MODELING AND ANALYSIS OF PRODUCTION AND CAPACITY PLANNING CONSIDERING PROFITS, THROUGHPUTS, CYCLE TIMES, AND INVESTMENT

A Thesis
Presented to
The Academic Faculty

by

SugJe Sohn

In Partial Fulfillment
Of the Requirements for the Degree
Doctor of Philosophy in the
Industrial and Systems Engineering

Georgia Institute of Technology

July 2004

MODELING AND ANALYSIS OF PRODUCTION AND CAPACITY PLANNING CONSIDERING PROFITS, THROUGHPUTS, CYCLE TIMES, AND INVESTMENT

Approved by:

Dr. Chen Zhou, Co-Advisor

Dr. Shabbir Ahmed, Co-Advisor

Dr. Leon F. McGinnis

Dr. Ronald Billings

Dr. Jeff K. Stratman

June 30, 2004

To my wife and family

for their love

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to people, without whom this dissertation would not have been completed. First and foremost, it is a great honor to express my deep gratitude to my advisor, Dr. Chen Zhou, for his guidance, assistance, encouragement, and warm support in all steps of my doctoral program at Georgia Institute of Technology. Also, I especially thank Dr. Shabbir Ahmed, co-advisor, for his outstanding insights and hearty helps for the quality of my thesis. I would like to deliver a special gratitude to Dr. Leon McGinnis, who has supported and mentored me as Director of HiFiVE project in my entire Ph.D. track. And, I thank so much Dr. Ronald Billings and Dr. Jeff Stratman, for being the reading committee as well.

Next, I am so thankful to my colleagues and friends in laboratory and projects, professors of the classes in Industrial and Systems Engineering and Dupree College of Management, coworkers and supervisors in Samsung Electronics, and spiritual mentors and prayers in Saehan Presbyterian Church and the praise team “Sound of Zion”. It is so a great pleasure for me to have those friendly, talented, and supportive people during these years.

Most importantly, to my family in Korea and Canada, I owe a debt of gratitude for their love and support along the path of my academic pursuits. Last but not least, I would like to express my deepest love and gratitude to my wife, *Jeenah*, for her continuous support and encouragement.

It is the end of my Ph.D. endeavor. However, it is the beginning of another journey in my life with the greatest love of God.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

This research focuses on large-scale manufacturing systems having a number of stations with multiple tools and product types with different and deterministic processing steps. The objective is to determine the production quantities of multiple products and the tool requirements of each station that maximizes net profit while satisfying strategic constraints such as cycle times, required throughputs, and investment. The formulation of the problem, named OptiProfit, is a mixed-integer nonlinear programming (MINLP) with the stochastic issues addressed by mean-value analysis (MVA) and queuing network models. Observing that OptiProfit is an NP-complete, nonconvex, and nonmonotonic problem, the research develops a heuristic method, Differential Coefficient Based Search (DCBS). It also performs an upper-bound analysis and a performance comparison with six variations of Greedy Ascent Procedure (GAP) heuristics and Modified Simulated Annealing (MSA) in a number of randomized cases. An example problem based on a semiconductor manufacturing minifab is modeled as an OptiProfit problem and numerically analyzed. The proposed methodology provides a very good quality solution for the high-level design and operation of manufacturing facilities.

CHAPTER 1

INTRODUCTION

One of the critical paradigms in recent manufacturing systems is *lean* manufacturing. Lean systems typically have the benefits of lower cost, higher profitability, shorter cycle times, flexible manufacturing facilities, and fewer work-in-process (WIP). Currently, however, most manufacturing systems are not flexible enough to completely implement lean manufacturing and realize the benefits. Therefore, it is very important to consider the basic design of facilities at an early phase of system development, particularly when the application area is relatively inflexible with respect to reconfiguration and capital-intensiveness. Semiconductor manufacturing is a typical example. The management of manufacturing systems frequently seeks to understand the exact maximum production capacity whenever market demands grow. Since system behavior with respect to an increase in throughput and the amount of equipment is very difficult to evaluate, particularly in complicated cases such as reentrant systems, a critical decision is to determine when to expand and how much additional equipment to deploy. Even though slight overestimates or underestimates could significantly effect cost, managers in the semiconductor industry have been using rough approximations from simple heuristic analysis or past experience. This research is directed at describing an appropriate mathematical and engineering approach to find the optimal configuration to make a system produce the best profit. In this thesis, we refer to this problem as the *OptiProfit problem*.

Research in reentrant manufacturing systems – for example, semiconductor manufacturing systems – has been categorized based on three issues: design, operations, and control. Design issues deal with the reasonable or optimal designs of capacity, throughputs, cycle times, WIPs, and costs in the initial installation or capacity expansions of manufacturing systems. A number of studies concerned with operations issues have focused on embracing production planning and product-mix strategies (Horiguchi, 2001; Lee, 1997). Also, studies on control issues have included the topics of vehicle control, part-release scheduling, dispatching policies with priorities, etc. (Holthaus, 1997; Hwang, 1997).

The focus of this research is on the design issues. Reviewing the past literature, deterministic models for the optimization of capacity expansions have been frequently used, as discussed in Toktray (1998), whereas some recent works in this category consider the variability of reentrant manufacturing systems, as seen in Rajagopalan (2001). Also, Iwata (2002) used an approximate-cost model in semiconductor manufacturing. In addition, queuing network models have given fair solutions to several problems involving cycle time and WIP estimations, for example, in Chung (2002) and Lin (2001).

In real situations, however, the management of manufacturing systems has raised questions about the integrated effects of key performance measures and design factors such as throughput, cycle times, and cost constraints. Three typical problems faced during the initial design phase, production and capacity planning, can be summarized as follows: (i) minimizing cycle time with expansion cost and throughput constraints met;

(ii) minimizing expansion cost with cycle time and throughput constraints met; or (iii) maximizing profit with cycle time, expansion cost, and throughput constraints met.

Regarding category (i), Bard (1999) performed milestone research on capacity expansion, in which he determined the capacity or the amount of equipment that minimizes the sum of cycle times for a single product. The optimization model had constraints of cost and fixed throughput settings. Hopp (2002) approached the capacity decision for problems in category (ii) in another way, minimizing the cost with the constraints of allowable cycle times. This approach included a delicate mean-value analysis (MVA) in evaluating the parameters of the optimization model, and a simple and intuitive heuristic was proposed to solve the nonlinear constraints with integer properties.

This research proposes an approach for problems in category (iii), modeling these problems as OptiProfit problems. A systematic solution method is described for the situation where the configuration with maximal net profit achievable, while meeting the strategic constraints on expansion cost, cycle times, and required minimum throughputs, is desired. The most prominent difference of the research in this thesis from the other two (Bard, 1999; Hoop, 2002) is that this approach finds the best possible performance and capacity of systems in terms of the net profit – the ultimate goal of manufacturing systems. From a managerial perspective, it is also possible to incorporate the OptiProfit model with business strategies such as the timing and magnitude of facility expansion. Table 1.1 compares the characteristics of Bard (1999), Hopp (2002), and this thesis.

Table 1.1 Summary of the Comparison with Related Researches

	Bard (1999)	Hopp (2002)	Sohn (2004)
Objective	Minimizing the average cycle time (queuing time)	Minimizing the investment cost	Maximizing the profit (production profit against investment cost)
Decision variables	Number of tools in stations	Number of tools in stations	In-flow rate into the system Number of tools in stations
Constraints	Investment cost (Fixed throughput)	Average cycle times of products (Fixed throughput)	Investment cost Average cycle times of products Minimum throughput requirements
Property of problem	Nonlinear knapsack problem (NLIP) with nonlinear state equations	Nonlinear integer problem (NLIP) with nonlinear state equations	Mixed-integer nonlinear problem (NLMIP) with nonlinear state equations
Product mix	Single product	Multiple products	Multiple products
Eff. processing times and yield	No	Yes	Yes
Consideration on batching effect	Yes (Process batching)	Yes (Moving batching, setup batching, processing batching and unbatching)	Yes (Moving batching, setup batching, processing batching and unbatching, product-type-sensitive batching)
Cycle time evaluation base model	GI/G/M queue of Jackson network using Hybrid Queuing Network Analyzer (HQNA) (Srinivasan, 1995)	GI/G/M queue of Jackson network using Traffic Variability Equations (The queueing network analyzer) (Whitt, 1983)	GI/G/M queue of Jackson network using Traffic Variability Equations (The queueing network analyzer) (Whitt, 1983)
Solution heuristic	Four heuristics were tested, 1. Greedy Ascent Procedure (GAP) 2. Modified Ascent Procedure (MGAP) 3. Simulated Annealing Procedure (SAP) 4. Implicit numeration	Optimized Queuing Network (OQNet), a penalty-based heuristic	1. Differential Coefficient Based Search (DCBS) heuristic 2. Six variations of Modified GAPs 3. Modified Simulation Annealing
Example domain	Semiconductor	Semiconductor	Semiconductor
Result observations	Comparison of the analytic results from the four heuristics	1. Comparison of the result from OQNet heuristic with the result found by naïve enumeration in a simple example 2. Cycle time evaluation with the comparison with simulation results using ManSim™	1. Comparison of the result from DCBS heuristic with the results from other basic GAP and meta-heuristics 2. Relative optimality gap analysis using upper-bound analysis 3. Cycle time evaluation with the comparison with simulation results using Arena™

As discussed in CHAPTER 2, the base OptiProfit model is formulated and solved as a MINLP. Generally, MINLP problems are more complicated to solve compared to mixed-integer linear programming (MIP) and continuous nonlinear programming (NLP) problems. To further describe the properties of MINLP problems, difficulty in tracking arises within two major areas, the combinatorial domain and the continuous domain. As the number of integer variables increases in MINLP problems such as OptiProfit, one is faced with a large combinatorial problem, and the resulting complexity analysis characterizes the problems as NP-complete (Nemhauser, 1988). The determination of a global solution to a non-convex MINLP is also NP-hard (Murty, 1987) since even the global optimization of constrained nonlinear programming problems can be NP-hard (Pardalos, 1988) and even quadratic problems with one negative eigenvalue are NP-hard (Pardalos, 1991).

Numerous approaches and algorithms for the solution of MINLP problems such as Outer Approximation (OA), Generalized Benders Decomposition (GBD), Extended Cutting Plane (ECP), Branch and Bound (BB), and Adaptive Random Search (ARS) have been proposed in the literature (Gruhn, 1998; Floudas, 1995). Johnson and Brandeau (1999) formulate an MINLP problem for the design of shop floor material handling systems and seek solutions using the decomposition of workflow. Basically, the decomposition approach for MINLP decomposes the problem into several subproblems. Figure 1.1 shows a typical approach to the MIP subproblem and the NLP subproblem that is iteratively solved in a solution loop. The algorithms shown assume that nonlinear functions are convex to allow for convergence to a global optimum. However, very frequently the decomposed subproblems are not guaranteed to find the globally optimal

solution when the nonlinear problem is non-convex. Grznar (1994) dealt with an MINLP problem to minimize a surrogate-weighted cost of intercellular material movement under capacity and part-requirement constraints. They found that the model is neither convex nor concave in its relaxed noninteger structure, and that the emphasis in the formulation was to suggest "good" solutions rather than optimal ones.

In CHAPTER 3, more insights into the mathematical properties of OptiProfit as a MINLP problem are presented. Showing that OptiProfit is NP-complete, nonconvex, and nonmonotone, in CHAPTER 4 the research suggests a heuristic method, Differential Coefficient Based Search (DCBS), which is compared in CHAPTER 5 with other practically used heuristics and a modified meta-heuristic, Modified Simulated Annealing (MSA). Finally, in CHAPTER 6 a practical example of semiconductor manufacturing is applied to OptiProfit.

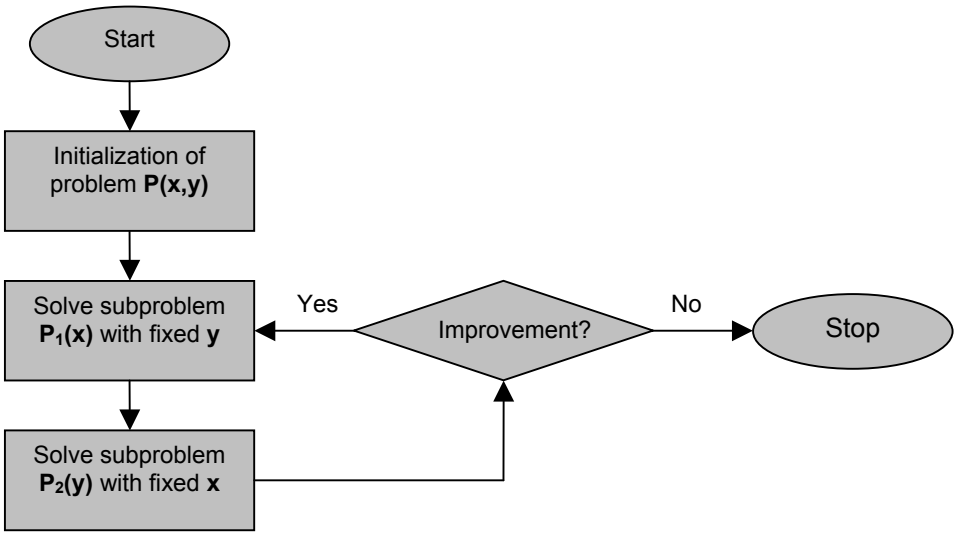


Figure 1.1 A Decomposition Method

CHAPTER 2

PROBLEM MODELING

2.1. Problem Description

The reentrant manufacturing system in the OptiProfit problem is assumed to have a number of functional areas in which homogeneous manufacturing equipment forming the *station* is logically located. Figure 2.1 depicts a simple example with two products and five stations. Physically, identical tools in a station may be deployed in different locations. Material moves along its material-flow route which is deterministic and varies according to the product type or *product*. In the design phase of the reentrant manufacturing system, due to the aggregated effects of variability in material flows it is highly complex to obtain a reasonable configuration of equipment and to predict system performance. Critical design objectives are to reduce cycle times, to increase throughputs, and to decrease resource investment costs. Considering the interactions of these objectives, the main objective of the work in this thesis is to find a mathematical formulation to obtain the *maximum profit* and the *corresponding configurations of equipment* with constrained total cost, allowable cycle times, and required throughputs.

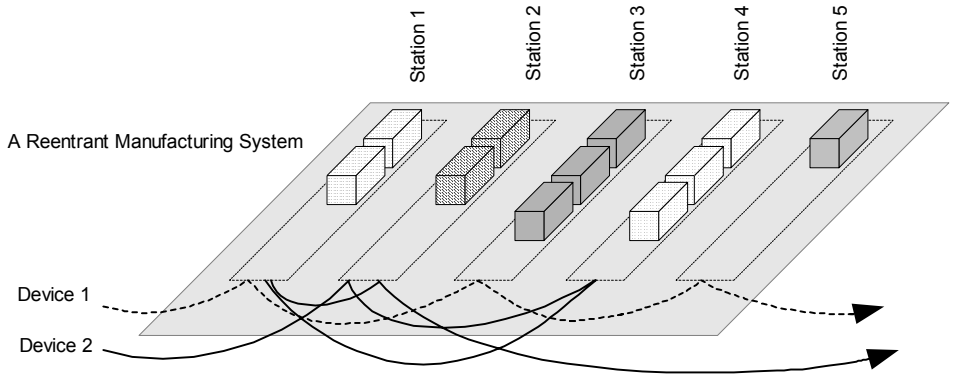


Figure 2.1 A Reentrant Manufacturing System with Multiple Products and Stations

A tradeoff in the OptiProfit problem is conceptually presented in Figure 2.2. With a given basic configuration, the increments in tool counts cause an increase in investment cost. If the cost constraints are still not violated, the increments are doable; however, they cause a deterioration in the objective function, the profit. The increase in part-releasing rates will consume the slack cycle times obtained by the increments of tool counts. The profit goes up, but the increased cycle time will be bounded by the cycle time constraints and require another increment in production capacity, i.e., additional tools in stations. In the general situation with multiple stations and products, the decision as to which station and product should be selected to increase or decrease the capacity or production is critical.

In general, with numerous stations and product types, it is impossible to visualize the alternatives graphically. Figure 2.3 gives a simple example of the 1-station and 2-product problem. The number of decision variables is three: the part-releasing rates of product 1 and 2 are $\hat{x}_1$ and $\hat{x}_2$, and the integer tool count of the station is y. The

objective function is formulated as a 3-dimensional surface in the form of $A\hat{x}_1 + B\hat{x}_2 + Cy + D = 0$, and the feasible region is discontinuous due to the integer property of y. The linear sides of each feasible region imply the bounds of minimum throughput requirements, which is proportional to the part-releasing rates with consideration for the yield rates in stations. The behavior of the nonlinear constraints on cycle times is not clearly understood due to the complexity of variability evaluations of the incoming product streams to every station.

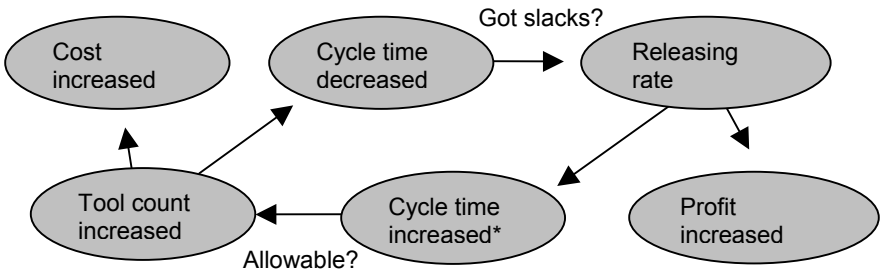


Figure 2.2 Tradeoffs of the Problem

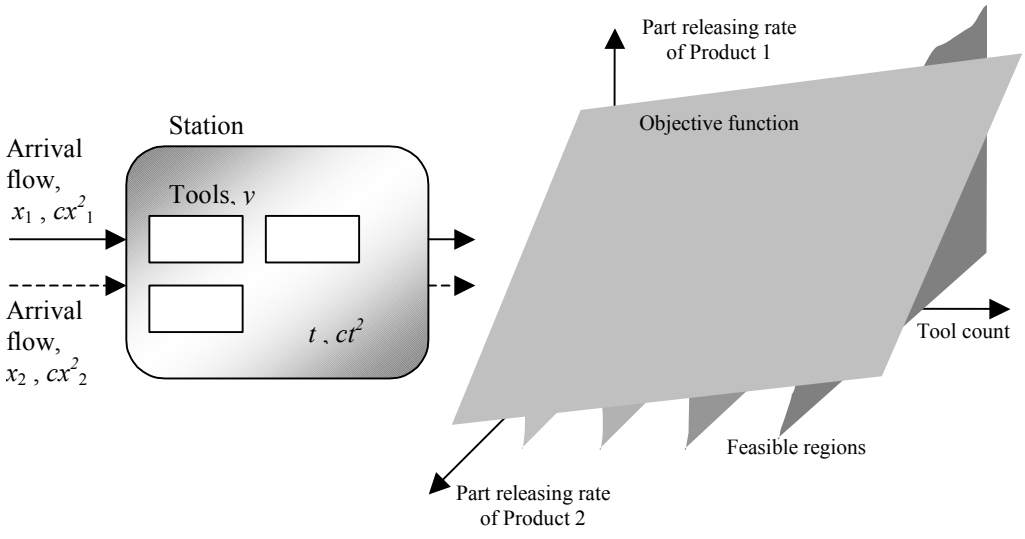


Figure 2.3 A Conceptual Example with Two Products and One Station

The performance of manufacturing systems has a large variability with respect to the kind of control logic used. The control logic, such as the part-releasing policy, has been one of the main concerns in manufacturing. Control rules and their effects on the measures of performance have been investigated (Lu, 1991; Kim, 1998). In this thesis, we concentrate on the design issues of reentrant manufacturing systems, assuming that a system deploys basic and naïve control rules. Hence, the assumptions in the research imply that, if better control schemes are found later, a good possibility of additional improvement in performance. In this research, we have the same understanding in initial system design, i.e., we consider simple and fixed control policies in modeling a system.

- Part-releasing policy: UNIF (Uniform parts inter-releasing time rule) - Parts to be released are selected proportional to the product-mix with constant releasing interval.
- Dispatching policy: FIFO (First-In-First-Out rule) - The stations serve the arrived parts in FIFO order.
- Batch size determination: MBS (Minimum-Batch-Size rule) - The sizes of the process batch and the setup batch are assumed to be identical and fixed to the minimum batch size of the manufacturing tool. The moving – arriving-at-equipment and departing-from-equipment – batch size is assumed to be given and constant between the steps.
- Setup times: The setup times are assumed to have general statistical distributions with different means and SCVs. They may vary according to the equipment type, product type, and step.

2.2. Formulation

2.2.1. Notation

K Number of products in product-mix

N Number of stations in manufacturing facility

TH_k Minimum required throughput rate of product k ($k = 1,2,\cdots,K$)

m_i Number of existing tools in station i ($i = 1,2,\cdots,N$)

p_k Margin of unit production rate of product k for unit period of production time considering sales revenue and operational costs, which are proportional to production rates

c_i Fixed cost of a tool in station i for unit period of production time considering the fixed costs such as purchase, installation, and operator wages, which are proportional to tool counts

C Allowable investment in unit period of production which is available for the new design or expansion in the facility[1]

n_{kl} The station that product k visits at step l

TCT_k Expected total cycle time (flow time) of product k

CT_{kl} Expected cycle time of product k at step l in station n_{kl}

CTq_{kl} Expected waiting time in queue of product k of step l in station n_{kl}

BT_{kl} Expected waiting time for batching and unbatching of product k at step l in station n_{kl}

[1] The investment in unit period of production is the fiscal amount flattened over the entire investment period. For example, if the investment period is five years and \$5M in the first year and \$3M in the third year are invested, the investment in unit period of year is \$1.6M without considering inflation and the interest rate.

TT_{kl} Expected transportation time from station $n_{k,l-1}$ to station n_{kl}

ACT_k Allowable cycle time of product k which is predefined strategically in design

L_k Number of steps in the route of product k

α_i Survival rate due to yield loss at station i

α_{kl} Cumulative yield of product k after completing the first l steps in its routing (i.e. $\alpha_{kl} = \prod_{l'=1}^{l} \alpha_{n_{kl'}}$)

rb_k Part-releasing batch size for product k

$\hat{x}_k$ Part-releasing rate into the route of product k in rb_k -batches (Decision variable)

y_i Number of tools in station i (Decision variable)

2.2.2. Descriptions of the decision variables

We suppose a reentrant manufacturing system with a product-mix of K products. Each product has a minimum throughput TH_k that the system must meet, where $k = 1,2,\cdots,K$. The mathematical model must determine the optimal throughputs $\hat{x}_k$ (the decision variables) which incur the maximum total net profit in a single period. Station i, where $i = 1,2,\cdots,N$, has y_i (the decision variables) pieces of equipment of the same type, each of which performs an identical process. Their mechanical properties are assumed to be known, e.g., process times and failure information. The number of pieces of equipment directly affects the utilization of the corresponding station as well as the expected level of the WIP and, finally, average cycle time. The "hat" symbol, as shown in $\hat{x}_k$ implies that the flow is in batch form.

2.2.3. Objective function

We maximize the net profit in a single period of operation, e.g., a month. The objective function is linear with respect to the profit and cost. However, in real situations, frequently we face a nonlinear property of net profit curve from, e.g., market behavior and discounts with production quantity changes. This observation leads to future research topics. A dimension analysis on profit coefficients and cost coefficients is given in Section 2.3.

$$\text{Maximize } -\sum_{i=1}^{N} c_i y_i + \sum_{k=1}^{K} p_k \alpha_{kL_k} r b_k \hat{x}_k \qquad \text{(Total net profit)} \tag{2.1}$$

2.2.4. Cycle time constraints

The expected cycle times cannot exceed the assigned limits. These constraints are derived from the queuing network models resulting in a complex nonlinear property. The expected total cycle time of product k, $TCT_k(\hat{\mathbf{x}}, \mathbf{y})$, is a function of the decision vectors of the part-releasing rates, $\hat{\mathbf{x}} = (\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_K)$, and the quantity of equipment, $\mathbf{y} = (y_1, y_2, \cdots, y_N)$.

$$TCT_k(\hat{\mathbf{x}}, \mathbf{y}) \le ACT_k, \; k = 1,2,\cdots,K \qquad \text{(Cycle time constraints)} \tag{2.2}$$

The total cycle time terms $TCT_k(\hat{\mathbf{x}}, \mathbf{y})$ present highly complex nonlinear functions including queuing approximation. $TCT_k(\hat{\mathbf{x}}, \mathbf{y})$ has the summation of four components: batching waiting time, waiting time in queue, process time, and transportation time. Batching waiting time is a function of $\hat{\mathbf{x}}$ only, while the waiting time in queue is a function of both $\hat{\mathbf{x}}$ and $\mathbf{y}$. Processing time and transportation time are constant in and between stations respectively.

Traffic variability equations are needed for the waiting time in queue. They form a system of equations used to calculate $\mathbf{a} = \{ca_1^2, ca_2^2, \cdots, ca_N^2\}$, the squared coefficient of variation (SCV) of the aggregated incoming material streams for each station. Therefore, considering the traffic variability equations, we can extend the cycle time constraints as follows

$$TCT_k(\hat{\mathbf{x}}, \mathbf{y}) \le ACT_k,\ k = 1,2,\cdots,K \qquad \text{(Cycle time constraints)}$$

$$ca_j^2 = b_j(\hat{\mathbf{x}}, \mathbf{y}) + \sum_{i=1}^{N} a_{ij}(\hat{\mathbf{x}}, \mathbf{y}) \cdot ca_i^2,\ j = 1,2,\cdots,N \qquad \text{(Traffic variability equations).} \tag{2.3}$$

2.2.5. Investment and operational cost constraints

The cost in configuration change cannot exceed the budget limitations. The cost coefficient c_i is the converted cost considering equipment installation and operations per unit period.

$$\sum_{i=1}^{N} c_i (y_i - m_i) \le C \qquad \text{(Investment constraint)}$$

(2.4)

2.2.6. Throughput constraints

The throughput of each product should at least meet the required quantity.

$$\alpha_{kL_k} rb_k \hat{x}_k \ge TH_k,\ k = 1,2,\cdots,K \qquad \text{(Throughput constraints)}$$

(2.5)

2.2.7. Existing equipment constraints

The changed configuration of equipment still has the existing equipments.

$$y_i \ge m_i,\ i = 1,2,\cdots,N \qquad \text{(Existing tool constraints)}$$

(2.6)

2.2.8. Integer property

Changes in the amount of equipment are expressed in integer form.

y_i, positive integer

(2.7)

Table 2.1 Formulation Summary

OptiProfit:

Maximize $z = -\sum_{i=1}^{N} c_i y_i + \sum_{k=1}^{K} p_k \alpha_{kL_k} rb_k \hat{x}_k$ (Total net profit)

Subject to,

$TCT_k(\hat{\mathbf{x}}, \mathbf{y}) \le ACT_k,\ k = 1,2,\cdots,K$ (Cycle time constraints)

$ca_j^2 = b_j(\hat{\mathbf{x}}, \mathbf{y}) + \sum_{i=1}^{N} a_{ij}(\hat{\mathbf{x}}, \mathbf{y}) \cdot ca_i^2,\ j = 1,2,\cdots,N$ (Traffic variability equations)

$\sum_{i=1}^{N} c_i (y_i - m_i) \le C$ (Investment constraint)

$\alpha_{kL_k} rb_k \hat{x}_k \ge TH_k,\ k = 1,2,\cdots,K$ (Throughput constraints)

$y_i \ge m_i,\ i = 1,2,\cdots,N$ (Existing tool constraints)

where, $\hat{\mathbf{x}} = \{\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_K\}$, $\mathbf{y} = \{y_1, y_2, \cdots, y_N\}$, y_i, positive integer

2.3. Profit and Cost Modeling

2.3.1. Introduction

The basic idea of the profit and cost modeling derives from the theory of constraints (TOC). TOC models the *throughput value* of each product in a product mix by selling price subtracted by raw material cost. Traditional *contribution margin* modeling involves the direct labor and overhead cost into the margin of the products; however, in today's manufacturing environment, variable costs represent a small percentage of total cost with the shift to automation that increases a firm's fixed production costs. TOC finally determines the production priority and optimal product mix (Atwater, 1997).

In this research, the margin coefficients, p_k , are defined according to the definition of TOC, i.e., throughput value. With the assumption that most of the modern manufacturing cost is related to equipment installation and operation, we calculate the manufacturing cost coefficient, c_i as a fixed cost.

2.3.2. Definitions and assumptions

We propose some definitions and assumptions for the profit and cost modeling as follows.

(1) The earnings of the business are uniquely obtained from the sales of products.

(2) The tools in a station have identical purchase cost, installation cost, salvage value, and length of lifecycle.

(3) Tools are replaced with the same ones at the end of their lifecycle.

(4) Cost in a unit period of production is comprised of the *overhead cost*, *fixed cost,* and *variable cost* (Lewis, 1995).

(5) Overhead cost in a unit period of production is a constant cost incurred in the business which is proportional neither to the number of tools nor to the throughput of production, e.g., computer clusters, customer support, building management, etc.[2] The modeling of total overhead cost is not included.

(6) Total fixed cost in a unit period of production is any cost which is proportional to the number of tools but not proportional to the throughput of production. For example, the costs converted onto the unit period of production for purchase and

[2] In reality, some overhead is variable, rising and falling with production, and other overhead is fixed remaining fairly constant on the production time horizon.

installation of tools considering salvage values at the end of the life cycle of the tools and fixed monthly wages of the operators at the tools would be a fixed cost. The cost conversion onto the unit period can include the effect of interest rates and inflation. Further models for fixed cost can embrace the depreciation, maintenance, taxes, insurance, lease rentals, interest on invested capital, and sales programs.

(7) Total variable cost in a unit period of production is the cost of raw materials and manufacturing resources including any operational cost which is proportional to the throughput of production but not proportional to the number of tools, e.g., the cost of raw materials and utilities consumed for the production in a unit period of production.

(8) *Incremental* costs or *marginal costs* are not considered.[3]

2.3.3. Margin and cost coefficients

We can formulation the *total fixed cost* as

$$\sum_{j=1}^{N} c_j y_j \tag{2.8}$$

where, c_j is the *fixed cost* in a unit period of production to operate a tool in station j.

[3] If they are considered, they might present a nonlinearity in the objective function.

See the cash flow diagrams in Figure 2.4. Given that the tools in station j have a lifecycle length of n periods over the time horizon, all costs are flattened to equivalent and constant costs over all periods. This gives a dimension of 'cost in dollars per tool per period.' In the cost analysis for c_j – the equivalence calculation for equal-payment-series – we could accommodate the effects of interest rates and inflation.

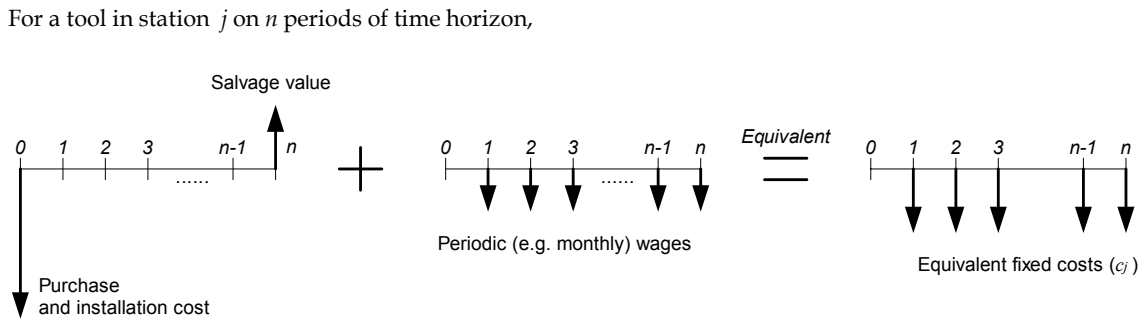


Figure 2.4 Cost Analysis for c_j (Equivalence Calculation for Equal-Payment-Series)

A brief and simple dimensional analysis shows that $[c_j]$ = V/NT, $[y_j]$ = N. Hence, $[\sum_{j=1}^{N} c_j y_j]$ = V/T, where $[\xi]$ is the dimension of ξ, V is the unit of values, e.g. 'dollar', N is the number of tools, and T is the unit of time periods, e.g. 'month.' Therefore, the dimension of the total cost is (V/NT) (N) = V/T, e.g. 'dollar/month.'

In a similar fashion, the *margin* can be expressed considering the sales revenue, yield loss, raw material cost, and operational cost. From the basic notation, $\hat{x}_k$ is the part-release rate of product k into the production system and $\alpha_{kL_k} rb_k \hat{x}_k$ gives the expected throughput of product k considering the cumulative yield rate. p_k is defined as the

margin of product k per unit period – the sales revenue subtracted by the yield loss, raw material cost, and operational cost per unit period. Note that p_k describes the margin and cost which are proportional to the part-releasing rates or throughputs.[4] Multiplying these together and summing, we find the *gross margin* in a unit period of production as

$$\sum_{k=1}^{K} p_k \alpha_{kL_k} rb_k \hat{x}_k \ . \tag{2.9}$$

Again, the dimensional analysis gives $[\, p_k \,]$ = V/N, $[\, \alpha_{kL_k} \,] = 1$, $[\, rb_k \,]$ = B, and $[\, \hat{x}_k \,]$ = N/B/T where N is the number of products and B is the number of batches. Therefore, $[\, \sum_{k=1}^{K} p_k \alpha_{kL_k} rb_k \hat{x}_k \,]$ = (V/N) (1) (B) (N/B/T) = V/T.

2.3.4. Allowable investment constant *C*

C is a converted cost value, which has a dimension of V/T, e.g., dollar/month. The overhead cost can be considered in the evaluation of the investment constant *C*. Suppose management has an allowable investment of $600 million for the next five fiscal years for the manufacturing facility and the monthly overhead cost is $0.5 million, then a simple

[4] In brief, (margin coefficient) = (unit sales revenue) – (unit raw material cost) – (unit raw material cost + unit operational cost) (1 – cumulative yield rate) / (cumulative yield rate). As a simple example, suppose we have a monthly sales revenue of $50 with one final product per month. The cumulative yield rate is 80%. The raw material cost is $20 for a product and the operation cost is $10 for a product including the scrap. Then, the margin coefficient is $50 - $20 - $ 10 - ($20 + $10) (1 - 0.8) / (0.8) = $12.5.

calculation shows that C is ($600M) / (5 years) (12 months/year) – ($0.5M) = ($9.5M/month) without any consideration of interest rates, inflation, and so forth.

2.4. Cycle Time Evaluation

2.4.1. Introduction

The cycle time or *flow time* of production is the expected time elapsed from the beginning to finishing of a production process. If a manufacturing system is composed of a separate sequence of processing steps such as job shop production, the cycle time of a product type in the system would be the sum of the individual cycle time at each processing step with an assumption that one processing step is independent from any other. This assumption is applied to the cycle time estimation in this research, as depicted in Figure 2.5.

The cycle time at each processing step is again decomposed into four parts: batching waiting time, waiting time in queue, processing time, and transportation time. The batching waiting time includes the batching and unbatching effects considering product-type-sensitive-batching and non-product-type-sensitive batching. The waiting time in queue is the time elapsed in front of the processing tools. The products in a given batch wait for the process in queue because of the variability of production processes. It is assumed that the mean and squared coefficient of variation (SCV) values of processing times are all known. Finally, the average transportation time of every transportation route between two stations is assumed to be given. Therefore, the main challenge in cycle time modeling lies in batching waiting time and waiting time in queue, which are generated by the variability in production systems.

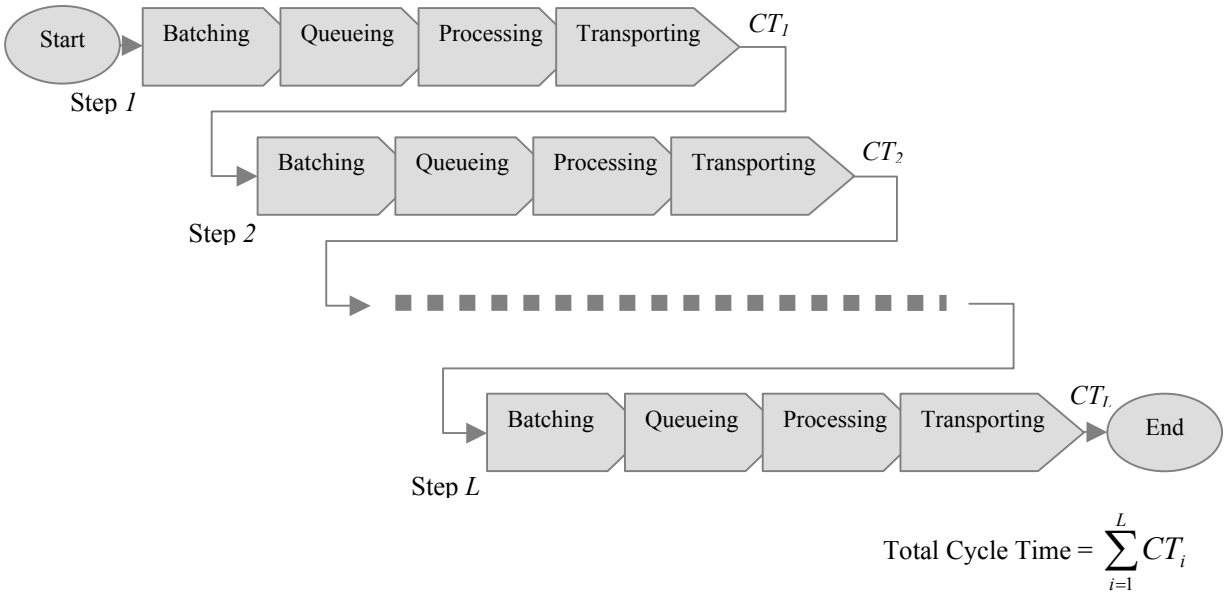


Figure 2.5 Evaluation of the Total Cycle Time

2.4.2. Additional notation for cycle time evaluation

τ_{kl}, γ_{kl}^2 Mean natural batch process time and SCV of product k at step l

$\hat{t}_i$, $\hat{c}t_i^2$ Aggregated mean natural batch process time and SCV in station i

$\tilde{t}_i$, $\tilde{c}t_i^2$ Effective batch process time and SCV in station i with failure

s_{kl}, cs_{kl}^2 Mean natural batch setup time and SCV of product k at step l

$\hat{s}_i$, $\hat{c}s_i^2$ Aggregated batch setup time and SCV in station i

t_i, ct_i^2 Effective batch process time and SCV in station i with setup and failure

mb_{ij} Moving batch from station i to station j in lots (i.e. Arrival batch at station j from station i)

pb_j Process or setup batch size for station j

λ_j Effective arrival rate at the station j

ρ_j Effective utilization of station j

$\hat{c}r_k$ SCV of part-releasing into the routing of product k in rb_k-batches

2.4.3. Flow rates

The flow rates from station i to station j are

$$\lambda_{0j} = \sum_{k=1}^{K} rb_k \hat{x}_k [n_{k1} = j]$$

$$\lambda_{ij} = \sum_{k=1}^{K} \sum_{l=1}^{L_k - 1} rb_k \hat{x}_k \alpha_{kl} [n_{kl} = i, n_{k,l+1} = j]$$

$$\lambda_{j0} = \sum_{k=1}^{K} rb_k \hat{x}_i \alpha_{kL_k} [n_{kL_k} = i]$$

(2.10)

Station 0 represents the raw material inventory (RMI) and the notation $[S]$ represents one if statement S is true and zero otherwise, mainly following the notation and expressions of Hopp (2002). Therefore, the in-flow rate into station j in the unit of individual product is

$$\lambda_j = \lambda_{0j} + \sum_{i=1}^{N} \lambda_{ij} .$$

(2.11)

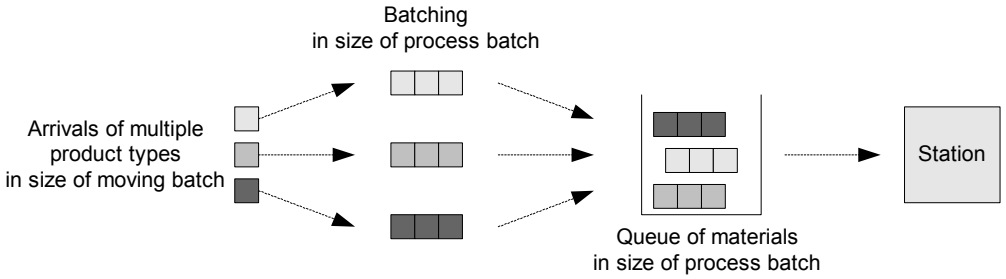
2.4.4. Batching effects

A basic assumption in batching analysis is that in every station there is a queue for processing batches. The processing queue can be located either physically in front of the station or logically at different places in the facility. Every product to be processed in the station should form a process batch. As soon as each processing batch is constructed, it is physically cumulated or logically registered to be processed in the station on a FIFO basis.[5]

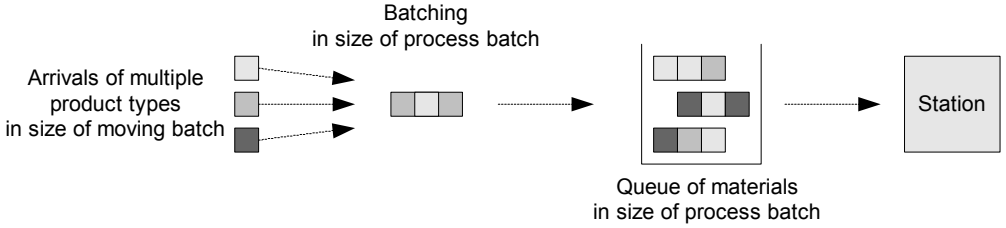
The batching effect incurs delays in two parts, *batching* and *unbatching*. Batching occurs before the processing queue in front of each station while unbatching is done after the station to form the moving batches. Therefore, the batching effect is decomposed into two components $BT_{kl} = BT_{kl}^{B} + BT_{kl}^{U}$, where BT_{kl}^{B} is the batching time in front of the station and BT_{kl}^{B} is the unbatching time before departure to the next step.

While the product types can be different in processing batches for certain process equipment, some kinds of equipment require material of the same type in each processing batch, as illustrated in Figure 2.6 (a). Typically, when compared with non-product-type-sensitive batching, product-type-sensitive batching in Figure 2.6 (b) presents a longer wait-to-batch time (WTBT) to form process batches in front of the queue in the corresponding station.[6]

[5] If we do not have a FIFO assumption, we could deploy other smarter policies, for example, using a pool of individual products to dynamically form the processing batches without a queue for processing batches.
[6] As matter of fact, we can intuitively claim that the variabilities of *batch arrivals in front of the batch queue* in two cases are not identical. While the traffic variability equations assume the non-product-type-sensitive case, in this study we approximate the variability using the traffic variability equations also in the product-type-sensitive case. More thorough considerations of this topic could be studied in future research.



(a) Product-type-sensitive batching



(b) Non-product-type-sensitive batching

Figure 2.6 Product-type-sensitive Batching and Non-product-sensitive Batching

(Case 1) WTBT in product-type-sensitive stations

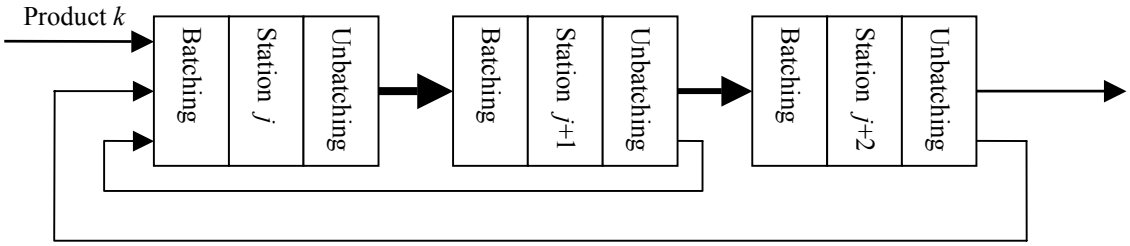
$$BT_{kl}^{B} = \frac{1}{2}\frac{\left(pb_{n_{kl}} - mb_{kl}\right)^{+}}{mb_{k,l-1}}\frac{mb_{k,l-1}}{\hat{x}_k rb_k \beta_{k,n_{kl}}} = \frac{1}{2}\frac{\left(pb_{n_{kl}} - mb_{kl}\right)^{+}}{\hat{x}_k rb_k \beta_{k,n_{kl}}} \qquad \text{(Batching)}$$

$$BT_{kl}^{U} = \frac{1}{2}\frac{\left(mb_{k,l+1} - pb_{kl}\right)^{+}}{\lambda_{n_{kl}}\alpha_{n_{kl}}} \qquad \text{(Unbatching)}$$

where, $\beta_{kj} = \sum_{l \in S_{kj}} \alpha_{k,l-1}$ and S_{kj} is the set of steps in station j of product k.

(2.12)

β_{kj}, as seen in Figure 2.7, is the sum of cumulative yields of product k streams in front of station j. Likewise, the unbatching process is also analyzed with β'_{kj} which is the sum of cumulative yields of product k streams after station j.



Step information
……
$n_{k,l} = j$
$n_{k,l+1} = j+1$
$n_{k,l+2} = j$
$n_{k,l+3} = j+1$
$n_{k,l+4} = j+2$
$n_{k,l+4} = j$
……

At the station *j*, there exist three incoming streams, whose arrival rates in individual product are $\hat{x}_k rb_k \alpha_{k,l-1}$, $\hat{x}_k rb_k \alpha_{k,l+1}$, and $\hat{x}_k rb_k \alpha_{k,l+4}$ respectively. Aggregated rates of incoming streams at station *j* is, hence, $\sum_{l \in S_{kj}} \hat{x}_k rb_k \alpha_{k,l-1} = \hat{x}_k rb_k \sum_{l \in S_{kj}} \alpha_{k,l-1} = \hat{x}_k rb_k \beta_{kj}$, where S_{kj} is the set of steps in station *j* of product *k*, e.g., $S_{kj} = \{l, l+2, l+4\}$.

Figure 2.7 Example of Batching Analysis at the Product-sensitive-batching Station

(Case 2) WTBT in non-product-type-sensitive stations

$$BT_{kl}^{B} = \frac{1}{2} \frac{\left(pb_{n_{kl}} - mb_{kl}\right)^{+}}{\lambda_{n_{kl}}} \qquad \text{(Batching)}$$

$$BT_{kl}^{U} = \frac{1}{2} \frac{\left(mb_{k,l+1} - pb_{kl}\right)^{+}}{\lambda_{n_{kl}} \alpha_{n_{kl}}} \qquad \text{(Unbatching)} \qquad (2.13)$$

2.4.5. Effective processing time and SCV

Figure 2.8 describes the calculation steps for the effective batch processing time with setup and failure effects. Basically, the natural batch processing time information is given including the mean batch processing time and SCV with respect to the product type and its step in the route. We aggregate the processing time in order to obtain the aggregated batch processing time at each station. From the assumption that failures occur on the processing time horizon, the failure adjustment on the processing time is performed thereafter.[7] On the other hand, the natural setup time information gives the mean batch setup time and SCV with respect to the product type and its step in the route, which is equivalent to the information of batch processing time. Likewise, the aggregated batch setup time is calculated on each station. Finally, the aggregated batch processing time and the aggregated batch setup time is integrated into the effective batch processing time with failure and setup effects.

We first calculate the aggregated class of batch size batch pb_j for station j. Defining τ_j for the mean process time and γ_j^2 for the SCV of a pb_j-batch of lots at station j, we obtain

$$t_j = \sum_{k=1}^{K}\sum_{l=1}^{L_k} rb_k \hat{x}_k \alpha_{k,l-1}\tau_{kl}[n_{kl} = j] \Big/ \sum_{k=1}^{K}\sum_{l=1}^{L_k} rb_k \hat{x}_k \alpha_{k,l-1}[n_{kl} = j],$$

$$t_j^{\,2}(ct_j^{\,2}+1) = \sum_{k=1}^{K}\sum_{l=1}^{L_k} rb_k \hat{x}_k \alpha_{k,l-1}\tau_{kl}^{\,2}(\gamma_{kl}^{\,2}+1)[n_{kl} = j] \Big/ \sum_{k=1}^{K}\sum_{l=1}^{L_k} rb_k \hat{x}_k \alpha_{k,l-1}[n_{kl} = j]. \quad (2.14)$$

[7] It is also possible to incorporate the effect of preventive maintenance (PM) (Hopp, 1999).

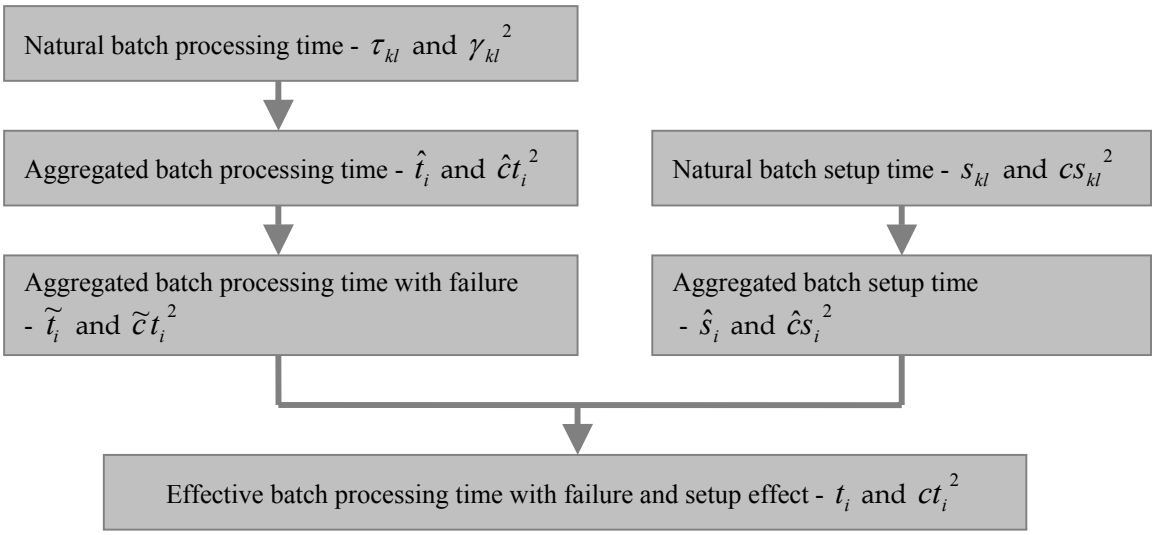


Figure 2.8 Effective Batch Processing Time with Failure and Setup Effects

From the assumption that the processes in the stations are performed in minimum batching size (MBS), the process batches are the same size at each station. Therefore, we use MBS as the effective process batch size.

We consider the effects from random failures in order to obtain the aggregated batch process time with failure effect in corresponding stations. An analysis of the preemptive failures situation gives the failure-adjusted batch-process time and SCV as

$$\tilde{t}_j = t_j / A_j,\ A_j = \frac{mf_j}{mf_j + mr_j}$$

$$\tilde{c}t_j^2 = ct_j^2 + (1 + cr_j^2)A_j(1 - A_j)\frac{mr_j}{t_j} \qquad (2.15)$$

where, mf_j is the mean time of failures, mr_j is the mean time of repairs, and cr_j^2 is the SCV of repairs in station j.

We calculate the aggregated batch setup time in the same manner as for the case of aggregated processing time.

$$\hat{s}_j = \sum_{k=1}^{K}\sum_{l=1}^{L_k} \lambda_k \alpha_{k,l-1} s_{kl}[n_{kl} = j] \Big/ \sum_{k=1}^{K}\sum_{l=1}^{L_k} \lambda_k \alpha_{k,l-1}[n_{kl} = j],$$

$$\hat{s}_j{}^2(\hat{cs}_j{}^2 + 1) = \sum_{k=1}^{K}\sum_{l=1}^{L_k} \lambda_k \alpha_{k,l-1} s_{kl}{}^2(cs_{kl}{}^2 + 1)[n_{kl} = j] \Big/ \sum_{k=1}^{K}\sum_{l=1}^{L_k} \lambda_k \alpha_{k,l-1}[n_{kl} = j].$$

(2.16)

Finally, the effect of setups imposes another adjustment on the aggregated processing time information, t_j and $ct_j{}^2$. Using the notation in Hopp (2002),

$$t_j = \tilde{t}_j + \hat{s}_j$$

$$t_j{}^2 ct_j{}^2 = \tilde{t}_j{}^2 \tilde{c} t_j{}^2 + \hat{s}_j{}^2 \hat{cs}_j{}^2$$

(2.17)

where, $\hat{s}_j$ is the average setup time and $\hat{cs}_j{}^2$ is the average SCV at station j.

2.4.6. Waiting times in queues

Products arriving at a station normally come from more than one station. The mean interarrival time can be easily computed if the mean interarrival times of the streams are known. However, even though the in-flows have their own probabilistic

distributions for inter-arrival time, the deviation or SCV of the aggregated in-flows with all the distributions cannot be calculated in meaningful form. This difficulty is an obstacle in the evaluation of cycle time since the variability information of the aggregated streams is required for the queuing network analysis. An experimentally fitted distribution might be obtained from numerical analysis, but this approach cannot be incorporated with the analytical and optimization models in this study. Consequently, we approximate the SCV of arrivals using the *traffic variability equations* (TVEs) (Whitt, 1983; Hopp, 2002).

TVEs are based on the multi-class queuing network model for steady state analysis. This queuing network model is known to have an advantage in modeling various design factors. For example, it can incorporate the effects of yield loss, batching, unbatching, setup, failure, preventive maintenance, and most importantly, variability in material flows and processing times. On the other hand, when compared to other models, such as the fluid model, it has the disadvantage that it is not effective for the reentrant material flows and initial system configuration on a finite time horizon. However, because of the advantage of the network model in including the consideration of variability, this work uses the network model, assuming that the design time horizon is sufficiently long.[8]

In order to evaluate the waiting time in queue, TVEs need the expected batch size of effective arrivals to station j. The expected batch size of external arrivals can be

[8] By the nature of large scale manufacturing systems in general, it is assumed that substantial reconfiguration of a facility is not frequent.

calculated as $xb_j = \lambda_{0j} \Big/ \sum_{k=1}^{K} \hat{x}_k[n_{kl} = j]$, $j = 1,2,\cdots,N$. In front of station j, we obtain the effective batch size

$$eb_{ij} = \begin{cases} \max(xb_j, pb_j) & i = 0 \\ \max(pb_i, pb_j) & 1 \le i \le N \end{cases}.$$

(2.18)

Defining the effective batch arrival rate, $\hat{\lambda}_{ij} = \lambda_{ij} / eb_{ij}$, we find the rate sum of incoming effective batch streams at station j as,

$$\hat{\Lambda}_j = \sum_{i=0}^{N} \hat{\lambda}_{ij} .$$

(2.19)

Now, according to the TVEs, the arrival SCV of externally arriving batches to each station is approximated as follows.

$$cx_j^2 = 1 - \tilde{w}_j + \tilde{w}_j \sum_{k=1}^{K} cr_k^2 \left(\hat{\lambda}_k[n_{k1} = j] \Big/ \sum_{k'=1}^{N} \hat{\lambda}_{k'}[n_{k'1} = j] \right),$$

where, $\tilde{w}_j = [1 + 4(1-\rho_j)^2(\tilde{v}_j - 1)]^{-1}$, $\tilde{v}_j = \left[\sum_{k=1}^{K} \left(\hat{\lambda}_k[n_{k1} = j] \Big/ \sum_{k'=1}^{N} \hat{\lambda}_{k'}[n_{k'1} = j] \right)^2 \right]^{-1}$, and the effective utilization at station j is $\rho_j = \hat{\Lambda}_j t_j / y_j$.

(2.20)

Between the stations inside of the reentrant queuing network, the SCVs can be obtained from the expression of Hopp (2002).

$$ca_j^2 = b_j + \sum_{i=1}^{N} a_{ij} ca_i^2 \ , \ j = 1,2,\cdots,N \ ,$$

$$\text{where, } a_{ij} = \frac{w_j}{\hat{\Lambda}_j}\left(\frac{\hat{\lambda}_{ij}}{\max(pb_i, pb_j)}\right)\frac{\lambda_i}{\hat{\Lambda}_i}\alpha_i q_{ij}(1-\rho_i^2) \ ,$$

$$b_j = 1 - w_j + \frac{w_j}{\hat{\Lambda}_j}\left(\frac{xb_j\lambda_{0j}}{eb_{0j}^2}\right)cx_j^2 + \frac{w_j}{\hat{\Lambda}_j}\sum_{i=1}^{N}\frac{\hat{\lambda}_{ij}}{eb_{ij}}\left[\frac{\lambda_i}{\hat{\Lambda}_i}\alpha_i q_{ij}\rho_i^2\phi_i + pb_i(1-\alpha_i q_{ij})\right],$$

$$q_{ij} = \lambda_{ij} \Big/ \sum_{j'=0}^{N}\lambda_{ij'} \ , \ \phi_i = 1 + (\max\{ct_i^2, 0.2\} - 1)/\sqrt{y_i} \ ,$$

$$w_j = \left[1 + 4(1-\rho_j)^2(v_j - 1)\right]^{-1} , \ v_j = \left[\sum_{i=0}^{N}(\hat{\lambda}_{ij} \Big/ \hat{\Lambda}_j)^2\right]^{-1} .$$

(2.21)

This formulation induces a system of equations that requires an inverse calculation of an $N \times N$ matrix. We use a G/G/m queuing model for each station. The waiting time in queue at station n_{kl}, $CTq_{kl}(\mathbf{x}, \mathbf{y})$, is approximated using *Kingman's equation* considering the batching effect in shared queue $y_{n_{kl}}$ of tools in station n_{kl}, i.e.,

$$CTq_{kl} = \left(\frac{ca_{n_{kl}}^2 + ct_{n_{kl}}^2}{2}\right)\left(\frac{\rho_{n_{kl}}^{\sqrt{2(y_{n_{kl}}+1)}-1}}{y_{n_{kl}}(1-\rho_{n_{kl}})}\right)t_{n_{kl}} .$$

(2.22)

This system experiences *variability pooling* occurring in arrivals when the station has a process batch size greater than one.

2.4.7. Transportation time

The statistical distribution for transportation time between two stations is hard to obtain in real situations due to dynamic properties such as transportation route selection, vehicle characteristics, and traffic congestion. Consequently, the variability in transportation times affects the variability of arrivals in the next station; however, assuming that the variability in transportation is relatively low, we consider only the variability analysis from the traffic variability equations without the transportation effects.[9]

Defining TT_{kl} as the average transportation time of product k from step $l-1$ at step l – dispatching time from station $n_{k,l-1}$ to station n_{kl} – we can simply add it to the expected total cycle time along the routes of corresponding products. As a result, transportation time is modeled as a constant in each itinerary. Note that TT_{k1} is the transportation time between part-releasing to the first station in the route of product k.

[9] In reality, stable and fixed path systems such as conveyer systems have a relatively low coefficient of variation. In contrast, free-path transportation systems such as fork lift systems can present a higher coefficient of variation, particularly, when congestion situations are frequent.

2.4.8. Expected total cycle time

Combining the above results, we sum the batching, queuing, processing, and transportation times of product k at station n_{kl}. Finally, the expected total cycle time of product k is expressed.

$$TCT_k(\hat{\mathbf{x}}, \mathbf{y}) = BT_k(\hat{\mathbf{x}}) + CTq_k(\hat{\mathbf{x}}, \mathbf{y}) + PT_k + TT_k,$$

$$\text{where } BT_k(\hat{\mathbf{x}}) = \sum_{l=1}^{L_k} BT_{kl}(\hat{\mathbf{x}}),\ CTq_k(\hat{\mathbf{x}}, \mathbf{y}) = \sum_{l=1}^{L_k} CTq_{kl}(\hat{\mathbf{x}}, \mathbf{y}),\ PT_k = \sum_{l=1}^{L_k} t_{n_{kl}}, \text{ and}$$

$$TT_k = \sum_{l=1}^{L_k} TT_{kl} + TT_{L_k,FGI} \quad l = 1,2,\cdots,L_k,\ k = 1,2,\cdots,K. \tag{2.23}$$

Note that $TT_{k,FGI}$ is the expected transportation time from the last station of product k, i.e., station n_{kL_k}, to the finished good inventory (FGI).

CHAPTER 3

MODEL ANALYSIS

3.1. Model Observation

Observation of the OptiProfit problem begins with the simplest case, which models a system of one product flow with one processing step and one station with multiple tools, as illustrated in Figure 3.1 and Figure 3.2. Since it does not have aggregated or reentrant flows, the main formulation does not contain the traffic variability equations. The process queue is assumed to be stable; in other words, the utilization is below 100% assuming that the maximum utilization $\rho_{\max}$ is less than or equal to, say, 0.98. The processing batch size is given as b. The G/G/m/inf queuing model is used to estimate the average cycle time with the waiting time to batch. The cycle time constraint prevents the part arrival rate from increasing excessively, and the cost constraint gives an upper bound on the number of tools deployed. The minimum throughput constraint should be met simultaneously. One-product-one-station model analysis can be practically applied to the service and manufacturing systems with one type of server or station and one type of customer or material, e.g., bank teller service, vehicle repair, fast food service, one-process manufacturing, etc.

The formulation of the simplest case is as follows.

Objective function:

$$\text{Maximize } z(x, y) = px - cy$$

Constraints:

$$TCT(x, y) \le ACT$$

$$cy \le C$$

$$y \ge \frac{xt}{b\rho_{\max}}$$ [10]

$$x \ge TH$$

y , positive integer

where, $TCT(x, y) = BT(x) + CTq(x, y) + t = \frac{b}{2x} + \left(\frac{ca^2 + ct^2}{2}\right) \cdot \left(\frac{u^{\sqrt{2(y+1)}-1}}{y\ (1-u\ )}\right) \cdot t + t$,

$u = \frac{xt}{yb} \le \rho_{\max}$.

(3.1)

Applying calculus, we easily find that $TCT(x, y)$ is convex with respect to x and is concave with respect to y .[11] Therefore, it is possible to find the set of x s, $S_j = \{x \mid TCT(x, y_j) \le ACT, \quad TH \le x \le b\rho_{\max} y_j / t\}$ for every integer $y_j = j$, $1 \le j \le \lfloor C/c \rfloor$. Extracting the maximum value x_j^* in each S_j , i.e., $x_j^* = \max S_j$, we can conclude that the global optimal value $z^* = \max_j z(x_j^*, y_j)$. This procedure applies only to the simplest case of OptiProfit; the global optimal solution is not easy to find. We see in

[10] Theoretically, *utilization constraints* should be $y > xt/b$. However, we assign a value close to one for $\rho_{\max}$ for the practical tractability of the equation. $\rho_{\max}$ denotes the allowable highest utilization, which is strategically assigned.

[11] $\partial^2 TCT(x, y)/\partial x^2 > 0$ with fixed y . $\partial^2 TCT(x, y)/\partial y^2 < 0$ with fixed x for integer-relaxed TCT .

this chapter that the interactions between the decision variables and the effects of aggregated variability drive the intractability of OptiProfit.

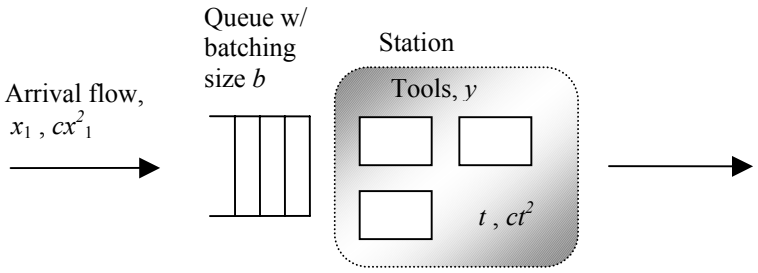


Figure 3.1 One-product-one station Case

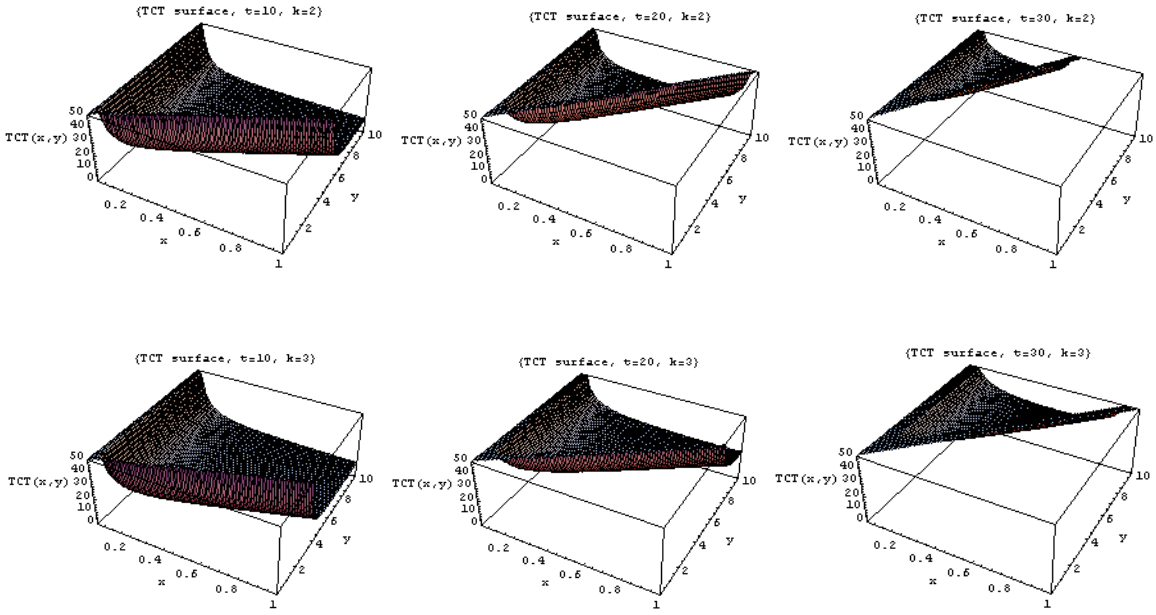


Figure 3.2 TCT Surfaces with Varying Processing Time and Batching Size

3.2. Properties of OptiProfit

3.2.1. Complexity classification

NP-hard problems are computationally intractable; no NP-hard problem can be solved by any known polynomial-time algorithm (Papadimitriou, 1982). This section shows the NP-hardness of OptiProfit by reduction to the known NP-hard 0-1 knapsack problem.

Consider a simple version of OptiProfit. Fix the real values $\hat{x}_k$ and we have a new set of integer variables $\tilde{\mathbf{y}} = \{\tilde{y}_1, \tilde{y}_2, \cdots, \tilde{y}_N\}$ where $\tilde{y}_j = m_j + y'_j$, $y'_j \in \{0,1\}^N$, and $m_j \geq 1$.[12] In addition, assume the number of products to be one, non-reentrant routing, a batching size of one at any station, yield rates of one, a squared coefficient of variation of arriving flows and processing time at every station of one[13], and negligible transportation time. We formulate these assumptions as follows, and call it **P**. The problem **P** is specified by the parameters, $\mathbf{c} = \{c_1, c_2, \cdots, c_N\}$, $\mathbf{t} = \{t_1, t_2, \cdots, t_N\}$, $\mathbf{n} = \{n_{11}, n_{22}, \cdots, n_{1N}\} = \{1,2,\cdots, N\}$, $\lambda = \lambda_1$, and $ACT = ACT_1$.

P: $\{(\mathbf{c}, \mathbf{t}, \mathbf{n}, \lambda, ACT) \mid N \geq 1, \text{all numbers are positive}\}$

i.e.,

P: Minimize $z = \mathbf{c}\tilde{\mathbf{y}} = \sum_{i=1}^{N} c_i \tilde{y}_i$ (Total cost)

[12] The solution of **P**, if solved correctly, provides the decision as to whether to increase one additional tool in each station to make the system more profitable.

[13] The system **P** has exponential interarrival and processing time at every station. As a result, we can observe that the traffic variability equations are trivial.

Subject to,

$$TCT(\tilde{\mathbf{y}}) = \sum_{j=1}^{N} ctq(\tilde{y}_j) = \sum_{j=1}^{N} \frac{u(\tilde{y}_j)^{\sqrt{2(\tilde{y}_j+1)}-1}}{\tilde{y}_j\left(1-u(\tilde{y}_j)\right)} \le ACT\ ,\ k = 1,2,\cdots,K$$

where, $u(y_j) = \dfrac{\lambda \cdot t_j}{y_j}$, $\tilde{\mathbf{y}} = \{\tilde{y}_1, \tilde{y}_2, \cdots, \tilde{y}_N\}$, $\tilde{y}_j = m_j + y'_j$, and $y'_j \in \{0,1\}^N$.

(Cycle time constraints)

(3.2)

We find that $TCT(\tilde{\mathbf{y}})$ is a linear summation of nonlinear functions, $ctq_j(\tilde{y}_j)$, each of which is a function of y_j only. Therefore, $ctq_j(\tilde{y}_j)$ can be replaced by a linear function of y'_j, i.e.,

$$\begin{aligned} ctq_j(\tilde{y}_j) &= ctq_j(m_j + y'_j) \\ &= \left(ctq_j(m_j+1) - ctq_j(m_j)\right)y'_j + ctq_j(m_j) = a_j y'_j + d_j . \end{aligned}$$

(3.3)

Setting $b = ACT - \sum_{j=1}^{N} d_j$, we can rewrite **P** as,

P:

Minimize $\sum_{i=1}^{N} c_i y'_i$ (Total cost)

Subject to,

$$\sum_{j \in R_k} a_j y'_j \le b \,,\; y'_j \in \{0,1\}^N \qquad \text{(Cycle time constraints)}$$

(3.4)

Note that a_j and b are negative. Otherwise, the problem would be trivial.

Now, we take an instance of a 0-1 knapsack problem **Q**, as follows.

Q:

Minimize $\omega = \sum_{i=1}^{N} \gamma_i y'_i$

Subject to $\sum_{j=1}^{N} \alpha_j y'_j \ge \beta \,,\; y'_j \in \{0,1\}^N$

(3.5)

For arbitrary **Q**, we claim that it can be transformed to an instance of **P**. To show this, the following relations should hold.

(1) $c_j = \gamma_j \,,\; j = 1,2,\cdots,N$

We can simply assign the values of c_j to secure the relation.

(2) $a_j = -\alpha_j \,,\; j = 1,2,\cdots,N$

In order to make a_j equivalent to the arbitrary negative value of $-\alpha_j$, we show

a_j can be set to any real positive number. Since $ctq_j(n) = \dfrac{u_j(n)^{\sqrt{2(n+2)}-1}}{n\left(1-u_j(n)\right)}$ where

n is a positive integer is monotonically decreasing with respect to n, $a_j = ctq_j(m_j+1) - ctq_j(m_j)$ is always negative. With fixed n, $ctq_j(n)$ is a function of t_j only in $u_j(n) = x_j t_j / n$, where x_j is also determined from the assumption. From the observation that $\lim_{t_j \to 0}\left(ctq_j(m_j+1) - ctq_j(m_j)\right) = 0$, $\lim_{t \to n/x_j}\left(ctq_j(m_j+1) - ctq_j(m_j)\right) = -\infty$, and that $ctq_j(m_j+1) - ctq_j(m_j)$ is monotonically decreasing, the value of t_j is uniquely determined to make the relation (2) hold.

(3) $b = -\beta$

Regarding (3), it is simply possible to have the value of b equivalent to $-\beta$ by setting an appropriate value of $ACT = -\beta + \sum_{j=1}^{N} d_j$.

By this reduction, the *NP*-hardness of the OptiProfit problem follows from the *NP*-hardness of the 0-1 knapsack problem.

3.2.2. Convexity

In this section, we observe that the nonlinear constraints of the integrality-relaxed version of OptiProfit, i.e., $TCT_k(\mathbf{x}, \mathbf{y})$, show nonconvexity. Consider the example shown in Figure 3.3.

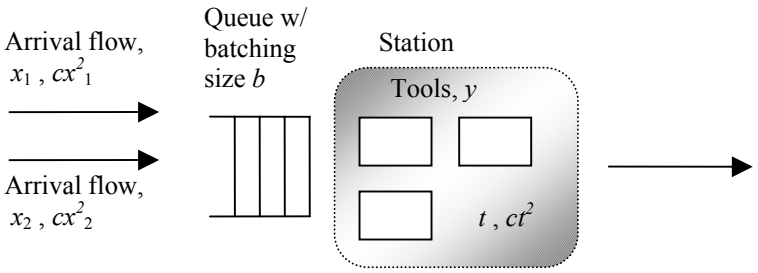


Figure 3.3 A Simple Example for Convexity Analysis

In order to illustrate that $TCT_k(\mathbf{x},\mathbf{y})$ may violate the general property of convexity, we show that there exist α_1 and α_2 satisfying

$$TCT(\mathbf{x}_1,\mathbf{y}_1)+TCT(\mathbf{x}_2,\mathbf{y}_2) < 2\left[TCT(\alpha_1\mathbf{x}_1+(1-\alpha_1)\mathbf{x}_2,\alpha_1\mathbf{y}_1+(1-\alpha_1)\mathbf{y}_2)\right]$$

$$TCT(\mathbf{x}_1,\mathbf{y}_1)+TCT(\mathbf{x}_2,\mathbf{y}_2) > 2\left[TCT(\alpha_2\mathbf{x}_1+(1-\alpha_2)\mathbf{x}_2,\alpha_2\mathbf{y}_1+(1-\alpha_2)\mathbf{y}_2)\right]$$

$$0 \le \alpha_1 \le 1,\ 0 \le \alpha_2 \le 1.$$

(3.6)

Assume a model with two inflows and one station with three tools, i.e., $\mathbf{x}_1 = [0 \quad 12]$, $\mathbf{x}_2 = [10 \quad 0]$, $cx_1^2 = 0$, $cx_2^2 = 2.5$, $b = 2$, $t = 0.4$, $ct^2 = 0$, $\mathbf{y}_1 = [3]$, $\mathbf{y}_2 = [3]$.

Mathematica™ produces numeric results for $TCT_k(\mathbf{x},\mathbf{y})$ with traffic variability equations for the two incoming flows.[14] Plotting $TCT_k(\mathbf{x},\mathbf{y})$ with respect to $0 \le \alpha \le 1$, we have Figure 3.4.

[14] Mathematica code for the convex analysis is given in APPENDIX A.

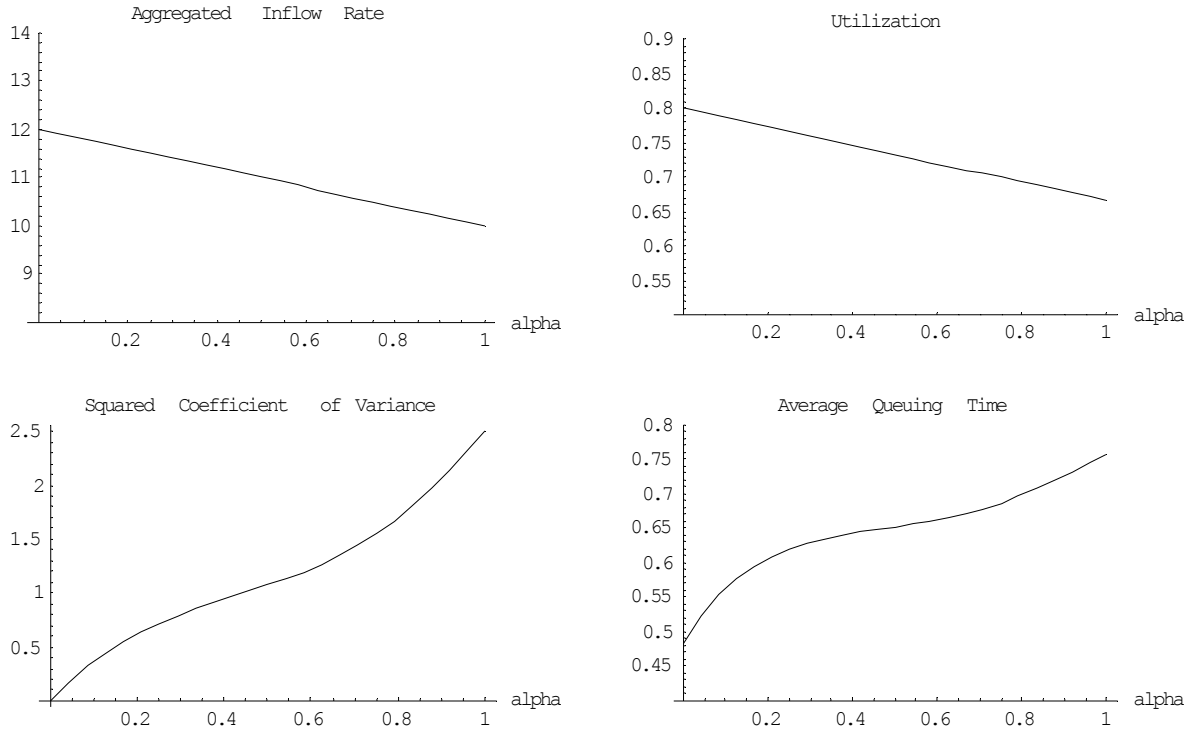


Figure 3.4 Nonconvex Property of OptiProfit

From the plotted Average Queuing Time in Figure 3.4, it is evident that there are two distinguished parts, concave and convex, showing that $TCT_k(\mathbf{x},\mathbf{y})$ is not always convex. The non-convexity is due to the difference in variability of the two incoming streams. The variability at a station is affected by the tool counts and flow rates of preceding stations. Therefore, any system with more than one stream and one station inevitably has a varying variability at each station which could result in non-convexity of the total cycle times.

3.2.3. Monotonicity

If a nonlinear programming problem has monotonicity, it can become more tractable, leading to simplified solution methods and increased insight into the problem (Papalambros and Wilde, 2000). After a preliminary analysis of $TCT_k(\mathbf{x},\mathbf{y})$, it is evident that it has the nonmonotonic property in the nonlinear constraints. To illustrate this, take the example problem in the previous section with a different data set, i.e., $\mathbf{x}_1 = [0 \quad 4]$, $\mathbf{x}_2 = [8 \quad 0]$, $cx_1^2 = 0$, $cx_2^2 = 0.5$, $b = 2$, $t = 0.6$, $ct^2 = 0$, $\mathbf{y}_1 = [3]$, $\mathbf{y}_2 = [3]$. A plot of the numerical results is shown in Figure 3.5. [15]

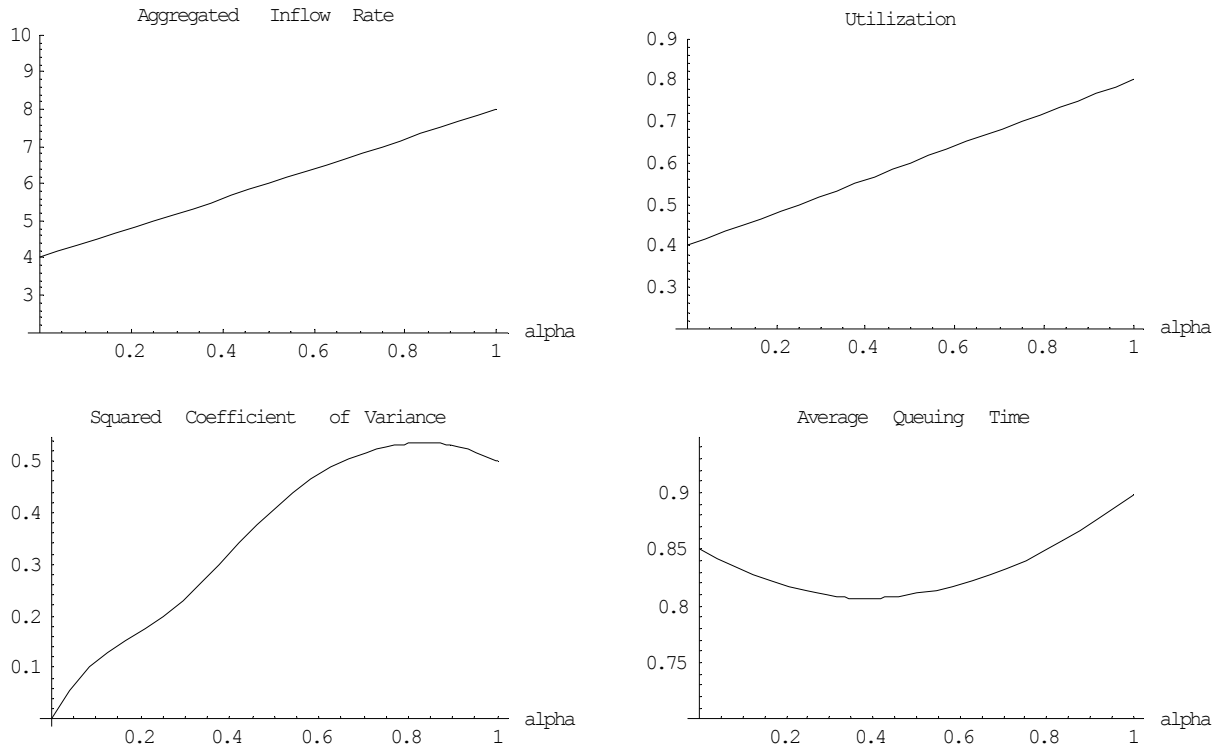


Figure 3.5 Nonmonotone Property of OptiProfit

[15] Mathematica code for the monotonicity analysis is given in APPENDIX A.

The aggregated part-releasing rate increases from four to eight. From the plot, one can observe that the average waiting time in queue decreases in the interval [0, 0.4] of α approximately. This is not surprising since the inflow $\mathbf{x}_2$ has a much higher variability than $\mathbf{x}_1$. One observes that, before the utilization becomes sufficiently high, the total cycle time is dominated by the variability terms. The total cycle time sharply increases as the utilization approaches one.

In more detail, $TCT_k(\mathbf{x},\mathbf{y})$ contains the $\lambda_j(\mathbf{x})$, which are the linear functions of $\mathbf{x}$. With fixed $\mathbf{y}$, we find the behavior of $TCT_k(\mathbf{x},\mathbf{y})$ in domain of $\mathbf{x}$, $D_{\mathbf{x}}$ as,

$$TCT_k(\mathbf{x},\mathbf{y}) = \infty \text{ as } \max_{j\in S_k} u_j = \max_{j\in S_k} \frac{\lambda_j(\mathbf{x})\cdot t_j}{b_j y_j} \to 1$$

where $S_k = \{n_{kl} \mid l = 1,2,\cdots,L_k\}$ is the set of steps in the routing of product k,

$$TCT_k(\mathbf{x},\mathbf{y}) \geq 0,\ D_{\mathbf{x}} = \left\{ \mathbf{x} \mid \frac{\lambda_j(\mathbf{x})\cdot t_j}{b_j y_j} < 1, j = 1,2,\cdots N \right\}. \tag{3.7}$$

Figure 3.6 (a) depicts conceptually the behavior of $TCT_k(\mathbf{x},\mathbf{y})$ with respect to a single variable $x_{k'}$, for example. If the utilization of station j approaches one as $x_{k'} \to x_{k'}^U$, i.e., $\lambda_j(\mathbf{x}) \to b_j y_j / t_j$, $b_j > 1$ with $x_{k'} \to x_{k'}^U$, the waiting time in queue in station j, $CTq_j(\mathbf{x},\mathbf{y})$, approaches infinity. So, therefore, does $TCT_k(\mathbf{x},\mathbf{y})$, but only if it has at least one visit to station j in the routing of product k. Figure 3.6 (b) shows a $TCT_k(\mathbf{x},\mathbf{y})$ curve, with the possible fluctuation due to the variability-dominated effect in

the low utilization region. In this setting, if a certain heuristic is intended to find a maximum $x_{k'}$, satisfying the cycle time constraints with ACT_k, it should search from $x_{k'}^U$ with decreasing $x_{k'}$ until it finds $x_{k'} = b$.[16] If it searches from the left edge with increasing $x_{k'}$, the heuristic would stop with $x_{k'} = a$, an incorrect termination point. This important observation is reflected in the suggested heuristic for the OptiProfit problem, Differential Coefficient Based Search (DCBS) in the next chapter.

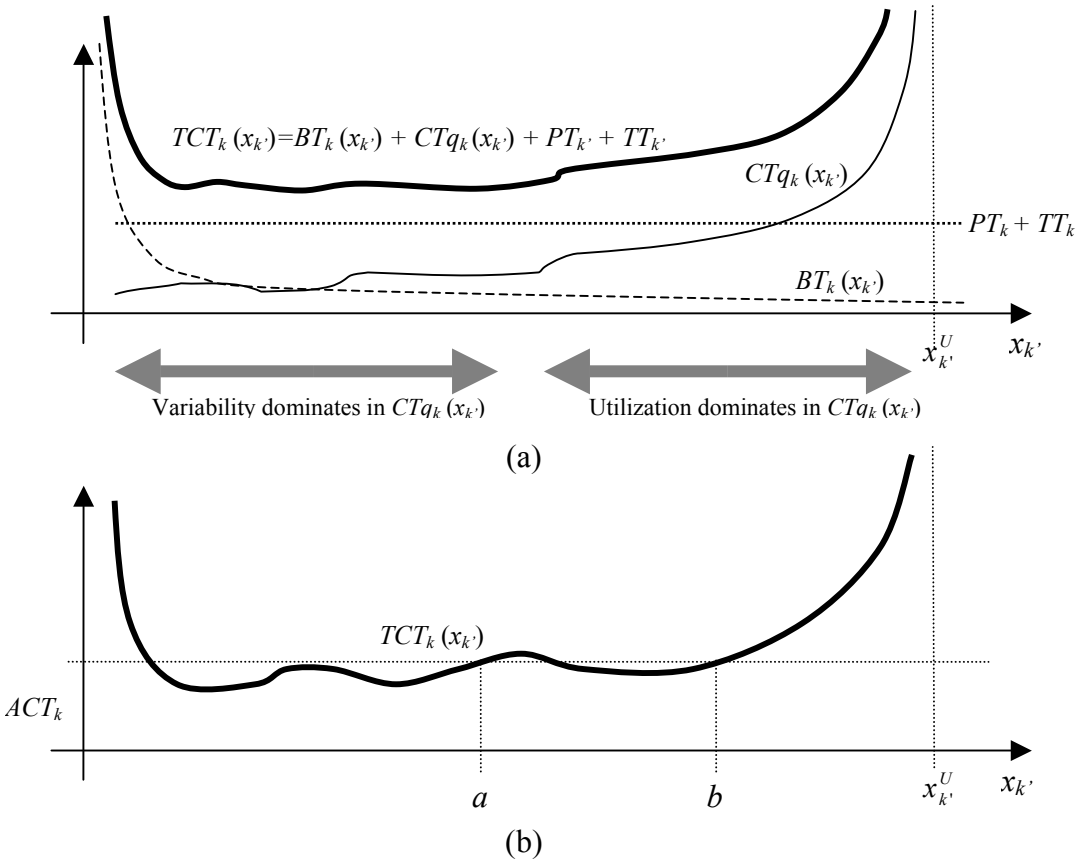


Figure 3.6 Behavior of $TCT_k(\mathbf{x}, \mathbf{y})$

[16] In DCBS, the concept of "look-ahead" is implemented to find b. It first searches for a with increasing x_k with a step size of Δx. When a is found, DCBS does not stop but proceeds with a predefined number of look-ahead steps, $n_{lookahead}$, to find a possible b.

3.3. Upper-bound Analysis

In order to obtain an efficient upper bound for OptiProfit, a maximization problem, we define a modified version of OptiProfit, called *OptiProfitUB*. As mentioned above, nonconvexity and nonmonotonicity are properties attributed to the nature of the squared coefficient of variation (SCV) in arrival flows at the stations. Consequently, OptiProfitUB does not include the squared coefficient of variation of arrivals to each station, ca^2 , nor, accordingly, the traffic variability equations. Mathematically, OptiProfitUB is expressed as follows.

OptiProfitUB:

Maximize $z^{UB} = -\sum_{i=1}^{N} c_i y_i + \sum_{k=1}^{K} p_k \alpha_{kL_k} rb_k \hat{x}_k$ (Total net profit)

Subject to,

$TCT_k^{UB}(\hat{\mathbf{x}}, \mathbf{y}) \le ACT_k,\ k = 1,2,\cdots,K$ (Cycle time constraints)

$\sum_{i=1}^{N} c_i (y_i - m_i) \le C$ (Investment constraint)

$\alpha_{kL_k} rb_k \hat{x}_k \ge TH_k,\ k = 1,2,\cdots,K$ (Throughput constraints)

$y_i \ge m_i,\ i = 1,2,\cdots,N$ (Existing tool constraints)

where, $TCT_k^{UB}(\hat{\mathbf{x}}, \mathbf{y}) = BT_k(\hat{\mathbf{x}}) + CTq_k^{UB}(\hat{\mathbf{x}}, \mathbf{y}) + PT_k + TT_k$,

$$CTq_k^{UB}(\hat{\mathbf{x}}, \mathbf{y}) = \sum_{l=1}^{L_k} CTq_{kl}^{UB}(\hat{\mathbf{x}}, \mathbf{y}),\ CTq_{kl}^{UB}(\hat{\mathbf{x}}, \mathbf{y}) = \left(\frac{ct_{n_{kl}}^2}{2}\right)\left(\frac{\rho_{n_{kl}}^{\sqrt{2(y_{n_{kl}}+1)}-1}}{y_{n_{kl}}(1-\rho_{n_{kl}})}\right) t_{n_{kl}},$$

$\hat{\mathbf{x}} = \{\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_K\}$, $\mathbf{y} = \{y_1, y_2, \cdots, y_N\}$, y_i are positive integers.

(3.8)

We claim that the optimum value of OptiProfitUB is always greater than or equal to that of OptiProfit. To prove this, first conceptualize the waiting time in queue into a sum of two parts, i.e., the waiting time in queue by the variability of aggregated arriving flows, CTq^{CA}, and the waiting time in queue by the variability of processing times, CTq^{CT}:

$$CTq_{kl}(\hat{\mathbf{x}},\mathbf{y}) = CTq_{kl}^{CA}(\hat{\mathbf{x}},\mathbf{y}) + CTq_{kl}^{CT}(\hat{\mathbf{x}},\mathbf{y}), \text{ where}$$

$$CTq_{kl}^{CA}(\hat{\mathbf{x}},\mathbf{y}) = \left(\frac{ca_{n_{kl}}^2}{2}\right)\left(\frac{\rho_{n_{kl}}^{\sqrt{2(y_i+1)}-1}}{y_{n_{kl}}(1-\rho_{n_{kl}})}\right)t_{n_{kl}} \text{ and } CTq_{kl}^{CT}(\hat{\mathbf{x}},\mathbf{y}) = \left(\frac{ct_{n_{kl}}^2}{2}\right)\left(\frac{\rho_{n_{kl}}^{\sqrt{2(y_i+1)}-1}}{y_{n_{kl}}(1-\rho_{n_{kl}})}\right)t_{n_{kl}}. \quad (3.9)$$

For $0 \le \rho_{n_{kl}} < 1$, $CTq_{kl}^{CA}(\hat{\mathbf{x}},\mathbf{y}) \ge 0$ since $ca_{n_{kl}}^2$ is non-negative. Therefore, $CTq_{kl}^{CA}(\hat{\mathbf{x}},\mathbf{y}) = CTq_{kl}(\hat{\mathbf{x}},\mathbf{y}) - CTq_{kl}^{CT}(\hat{\mathbf{x}},\mathbf{y}) \ge 0$. Thus, $CTq_{kl}(\hat{\mathbf{x}},\mathbf{y}) - CTq_{kl}^{UB}(\hat{\mathbf{x}},\mathbf{y}) \ge 0$, by definition. Consequently, $CTq_k(\hat{\mathbf{x}},\mathbf{y}) \ge CTq_k^{UB}(\hat{\mathbf{x}},\mathbf{y})$. Thus, $TCT_k(\hat{\mathbf{x}},\mathbf{y}) \ge TCT_k^{UB}(\hat{\mathbf{x}},\mathbf{y})$ for arbitrary $\hat{\mathbf{x}}$ and $\mathbf{y}$. From this result, we find that the feasible region of OptiProfitUB, D_{UB}, is definitely equal to or larger than that of OptiProfit, D, i.e., $D \subset D_{UB}$. Consequently, the optimal objective value of OptiProfitUB is always greater than or equal to that of OptiProfit, i.e., $z^* \le z^{UB^*}$.

Furthermore, by eliminating ca^2, the source of nonconvexity, OptiProfitUB is a convex and monotone program, and we can express OptiProfitUB in a simplified form:

OptiProfitUB: Maximize $z^{UB} = z(\hat{\mathbf{x}}^+, \mathbf{y}^-)$

Subject to,

$$f_k(\hat{\mathbf{x}}^+, \mathbf{y}^-) \le ACT_k,\ k = 1,2,\cdots,K$$

$$g(\mathbf{y}^+) \le C$$

$$h_k(\mathbf{x}^+) \ge TH,\ k = 1,2,\cdots,K$$

where, f, g, and h represent the cycle time constraints for the upper bounds (TCT_k^{UB}), investment constraint, and required throughput constraints respectively.

(3.10)

Although the modified model does not represent any reasonable physical system, it mathematically serves to develop upper bounds for OptiProfit. A number of state-of-the-art commercial solvers are especially efficient for convex MINLP. In this work we use GAMS™ to calculate OptiProfitUB for example cases and compare the outputs with the results from various heuristic solution approaches, including a meta-heuristic. See CHAPTER 5.

3.4. Heuristic Solution for the General Cases of OptiProfit

3.4.1. Existing approaches

MINLP problems such as OptiProfit appear in many different applications in engineering design, computational chemistry, computational biology, communication, finance, and other areas. In particular, there is a lack of MINLP methods for solving large-scale MINLPs arising in real-world applications (Lasschuit, 2004; Barton, 2004; Navarro, 2003; Chattopadhyay, 2002).

Currently, scientists and engineers believe that NP-complete problems cannot be solved by algorithms with less than exponential computation time, in the worst case. One way to approach these problems is to design algorithms that do not guarantee a solution to every problem instance, but which solve many if not most problems on average, and which run fast. Approximation algorithms have been developed in response to the impossibility of solving many problems exactly. In the case of NP-complete problems, we sacrifice optimality to find a "good" solution that can be computed efficiently. Trading-off optimality in favor of tractability is the paradigm of heuristics and approximation algorithms.

Some of the most popular methods for convex MINLP problems are branch-and-bound (Beale, 1977), generalized Benders decomposition (GBD) (Geoffrion, 1972), and outer approximation (OA) (Duran and Grossmann, 1986). The branch-and-bound method, applied to MILP, can be extended in a straightforward way to MINLP, using a number of tricks that can be used to improve the performance of branch-and-bound for MINLP. There exist powerful programs for solving large-scale MILPs (Mixed Integer Linear Programs). These are based on a branch-and-cut framework combined with methods from constraint programming. Still, difficulties remain in generalizing MILP-techniques to MINLP: (i) The LP relaxation must be replaced by a different relaxation, which is often not tight enough or is expensive to generate; (ii) The computation of local solutions can be expensive; (iii) It can be difficult to derive efficient cuts. As a result, only medium-sized MINLPs can usually be solved by branch-and-cut. In practice, large problems are often solved either by a MILP approximation or by meta-heuristics combined with local-search methods.

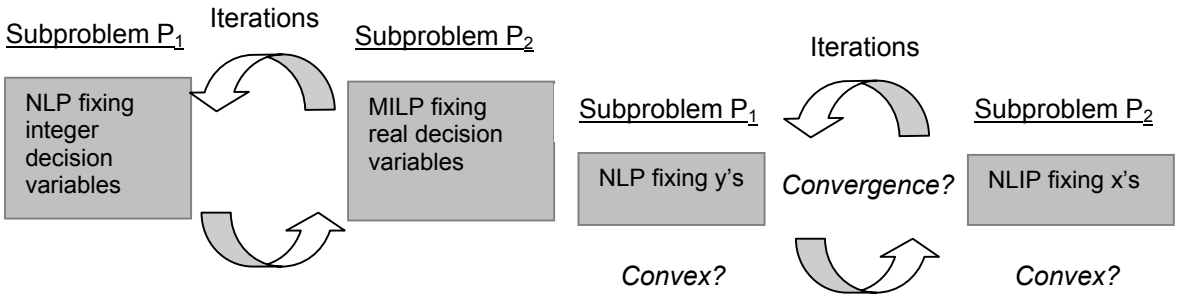
Generalized Benders decomposition and outer approximation solve the MINLP by an iterative process. The problem is decomposed into an NLP subproblem, which has the integer values fixed, and an MILP master problem. The NLP subproblems optimize the continuous variables and provide an upper bound to the MINLP solution, while the MILP master problems have the role of predicting a new lower bound for the MINLP solution, as well as new integer variables for each iteration. The search terminates when the predicted lower bound equals or exceeds the current upper bound. The main difference between GBD and OA is in the definition of the MILP master problem. In the GBD algorithm, the MILP master problem is given by a dual representation of a continuous space, while in the OA method, it is given by a primal approximation. In general, the OA method requires fewer iterations and thus the solution of fewer NLP subproblems, but the MILP problems require more computation as compared with GBD. For more details, see Grossmann (1990). To meet sufficient conditions for convergence, all three solution methods require that the MINLP satisfy some form of convexity conditions.

3.4.2. Intractability of OptiProfit

(1) No benefits from the decomposition methods

OptiProfit does not always have the convexity property and is often used for large problems. Moreover, the decomposition into subproblems of NLP and MIP is not always possible, which is a condition assumed in decomposition methods such as Generalized Bender's. In Figure 3.7, the continuous variables and discrete variables are coupled in

highly complex nonlinear terms; even though we decompose the problem into two subproblems, one subproblem inevitably is a nonlinear integer program (NLIP), not an integer program as intended. In addition there arises a convergence problem; the iterations of the two decomposed subproblems do not necessarily converge to a solution.



(a) Typical decomposition iterations (b) Possible decomposition of OptiProfit

Figure 3.7 Problem in the Decomposition of OptiProfit

(2) Strategic situation in execution time

For practical use, several well-designed commercial solvers for MINLP have been produced. For example, GAMS/BARON is a numerical solution widely used in a variety of problems including the nonconvex cases. Basically, all solvers ultimately seek an exact global solution but the calculation frequently requires an excessive execution time and sometimes fails to find the answer. As problem size grows, execution-time cost becomes more significant. In addition, in most real situations, the analysts and managers of industrial systems need a rapid analysis tool to deliver the outputs for various settings of their systems. Therefore, a fast and good heuristic approach is highly desirable in large-scale and concurrent system development.

3.4.3. Heuristic for OptiProfit

To find an adequate heuristic for OptiProfit, we review some categories of frequently used heuristic algorithms.

(1) Random methods

A simple way to generate approximations is to find a random feasible solution – i.e. a random permutation. Of course, this method yields poor results in general. But the runtime of such simple schemes is typically negligible.

(2) Successive augmentation (greedy heuristic)

Under the *successive augmentation* approach, a partial layout is extended to a neighborhood solution at which point the arrangement is produced without any attempt to improve it. At each step, a better possible free label is assigned to the current solution. This class of algorithm has been applied to optimization problems such as the Graph Coloring problem and the Traveling Salesman problem.

(3) Local search

Local search has been described as an approach in which intuition and empirical tests play a crucial role. In spite of this, local search is one of the most-used techniques to approximate many combinatorial problems because of its performance and simplicity. The basic principle of this heuristic is to iteratively improve a given solution by performing local changes. Normally, changes that improve the solution are accepted while those that make it worse are rejected.

(4) Hill climbing

A *hill climbing* algorithm is implemented as follows. An initial arrangement is generated. Then, proposed moves in the corresponding neighborhood are generated at

random and accepted if their gain is positive or, in order to go across a plateau, if it is zero. Once a predefined maximum number of consecutive proposed moves have not strictly reduced the cost of the arrangement, the algorithm terminates.

(5) Full search

At each step of a *full search* algorithm, the gain of each possible transition is computed in order to choose the move with the maximum gain in the current neighborhood. Exploiting the fact that the graph is sparse, and using a priority queue, time savings are possible because it is not required to re-compute the moves of nodes that are not neighbors of previously interchanged nodes.

(6) Meta-heuristics

Generally speaking, a heuristic method is developed and tailored for a particular problem domain. In contrast, a *meta-heuristic* is designed for general use in many optimization problems. Meta-heuristics such as *genetic algorithm*, *tabu search*, and *simulated annealing*, are developed for combinatorial problems. However, some variants are intended to accommodate the continuous variables in optimization models (Corana, 1987). Although most heuristics tend to become trapped in local optima, meta-heuristics have mechanisms to escape them and find a better solution closer to the global optimum. Consequently, meta-heuristics can be effective in finding a good solution for nonconvex problems. Nevertheless, the question of computation time for excessive trials arises when one applies a meta-heuristic such as simulated annealing to an MINLP problem such as OptiProfit. Computing time is very sensitive to the initial solution and parameter inputs. If one imposes a restriction on computation time, requires a fast heuristic, and still

requires good results, meta-heuristics are not necessarily appropriate for large MINLP problems.

CHAPTER 4

SOLUTION APPROACHES

4.1. Introduction

Due to the intractability of the nonlinear cycle time constraints and integer decision variables, an exact solution method is not the best approach for large problems. Consequently, we investigate both a heuristic and a modified meta-heuristic and compare them with some basic heuristics and a numerical solver for upper bound analysis. See Table 4.1.

(1) *Basic GAP* uses a greedy ascent procedure (GAP), or hill climbing, in a decomposed framework of an integer domain and a real domain. This heuristic determines the decision variable for incrementing at each iteration step based on certain values of product types and stations.

(2) *Differential Coefficient Based Search* (DCBS) has the same heuristic framework as basic GAP, but uses unique schemes to determine the changes of decision variables at each iteration step.

(3) *Modified Simulated Annealing* (MSA) for MINLP, the simulated annealing algorithm for continuous variables by Corana (1987), is applied to OptiProfit.

(4) *Upper bound analysis* is used as OptiProfitUB is programmed in GAMS™ for convex and monotone MINLP optimization. The MINLP solver used is DICOPT.

Table 4.1 Heuristic Solvers and Performance Comparison Method for OptiProfit

Solver			Descriptions	Classification
1	Basic Greedy Ascent Procedure (GAPs)	High Utilization / High Profit (HUHP)	Intuitive, conventional, and practical selection of tools and product to control (6 variants)	A greedy ascent procedure (GAP) or hill climbing in the decomposition framework
		High Utilization / Large Slack (HULS)		
		High Utilization / Small Slack (HUSS)		
		High Queuing Time / High Profit (HQHP)		
		High Queuing Time / Large Slack (HQLS)		
		High Queuing Time / Small Slack (HQSS)		
2	Differential Coefficient Based Search (DCBS)		Smarter tool and product selection using Differential Coefficient Based Search (DCBS)	
3	Modified Simulated Annealing for MINLP		Customized for MINLP problems with a modification from a meta-heuristic	A modified meta-heuristic of simulation annealing

Performance Comparison Method	Upper Bound Formulation	Solution Method
Upper Bound Analysis	OptiProfitUB (ca^2 - eliminated version of OptiProfit)	GAMS/DICOPT: A numerical solver for convex MINLP optimization

4.2. Basic Greedy Ascent Procedures

Figure 4.1 summarizes the decomposition framework for a GAP heuristic in this work.[17]

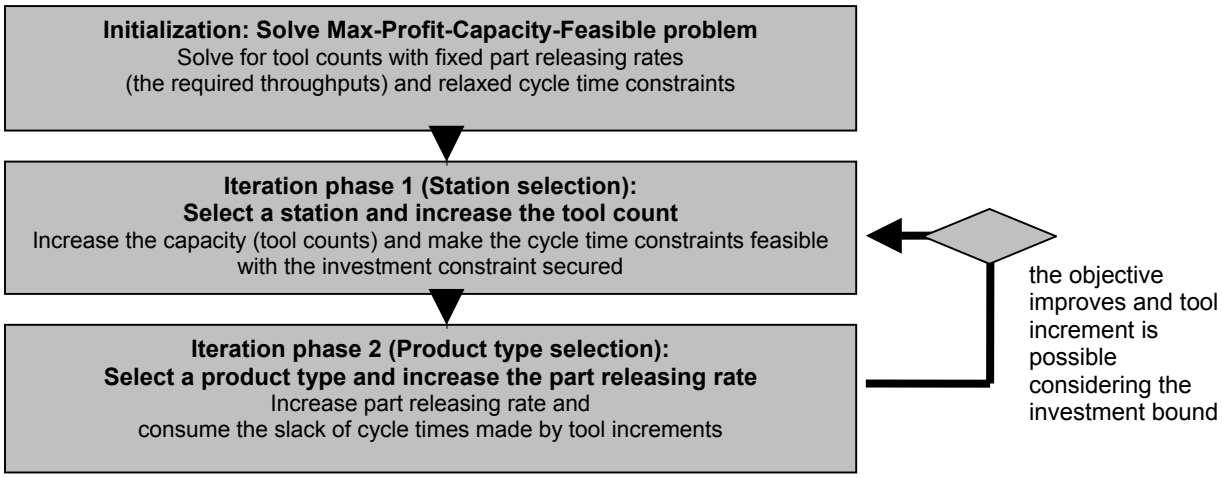


Figure 4.1 A Greedy Ascent Procedure in Decomposition Framework

4.2.1. Initialization phase

The initialization phase is the first phase to find a solution that satisfies all constraints except that of cycle time in OptiProfit. Initialization finds a system configuration that can manufacture the required throughputs with a stable utilization of each station, i.e., less than one. This solution provides a seed with which to begin the process of finding a basic feasible solution.

[17] The basic GAP framework considers the improvement of the objective at the end of each iteration. Therefore, it is not a pure greedy ascent procedure.

Fixing the part-releasing rates, the formulation becomes a nonlinear integer program, albeit a trivial one. We call this simplified form the Max-Profit-Capacity-Feasible formulation.

Max-Profit-Capacity-Feasible (MPCF):

Maximize $z = -\sum_{j=1}^{N} c_j y_j + \sum_{k=1}^{K} p_k \alpha_{kL_k} rb_k \hat{x}_k$ (Total net profit)

Subject to,

$\rho_i \le \rho_{\max}$, $1 \le i \le N$ (Utilization constraints)

$\sum_{i=1}^{N} c_i (y_i - m_i) \le C$ (Investment constraint)

$\alpha_{kL_k} rb_k \hat{x}_k = TH_k$, $k = 1,2,\cdots,K$ (Throughput constraints)

$y_i \ge m_i$, $i = 1,2,\cdots,N$ (Existing tool constraints)

where, $\hat{\mathbf{x}} = \{\hat{x}_1, \hat{x}_2, \cdots, \hat{x}_K\}$, $\mathbf{y} = \{y_1, y_2, \cdots, y_N\}$, $\rho_i = \hat{\Lambda}_i t_i / y_i$, y_i, positive integer

(4.1)

Note that $\rho_{\max}$ is strategically assigned to secure a stable system. A quick inspection of the formulation to find a solution is quite straightforward using simple algebra. From the required throughput constraints, we fix $\hat{\mathbf{x}}^* = \{\hat{x}_1^*, \hat{x}_2^*, \cdots, \hat{x}_K^*\}$ where $\hat{x}_k^* = TH_k / \alpha_{kL_k} rb_k$. From the utilization constraints, we can find y_i maximizing MPCF,

$y_i^* = \max(\lfloor \hat{\Lambda}_i t_i / \rho_{\max} \rfloor, m_i)$. Note that $\mathbf{y}^* = \{y_1^*, y_2^*, \cdots, y_1^*\}$ satisfies the investment constraint; otherwise, the original problem is infeasible.[18]

Even though $\mathbf{y}^* = \{y_1^*, y_2^*, \cdots, y_1^*\}$ satisfies the investment constraint, the original OptiProfit problem would typically be infeasible since a nontrivial OptiProfit will have strict cycle time constraints. During the first visit in Phase 1, we find a feasible solution with which to begin the objective improving iterations.

4.2.2. Iteration phase 1: Station selection

The first phase of an iteration is the selection of a station to increase its tool count. The total cycle time is monotone, decreasing with respect to tool count. Hence, we can control the cycle time below that allowable in the cycle time constraints as long as the investment constraint is satisfied.

Practically, we can increase the tool count of the station with the highest utilization, a widely accepted scheme. However, the highest utilization of a particular station does not always mean that it has the largest average waiting time in queue. Since *bottlenecking* is more related to the time delay at a station rather than its utilization, it is more reasonable to select the station with the highest waiting time in queue. Therefore, two schemes in station selection are considered, High Utilization (HU) and High Queuing Time (HQ).

[18] Physically, the manufacturing system would be unable to achieve even the required minimum throughputs with given investment.

Table 4.2 Station Selection Schemes

Station selection scheme	Description
Highest Utilization (HU)	The station that has the highest utilization among stations is selected to have additional tool increment.
Highest Queuing Time (HQ)	The station that has the largest average waiting-in-queue time is selected.[19]

4.2.3. Iteration phase 2: Product type selection

The second iteration phase selects the product type to increase the part-releasing rate. As discussed in CHAPTER 3, the total cycle time of a product type is not necessarily monotone or increasing. In addition, an increase in the part-releasing rate does not guarantee an improvement in the objective, compensating for the tool increase in the first phase. Consequently, it is necessary to check if the objective has been increased each time the part-releasing rate of the selected product type is increased. If it is not improved, the heuristic performs several more iterations. The number of such *look-ahead* iterations is predefined. The heuristic terminates when it cannot find improvement or when the investment does not allow additional tool increases.

We consider three schemes for product type selection. The *High Profit* (HP) scheme selects the product type with the highest unit profit. The *Largest Slack* (LS) scheme chooses the product type with the largest *slack time* in total cycle time, where slack time is the allowable cycle time subtracted by current cycle time evaluated. The *Small Slack* (SS) scheme is similar to Largest Slack except that it chooses the product type with the smallest slack time.

[19] The station with the highest utilization does not always have the longest average waiting-in-queue time.

Table 4.3 Product Type Selection Scheme

Product type selection scheme	Description
Highest Profit (HP)	The product type that has the biggest unit sales profit is selected to increase to consume the slack cycle time.
Largest Slack Cycle Time (LS)	The product type that has the largest slack in current average cycle time is selected.
Smallest Slack Cycle Time (SS)	The product type that has the smallest slack in current average cycle time is selected.[20]

4.2.4. Variations

The two schemes for station selection with the three schemes for product type selection constitute six variants of the Basic GAP heuristics, as follows.

(1) High Utilization / High Profit (HUHP) scheme

(2) High Utilization / Large Slack (HULS) scheme

(3) High Utilization / Small Profit (HUSS) scheme

(4) High Queuing Time / High Profit (HQHP) scheme

(5) High Queuing Time / Large Slack (HQLS) scheme

(6) High Queuing Time / Small Slack (HQSS) scheme

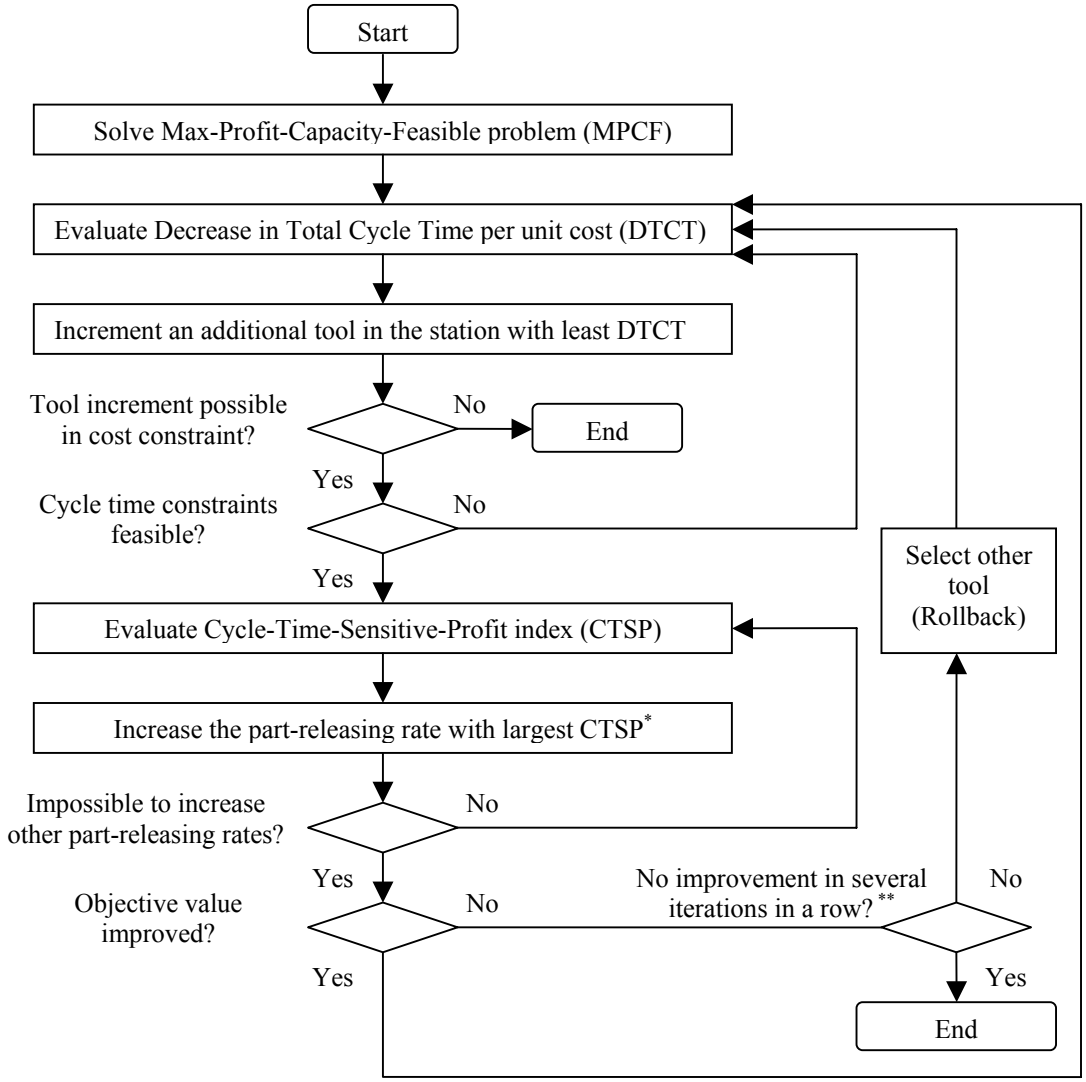
4.3. Differential Coefficient-based Search Heuristic

4.3.1. Principal idea

The fundamental intention in developing the DCBS heuristic is to incorporate several decision factors in one indexing system. For example, in station selection, we are more likely to select a station that has a lower unit cost for the tool and a greater decrease

[20] Since the additional increase of production is frustrated by the firstly met bound of cycle time constraints, the product type that has the smallest slack may have significant consideration in increasing the part releasing rate.

in cycle time with one tool increment. To quantify the combined effects of the factors, we include the differential coefficient of cycle time with respect to the change in tool number. Therefore, DCBS can be categorized as a steepest-ascent heuristic. Figure 4.2 illustrates DCBS at a conceptual level.



Note:
* The selected part releasing rate increases stepwise unless the cycle time constraints are violated
** The number of allowed iterations with non-improving objective is predefined

Figure 4.2 Flow Diagram of the DCBS Heuristic

4.3.2. Decrease of the Total Cycle Time Index

(1) Definition

The *Decrease of Total Cycle Time* per unit cost (DTCT) is given by

$$DTCT_j = \sum_{k=1}^{K} \frac{1}{c_j} \left[\frac{\partial TCT_k(\hat{\mathbf{x}}, \mathbf{y})}{\partial y_j} \bigg|_{\mathbf{x}=\tilde{\mathbf{x}}, \mathbf{y}=\tilde{\mathbf{y}}'} \right] \text{ for station } j \,. \tag{4.2}$$

The DTCT indexing system quantifies the aggregated effect of tool increment cost and cycle time reduction by one tool increment. We select the station with the least DTCT, i.e., $j^* = \arg\min_j DTCT_j$. Note that all indices are negative.

(2) Evaluation

The differential coefficient of the total cycle time with respect to the tool increment at station j is numerically estimated as follows.

$$\frac{\partial TCT_k(\hat{\mathbf{x}}, \mathbf{y})}{\partial y_j} = \frac{TCT_k(\hat{\mathbf{x}}, \mathbf{y} + \varepsilon \cdot \mathbf{v}_j) - TCT_k(\hat{\mathbf{x}}, \mathbf{y})}{\varepsilon} \tag{4.3}$$

where, $\mathbf{v}_j$ is a unit vector in which the jth element is one, and ε is a sufficiently small real number.

4.3.3. Cycle Time Sensitive Profit Index

(1) Definition

The *Cycle Time Sensitive Profit* (CTSP) index is used in the DCBS heuristic for the policy to select an appropriate product type whose part-releasing rate is to be increased. In brief, the CTSP index system quantifies the priority in selecting the safest product type to increase its rate. The increment in a rate is limited by the cycle time constraints. Therefore, in the decision to choose a product, the slack cycle times are regarded as resources for the rate increase. The increasing part-releasing rates tend to draw a steep curve in the cycle times.

A description of CTSP reveals three components:

- *Unit profit*: The unit profit of product k is given by $p_k \alpha_{kL_k} rb_k$ considering cumulative yield and part-releasing batch size. If a product has a high unit profit, it is more likely to be select for increase.

- *Slack cycle time*: The slack cycle time (SCT) is the allowable cycle time subtracted by the current total average cycle time of the product type, i.e., $SCT_k = ACT_k - TCT_k$. Typically, the slack is consumed as the part-releasing rates increase. At the time a product meets its cycle time limitation, i.e., its slack time is zero, the heuristic does not further increase the part-releasing rate. Therefore, the more slack time a product has, the more it is likely to have a higher allowed in-flow rate.

- The partial differential coefficient of total cycle time, with respect to the part-releasing rate, is given by $\partial TCT_{k'}(\mathbf{x},\mathbf{y})/\partial\hat{x}_k$. This coefficient denotes, intuitively speaking, how much the total cycle time (TCT) increases with a unit increase in the part-releasing rate. The higher is the coefficient the more is slack consumed, so the product with a smaller coefficient value is recommended for selection.

Incorporating the effects of the above three components of the TCT of product k', the unit increase of the part-releasing rate of product k is given by

$$CTSP_k = p_k \alpha_{kL_k} rb_k \min_{k'} \left(SCT_{k'} \Big/ \frac{\partial TCT_{k'}(\hat{\mathbf{x}},\mathbf{y})}{\partial \hat{x}_k} \right). \tag{4.4}$$

Since the increase of in-flows is limited by the first-met constraints, $\min_{k'} \left(SCT_{k'} \Big/ \frac{\partial TCT_{k'}(\hat{\mathbf{x}},\mathbf{y})}{\partial \hat{x}_k} \right)$ could be a limit on the in-flow of product k . Multiplying by $p_k \alpha_{kL_k} rb_k$, one can quantify the profit from the possible increase of product type k . The product with the largest CTSP index is considered to be the most profitable product to manufacture at a higher rate.

$$k^c = \arg\max_k CTSP_k \tag{4.5}$$

(2) Evaluation

We perform numeric differentiations as follows.

$$\frac{\partial TCT_{k'}(\hat{\mathbf{x}}, \mathbf{y})}{\partial \hat{x}_k} = \frac{TCT_{k'}(\hat{\mathbf{x}} + \mathbf{u}_k \cdot \delta, \mathbf{y}) - TCT_{k'}(\hat{\mathbf{x}}, \mathbf{y})}{\delta}$$

where, $\mathbf{u}_k$ is a unit vector in which the k th element is one, and δ is a sufficiently small real number.

(4.6)

4.3.4. Heuristic summary

(1) Initialization Phase

(1.1) Set the number of look-ahead iterations $m_{lookahead}$, the number of look-aheads of the part-releasing rate $n_{lookahead}$, and the rate-increment step ε.[21]

(1.2) Solve the Max-Profit-Capacity-Feasible problem (MPCF), of which the solution is simply $\hat{x}_k^* = TH_k / \alpha_{kL_k} rb_k$, $k = 1,2,\cdots,K$ and $y_i^* = \max(\lfloor \hat{\Lambda}_i t_i / \rho_{\max} \rfloor, m_i)$, $i = 1,2,\cdots,N$.

(1.3) Set $z^* = z(\hat{\mathbf{x}}^*, \mathbf{y}^*)$.

[21] The increment step size ε can be modeled differently for different product types, i.e., ε_k, $1 \le k \le K$.

(2) Iteration Phase 1

(2.1) Set $\hat{\mathbf{x}} = \hat{\mathbf{x}}^*$, $\mathbf{y} = \mathbf{y}^*$, $cm_{lookahead} = 0$, and $S = \{j \mid 1,2,\cdots,N\}$.

(2.2) Evaluate the DTCT index, i.e., $DTCT_j$, $j = 1,2,\cdots,N$.

(2.3) Obtain the station that has the least value, i.e., $j^* = \arg\min_{j \in S} DTCT_j$.

(2.4) If the investment constraint allows an increment in tools in station j^*,

update $y_j = y_j + 1$.

Else,

$S = S \setminus \{j^*\}$ and

if S is empty,

stop.

Else,

go to (2.3).

(2.5) If cycle time constraints are still violated, go to (2).

(3) Iteration Phase 2

(3.1) Set $P = \{1,2,\cdots,K\}$ and the current number of look-aheads of the part-releasing rate $cn_{lookahead} = 0$.

(3.2) Evaluate the CTSP index, i.e., $CTSP_k$, $k = 1,2,\cdots,K$.

(3.3) Obtain the product type that has the largest value, i.e., $k^* = \arg\max_{k \in P} CTSP_k$

(3.4) If current $\hat{\mathbf{x}}$ and $\mathbf{y}$ satisfies the cycle time constraints, mark $\hat{\mathbf{x}}^* = \hat{\mathbf{x}}$ and update $cn_{lookahead} = 0$. Else, update $cn_{lookahead} = cn_{lookahead} + 1$.

(3.5) Increase the part releasing rate, i.e., $\hat{x}_{k^*} = \hat{x}_{k^*} + \varepsilon$.

(3.6) If $cn_{lookahead} < n_{lookahead}$ and $\rho(\hat{\mathbf{x}}, \mathbf{y}) \le \rho_{\max}$, go to (3.4).

(3.7) If $z(\hat{\mathbf{x}}^*, \mathbf{y}^*) > z^*$,

update $z^* = z(\hat{\mathbf{x}}^*, \mathbf{y}^*)$ and go to (2.1).

Else,

$cm_{lookahead} = cm_{lookahead} + 1$,

$S = S \setminus \{j^*\}$ and

if S is empty,

stop.

Else,

go to (2.3).

4.4. Modified Simulated Annealing for MINLP

4.4.1. Simulated Annealing algorithm for continuous variables

The basic idea of the Simulated Annealing (SA) algorithm originates in the analogy of liquids freezing or metals recrystalizing in the process of annealing. A cooling process controls melting to be structurally ordered, and to slowly approach a thermodynamic equilibrium at a "frozen" ground level at a temperature $T = 0$. When a system has too low an initial temperature or too abrupt cooling, it can form defects or freezing in metastable states, i.e., trapped in a local minimum energy state. Adopting this concept in an algorithm for global optimization, SA allows uphill moves under the control of a temperature parameter. At higher temperatures only the gross behavior of the cost function is relevant to the search. As temperature decreases, finer details can be developed to get a good final point. While the optimality of the final point cannot be guaranteed, the method is able to proceed toward better minima even in the presence of many local minima.

Corana (1987) presents a global optimum algorithm for functions of continuous variables, which is derived from the original SA algorithm in combinatorial optimization. A detailed description of SA for continuous variables is shown in Table 4.4. A clear contrast from the original SA algorithm is that every move of the solution point occurs inside the continuous domain, as seen in Step 1. New candidate points are generated around the current point $\mathbf{x}_i$ applying, in turn, continuous random moves along each coordinate direction. The new coordinate values are uniformly distributed in intervals centered around the corresponding coordinates of $\mathbf{x}_i$. Half the size of these intervals along each coordinate is recorded in the step vector $\mathbf{v}$. If the point falls outside the

definition domain, a new point is randomly generated until a point belonging to the definition domain is found (Corona, 1987).

Table 4.4 Simulated Annealing Algorithm for Continuous Variables (Corana, 1987)

Step 0 (Initialization)

Choose

A starting point $\mathbf{x}_0$.

A starting step vector $\mathbf{v}_0$.

A starting temperature T_0.

A terminating criterion ε and a number of successive temperature reductions to test for termination N_ε,.

A test for step variation N_s and a varying criterion $\mathbf{c}$.

A test for temperature reduction N_T and a reduction coefficient r_T.

Set i, j, m, k to 0. i is the index denoting successive points, j denotes successive cycles along every direction, m describes successive step adjustments, and k covers successive temperature reductions.

Set h to 1. h is the index denoting the direction along which the trial point is generated, starting from the last accepted point.

Compute $f_0 = f(\mathbf{x}_0)$.

Set $\mathbf{x}_{opt} = \mathbf{x}_0$, $f_{opt} = f_0$.

Set $n_u = 0,\ u = 1,2,\cdots,n$.

Set $f_u^* = f_0,\ u = 0,-1,\cdots,-N_\varepsilon + 1$.

Step 1

Starting from the point $\mathbf{x}_i$, generate a random point $\mathbf{x}'$ along the direction h:

$$\mathbf{x}' = \mathbf{x}_i + rv_{m_h}\mathbf{e}_h$$

where r is a random number generated in the range [-1, l] by a pseudorandom number generator; $\mathbf{e}_h$ is the vector of the h th coordinate direction; and v_{m_h} is the component of the step vector v_m along the same direction.

Table 4.4 (continued)

Step 2

If the h th coordinate of $\mathbf{x}'$ lies outside the definition domain of f, that is, if $\mathbf{x}'_h < a_h$ or $\mathbf{x}'_h < b_h$, then return to step 1.

Step 3

Compute $f' = f(\mathbf{x}')$.

If $f' \le f_i$ then accept the new point:

set $\mathbf{x}_{i+1} = \mathbf{x}'$,

set $f_{i+1} = f'$,

add 1 to i,

add 1 to n_h;

if $f' \le f_{opt}$, then set

$\mathbf{x}_{opt} = \mathbf{x}'$,

$f_{opt} = f'$.

endif;

else ($f' > f_i$) accept or reject the point with acceptance probability p (Metropolis move):

$$p = \exp\left(\frac{f_i - f'}{T_k}\right).$$

In practice, a pseudorandom number p' is generated in the range [0, l] and is compared with p. If $p' < p$, the point is accepted, otherwise it is rejected.

In the case of acceptance:

set $\mathbf{x}_{i+1} = \mathbf{x}'$,

set $f_{i+1} = f'$,

add 1 to i,

add 1 to n_h.

Step 4

Add 1 to h.

If $h \le n$, then go to step 1;

else set h to 1 and add 1 to j.

Table 4.4 (continued)

Step 5

If $j < N_s$, then go to step 1;

else update the step vector v_m:

for each direction u the new step vector component v'_u is

$$v'_u = v_{m_u}\left(1 + c_u \frac{n_u/N_s - 0.6}{0.4}\right) \qquad \text{if } n_u > 0.6N_s,$$

$$v'_u = \frac{v_{m_u}}{1 + c_u \dfrac{0.4 - n_u/N_s}{0.4}} \qquad \text{if } n_u < 0.4N_s,$$

$$v'_u = v_{m_u} \qquad \text{otherwise.}$$

Set $\mathbf{v}_{m+1} = \mathbf{v}'$,

set j to 0,

set n_u to 0, $u = 1, \cdots, n$,

add 1 to m.

The aim of these variations in step length is to maintain the average percentage of accepted moves at about one-half of the total number of moves. The rather complicated formula used is discussed at the end of this chapter. The c_u, parameter controls the step variation along each u th direction.

Step 6

If $m < N_T$, then go to step 1;

else, it is time to reduce the temperature T_k:

set $T_{k+1} = r_T \cdot T_k$,

set $f_k^* = f_i$,

add 1 to k,

set m to 0.

It is worth noting that a temperature reduction occurs every N_s. N_T cycles of moves along every direction and after N_T step adjustments.

Table 4.4 (continued)

Step 7 (terminating criterion)

If:

$$\left|f_k^* - f_{k-u}^*\right| \le \varepsilon, \quad u = 1, \cdots, N_\varepsilon$$

$$f_k^* - f_{opt}^* \le \varepsilon$$

then stop the search;

else:

add 1 to i,

set $\mathbf{x}_i = \mathbf{x}_{opt}$,

set $f_i = f_{opt}$.

Go to step 1.

4.4.2. Modified Simulated Annealing

Based on SA for continuous variables, an additional modification is applied to accommodate both the continuous real variables and the discrete integer variables.

Table 4.5 shows pseudocode for the modified parts of MSA for MINLP, i.e., the generation of randomized solution alternatives (Step 1) and feasibility testing (Step 2). In Step 1, MSA assigns a real variate from a random number generated by a uniform distribution from –1 to 1 for any continuous variable when the moving direction of a new point is along the continuous dimension. For a new point moving to any discrete dimension, it rounds the moving distance to obtain an integer variate. The feasibility testing in the original algorithm is a simple comparison of the random alternative with the corresponding lower and upper bounds since the original version assumed that there is no constraint. In OptiProfit, the feasible region D is determined by a number of complex nonlinear constraints, and each alternative is determined to be in the region so that it can

be accepted for the next procedure. Thus, MSA is applicable to any form of MINLP, no matter where the nonlinear part is located in the formulation.[22]

Table 4.5 Modified Steps in Modified Simulated Annealing for MINLP

Step 1 (Modified)

Starting from the point $\mathbf{x}_i = \left[\mathbf{x}_i^C \mid \mathbf{x}_i^D\right]$, generate a random point $\mathbf{x}' = \left[\mathbf{x}'^C \mid \mathbf{x}'^D\right]$ along the vector of the h th coordinate direction $\mathbf{e}_h = \left[\mathbf{e}_h^C \mid \mathbf{e}_h^D\right]$:

$$\mathbf{x}' = \mathbf{x}_i + \mathbf{e}'_h$$

$$\mathbf{e}'_h = rv_{m_h}\mathbf{e}_h^C + \left\langle rv_{m_h}\right\rangle \mathbf{e}_h^D$$

where, the superscript C means the vector with continuous variables, superscript D means the vector with discrete variables, $\langle x \rangle$ is the rounded value of a real number x at the first digit, r is a random number generated in the range [-1, 1] by a pseudorandom number generator, v_{m_h} is the component of the step vector $\mathbf{v}_m$ along the same direction.

Step 2 (Modified)

If $\mathbf{x}'$ lies outside the definition domain or feasible region of problem, D, return to Step 1.

[22] APPENDIX B contains the Mathematica™ code instance of MSA algorithm for OptiProfit.

CHAPTER 5

SOLUTION PERFORMANCE

5.1. Description of Test Cases

In order to quantify and compare the performance of the heuristics, a number of test cases with some randomized parameters are analyzed numerically. The test cases have several material flows and stations. The flows are deterministic and can be reentrant. The model has parameters for process batching, unit margin, unit cost, investment, required throughput, and mean and deviation information. However, the effects of setup, yield, and existing tools to simplify basic formulations are not included. This does not incur any fundamental loss of model structure. Figure 5.1 shows an example with two arrival flows and three stations.[23]

Sixty non-trivial cases are tested, three groups with 20 cases each. The number of processing steps is fixed for each group. The randomized parameters include the allowable investment cost, allowable total cycle time, average processing times, SCV of processing time, unit profit coefficients, unit cost coefficients, and required throughputs. Note that the uniform random function, UNIF, has somewhat different parameters in order to effectively produce feasible and non-trivial cases with respect to the size of test groups. Table 5.1 exhibits the specifications of the test groups.

The upper bound is obtained from the OptiProfitUB model using a numerical solver GAMS/DICOPT for the convex programs.[24]

[23] For the detailed formulation of this example, see APPENDIX C.
[24] For example, the GAMS code for Case Group 3, i.e., five products, six stations, and seven steps, is presented in APPENDIX D.

Table 5.1 Major Model Parameters of 60 Numerical Cases in Three Groups

Major model parameters	Group 1 Case(3, 4, 5)	Group 2 Case(4, 5, 6)	Group 3 Case (5, 6, 7)
Number of test cases	20	20	20
Number of product types	3	4	5
Number of stations	4	5	6
Number of process steps	5	6	7
Unit profit coefficients	Randomized in UNIF[25, 45]	Randomized in UNIF[20, 40]	Randomized in UNIF[20, 40]
Unit cost coefficients	Randomized in UNIF[1.0,5.0]	Randomized in UNIF[1.0,4.0]	Randomized in UNIF[1.0,4.0]
Allowable investment	Randomized in UNIF[120.0, 160.0]	Randomized in UNIF[130.0, 160.0]	Randomized in UNIF[220.0, 250.0]
Allowable total cycle times	Randomized in UNIF[23, 26]	Randomized in UNIF[23, 26]	Randomized in UNIF[32, 36]
Required minimum throughputs	Randomized in UNIF[1.0, 2.0]	Randomized in UNIF[1.0, 2.0]	Randomized in UNIF[1.0, 2.0]
Average processing times	Randomized in UNIF[1.0, 3.0]	Randomized in UNIF[1.0, 3.0]	Randomized in UNIF[1.5, 2.5]
Processing time SCVs	Randomized in UNIF[0.0, 0.5]	Randomized in UNIF[0.0, 0.5]	Randomized in UNIF[0.2, 0.4]
Process batching size	2,2,3,3 for product types respectively	2,2,3,3,2 for product types respectively	2,2,3,3,2,2 for product types respectively
Parameters for MSA[25]	$\mathbf{v}_0$={0.1, 0.1, 0.1, 2, 2, 2, 2}, $T_0 = 10,\ \varepsilon = 0.1,$ $N_\varepsilon = 4,\ N_S = 10,$ $\mathbf{c}$ = {0.02, 0.02, 0.02, 0.01, 0.01, 0.01, 0.01}, $N_T = 100,\ r_T$ = 0.85	$\mathbf{v}_0$={0.1, 0.1, 0.1, 0.1, 2, 2, 2, 2, 2}, $T_0 = 10,\ \varepsilon = 0.1,$ $N_\varepsilon = 4,\ N_S = 10,$ $\mathbf{c}$ = {0.02, 0.02, 0.02, 0.02, 0.01, 0.01, 0.01, 0.01, 0.01}, $N_T = 100,\ r_T$ = 0.85	$\mathbf{v}_0$={0.1, 0.1, 0.1, 0.1, 0.1, 2, 2, 2, 2, 2, 2}, $T_0 = 10,\ \varepsilon = 0.1,$ $N_\varepsilon = 4,\ N_S = 10,$ $\mathbf{c}$ = {0.02, 0.02, 0.02, 0.02, 0.02, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01}, $N_T = 100,\ r_T = 0.85$

[25] The notation follows Corana (1987)

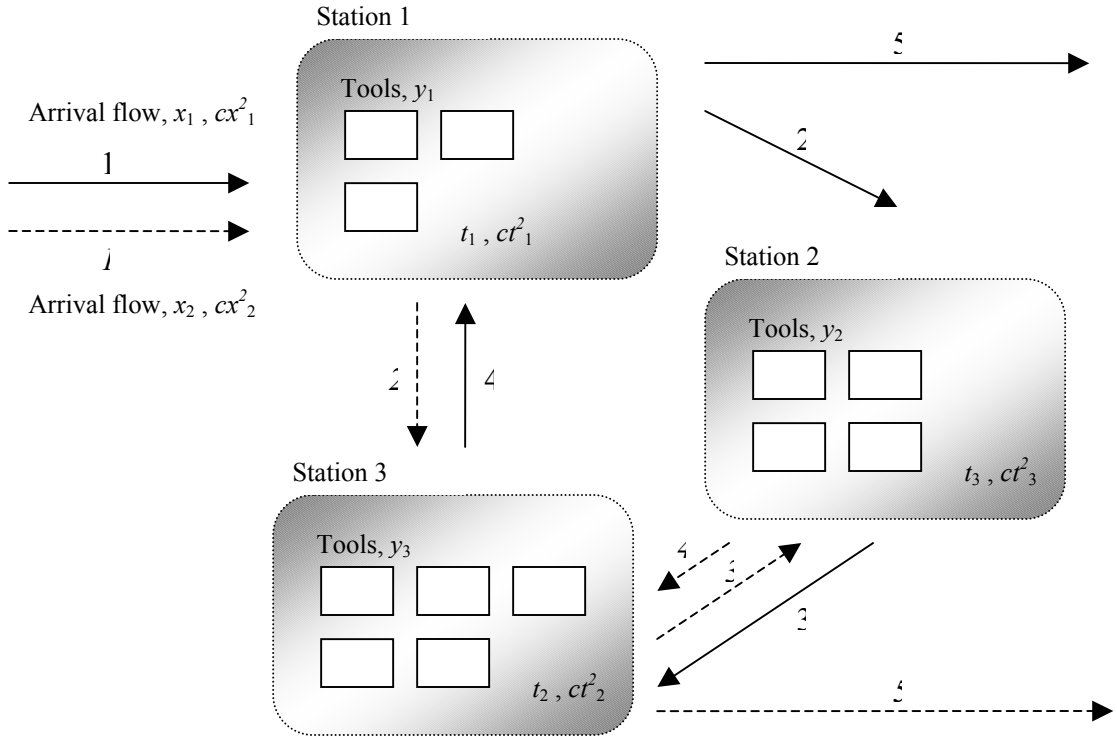


Figure 5.1 A Two-product Three-station Case

5.2. Performance Comparison

5.2.1. Performance measures

The calculations for all heuristics and MSA are performed on Mathematica™ version 5.0 on a personal computer with an Intel Pentium 3 processor at 2.4 GHz. On the same machine, GAMS/DICOPT for upper bound analysis was executed to generate the proved optimized solution in less than a few seconds. Since OptiProfitUB is a convex MINLP, DICOPT performed well and found the optimal value in most cases.

To evaluate the performance of the heuristics, we define the relative optimality gap (ROG) between the upper bound found from OptiProfitUB and the solution to evaluate its performance, as follows,

$$ROG = \frac{z^{UB^*} - z^*}{z^*} \times 100 \ (\%)$$

where, z^{UB^*} is the optimal value of OptiProfitUB and z^* is the final objective value determined by the corresponding heuristic.

(5.1)

Note that the upper bound solution is more, sometimes far more, than the true maximum value of OptiProfit. Therefore, the gap between a result from the heuristic and the optimum is well under the gap between the result and the upper bound. The heuristic evaluation time (HET) is obtained by a command program, `timeUsed[]`, on Mathematica™ version 5.0. For 60 cases in three groups, the objective values determined by heuristic and upper bound analysis are calculated.[26]

5.2.2. Results of test cases

From the average over the cases, it can be seen from Table 5.2 that DCBS performed better than any of the GAP-based heuristics – by approximately 2.89% over HQLS in Case (3,4,5), 4.72% over HUHP in Case (4,5,6), and 7.52% over HQHP in Case (5,6,7)). MSA performed well, just a few percentage points below DCBS. The performance of MSA tends to be dependent on the number of iterations with significant improvement in the solution at the expense of computational cost. Hence the tradeoff in performance and time consumption should be considered in implementing MSA, a meta-

[26] Detailed results are listed in APPENDIX E.

heuristic. The experiment applies to an observation of MSA in a specific situation and is not a statement that MSA performs inferior to any other method.

From the standard deviation of the results, it is found that MSA and DCBS are relatively more likely to give a stable result. More often, the basic GAP-based heuristics generate locally trapped solutions, e.g., the ROG ratio of 44.92% in HQLS, the 12th case of test group 1 in Table E.1[27]

Figure 5.3 shows the minimum, average, and maximum solutions of ROG in 20 test cases in each test group. DCBS has a few cases in which it falls behind other heuristics; however its average is best among the methods. The maximum of ROG means the worst solution the heuristic might find. DCBS shows the best performance in terms of the maximum error in all test groups. The standard deviation and gap between the maximum and minimum of ROG imply the stability of the solutions in the corresponding heuristic.

Figure 5.3 presents the ROG values with respect to three groups. Since the problem size is increasing in terms of the number of product types, stations, and step numbers in those groups, the sensitivity of performance can be roughly observed from the graph.[28] Again, DCBS as well as MSA show a stable performance over the test groups.

[27] The stability of results can be illustrated in a histogram as shown in APPENDIX E.

[28] Strictly speaking, the performance sensitivity with respect to problem size should be performed in accordance with the change in each dimension, e.g., the number of stations. In this work, the three groups have a simultaneous change in three dimensions.

Table 5.2 Summary of Performance Comparison

(a) Average ROG

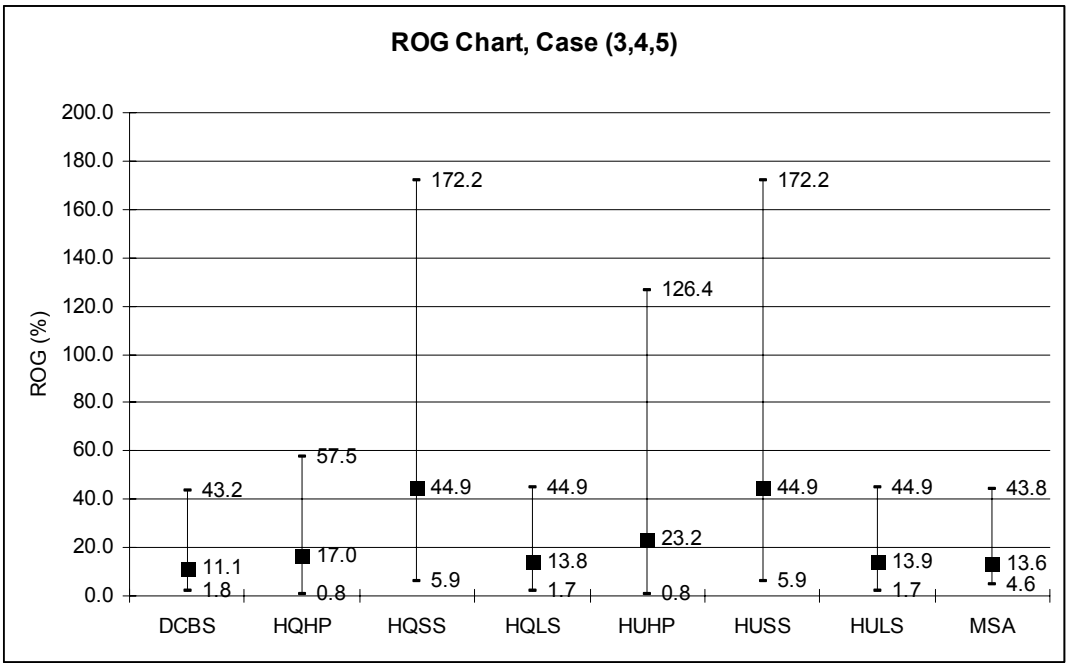
Test Group	DCBS	GAPs	MSA
Case (3,4,5)	11.06 %	13.82 % (HQLS) ~ 44.94 % (HUSS)	13.60 %
Case (4,5,6)	10.75 %	15.47 % (HUHP) ~ 48.35 % (HUSS)	16.31 %
Case (5,6,7)	11.03 %	18.65 % (HQHP) ~ 57.37 % (HQSS)	14.45 %

(b) Average HET

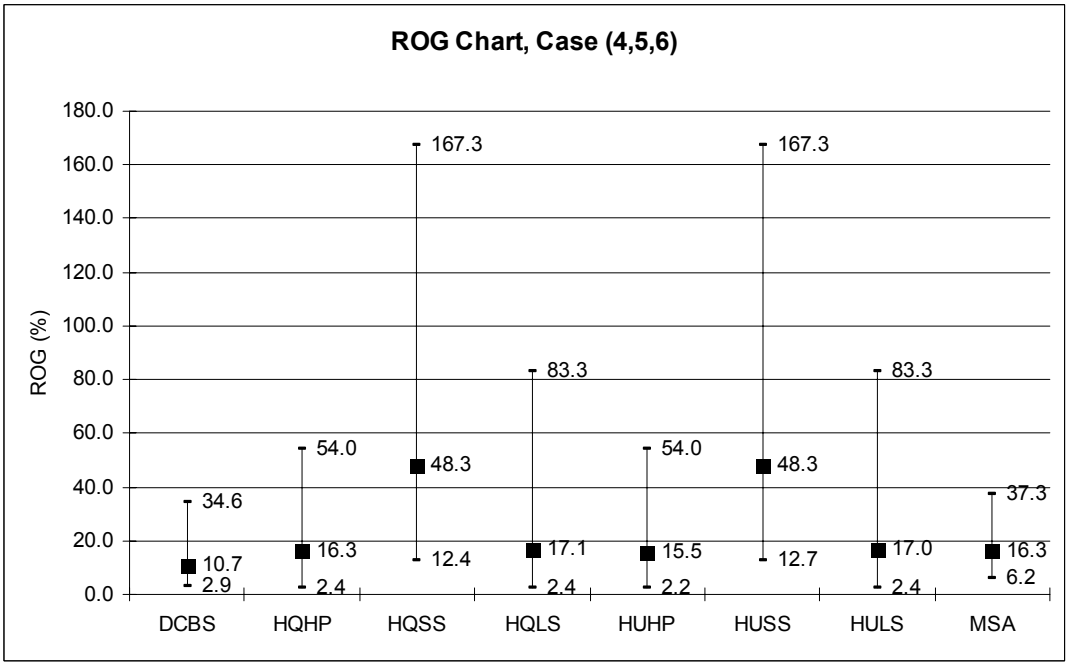
Test Group	DCBS	GAPs	MSA
Case (3,4,5)	10.97 sec	6.83 sec (HUSS) ~ 9.52 sec (HULS)	91.35 sec
Case (4,5,6)	17.32 sec	11.61 sec (HUSS) ~ 14.35 sec (HUHP)	203.78 sec
Case (5,6,7)	28.24 sec	8.36 sec (HQUU) ~ 19.13 sec (HULS)	456.12 sec

(c) Standard Deviation of ROG

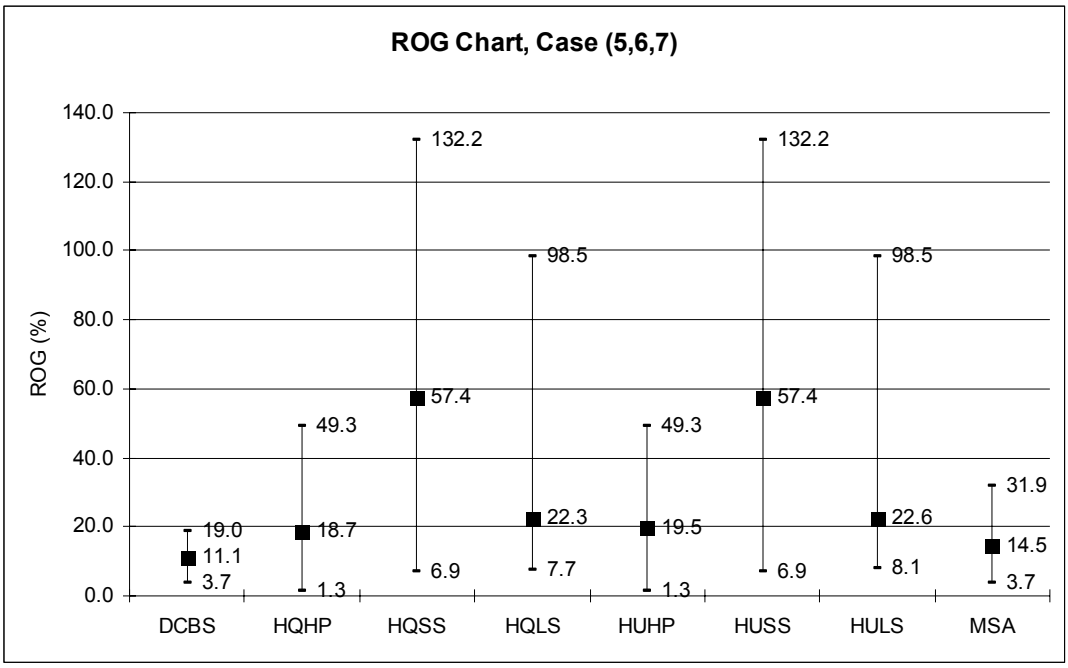
Test Group	DCBS	GAPs	MSA
Case (3,4,5)	8.53 %	11.46 % (HQLS) ~ 48.41 %(HQSS)	8.84 %
Case (4,5,6)	7.78 %	17.02 % (HQHP) ~ 36.80 % (HQSS)	7.85 %
Case (5,6,7)	4.60 %	13.75 % (HQHP) ~ 42.79 (HQSS)	8.02 %



(a) ROG Chart, Case (3,4,5)



(b) ROG Chart, Case (4,5,6)



(c) ROG Chart, Case (5,6,7)

Figure 5.2 Relative Optimality Gap Charts

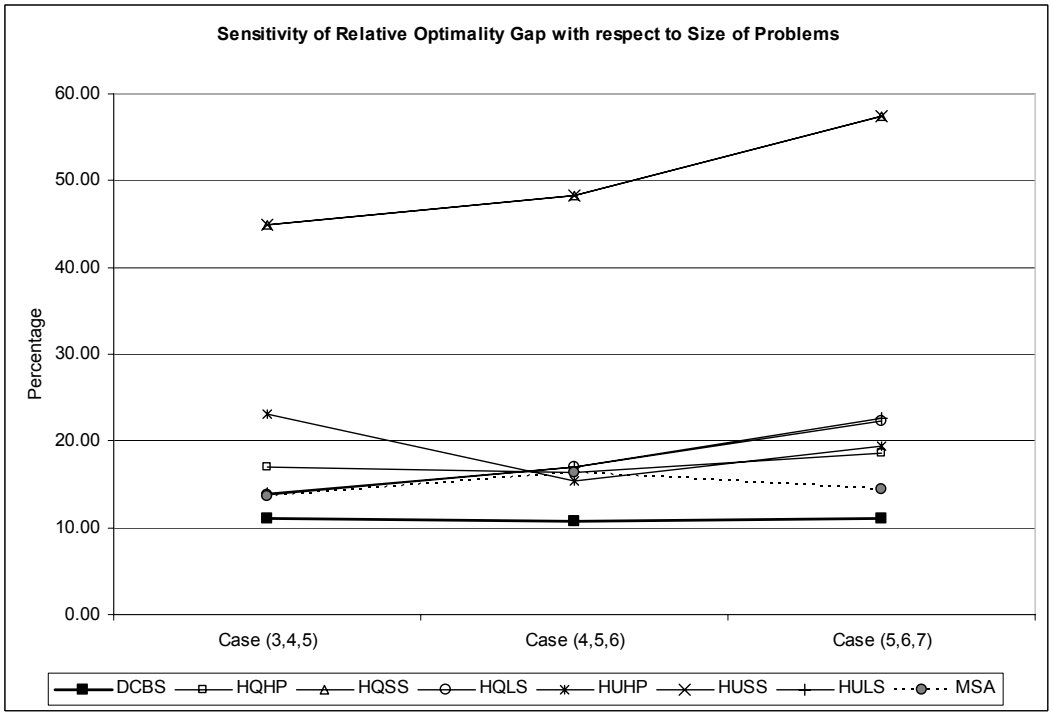


Figure 5.3 Average Relative Optimality Gap in Test Groups

5.2.3. Comparison from statistical inferences

One can observe if the results reject the following null hypothesis H_0 in the one-sided hypothesis testing problem.

H_0: The ROG of DCBS is not less than those of any other heuristics in each test group.

By statistical inference, if the null hypothesis is rejected, i.e., H_1 is accepted, the experimenter can conclude that there is evidence that DCBS has less ROG than any other heuristic in each test group.

For a testing measure we take the "paired t-test in one tail" between DCBS and all other heuristics, which makes seven t-tests. This is done to alleviate the effect of variabilities in a factor other than the difference between the two populations. The major assumptions in the paired t-test for this study are as follows.

(1) In each test group, the solutions of all heuristics are related, i.e., dependent scores. Since all solutions from heuristics are based on the same OptiProfit instance, they are not independent and should be compared in pairs.

(2) The scale of measurement is in terms of ratio, not ordinal. The scale of measurement is expressed in ROG, a ratio. Since there is no sequence or order in the randomized test cases, the data is not ordinal.

(3) The differences of the solutions in comparison have normal distributions or the number of samples is relatively large. This assumption is reinforced by the p-p plotting of the differences in pairs; one can confirm that the paired t-test is applicable to this study.

Table 5.3 lists the P-values of the paired t-tests in one tail among DCBS and each of other heuristic in 20 test cases in each test group.[29] With a significance level of α=0.10, the analysis of all tests reveals that H_0 is not plausible. That is, it provides evidence that DCBS is the most superior in each test group. For α=0.05, we find three P-values out of the rejection area $P \le \alpha$. Interestingly, however, with increasing problem size, most paired t-tests reveal that the performance of DCBS improves and is finally the best in the largest problem size group, test group 3. In particular, basic GAPs have a clear performance deterioration compared to DCBS with respect to the size of problem. In conclusion, we can claim that our data set provides evidence that DCBS outperforms all other heuristics.[30] In addition, DCBS is found to be more efficient in large-sized problems. Also, in the test with 60 sampling pairs, the null hypothesis is found to be rejected obviously with less than 0.003 P-values.

Table 5.3 P-values of Paired T-Tests

Heuristics	Group 1	Group 2	Group 3	All Groups
HQHP	0.052	0.029	0.012	0.000
HQSS	0.003	0.000	0.000	0.000
HQLS	0.099	0.047	0.020	0.002
HUHP	0.041	0.053	0.009	0.001
HUSS	0.003	0.000	0.000	0.000
HULS	0.099	0.051	0.017	0.001
MSA	0.017	0.007	0.027	0.000
Number of case pairs	20	20	20	60
Maximum p-value	0.099	0.053	0.027	0.002
H0 accepted heuristics(alpha =.05)	3	2	0	0
H0 accepted heuristics(alpha =.10)	0	0	0	0

[29] Detailed test information is in APPENDIX E.
[30] Drawn from 10% significance level tests.

5.3. Computational Costs

As expected, the computational cost is much higher for MSA compared with other heuristics. The heuristic methods are compared with MSA based on the computation time measured in seconds, as shown in Figure 5.4. All heuristics show approximately 3% to 8% of the computational cost of MSA. The most costly part is the evaluation of total cycle times. This includes complex nonlinear calculations and inverse operations on non-sparse $K \times K$ matrices. The number of cycle time evaluations (NCEs) is roughly proportional to the calculation time. Since DCBS does not necessarily guarantee the best result among the GAPs for every situation, it is not the best strategy to rely only on DCBS. It is recommended to test other heuristics as long as the sum of the computational costs is strategically acceptable.

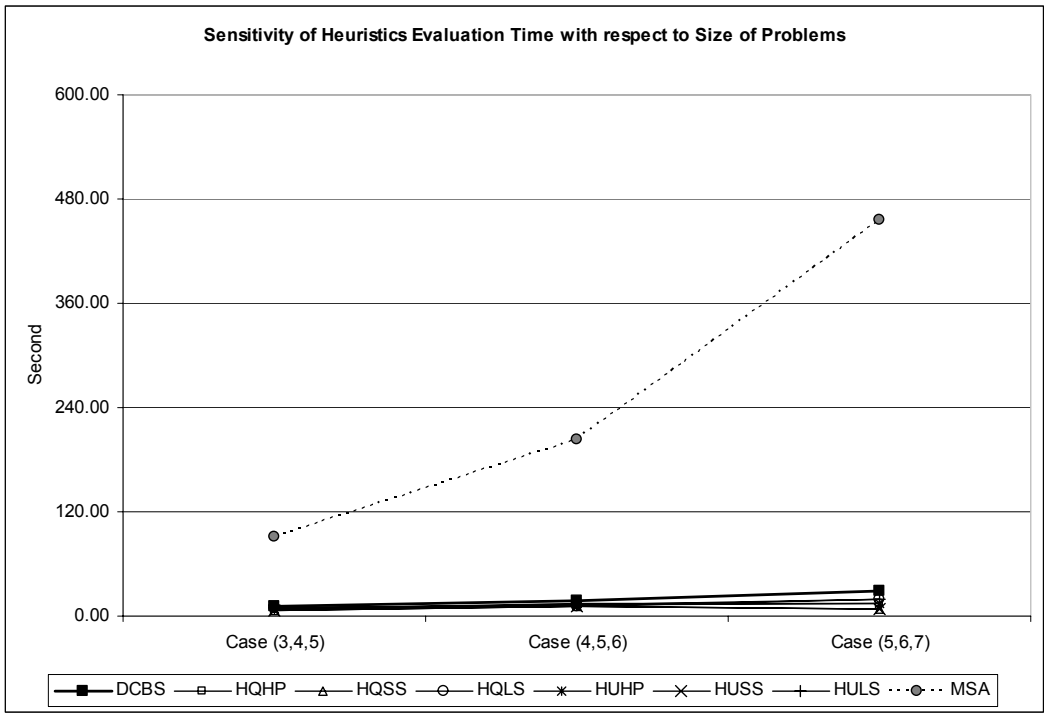


Figure 5.4 Average Heuristics Evaluation Time in Test Groups

CHAPTER 6

AN EXAMPLE: SEMICONDUCTOR MANUFACTURING

6.1. Semiconductor Manufacturing

Semiconductor manufacturing is a typical reentrant manufacturing system. Unlike non-reentrant manufacturing systems, such as automobile manufacturing [31], semiconductor manufacturing is one of the most advanced and complex types of manufacturing system. Generally, such a system has highly automated equipment and control modules, including automated material handling systems (AMHSs).

Figure 6.1 illustrates the major steps in semiconductor manufacturing; in a real situation a production facility can have hundreds of reentrant steps and decades of product types and stations. Typically, with a large amount of work in process, the average cycle time of products is estimated to be a couple of months. Motivated by the capital-intensive and process-complicated nature of the industry, much research has been directed at improving profit, investment, cycle time, and efficiency.

[31] In some cases, with reworking, automobile manufacturing may be reentrant.

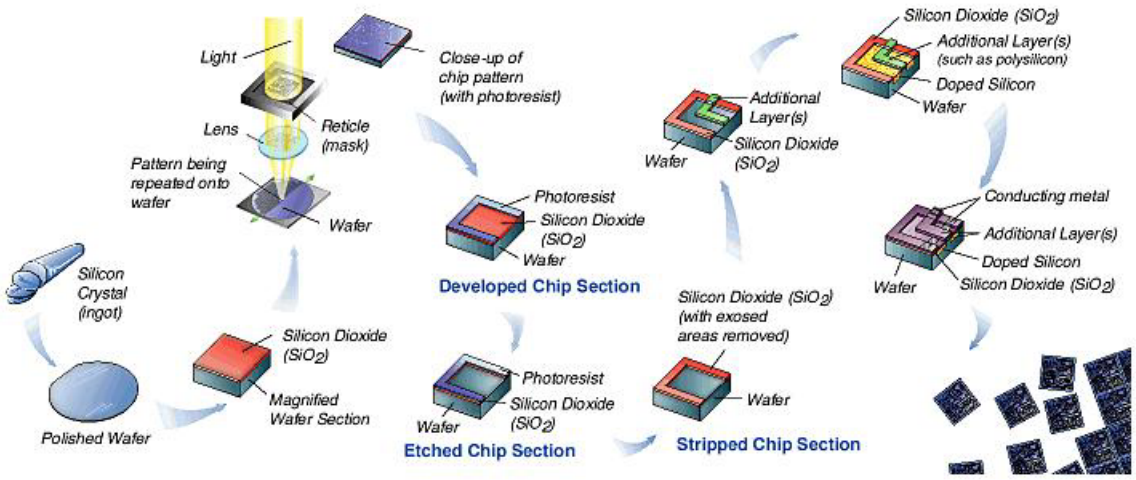


Figure 6.1 Semiconductor Production[32]

6.2. Problem Description

A small-sized fabrication facility is modeled and analyzed by DCBS in comparison with other basic GAPs. Table 6.1 summarizes the example specification and Table 6.2 describes the details of the model. The manufacturing system has three products and 12 stations.[33] The number of steps and allowable cycle times are different for the three products. The control schemes are primitive and fixed, such as UNIF for the part-releasing policy. Since the average cycle times, TCT_k, flatten the transportation times, they are assigned differently for each pair of stations, but are constant.

[32] Source: SEMATECH Inc. (www.sematech.org)
[33] The example is similar to one in the mini-fab model in Hopp (2002).

Table 6.1 Example Summary

Field of application	Semiconductor manufacturing (Minifab)
General properties of system	Reentrant flows
Number of products	3
Number of stations	12
Number of steps in recipes	20, 24, and 26 for each product type
Allowable cycle times	1000, 1100, 1200 for each product type
Allowable investment	14.00
Unit profit of product	3.8, 4.6, 6.2 for each product type
Unit cost of tool	0.25, 0.32, 0.12, 0.18, 0.65, 0.48, 0.32, 0.22, 0.15, 0.38, 0.28, 0.14 (Proclean, Laser, Alignment, Clean, Photo, Etch, Strip, Oxide, Mask, Nitride, Poly, Probe)
Yield, failures, setup information	Specified
Distributions for random variates	Normal and Gamma
Part releasing policy	UNIF (Uniform inter-release time)
Lot-selection (dispatching) policy	FIFO (First In First Out)
Transportation times	Constant
Transporter capacity	1

Table 6.2 Minifab Information

	Product 1	Product 2	Product 3
Index	1	2	3
Min TH	1.50	1.20	1.00

	Unit profit of product (K$/unit)		
Index	1	2	3
Pro./cost	3.8	4.6	6.2

	Unit cost of tool in station (K$/hour/unit)											
Index	1	2	3	4	5	6	7	8	9	10	11	12
Pro./cost	0.25	0.32	0.12	0.18	0.65	0.48	0.32	0.22	0.15	0.38	0.28	0.14

	Fab-in	Proclean	Laser	Alignment	Clean	Photo	Etch	Strip	Oxide	Mask	Nitride	Poly	Probe	Fab-out
Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Min batch size (MBS)	1	1	1	1	4	1	4	4	4	3	2	1	1	1
Prod. Sens. Batch							YES			YES	YES			
MTBF		800	600	600	600	500	500	600	600	600	800	800	800	
SCV		0.5	0.5	0.5	0.3	0.3	0.3	0.3	0.3	0.5	0.5	0.5	0.5	
Dev.		565.6854	424.2641	424.2641	328.6335	273.8613	273.8613	328.6335	328.6335	424.2641	565.6854	565.6854	565.6854	
MTTR		1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	
SCV		0.15	0.15	0.15	0.15	0.3	0.3	0.3	0.15	0.15	0.15	0.15	0.3	
Dev.		0.5809	0.5809	0.5809	0.5809	0.8216	0.8216	0.8216	0.5809	0.5809	0.5809	0.5809	0.8216	
Availability		0.9981	0.9975	0.9975	0.9975	0.9970	0.9970	0.9975	0.9975	0.9975	0.9981	0.9981	0.9981	
Yield		1	1	0.99	1	1	0.98	1	1	1	0.97	1	0.94	

Table 6.2 (continued)

Product 1	Proc. time	P.t. SCV	Setup time	S.t. SCV	Cum. yield
Preclean	10	0.06	5	0.06	1.0000
Laser	20	0.05	10	0.08	1.0000
Alighnment	25	0.04	10	0.04	0.9900
Clean	20	0.08	5	0.03	0.9900
Photo	15	0.12	10	0.25	0.9900
Etch	10	0.09	5	0.09	0.9702
Strip	5	0.09	2	0.09	0.9702
Oxide	25	0.04	10	0.04	0.9702
Mask	15	0.11	5	0.11	0.9702
Photo	25	0.12	10	0.25	0.9702
Etch	15	0.09	8	0.09	0.9508
Strip	10	0.09	5	0.09	0.9508
Clean	25	0.08	10	0.03	0.9508
Nitride	20	0.06	10	0.06	0.9223
Mask	15	0.11	5	0.11	0.9223
Photo	15	0.12	12	0.25	0.9223
Etch	20	0.09	10	0.09	0.9038
Strip	5	0.09	2	0.09	0.9038
Poly	20	0.12	10	0.12	0.9038
Probe	45	0.04	30	0.04	0.8496

Product 2	Proc. time	P.t. SCV	Setup time	S.t. SCV	Cum. yield
Preclean	12	0.06	5	0.06	1.0000
Laser	22	0.05	10	0.08	1.0000
Alighnment	30	0.04	15	0.04	0.9900
Clean	20	0.08	12	0.03	0.9900
Photo	12	0.12	5	0.25	0.9900
Etch	8	0.09	3	0.09	0.9702
Strip	5	0.09	2	0.09	0.9702
Oxide	20	0.04	10	0.04	0.9702
Mask	20	0.11	15	0.11	0.9702
Photo	20	0.12	15	0.25	0.9702
Etch	10	0.09	5	0.09	0.9508
Strip	15	0.09	5	0.09	0.9508
Clean	20	0.08	10	0.03	0.9508
Nitride	15	0.06	5	0.06	0.9223
Mask	15	0.11	10	0.11	0.9223
Photo	15	0.12	12	0.25	0.9223
Etch	25	0.09	15	0.09	0.9038
Strip	5	0.09	2	0.09	0.9038
Clean	25	0.08	12	0.03	0.9038
Mask	10	0.11	5	0.11	0.9038
Photo	20	0.12	12	0.25	0.9038
Etch	15	0.09	5	0.09	0.8858
Strip	5	0.09	3	0.09	0.8858
Probe	40	0.04	30	0.04	0.8326

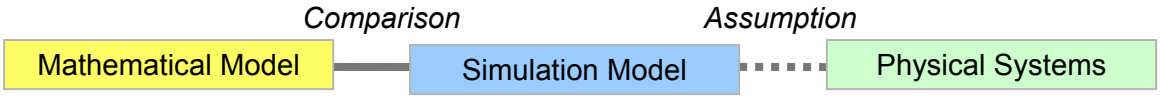
Product 3	Proc. time	P.t. SCV	Setup time	S.t. SCV	Cum. yield
Preclean	12	0.06	10	0.06	1.0000
Laser	20	0.05	12	0.08	1.0000
Alighnment	28	0.04	10	0.04	0.9900
Clean	20	0.08	5	0.03	0.9900
Photo	18	0.12	10	0.25	0.9900
Etch	10	0.09	5	0.09	0.9702
Strip	5	0.09	2	0.09	0.9702
Clean	24	0.08	15	0.03	0.9702
Nitride	25	0.06	10	0.06	0.9411
Mask	15	0.11	5	0.11	0.9411
Photo	18	0.12	10	0.25	0.9411
Etch	20	0.09	5	0.09	0.9223
Strip	5	0.09	3	0.09	0.9223
Oxide	20	0.04	10	0.04	0.9223
Mask	18	0.11	8	0.11	0.9223
Photo	20	0.12	10	0.25	0.9223
Etch	10	0.09	12	0.09	0.9038
Strip	5	0.09	2	0.09	0.9038
Clean	20	0.08	10	0.03	0.9038
Nitride	20	0.06	12	0.06	0.8767
Mask	15	0.11	8	0.11	0.8767
Photo	25	0.12	15	0.25	0.8767
Etch	10	0.09	10	0.09	0.8592
Strip	15	0.09	10	0.09	0.8592
Poly	20	0.12	5	0.12	0.8592
Probe	40	0.04	35	0.04	0.8076

6.3. Simulation Modeling for Cycle-time Verification

6.3.1. Simulation analysis

As illustrated in Figure 6.2, the ideal verification of the mathematical model should be based on real data from the physical system described. Many researchers, however, frequently encounter a lack of realistic data and rely on other experimental alternatives such as the simulation. A fundamental assumption is that the simulation model in use is equivalent or similar enough to represent the real system. However, no well-designed simulation can perfectly model a real system. Therefore, even though a mathematical model can produce similar solutions through simulation, it is not valid to assert that the mathematical model perfectly describes the real system. One can only claim that the mathematical model has a comparable accuracy to simulation. However, it does have the benefit of saving the high cost of simulation modeling and execution. In addition, it can be efficiently evaluated using numerical or heuristic algorithms.

(a) Ideal Verification



(b) Simulation Analysis

Figure 6.2 Model Verification

6.3.2. Case modeling

A simulation model using Arena™ has been designed for the verification of cycle time evaluation, which has several complex approximations and nonlinear formulations. The simulation model has the same level of detail as OptiProfit. Figure 6.3 presents a part of the simulation model for a station with product-type-sensitive batching. It describes four components of cycle time, batching, queuing, processing, and transporting time. Also, it has an identical specification of all deterministic and randomized design factors such as step information, yield, failure, batch size, etc. The simulation model for each station is composed of seven modules, or *block*s in Arena™. The first is "ENTER", which forms a queue incurring the waiting time for batching. The "BATCH" block generates either a product-type-sensitive batch or a non-product-type-sensitive batch. Once the batches go through a process queue in "SERVER", the time delay for processing takes place. The next blocks, "SPLIT", "CHOOSE", "ASSIGN", and "LEAVE", sort and dispatch the products in accordance with product type and route information.

Table 6.3 shows four cases of model configuration. The simulation model is built on Arena™ version 5.0 and executed with 20 replications for each of four configurations. Each is a snapshot of the iteration process of DCBS for a certain example. Case 1 shows the tool count configuration of an MPCF situation, i.e., the initialization phase. Cases 2, 3, and 4 are variations in fab-in rates and tool counts maintaining the feasibility.

6.3.3. Comparison results

Table 6.4 presents the differences in the cycle times between the analytic solution and the simulation analysis. For the error calculation, the following definition is used, assuming the average of two simulation results to be the true cycle time.

Simulation cycle time =

$\frac{1}{2}$ [(Result using Normal distributions) + (Result using Gamma distributions)],

Error of cycle time evaluation =

[(Analytical cycle time) – (Simulation cycle time)] / (Simulation cycle time)

(6.1)

We obtain fair results with less than 10% of error in cycle time evaluation, except for Case 1, which has the highest maximum station utilization. The MPCF configuration generally violates the cycle time constraints and has a very high average utilization compared to the feasible configurations in iteration steps. The error tends to be larger with the higher maximum station utilization. Also, it can be observed that the analytically evaluated cycle times are generally higher than the simulation results. The source of error can be traced to several causes including the nature of the approximation functions and the effect of reentrant flows. For example, OptiProfit uses a multi-class queuing network model with the G/G/m/inf queue approximation. The simulation analysis, using specific statistical distributions such as Normal or Gamma, tends to give smaller results for waiting time in queue when compared to the G/G/m/inf queue approximation. [34]

[34] Experimental observations using Normal and Gamma are shown in APPENDIX G.

Table 6.3 Testing Cases for Cycle-time Evaluation

(a) Case 1

	Product 1	Product 2	Product 3
Index	1	2	3
Fab-in rate	1.78	1.45	1.25
Fab-in SCV	0.00	0.00	0.00

	Proclean	Laser	Alignment	Clean	Photo	Etch	Strip	Oxide	Mask	Nitride	Poly	Probe	Average
Index	1	2	3	4	5	6	7	8	9	10	11	12	
Tool count	2	3	3	2	8	2	1	1	2	2	2	5	
Exp. util.	0.6579	0.7780	0.9742	0.7329	0.9320	0.6831	0.6598	0.5728	0.6912	0.6548	0.6268	0.9714	0.7446

(b) Case 2

	Product 1	Product 2	Product 3
Index	1	2	3
Fab-in rate	1.78	1.45	1.25
Fab-in SCV	0.00	0.00	0.00

	Proclean	Laser	Alignment	Clean	Photo	Etch	Strip	Oxide	Mask	Nitride	Poly	Probe	Average
Index	1	2	3	4	5	6	7	8	9	10	11	12	
Tool count	2	3	4	2	9	2	2	1	2	2	2	6	
Exp. util.	0.6579	0.7780	0.7307	0.7329	0.8284	0.6831	0.3299	0.5728	0.6912	0.6548	0.6268	0.8095	0.6747

Table 6.3 (continued)

(c) Case 3

	Product 1	Product 2	Product 3
Index	1	2	3
Fab-in rate	1.80	1.46	1.36
Fab-in SCV	0.00	0.00	0.00

	Proclean	Laser	Alignment	Clean	Photo	Etch	Strip	Oxide	Mask	Nitride	Poly	Probe	Average
Index	1	2	3	4	5	6	7	8	9	10	11	12	
Tool count	2	3	4	2	9	2	2	1	2	2	2	6	
Exp. util.	0.6820	0.8027	0.7529	0.7579	0.8573	0.7050	0.3415	0.5895	0.7145	0.6868	0.6510	0.8347	0.6980

(d) Case 4

	Product 1	Product 2	Product 3
Index	1	2	3
Fab-in rate	1.80	1.50	1.70
Fab-in SCV	0.00	0.00	0.00

	Proclean	Laser	Alignment	Clean	Photo	Etch	Strip	Oxide	Mask	Nitride	Poly	Probe	Average
Index	1	2	3	4	5	6	7	8	9	10	11	12	
Tool count	2	3	4	3	9	3	2	1	3	2	2	7	
Exp. util.	0.7500	0.8703	0.8144	0.5529	0.9388	0.5105	0.3745	0.6337	0.5213	0.7792	0.7120	0.7736	0.6859

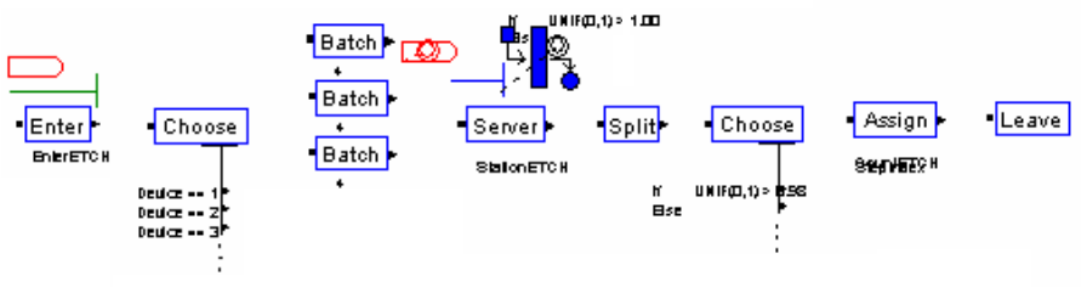


Figure 6.3 Simulation Model for a Station in Arena™

Table 6.4 Comparison of Cycle Times from Evaluation and Simulation

Case	Utilization			Cycle Times **Evaluated**, Simulated w/ Normal dist.'s, Simulated w/ Gamma dist.'s **Error in %**		
	Min	Ave	Max	Prod.1	Prod.2	Prod.3
1	0.573	0.745	0.974	**1126.4**, 960.0, 944.1 **18.32%**	**1305.4**, 1123.4, 1101.1 **17.40%**	**1408.6**, 1247, 1219.8 **14.22%**
2	0.330	0.675	0.828	**886.9**, 835, 849.5 **5.32%**	**1041.9**, 989.1, 991.5 **5.23%**	**1145.0**, 1104.8, 1113.2 **3.29%**
3	0.341	0.698	0.857	**901.2**, 843.5, 854.9 **6.16%**	**1058.7**, 993.6, 996.56 **6.43%**	**1152.2**, 1107.6, 1112.3 **3.84%**
4	0.375	0.686	0.939	**929.6**, 839.28, 850.85 **9.66%**	**1076.84**, 984.7, 982.5 **9.48%**	**1152.4**, 1070.2, 1076.4 **7.37%**

6.4. Heuristic Solutions

Six variants of the basic GAP heuristics and DCBS are applied to the semiconductor manufacturing example. Performance is compared in terms of final net profit obtained, i.e., objective value. Applying DCBS to the example problem, seven iterations provide a heuristic solution. The summary of the iteration information is presented in Table 6.5.

Table 6.5 Iterations in DCBS Heuristics

	Part releasing rate of products			Tool count in stations												Profit	Investment
	Product 1	Product 2	Product 3	Proclean	Laser	Alignment	Clean	Photo	Etch	Strip	Oxide	Mask	Nitride	Poly	Probe		
Initialization	1.780	1.450	1.250	2	3	4	2	9	2	1	1	2	2	2	6	9.074	12.11
Iteration 1	1.780	1.450	1.583	2	3	4	2	9	2	2	1	2	2	2	6	10.8186	12.43
Iteration 2	1.780	1.566	1.583	2	3	4	3	9	2	2	1	2	2	2	6	11.1722	12.61
Iteration 3	1.780	1.640	1.583	2	3	4	3	9	2	2	1	3	2	2	6	11.3626	12.76
Iteration 4	1.780	1.674	1.583	2	3	4	3	9	2	2	1	3	2	2	7	11.379	12.9
Iteration 5	1.780	1.674	1.914	2	3	4	3	10	2	2	1	3	2	2	7	12.7812	13.55
Iteration 6	1.780	1.806	1.914	2	4	4	3	10	2	2	1	3	2	2	7	13.0684	13.87
Iteration 7	1.780	1.844	1.914	2	4	5	3	10	2	2	1	3	2	2	7	13.1232	13.99

Table 6.6 shows the results of the basic heuristics and DCBS on the example problem. It is assumed that the basic GAP heuristics are executed in the same iteration framework as is DCBS. They differ only in station selection and production selection scheme, which are critical in the efficiency of the heuristic.

In selecting a station for tool increment, the HQ scheme tends to be superior to HU. In selecting a product-type for in-flow increment, SS and HP outperformed LS in this specific example, even though LS performed better in most cases discussed in CHAPTER 4. The choice of the production of highest-profit products only (HP) was not always a good way to increase the fab-in rates. As expected, the results show that DCBS generates a better solution, from 0.91 (7.45%) to 3.49 (26.60%) respectively.[35]

Table 6.6 Performance Comparison of DCBS with Basic Heuristics

Basic GAP Heuristics		Fab-in rates			Investment (UB: 14.00)	Obj. value
Station sel.	Prod. sel.	Prod. 1	Prod. 2	Prod. 3		
High util.	High profit	1.78	1.45	1.73	13.87	10.29
High util.	Large slack	2.42	1.45	1.25	13.99	9.63
High util.	Small slack	1.78	1.72	1.58	13.87	10.60
High q. t.	High profit	1.78	1.45	2.06	14.00	**12.21**
High q. t.	Large slack	2.92	1.45	1.25	14.00	11.52
High q. t.	Small slack	1.78	1.98	1.66	14.00	12.16

DCBS		Fab-in rates			Investment (UB: 14.00)	Obj. value
Station sel.	Prod. sel.	Prod. 1	Prod. 2	Prod. 3		
DTCT	CTSP	1.78	1.85	1.91	13.99	**13.12**

[35] The percentage difference is based on the less objective values, e.g., (13.12 – 12.21) / 12.21 = 7.45%.

CHAPTER 7

CONCLUSION

7.1. System-level Design of Large-scale Manufacturing Systems

This research focuses on an optimization model for operations capacity planning integrating the critical design factors of net profit, investment, cycle times, and throughputs at the initial design phase of large-scale manufacturing systems. The OptiProfit model is based on mixed integer non-linear programming. From the basic concept of the theory of constraints in cost accounting, the objective function, net profit, is constructed using margin and cost analysis. The cycle time evaluation, which includes complex non linear characteristics, requires a clear breakdown into four components: batching, queuing, processing, and transporting time. Mean-value analysis, queuing network models, and traffic variability equations are used to effectively evaluate the cycle times considering yield rates, batching effect, failure and repair, and variability aggregation.

OptiProfit is found to be intractable from its properties of NP-completeness, nonconvexity, and nonmonotonicity. The complexity classification of NP-completeness of OptiProfit is derived and proved from a reduction to the binary knapsack problem. By showing counterexamples, OptiProfit has constraints that are not always convex and monotone.

To handle the intractability, heuristics are considered. Based on an intuitive and practical approach, six variants of greedy ascent procedures and a modified meta-heuristic, MSA for MINLP, are introduced. A new heuristic approach, Differential

Coefficient Based Search is suggested to integrate the design factors such as profit, investment, cycle time, and throughput. A relaxed and convex version of OptiProfit, OptiProfitUB, provides an upper bound analysis and a quantified performance measure for the test heuristics in a number of numerically randomized cases. The heuristics are implemented and compared with exact solutions of OptiProfitUB in terms of the relative optimality gap. DCBS performs better than any other GAP-based heuristic. The performance of MSA is dependent on the number of iterations, which is proportional to the execution time cost, DCBS shows a superior result over MSA for OptiProfit problems.

Finally, a semiconductor manufacturing system with 12 stations and three product types is modeled as an OptiProfit problem. This model is successfully formulated and implemented, including the detailed system characteristics such as reentrant material flows. A simulation model at the same level of fidelity is constructed on Arena™ for the validation of cycle-time evaluation. The numerical results show that DCBS performs well in this specific example.

7.2. Next Steps

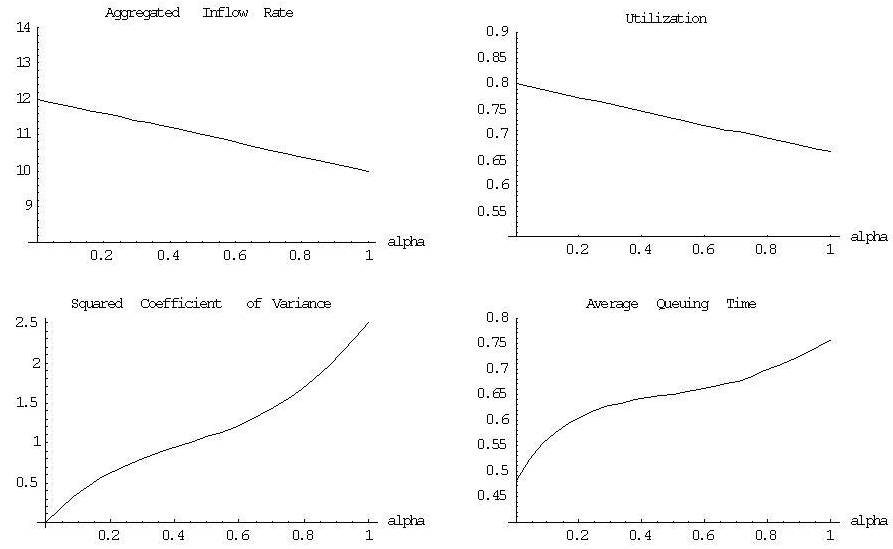
This research is conceived for the initial and system-level design and planning encompassing a wider spectrum of decision factors for manufacturing systems. The results can be used as a basic solution for the next analysis such as control policies, e.g., the part-releasing scheme. This research is expected to be an excellent beginning point for the study of control issues, which are significant decisions for performance improvement.

Other further work could include more realistic modeling of the objective function. With a consideration of pricing and cost fluctuations and exceptions, the objective can be nonlinear or piecewise linear. Finally, an elaborated financial analysis will enrich the model quality.

APPENDIX A

EXAMPLES FOR NONCONVEXITY AND NONMONOTONICITY ANALYSIS

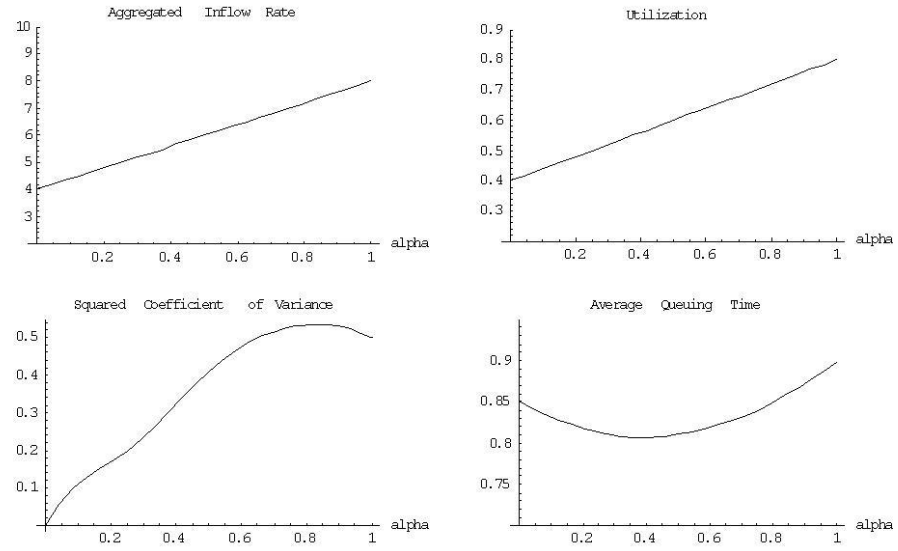
```
x11 = 0.; x12 = 12.; cr1 = 0.; x21 = 10.; x22 = 0.; cr2 = 2.5; b = 2; t = 0.4; ct = 0.; y = 3;
x1[a_] := x11 a + x12 (1 - a);
x2[a_] := x21 a + x22 (1 - a);
v[a_] := 1 / ((x1[a] / (x1[a] + x2[a]))^2 + (x2[a] / (x1[a] + x2[a]))^2);
w[a_] := 1 / (1 + 4 (1 - (x1[a] + x2[a]) t / y)^2 (v[a] - 1));
cx[a_] := 1 - w[a] + w[a] (cr1 x1[a] / (x1[a] + x2[a]) + cr2 x2[a] / (x1[a] + x2[a]));
u[a_] := (x1[a] + x2[a]) t / (b y);
TCT[a_] := b / (2 (x1[a] + x2[a])) + (cx[a] + ct) / 2 * (u[a]^(Sqrt[2 y + 1] - 1) / y / (1 - u[a])) t + t

Table[TCT[a], {a, 0, 1, 0.1}]

{0.483333, 0.561623, 0.604671, 0.628648, 0.642452, 0.651987, 0.661971, 0.676387, 0.69776, 0.725723, 0.756547}

Table[u[a], {a, 0, 1, 0.1}]

{0.8, 0.786667, 0.773333, 0.76, 0.746667, 0.733333, 0.72, 0.706667, 0.693333, 0.68, 0.666667}

Table[cx[a], {a, 0, 1, 0.1}]

{0., 0.365134, 0.614876, 0.796991, 0.941342, 1.07344, 1.22152, 1.41838, 1.69446, 2.06168, 2.5}

g21 = Plot[x1[a] + x2[a], {a, 0, 1}, PlotLabel → "Aggregated Inflow Rate", AxesLabel → {"alpha", ""}, PlotRange → {8.0, 14.0}]

g22 = Plot[TCT[a], {a, 0, 1}, PlotLabel → "Average Queuing Time", AxesLabel → {"alpha", ""}, PlotRange → {0.40, 0.80}]

g23 = Plot[u[a], {a, 0, 1}, PlotLabel → "Utilization", AxesLabel → {"alpha", ""}, PlotRange → {0.5, 0.9}]

g24 = Plot[cx[a], {a, 0, 1}, PlotLabel → "Squared Coefficient of Variance", AxesLabel → {"alpha", ""}]

Show[GraphicsArray[{{g21, g23}, {g24, g22}}]]
```



- GraphicsArray -

Figure A.1 Nonconvexity

```
x11 = 0.; x12 = 4.; cr1 = 0.; x21 = 8.; x22 = 0.; cr2 = 0.5; b = 2; t = 0.6; ct = 0.; y = 3;
x1[a_] := x11 a + x12 (1 - a);
x2[a_] := x21 a + x22 (1 - a);
v[a_] := 1 / ((x1[a] / (x1[a] + x2[a]))^2 + (x2[a] / (x1[a] + x2[a]))^2);
w[a_] := 1 / (1 + 4 (1 - (x1[a] + x2[a]) t / y)^2 (v[a] - 1));
cx[a_] := 1 - w[a] + w[a] (cr1 x1[a] / (x1[a] + x2[a]) + cr2 x2[a] / (x1[a] + x2[a]));
u[a_] := (x1[a] + x2[a]) t / (by);
TCT[a_] := b / (2 (x1[a] + x2[a])) + (cx[a] + ct) / 2 * (u[a]^(Sqrt[2y + 1] - 1) / y / (1 - u[a])) t + t

Table[TCT[a], {a, 0, 1, 0.1}]

{0.85, 0.832478, 0.818155, 0.80904, 0.806854, 0.810777, 0.819415, 0.832341, 0.84983, 0.872154, 0.898161}

Table[u[a], {a, 0, 1, 0.1}]

{0.4, 0.44, 0.48, 0.52, 0.56, 0.6, 0.64, 0.68, 0.72, 0.76, 0.8}

Table[cx[a], {a, 0, 1, 0.1}]

{0., 0.112558, 0.170912, 0.235604, 0.323142, 0.408983, 0.473977, 0.514825, 0.53339, 0.529995, 0.5}

g11 = Plot[x1[a] + x2[a], {a, 0, 1}, PlotLabel → "Aggregated Inflow Rate", AxesLabel → {"alpha", ""}, PlotRange → {2, 10}]

g12 = Plot[TCT[a], {a, 0, 1}, PlotLabel → "Average Queuing Time", AxesLabel → {"alpha", ""}, PlotRange → {0.70, 0.95}]

g13 = Plot[u[a], {a, 0, 1}, PlotLabel → "Utilization", AxesLabel → {"alpha", ""}, PlotRange → {0.2, 0.9}]

g14 = Plot[cx[a], {a, 0, 1}, PlotLabel → "Squared Coefficient of Variance", AxesLabel → {"alpha", ""}]

Show[GraphicsArray[{{g11, g13}, {g14, g12}}]]
```



- GraphicsArray -

Figure A.2 Nonmonotonicity

APPENDIX B

MODIFIED SIMULATED ANNEALING (MSA) USING MATHEMATICA™

```
ts = TimeUsed[];
Clear[x, y, metaTCT, TCT, CTq, rho, SCT, minCTq, maxCTq, maxProfit, minSCT,
maxSCT, xTEMP, yTEMP, zTEMP, slackINVEST, i, j, k, s]; numCal = 0;
dx = 0.01;
maxINCx = 10;
maxLookahead = 10;
gMaxLookahead = 3;
xINIT = TH;
yINIT = initConf[xINIT];
While[True,
    metaTCT = funcTCT[xINIT, yINIT];
    TCT = metaTCT[[1]];
    rho = metaTCT[[2]];
    If[tctViolated[TCT], , Break[]];
    maxRho = rho[[1]]; jc = 1;
    For[j = 2, j <= NE, j++,
      If[maxRho < rho[[j]], maxRho = rho[[j]]; jc = j]];
    yINIT = funcINCy[yINIT, jc, 1];
    ];
x[0] = Join[xINIT, yINIT];
Nx = NP + NE;
typeVector = Join[Table["Real", {NP}], Table["Integer", {NE}]];
For[s = 1, s <= Nx, s++; n[s] = 0];
v[0] = Join[Table[0.1, {NP}], Table[2, {NE}]];
T[0] = 10;
e = 0.1;
Ne = 4;
Ns = 10;
vc = Join[Table[0.02, {NP}], Table[0.01, {NE}]];
Nt = 100;
rt = 0.85;
i = 0; j = 0; m = 0; k = 0;
h = 1;
cuthalf[vec_, l_] := {Table[vec[[s]], {s, 1, l}],
      Table[vec[[s]], {s, l + 1, Length[vec]}]};
ev[h_] := Table[If[s == h, 1, 0], {s, 1, Nx}];
(*Start of SA *)
(
    (*Step 0*)
    f[0] = -funcObj[xINIT, yINIT];
    xopt = x[0]; fopt = f[0];
    For[u = 1, u <= Nx, u++, n[u] = 0];
    For[u = 0, u >= -Ne + 1, u--, fstar[u] = f[0]];
    (*Step 1*)
    Label[Step1];
    xprime = x[i];
    If[h <= NP, xprime[[h]] = x[i][[h]] + Random[Real, {-1, 1}] v[m][[h]]];
    If[h > NP,
      xprime[[h]] = x[i][[h]] + Round[Random[Real, {-1, 1}] v[m][[h]]]];
    (*Step 2*)
    xx = cuthalf[xprime, NP][[1]]; xy = cuthalf[xprime, NP][[2]];
    For[s = 1, s <= NE, s++, If[xy[[s]] < 1, Goto[Step1]]];
    For[s = 1, s <= NP, s++, If[xx[[s]] < TH[[s]], Goto[Step1]]];
    If[Sum[c[[s]] xy[[s]], {s, 1, NE}] > INVEST, Goto[Step1]];
    metaTCTprime = funcTCT[xx, xy];
    For[s = 1, s <= NP, s++,
      If[metaTCTprime[[1, s]] > ACT[[s]], Goto[Step1]]];
    For[s = 1, s <= NE, s++,
      If[metaTCTprime[[2, s]] > MAXUTIL, Goto[Step1]]];
    (*Step 3*)
    fprime = -funcObj[xx, xy];
    If[fprime <= f[i],
      x[i + 1] = xprime;
```

```
      f[i + 1] = fprime;
      i++;
      n[h]++;
      If[fprime < fopt,
        xopt = xprime;
        fopt = fprime;
        ],
      pprime = Random[Real, {0, 1}]; pMet = Exp[(f[i] - fprime)/T[k]];
      If[pprime < pMet,
        x[i + 1] = xprime;
        f[i + 1] = fprime;
        i++;
        n[h]++;
        ]
      ];
    (*Step 4*)
    h++;
    If[h <= Nx, Goto[Step1], h = 1; j++];
    (*Step 5*)
    If[j < Ns,
      Goto[Step1],
      vtemp = Table[0, {Nx}];
      For[u = 1, u <= Nx, u++,
        If[n[u] > 0.6 Ns,
          vtemp[[u]] = v[m][[u]] (1 + vc[[u]] (n[u]/Ns - 0.6)/0.4)];
        If[n[u] < 0.4 Ns,
          vtemp[[u]] = v[m][[u]] /(1 + vc[[u]] (0.4 - n[u]/Ns)/0.4)];
        If[0.4 Ns <= n[u] <= 0.6 Ns, vtemp[[u]] = v[m][[u]]];
        ];
      v[m + 1] = vtemp;
      j = 0;
      For[u = 1, u <= Nx, u++, n[u] = 0];
      m++;
      ];
    (*Step 6*)
    If[m < Nt, Goto[Step1],
      T[k + 1] = rt T[k];
      fstar[k] = f[i];
      k++;
      m = 0;
      ];
    (*Step 7*)
    Print["Checking termination condition"];
    isTerminiting = True;
    For[u = 1, u <= Ne, u++,
      If[Abs[fstar[k] - fstar[k - u]] <= e, , isTerminating = False]];
    If[fstar[k] - fopt <= e, , isTerminating = False];
    If[isTerminating, ,
      i++;
      x[i] = xopt;
      f[i] = fopt;
      Goto[Step1];
      ];
    );
(*End of SA*)
Print["Terminated============"];
Print["xopt"];
Print[xopt];
Print["fopt"];
Print[fopt];
xOptSA = xopt; zOptSA = fopt; numCalSA = numCal; timeSA = ts -
TimeUsed[];
```

APPENDIX C

FORMULATION FOR A SIMPLE CASE

(The system with 2 product types and 3 stations in Figure 5.1)

Objective function:

$$\text{Maximize } -\sum_{i=1}^{3} c_i y_i + \sum_{k=1}^{2} p_k x_k$$

Constraints:

$$TCT_k(\mathbf{x}, \mathbf{y}, \mathbf{a}) \le ACT_k,\ k = 1,2,\cdots,K$$

$$ca_j^2 = b_j + \sum_{i=1}^{N} a_{ij} ca_i^2,\ 1 \le j \le N,$$

$$\sum_{i=1}^{N} c_i y_i \le C$$

$$x_k \ge TH_k,\ k = 1,2,\cdots,K$$

$$y_i, \text{ Integer}$$

Inter-station flow rates:

$$\left[\lambda_{ij}\right] = \begin{bmatrix} 0 & x_1 + x_2 & 0 & 0 \\ x_1 & 0 & x_1 & x_2 \\ 0 & 0 & 0 & x_1 + x_2 \\ x_2 & x_1 & x_2 & 0 \end{bmatrix},\ i = 0,1,2,3,\ j = 0,1,2,3.$$

In-station flow rate:

$$\Lambda_j = \begin{bmatrix} x_1 + x_2 & 2x_1 + x_2 & x_1 + x_2 & x_1 + 2x_2 \end{bmatrix}$$

Minimum batch size:

$$\left[pb_j\right] = \begin{bmatrix} 2 & 2 & 3 \end{bmatrix}$$

Effective batch size:

$$\left[eb_{ij}\right]=\begin{bmatrix}0 & 2 & 2 & 3\\ 2 & 2 & 2 & 3\\ 2 & 2 & 2 & 3\\ 3 & 3 & 3 & 3\end{bmatrix}$$

Inter-station effective batch arrival rate:

$$\left[\hat{\lambda}_{ij}\right]=\left[\lambda_{ij}/eb_{ij}\right]=\begin{bmatrix}0 & \frac{x_1+x_2}{2} & 0 & 0\\ \frac{x_1}{2} & 0 & \frac{x_1}{2} & \frac{x_2}{3}\\ 0 & 0 & 0 & \frac{x_1+x_2}{3}\\ \frac{x_2}{3} & \frac{x_1}{3} & \frac{x_2}{3} & 0\end{bmatrix}$$

In-station effective batch arrival rate:

$$\hat{\Lambda}_j=\left[\sum_{i=0}^{N}\hat{\lambda}_{ij}\right]=\left[\frac{x_1}{2}+\frac{x_2}{3}\quad \frac{5x_1}{6}+\frac{x_2}{2}\quad \frac{x_1}{2}+\frac{x_2}{3}\quad \frac{x_1+2x_2}{3}\right]$$

Traffic Variability Equations:

$$a_{ij}=\frac{w_j}{\hat{\Lambda}_j}\left(\frac{\hat{\lambda}_{ij}}{\max(pb_i,pb_j)}\right)\frac{\lambda_i}{\hat{\Lambda}_i}q_{ij}(1-\rho_i^2),$$

$$b_j=1-w_j+\frac{w_j}{\hat{\Lambda}_j}\left(\frac{\lambda_{0j}}{eb_{0j}^2}\right)cx_j^2+\frac{w_j}{\hat{\Lambda}_j}\sum_{i=1}^{N}\frac{\lambda_{ij}}{eb_{ij}}\left[\frac{\lambda_i}{\hat{\Lambda}_i}q_{ij}\rho_i^2\phi_i+pb_i(1-q_{ij})\right],$$

$$q_{ij}=\lambda_{ij}\Big/\sum_{j'=0}^{N}\lambda_{ij'},\ \phi_i=1+(\max\{ct_i^2,0.2\}-1)/\sqrt{y_i},$$

$$w_j=\left[1+4(1-\rho_j)^2(v_j-1)\right]^{-1},\ v_j=\left[\sum_{i=0}^{N}(\hat{\lambda}_{ij}\big/\hat{\Lambda}_j)^2\right]^{-1},\ \rho_j=\hat{\Lambda}_j t_j\big/y_j.$$

APPENDIX D

GAMS CODE FOR A 5-STATION 6-PRODUCT 7-STEP CASE

```
* 5-Station 6-Product 7-Step Case
(1/20) *
* All randomized parameters have
been generated in Mathematica(TM) *

Sets
i product type / 1*5 /
j station / 1*6 /
k station (aux) / 1*6 /
l station including outer system /
0*6 /
m station including outer system
(aux)
/ 0*6 /

Scalar INVEST investment limitation
/ 239.369 / ;
Scalar MAXUTIL maximum utilization
of station / 0.99 / ;

Parameters

p(i) sales profit of product type i
/        1       25.4601
         2       27.0266
         3       22.9961
         4       34.7797
         5       39.7105   /
c(j) operation cost of station j
/        1       3.099
         2       1.31298
         3       3.87242
         4       3.13781
         5       3.91858
         6       2.48155   /
cx(j) arriving SCV (from outside of
system) to station j
/        1       0.0
         2       0.0
         3       0.0
         4       0.0
         5       0.0
         6       0.0     /
ca(j) arriving SCV to station j
/        1       0.0
         2       0.0
         3       0.0
         4       0.0
         5       0.0
         6       0.0     /
xb(j) arriving-from-outside batching
size of station j
/        1       1
         2       1
         3       1
         4       1
         5       0
         6       0       /
pb(j) process batching size of
station j
/        1       2
         2       2
         3       3
         4       3
         5       2
         6       2       /
t(j) average processing time of
station j
/        1       1.89393
         2       1.86584
         3       2.21652
         4       1.77928
         5       2.31486
         6       2.02615  /
ct(j) processing SCV of station j
/        1       0.232227
         2       0.207838
         3       0.279101
         4       0.391742
         5       0.252781
         6       0.371712  /
TH(i) required minimum throughput of
product type i
/        1       2.24986
         2       2.1857
         3       2.41258
         4       2.20876
         5       1.51087  /
ACT(i) allowable cycle time of
product type i
/        1       34.8007
         2       32.8516
         3       34.4177
         4       32.2136
         5       35.9503  / ;

Table eb(l,m) effective batching
size of station j
    0   1   2   3   4   5   6
0   1   2   2   3   3   2   2
1   2   2   2   3   3   2   2
2   2   2   2   3   3   2   2
3   3   3   3   3   3   3   3
4   3   3   3   3   3   3   3
5   3   2   2   3   3   2   2
6   2   2   2   3   3   2
2       ;

Variables
z total profit in unit period
x(i) part releasing rate
y(j) number of tool
lamda(l,m) inter-station flow rate
lamda_t(l) in-station flow rate
lamdah(l,m) inter-station flow rate
in effective batch
lamdah_t(l) in-station flow rate in
effective batch
et(j) effective processting time
ect(j) effective SCV processting
time
rho(j) rho value
TCT(i) cycle times ;

Positive variable x;
Integer variable y;
```

```
* Bounds *
x.lo(i) = TH(i) ;
y.lo('i') = 1 ;
rho.lo(j) = 0 ;
rho.up(j) = MAXUTIL ;
TCT.lo(i) = 0 ;
lamda.lo('0','0') = 0 ;
lamda.lo('0','1') = TH('1') ;
lamda.lo('0','2') = TH('2') +
TH('3') ;
lamda.lo('0','3') = TH('4') ;
lamda.lo('0','4') = TH('5') ;
lamda.lo('0','5') = 0 ;
lamda.lo('0','6') = 0 ;
lamda.lo('1','0') = TH('5') ;
lamda.lo('1','1') = 0 ;
lamda.lo('1','2') = TH('1') +
TH('2') ;
lamda.lo('1','3') = TH('3') +
TH('5') ;
lamda.lo('1','4') = 0 ;
lamda.lo('1','5') = TH('4') ;
lamda.lo('1','6') = 0 ;
lamda.lo('2','0') = 0 ;
lamda.lo('2','1') = TH('4') +
TH('5') ;
lamda.lo('2','2') = 0 ;
lamda.lo('2','3') = TH('2') ;
lamda.lo('2','4') = TH('1') +
TH('3') ;
lamda.lo('2','5') = TH('2') ;
lamda.lo('2','6') = 0 ;
lamda.lo('3','0') = TH('2') ;
lamda.lo('3','1') = TH('3') ;
lamda.lo('3','2') = TH('4') +
TH('5') ;
lamda.lo('3','3') = 0 ;
lamda.lo('3','4') = TH('1') ;
lamda.lo('3','5') = 0 ;
lamda.lo('3','6') = TH('4') +
TH('5') ;
lamda.lo('4','0') = 0 ;
lamda.lo('4','1') = 0 ;
lamda.lo('4','2') = 0 ;
lamda.lo('4','3') = TH('4') ;
lamda.lo('4','4') = 0 ;
lamda.lo('4','5') = TH('1') +
TH('5') ;
lamda.lo('4','6') = TH('1') +
TH('3') ;
lamda.lo('5','0') = 0 ;
lamda.lo('5','1') = TH('2') ;
lamda.lo('5','2') = 0 ;
lamda.lo('5','3') = TH('1') ;
lamda.lo('5','4') = TH('4') ;
lamda.lo('5','5') = 0 ;
lamda.lo('5','6') = TH('2') +
TH('5') ;
lamda.lo('6','0') = TH('1') +
TH('3') + TH('4');
lamda.lo('6','1') = TH('5') ;
lamda.lo('6','2') = 0 ;
lamda.lo('6','3') = TH('3') ;
lamda.lo('6','4') = 0 ;
lamda.lo('6','5') = TH('2') ;
lamda.lo('6','6') = 0 ;
lamda_t.lo(l) = sum(m,
lamda.lo(l,m)) ;
lamdah.lo(l,m) = lamda.lo(l,m) /
eb(l,m) ;
lamdah_t.lo(l) = sum(m,
lamdah.lo(l,m)) ;

Equations
profit define objective function

eq_lamda00 flow rate from station to
station
eq_lamda01 flow rate from station to
station
eq_lamda02 flow rate from station to
station
eq_lamda03 flow rate from station to
station
eq_lamda04 flow rate from station to
station
eq_lamda05 flow rate from station to
station
eq_lamda06 flow rate from station to
station
eq_lamda10 flow rate from station to
station
eq_lamda11 flow rate from station to
station
eq_lamda12 flow rate from station to
station
eq_lamda13 flow rate from station to
station
eq_lamda14 flow rate from station to
station
eq_lamda15 flow rate from station to
station
eq_lamda16 flow rate from station to
station
eq_lamda20 flow rate from station to
station
eq_lamda21 flow rate from station to
station
eq_lamda22 flow rate from station to
station
eq_lamda23 flow rate from station to
station
eq_lamda24 flow rate from station to
station
eq_lamda25 flow rate from station to
station
eq_lamda26 flow rate from station to
station
eq_lamda30 flow rate from station to
station
eq_lamda31 flow rate from station to
station
eq_lamda32 flow rate from station to
station
eq_lamda33 flow rate from station to
station
```

```
eq_lamda34 flow rate from station to
station
eq_lamda35 flow rate from station to
station
eq_lamda36 flow rate from station to
station
eq_lamda40 flow rate from station to
station
eq_lamda41 flow rate from station to
station
eq_lamda42 flow rate from station to
station
eq_lamda43 flow rate from station to
station
eq_lamda44 flow rate from station to
station
eq_lamda45 flow rate from station to
station
eq_lamda46 flow rate from station to
station
eq_lamda50 flow rate from station to
station
eq_lamda51 flow rate from station to
station
eq_lamda52 flow rate from station to
station
eq_lamda53 flow rate from station to
station
eq_lamda54 flow rate from station to
station
eq_lamda55 flow rate from station to
station
eq_lamda56 flow rate from station to
station
eq_lamda60 flow rate from station to
station
eq_lamda61 flow rate from station to
station
eq_lamda62 flow rate from station to
station
eq_lamda63 flow rate from station to
station
eq_lamda64 flow rate from station to
station
eq_lamda65 flow rate from station to
station
eq_lamda66 flow rate from station to
station
eq_lamda_t(l) flow rate in station
eq_lamdah(l,m) flow rate from
station to station
eq_lamdah_t(l) flow rate in station
eq_et1 equation effective processing
time
eq_et2 equation effective processing
time
eq_et3 equation effective processing
time
eq_et4 equation effective processing
time
eq_et5 equation effective processing
time
```

```
eq_et6 equation effective processing
time
eq_ect1 equation effective SCV
processing time
eq_ect2 equation effective SCV
processing time
eq_ect3 equation effective SCV
processing time
eq_ect4 equation effective SCV
processing time
eq_ect5 equation effective SCV
processing time
eq_ect6 equation effective SCV
processing time
eq_rho1 equation rho
eq_rho2 equation rho
eq_rho3 equation rho
eq_rho4 equation rho
eq_rho5 equation rho
eq_rho6 equation rho
eq_CT1 cycle time constraint
eq_CT2 cycle time constraint
eq_CT3 cycle time constraint
eq_CT4 cycle time constraint
eq_CT5 cycle time constraint
eq_TCT(i) cycle time constraints
eq_investment investment
constraint ;

profit .. z =e= sum(i, p(i)*x(i)) -
sum(j,c(j)*y(j)) ;

eq_lamda00 .. lamda('0','0') =e= 0 ;
eq_lamda01 .. lamda('0','1') =e=
x('1') ;
eq_lamda02 .. lamda('0','2') =e=
x('2') + x('3') ;
eq_lamda03 .. lamda('0','3') =e=
x('4') ;
eq_lamda04 .. lamda('0','4') =e=
x('5') ;
eq_lamda05 .. lamda('0','5') =e= 0 ;
eq_lamda06 .. lamda('0','6') =e= 0 ;
eq_lamda10 .. lamda('1','0') =e=
x('5') ;
eq_lamda11 .. lamda('1','1') =e= 0 ;
eq_lamda12 .. lamda('1','2') =e=
x('1') + x('2');
eq_lamda13 .. lamda('1','3') =e=
x('3') + x('5') ;
eq_lamda14 .. lamda('1','4') =e= 0 ;
eq_lamda15 .. lamda('1','5') =e=
x('4') ;
eq_lamda16 .. lamda('1','6') =e= 0 ;
eq_lamda20 .. lamda('2','0') =e= 0 ;
eq_lamda21 .. lamda('2','1') =e=
x('4') + x('5') ;
eq_lamda22 .. lamda('2','2') =e= 0 ;
eq_lamda23 .. lamda('2','3') =e=
x('2') ;
eq_lamda24 .. lamda('2','4') =e=
x('1') + x('3');
```

```
eq_lamda25 .. lamda('2','5') =e=
x('2') ;
eq_lamda26 .. lamda('2','6') =e= 0 ;
eq_lamda30 .. lamda('3','0') =e=
x('2') ;
eq_lamda31 .. lamda('3','1') =e=
x('3') ;
eq_lamda32 .. lamda('3','2') =e=
x('4') + x('5') ;
eq_lamda33 .. lamda('3','3') =e= 0 ;
eq_lamda34 .. lamda('3','4') =e=
x('1') ;
eq_lamda35 .. lamda('3','5') =e= 0 ;
eq_lamda36 .. lamda('3','6') =e=
x('4') + x('5');
eq_lamda40 .. lamda('4','0') =e= 0 ;
eq_lamda41 .. lamda('4','1') =e= 0 ;
eq_lamda42 .. lamda('4','2') =e= 0 ;
eq_lamda43 .. lamda('4','3') =e=
x('4') ;
eq_lamda44 .. lamda('4','4') =e= 0 ;
eq_lamda45 .. lamda('4','5') =e=
x('1') + x('5') ;
eq_lamda46 .. lamda('4','6') =e=
x('1') + x('3') ;
eq_lamda50 .. lamda('5','0') =e= 0 ;
eq_lamda51 .. lamda('5','1') =e=
x('2') ;
eq_lamda52 .. lamda('5','2') =e= 0 ;
eq_lamda53 .. lamda('5','3') =e=
x('1') ;
eq_lamda54 .. lamda('5','4') =e=
x('4') ;
eq_lamda55 .. lamda('5','5') =e= 0 ;
eq_lamda56 .. lamda('5','6') =e=
x('2') + x('5') ;
eq_lamda60 .. lamda('6','0') =e=
x('1') + x('3') + x('4') ;
eq_lamda61 .. lamda('6','1') =e=
x('5') ;
eq_lamda62 .. lamda('6','2') =e= 0 ;
eq_lamda63 .. lamda('6','3') =e=
x('3') ;
eq_lamda64 .. lamda('6','4') =e= 0 ;
eq_lamda65 .. lamda('6','5') =e=
x('2') ;
eq_lamda66 .. lamda('6','6') =e= 0 ;
eq_lamda_t(l) .. lamda_t(l) =e=
sum(m, lamda(l,m)) ;
eq_lamdah(l,m) .. lamdah(l,m) =e=
lamda(l,m) / eb(l,m) ;
eq_lamdah_t(l) .. lamdah_t(l) =e=
sum(m, lamdah(l,m)) ;
eq_et1 ..
lamdah_t('1')*pb('1')*et('1') =e=
lamda_t('1')*t('1');
eq_et2 ..
lamdah_t('2')*pb('2')*et('2') =e=
lamda_t('2')*t('2');
eq_et3 ..
lamdah_t('3')*pb('3')*et('3') =e=
lamda_t('3')*t('3');
eq_et4 ..
lamdah_t('4')*pb('4')*et('4') =e=
lamda_t('4')*t('4');
eq_et5 ..
lamdah_t('5')*pb('5')*et('5') =e=
lamda_t('5')*t('5');
eq_et6 ..
lamdah_t('6')*pb('6')*et('6') =e=
lamda_t('6')*t('6');
eq_ect1 ..
power(et('1'),2)*power(pb('1'),2)*la
mdah_t('1')*(ect('1')+1) =e=
power(t('1'),2)*(pb('1')*lamda_t('1'
)*ct('1')+sum(l,lamda(l,'1')*eb(l,'1
')));
eq_ect2 ..
power(et('2'),2)*power(pb('2'),2)*la
mdah_t('2')*(ect('2')+1) =e=
power(t('2'),2)*(pb('2')*lamda_t('2'
)*ct('2')+sum(l,lamda(l,'2')*eb(l,'2
')));
eq_ect3 ..
power(et('3'),2)*power(pb('3'),2)*la
mdah_t('3')*(ect('3')+1) =e=
power(t('3'),2)*(pb('3')*lamda_t('3'
)*ct('3')+sum(l,lamda(l,'3')*eb(l,'3
')));
eq_ect4 ..
power(et('4'),2)*power(pb('4'),2)*la
mdah_t('4')*(ect('4')+1) =e=
power(t('4'),2)*(pb('4')*lamda_t('4'
)*ct('4')+sum(l,lamda(l,'4')*eb(l,'4
')));
eq_ect5 ..
power(et('5'),2)*power(pb('5'),2)*la
mdah_t('5')*(ect('5')+1) =e=
power(t('5'),2)*(pb('5')*lamda_t('5'
)*ct('5')+sum(l,lamda(l,'5')*eb(l,'5
')));
eq_ect6 ..
power(et('6'),2)*power(pb('6'),2)*la
mdah_t('6')*(ect('6')+1) =e=
power(t('6'),2)*(pb('6')*lamda_t('6'
)*ct('6')+sum(l,lamda(l,'6')*eb(l,'6
')));
eq_rho1 .. rho('1') =e=
lamdah_t('1') * et('1') / y('1') ;
eq_rho2 .. rho('2') =e=
lamdah_t('2') * et('2') / y('2') ;
eq_rho3 .. rho('3') =e=
lamdah_t('3') * et('3') / y('3') ;
eq_rho4 .. rho('4') =e=
lamdah_t('4') * et('4') / y('4') ;
eq_rho5 .. rho('5') =e=
lamdah_t('5') * et('5') / y('5') ;
eq_rho6 .. rho('6') =e=
lamdah_t('6') * et('6') / y('6') ;
eq_CT1 .. TCT('1') =e=
*Step 1 1
         0.5*(pb('1')-
1)/lamda_t('1')+0.5*(ca('1')+ect('1'
))
```

```
*rho('1')**(sqrt(2*(y('1'))+1)-
1)/y('1')/(1-rho('1'))*et('1')
         +et('1')+
*Step 2 2
         0.5*(pb('2')-
1)/lamda_t('2')+0.5*(ca('2')+ect('2'
))

*rho('2')**(sqrt(2*(y('2'))+1)-
1)/y('2')/(1-rho('2'))*t('2')
         +et('2')+
*Step 3 4
         0.5*(pb('4')-
1)/lamda_t('4')+0.5*(ca('4')+ect('4'
))

*rho('4')**(sqrt(2*(y('4'))+1)-
1)/y('4')/(1-rho('4'))*et('4')
         +et('4')+
*Step 4 5
         0.5*(pb('5')-
1)/lamda_t('5')+0.5*(ca('5')+ect('5'
))

*rho('5')**(sqrt(2*(y('5'))+1)-
1)/y('5')/(1-rho('5'))*et('5')
         +et('5')+
*Step 5 3
         0.5*(pb('3')-
1)/lamda_t('3')+0.5*(ca('3')+ect('3'
))

*rho('3')**(sqrt(2*(y('3'))+1)-
1)/y('3')/(1-rho('3'))*et('3')
         +et('3')+
*Step 6 4
         0.5*(pb('4')-
1)/lamda_t('4')+0.5*(ca('4')+ect('4'
))

*rho('4')**(sqrt(2*(y('4'))+1)-
1)/y('4')/(1-rho('4'))*et('4')
         +et('4')+
*Step 7 6
         0.5*(pb('6')-
1)/lamda_t('6')+0.5*(ca('6')+ect('6'
))

*rho('6')**(sqrt(2*(y('6'))+1)-
1)/y('6')/(1-rho('6'))*et('6')
         +et('6') ;

eq_CT2 .. TCT('2') =e=
*Step 1 2
         0.5*(pb('2')-
1)/lamda_t('2')+0.5*(ca('2')+ect('2'
))

*rho('2')**(sqrt(2*(y('2'))+1)-
1)/y('2')/(1-rho('2'))*et('2')
         +et('2')+
*Step 2 5
         0.5*(pb('5')-
1)/lamda_t('5')+0.5*(ca('5')+ect('5'
))

*rho('5')**(sqrt(2*(y('5'))+1)-
1)/y('5')/(1-rho('5'))*et('5')
         +et('5')+
*Step 3 6
         0.5*(pb('6')-
1)/lamda_t('6')+0.5*(ca('6')+ect('6'
))

*rho('6')**(sqrt(2*(y('6'))+1)-
1)/y('6')/(1-rho('6'))*et('6')
         +et('6')+
*Step 4 5
         0.5*(pb('5')-
1)/lamda_t('5')+0.5*(ca('5')+ect('5'
))

*rho('5')**(sqrt(2*(y('5'))+1)-
1)/y('5')/(1-rho('5'))*et('5')
         +et('5')+
*Step 5 1
         0.5*(pb('1')-
1)/lamda_t('1')+0.5*(ca('1')+ect('1'
))

*rho('1')**(sqrt(2*(y('1'))+1)-
1)/y('1')/(1-rho('1'))*et('1')
         +et('1')+
*Step 6 2
         0.5*(pb('2')-
1)/lamda_t('2')+0.5*(ca('2')+ect('2'
))

*rho('2')**(sqrt(2*(y('2'))+1)-
1)/y('2')/(1-rho('2'))*et('2')
         +et('2')+
*Step 7 3
         0.5*(pb('3')-
1)/lamda_t('3')+0.5*(ca('3')+ect('3'
))

*rho('3')**(sqrt(2*(y('3'))+1)-
1)/y('3')/(1-rho('3'))*et('3')
         +et('3') ;

eq_CT3 .. TCT('3') =e=
*Step 1 2
         0.5*(pb('2')-
1)/lamda_t('2')+0.5*(ca('2')+ect('2'
))

*rho('2')**(sqrt(2*(y('2'))+1)-
1)/y('2')/(1-rho('2'))*et('2')
         +et('2')+
*Step 2 4
         0.5*(pb('4')-
1)/lamda_t('4')+0.5*(ca('4')+ect('4'
))
```

```
*rho('4')**(sqrt(2*(y('4'))+1)-
1)/y('4')/(1-rho('4'))*et('4')
         +et('4')+
*Step 3 6
         0.5*(pb('6')-
1)/lamda_t('6')+0.5*(ca('6')+ect('6'
))

*rho('6')**(sqrt(2*(y('6'))+1)-
1)/y('6')/(1-rho('6'))*et('6')
         +et('6')+
*Step 4 3
         0.5*(pb('3')-
1)/lamda_t('3')+0.5*(ca('3')+ect('3'
))

*rho('3')**(sqrt(2*(y('3'))+1)-
1)/y('3')/(1-rho('3'))*et('3')
         +et('3')+
*Step 5 1
         0.5*(pb('1')-
1)/lamda_t('1')+0.5*(ca('1')+ect('1'
))

*rho('1')**(sqrt(2*(y('1'))+1)-
1)/y('1')/(1-rho('1'))*et('1')
         +et('1')+
*Step 6 3
         0.5*(pb('3')-
1)/lamda_t('3')+0.5*(ca('3')+ect('3'
))

*rho('3')**(sqrt(2*(y('3'))+1)-
1)/y('3')/(1-rho('3'))*et('3')
         +et('3')+
*Step 7 6
         0.5*(pb('6')-
1)/lamda_t('6')+0.5*(ca('6')+ect('6'
))

*rho('6')**(sqrt(2*(y('6'))+1)-
1)/y('6')/(1-rho('6'))*et('6')
         +et('6') ;

eq_CT4 .. TCT('4') =e=
*Step 1 3
         0.5*(pb('3')-
1)/lamda_t('3')+0.5*(ca('3')+ect('3'
))

*rho('3')**(sqrt(2*(y('3'))+1)-
1)/y('3')/(1-rho('3'))*et('3')
         +et('3')+
*Step 2 2
         0.5*(pb('2')-
1)/lamda_t('2')+0.5*(ca('2')+ect('2'
))

*rho('2')**(sqrt(2*(y('2'))+1)-
1)/y('2')/(1-rho('2'))*et('2')
         +et('2')+
*Step 3 1
         0.5*(pb('1')-
1)/lamda_t('1')+0.5*(ca('1')+ect('1'
))

*rho('1')**(sqrt(2*(y('1'))+1)-
1)/y('1')/(1-rho('1'))*et('1')
         +et('1')+
*Step 4 5
         0.5*(pb('5')-
1)/lamda_t('5')+0.5*(ca('5')+ect('5'
))

*rho('5')**(sqrt(2*(y('5'))+1)-
1)/y('5')/(1-rho('5'))*et('5')
         +et('5')+
*Step 5 4
         0.5*(pb('4')-
1)/lamda_t('4')+0.5*(ca('4')+ect('4'
))

*rho('4')**(sqrt(2*(y('4'))+1)-
1)/y('4')/(1-rho('4'))*et('4')
         +et('4')+
*Step 6 3
         0.5*(pb('3')-
1)/lamda_t('3')+0.5*(ca('3')+ect('3'
))

*rho('3')**(sqrt(2*(y('3'))+1)-
1)/y('3')/(1-rho('3'))*et('3')
         +et('3')+
*Step 7 6
         0.5*(pb('6')-
1)/lamda_t('6')+0.5*(ca('6')+ect('6'
))

*rho('6')**(sqrt(2*(y('6'))+1)-
1)/y('6')/(1-rho('6'))*et('6')
         +et('6') ;

eq_CT5 .. TCT('5') =e=
*Step 1 4
         0.5*(pb('4')-
1)/lamda_t('4')+0.5*(ca('4')+ect('4'
))

*rho('4')**(sqrt(2*(y('4'))+1)-
1)/y('4')/(1-rho('4'))*et('4')
         +et('4')+
*Step 2 5
         0.5*(pb('5')-
1)/lamda_t('5')+0.5*(ca('5')+ect('5'
))

*rho('5')**(sqrt(2*(y('5'))+1)-
1)/y('5')/(1-rho('5'))*et('5')
         +et('5')+
*Step 3 6
         0.5*(pb('6')-
1)/lamda_t('6')+0.5*(ca('6')+ect('6'
))
```

```
*rho('6')**(sqrt(2*(y('6'))+1)-
1)/y('6')/(1-rho('6'))*et('6')
         +et('6')+
*Step 4 1
         0.5*(pb('1')-
1)/lamda_t('1')+0.5*(ca('1')+ect('1'
))

*rho('1')**(sqrt(2*(y('1'))+1)-
1)/y('1')/(1-rho('1'))*et('1')
         +et('1')+
*Step 5 3
         0.5*(pb('3')-
1)/lamda_t('3')+0.5*(ca('3')+ect('3'
))

*rho('3')**(sqrt(2*(y('3'))+1)-
1)/y('3')/(1-rho('3'))*et('3')
         +et('3')+
*Step 6 2
         0.5*(pb('2')-
1)/lamda_t('2')+0.5*(ca('2')+ect('2'
))

*rho('2')**(sqrt(2*(y('2'))+1)-
1)/y('2')/(1-rho('2'))*et('2')
         +et('2')+
*Step 7 1
         0.5*(pb('1')-
1)/lamda_t('1')+0.5*(ca('1')+ect('1'
))

*rho('1')**(sqrt(2*(y('1'))+1)-
1)/y('1')/(1-rho('1'))*et('1')
         +et('1') ;

eq_investment .. sum(j, c(j) * y(j))
=l= INVEST ;
eq_TCT(i) .. TCT(i) =l= ACT(i) ;

Model example_5_6 /all/ ;
solve example_5_6 using minlp
maximizing z ;
display x.l, x.m, y.l, y.m ;
```

APPENDIX E

RESULTS FROM HEURISTICS AND UPPER BOUND ANALYSIS

Table E.1 Result Table, 3-Product 4-Station 5-Step Case

DCBS																					**Avg**	**Dev**
z	120.68	164.89	668.57	240.15	321.2	103.44	279.13	131.72	139.38	253.85	580.16	135.14	335.52	345.22	199.54	119.38	217.42	81.225	555.68	244.09		
ROG	12.45	11.58	2.79	43.17	7.05	13.47	5.86	8.67	15.09	9.27	1.76	12.46	8.46	6.53	7.13	12.70	12.67	16.60	7.41	6.11	**11.06**	**8.53**
NCE	817	1236	2165	1424	1235	1148	1805	1402	1145	1269	2431	1096	1589	1482	1054	848	1134	586	2128	1080	**1354**	**472**
HET	6.659	10.715	16.173	11.207	9.975	9.875	14.471	11.286	9.534	10.415	19.347	8.913	13.229	12.038	8.522	7.511	8.732	4.647	16.754	9.394	**10.97**	**3.57**
HQHP																					**Avg**	**Dev**
z	118.53	170.49	622.3	242.65	334.97	100.78	293.04	134.18	133.37	240.59	575.7	138.64	231.08	334.72	137.16	120.8	213.13	82.39	484.96	251.81		
ROG	14.49	7.92	10.43	41.70	2.65	16.47	0.84	6.68	20.28	15.29	2.54	9.63	57.48	9.87	55.86	11.37	14.94	14.95	23.07	2.86	**16.97**	**16.33**
NCE	588	917	1560	1127	1009	886	1430	959	859	883	2035	877	744	1183	378	737	862	585	1480	883	**999**	**385**
HET	4.877	7.922	11.497	8.772	7.971	7.751	11.536	7.801	7.14	7.191	15.983	7.26	6.049	9.633	3.084	6.319	6.58	4.647	11.527	7.56	**8.06**	**2.90**
HQSS																					**Avg**	**Dev**
z	116.61	82.167	537.72	231.6	284.04	91.976	108.56	119.15	134.3	204.49	486.02	137.85	150.67	329.26	108.15	120.8	231.39	66.363	479.57	218.74		
ROG	16.38	123.91	27.80	48.46	21.05	27.61	172.19	20.14	19.45	35.64	21.47	10.25	141.52	11.70	97.68	11.37	5.87	42.72	24.45	18.40	**44.90**	**48.41**
NCE	555	180	1832	1095	667	823	411	838	919	1017	1836	848	360	1247	200	731	1091	277	1489	884	**865**	**482**
HET	4.457	1.562	13.519	8.573	5.348	7.16	3.305	6.81	7.511	8.372	14.451	7.03	2.914	10.085	1.633	6.269	8.532	2.173	11.686	7.241	**6.93**	**3.69**
HQLS																					**Avg**	**Dev**
z	128.56	163.21	675.47	252.53	315.18	101.22	245.47	128.15	138.77	258.52	576.57	104.88	352.04	340.42	171.94	117.24	213.55	77.637	580.08	251.81		
ROG	5.55	12.73	1.74	36.16	9.09	15.95	20.38	11.70	15.60	7.29	2.39	44.92	3.37	8.04	24.33	14.75	14.72	21.99	2.89	2.86	**13.82**	**11.46**
NCE	889	993	1880	1261	1042	1039	1383	1104	1000	1136	2033	998	1374	1206	1166	728	1000	611	1829	893	**1178**	**370**
HET	7.21	8.662	13.95	9.794	8.302	9.013	11.116	8.893	8.392	9.303	16.424	8.262	11.246	9.814	9.363	6.429	7.641	4.767	14.171	7.34	**9.50**	**2.77**
HUHP																					**Avg**	**Dev**
z	119.03	170.49	622.3	242.65	151.87	100.78	293.04	134.18	133.37	240.59	575.7	138.64	231.08	334.72	137.16	120.07	213.13	82.39	484.96	251.81		
ROG	14.01	7.92	10.43	41.70	126.39	16.47	0.84	6.68	20.28	15.29	2.54	9.63	57.48	9.87	55.86	12.04	14.94	14.95	23.07	2.86	**23.16**	**29.08**
NCE	604	910	1574	1135	145	865	1541	958	838	928	2029	865	744	1219	386	653	879	584	1491	883	**962**	**441**
HET	5.018	7.812	11.928	8.862	1.152	7.301	12.418	7.691	6.86	7.561	16.514	7.14	6.169	9.975	3.165	5.458	6.78	4.526	11.596	7.161	**7.75**	**3.48**
HUSS																					**Avg**	**Dev**
z	116.61	82.167	537.72	231.6	284.04	91.976	108.56	119.15	134.3	204.49	486.02	137.85	150.67	329.26	108.15	120.07	231.39	66.363	479.57	218.74		
ROG	16.38	123.91	27.80	48.46	21.05	27.61	172.19	20.14	19.45	35.64	21.47	10.25	141.52	11.70	97.68	12.04	5.87	42.72	24.45	18.40	**44.94**	**48.38**
NCE	584	180	1818	1080	659	813	408	838	899	1017	1856	801	360	1258	200	648	1054	290	1484	884	**857**	**482**
HET	4.736	1.482	13.569	8.412	5.267	6.89	3.305	6.87	7.421	8.412	14.791	6.56	2.885	10.264	1.572	5.438	8.192	2.274	11.537	7.22	**6.85**	**3.73**
HULS																					**Avg**	**Dev**
z	128.56	163.21	675.47	252.53	315.18	101.22	239.8	128.15	138.77	265.23	575.7	104.88	357.34	340.99	171.94	116.3	213.55	75.895	578.64	251.81		
ROG	5.55	12.73	1.74	36.16	9.09	15.95	23.23	11.70	15.60	4.58	2.54	44.92	1.83	7.85	24.33	15.68	14.72	24.79	3.14	2.86	**13.95**	**11.84**
NCE	907	993	1871	1319	1048	1018	1449	1104	1022	1079	2030	1014	1377	1222	1186	681	1012	577	1845	893	**1182**	**378**
HET	7.351	8.542	13.89	10.305	8.342	8.582	11.717	8.922	8.282	8.903	16.333	8.432	11.146	10.015	9.574	5.728	8.151	4.556	14.301	7.371	**9.52**	**2.84**
MSA																					**Avg**	**Dev**
z	117.28	166.08	618.33	239.03	328.6	102.05	258.53	120.5	132.23	252.29	543.7	122.69	345.04	337.71	198.22	118.78	232.57	83.733	525.93	242.82		
ROG	15.70	10.78	11.14	43.84	4.63	15.01	14.30	18.80	21.31	9.94	8.58	23.88	5.46	8.90	7.85	13.26	5.33	13.11	13.48	6.66	**13.60**	**8.84**
NCE	11030	10244	11674	11312	11974	11559	9290	10268	10136	10942	10683	10469	11609	10891	10354	11515	11235	11623	10039	11980	**10941**	**736**
HET	94.206	90.831	91.822	90.17	98.422	102.75	77.811	85.383	85.162	91.842	86.795	88.447	97.049	90.219	86.314	101.05	90.671	94.166	83.49	100.39	**91.35**	**6.41**
Objective value at upperbound																						
z	135.7	183.98	687.23	343.83	343.83	117.37	295.49	143.14	160.41	277.37	590.35	151.99	363.89	367.77	213.78	134.53	244.97	94.71	596.83	259		

Note: z: final objective value, ROG: relative optimality gap, NCE: Number of cycle time evaluation, HET: Heuristics evaluation time

Table E.2 Result Table, 4-Product 5-Station 6-Step Case

DCBS																					**Avg**	**Dev**
z	141.16	167.71	184.87	144.82	305.04	261.61	271.17	93.194	122.17	126.69	123.79	305.12	161.98	135.97	205.36	203.68	247.15	225.93	506.76	183.93		
ROG	7.71	6.09	2.89	5.08	13.01	17.77	6.84	11.07	18.91	4.94	34.61	5.97	6.01	11.53	15.59	4.27	4.85	16.50	3.36	17.99	**10.75**	**7.78**
NCE	743	1053	1250	1198	1730	1510	1373	898	1020	776	1549	1812	945	882	1260	1227	1069	1128	2659	1037	**1256**	**443**
HET	9.914	13.84	14.921	16.293	23.263	18.427	17.405	11.637	12.849	9.744	23.214	26.758	16.834	13.559	24.044	15.362	18.497	14.902	31.385	13.549	**17.32**	**5.74**
HQHP																					**Avg**	**Dev**
z	140.72	164.99	182.9	144.9	332.67	206.9	257.01	93.014	114.14	116.09	109.58	315.84	164.28	128.51	154.14	201.11	237.29	203.3	511	207.08		
ROG	8.04	7.84	4.00	5.02	3.62	48.90	12.72	11.28	27.28	14.52	52.07	2.37	4.53	18.01	54.00	5.60	9.21	29.46	2.51	4.79	**16.29**	**17.02**
NCE	500	852	1068	874	1553	958	959	635	670	606	1045	1289	800	680	500	850	800	553	1948	720	**893**	**362**
HET	6.58	11.206	12.708	12.008	20.9	12.198	12.107	8.162	8.422	7.711	15.762	19.077	13.99	11.206	9.594	10.676	14.09	7.17	22.733	9.354	**12.28**	**4.47**
HQSS																					**Avg**	**Dev**
z	134.85	158.31	132.65	105.46	128.96	206.9	223.22	89.229	108.67	96.204	105.5	162.1	148.59	110.33	172.37	124.19	200.01	139.64	427.83	136.66		
ROG	12.75	12.39	43.40	44.29	167.30	48.90	29.79	16.01	33.68	38.19	57.96	99.47	15.56	37.46	37.72	71.01	29.57	88.48	22.43	58.80	**48.26**	**36.80**
NCE	570	897	1191	878	1083	959	974	624	857	534	1154	597	768	610	946	554	815	195	1889	669	**838**	**347**
HET	7.801	11.917	14.181	11.716	14.892	12.037	12.278	8.071	10.926	6.8	17.365	8.802	13.38	9.483	18.246	6.9	15.041	2.584	22.643	8.692	**11.69**	**4.63**
HQLS																					**Avg**	**Dev**
z	142.01	137.54	184.52	142.68	288.45	168.08	280.67	88.379	112.67	105.31	122.93	315.84	147.86	139.87	219.45	201.3	246.64	234.21	487.02	187.24		
ROG	7.07	29.36	3.09	6.65	19.50	83.30	3.22	17.12	28.94	26.23	35.56	2.37	16.14	8.43	8.17	5.50	5.07	12.38	7.55	15.90	**17.08**	**18.45**
NCE	553	822	1291	1001	1534	437	1030	708	903	613	1380	1303	914	964	1108	1030	784	880	2460	828	**1027**	**437**
HET	6.759	10.415	11.196	11.467	19.458	11.576	12.719	8.262	8.152	7.932	11.987	19.358	9.964	13.109	9.814	10.866	15.092	7.23	23.955	10.405	**11.99**	**4.45**
HUHP																					**Avg**	**Dev**
z	140.72	164.99	182.9	144.9	331.97	206.9	257.03	93.014	117.81	116.09	109.58	315.84	164.28	143.65	154.14	201.11	237.29	203.3	512.75	206.87		
ROG	8.04	7.84	4.00	5.02	3.84	48.90	12.72	11.28	23.31	14.52	52.07	2.37	4.53	5.57	54.00	5.60	9.21	29.46	2.16	4.90	**15.47**	**17.06**
NCE	494	781	906	866	1463	936	1014	635	640	629	798	1295	694	720	499	870	835	524	1952	803	**868**	**353**
HET	8.022	10.846	15.532	13.619	21.03	5.378	12.938	9.153	11.486	7.781	20.75	19.348	13.509	16.824	21.381	13.068	14.271	11.567	29.722	10.746	**14.35**	**5.81**
HUSS																					**Avg**	**Dev**
z	134.85	152.67	132.65	105.46	128.96	206.9	223.22	89.229	108.88	96.204	105.5	162.1	148.59	110.33	173.58	124.19	200.01	139.64	431.92	136.66		
ROG	12.75	16.54	43.40	44.29	167.30	48.90	29.79	16.01	33.42	38.19	57.96	99.47	15.56	37.46	36.76	71.01	29.57	88.48	21.27	58.80	**48.35**	**36.66**
NCE	584	180	1818	1080	659	813	408	838	899	1017	1856	801	360	1258	200	648	1054	290	1484	884	**857**	**482**
HET	7.021	12.998	11.417	11.797	14.06	11.677	13.89	7.902	13.599	7.04	14.932	9.764	10.805	10.225	19.068	7.03	15.272	2.684	22.222	8.832	**11.61**	**4.47**
HULS																					**Avg**	**Dev**
z	141.77	139.25	183.27	142.68	289.12	168.08	280.57	88.379	117.81	105.88	126.39	315.84	137.81	128.66	219.45	203.68	246.64	228.4	499.75	199.2		
ROG	7.24	27.77	3.79	6.65	19.23	83.30	3.26	17.12	23.31	25.56	31.84	2.37	24.60	17.87	8.17	4.27	5.07	15.23	4.81	8.94	**17.02**	**18.19**
NCE	558	963	935	891	1070	937	1105	615	1026	561	977	663	731	622	997	569	841	192	1881	669	**840**	**336**
HET	7.27	12.298	11.406	11.306	20.269	5.418	13.379	9.223	8.693	8.512	19.338	19.498	11.507	14.351	19.468	11.927	14.15	11.757	23.674	10.906	**13.22**	**4.88**
MSA																					**Avg**	**Dev**
z	142.22	156.44	162.91	110.81	296.2	283.09	259.72	95.215	116.93	112.81	128.96	268.79	148.33	123.12	209.85	198.49	243.96	223.85	457.01	185.72		
ROG	6.90	13.73	16.77	37.33	16.38	8.83	11.55	8.71	24.24	17.85	29.22	20.29	15.77	23.17	13.12	6.99	6.23	17.58	14.61	16.84	**16.31**	**7.85**
NCE	16414	13367	13524	13267	14320	14820	14203	16366	13744	15041	11744	13238	14781	16808	14175	14093	14999	12872	15538	14047	**14368**	**1271**
HET	218.68	219.01	166.87	179.15	187.03	187.07	183.56	212	178.79	193.41	177.69	202.16	209.4	297.41	268.96	179.67	267.99	172.4	188.09	186.28	**203.78**	**35.66**
Objective value at upperbound																						
z	152.04	177.92	190.22	152.17	344.71	308.09	289.72	103.51	145.27	132.94	166.64	323.33	171.72	151.65	237.38	212.37	259.15	263.2	523.8	217.01		

Note: z: final objective value, ROG: relative optimality gap, NCE: Number of cycle time evaluation, HET: Heuristics evaluation time

Table E.3 Result Table, 5-Product 6-Station 7-Step Case

DCBS																					**Avg**	**Dev**
z	132.36	130.12	172.04	293.56	231.79	439.96	274.95	268.9	133.56	144.87	303.98	283.3	144.76	171.76	95.604	273.98	429.23	144.43	400.11	152.49		
ROG	9.49	16.94	12.75	5.54	7.60	4.19	18.96	16.78	9.97	15.53	13.86	14.12	7.89	3.66	12.15	9.43	16.90	5.79	12.54	8.63	**11.13**	**4.60**
NCE	917	410	741	2020	1290	2347	1688	2190	445	900	1720	1554	1076	615	441	2213	2623	1387	2653	1320	**1428**	**736**
HET	18.447	8.272	14.892	40.308	25.847	46.817	33.378	43.803	8.823	18.476	32.106	30.614	21.29	12.478	8.652	46.116	52.536	25.207	48.78	27.91	**28.24**	**14.42**
HQHP																					**Avg**	**Dev**
z	140.94	131.38	171.92	207.57	230.41	452.52	290.47	231.73	132	129.44	286.3	284.59	145.26	172.82	94.739	273.36	382.3	108.7	330.11	140.32		
ROG	2.82	15.82	12.82	49.26	8.24	1.29	12.60	35.52	11.26	29.31	20.89	13.61	7.52	3.02	13.17	9.68	31.25	40.55	36.40	18.05	**18.65**	**13.75**
NCE	499	338	562	408	903	1522	1259	1014	290	320	701	1075	668	321	292	1231	1581	417	1017	466	**744**	**424**
HET	10.014	6.77	11.276	8.101	17.986	30.404	25.116	20.239	5.838	6.579	13.339	21.271	13.299	6.379	5.779	24.966	31.475	7.47	18.707	9.944	**14.75**	**8.44**
HQSS																					**Avg**	**Dev**
z	121.76	128.73	155.39	196.89	203.98	197.41	152.89	135.25	132.55	117.26	278.55	199.7	101.84	166.48	95.452	159.8	227.98	84.259	236.17	122.3		
ROG	19.02	18.20	24.83	57.37	22.27	132.20	113.93	132.18	10.80	42.74	24.25	61.90	53.36	6.95	12.33	87.62	120.10	81.33	90.66	35.45	**57.37**	**42.79**
NCE	409	251	527	361	1114	220	199	393	244	314	759	650	217	369	311	294	701	263	639	240	**424**	**238**
HET	8.222	5.047	10.715	7.17	22.252	4.416	3.976	7.752	4.918	6.419	14.301	12.849	4.347	6.95	6.148	5.989	14.02	4.767	11.797	5.058	**8.36**	**4.66**
HQLS																					**Avg**	**Dev**
z	118.25	130.12	175.31	278.98	153.15	377.61	272.45	268.4	133.56	149.64	295.05	162.89	134.42	164.77	97.435	271.83	434.63	108.37	418.23	142.09		
ROG	22.55	16.94	10.64	11.06	62.85	21.39	20.05	17.00	9.97	11.85	17.30	98.48	16.18	8.06	10.04	10.29	15.45	40.99	7.66	16.58	**22.27**	**22.05**
NCE	481	282	537	1484	213	1528	1297	1445	302	779	1529	267	959	415	307	1762	1834	790	1990	958	**958**	**601**
HET	9.624	5.608	10.846	29.683	4.286	30.384	25.867	28.831	6.038	15.983	29.552	5.357	19.167	7.861	6.019	36.933	36.713	14.371	36.743	20.359	**19.01**	**11.87**
HUHP																					**Avg**	**Dev**
z	140.94	131.38	171.92	207.57	230.41	452.52	290.47	231.73	132	129.44	285.84	284.59	145.26	172.82	94.739	273.36	384.26	108.7	330.11	123		
ROG	2.82	15.82	12.82	49.26	8.24	1.29	12.60	35.52	11.26	29.31	21.08	13.61	7.52	3.02	13.17	9.68	30.58	40.55	36.40	34.68	**19.46**	**14.18**
NCE	490	328	561	410	963	1635	1284	1047	290	319	826	1046	707	325	301	1255	1578	440	1068	238	**756**	**451**
HET	9.774	6.559	11.266	8.292	19.338	32.747	25.697	20.97	5.789	6.52	15.983	20.83	14.201	6.289	5.969	26.718	31.626	7.931	19.618	4.977	**15.05**	**9.11**
HUSS																					**Avg**	**Dev**
z	121.76	128.73	155.39	196.89	203.98	197.41	152.89	135.25	132.55	117.26	278.55	199.7	101.84	166.48	95.452	159.8	227.98	84.259	236.17	122.3		
ROG	19.02	18.20	24.83	57.37	22.27	132.20	113.93	132.18	10.80	42.74	24.25	61.90	53.36	6.95	12.33	87.62	120.10	81.33	90.66	35.45	**57.37**	**42.79**
NCE	422	244	527	361	1124	220	199	393	244	314	759	650	230	369	311	294	701	276	639	229	**425**	**240**
HET	8.362	4.917	10.585	7.2	22.592	4.406	4.036	7.882	4.917	6.429	14.632	12.939	4.566	7.931	6.169	6.059	14.02	5.008	11.767	4.917	**8.47**	**4.71**
HULS																					**Avg**	**Dev**
z	118.25	130.12	175.31	260.83	153.15	353.99	283.3	268.4	133.56	149.64	295.05	162.89	134.42	164.77	97.435	271.78	434.63	116.07	402.88	142.09		
ROG	22.55	16.94	10.64	18.78	62.85	29.49	15.45	17.00	9.97	11.85	17.30	98.48	16.18	8.06	10.04	10.31	15.45	31.63	11.76	16.58	**22.57**	**21.58**
NCE	456	272	537	1512	213	1600	1322	1475	302	779	1420	267	1011	415	307	1766	1891	934	1761	972	**961**	**590**
HET	9.073	5.458	10.726	30.584	4.317	31.906	26.348	29.352	6.088	16.033	28	5.348	20.039	8.402	6.038	35.922	37.714	16.974	33.448	20.76	**19.13**	**11.71**
MSA																					**Avg**	**Dev**
z	139.19	137.64	173.17	277.04	223.35	423.91	266.04	276.68	133.06	143.17	307.97	245.13	135.31	171.77	98.238	268.34	382.41	120.74	383.35	153.93		
ROG	4.11	10.56	12.01	11.83	11.66	8.13	22.94	13.50	10.38	16.91	12.38	31.90	15.42	3.65	9.14	11.73	31.22	26.55	17.46	7.61	**14.45**	**8.02**
NCE	22265	28079	23305	16474	16364	16154	17158	19676	65459	20059	16325	17657	20110	28847	40091	15556	16391	15675	16541	19246	**22572**	**11763**
HET	448.4	564.61	470.26	332.84	332.43	327.08	345.99	396.58	1306.7	419.11	313.23	354.63	404.19	655.5	797.06	322.21	328.89	290.67	316.83	395.32	**456.12**	**237.71**
Objective value at upperbound																						
z	144.92	152.17	193.97	309.83	249.4	458.37	327.08	314.03	146.86	167.38	346.1	323.32	156.18	178.04	107.22	299.82	501.78	152.79	450.26	165.65		

Note: z: final objective value, ROG: relative optimality gap, NCE: Number of cycle time evaluation, HET: Heuristics evaluation time

Figure E.1 ROG Chart, 3-Product 4-Station 5-Step Case

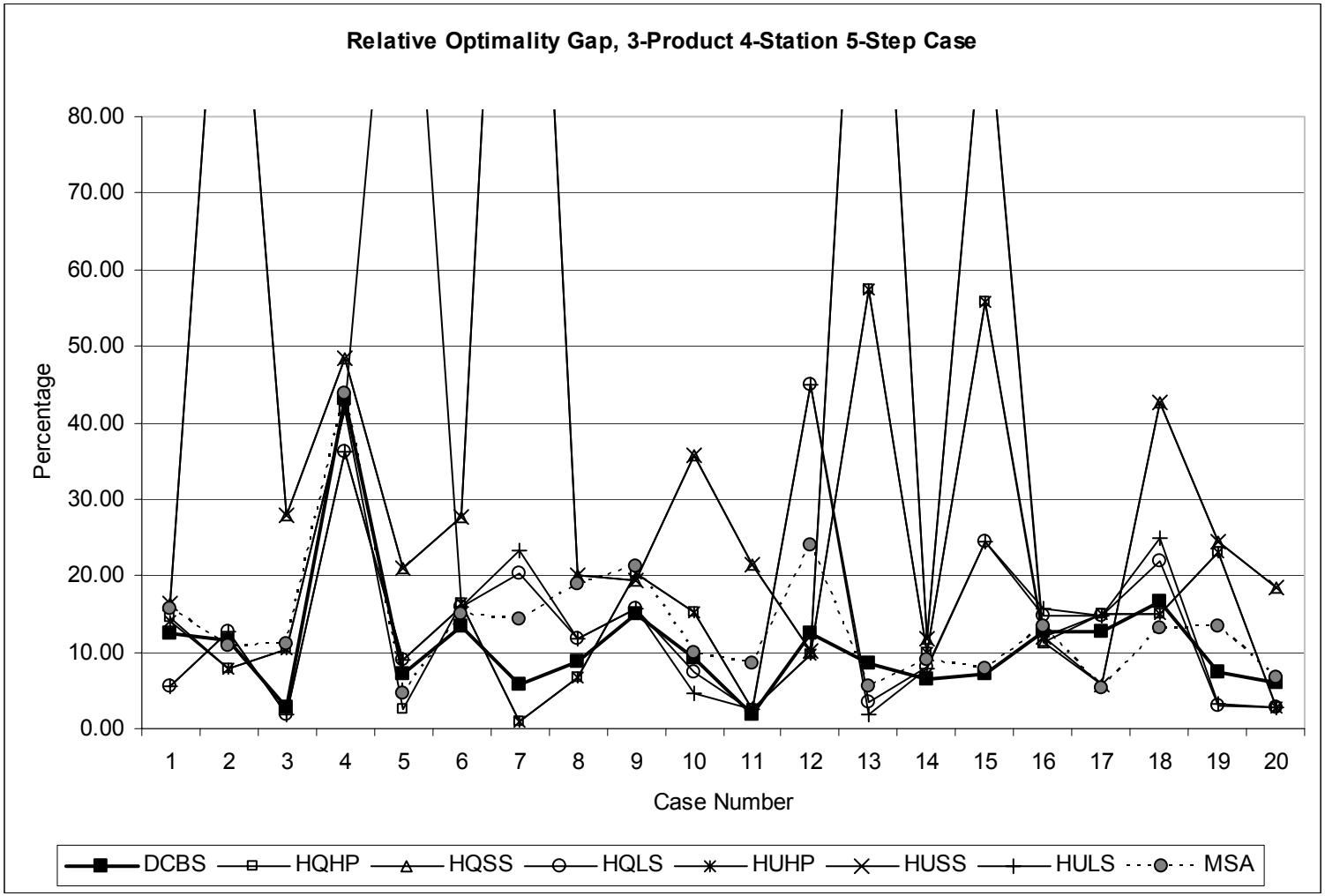


Figure E.2 ROG Chart, 4-Product 5-Station 6-Step Case

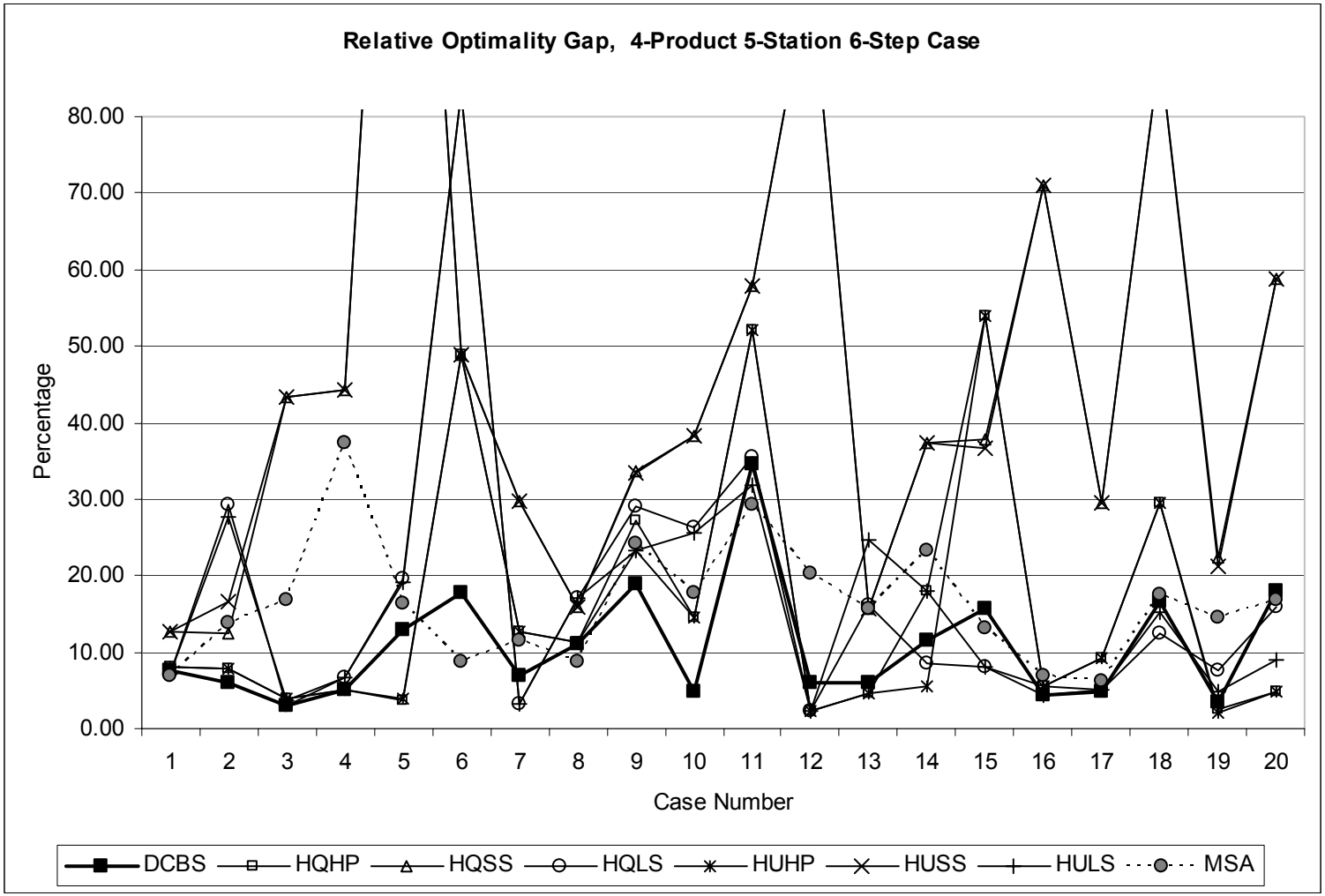


Figure E.3 ROG Chart, 5-Product 6-Station 7-Step Case

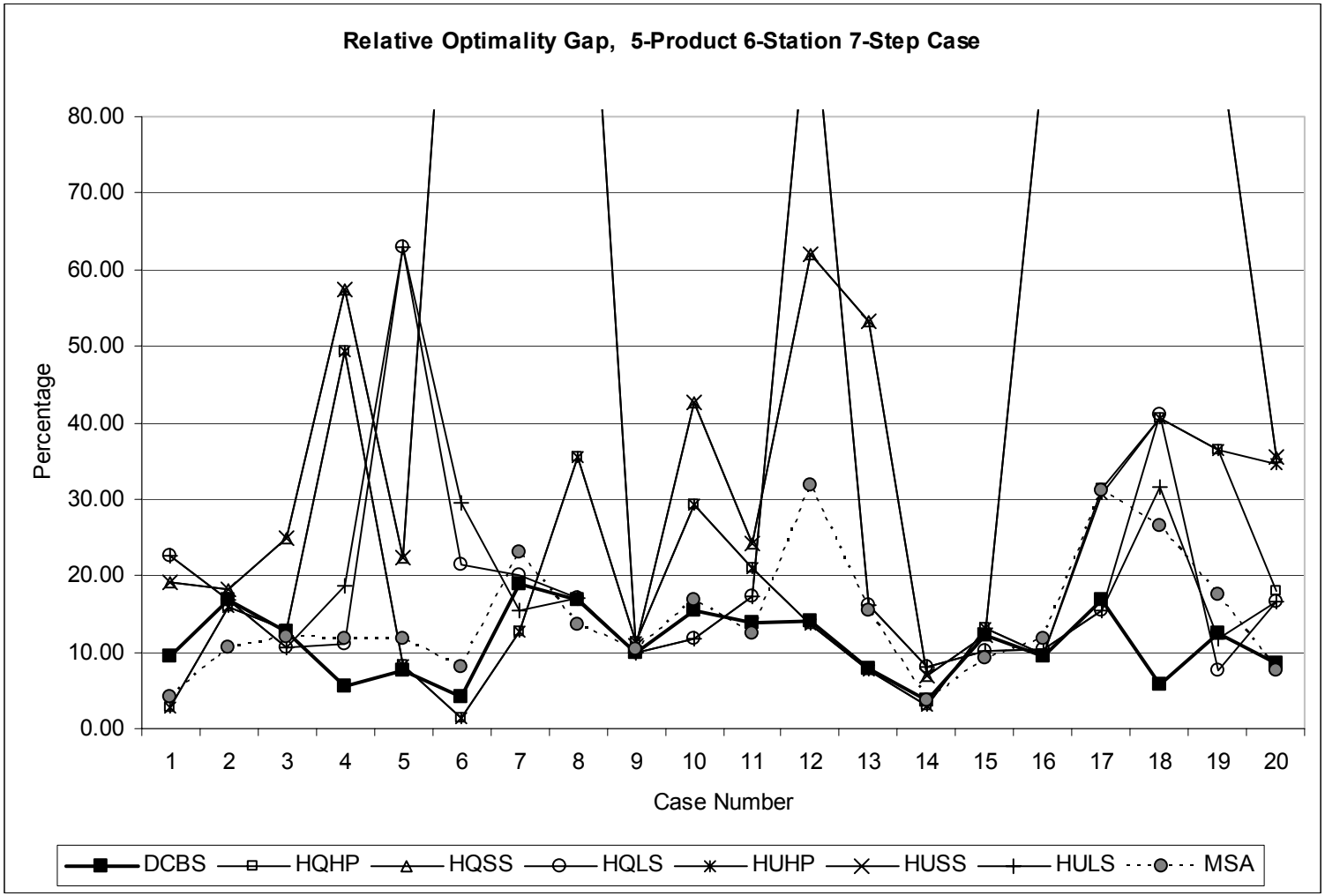


Figure E.4 ROG Histogram, 3-Product 4-Station 5-Step Case

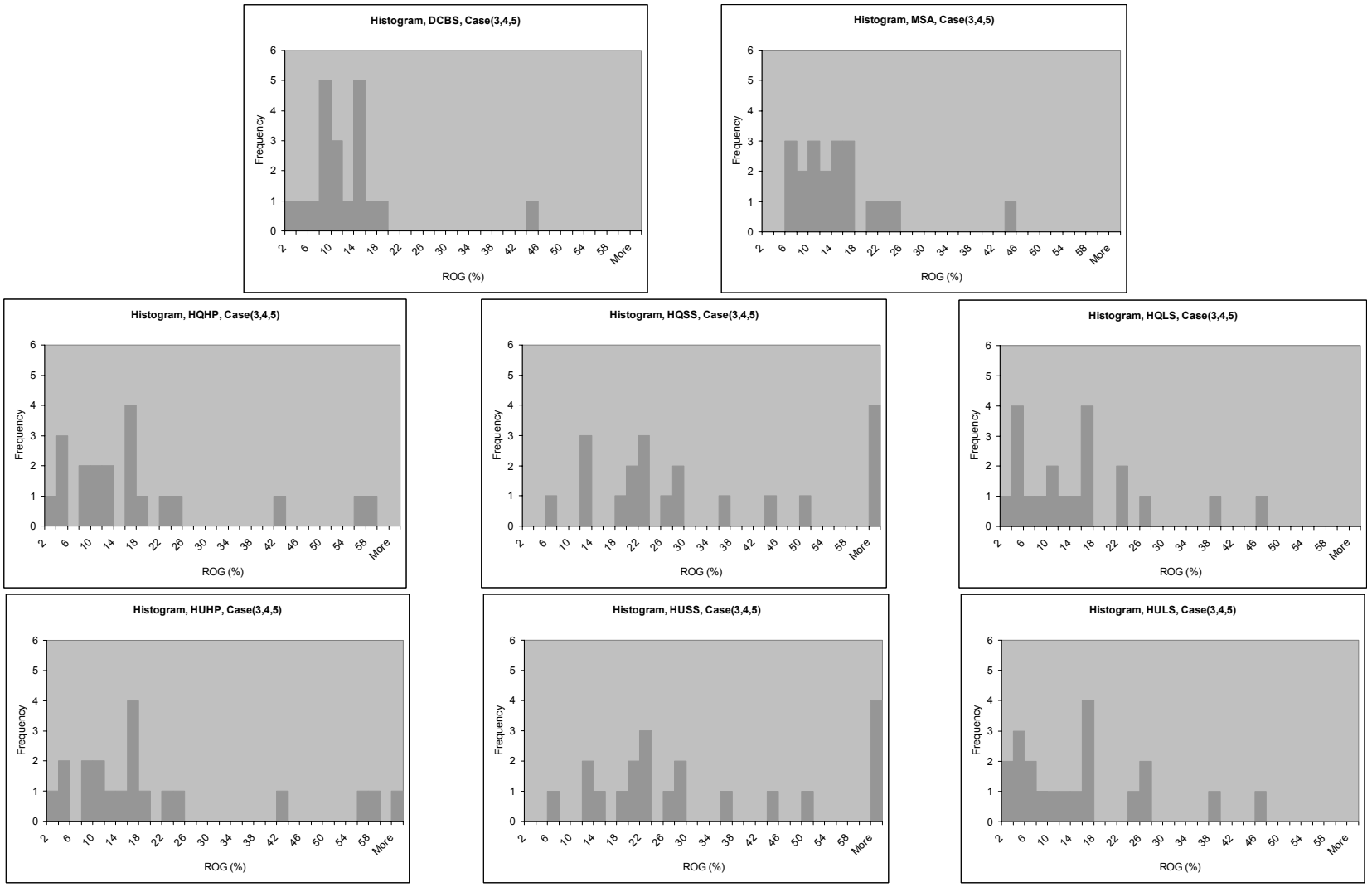


Figure E.5 4-Product 5-Station 6-Step Case

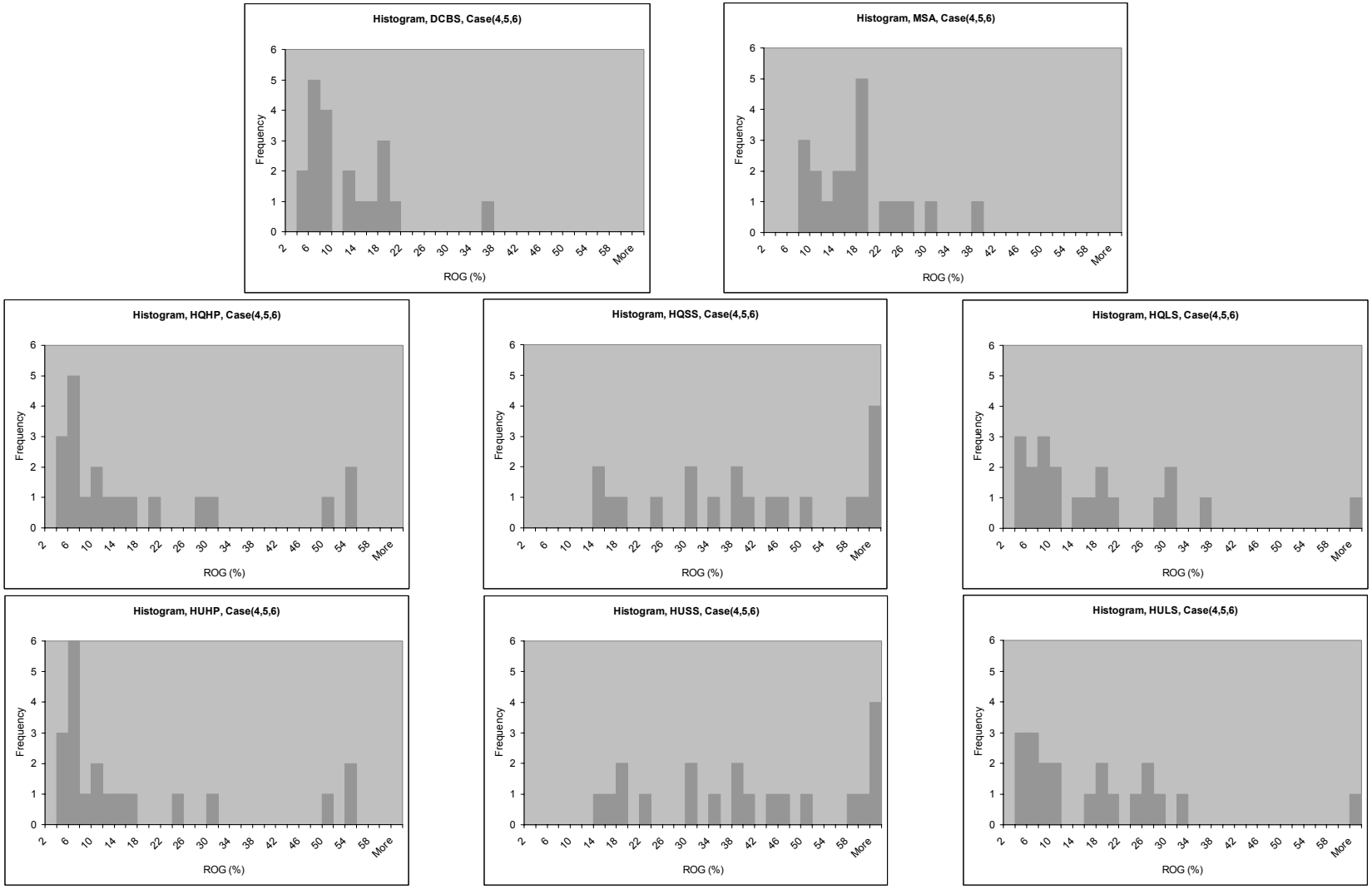


Figure E.6 5-Product 6-Station 7-Step Case

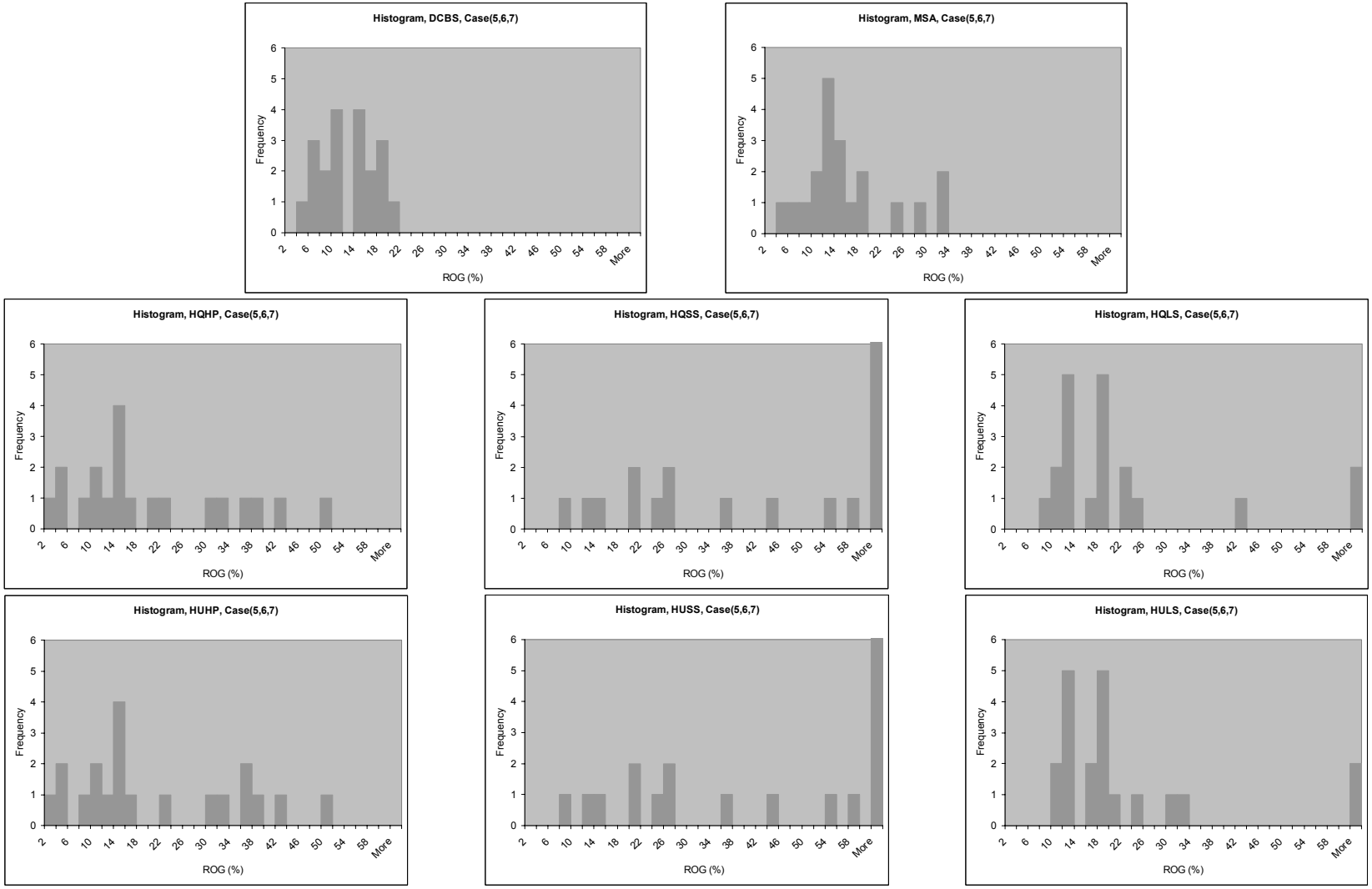


Figure E.7 Heuristics Evaluation Time, 3-Product 4-Station 5-Step Case

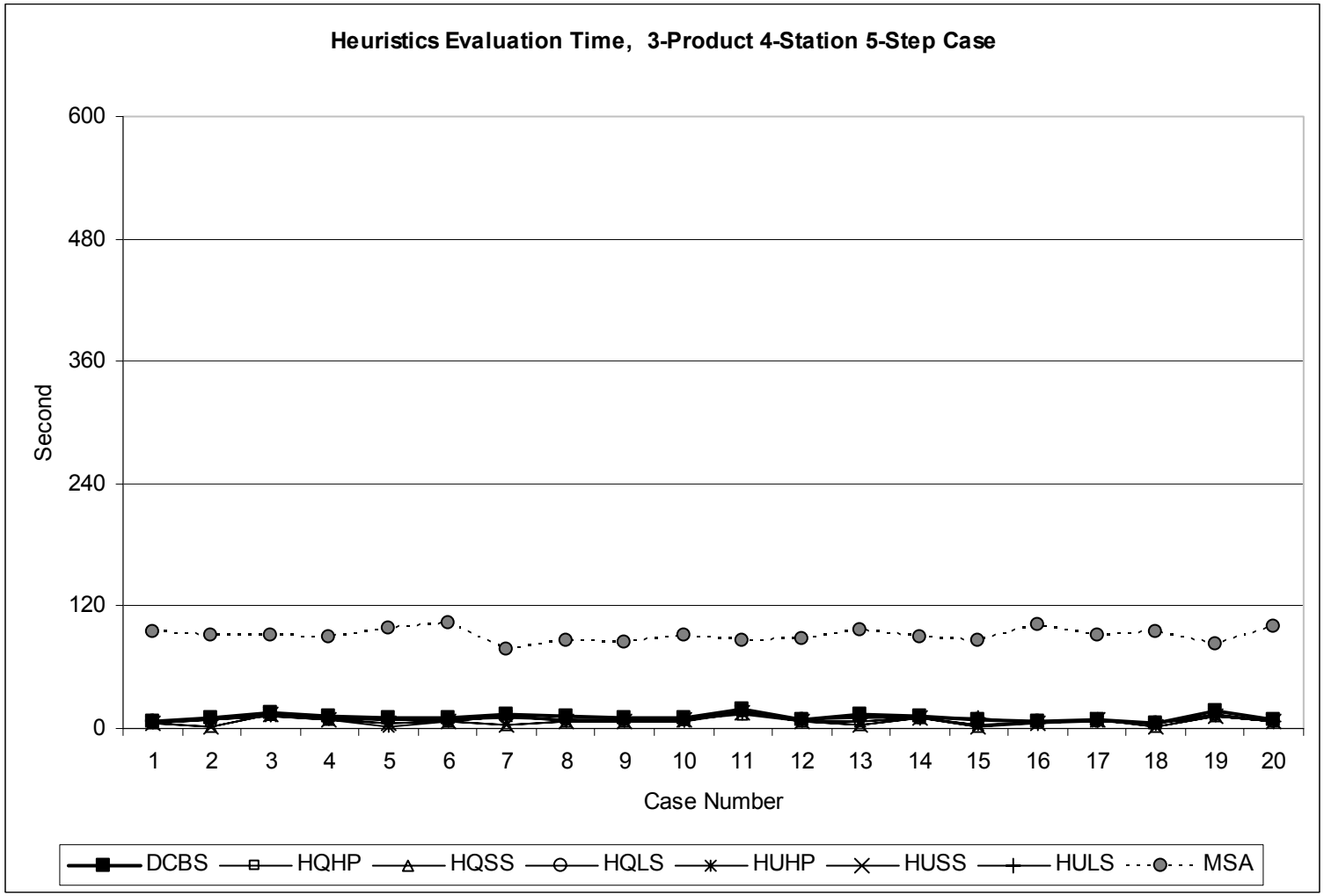


Figure E.8 Heuristics Evaluation Time, 4-Product 5-Station 6-Step Case

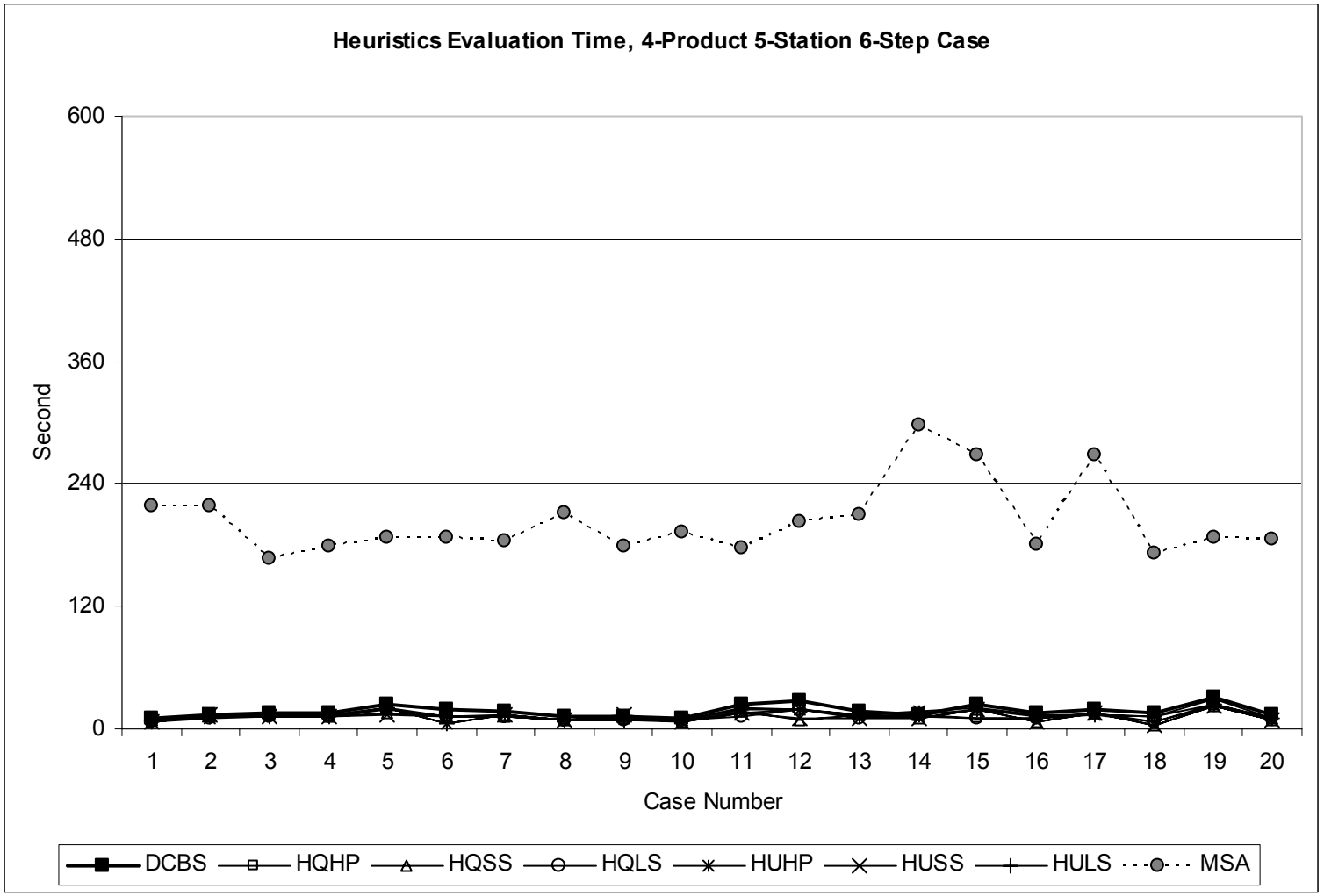
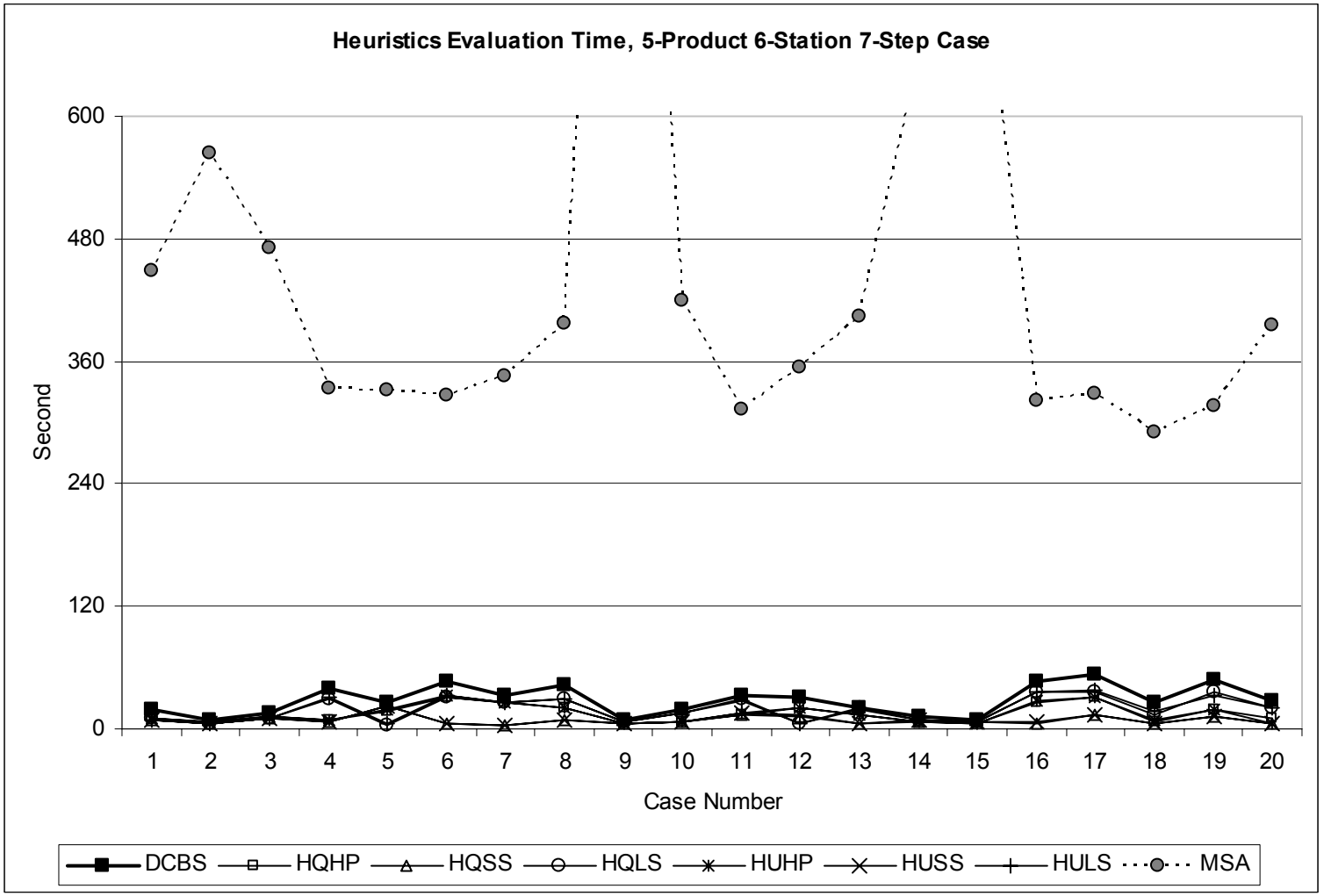


Figure E.9 Heuristics Evaluation Time, 5-Product 6-Station 7-Step Case



APPENDIX F

PAIRED T TEST TABLES

Table F.1 Result Table of Paired T Test in Group 1

	DCBS	*HQHP*
Mean	11.0618388	16.96530076
Variance	72.68409454	266.7770724
Observations	20	20
df	19	
t Stat	1.703412631	
p Value, one-tail	0.052396383	
t Critical one-tail	1.729131327	
p Value, two-tail	0.104792766	
t Critical two-tail	2.093024705	

	DCBS	*HQSS*
Mean	11.0618388	44.90240599
Variance	72.68409454	2343.415588
Observations	20	20
df	19	
t Stat	3.049769129	
p Value, one-tail	0.003296166	
t Critical one-tail	1.729131327	
p Value, two-tail	0.006592332	
t Critical two-tail	2.093024705	

	DCBS	*HQLS*
Mean	11.0618388	13.821912
Variance	72.68409454	131.2462652
Observations	20	20
df	19	
t Stat	1.333795554	
p Value, one-tail	0.099020116	
t Critical one-tail	1.729131327	
p Value, two-tail	0.198040232	
t Critical two-tail	2.093024705	

	DCBS	*HUHP*
Mean	11.0618388	23.16209919
Variance	72.68409454	845.5542024
Observations	20	20
df	19	
t Stat	1.833325787	
p Value, one-tail	0.041234997	
t Critical one-tail	1.729131327	
p Value, two-tail	0.082469994	
t Critical two-tail	2.093024705	

	DCBS	*HUSS*
Mean	11.0618388	44.93598228
Variance	72.68409454	2341.067856
Observations	20	20
df	19	
t Stat	3.054323147	
p Value, one-tail	0.003262979	
t Critical one-tail	1.729131327	
p Value, two-tail	0.006525959	
t Critical two-tail	2.093024705	

	DCBS	*HULS*
Mean	11.0618388	13.821912
Variance	72.68409454	131.2462652
Observations	20	20
df	19	
t Stat	1.333795554	
p Value, one-tail	0.099020116	
t Critical one-tail	1.729131327	
p Value, two-tail	0.198040232	
t Critical two-tail	2.093024705	

	DCBS	*MSA*
Mean	11.0618388	13.59957251
Variance	72.68409454	78.14667974
Observations	20	20
df	19	
t Stat	2.284009325	
p Value, one-tail	0.017024454	
t Critical one-tail	1.729131327	
p Value, two-tail	0.034048908	
t Critical two-tail	2.093024705	

Table F.2 Result Table of Paired T Test in Group 2

	DCBS	*HQHP*
Mean	10.74887856	16.2897609
Variance	60.55929571	289.5214795
Observations	20	20
df	19	
t Stat	2.024926782	
p Value, one-tail	0.028579614	
t Critical one-tail	1.729131327	
p Value, two-tail	0.057159228	
t Critical two-tail	2.093024705	

	DCBS	*HQSS*
Mean	10.74887856	48.25758115
Variance	60.55929571	1354.174255
Observations	20	20
df	19	
t Stat	4.664372539	
p Value, one-tail	8.44921E-05	
t Critical one-tail	1.729131327	
p Value, two-tail	0.000168984	
t Critical two-tail	2.093024705	

	DCBS	*HQLS*
Mean	10.74887856	17.07803286
Variance	60.55929571	340.4439598
Observations	20	20
df	19	
t Stat	1.761548	
p Value, one-tail	0.047114785	
t Critical one-tail	1.729131327	
p Value, two-tail	0.09422957	
t Critical two-tail	2.093024705	

	DCBS	*HUHP*
Mean	10.74887856	15.46773947
Variance	60.55929571	291.0345992
Observations	20	20
df	19	
t Stat	1.693450209	
p Value, one-tail	0.053350721	
t Critical one-tail	1.729131327	
p Value, two-tail	0.106701442	
t Critical two-tail	2.093024705	

	DCBS	*HUSS*
Mean	10.74887856	48.34610497
Variance	60.55929571	1344.147524
Observations	20	20
df	19	
t Stat	4.690239101	
p Value, one-tail	7.96978E-05	
t Critical one-tail	1.729131327	
p Value, two-tail	0.000159396	
t Critical two-tail	2.093024705	

	DCBS	*HULS*
Mean	10.74887856	17.02107824
Variance	60.55929571	330.8911704
Observations	20	20
df	19	
t Stat	1.718681677	
p Value, one-tail	0.050962128	
t Critical one-tail	1.729131327	
p Value, two-tail	0.101924256	
t Critical two-tail	2.093024705	

	DCBS	*MSA*
Mean	10.74887856	16.30581749
Variance	60.55929571	61.69740221
Observations	20	20
df	19	
t Stat	2.710095458	
p Value, one-tail	0.006941724	
t Critical one-tail	1.729131327	
p Value, two-tail	0.013883447	
t Critical two-tail	2.093024705	

Table F.3 Result Table of Paired T Test in Group 3

	DCBS	*HQHP*
Mean	11.13490979	18.65492071
Variance	21.14498036	189.029547
Observations	20	20
df	19	
t Stat	2.463124883	
p Value, one-tail	0.011746734	
t Critical one-tail	1.729131327	
p Value, two-tail	0.023493468	
t Critical two-tail	2.093024705	

	DCBS	*HQSS*
Mean	11.13490979	57.37301641
Variance	21.14498036	1831.057019
Observations	20	20
df	19	
t Stat	4.916943185	
p Value, one-tail	4.78641E-05	
t Critical one-tail	1.729131327	
p Value, two-tail	9.57283E-05	
t Critical two-tail	2.093024705	

	DCBS	*HQLS*
Mean	11.13490979	22.26686916
Variance	21.14498036	486.1550362
Observations	20	20
df	19	
t Stat	2.210135441	
p Value, one-tail	0.019781803	
t Critical one-tail	1.729131327	
p Value, two-tail	0.039563606	
t Critical two-tail	2.093024705	

	DCBS	*HUHP*
Mean	11.13490979	19.46222941
Variance	21.14498036	201.0206442
Observations	20	20
df	19	
t Stat	2.612889262	
p Value, one-tail	0.008553907	
t Critical one-tail	1.729131327	
p Value, two-tail	0.017107814	
t Critical two-tail	2.093024705	

	DCBS	*HUSS*
Mean	11.13490979	57.37301641
Variance	21.14498036	1831.057019
Observations	20	20
df	19	
t Stat	4.916943185	
p Value, one-tail	4.78641E-05	
t Critical one-tail	1.729131327	
p Value, two-tail	9.57283E-05	
t Critical two-tail	2.093024705	

	DCBS	*HULS*
Mean	11.13490979	22.56680096
Variance	21.14498036	465.7006182
Observations	20	20
df	19	
t Stat	-2.297153114	
p Value, one-tail	0.016572674	
t Critical one-tail	1.729131327	
p Value, two-tail	0.033145348	
t Critical two-tail	2.093024705	

	DCBS	*MSA*
Mean	11.13490979	14.45448936
Variance	21.14498036	64.37518929
Observations	20	20
df	19	
t Stat	2.045836304	
p Value, one-tail	0.02743425	
t Critical one-tail	1.729131327	
p Value, two-tail	0.054868499	
t Critical two-tail	2.093024705	

Table F.4 Result Table of Paired T Test in All Groups

	DCBS	*HQHP*
Mean	10.98187572	17.30332746
Variance	49.74679772	241.0271449
Observations	60	60
df	59	
t Stat	3.586783632	
P(T<=t) one-tail	0.000340337	
t Critical one-tail	1.671091923	
P(T<=t) two-tail	0.000680673	
t Critical two-tail	2.000997483	

	DCBS	*HQSS*
Mean	10.98187572	50.17766785
Variance	49.74679772	1808.644977
Observations	60	60
df	59	
t Stat	7.143146636	
P(T<=t) one-tail	7.74113E-10	
t Critical one-tail	1.671091923	
P(T<=t) two-tail	1.54823E-09	
t Critical two-tail	2.000997483	

	DCBS	*HQLS*
Mean	10.98187572	17.72227134
Variance	49.74679772	320.7573606
Observations	60	60
df	59	
t Stat	3.086838978	
P(T<=t) one-tail	0.00153998	
t Critical one-tail	1.671091923	
P(T<=t) two-tail	0.00307996	
t Critical two-tail	2.000997483	

	DCBS	*HUHP*
Mean	10.98187572	19.36402269
Variance	49.74679772	440.7949241
Observations	60	60
df	59	
t Stat	3.225352243	
P(T<=t) one-tail	0.001026616	
t Critical one-tail	1.671091923	
P(T<=t) two-tail	0.002053232	
t Critical two-tail	2.000997483	

	DCBS	*HUSS*
Mean	10.98187572	50.21836788
Variance	49.74679772	1804.426014
Observations	60	60
df	59	
t Stat	7.157797911	
P(T<=t) one-tail	7.31112E-10	
t Critical one-tail	1.671091923	
P(T<=t) two-tail	1.46222E-09	
t Critical two-tail	2.000997483	

	DCBS	*HULS*
Mean	10.98187572	17.84589165
Variance	49.74679772	314.6350982
Observations	60	60
df	59	
t Stat	3.132952767	
P(T<=t) one-tail	0.001347035	
t Critical one-tail	1.671091923	
P(T<=t) two-tail	0.002694069	
t Critical two-tail	2.000997483	

	DCBS	*MSA*
Mean	10.98187572	14.78662645
Variance	49.74679772	67.06293599
Observations	60	60
df	59	
t Stat	4.02353924	
P(T<=t) one-tail	8.27457E-05	
t Critical one-tail	1.671091923	
P(T<=t) two-tail	0.000165491	
t Critical two-tail	2.000997483	

APPENDIX G

ANALYTIC MODEL AND SIMULATION MODEL FOR G/G/M QUEUE

Table G.1 Test Models

Model	Arrival	Process	Number of servers
Analytic Model	General distribution with mean of 1 and squared coefficient of variation (SCV) of 0.25	General distribution with mean of 1.8 and squared coefficient of variation (SCV) of 0.25	2
Simulation Model 1	Normal distribution with mean of 1 and squared coefficient of variation (SCV) of 0.25, i.e., Normal (1, 0.5)	Normal distribution with mean of 1.8 and squared coefficient of variation (SCV) of 0.25, i.e., Normal (1.8, 0.45)	2
Simulation Model 2	Gamma Distribution with mean of 1 and squared coefficient of variation (SCV) of 0.25, i.e., Gamma (0.25, 4)	Gamma Distribution with mean of 1.8 and squared coefficient of variation (SCV) of 0.25, i.e., Gamma (0.45, 4)	2

Table G.2 Test Results

Simulation Tool	Arena™, version 5.0
Number of Replications	20
Length of simulation	1000 time units
Average time in system from the analytic model	3.73 time units

	Average	Error in %	0.950 C.I. Half Width	Inside C.I.?
Simulation model 1	2.84	31.34% smaller	0.233	No
Simulation model 2	3.32	12.35% smaller	0.436	Yes

REFERENCES

Atwater, B. and Gagne, M. L. (1997). The Theory of Constraints Versus Contribution Margin Analysis for Product Mix Decisions, Journal of Cost Accounting, Vol. 6, No. 15

Bard, J. F., Srinivasan, K., and Tirupati, D. (1999). An Optimization Approach To Capacity Expansion In Semiconductor Manufacturing Facilities, International Journal of Production Research, Vol. 37, No. 15, 3359-3382

Barton, P. and Lee, C. K. (2004). Design of process operations using hybrid dynamic optimization, Computers & Chemical Engineering, Vol. 28, No. 6, 955-968

Beale, E. M. L. (1997). Integer Programming, in The State of the Art in Numerical Analysis (D. Jacobs, ed.) Academic Press, London, 409-448

Chattopadhyay, D. and Chakrabarti, B. (2002). Reactive power planning incorporating voltage stability, International Journal of Electrical Power & Energy Systems, Vol. 24, No. 3, 185-200

Chung, S. and Huang, H. (2002). Cycle Time Estimation for Wafer Fab with Engineering Lots, IIE Transactions, Vol. 34, 105-118

Corana, A., Marchesi, M., Martini, C., and Ridella, S. (1987). Minimizing Multimodal Functions of Continuous Variables with the 'Simulated Annealing Algorithm, ACM Transactions on Mathematical Software, Vol. 13, No. 3, 262-280

Duran, M. A. and Grossmann, I. E. (1986). An Outer Approximation Algorithm for a Class of Mixed Integer Nonlinear Programs, Mathematical Programming, Vol. 36, 307-339

Floudas, C. A. (1995). Nonlinear and Mixed-Integer Optimization – Fundamentals and Applications, Oxford University Press, New York

Geoffrion, A. M. (1972). Generalized Benders Decomposition, Journal of Optimization Theory and Applications, Vol. 10, No. 4, 237-260

Grossman, I. E. (1990). Mixed-Integer Nonlinear Programming Techniques for the Synthesis of Engineering Systems, Research in Engineering Design, Vol. 1, 205-228

Gruhn, G. and Noronha, S. (1998). Process Synthesis by MINLP Optimization – Requirements, Potential, Limitation, 36th Tutzing Symposion, Computer Application in Process and Plant Engineering, Vol. 135, 141-155, WILEY-VCH

Grznar, J., Mehrez, A., and Offodile, O. F. (1994). Formulation of the Machine Cell Grouping Problem with Capacity and Material Movement Constraints, Journal of Manufacturing Systems, Vol. 13, No. 4, 241-250

Holthaus, O. and Rajendran, C. (1997). Efficient Dispatching Rules for Scheduling in a Job Shop, International Journal of Production Economics, Vol. 48, No. 10, 2857-2870

Hopp, W. J., Spearman, M. L., Chayet, S. C., Donohue, K. L., and Gel, E. S. (2002). Using an Optimized Queuing Network Model to Support Wafer Fab Design, IIE Transactions, Vol. 34, 119-130

Hopp, W. J. and Spearman, M. L. (1999). Factory Physics, 2nd Edition, Irwin McGrow-Hill, New York, NY

Horiguchi, K., Raghavan, N., Uzsoy, R., and Venkateswaran, S. (2001). Finite-Capacity Production Planning Algorithm for a Semiconductor Wafer Fabrication Facility, International Journal of Production Research, Vol. 39, No. 5, 825-842

Hwang, H., and Sun, J. U. (1997). Production Sequencing Problem With Reentrant Work Flows And Sequence Dependent Setup Times, Computers & Industrial Engineering, Vol. 33, No. 3-4, 773-776

Ioannou, G. and Sullivan, W. G. (1999). Use of Activity-Based Costing and Economic Value Analysis for the Justification of Capital Investments in Automated Material Handling Systems, International Journal of Production Research, Vol. 37, No. 9, 2109-2134

Iwata, Y. and Wood, S. C. (2002). Simple Cost Models of High-Product-Mix Wafer Fabs at Different Capacities, IEEE Transactions on Semiconductor Manufacturing, Vol. 15, No. 2, 267-273

Johnson, M. E. and Brandeau, M. L. (1999). Design of an Automated Shop Floor Material Handling System with Inventory Considerations, Operations Research, Vol. 47, No. 1, 65-80

Kim, Y., Lee, D. and Kim, J. (1998). A Simulation Study on Lot Release Control, Mask Scheduling, and Batch Scheduling in Semiconductor Wafer Fabrication Facilities, Journal of Manufacturing Systems, Vol. 17, No. 2, 107-117

Kuroda, M. and Kawada, A. (1995). Adaptive Input Control for Jop-shop Type Production Systems with Varying Demands Using Inverse Queueing Network Analysis, International Journal of Production Economics, Vol. 41, 217-225

Lasschuit, W. and Thijssen, N. (2004). Supporting supply chain planning and scheduling decisions in the oil and chemical industry, Computers & Chemical Engineering, Vol. 28, No. 6, 863-870

Lee, Y., Kim, S., Yea, S., and Kim, B. (1997). Production Planning In Semiconductor Wafer Fab Considering Variable Cycle Times, Computers & Industrial Engineering, Vol. 33, No. 3-4, 713-716

Lewis, R. J. (1995). Activity-Based Models for Cost Management Systems, Quorum Books, Westport, CT

Lin, Y. H. and Lee, C. E. (2001). A Total Standard WIP Estimation Method for Wafer Fabrication, European Journal of Operational Research, Vol. 131, No. 1, 78-94

Lu, S. H. and Kumar, P. R. (1999). Distributed Scheduling Based on Due Dates and Buffer Priorities, IEEE Transactions on Automatic Control, Vol. 36, No. 12, 1406-1416

Murty, K. G. and Kabadi, S. N. (1987). Some NP-complete Problems in Quadratic and Nonlinear Programming, Mathematical Programming, Vol. 39, No. 117

Navarro, A., Zapata, E., and Padua, D. (2003). Compiler Techniques for the Distribution of Data and Computation, IEEE Transactions on Parallel & Distributed Systems, Vol. 14, No. 6, 545-562

Nemhauser, G. L. and Wolsey, L. A. (1998). Integer and Combinatorial Optimization, J. Wiley, New York, NY

Papadimitriou, C. H. and Steiglitz, K. (1982). Combinatorial Optimization – Algorithms and Complexity, Prentice Hall, Inc., Englewood Cliff, NJ

Pardalos, P. M. and Schnitger, G. (1988). Checking Local Optimality in Constrained Quadratic Programming is NP-hard, Operations Research Letter, Vol. 7 (1), No. 33

Pardalos, P. M. and Vavasis, S. A. (1991). Quadratic Programming with One Negative Eigenvalue is NP-hard, Journal of Global Optimization, Vol. 1, No. 15

Rajagopalan, S. and Yu, H. L. (2001). Capacity Planning with Congestion Effects, European Journal of Operational Research, Vol. 134, No. 2, 365-377

Reveliotis, S. A., Lawley, M. A. and Ferreira, P. M. (1997). Polynomial Complexity Deadlock Avoidance Policies for Sequential Resource Allocation Systems, IEEE Trans. on Automatic Control, Vol. 42, 1344-1357

Sherali, H. D., B. Fraticelli, M. P., and Meller, R. D. (2003). Enhanced Model Formulations for Optimal Facility Layout, Operations Research, Vol. 51, No. 4

Silvestre, J. P. (1998). Approximation Heuristics and Benchmarkings for the MinLA Problem , Algorithms and Experiments (ALEX) - Building bridges between theory and applications, Trento, Italy, 112-128

Srinivasan, K., Sandell, R., and Brown, S. (1995). Correlation between yield and waiting time: a quantitative study, Proceedings of the International Electrics Manufacturing Technology Symposium, Austin, Texas

Swaminathan, J. M. (2000). Tool Capacity Planning for Semiconductor Fabrication Facilities Under Demand Uncertainty, European Journal of Operations Research, Vol. 120, 545-558

Ting, J. and Tanchoco, J. M. A. (2001). Optimal Bidirectional Spine Layout for Overhead Material Handling Systems, IEEE Transactions on Semiconductor Manufacturing, Vol. 14, No. 1, 57-64

Toktray, L. B. and Uzsoy, R. (1998). A Capacity Allocation Problem with Integer Side Constraints, European Journal of Production Research, Vol. 109, No. 1, 170-182

Whitt, W. (1983). The Queuing Network Analyzer, The Bell System Technical Journal, Vol. 62, No. 9, 2279-2815

Wei, S. Y., Lo, C. C. and Chang, C. A. (1997). Using Throughput Profit For Selecting Manufacturing Process Plan, Computers and Industrial Engineering, Vol. 32, No. 4, 939-948

Zhang, W. and Chen, M. (2001). A Mathematical Programming Model for Production Planning Using CONWIP, International Journal of Production Research, Vol. 39, No. 12, 2723-2734

VITA

SugJe Sohn was born in Seoul, South Korea. His early childhood interest in science and engineering has developed in a couple of decades to applied operations research and the management of large-scale manufacturing and information systems such as semiconductor production. His interdisciplinary background in Industrial Engineering, Mechanical Engineering, and Operations and Technology Management derives from his studies and degrees in Industrial and Systems Engineering at Georgia Institute of Technology (M.S.I.E., 2001), in Naval Architecture and Ocean Engineering at Seoul National University (M.S., 2000 and B.S., 1996) and in Technology Management at the Dupree Business School of Georgia Institute of Technology (Management of Technology (MOT) Certificate, 2004). He also has professional work experience in industry including independent technical consulting at the Semiconductor Division of Samsung Electronics (2002). He has published several research papers in international journals and conferences in the field of manufacturing systems and operations. He is currently appointed as Senior Supply Chain Management Engineer at Intel Corporation in Chandler, Arizona. Also, he has been a member of the worship team, Sound of Zion, at Saehan Presbyterian Church in Atlanta, Georgia.