains

		Baseline case	Proposed case
F	1st DL	1.32	1.42
	2nd DL	1.29	1.39
	3rd DL	1.30	1.32
g		58.3	74.9

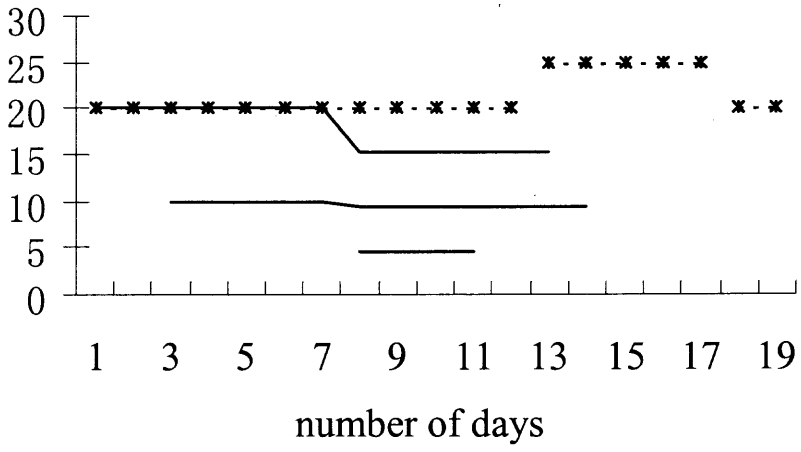


Fig. 6.1: DLs' speed for baseline (dotted lines) and proposed cases (solid lines)

6.4 Summary

Disassembly of discarded/faulty products for components and material recovery is an emerging field of research. This chapter proposed a systematic approach for the DL

design. Preliminary inspection is conducted off-line to test the used products' potential disassembility. Keeping batches of products of same disassembly families as the supply of a DL provides faster disassembly and lower setup and material handling time, which optimizes the system productivity. Based on the proposed method, a DL can be configured and formed on the fly by using the inventory information and system status. From the example, the approach is found to be effective in increasing the system throughput and maintaining a better tradeoff between disassembly cost and market price.

CHAPTER 7

CONCLUSIONS AND FUTURE RESEARCH

7.1 Contributions and Limitations

Implementation of successful CIM systems has taken place in many enterprises. A central technical challenge is the modeling, planning, and scheduling of automated factory systems, in particular, flexible manufacturing systems that are often most precious resources [Zhou and Venkatesh, 1998]. Flexible and agile manufacturing, reducing the product life cycle, improving the product quality and reliability, increasing the productivity, and lowering the product cost, has earned much attention in the current industries. It makes it possible for manufacturers to respond to the market change instantly. Due to the several features of semiconductor manufacturing systems, such as complex product flow, high investment, diverse equipment characteristics, etc., the implementations of MES integrated with shop floor automation are very limited. A part of this research focuses on modeling and scheduling of flexible semiconductor manufacturing systems.

Increased public awareness of environmental issues has led to growing concern about the end-of-life options for discarded products and material. This concern is becoming more acute with effecting and pending legislations in a number of countries worldwide, the increasing shortage of landfill space, expanding regulation of waste disposal, as well as a growing consumer preference for "green" product characteristics. These environmental drivers identify a need for developing a new approach towards modeling

and planning of products dismantling and demanufacturing processes, which is another focus of our research.

After a review of recent methodology and technology development activities in the manufacturing and demanufacturing areas, it is clear that there is a need for semiconductor manufacturing companies to implement highly responsive and efficient shop floor automation to minimize production costs and increase productivity while improve both quality and delivery time performance. Base on this observation, a discrete-event driven VPL design methodology is proposed. Using this approach, an entire semiconductor manufacturing system can be configured into many production lines according to several factors (i.e., due data, system status), each of which is used to process its corresponding work-order. A criterion allowing machine sharing among VPLs is also used to deal with imbalance caused by malfunctions or exceptions. Furthermore, a simulation software package is developed, and a case study performed by applying the methodology to a simplified back-end semiconductor line. The case study shows that the proposed methodology is effective to increase the system throughput and decrease the tardiness, since the system can concurrently handle multiple work-orders and correctly estimate the completion time of each work-order.

Since some semiconductor-manufacturing equipment may become unreliable, regardless of its age, unscheduled equipment downtime is an important factor that makes effective production control a complex task. Based on this consideration, an adaptive algorithm is presented to design reconfigurable semiconductor manufacturing systems with maintenance and failure. Different machine models are used to calculate machines' speeds in their corresponding conditions (i.e., machine with periodical maintenance).

Moreover, a priority is introduced to idle machines, which facilitates the decision-making. The efficiency of the approach is also demonstrated through three cases running in a back-end semiconductor line.

To address the environmental impacts from the discarded products and materials, this thesis has also investigated the demanufacturing processes from a system perspective. An integrated approach to disassembly planning and demanufacturing operations is proposed. EOL value, cost value and delay time value are embedded into three Petri nets models, which are designed and implemented to derive the disassembly sequence of a used product and machines scheduling. To show the advantage of the methodology, a simulation software package is developed, and a benchmark study is also performed through the disassembly of a batch of obsolete personal computers in the simulation. This approach provides a systematic way to guarantee a reliable and environmentally friendly execution of robotic disassembly sequences.

Finally, motivated by the success of a virtual production line design in manufacturing, a systematic approach to optimal design and efficient operation of demanufacturing systems is proposed. Based on the previous method to derive the disassembly sequence of a used product, preliminary inspection is conducted off-line to test the used products' potential disassembility. The efficiency of the methodology is demonstrated through a simplified disassembly system.

The contributions of this dissertation are summarized as follows:

(1) The methodologies are provided to reconfigure the shop floor on the fly while the impact on the shop floor is minimized, demonstrating agile and reconfigurable manufacturing/demanufacturing in terms of flexibility and responsiveness;

(2) Environmental impacts, among other considerations, are integrated into disassembly processes, which make it possible for the reliable and environmentally friendly execution of demanufacturing; and

(3) The proposed methodologies are illustrated through case studies to reduce the cycle time and improve the system throughput, thereby increasing the competitiveness of a company in the global market.

This research has the following limitations:

(1) To strengthen the theoretical validity of the developed methodologies, this work starts with a relatively simpler model, where setup time is assumed to be included in process time. However, such handling may not be practical in an industrial environment;

(2) The labor cost, facility cost, and environment impacts are all addressed in this research. Thus, the major aims are to increase the system throughput, lower the cycle time, and decrease the environmental damage in the proposed scheduling methodologies. On the other hand, inventory cost is a significant issue for a manufacturer, from the economic point of view. However, it is overlooked in the proposed methodologies;

(3) Due to the time limit, this research does not consider the operational failure and machine maintenance for the demanufacturing systems. However, these variability factors need to be analyzed;

(4) Batch processing machines are special facilities in semiconductor manufacturing. For the simplicity, they are all treated as ordinary machines in our study. However,

it may not reflect the practical situations accurately and their modeling and scheduling problems need to be addressed in real-time applications; and

(5) Much professional knowledge and expertise are required to apply the methodologies successfully to an industrial environment, and extensive data collection is needed to obtain the valid data.

7.2 Future Research

Further intensive research is required in many aspects. Some directions are listed as follows:

(1) Theoretical Research

This research mainly focuses on providing system level solutions to computer-integrated manufacturing and demanufacturing processes. Detailed theoretical research is needed in the following aspects:

- As mentioned in our limitations, the assumption that the setup time is included in process time may not be true or appropriate in the real-time practice. Moreover, setup time is a very important factor in production planning and shop floor control. In order to use these methodologies successfully in practical applications, efficient optimization methods with setup time consideration deserve research effort;
- The earliness increases the inventory cost that is an important economic factor. Therefore, it is necessary to use the inventory control techniques to optimize the system productivity;
- Although the work focuses on semiconductor manufacturing, developments of VPLs are not restricted to that. The advancements anticipated with this research are

needed to be applied with little or no refinement to many other manufacturing /demanufacturing environments. The approaches to model machines' maintenance and failure in VPL design can be introduced to demanufacturing lines; and

- Scheduling batch processing machines is an interesting class of scheduling problems in semiconductor manufacturing. A mathematical tool is needed to model and analyze its special characteristics. Petri nets and Queuing network are recommended.

(2) Software Development

Simulation software for VPL design and demanufacturing operations has been developed using MFC and SQL and Visual C++, respectively (Chapters 3 and 5). Using this software, manufacturers can track real-time work-orders and check whether the work-order can be dealt with in the production facility within the delivery date a customer wants before making a commitment for on-time delivery.

More effort is required to further implement the methodologies into the software packages. Some visualization functions are needed to make these packages more user-friendly.

(3) Industrial Case Study

Although several case studies are performed in the research, they are limited to laboratory tests. Collaboration with industrial firms in real manufacturing and demanufacturing applications would be valuable to putting the theoretical research results into practical uses.

Significant academic and industrial effort has been focused on design and implementation approaches for manufacturing and demanufacturing systems [Gungor and

Gupta, 1999b; Fargher *et al.*, 1994; Kim *et al.*, 1998]. There is a need to carry out performance analysis and benchmark studies of all these approaches by using a large number of industrial cases. Such studies will be very helpful to facilitate engineers in selecting most appropriate design and implementation methodologies and using them in particular industrial applications.

REFERENCES

[1] Adams, J., Balas, E. and Zawack, D., "The shifting bottleneck procedure for job-shop scheduling," *Management Science,* Vol. 34, No. 3, pp. 391-341, 1988.

[2] Ahmadi, J. H., Ahmadi, R. H., Dasu, S. and Tang, C. S., "Batching and scheduling jobs on batch and discrete processors," *Operations Research,* Vol. 40, pp. 750-763, 1992.

[3] Arai, E. and Iwata, K., "CAD system with product assembly/disassembly planning function," *Robotics and Computer-Integrated Manufacturing*, 10(1/2), pp. 41-48, 1993.

[4] Ashai, Z. and Gadh, R., *Computer-Aided Design-for-Disassembly: A Non-Destructive Approach*, Technical Report, UW-Madsion, Mechanical Engineering Dept., Sept. 1994.

[5] Bai, X., Srivatsan, N., and Gershwin, S. B., "Hierarchical real-time scheduling of a semiconductor fabrication facility," *Proc. Ninth IEEE Int. Electronics Manufacturing Technology Symp.,* Washington DC, Oct. 1990, pp. 312-317.

[6] Baldwin, D. F., Abell, T. E., Lui, M. C. M., De Fazio, T. L., and Whitney, D. E., "An Integrated Computer Aid for Generating and Evaluating Assembly Sequences for Mechanical Product," *IEEE Trans. on Rob. & Aut.*, Vol. 7, No. 1, Feb. pp. 78-94, 1991.

[7] Baliga, J., "MES and CIM: at the center of productivity," *Semiconductor International*, Vol. 8, pp. 104-112, 1998.

[8] Banks, J., Carson, J. S. II, Nelson, B. L. and Nicol, D. M., *Discrete-Event System Simulation*, Prentice Hall, 2000.

[9] Bartholdi, J. J. and Eisenstein, D. D., "A production line that balances itself," *Operation Research*, 44(1), pp. 21-34, 1996.

[10] Beasley, D. and Martin, R. R., "Disassembly Sequences for Objects Built from Unit Cubes," *Computer Aided Design,* Vol. 25, No. 12, pp. 751-761, Dec. 1993.

[11] Berry, B., "Environmental Pressures Drive Auto Design," *Iron Age*, 7, 21, 1991.

[12] Bitran, G. R., Haas, E. A., and Hax, A. C., " Hierarchical production planning: A single stage system," *Operations Research,* Vol. 29, No. 4, pp. 717-743, 1981.

[13] Bitran, G. R., Haas, E. A., and Hax, A. C., "Hierarchical production planning: A two-stage system," *Operations Research,* Vol. 30, No. 2, pp. 232-251, 1982.

[14] Brennan, L., Gupta, S. M. and Taleb, K. N., "Operations planning issues in an assembly/disassembly environment," *Int. J. of Operations and Production Management*, Vol. 14, No. 9, pp. 57-67, 1994.

[15] Cao, T. H., and Sanderson, A. C., "Task Sequence Planning using Fuzzy Petri Nets," *IEEE Trans. on Systems Man and Cybernetics*, Vol. 25, No. 5, pp. 755-768, May 1995.

[16] Cao, T. H., and Sanderson, A. C., "AND/OR Net Representation for Robotic Task Sequence Planning," *IEEE Trans. on Systems Man and Cybernetics-Part C: Application and Review,* Vol. 28, No. 2, pp. 204-218, May 1998.

[17] Chandra, V. and Gupta, S., "An analysis of a last-station-bottleneck semiconductor packaging line," *Faculty of Management,* McGill University, 1992.

[18] Chen, S. F., Oliver, J. H., Chou, S. Y., and Chen, L. L., "Parallel Disassembly by Onion Peeling," *Journal of Mechanical Design*, 119(2), pp. 267-274, 1997.

[19] Cheng, F. T., Shen, E., Deng, J. Y. and Nguyen, K., "Development of a distributed object-oriented system framework for the computer-integrated manufacturing execution system," *Proc. of 1998 IEEE Int. Conf. on Rob. & Aut.,* Leuven, Belgium, May 1998, pp. 2116-2121.

[20] Chynoweth, E., and Rotman, L., "Germany is in the fast lane as recycling gains speed," *Chemical Week,* 152, 18,1993.

[21] Connors, D. P., Feigin, G. E. and Yao, D. D., "A Queueing network model for semiconductor manufacturing," *IEEE Trans. on Semiconductor Manufacturing,* Vol. 9, No. 3, pp. 412-427, August 1996.

[22] Crognale, G., "The next generation of environmental management," *Proc. of the IEEE Int. Symp. on Electronics and the Environment,* Dallas, TX, May 6-8, 1996, pp. 323-327.

[23] Dutta, D., "Automatic disassembly and Total Ordering in Three Dimensions," *Journal of Engineering for Industry, Trans. of the ASME*, Vol. 117, No. 2, pp. 207-213, May 1991.

[24] Dutta, D. and Woo, T. C., "Algorithm for Multiple Disassembly and Parallel Assemblies," *Journal of Engineering for Industry, Trans. of the ASME*, Vol. 117, pp. 102-109, Feb. 1995.

[25] Fargher, H. E., Kilgore, M. A., Kline, P. J. and Smith, R. A., "A planner and scheduler for semiconductor manufacturing," *IEEE Trans. on Semiconductor Manufacturing,* Vol. 7, No. 2, pp.117-126, May 1994.

[26] Flapper, S. D. P., One-way or reusable distribution items, TUE/BDK/LBS/95-04, Graduation School of Industrial Engineering and Mang. Science, Eindhoven University of Technology, Eindhoven, Netherland, 1995.

[27] Flapper, S. D. P., "Logistic aspects of reuse: An overview," *Proc. of the First Int. Working Seminar on Reuse,* Eindhoven, The Netherlands, 1996, pp. 109-118.

[28] Fujiwara, F., Suzuki, S., and Okuma, S., "Integration of Planning and Scheduling for Mechanical Assembly Based on Timed Petri Net," *Proc. of Japan-US Symp. on Flexible Automation*, Osaka, Japan, 1998, pp. 347-354.

[29] Furuhjelm, J., Yasuda, Y. and Trankell, R., "Recycling of telecommunication products in Europe, Japan and USA," *Proc. of the IEEE Int. Symp. on Electronics and the Environment,* San Francisco, CA, May 8-10, 2000, pp. 143-148.

[30] Gadh, R., Srinivasan, H., Nuggehalli, S. and Figueroa, R., "Virtual Disassembly – A Software Tool for Developing Product Disassembly and Maintenance Systems," *1998 Proc. Annual Reliability and Maintainability Symposium*, Anaheim, California USA, 1998 Jan. 19-22, pp. 120-125.

[31] Gatenby, D. A. and Foo, G., "Design for X (DFX): key to competitive, profitable products," *AT & T Technical Journal,* 69(3), pp. 2-15, 1990.

[32] Glassey, C. R. and Resende, R. G., "Close-loop job release control for VLSI circuit manufacturing," *IEEE Trans. on Semiconductor Manufacturing,* Vol. 1, No. 1, pp. 36-46, 1988.

[33] Goh, T. N. and Varaprasad, N., "A statistical methodology for the analysis of the life-cycle of reusable containers," *IIE Transactions*, 18, pp. 42-47, 1986.

[34] Golovin, J. J., "A total framework for semiconductor production planning and scheduling," *Solid State Technology,* pp. 167 -170, May 1986.

[35] Grenchus, E., Keene, R., and Nobs, C., "Demanufacturing of information technology equipment," *Proc. of IEEE Int. Symp. on Electronics and the Environment*, May 5-7, 1997, pp. 157-160.

[36] Grogan, P., "Auto Wreckers," *Biocycle*, 35,86, 1994.

[37] Groover, M. P., *Automation Production Systems, and Computer-Aided Manufacturing,* Prentice-Hall, New Jersey, 1980.

[38] Groover, M. P., *Fundamentals of Modern Manufacturing Materials, Processes, And Systems,* Prentice Hall, New Jersey, 1996.

[39] Gungor, A. and Gupta, S.M., "Disassembly Sequence Planning for Products with Defective Parts in Product Recovery," *Computers and Industrial Engineering*, v35, pp. 161-164, 1998a.

[40] Gungor, A. and Gupta, S.M., "Disassembly Sequence Planning for Complete Disassembly in Product Recovery," *Proc. of the Northeast Decision*

Sciences Institute Conference, Boston, MA, March 25-27, 1998b, pp. 250-252.

[41] Gungor, A. and Gupta, S.M., "Issues in Environmentally Conscious Manufacturing and Product Recovery: A Survey," *Computer and Industrial Engineering (An International Journal)*, Vol. 36, n 4, pp. 811-853, 1999a.

[42] Gungor, A. and Gupta, S. M., "A systematic solution approach to the disassembly line balancing problem," *Proc. of the Int. Conf. on Computers and Industrial Engineering*, 1999b, pp. 75-78.

[43] Gungor, A. and Gupta, S. M., "Disassembly line balancing," *Proc. of the Northeast Decision Science Institute*, 1999, pp. 193-195.

[44] Gupta, S. M., and Taleb, K., "Scheduling disassembly," *Int. Journal of Production Research*, Vol. 32, pp. 1857-1866, 1994.

[45] Hadavi, K., "Design and implementation of a real-time scheduling system at Siemens," *Proc. of AUTOFACT'94 Conf.,* Detroit, MI, Nov. 13-17, 1994, pp. 1-4.

[46] Harrington, J., *Computer Integrated Manufacturing*, Industrial Press, New York, 1973.

[47] Ham, H. S., "Job route selection model for workload balancing between workstations in flexible flow line," *Production Planning & Control*, Vol. 7, No. 4, pp. 430-438, 1996.

[48] Homem de Mello L.S. and Sanderson A. C., "AND/OR Graph Representation of Assembly Plans," *IEEE Trans. on Rob. & Aut.*, Vol.6, No.2, pp. 188-199, 1990.

[49] Homem de Mello L.S. and Sanderson A. C., "Representation of Mechanical Assembly Sequences," *IEEE Trans. on Rob. & Aut.*, Vol.7, No.2, pp. 211-217, 1991a.

[50] Homem de Mello L.S. and Sanderson A. C., "A Correct and Complete Algorithm for the Generation of Mechanical Assembly Sequences," *IEEE Trans. on Rob. & Aut.*, Vol. 7, No. 2, pp. 228-240, Apr. 1991b.

[51] Huang, H. H. and Wang, M. H., "Optimal Disassembly Sequence Generation using Neural Network," *Proc. of the Fourth Int. Conf. on Environmentally Conscious Design and Manufacturing*, Cleveland, Ohio, July 23-25, 1996, pp. 89-98.

[52] Ikura, Y. and Gimple, M., "Scheduling algorithms for a single batch processing machine," *Operations Research Letters,* 5, pp. 61-65, 1986.

[53] Inaba, A., Suzuki, T. and Okuma, S., "Feasibility of Disassembly Tasks Considering a Posture of a Subassembly using Genetic Algorithm," *Proc. of IEEE/ASME Int. Conf. on Advanced Intelligent Mechatronics*, Tokyo, Japan, Jun. 16-20, 1997, pp. 79-84.

[54] Jorgensen, T. M. and Andersen, A. W., " Shape Recognition System for Automatic Disassembly of TV-sets," *Proc. of IEEE Int. Conf. on Image Processing*, Vol. 2, Lausanne, Switzerland, Sept. 16-19, 1996, pp. 653-656.

[55] Kane, N. F., " Restructuring manufacturing process flows," *IEEE/CPMT Int. Electronics Manufacturing Technology Symp.*, Oct. 14-16, 1996, Austin, TX, pp. 312-316.

[56] Kelle, P. and Silver, E. A., "Purchasing policy of new containers considering the random returns of previously issued containers," *IIE Transactions*, 21(4), pp. 349-354, 1989.

[57] Kim, Y. D., Kim, J. U., Lim, S. K. and Jun, H. B., "Due-date based scheduling and control policies in a multiproduct semiconductor wafer fabrication facility," *IEEE Tran. on Semiconductor Manufacturing,* Vol. 11, No. 1, pp. 155-164, Feb. 1998.

[58] Kimemia, J. and Gershwin, S. B., "An algorithm for the computer control of a flexible manufacturing system," *IIE Trans.,* Vol. 15, No. 4, pp. 353-362, 1983.

[59] Kooi, E., Krikke, H. and Schuur, P., "Physical design of a reverse logistic network: a multi-echelon model," *Proc. of the First Int. Working Seminar on Reuse*, Eindhoven, Netherlands, Nov. 11-13,1996, pp. 205-212.

[60] Kopacek, P. and Kronreif, G., "Semi-Automated Robotized Disassembly of Personal Computers," *Proc. of IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA '96)*, Haiwii, Nov. 1996, pp. 567-572.

[61] Koren, Y., Hu, J.S., and Weber, T. W., "Impact of Manufacturing System Configuration on Performance," *Annals of the CIRP*, Vol. 47, pp. 369-372, 1998.

[62] Kotera, Y and Sato, S., "An Integrated Recycling Process for Electric Home Appliance," *Mitsubishi Electric Advance*, Vol. 80, pp. 23-26, Sept. 1997.

[63] Kroon, L. and Vrijens, G., "Returnable containers: an example of reverse logistics," *Int. J. of Physical Distribution & Logistics Management*, 25(2), pp. 56-68, 1995.

[64] Lambert, A. J. D., " Optimal Disassembly of Complex Products," *Int. J. of Production Research*, Vol. 35, No. 9, pp. 2509-2523, Sept. 1997.

[65] Lambert, A. J. D. and Splinter, M. A. N., "Reuse of sophisticated product carriers," *Proc. of the First Int. Working Seminar on Reuse*, Eindhoven, Netherlands, Nov. 11-13, 1996, pp. 213-222.

[66] Leachman, R. C., *Preliminary desing and development of a corporate-level production planning system for the semiconductor industry*, Technical Report of OR Center, University of California, Berkeley, Feb. 1986.

[67] Lee, C. Y., Uzsoy, R., and Martin-Vega, L. A., "Efficient algorithms for scheduling batch processing machines," *Operations Research,* Vol. 40, pp. 764-775, 1992.

[68] Lee, K. M. and Bailey-Vankuren, M., "Supervisory Control of an Automated Disassembly Workcell Based Blocking Topology," *Proc. of the 1997 IEEE Int. Conf. on Rob. & Aut.*, Albuquerque, New Mexico, April, 20-25, 1997, pp. 1523-1528.

[69] Lee, K. and Gadh, R., " Computer Aided Design-For-Disassembly: A Destructive Approach," *Concurrent Product and Process Engineering, ASME*, MED-Vol. 1/DE-Vol.85, pp. 237-249, 1995.

[70] Lee, K. and Gadh, R., "Computer Aided Design for Disassembly: A Destructive Approach," *IEEE Int. Symp. on Electronics and the Environment*, Dallas, Texas, May 6-8, 1996, pp. 173-178.

[71] Liao, D. Y., Chang, S. C., Pei, K. W. and Chang, C. M., "Daily scheduling for R&D semiconductor fabrication," *IEEE Tran. on Semiconductor Manufacturing,* Vol. 9, No. 4, pp. 550-561, Nov. 1996.

[72] Livingstone, S. and Sparks, L., "The new German packaging laws: effects on firms exporting to Germany," *Int. J. of Physical Distribution & Logistics Management*, 24(7), 1994, pp. 15-25.

[73] Lou, S. X. C. and Kager, P. W., "A robust production control policy for VLSI wafer fabrication," *IEEE Trans. on Semiconductor Manufacturing,* Vol. 2, No. 4, pp. 159-164, 1989.

[74] Meacham, A., Uzsoy, R. and Venkatadri, U., "Optimal Disassembly Configurations for Single and Multiple Products," *Journal of Manufacturing Systems*, 18: (5), pp. 311-322, 1999.

[75] Meier, B., "Breaking down an arms buildup: dismantling and recycling weapons," *New York Times (Late New York Edition),* pp. D1-D2, 1993.

[76] Moore, K. E., Gungor, A. and Gupta, S. M., "Petri Net Approach to Disassembly Process Planning," *Computers and Industrial Engineering*, Vol. 35, No. 1-2, pp. 165-168, 1998a.

[77] Moore, K. E., Gungor, A. and Gupta, S. M., "Disassembly Petri Net Generation in the Presence of XOR precedence Relationships," *Proc. of the IEEE Int.*

Conf. on Systems, Man, & Cybernetics, Vol. 1, San Diego, California, Oct. 11-14, 1998b, pp. 13-18.

[78] Nasr, N., "Environmentally conscious manufacturing," *Careers and the Engineer,* Spring, pp. 26-27, 1997.

[79] Nilsson, N. J., *Principles of Artificial Intelligence,* Springer-Verlag, New York, 1980.

[80] Nof, S. Y., Wilhelm, W. E. and Warnecke, H. J., *Industrial Assembly,* Chapman & Hall, London, 1997.

[81] Ouellette, R. P., Thomas, L. W., Mangold, E. C., and Cheremisinoff, P. N., *Automation Impacts On Industry*, Ann Arbor Science, Ann Arbor, 1983.

[82] O'Grady, P. J., *Controlling Automated Manufacturing Systems,* Chapman & Hall, London, 1986.

[83] Owen, J. V., "Environmentally conscious manufacturing," *Manufacturing Engineering,* 111(4), pp. 44-55, 1993.

[84] Pellichero, F., Rekiek, B., Falkenauer, E. and Delchambre, A., "Interactive tool for an optimal equipment selection," *Conf. FAIM' 99*, Tilburg, The Netherlands, June 1999, pp. 111-122.

[85] Proth, J. M. and Xie, X. L., "Cycle time of stochastic event graphs: evaluation and making optimization," *IEEE Trans. on Automatic Control,* Vol. 39, No. 7, pp. 1482-1486, 1994.

[86] Qiu, R. and Wysk, R., "Design and implementation of virtual production lines for discrete automated manufacturing systems," *The 14th Int. Federation of*

Automatic Control World Congress, Beijing, P. R. China, July 5-9, 1999, pp. 455-460.

[87] Reijnders, L., *Environmentally Improved Production Processes and Products: An Introduction,* Kluwer Academic Publishers, Dordrecht,

[88] Rekiek B. and Delchambre, A., "Ordering variants and simulation in multi-product assembly lines," *VR-Mech' 98 Conf.*, Brussels, Belgium, Nov. 1998, pp. 49-54.

[89] Rekiek, B., Fabrice, P., Lit, P. D., L'Eglise, T., Falkenauer, F. and Delchambre, A., "Balancing and resource planning for assembly lines: the gap between theory and practice," *Proc. of the CPI'99*, Tangier, Morocco, November 1999, pp. 239-248.

[90] Rekiek, B., Falkenauer, E., and Delchambre, A., "Two problem in design and operation of assembly lines: line balancing and ordering variants," *CARS & FOF conf.*, Combatore, India, Dec. 1998, pp. 243-250.

[91] Rembert, T. C., "Package deal: the European war on waste," *The Environmental magazine*, 8(3), pp. 38, 1997.

[92] Rembold, U., Nnaji, B. O. and Storr, A., *Computer Integrated Manufacturing And Engineering,* Addison Wesley, England, 1993.

[93] Rembold, U., Blume, C. and Dillmann, R., *Computer-Integrated Manufacturing Technology And Systems,* Marcel Dekker, New York, 1985.

[94] Srinivassan, H., and Gadh, R., "Complexity Reduction in Geometric Selective Disassembly Using the Wave Propagation Abstraction," *Proc. of the 1998*

IEEE Int. Conf. on Rob. & Aut., Leuven, Belgium, May16-20, 1998, pp. 1478-1483.

[95] Scott, D., "Comparative advantage through manufacturing execution systems," *IEEE/SEMI Advanced Semiconductor Manufacturing Conf.,* Cambridge, MA, Nov. 12-14, 1996, pp. 179-184.

[96] Srinivasan H., and Gadh R., "A geometric algorithm for single selective disassembly using the wave propagation abstraction," *Computer Aided Design*, Vol. 30, No. 8, pp. 603-613, 1998a.

[97] Srinivassan, H., and Gadh, R., "Complexity Reduction in Geometric Selective Disassembly Using the Wave Propagation Abstraction," *Proc. of the 1998 IEEE Int. Conf. on Rob. & Aut.*, Leuven, Belgium, May16-20, 1998b, pp. 1478-1483.

[98] Shyamsundar, H., Srinivassan, H. and Gadh, R., "Virtual de-manufacturing via virtual disassembly to design environmentally conscious products", *Int. Journal of Environmentally Conscious Design & Manufacturing*, Vol. 6, No. 1, pp. 37-50, 1997.

[99] Suzuki, T., Kanehara, T., Inaba, A. and Okuma, S., "On Algebraic and Graph Structural Properties of Assembly Petri Net," *Proc. of IEEE Int. Conf. on Rob. & Aut.*, Atlanta, Georgia, May 2-6, 1993, pp. 507-514.

[100] Subramani, A. K. and Dewhurst, P., "Automatic Generation of Product Disassembly Sequences," *CIRP Annals*, Vol. 40, NO. 1, pp. 115-118, Aug. 18-24, 1991.

[101] Tang, Y., Zhou, M. C., Zussman, E., and Caudill, R., "Disassembly modeling, planning, and application: A review," *Proc. of IEEE Int. Conf. on Rob. & Aut.*, San Francisco, CA, April 22-28, 2000, pp. 2197-3002.

[102] Thomas, J. P., Nissanke, N. and Baker, K. D., "A Hierarchical Petri Net Framework for the Representation and Analysis of Assembly," *IEEE Trans. on Rob. & Aut.*, Vol.12, No. 2, pp. 268-279, April 1994.

[103] Thierry, M., Salomon, M., Van Nunen, J., and Van Wassenhove, L., "Strategic issue in product recovery management", *California Management Review*, 37(2), pp. 114-135, 1995.

[104] Tzafestas, A.G., Anthopoulos, A., Katevas, N., and Spyropoulou, E., "Architecture and Implementation of an Autonomous Car-Disassembly System," *Systems Analysis, Modeling, Simulation*, Vol.29, pp. 129-149, 1997.

[105] Uzsoy, R., Lee, C. Y. and Martin-Vega, L. A., "A review of production planning and scheduling models in the semiconductor industry part I: system characteristics, performance evaluation and production planning," *IIE Transactions,* Vol. 24, No. 4, pp. 47-60, Sept. 1992.

[106] Uzsoy, R., Lee, C. Y. and Martin-Vega, L. A., "A review of production planning and scheduling models in the semiconductor industry part II: shop-floor control," *IIE Transactions,* Vol. 26, No. 6, pp. 44-55, Sept. 1994.

[107] Venkatesh, K., M. C. Zhou, M. Kaighobadi, and R. Caudill, "A Petri net approach to modeling and analysis of flexible factory automated systems under push

and pull paradigms," *Int. J. of Production Research*, 34(3), pp. 595-620, 1996.

[108] Vinod, B. and Altiok, T., "Approximating unreliable queuing networks under the assumption of exponentiality," *Operational Research Society,* Vol. 37, No. 3, pp. 309-316, 1986.

[109] *Visionary Manufacturing Challenges for 2020*, National academy press, Washington, D. C., 1998.

[110] Wein, W. L., "Scheduling semiconductor wafer fabrication," *IEEE Tran. on Semiconductor Manufacturing,* Vol. 1, No. 3, pp. 115-130, 1998.

[111] Xue, Y. and Gu, P., "Graph Based heuristic Approach to Automated Assembly planning," *American Society of Mechanical Engineers, Design Engineering Division*, Vol. 73, pp. 97-106, Sept. 11-14, 1994.

[112] Xue, Y. and Gu, P., "Assembly/Disassembly Sequence Planning for Life-Cycle Cost Estimation," *American Society of Mechanical Engineers, Manufacturing Engineering Division*, Vol. 2-2, pp. 935-956, Nov. 12-17, 1995.

[113] Zeid, I., Gupta, S. M. and Bardasz, T., "Case-Based Reasoning Approach to Planning for Disassembly," *Journal of Intelligent Manufacturing*, Vol. 8, No. 2, pp. 97-106, Apr. 1997.

[114] Zeid, I., Gupta, S. M. and Bardasz, T., "Analogical Problem Solving Approach to Planning for Disassembly," *American Society of Mechanical Eningeers, Design Engineering Divisi*on, Vol. 89, Nov. 17-22, pp. 79-88, 1996.

[115] Zhang, H. C. and Kuo, T. C., "A Graph-Based Disassembly Sequence Planning for EOL Product Recycling," *21st IEEE/CPMT Int. Electronics Manufacturing Technology Symp*, Austin, TX, Oct. 13-15, 1997, pp. 140-151.

[116] Zhang, H. C. and Kuo, T. C., "A Graph-Based Approach to Disassembly Model for End-of-life Product Recycling," *19th Proc. of Int. Electronics Manufacturing Technology Symp.*, Austin, TX, Oct. 14-16, 1996, pp. 247-254.

[117] Zhou, M. C., and Venkatesh, K., *Modeling, Simulation, And Control Of Flexible Manufacturing Systems-A Petri Net Approach*, World Scientific, Singapore, 1998a.

[118] Zhou, M. C. and Jeng, M. D., "Modeling, Analysis, Simulation, Scheduling, and Control of Semiconductor Manufacturing Systems, A Petrin Net Approach," *IEEE Trans. on Semiconductor Manufacturing,* Vol. 11, No. 3, pp. 333-357, 1998b.

[119] Zurawski, R. and Zhou, M. C., "Petri nets and industrial applications: a tutorial", *IEEE Trans. on Industrial Electronics*, Vol. 41, No. 6, pp. 567-583, December 1994.

[120] Zussman, E., Reiter, B. S., and Scharke, H., "Modeling and Planning of Disassembly Processes," *Proc. of the IFIP WG5.3 Int. Conf. on Life-Cycle Modeling for Innovative Products and Processes,* Berlin, Germany, Nov./Dec., 1995, pp. 221-232.

[121] Zussman, E., Zhou, M.C., and Caudill, R., "Disassembly Petri Net Approach to Modeling and Planning Disassembly Processes of Electronic Products," *Proc. of IEEE Int. Symp. on Electronic and Environment*, Oak Brook, IL, May 4-6, 1998, pp. 331-337.

[122] Zussman, E. and Zhou, M.C., "Methodology for Modeling and Adaptive Planning of Disassembly Processes," *IEEE Trans. on Rob. & Aut.*, Vol. 15, No. 1, pp. 190-194, Feb. 1999.

[123] Zussman, E. and Zhou, M.C., "Design and Implementation of an Adaptive Process Planner for Disassembly Processes," *IEEE Trans. on Rob. & Aut.*, Vol. 16, No. 2, pp. 171-179, April, 2000.