THE COST MINIMIZATION OF MACHINING OPERATIONS USING GEOMETRIC PROGRAMMING

by

PAUL FREEDMAN
B.A.Sc. (Engineering Science), University of Toronto, 1978.

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES
(Department of Electrical Engineering)

We accept this thesis as conforming to the
required standard

THE UNIVERSITY OF BRITISH COLUMBIA
May, 1983

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of ELECTRICAL ENGINEERING

The University of British Columbia
1956 Main Mall
Vancouver, Canada
V6T 1Y3

Date JUNE 6 1983

ABSTRACT

The cost minimization of a machining operation can be described as a non-linear constrained optimization problem. Geometric Programming (GP) is a relatively new technique well-suited to this class of problems. In this thesis, three GP methods are described that are practical for real-time microprocessor controllers; numerical examples are presented based on three machining operations.

For high volume production, several machines can be serially linked with buffers in between to form an Automatic Transfer Line (ATL). The minimization of the combined costs is accomplished in a new way by modelling the cost of each machine as a polynomial function of cycle time, the machining time per workpiece. A new control strategy is described that dynamically re-assigns the cycle times of working machines when others in the ATL fail. To implement this strategy, a two level hierarchy is proposed with a local controller associated with each machine, and a single supervisory controller.

Simulation of some typical ATL configurations is used to compare the performance of this strategy with the strategy of no change in cycle time. For the cases studied, modest savings in cost and substantial savings in mean in-process buffer level are obtained when the machines are arranged in the appropriate order.

TABLE OF CONTENTS

page

LIST OF SYMBOLS

Chapter 2

x	primal vector
δ	dual vector
$g_0(x)$	objective function
$g_0(x,\hat{x})$	condensed objective function
u	a monomial
$g_1, g_2, \cdots$	constraints
λ_i	Lagrange multiplier for ith constraint
σ_i	weight of the ith monomial in the posynomial constraint
$\Psi(\delta)$	dual function
$V(\delta)$	natural logarithm of the dual function
s	optimal search direction
α	optimal step size
v	cutting speed vector
f	feedrate vector
T	Taylor Tool Life
X	Machine plus operator Overhead

Chapter 3

c	cost
N	Number of machines in the ATL
t	cycle time

Chapter 4

λ	Defining parameter for the Failure distribution
μ	Defining parameter for the Repair distribution
D	Drilling operation
M	Milling operation
T	Turning operation

Throughout this thesis, the superscript '^' denotes a trial value, and the superscript '*' denotes the optimal value.

LIST OF ABBREVIATIONS

Chapter 1

AC	Adaptive Control
ATL	Automatic Transfer Line
DCT	Dynamic Cycle Time (strategy)
FCT	Fixed Cycle Time (strategy)
GP	Geometric Programming
MEP	Machining Economics Problem
NC	Numerical Control (machine)

Chapter 2

Cm	Machining Cost
Cr	Tool Change Cost
Ct	Tool Cost
dod	degree of difficulty
Tm	Machining Time
Tr	Tool Change Time

Chapter 3

tmin	lower bound on cycle time for curve-fitting
tmax	upper bound on cycle time for curve-fitting

Chapter 4

Mi	ith machine in the ATL
MTTF	Mean Time to Fail
MTTR	Mean Time to Repair
TTF	Time to Fail
TTR	Time to Repair

LIST OF TABLES AND FIGURES

Tables:

Figures:

ACKNOWLEDGEMENTS

I was fortunate to receive advice and encouragement from many people as this work progressed:

Mr. Willem Vaessen of the Computing Centre, for many rambling discussions about SIMULA, non-linear programming codes, and simulation methodology.

Dr. F. Sassani of the Department of Mechanical Engineering, for technical help with NC machines and modern manufacturing systems.

Dr. E. Bohn, my thesis supervisor, for suggesting this topic and helping to direct my efforts.

Finally, my office pals Richard Jankowski and Brian Maranda, for their patience and humour when everything looked bleak.

I am also grateful for the financial support of this University in the form of a Research Assistantship and several Tutorial Assistantships.

"In this work, when it shall be found that much is omitted, let it not be forgotten that much likewise is performed."
-- S. Johnson, Preface to the Dictionary (1755)

"The first industrial revolution, the revolution of the 'dark satanic mills', was the devaluation of the human arm by the competition of machinery... The modern industrial revolution is similarly bound to devalue the human brain."
-- N. Wiener, Cybernetics (1947)

"It must be realized that ... the elimination of 'humanly unnecessary' work is essentially the pre-requisite for every kind of cultural development process."
-- G. Arndt, A Survey of FMS Layouts (1982)

CHAPTER 1: Introduction

From humble beginnings in the 1940's, programmable automation has come to dominate modern manufacturing industries [1]. The early Numerical Control (NC) machines used paper tape programs and simple digital computers for precise automatic positioning and cutting of metal workpieces. Modern NC systems have dedicated or shared controllers that transform simple operator statements in a command language into 'optimal' trajectories for the cutting tool and workpiece.

For high volume production of relatively simple items, several NC work stations (NC machine + controller) can be sequentially linked with a workpiece handling system to form an Automatic Transfer Line (ATL). Each work station typically performs a different machining operation. In most cases, 'in-process' buffers are added between stations for temporary storage of partly finished workpieces. A typical three stage ATL is sketched in Figure 1.1.

One important development in NC machine tool systems is Adaptive Control (AC) [2 to 11]. As shown in Figure 1.2, AC allows the control action to optimize some performance metric in accordance with the monitoring of secondary variables such as tool temperature. Note that there remains separate closed loop control action based on primary variables such as cutting speed.

There are three common forms of optimization: minimizing cost per workpiece, maximizing the rate of cutting (metal removal), or maximizing productivity, usually defined

Figure 1-1 A Two Station ATL

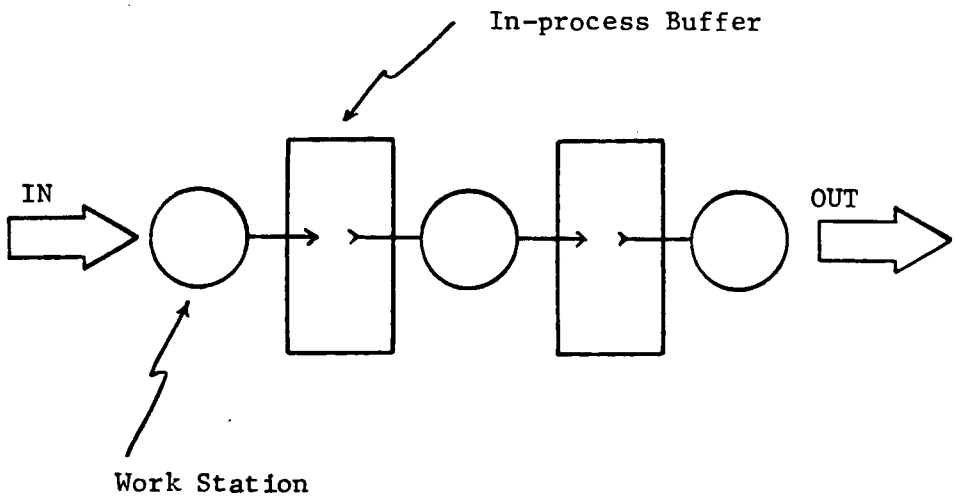
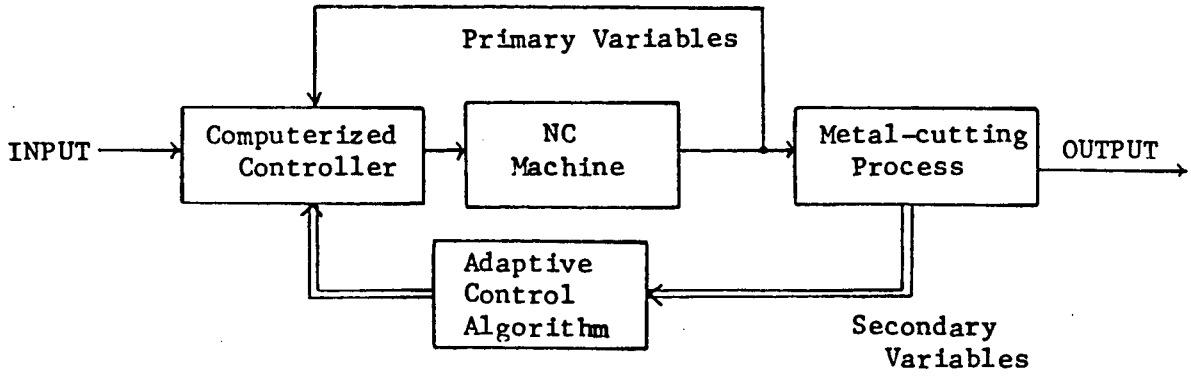


Figure 1-2 Elements of Adaptive Control



as the average number of workpieces produced per unit time. For example, according to a recent survey [12] of NC systems,

> "Productivity gains achieved by the use of adaptive control vary from 20% to as high as 100% for additional system costs of 5% to 20%. This has come about from the advances in sensor reliability, the developments in control strategies, improved reliability of hardware and software systems, and the dramatic improvements in microelectronics and computer technology." [p.194]

It is well known that the production cost per workpiece will change over time as the cutting tool becomes dull, and as the workpiece characteristics vary. This has encouraged the introduction of AC and the modelling of the machining process in a mathematical form. In this thesis, we will concentrate on cost minimization. For a single work station, this is called the Machining Economics Problem (MEP), and belongs to the class of constrained non-linear optimization problems. In Chapter 2, a relatively new optimization technique called Geometric Programming (GP) is described to solve the general MEP. Three GP methods are discussed, and a numerical example is presented for each one, based on MEP formulations for turning, drilling, and milling operations.

In Chapter 3, the MEP is examined in a new way by modelling cost as a function of cycle time, the machining time per workpiece. This is an indirect relationship, since both cost and cycle time are non-linear functions of the control variables. Pairs of (cycle time,cost) 'points' are generated from the MEP using GP and then a polynomial curve is fitted to

the data. Cost is then easily minimized by locating the minimum of the curve. It is shown that the constraints in the MEP set a lower limit on cycle time. As before, some numerical examples are presented.

In Chapter 4, we proceed from optimizing isolated work stations to optimizing a set of stations arranged as an ATL. Because the stations are serially linked with buffers of finite size, we must ensure that their cycle times are equal. If this is not the case, a faster machine would eventually empty its upstream buffer or overflow its downstream buffer and be forced to shut down and wait. To obtain a single cycle time that minimizes the total ATL cost, the cost versus cycle time curves of the component stations are summed, and the minimum of this new curve is then found.

At this point, we begin to discuss machine failure. A new optimization idea for an ATL is described based on the cost modelling of Chapter 3. In the Dynamic Cycle Time (DCT) strategy, the cycle times of working stations are dynamically changed when other stations fail, so that contiguous sub-sets of the ATL always operate to minimize cost. A two level control hierarchy is proposed to accomodate this sophisticated on-line optimization. In the Fixed Cycle Time (FCT) strategy, the cycle times are not changed when stations fail.

Computer simulations of some typical two station ATL's based on turning and drilling operations are used to compare these two strategies. Performance is measured in terms of mean cost/workpiece, mean in-process buffer level, and % machine

utilization.

In the first simulation, the stations are arranged Turning -> Drilling, and in the second simulation, the order is reversed. The results indicate that when the operations are properly ordered, the DCT strategy leads to modest cost savings and a substantial reduction in mean buffer level when compared to the FCT strategy, for a slightly lower machine utilization.

In Chapter 5, some conclusions are presented along with suggestions for future work.

There are four appendices. In Appendix A, control details are presented for turning, drilling, and milling operations. Appendix B is a brief description of the computer program used for the GP analysis and ATL simulation. Appendix C is a short note on using simulation to model stochastic processes. In Appendix D, an expression for the GP dual function is developed for primal constraints of the form g>=1.

CHAPTER 2

PART I: THE MACHINING ECONOMICS PROBLEM AND GEOMETRIC PROGRAMMING

The Machining Economics Problem [13] can be described as the minimization of a cost objective function subject to a set of metal-cutting constraints. The cost function has two parts: one is associated with the actual machining performed, and the other with the wear and subsequent replacement of the cutting tool. Typical constraints relate to machining power, final surface finish of the workpiece, and absolute limits on the control variables. The problem then is to find optimal values for the control variables (typically cutting speed, v, and feedrate,f) that minimize cost and satisfy the constraints. Mathematically, the MEP can be described as follows:

$$
\begin{aligned}
&\min_{v,f} \text{ (total) Cost} = \text{Machining cost}(v,f) + \text{Tool change cost}(v,f) \qquad (2.1)\\
&\text{subject to}\\
&\text{Cutting power}(v,f) \le Kp\\
&\text{Surface finish}(v,f) \le Kf\\
&v \le vmax\\
&f \le fmax
\end{aligned}
$$

In general, the Machining cost, Tool change cost, Cutting power, and Surface finish terms are all non-linear functions of v and f. Thus, the MEP represents a non-linear optimization problem with non-linear constraints.

Geometric Programming (GP) is a relatively new mathematical technique well suited to this kind of problem. The first comprehensive text [14] appeared in 1967 and since then, the original theory has been greatly extended for many applications [15,16,17].

Before reviewing the relevant theory, it is worthwhile placing GP in context. There are many different techniques available for non-linear optimization problems, as shown in Figure 2.1 [18].

With most 'indirect' methods, the Lagrangian $L(x,\lambda)$ is formed to create an unconstrained problem. Then optimality (Kuhn-Tucker) conditions are used to obtain a set of simultaneous equations in x and λ. At this point, 'primal' methods focus on x, while 'dual' methods key on λ.

'Direct' methods search for the solution in x space bounded by the constraints. Convergence behavior depends upon the starting position and the techniques used to determine step size and search direction at each iteration. 'First order' methods work with a Taylor expansion of the objective function to first order (gradient) while 'second order' methods use a quadratic expansion. In general, the second order methods will have faster convergence since they are better at selecting the best search direction, but they also require more computation.

As a rule, these direct and indirect methods are not suitable for real-time microprocessor applications. Except for very simple problems of low order, there is much computation required. In addition, these methods use complex algorithms and often need heuristic information to improve performance. Nonetheless, some researchers have tried standard non-linear programming techniques to solve the MEP. For example, a conjugate gradient method was described in [4] that required a logarithmic transformation to linearize the constraints.

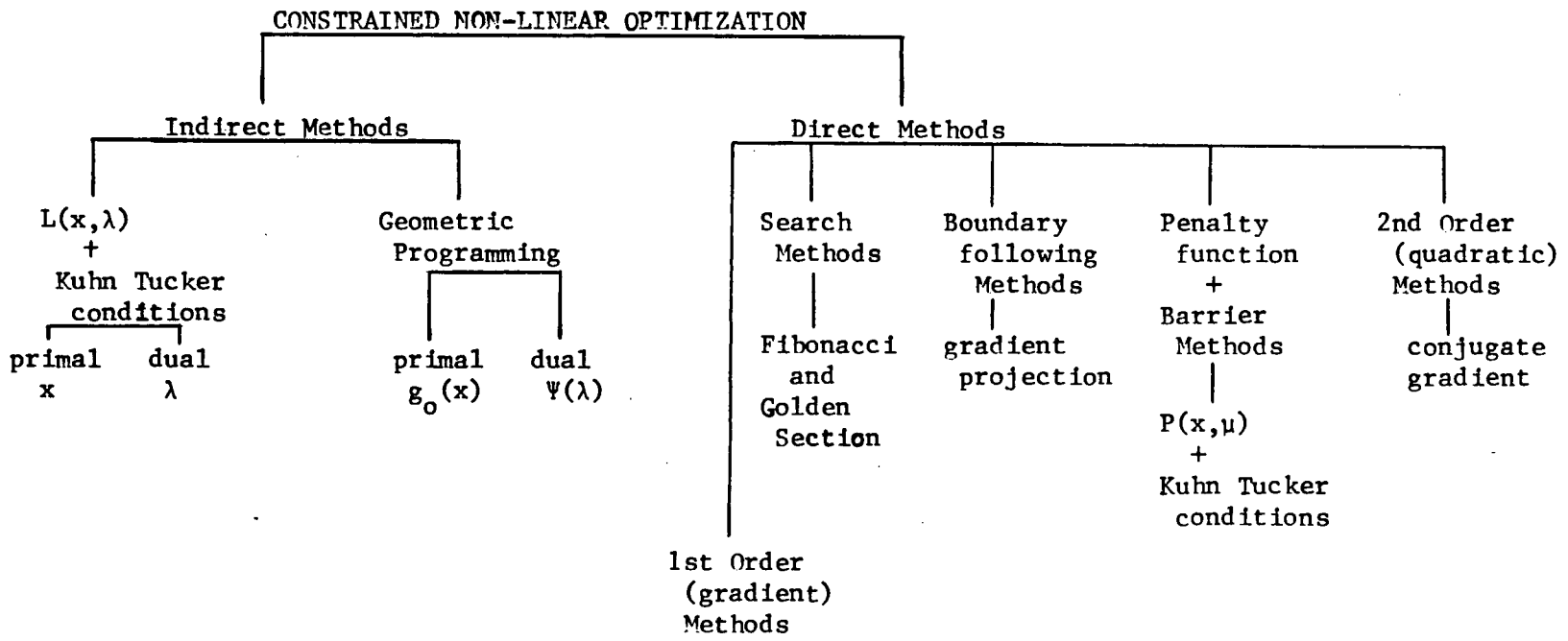


Figure 2-1. General Methods for Constrained Non-Linear Optimization

Other authors used the classical gradient approach [9], the Golden Section search [2], variable transformation plus direct search, and the penalty function method [5]. The intent was off-line optimization, using mini or mainframe computer systems. We will now proceed with a different approach that is much more promising.

Geometric Programming is a special kind of indirect optimization procedure with both primal and dual formulations. The following review is mainly drawn from [14] and [18]. To introduce some GP terminology, consider the following unconstrained optimization problem:

$$\min_{x} g_0(x) = u_1(x) + u_2(x) + \ldots + u_m(x) \qquad (2.2)$$

where x is a control vector of dimension n.

The single terms $\{u_i\}$ in the objective function are non-linear functions of x called 'monomials', with the general form

$$u_i = c_i x_1^{a_i^1} x_2^{a_i^2} \ldots x_n^{a_i^n} \qquad (2.3)$$

(Note that the 'k' in a_i^k represents a superscript.) A sum of monomials is called a 'posynomial'. When the monomials are not strictly additive, the name 'signomial' is used.

Primal methods use a procedure called 'condensation' to approximate the posynomial $g_0(x)$ by a single monomial $g_0(x,\hat{x})$. This allows us to linearize (2.2) by taking logarithms of all terms and making the variable substitution $z=\ln(x)$. At this point, the new problem can be tackled by various linear programming methods to solve for z. Finally, we use $x = \exp(z)$ to obtain the desired answer.

The condensed monomial is defined by the original

posynomial and a trial solution $\hat{x}$. It will be shown that condensation is in fact a linear approximation transformation in z space.

$$g_0(x,\hat{x}) = g_0(\hat{x}) \left[\frac{u_1(x)}{u_1(\hat{x})}\right]^{\delta_1} \cdots \left[\frac{u_m(x)}{u_m(\hat{x})}\right]^{\delta_m} \qquad (2.4)$$

$$= c_c x_1^{a_c^1} x_2^{a_c^2} \cdots x_n^{a_c^n} \qquad (2.5)$$

(where subscript c denotes condensed.) It can be shown that the condensed monomial serves as a lower bound on g_0. When $\hat{x} = x^*$, we have equality.

i.e. $g_0(x) >= g_0(x,\hat{x})$

Now let $g_0' = \ln g_0(x,\hat{x})$

$$= \ln c_c + a_c^1 \ln x_1 + \ldots + a_c^n \ln x_n \qquad (2.6)$$

so that

$$\min g_0 <=> \min g_0' = a_c^1 z_1 + \ldots + a_c^n z_n \quad (+ \ln c_c) \qquad (2.7)$$

Thus, (2.2) becomes the following linearized problem:

$$\min_z g_0' \qquad (2.8)$$

Note that the solution of (2.2) using condensation is iterative. A trial solution $\hat{x}$ defines $g_0(x,\hat{x})$ which allows (2.8) to be solved for z and hence x. This x then serves as the $\hat{x}$ for the next iteration. In many cases, x converges rapidly to x^*.

Dual methods use special optimality conditions that relate the exponents of the monomials in (2.2) through a dual vector

$$\delta = \left[\delta_1 \quad \delta_2 \quad \cdots \quad \delta_m\right]^T$$

(where T denotes transpose). Practically, δ_i represents the relative weight of u_i in the objective function g_0. We require that $\delta >= 0$. The new optimality conditions are as follows:

Normality condition: $\delta_1 + \delta_2 + \ldots + \delta_m = 1$ (2.9)

Orthogonality conditions:

$$a_1^1\delta_1 + \ldots + a_m^1\delta_m = 0 \qquad (2.10)$$

$$\vdots \qquad\qquad \vdots$$

$$a_1^n\delta_1 + \ldots + a_m^n\delta_m = 0$$

or more simply, $A\,\delta = 0$, where A is the nxm matrix of monomial exponents.

There are n orthogonality conditions plus 1 normality condition to yield (n+1) equations in m unknowns. We define the 'degree of difficulty'

$$dod = m - (n+1) \qquad (2.11)$$

It should be clear that when dod=0, δ^* can be directly obtained from (2.9) and (2.10). Clearly, we must still solve for x^* once δ^* is known. Dual methods also use the dual function

$$\Psi(\delta) = (c_1/\delta_1)^{\delta_1} \ldots (c_m/\delta_m)^{\delta_m} \qquad (2.12)$$

It can be shown that $\Psi(\delta)$ serves as a lower bound on g_0 and in most cases,

$$\min_x g_0(x) \iff \max_\delta \Psi(\delta)$$

However, we will see later in this chapter that this relationship does not always hold. At optimality, we must have

$$g_0(x^*) = \Psi(\delta^*)$$

and this important fact can be used to obtain x^* from δ^* using

$$u_i(x^*) = \delta_i^* \cdot \Psi(\delta^*) \qquad i=1,\ldots,m \qquad (2.13)$$

Condensation can also be used to solve the dual problem and establish the mapping between x and δ i.e. between the primal

and dual spaces. For problems with dod=0, (2.9) plus (2.10) immediately yield the solution δ^* and we have

$$g_0(x^*) = \psi(\delta^*)$$

For problems with dod≠0, we can use (2.4) to replace g_0 by a condensed objective function and obtain a new 'condensed' problem with dod=0. Then we have the important relation

$$g_0(x) >= g_0(x,\hat{x}) = \psi(\hat{\delta}) \qquad (2.14)$$

The idea is to use trial solutions $\{ \hat{x} \}$ to obtain dual vectors $\{ \hat{\delta} \}$ and iterate to find x that minimizes g_0. More will be said of this in Part II of this Chapter.

Now let us look at constraints in Geometric Programming. All the original GP developments dealt with inequality constraints of the form

$$g_1 = u_1 + \ldots + u_k <= 1 \qquad (2.15)$$

where each term u_i is a monomial in x. For the dual problem, we associate $\sigma_i >=0$ with u_i to obtain a new normality condition:

$$\sigma_1 + \ldots + \sigma_k = 1 \qquad (2.16)$$

Thus, σ_i plays the same role in g_1 as δ_i in g_0. We can also associate a Lagrange multiplier λ_1 with g_1 such that

$\lambda_1 = 0$ -> constraint is slack

$\lambda_1 > 0$ -> constraint is tight

Now consider the following constrained problem

$$\min_x g_0(x) = u_1 + \ldots + u_m \qquad (2.17)$$
$$\text{subject to}$$
$$g_1 = u_{m+1} + \ldots + u_p <= 1$$

(Note that the monomials are numbered sequentially for later notational convenience.) Define an extended dual vector

where $\delta_i = \sigma_i \lambda_1$ for $i=m+1,\ldots,p$ (2.18)

Note that $\sum_{i=m+1}^{p} \delta_i = \lambda_1 \cdot \sum_{i}^{p} \sigma_i = \lambda_1$ (2.19)

We also have that $\delta_i >= 0$ for $i=m+1,\ldots,p$ (since $\sigma_i, \lambda_1 >= 0$).

We can now re-write the dual function as follows:

$$\Psi(\delta) = \prod_{i=1}^{m} (c_i/\delta_i)^{\delta_i} \cdot \prod_{j=m+1}^{p} (c_j/\delta_j)^{\delta_j} \cdot \lambda_1^{\lambda_1}$$

$$= \prod_{i}^{m} (c_i/\delta_i)^{\delta_i} \cdot \prod_{j}^{p} (c_j \lambda_1 / \delta_j)^{\delta_j} \quad (2.20)$$

The presence of the constraint in (2.17) also means that there are additional elements in the exponent matrix:

$$A = \begin{bmatrix} A(g_0) \\ \hline A(g_1) \end{bmatrix} = \left[\begin{array}{ccc} a_1^1 & \cdots & a_m^1 \\ \vdots & & \vdots \\ a_1^n & \cdots & a_m^n \\ \hline a_{m+1}^1 & \cdots & a_p^1 \\ \vdots & & \vdots \\ a_{m+1}^n & \cdots & a_p^n \end{array}\right] \quad (2.21)$$

As before, we have $A\,\delta = 0$. (2.22)

The problem degree of difficulty is now

$$dod = p - (n+1)$$

If dod=0, we can solve (2.17) directly using the orthogonality conditions (2.22) and the normality condition (2.9).

Other problems occur with constraints of the form

$$g_1 = u_{m+1} + \ldots + u_p >= 1 \quad (2.23)$$

We now expect $\lambda_1 <= 0$ so that

$$\delta_i = \sigma_i \lambda_1 <= 0 \quad \text{for } i=m+1,\ldots,p \quad (2.24)$$

To ensure positivity of the dual vector, let

$$\nu_1 = -\lambda_1 \text{ and } \delta_i = \sigma_i \nu_1 \quad (2.25)$$

It is shown in Appendix D that the dual function becomes

$$\Psi(\delta) = \prod_{i=1}^{m} (c_i/\delta_i)^{\delta_i} \cdot \prod_{j}^{p} (c_j \nu_1/\delta_j)^{-\delta_j} \qquad (2.26)$$

The dual formulation for primal problems with several constraints can be easily obtained by generalizing this review.

Most research has focused on primal methods. Attempts to develop generalized procedures for the dual problem have not met with much success. This is unfortunate since the dual vector offers special insight into the underlying structure of the problem.

In the next part of this chapter, three dual methods will be presented using numerical examples of the Machining Economics Problem (MEP). The examples are adapted from Appendix A, with the following changes in notation:

$x_1 \leftarrow v$ $\qquad$ $x_2 \leftarrow f$

$g_0 \leftarrow$ Cost objective function

$g_1 \leftarrow$ Cutting power constraint

$g_2 \leftarrow$ Surface finish constraint

$g_3 \leftarrow$ vmax constraint

$g_4 \leftarrow$ fmax constraint

PART II: METHODS OF SOLUTION

Let us examine the objective function of (2.1) more closely, using notation from [13].

$C_{total} = Cm + Cr$

where $Cm = X\ Tm$

$Cr = (Tm / T)(X\ Tr + Ct)$

and Cm = Machining (cutting) cost ($)

Cr = Tool change cost ($)

X = Machine plus Operator overhead ($/min)

Tm = Machining time (min)

T = Taylor tool life (min)

Tr = Tool change time (min)

Ct = Tool cost ($)

The ratio Tm/T represents the proportion of Tool change cost assigned to each workpiece. The expressions for T and Tm are themselves functions of various cutting parameters. The expression for Ct might be the cost of a new tool if the tool is a throw-away type, or include the cost of re-sharpening for a re-usable tool.

In most constrained optimization problems, the solution lies on a boundary in x space. When there are many constraints, we expect one to be tight and the rest to slack at the optimum. The 'decomposition' method to be presented here is designed to exploit this. The original problem P^0 is split into a set of sub-problems each defined by the objective function plus one

constraint. If a solution x^i to sub-problem P^i exists and if it does not violate the other constraints, we add it to the set of feasible solutions $\{x^i\}$. Once all sub-problems have been analyzed, we obtain g_0^* and x^* using

$$g_0(x^*) = \min_i \{ g_0(x^i) \} \tag{2.27}$$

To illustrate this method, consider the following MEP example for a drilling operation:

$$P^0: \quad \min_{x_1, x_2} g_0 = 0.111\, x_1^{-1} x_2^{-1} + 2.53\text{E-}9\, x_1^{8.8}\, x_2^{3.9} \tag{2.28}$$

subject to

$$g_1 = 1.25\text{E-}5\, x_1 x_2^{0.8} <= 1 \qquad g_2 = 13.3\, x_2 <= 1$$
$$g_3 = x_1/88 <= 1 \qquad g_4 = x_2/0.30 <= 1$$

In Figure 2-2, the problem in x space is sketched.

Let us examine sub-problem P^2 more closely:

$$P^2: \quad \min g_0 = 0.111\, x_1^{-1} x_2^{-1} + 2.53\text{E-}9\, x_1^{8.8}\, x_2^{3.9}$$

subject to

$$g_2 = 13.3\, x_2 <= 1$$

The dual problem has degree of difficulty zero (three equations in three unknowns). Since g_2 has just 1 term, we have

$$\sigma = 1 \quad \text{and} \quad \delta_3 = \lambda_2$$

$$\begin{aligned} \delta_1 + \delta_2 &= 1 \\ -\delta_1 + 8.8\, \delta_2 &= 0 \\ -\delta_1 + 3.9\, \delta_2 + \delta_3 &= 0 \end{aligned} \tag{2.30}$$

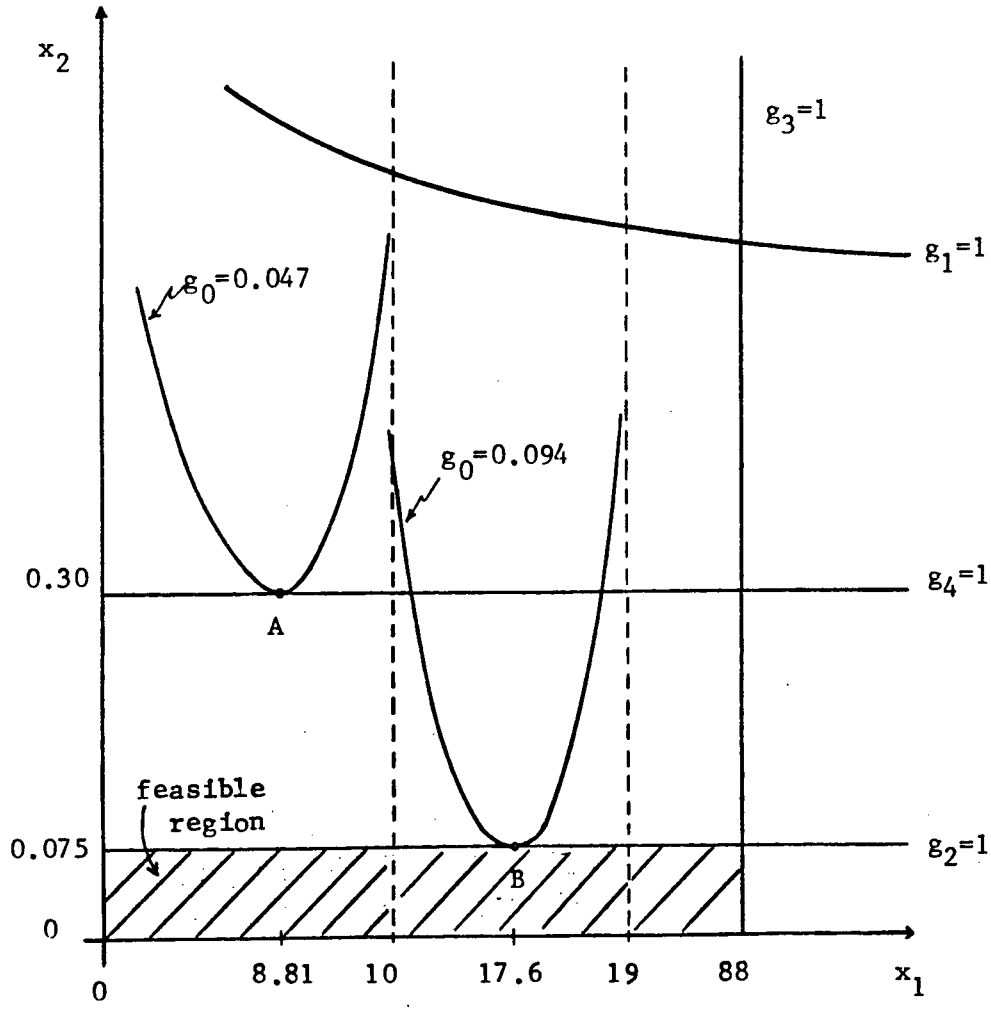
with the following solution, corresponding to point B:

$$\begin{aligned} \delta_1 &= 0.898 \quad \delta_2 = 0.102 \quad \delta_3 = 0.500 \\ g_0 &= 0.0935 \quad x_1 = 17.6 \quad x_2 = 0.075 \end{aligned} \tag{2.31}$$

In this example, sub-problems P^1 and P^3 have no solutions;

Figure 2-2

Drilling Example of a MEP



in each case, at least one dual variable is negative. The solution to P^4 corresponds to point A. Since there is only a feasible point, the solution to P^0 is just B.

The decomposition method can become very involved when there are many constraints, when x is of large dimension, and especially when the sub-problems $\{ P^i \}$ cannot be easily solved. However, it is simple and fast for problems of low order.

A second GP method uses condensation to approximate the posynomial objective function by a monomial to reduce the problem degree of difficulty to zero. The method of solution is iterative, and each solution is used to re-calculate the condensed monomial to start the next iteration. The following MEP example for a turning operation will be used to illustrate this dual method.

$$P^0: \quad \min_x \; g_0 = 2.76\, x_1^{-1} x_2^{-1} + 8.78\text{E-}5\, x_1^4\, x_2^{1.15} \qquad (2.32)$$

subject to

$$g_1 = 0.46\, x_1 x_2^{0.76} <= 1$$
$$g_2 = 71.4\, x_2 <= 1$$

As stated, P^0 has dod=1 and the dual problem cannot be solved directly. To obtain a trial solution, we can assume that the constraints are tight:

$$\hat{x}_1 = 55.7 \qquad \hat{x}_2 = 0.014 \qquad (2.33)$$

Now condense the objective function, using (2.4):

$$u_1(\hat{x}) = 3.54 \qquad u_2(\hat{x}) = 6.24 \qquad g_0(\hat{x}) = 9.78$$

$$\hat{\delta}_1 = u_1/g_0 = 0.36 \qquad \hat{\delta}_2 = u_2/g_0 = 0.64$$

$$g_0(x,x) = g_0(\hat{x}) \cdot \left[\frac{u_1(x)}{u_1(\hat{x})}\right]^{\hat{\delta}_1} \left[\frac{u_2(x)}{u_2(\hat{x})}\right]^{\hat{\delta}_2}$$

$$= 7.09E\text{-}3\ x_1^{2.19}\ x_2^{0.374} \qquad (2.34)$$

So the condensed primal problem (with dod=0) becomes:

$$P^1: \quad \min_x g_0 \quad \text{subject to} \quad g_1, g_2 <= 1$$

The dual problem follows directly:

$$\begin{aligned} \delta_1 &= 1 \\ 2.19\,\delta_1 + \delta_2 &= 0 \\ 0.374\,\delta_1 + 0.76\,\delta_2 + \delta_3 &= 0 \end{aligned} \qquad (2.35)$$

Since we require that $\delta >= 0$, (2.35) represents an inconsistent set of equations. One way out is to let $\delta_2 = 0$, implying that g_1 is slack. With this assumption, we can modify P^0 by deleting g_1 to form a 'reduced' problem P^2 with dod=0 that can be immediately solved:

$$P^2: \quad \min_x g_0 \quad \text{subject to} \quad g_2 <= 1 \qquad (2.36)$$

The dual formulation is

$$\begin{aligned} \delta_1 + \delta_2 &= 1 \\ -\delta_1 + 4\,\delta_2 &= 0 \\ -\delta_1 + 1.15\,\delta_2 + \delta_3 &= 0 \end{aligned} \qquad (2.37)$$

which yields

$$\begin{aligned} \delta_1 &= 0.80 \quad \delta_2 = 0.20 \quad \delta_3 = 0.57 \\ x_1 &= 37.7 \quad x_2 = 0.014 \quad g_0 = 6.53 \end{aligned} \qquad (2.38)$$

In fact, (2.38) is the solution to (2.32), proving that g_1 is slack and g_2 tight as assumed. However, we might have tried setting $\delta_3 = 0$ (g_2 slack) in P^1, but the reduced problem with g_2 deleted has no solution.

The third GP method [19] is a hybrid of primal and dual formulations. Iteration in x space is based on iteration in a 'reduced' dual space of dimension equal to the problem dod. Each iteration involves the determination of a trial dual

vector, a trial primal vector, and then a new dual vector. To illustrate this method, a MEP example for milling wil be used.

$$P^0: \quad \min_x g_0 = 96\, x_2^{-1} + 24\, x_1^{1.82}\, x_2^{0.212} \quad \text{subject to} \tag{2.39}$$

$$g_1 = 6.02E\text{-}3\, x_1^{-0.556}\, x_2^{0.751} \leq 1$$
$$g_2 = 1.21\, x_1^{-1} <= 1$$

The dual problem is then

$$\delta_1+\delta_2 = 1 \tag{2.40}$$
$$A\,\delta = 0$$

where $A = \begin{matrix} 0 \\ -1 \end{matrix} \begin{bmatrix} 1.82 & -0.556 & -1 \\ 0.212 & 0.751 & 0 \end{bmatrix}$ and

Note that dod=1 (3 equations in 4 unknowns).

In row echelon form, we have

$$A = \begin{bmatrix} 1 & 0 & -0.816 & -0.116 \\ 0 & 1 & -0.305 & -0.549 \end{bmatrix} \tag{2.41}$$

from which we can establish a basis for the kernel of A:

$$k_1 = \begin{bmatrix} 0.816 & 0.305 & 1 & 0 \end{bmatrix}^T \quad k_2 = \begin{bmatrix} 0.116 & 0.549 & 0 & 1 \end{bmatrix}^T \tag{2.42}$$

Let $\delta = c\, k_1 + r\, k_2$ represent a general vector in the ker(A):

$$\text{i.e. } \delta = \begin{bmatrix} 0.816\,c + 0.116\,r \\ 0.305\,c + 0.549\,r \\ c \\ r \end{bmatrix} \tag{2.43}$$

Using the normality condition,

$$\delta_1 + \delta_2 = 1 \;\rightarrow\; 1.21\,c + 0.665\,r = 1 \tag{2.44}$$
$$\rightarrow\; c = 0.892 - 0.593\,r$$

With this expression for c, (2.43) becomes

$$\delta = \begin{bmatrix} 0.728 - 0.368\,r \\ 0.272 + 0.368\,r \\ 0.892 - 0.593\,r \\ r \end{bmatrix} \tag{2.45}$$

Now we can use the positivity condition $\delta >= 0$ to establish bounds on r:

$$0.728 - 0.368r >= 0 \;\rightarrow\; r <= 1.978 \tag{2.46}$$

$0.272 + 0.368r >= 0 \quad -> \quad r >= -0.739$

Without additional information, we are free to choose the mid-point, r = 0.619. Using (2.45) and (2.12),

$$\delta = \left[0.500 \quad 0.500 \quad 0.525 \quad 0.619\right]^T \text{ and}$$

$$\Psi(\delta) = (96/0.5)^{0.5}(24/0.5)^{0.5}(6.02E\text{-}3)^{0.525}(1.21)^{0.619} \qquad (2.47)$$

$$= 7.376$$

Since $\delta > 0$, the constraints are tight so that

$$x_1 = 1.210 \quad \text{and} \quad x_2 = 1042.043 \qquad (2.48)$$

and from (2.39),

$$u_1 = 0.092 \quad u_2 = 148.140 \quad \text{and} \quad g_0 = 148.232 \qquad (2.49)$$

Using (2.13) and (2.45), we can solve for new δ_1, δ_2 and new r:

$$\delta_1 = 0.001 \quad \delta_2 = 0.999 \quad -> \quad r = 1.976 \qquad (2.50)$$

At this point, we have bounded g_0 by (2.47) and (2.49):

$$7.376 <= g_0 <= 148.232$$

With the new r from (2.50) in (2.45),

$$\delta = \left[0.001 \quad 0.999 \quad -0.280 \quad 1.976\right]^T \qquad (2.51)$$

Since we require that $\delta >= 0$, we must set $\delta_3 = 0$; this means that the constraint g_1 is slack at optimality. We can now return to the original dual problem P^0 minus g_1 (with dod=0) and directly solve for δ:

$$\delta^* = \left[0.175 \quad 0.825 \quad 1.502\right]^T \qquad (2.52)$$

and using (2.12),

$$\Psi(\delta^*) = 64.76 \qquad (2.53)$$

Since $\delta_3 > 0$, we must have

$$g_2(x) = 1 \quad -> \quad x_2^* = 8.47 \qquad (2.54)$$

and using (2.53) and (2.13),

$$u_2^* = \Psi(\delta_2^*) \quad -> \quad x_1^* = 1.21 \qquad (2.55)$$

With problems of higher dod, the scalar r becomes a vector and a step size must be determined along with a search direction at each iteration. In the next section, we will derive an expression for the optimum step size α. Consider the general problem of (2.17) once more:

$$\min_x \; g_0(x) = u_1 + \ldots + u_m \quad \text{subject to} \quad g_1(x) = u_{m+1} + \ldots + u_p <= 1$$

Using (2.20), we obtain the dual function

$$\psi(\delta) = (c_1/\delta_1)^{\delta_1} \ldots (c_m/\delta_m)^{\delta_m} (c_{m+1}\lambda_1/\delta_{m+1})^{\delta_{m+1}} \ldots (c_p\lambda_1/\delta_p)^{\delta_p} \quad (2.56)$$

$$\text{Let } V(\delta) = \ln \psi(\delta) \quad (2.57)$$

$$= [\delta_1 \ln(c_1/\delta_1) + \ldots + \delta_m \ln(c_m/\delta_m)] + [\delta_{m+1} \ln(c_{m+1}\lambda_1) + \ldots + \delta_p \ln(c_p\lambda_1)]$$

Now we approximate (2.57) using a Taylor expansion to second order:

$$V(\delta) = \overline{V} + \overline{V_\delta}^T \Delta\delta + 1/2\, \Delta\delta^T \overline{V_{\delta\delta}}\, \Delta\delta \quad (2.58)$$

where $V_\delta = \begin{bmatrix} \partial V/\partial\delta_1 \\ \vdots \\ \partial V/\partial\delta_p \end{bmatrix}$ and $V_{\delta\delta} = \begin{bmatrix} \partial^2 V/\partial\delta_1^2 & \cdots & \partial^2 V/\partial\delta_1\delta_p \\ \vdots & & \vdots \\ \partial^2 V/\partial\delta_p\delta_1 & \cdots & \partial^2 V/\partial\delta_p^2 \end{bmatrix}$

(The bar superscript denotes expansion at a trial solution.) Let s represent the new search direction in r space, and α the step size. After each iteration, we use the 'old' r to obtain a trial dual vector $\hat{\delta}$ and then a trial solution $\hat{g}_0$ from which a 'new' r is calculated.

$$\text{Let} \quad s = \text{new r} - \text{old r} \quad (2.59)$$

Now once we determine α, we can establish a 'newest' r value to begin the next iteration.

$$\text{i.e.} \quad \text{newest r} = \text{old r} + \Delta r \quad \text{where} \quad \Delta r = \alpha s \quad (2.60)$$

Now recall from (2.45) that δ can be separated into a constant part and variable part.

$$\text{i.e. } \delta = A + B\,r \quad \rightarrow \quad \Delta\delta = B\,\Delta r = B\,\alpha \cdot s \tag{2.61}$$

To simplify notation, let $h = Bs$.

Using (2.61) in (2.58), we obtain

$$V = \overline{V} + \overline{V}_{\delta}^{T}(h\alpha) + 1/2\,(h^{T}\alpha)\,\overline{V_{\delta\delta}}(h\alpha) \tag{2.62}$$
$$= \overline{V} + \alpha\overline{V}_{\delta}^{T}h + 1/2\,\alpha^{2}\,h^{T}\overline{V_{\delta\delta}}h$$

To minimize V with respect to α, we set the first partial derivative to zero and simplify, yielding:

$$\alpha = [\,V_{\delta}^{T}\,h\,] \,/\, [\,h^{T}(-V_{\delta\delta})\,h\,] \tag{2.63}$$

Earlier in this chapter, we remarked on the relationship between the primal and dual problems. In most posynomial cases, we have

$$\min_{x} g_0(x) \quad <=> \quad \max_{\delta} \psi(\delta) \tag{2.64}$$

However, this is not always true and has been the source of much confusion among GP researchers. In fact, all we can say in general is that

$$\min g_0(x) \quad <=> \quad d\,g_0 / dx = 0 \quad <=> \quad d\psi / d\delta = 0 \tag{2.65}$$

and $\psi(\delta)$ may be a minimum, maximum, or a saddle point. In this next section, we will present a counter-example for which

$$\min_{x} g_0(x) \quad <=> \quad \min_{\delta} \psi(\delta)$$

Consider the following primal problem (see Figure 2-3):

$$\min_{x} g_0 = x_1^{2} + x_2^{2} \text{ subject to } g_1 = x_1 + x_2 >= 1 \tag{2.66}$$

(By inspection, we can see that the solution is just $x_1 = x_2 = 0.5$ and $g_0 = 0.5$.)

Let $\nu_1 = -\lambda_1 >= 0$, and then define

$$\delta_3 = \nu_1 \sigma_3 \quad \text{and} \quad \delta_4 = \nu_1 \sigma_4 \tag{2.67}$$

Note that $\nu_1 = \delta_3 + \delta_4$. The dual problem is simply

$$\begin{aligned} \delta_1 + \delta_2 &= 1 \\ A\,\delta &= 0 \end{aligned} \quad \text{where} \quad A = \begin{bmatrix} 2 & 0 & -1 & 0 \\ 0 & 2 & 0 & -1 \end{bmatrix} \tag{2.68}$$

From row echelon form, we obtain the following basis for the ker(A):

$$k_1 = \begin{bmatrix} 1 & 0 & 2 & 0 \end{bmatrix}^T \qquad k_2 = \begin{bmatrix} 0 & 1 & 0 & 2 \end{bmatrix}^T \tag{2.69}$$

$$\text{Let} \quad \delta = ck_1 + rk_2 = \begin{bmatrix} c & r & 2c & 2r \end{bmatrix}^T \tag{2.70}$$

Now use the normality condition to obtain c in terms of r:

$$\delta_1 + \delta_2 = c + r = 1 \;\rightarrow\; c = 1 - r \tag{2.71}$$

and with this value in (2.70), we have

$$\delta = \begin{bmatrix} 1-r & r & 2(1-r) & 2r \end{bmatrix}^T \tag{2.72}$$

From the positivity condition $\delta >= 0$, we obtain bounds on r:

$$\begin{aligned} \delta_1 >= 0 &\;\rightarrow\; 1 - r >= 0 \;\rightarrow\; r <= 1 \\ \delta_2 >= 0 &\;\rightarrow\; r >= 0 \end{aligned} \tag{2.73}$$

Using (2.26), the dual function has the form

$$\Psi(\delta) = (1/\delta_1)^{\delta_1} (1/\delta_2)^{\delta_2} (\nu_1/\delta_3)^{-\delta_3} (\nu_1/\delta_4)^{-\delta_4} \tag{2.74}$$

$$\text{Let} \quad V = \ln \Psi = \delta_1 \ln(1/\delta_1) + \delta_2 \ln(1/\delta_2) - \delta_3 \ln(\nu_1/\delta_3) - \delta_4 \ln(\nu_1/\delta_4) \tag{2.75}$$

Using (2.72) and simplifying, we obtain

$$V = \ln(1-r) - r \ln(1-r) + r \ln(r) \tag{2.76}$$

$$\text{Finally,} \quad V = dV/dr = \ln r - \ln(1-r) \tag{2.77}$$

At the endpoints of r 'space', we have

$$\lim_{r \to 0} V = -\infty \quad \text{and} \quad \lim_{r \to 1} V = +\infty$$

This situation is sketched in Figure 2-4, and we can see that in this example,

$$\min_x g_o(x) \;<=>\; \min_\delta \Psi(\delta)$$

Of the three dual methods presented, decomposition

Figure 2-3 Primal Space for Counter-Example

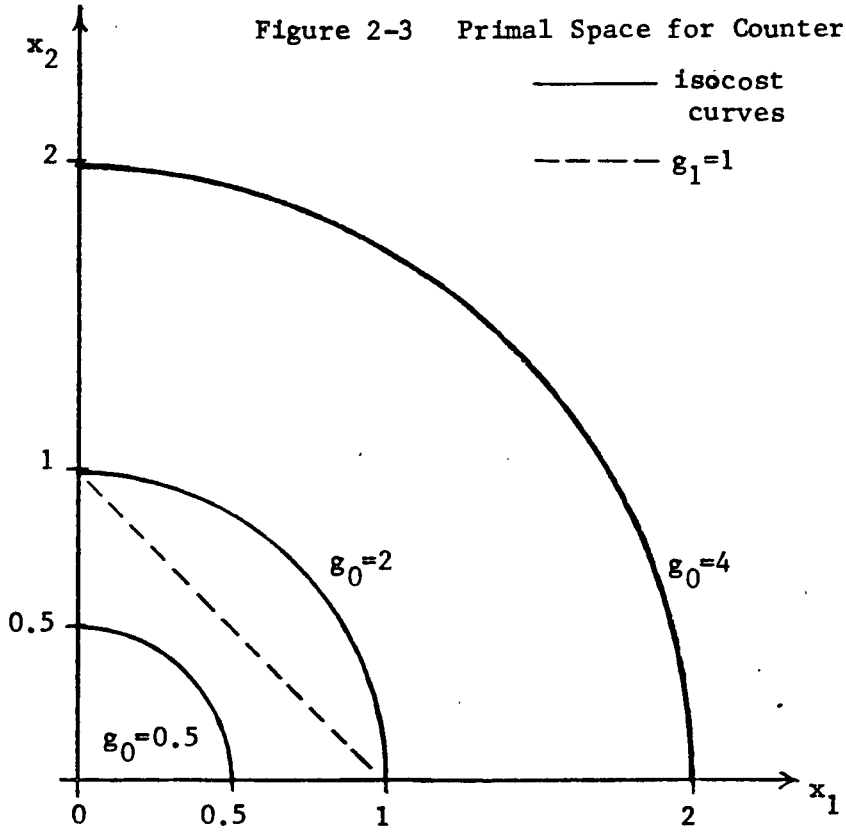
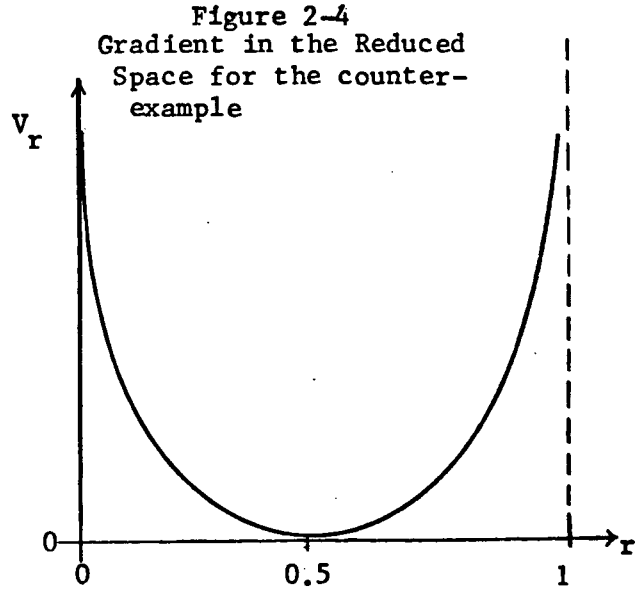


Figure 2-4
Gradient in the Reduced Space for the counter-example



is the most suitable for solving low order optimization problems using small computers. Because the MEP's of this thesis fall into that category, this was the method finally chosen for the computer programming analysis. Note that the decomposition method also guarantees that the true cost minimum will be found since it is algorithmic instead of iterative.

Of all the basic metal-working operations, turning is the only one to attract the attention of early GP researchers [20 to 25]. In these papers, the number of constraints in the MEP was limited to one or two and the degree of difficulty was zero, one, or two. All of the authors described the dual formulation; examples with dod≠0 were solved by maximizing the dual function over one or two selected dual variables. This required differential calculus or direct search. None discussed condensation.

CHAPTER 3: MODELLING COST AS A FUNCTION OF CYCLE TIME

A typical Automatic Transfer Line (ATL) is shown in Figure 3-1. There are three work stations, two in-process buffers, plus input and output buffers for unworked and finished items.

Define the 'cycle time' of a work station to be the machining time per workpiece. With in-process buffers of finite size, we require that the cycle times of all the work stations be nearly equal. To see why, consider Figure 3-1 once more. If M1 cycled more quickly than M2, buffer B2 would eventually overflow, forcing M1 to stop and wait for M2 to catch up; in this situation, M1 is said to be 'blocked'. If M2 cycled more quickly than M1, buffer B2 would eventually be emptied, forcing M2 to stop and wait; in this situation, M2 is said to be 'starved'. The 'starved' and 'blocked' machine states are sometimes together called 'forced down' states. When a machine fails (and is undergoing repair), it is said to be 'down'. An ATL is 'balanced' if all its work stations have equal cycle times; otherwise it is 'unbalanced'.

In Chapter 2, several GP methods were presented to solve the MEP associated with a single work station. Now when looking at a transfer line, we really want to minimize the combined cost of all the work stations. Consider the ATL optimization problem to be defined as follows:

$$\min_{\substack{v_1,\ldots,v_N \\ f_1,\ldots,f_N}} (\text{total})\ \text{Cost} = \text{Cost}_1 + \text{Cost}_2 + \ldots + \text{Cost}_N = (Cm_1 + Cr_1) + \ldots + (Cm_N + Cr_N) \qquad (3.1)$$

subject to Cutting power $(v_j, f_j) <= Kp_j$ $j=1,\ldots,N$
Surface finish $(v_j, f_j) <= Ks_j$
$v_j <= vmax_j$
$f_j <= fmax_j$
Cycle time$_1$ = Cycle time$_2$ = ... = Cycle time$_N$

where
N = Number of machines in the ATL
v_j, f_j = control variables for machine j
Cm_j = Cutting cost for machine j
Cr_j = Tool change cost for machine j

Intuitively, one might expect that the behavior of some machines will change when we operate to minimize total cost instead of individual costs. More will be said about this later in the chapter.

To date, the literature about ATL optimization has focused on production efficiency [eg. 26 to 33]. Given some probabilistic models for machine failure and repair, the transfer line efficiency can be maximized by judicious sizing of the in-process buffers. Because the analysis is so complex, most authors rely on simulation or approximation techniques.

Problems like (3.1) have not been discussed. Traditionally, transfer lines were designed to operate with one fixed cycle time based on expected product demand. Slower operations would require two identical stations in parallel to keep up. Once the line is operational, the cycle time would only be varied to respond to changes in market demands (and sometimes not even then. Some shops use a smaller batch production line to correct for market variations so that the high volume ATL can run unchanged). The idea that the cycle time of the transfer line itself can be optimized to minimize production cost seems to be new.

In the remainder of the chapter, we will show how (3.1) may

be solved using GP. First we need an alternative format for the cycle time equality. Let us introduce a constant T (as yet unknown) so that

$$\text{Cycle time}(v_j, f_j) = T \qquad j=1,\ldots,N \qquad (3.2)$$

and then 'hide' the constraint by expressing f_j in terms of v_j. Now we can re-phrase (3.1) as follows:

$$\min_{v_1,\ldots,v_N} \text{Cost} = (Cm_1 + Cr_1) + \ldots + (Cm_N + Cr_N) \qquad (3.3)$$

$$\text{subject to} \quad \begin{array}{l} \text{Cutting power } (v_j) <= Kp_j \qquad j=1,\ldots,N \\ \text{Surface finish } (v_j) <= Ks_j \\ v_j <= vmax_j \end{array}$$

Since we associate one dual variable with each monomial term, this represents a GP problem with 5N dual variables in (1+N) equations with dod=(1+4N). Note that the problem becomes impossibly difficult for N large; even for N=3, dod=13. More importantly, the solution of (3.3) does not indicate how the minimization 'trades-off' an increased cost at one station for a decreased cost at another. Indeed, just one control vector v is obtained for each cycle time T chosen. It appears that some iterative method would be required to solve (3.3) by selecting various values for T.

In this thesis, an entirely different approach is developed that does provide real insight into the cost minimization problem. The key to this approach is the modelling of cost versus cycle time at each station. The relationship between cost and cycle time is an indirect one, since both are non-linear functions of the control pair (v,f). To see this more clearly, consider Figure 3-2.

Figure 3-1
A Three Station ATL

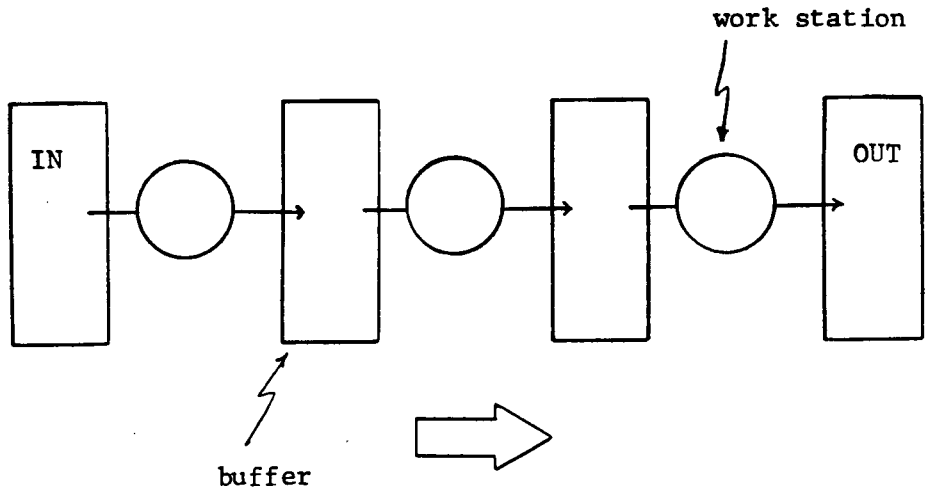
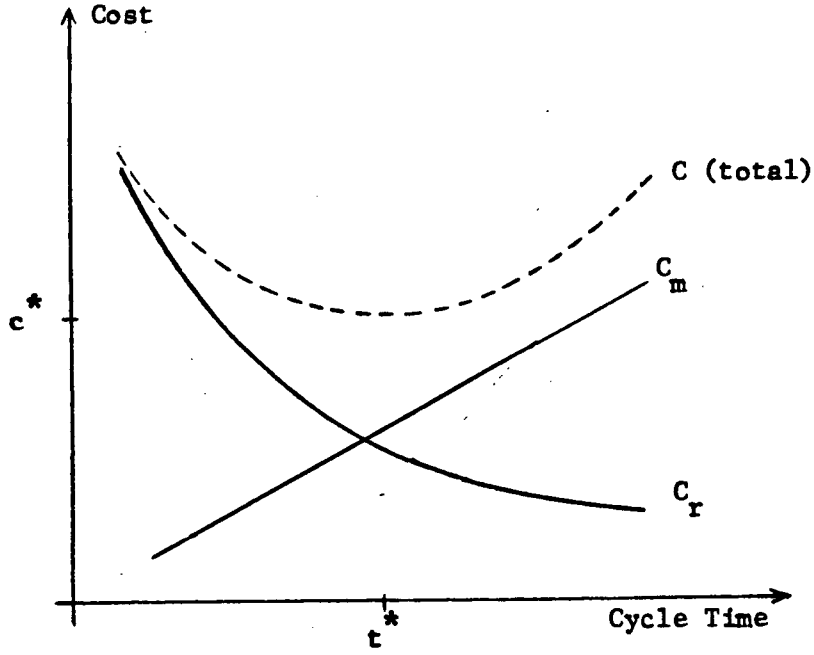


Figure 3-2
Cost vs. Cycle Time



A typical cost versus cycle time relationship is shown. The sub-cost Cm increases with time (longer cutting) while sub-cost Cr decreases with time (slower tool wear). For $t<t^*$, the increase in Cr is greater than the decrease in Cm so cost rises. Similarly, for $t>t^*$, the increase in Cm is greater than the decrease in Cr so cost rises. Moreover, Figure 3-2 also indicates how quickly cost changes as t moves away from t^*, so that the relative 'penalty' paid by $t \neq t^*$ can be easily seen. If the curve is fairly flat in the neighbourhood of t^*, most any cycle time could be selected without seriously changing cost. If the curve rises steeply from the minimum, then the selection of a $t \neq t^*$ could represent a substantial penalty.

Another important aspect of modelling cost as a function of cycle time t is the emergence of a lower bound on t imposed by the constraints associated with the MEP. This is not apparent from the sketch in Figure 3-2 which suggests that cost might continue to increase as t is decreased. Recall that the MEP associated with each work station has four constraints, each of which is a non-linear function of the two control variables. Thus we can obtain values for v ,f , and cycle time t by assuming that any two constraints are tight. Using this procedure, we can determine a set of values for t ; the lower bound would then be the smallest one. For a MEP with n constraints in the variables v and f, the number of possible distinct pairings is $n!/[2(n-2)!]$. To illustrate these ideas, we will look again at the MEP examples from Chapter 2.

The turning operation example is sketched in Figure 3-3

(not to scale). The intersections of pairs of constraints are labelled p1 to p5. Isotime curves (curves of equal cycle time) are only shown for the two feasible points p2 and p5. The true minimum cycle time corresponds to p2, the intersection of Cutting power and Surface finish constraints.

$$\text{cycle time} = 7.86\ v^{-1}\ f^{-1}$$
@ P2 = 10.1 minutes
@ p5 = 21.3 minutes

The drilling operation example is sketched in Figure 3-4 (not to scale). Here there is only one feasible point p1 from the intersection of Surface finish and vmax constraints.

$$\text{cycle time} = 1.96\ v^{-1}\ f^{-1}$$
@p1 = 0.30 minutes

The milling operation example is sketched in Figure 3-5 (not to scale). Note that cycle time is a function of f only so that just one isotime curve passes through both feasible points p1 and p2.

$$\text{cycle time} = 200\ f^{-1}$$
@ p1,p2 = 1.16 minutes

Thus far, no mention has been made of an upper bound on cycle time. In fact, there is none arising from the MEP formulation of Chapter 2. This is because the constraints impose no lower limits on v and f. (Strictly speaking, there are always lower bounds for any NC machine, but most authors ignore them to simplify the MEP formulation.) The need for a useful upper bound to define the cycle time interval for our cost modelling will be discussed later in this chapter.

Figure 3-3
Turning MEP

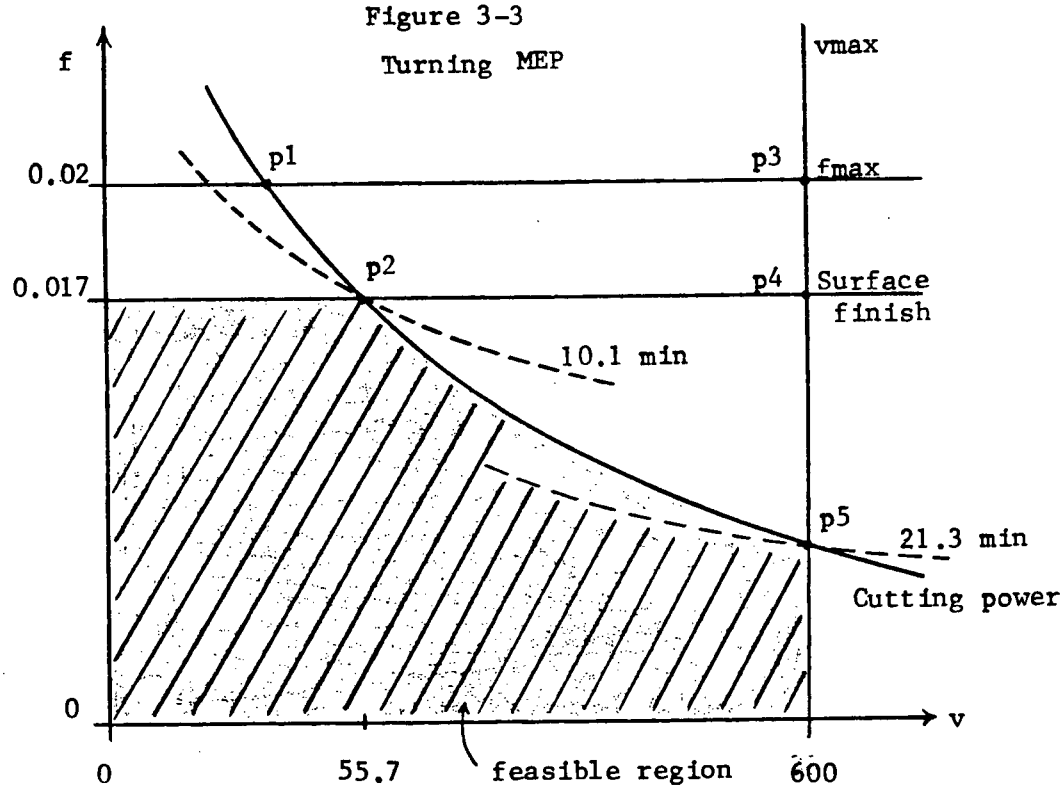
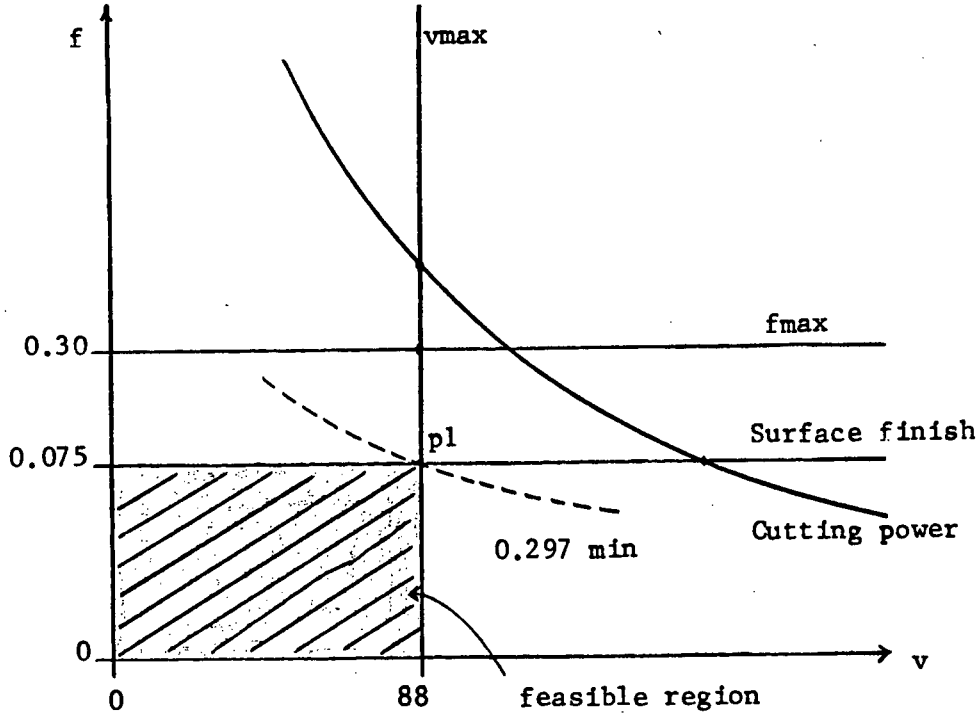


Figure 3-4
Drilling MEP



We can now describe a new approach to solving (3.3). If the cost of each work station can be modelled as a function of cycle time, then the total ATL cost can be minimized by summing the individual cost functions and locating the minimum using simple differential calculus.

Recall that cost c and cycle time t vary indirectly; we need to start with a control pair (v,f) to obtain a 'data' pair (t,c). To obtain c as a continuous function of t, we can fit a curve to a set of (t,c) points.

There are two fundamental aspects of any curve fitting: the kind of curve used, and the number of data points needed to obtain a reasonable fit. After some consideration, polynomial curve fitting was selected. The chief attraction was simplicity. The polynomial degree can be easily changed to obtain the desired goodness of fit. In addition, the derivatives of a polynomial are easy to compute.

$$\text{cost } c = c_0 + c_1 t^1 + c_2 t^2 + \ldots + c_p t^p \qquad (3.4)$$

where p is the degree of the polynomial. Of course, the sum of functions of the form (3.4) is itself a polynomial of degree p, so that the ATL cost is easily minimized as follows:

$$\text{ATL Cost} = \sum_{i=1}^{N} \sum_{j=0}^{p} c_{ij} t^j \qquad (3.5)$$

$$= \sum_{j} \left(\sum_{i} c_{ij} \right) t^j$$

$$\text{Now } dCost/dt = \sum_{j=1}^{p} j \left(\sum_{i=1}^{N} c_{ij} \right) t^j = 0 \qquad (3.6)$$

Thus, the cost minimum of (3.5) corresponds to one of the zeros of (3.6), which is a polynomial of degree (p-1). More details

about the curve fitting procedure can be found in Appendix B.

Before data points can be generated for each work station, an appropriate cycle time interval must be defined. To keep things simple, it was decided that just one time interval would be used for the curve-fitting at all work stations. In general, for a fixed number of data points, the curve fitting will be 'best' if the time interval is as small as possible. Recall that the MEP solution yields two pieces of information: the cost minimum c^* (and the corresponding t^*), plus the minimum cycle time t^{min} (imposed by the problem constraints). Clearly, the lower bound for all curve fitting must then be

$$tmin = \max_i \{ t^{min} \} \qquad i=1,\ldots,N \qquad (3.7)$$

If tmin were smaller, then at least one machine in the ATL could not operate.

An upper 'bound' on cycle time can be obtained as follows:

$$tmax = \max_i \{ t_i^* \} \qquad i=1,\ldots,N \qquad (3.8)$$

In fact, tmax is not a true limit in the way that tmin is, but it is a useful upper bound. To see this, consider the cost curves of two stations sketched in Figure 3-6. The minimum of the combined cost must occur at some $t_1^* < t^* < t_2^*$. In this way, the tmax value of (3.8) will always bound t^*.

Once tmin and tmax are defined, the data points for each work station can be generated by solving a modified MEP:

$$\min_v \text{Cost} = Cr(v) \quad \text{subject to} \qquad (3.9)$$
Cutting power(v), Surface finish(v)
and vmax constraints.

Figure 3-5

Milling MEP

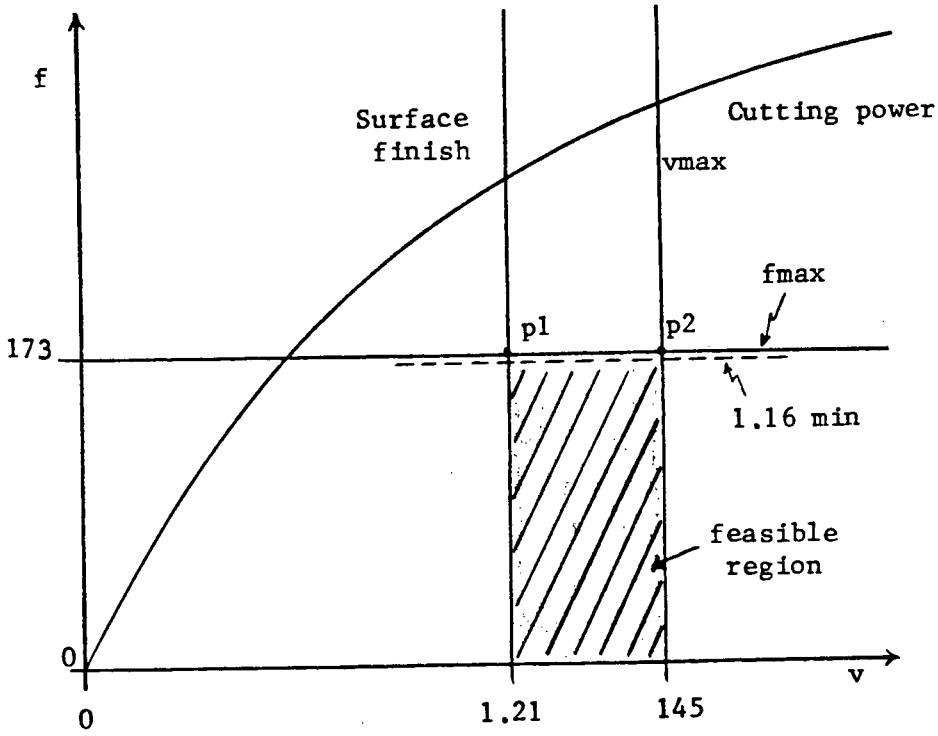
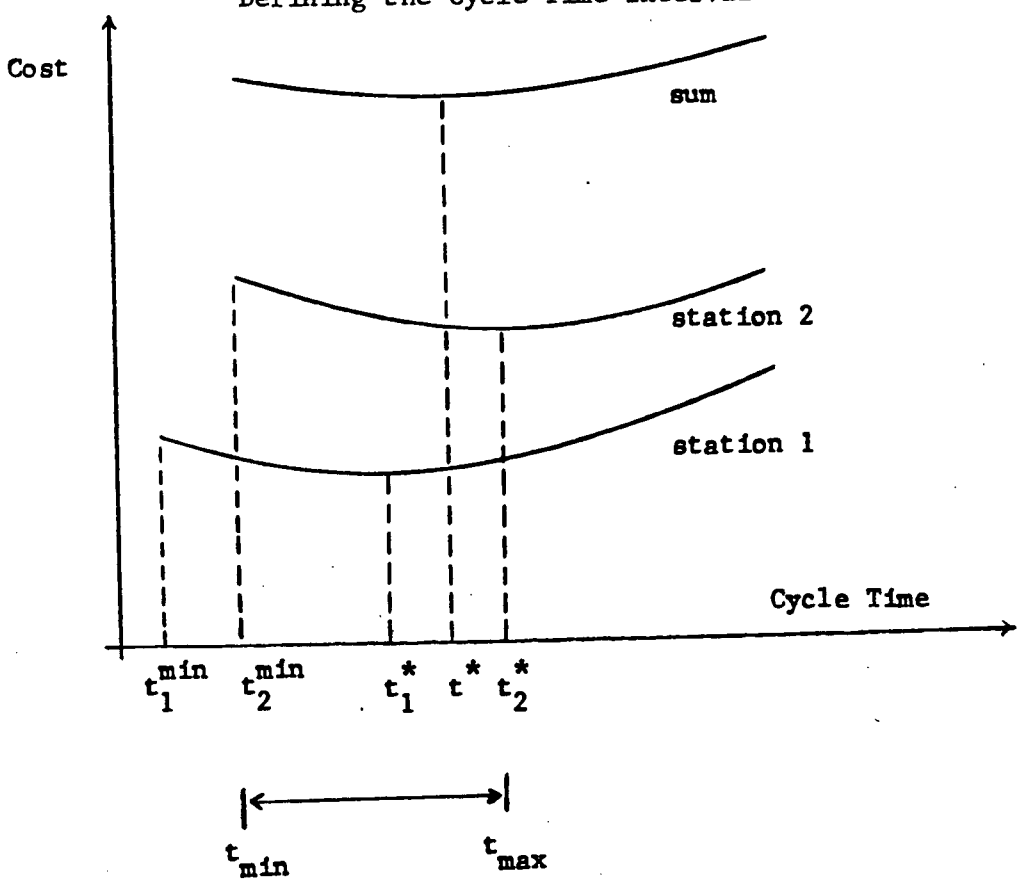


Figure 3-6

Defining the Cycle Time Interval



Note that the term Cm has disappeared from the objective function. This is because Cm = Overhead x cycle time. Once a cycle time is specified, Cm is just a constant and has no effect on the cost minimization. Note too that f has been dropped from (3.9); this is because the cycle time equality was used to solve for f in terms of v, as in (3.3). Given a cycle time T, the solution to (3.9) will yield a control pair (v,f) and a data pair (T,C).

Now we can look more closely at how to select an appropriate value for the degree p of the fitted polynomial. As p increases, the fit to the data improves but we need to generate more data points, requiring more computation. Typical computer methods need (p+3) points to fit a curve of degree p. Thus, the choice of p is usually a compromise between desired goodness of fit and available computing power. Let's return to the numerical examples presented earlier. In Figures 3-7,3-8,3-9, data points and fitted curves of various degrees are graphed for the three metal-cutting operations. Although turning and drilling show a good fit for smaller degree, it was decided to use one value p=6 to simplify the computer program.

In the next chapter, we will develop a new control strategy to minimize ATL cost when stations are subject to failure (and repair).

Figure 3-7
Turning :Data plus fitted curves

Data points generated for curve-fitting lie along p=6.

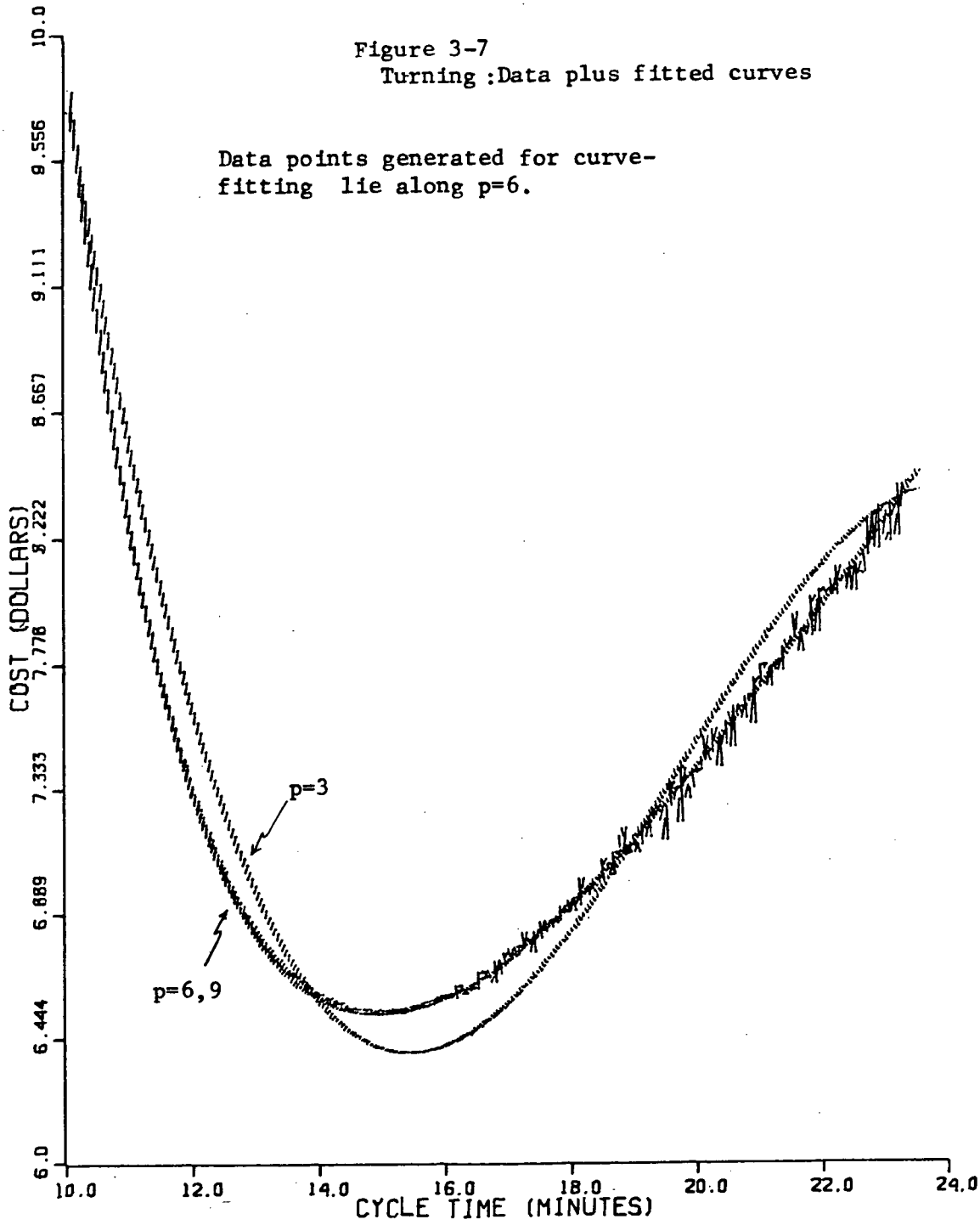


Figure 3-8

Drilling :Data plus fitted curves

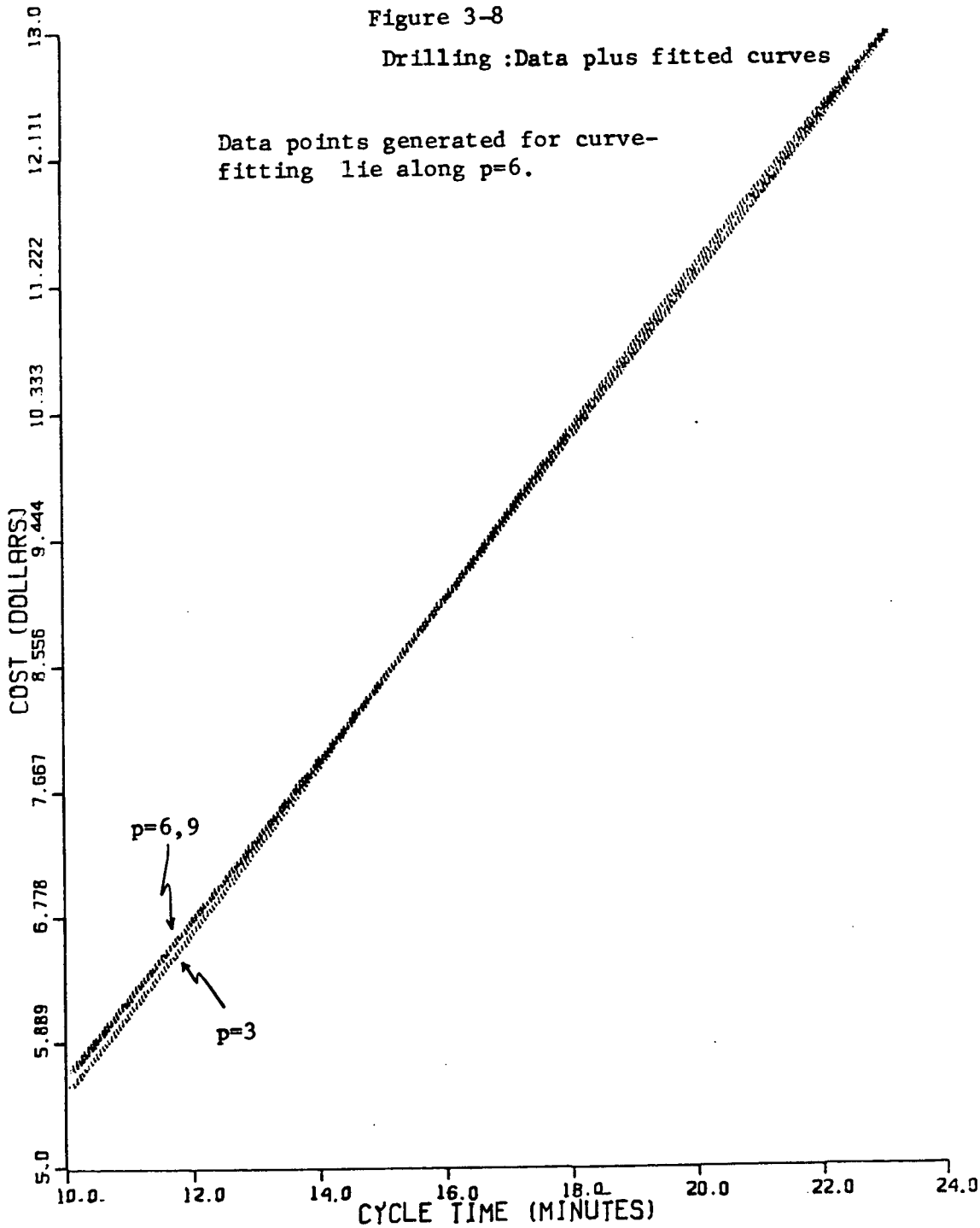
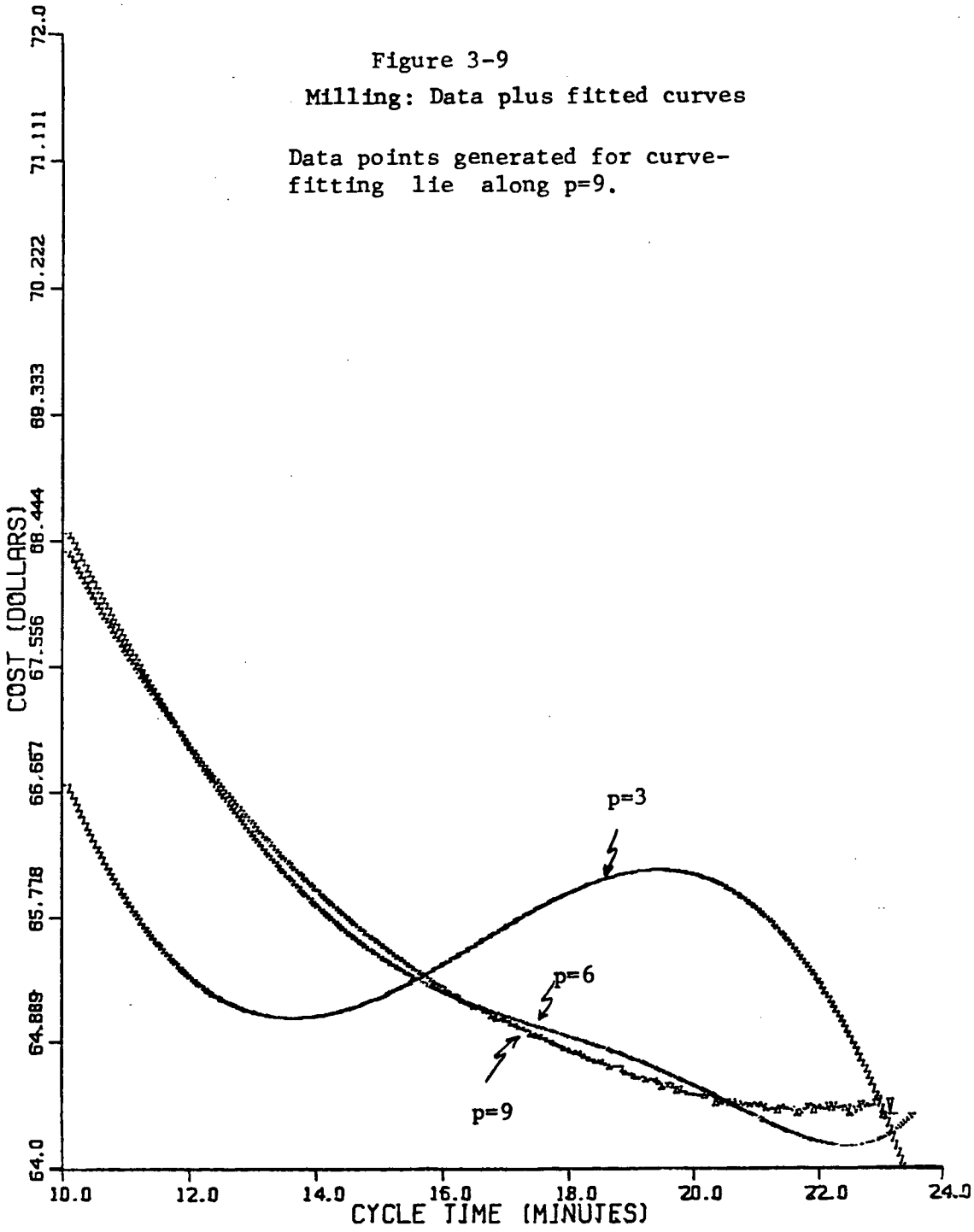


Figure 3-9
Milling: Data plus fitted curves

Data points generated for curve-fitting lie along p=9.



CHAPTER 4: THE DYNAMIC CYCLE TIME STRATEGY

In the previous chapter, the cost minimization of a transfer line was described using fitted curves to model the costs of the component work stations as functions of cycle time. What emerged was a single cycle time (to be assigned to each station) that would minimize total cost.

If machines never broke down, this would be the end of the story. However, stations are subject to failure and for high volume production runs, there could be substantial machine downtime. To see how this can affect the cost minimization problem, consider the two stage ATL of Figure 4-1.

As shown in Figure 4-2, the cycle time that corresponds to minimum total cost does not occur at the cost minimum of either M1 or M2. Now suppose that M1 fails. If in-process buffer B2 is not empty, M2 can still operate. In fact, its cycle time could be increased from t_{12}^* to t_2^* (slower machining) to reduce its cost; this would decrease the production cost of workpieces now in B2. When M1 is repaired, the cycle time of M2 must return to t_{12}^*. A similar case can be made for speeding up M1 from t_{12}^* to t_1^* when M2 fails, as long as buffer B2 does not overflow. This idea of re-assigning station cycle times in response to machine failure will be called the Dynamic Cycle Time (DCT) strategy. The alternative strategy, to be called the Fixed Cycle Time (FCT) strategy, makes no changes to cycle time when machines fail.

The implementation of the DCT strategy must satisfy two

Figure 4-1
A Two Station ATL

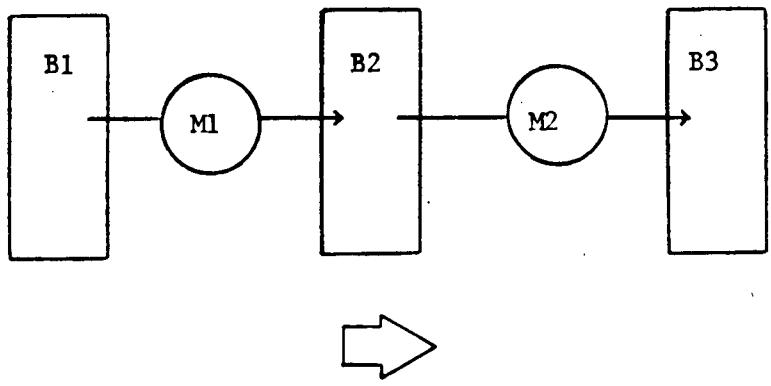
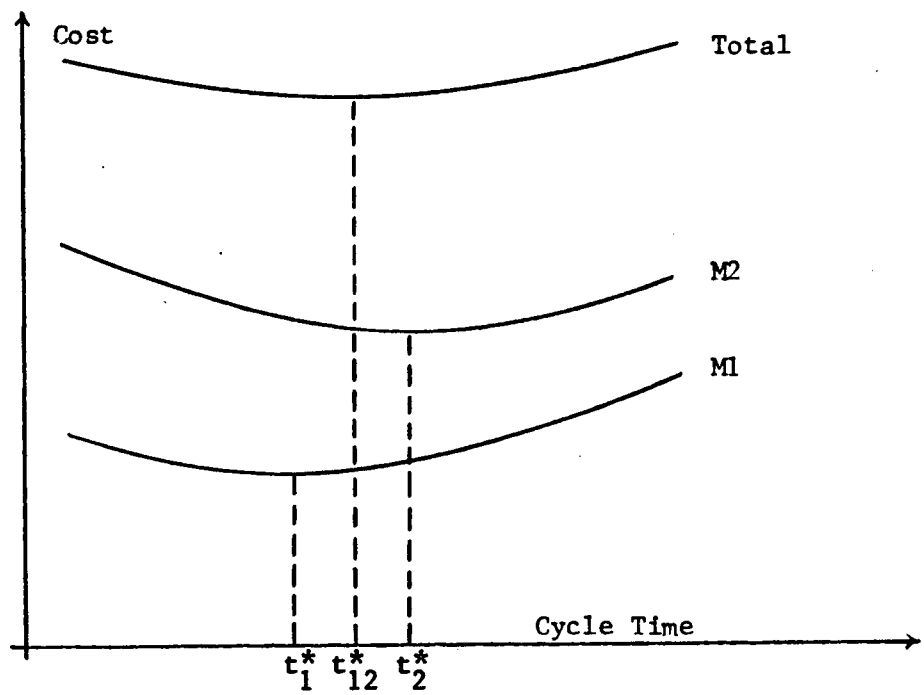


Figure 4-2 Combining Costs of two Stations



goals. Firstly, we need to know the status of each work station, in order to recognize the correct sub-sets of operating machines within the ATL. Secondly, we must be able to change the cycle time of each station on-line. To accomplish this, a two level control hierarchy is proposed, as sketched in Figure 4-3.

A Local controller is associated with each station, linked to a single Supervisory controller. The Local controller is responsible for the following activities:

(i) performs adaptive control of the metal-cutting machine (v,f) based on cycle time information supplied by Supervisory controller

(ii) monitors the machine status

(iii) gathers statistics (eg. tool wear) for on-line adjustment of the coefficients in the MEP equations

The Supervisory controller directs the DCT strategy in the following way:

(i) gathers information about the status and MEP coefficients from each work station

(ii) models costs as functions of cycle time

(iii) minimizes combined costs for appropriate sub-sets of the ATL, and determines the corresponding cycle times and control vectors

This kind of distributed control is in fact a recent development in modern manufacturing. Here is a quote from a

1981 conference on machine tool systems [12]:

> "Control systems will probably change from being central processor based to distributed processing systems. 'Local' processors ... will handle individual control functions with a control overseer system monitoring the 'local' activities and up-dating the sub-systems as required... Such systems would have been impractical both economically and technologically before now." (p.194)

To date, there has been some theoretical discussion of distributed control architectures in NC manufacturing [34,35]. In fact, this would just represent the lowest level in the Computer-Aided Manufacturing hierarchy [1]. Although information on installed systems is not available, there are NC implementations that could be refined to provide the kind of cost minimization suggested here.

When there are many stations in the ATL, the opportunity to re-assign cycle times occurs more frequently. Consider the three stage line sketched in Figure 4-4. When M1 fails, the cost of M2 might be minimized with the cost of M3 or by itself, depending upon the status of M3. There are really six groups of adjacent stations within the ATL for which we can determine optimal cycle times:

M1 alone (when M2 and M3 are down)
M2 alone (when M1 and M3 are down)
M3 alone (when M1 and M2 are down)
M1 + M2 (when just M3 is down)
M2 + M3 (when just M1 is down)
M1 + M2 + M3

There is a simple formula to determine the number of such sub-sets of an ATL with N stations:

Figure 4-3 Proposed Control Architecture

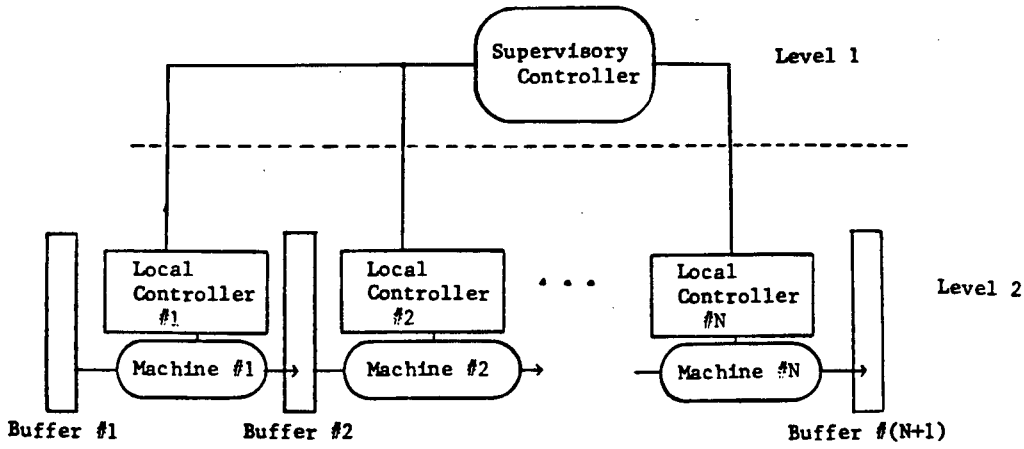
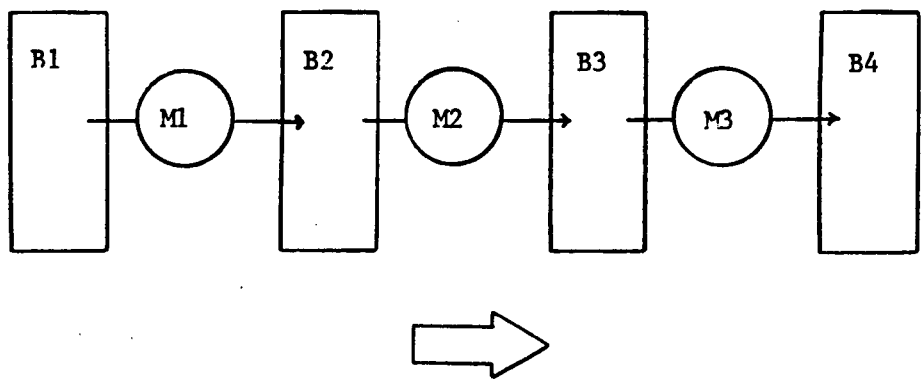


Figure 4-4

A Three Station ATL



$$\begin{aligned} \text{\# sub-sets} &= 1 \text{ combination of all N stations} \\ &\quad + 2 \text{ combinations of (N-1) stations} \\ &\quad + \ldots + N \text{ combinations of 1 station} \\ &= \sum_{i=1}^{N} i \\ &= N\,(\,N+1\,)\,/\,2 \end{aligned} \qquad (4.1)$$

For each sub-set, there is an optimal cycle time that will minimize the combined cost of the component stations. The cost minimization is accomplished by summing the appropriate fitted cost polynomials (all of degree p) to obtain a new polynomial of degree p, setting the first derivative to zero, solving for the (p-1) roots of that polynomial, and then isolating the one that corresponds to minimum cost.

In Table 4-1, optimal cost and cycle time data are listed for the six sub-sets of a three stage ATL based on the numerical data of the previous chapters.

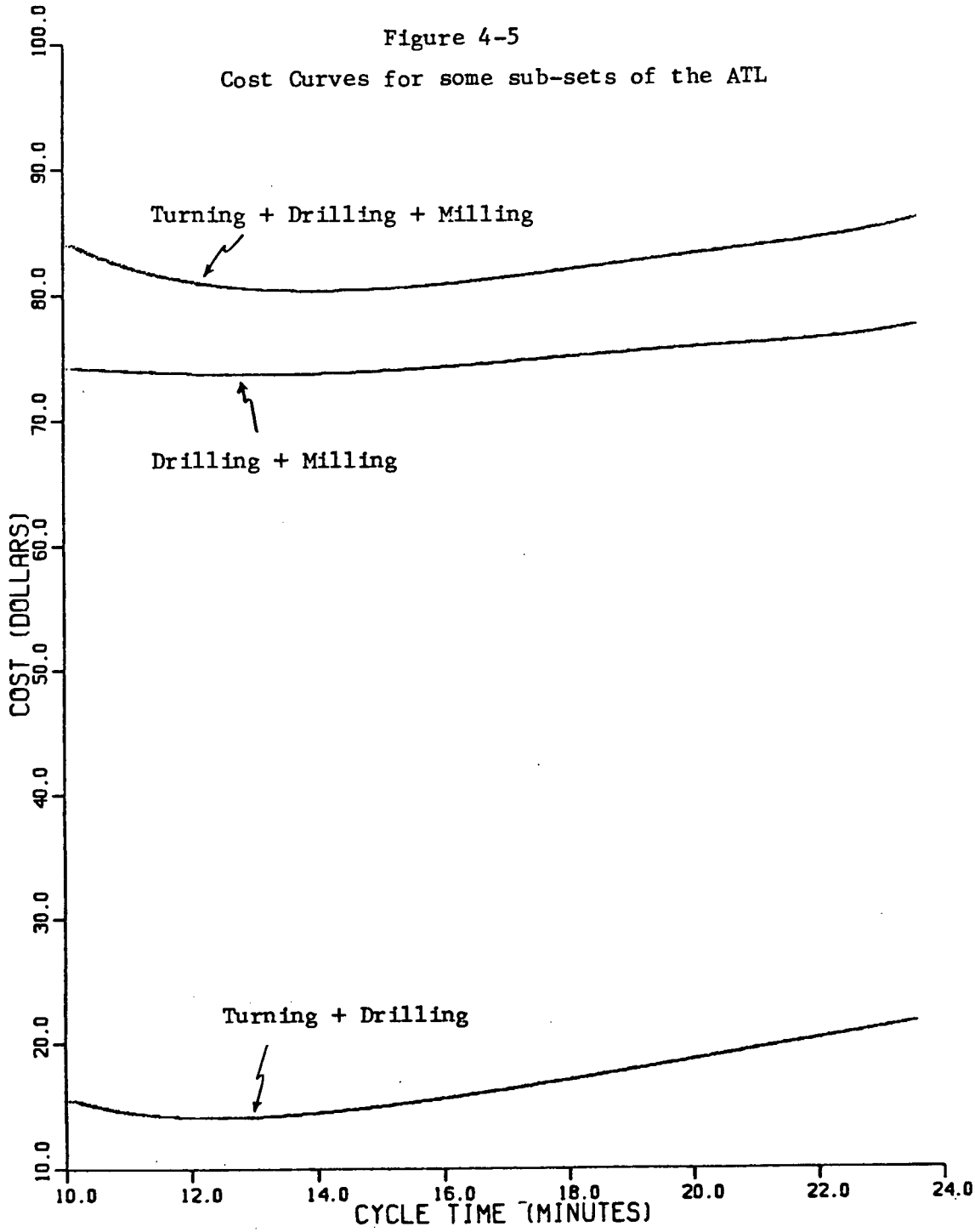
TABLE 4-1: Optimal Cost and Cycle Time Data for the Three Station ATL

Operation(s)	Optimal cycle time (min)	Optimal cost(s) ($/piece)
Turning T	14.89	6.53
Drilling D	1.49	0.94
Milling M	23.46	64.36
T + D	12.30	7.10 + 6.95
D + M	11.39	6.34 + 65.64
T + D + M	14.46	6.47 + 8.17 + 65.78

In Figure 4-5, fitted curves are shown for three of the

Figure 4-5

Cost Curves for some sub-sets of the ATL



sub-sets. They are based on the individual cost curves of Figures 3-7, 3-8, and 3-9.

To evaluate the relative merit of the DCT strategy, some typical ATL configurations adapted from [27 to 30] were selected for simulation. The computer program developed for the simulation is detailed in Appendix B.

For the purpose of simulation, there are three main features that characterize the performance of an ATL:

(i) the probability distributions for machine failure and repair

i.e. Pf[n,T] = prob[n failures over a time T]

Pr[1,T] = prob[1 repair takes time T]

(ii) the number of stations N

(iii) the size(s) of the in-process buffer(s)

Let Time-To-Fail TTF(t) = prob[waiting time t between succesive failures]

This is more useful for simulation work than Pf(n,t). To see how they are related, assume that failure is modelled as a Poisson process:

i.e. $Pf[n,T] = e^{-\lambda T} (\lambda T)^n / n!$

Now we can use a negative exponential distribution for the waiting time [36].

$$TTF(t) = \lambda e^{-\lambda t} \qquad (4.2)$$

and Mean-Time-To-Fail $MTTF = 1/\lambda$ (4.3)

Machine repair can be similarly modelled:

$$\text{Time-To-Repair} \quad TTR(t) = \mu e^{-\mu t} \qquad (4.4)$$

and Mean-Time-To-Repair MTTR = $1/\mu$ (4.5)

The following values were selected for the failure and repair distributions:

MTTF : 60 machine cycles (worst case)

270 machine cycles (best case)

MTTR : 6 machine cycles (typical case)

Note that all stations in the ATL are modelled with the same TTF and TTR distributions, although they perform different metal- cutting operations. This is a standard analytical convenience for simulation work of this kind. New TTF and TTR values for each station are 'drawn' at random from the frequency distributions after each machine failure and subsequent repair.

Before proceeding, let's consider these values selected for simulation. If MTTR=1 cycle (on average, repair after one machine cycle), we would expect the DCT strategy to show little cost savings compared to the FCT strategy, since any change to the station cycle time would only affect at most 1 workpiece; this would be true regardless of how often machines failed. However, if MTTR>>1 and MTTF is so large that machines rarely fail, we would again expect DCT to have little effect, since there would be few opportunities to make changes to the cycle times of the stations. Thus the values listed above are reasonable choices.

At this point, we need to define the component work stations of the ATL: the number, and the kinds of operations to

be performed. Let's look at the possible choices in Table 4-1. Note that milling costs are an order of magnitude greater than the costs of turning and drilling. This means that milling would completely dominate the total cost of a three stage ATL. This also means that the DCT strategy would not show significant cost savings compared to the FCT strategy unless the cycle time changes for milling resulted in substantial cost savings. In fact, this is not the case; the milling cost changes very little as cycle time changes from 11.39 min to 23.46 min. Because of this, a two station ATL based on just turning and drilling operations was simulated.

According to [37], 33% of all ATL configurations in the West have just two or three work stations (although 55% have four to six). Moreover, 13% involve turning and 45% involve drilling (although 61% involve milling). Thus, this ATL configuration is a reasonable choice for simulation.

The size of the in-process buffers in an ATL determines the degree of coupling between adjacent machines. As the size increases, a buffer can better 'cushion' the effects of different cycle times and even machine failures. However, the extra number of partly finished workpieces in each buffer represents additional in-process inventory which increases production cost. The following quote from [38] describes this trade-off in more detail:

> "In theory, when the size of the buffer storage b is greater than or equal to 2T/t [T=MTTR, t=cycle time], the workheads [stations] are completely isolated one from the other and further increases in the size of b are futile. However, the greatest benefit occurs with the smaller buffers [compared to no buffers]. So as a practical guide, it is often assumed that for a good free-transfer [automatic] ... machine design, b equals T/t." (p.216)

Using T=6 cycles and t=1 cycle, we have b=6. The three values used in the simulation were 5,10, and 15.

Two important statistics that characterize a production run are mean unit cost/workpiece and mean in-process buffer level. When simulating a stochastic process, we must be careful that the statistics of means properly reflect the process steady state. In effect, we must run the simulation 'long enough' to gain the desired statistical confidence. It's clear that the expected improvement in performance (if any) due to the DCT strategy will be related to the frequency of station failure; when stations fail more often, there are more opportunities to re-assign cycle times. Thus it's appropriate to measure the 'length' of a trial in terms of MTTF. After some experimentation, it was discovered that sufficient statistical accuracy (90%) could be obtained by averaging the results of four trials each of 'length' 150 failures. This issue is described in more detail in Appendix C.

A third performance variable, the % utilization of M2, was also recorded. This indicates how much of the time M2 was idle (starved) because the upstream buffer was empty. The % utilization of M1 was not recorded because the input buffer (by assumption) is never empty. In non-adaptive NC systems, idle

time is a direct measure of performance: more idle time means lower machine productivity, and extended machine life. However, in AC systems, lower machine utilization is not in itself a sign of poor performance. In fact, if the productivity remains acceptable, more idle time might be beneficial by prolonging the life cycle of the machine.

The following discussion of the simulation work is not intended to be an exhaustive analysis of transfer line optimization. The attempt is made however, to describe how the performance of some sample ATL's changes under the DCT control strategy when compared to the FCT strategy. It is hoped that the results will be sufficiently encouraging to confirm the value of modelling cost as a function of cycle time, and the value of using GP to solve the associated MEP's.

First let us return to Table 4-1. Under the DCT strategy, the drilling station is speeded up by 88% (from 12.30 min to 1.49 min) and its unit cost is lowered by 86% (from $6.95 to $0.94) when the turning station is down. Similarly, when the drilling station is down, the turning station is slowed down by 21% (from 12.30 min to 14.89 min) and its unit cost is decreased by 8% (from $7.10 to $6.53). Thus, we would expect that the mean level of the in-process buffer to be smaller for a transfer line arranged as Turning -> Drilling, instead of the other way around. To better compare the behavior of these two configurations, both were simulated with identical buffer sizes, TTF and TTR distributions.

Let's look at the results for M1=turning -> M2=drilling.

In Table 4-2, data for mean unit cost, mean buffer level, and The relative changes in each variable when comparing DCT to FCT are shown in Figures 4-6,4-7, and 4-8. In each graph, the horizontal axis is buffer size.

In general, the DCT strategy lowers unit cost and mean buffer level, and also reduces M2 utilization. Note that these changes become more significant as MTTF decreases i.e. as the machines fail more frequently. As shown in Figure 4-6, the % cost savings is small. This is because the mean buffer level under the DCT strategy is small (less than 4) so that there are few workpieces that 'benefit' from the reduced M2 cost when M1 is down. It's also true that the M1 cost is lowered when M2 is down but because M1 cycles so slowly, it does not process many workpieces at this reduced cost. The cost savings does increase slightly with buffer size since the mean buffer level rises.

On the other hand, the % buffer savings is large, as shown in Figure 4-7. This is because the mean repair time is long enough (6 cycles) to allow M2 to substantially reduce the mean buffer level (while M1 is down) from the FCT value. This also explains why the % buffer savings is largely independent of MTTF.

In Figure 4-8, we see that % M2 utilization decreases as the buffer size increases. We would expect the opposite effect, since a larger buffer means more workpieces available to M2 when M1 is down. A glance at Table 4-2 reveals one explanation: even when the buffer size is large, the mean level under the DCT

TABLE 4-2

Simulation Resluts for the Turning→Drilling ATL

MTTF (cycles)	Buffer size (# pieces)	Strategy	Mean unit cost ($)	Mean buffer level (# pieces)	% M2 utilization
270	5	DCT	13.960	1.405	96
		FCT	14.014	1.569	96
	10	DCT	13.923	2.797	96
		FCT	14.014	4.094	97
	15	DCT	13.901	3.506	96
		FCT	14.014	6.733	97
60	5	DCT	13.816	1.397	82
		FCT	14.059	1.683	86
	10	DCT	13.656	2.678	82
		FCT	14.059	4.219	87
	15	DCT	13.581	3.401	82
		FCT	14.059	6.855	88

Figure 4-6

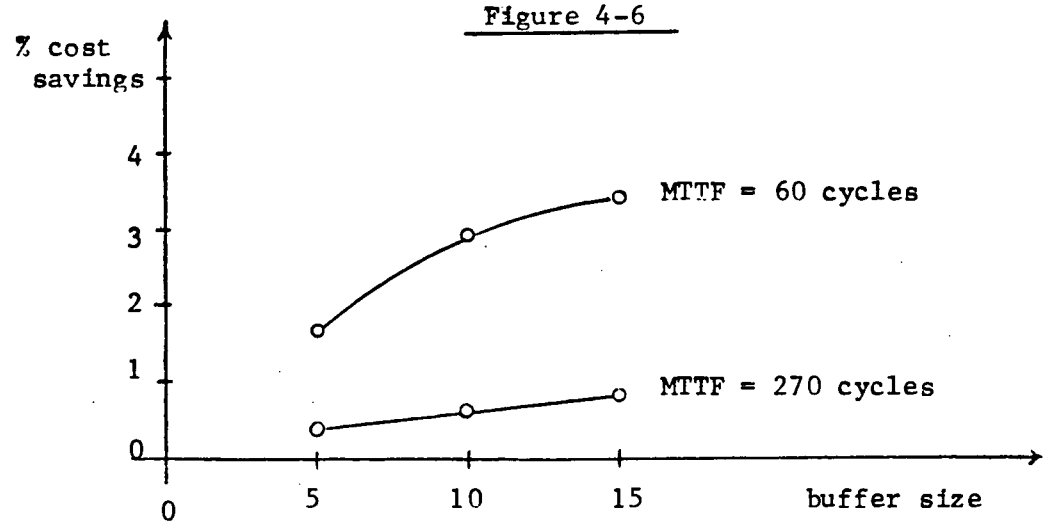


Figure 4-7

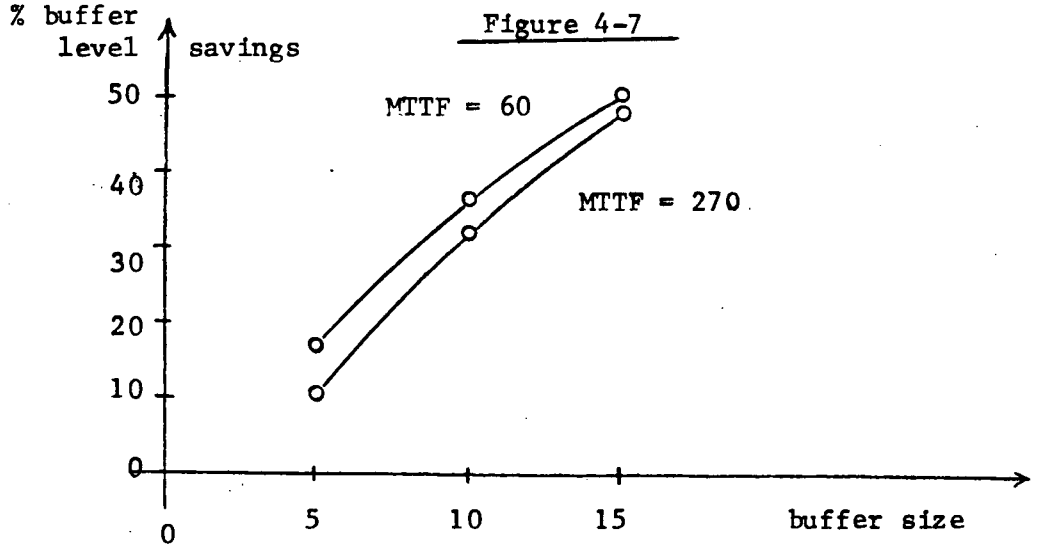
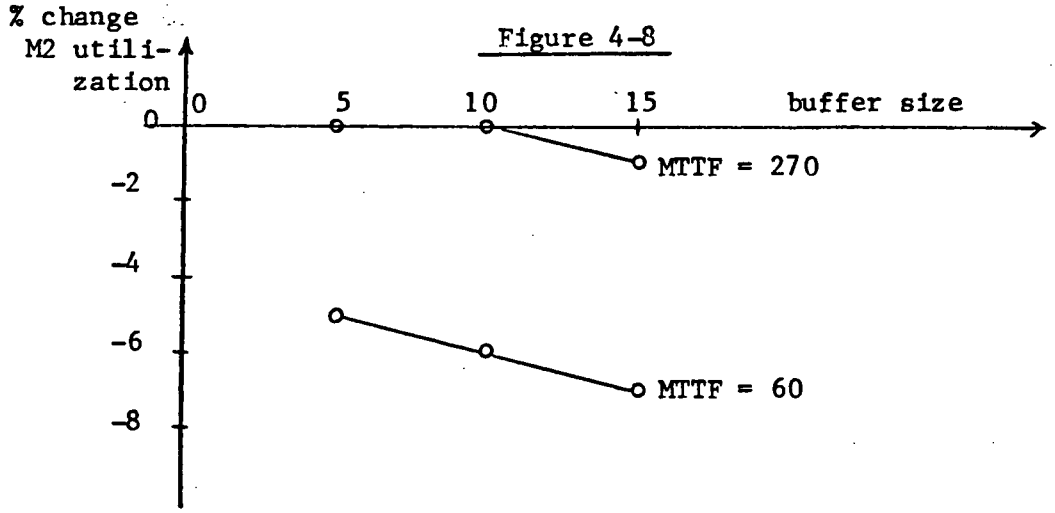


Figure 4-8



strategy is still small, so that the % M2 utilization cannot increase. However, this does not explain why it decreases slightly. In any event, this is a desirable effect. Although we seem to be 'wasting' more machine resources when utilization decreases, we are really just interested in the performance of the ATL as a whole.

Now let's proceed to the results of the second simulation. Here we have M1=drilling -> M2=turning. The data is listed in Table 4-3 and the % changes are sketched in Figures 4-9,4-10, and 4-11.

In general, the DCT strategy still lowers unit cost but now the mean buffer level increases and the % M2 utilization rises slightly too. Note that the % cost savings shown in Figure 4-9 almost duplicates the savings shown in Figure 4-6 (for the other ATL configuration). This is because the real cost savings is due to the reduced cost of drilling when the turning station is down. In fact, the data in Table 4-3 indicates that the reduced unit costs under the DCT strategy are almost the same for both transfer line configurations.

Although the % cost savings has not changed, the curves in Figures 4-10 and 4-11 are almost mirror images of Figures 4-7 and 4-8. Since the cycle time of M2 substantially increases under the DCT strategy when M1 is down, the mean buffer level rises as shown in Figure 4-10. Also note that the increase is largely unrelated to MTTF since M2 can only process a few workpieces while M1 is under repair.

TABLE 4-3

Simulation Results for the Drilling→Turning ATL

MTTF (cycles)	Buffer size (# pieces)	Strategy	Mean unit cost ($)	Mean buffer level (# pieces)	% M2 utilization
270	5	DCT	13.964	1.699	96
		FCT	14.014	1.532	96
	10	DCT	13.926	5.459	97
		FCT	14.014	3.911	97
	15	DCT	13.905	9.591	97
		FCT	14.014	6.229	97
60	5	DCT	13.834	1.904	85
		FCT	14.059	1.640	85
	10	DCT	13.674	5.578	88
		FCT	14.059	4.012	86
	15	DCT	13.576	9.926	89
		FCT	14.059	6.300	87

Figure 4-9

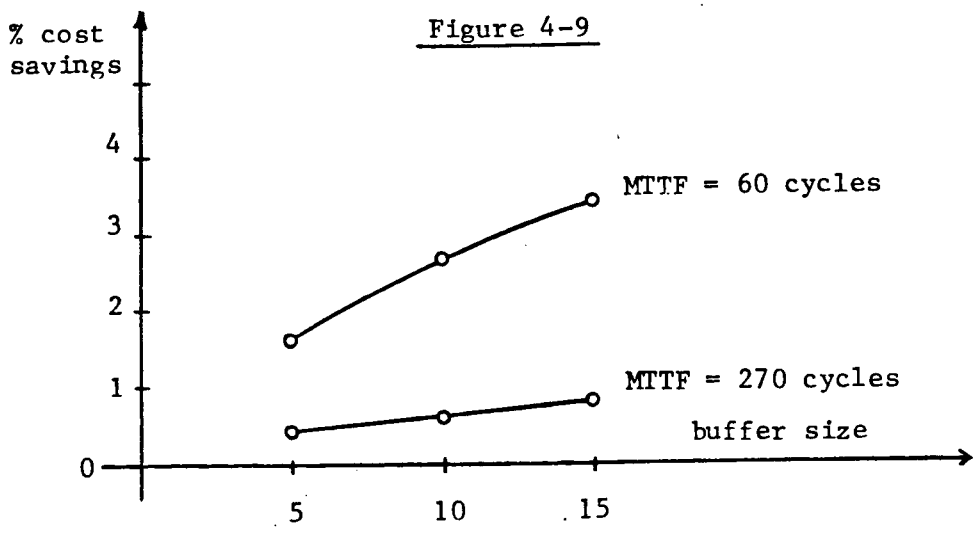


Figure 4-10

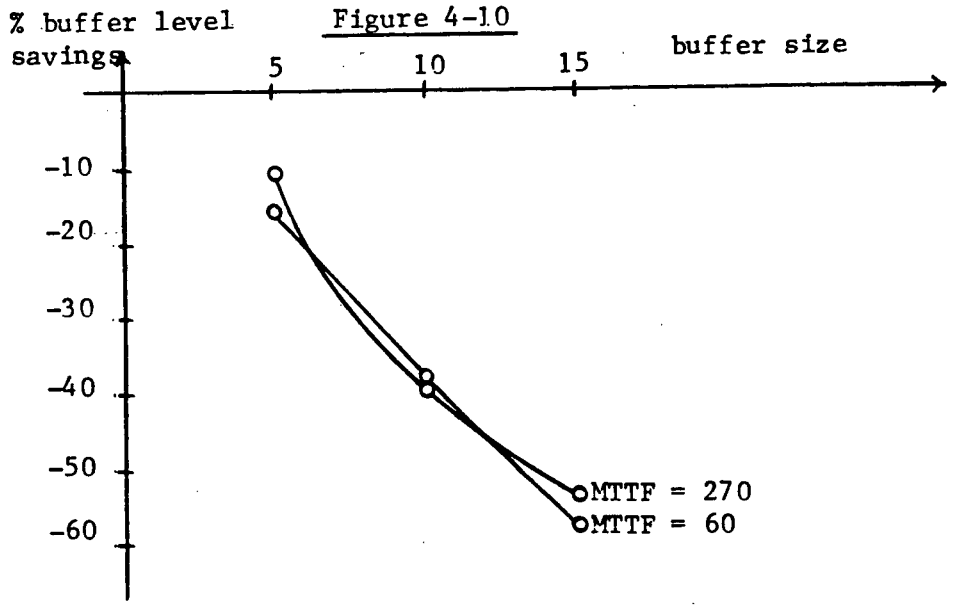
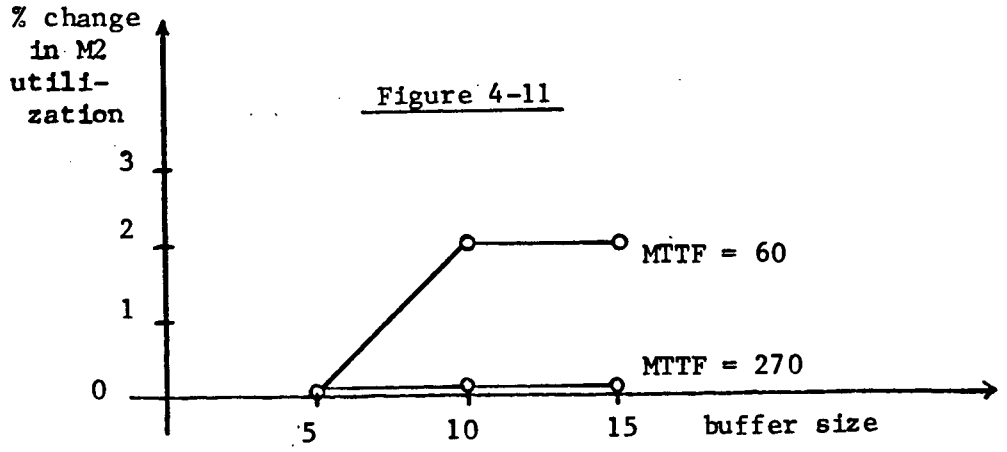


Figure 4-11



Finally, the % M2 utilization increases slightly under the DCT strategy but only for MTTF=60 cycles, as shown in Figure 4-11. This is because % utilization for MTTF=270 cycles is almost 100% (see Table 4-3); this indicates that M2 is very rarely idle because the buffer is empty. When failures occur more frequently (smaller MTTF), this idle time is reduced under the DCT strategy since the mean buffer level is higher.

CHAPTER 5: Conclusions and Suggestions for Future Work

In this thesis, a new control strategy called DCT (Dynamic Cycle Time) was developed for optimizing the performance of an ATL (Automatic Transfer Line). The basic idea is the modelling of the cost of each work station as a function of its cycle time. A Geometric Programming technique called the decomposition method was used to re-interpret the Machining Economics Problem as a set of (cycle time,cost) pairs to which a polynomial curve was fitted. The function that emerges is a new and powerful description of how cost varies with cycle time and what the smallest cycle time can be.

To minimize the combined costs of several stations, we sum the individual cost functions and then locate the minimum of the sum using simple differential calculus. When stations fail, the ATL is 're-configured' as sub-sets of operating stations. A new cycle time is then assigned to each sub-set that minimizes the combined cost of the component stations. For the sake of comparison, the strategy of no change in station cycle time was defined and called FCT (Fixed Cycle Time).

To compare the control strategies, some typical two station ATL's were simulated based on turning and drilling operations. It was shown that the DCT strategy results in modest cost savings and a substantial reduction in mean buffer level when the operations are suitably ordered (Turning->Drilling). In fact, simulations of the DCT strategy also indicate that there is a performance optimal ordering of operations which should be

used if technically feasible.

It is hoped that the simulation results are sufficiently encouraging to demonstrate the value of the DCT strategy and and the value of modelling the MEP as a cost function of cycle time. It should also be clear to the reader that Geometric Programming is a simple and powerful tool for solving constrained non-linear optimization problems. In particular, for problems of low order, the decomposition method was shown to be a fast and reliable (algorithmic) technique suitable for real-time microprocessor implementations.

There are many aspects of this thesis that could be explored further:

(i) The DCT and FCT strategies were compared using numerical data obtained from separate papers on adaptive turning and drilling. It would of course be better to use real data from a working ATL in industry. This would likely mean extending the simulation model to three or more stations.

(ii) We have seen that the ordering of operations in the ATL is a very important consideration for the DCT strategy. When there are many operations, it would be hard to anticipate which ordering would be optimal. Thus, the computer program should be extended to simulate all N! orderings of the N operations in a given ATL.

(iii)The performance of a given ATL under the DCT strategy compared to the FCT strategy is dependent on several factors:

the frequency of machine failure (MTTF), the machine repair time (MTTR), the size of the in-process buffer(s), and the relative change in the cycle times of each station across the various ATL sub-sets it belong to. It would be worthwhile trying to develop some figure of merit for each of the performance variables (eg. mean unit cost) based on these factors as an alternative to simulation studies.

(iv) The DCT strategy could be modified to direct appropriate stations to shut down (for short intervals) instead of waiting for machines to fail. In this way, the Supervisory controller could monitor and periodically optimize the in- process buffer levels. Alternatively, the Supervisory controller could also anticipate routine maintenance on a given work station by causing its upstream buffer to be emptied and its downstream buffer to be filled.

(v) Rather than operate work stations in an ATL continuously with one cycle time that minimizes their combined cost, it might be feasible to operate them in 'bursts'; each station would cycle so as to minimize its own cost and rely on buffer underflow/overflow to temporarily suspend execution. This strategy might be useful when the optimal cycle times of the stations are very different, and when the in-process buffers are large.

(vi) After turning, drilling, and milling, the fourth machining operation is grinding. The MEP formulation for grinding [5,7] involves a posynomial Cutting power constraint which cannot be

directly tackled by the decomposition method. This means that we would need an iterative approach. It would be worthwhile comparing the condensation and the reduced dual space methods of Chapter 2 using numerical examples of the grinding MEP.

(vii)In Chapter 3, we modelled cost as a polynomial function of cycle time. There are other curve-fitting methods that might be considered eg. cubic splines or Fourier series.

APPENDIX A: CONTROL DETAILS FOR THE MACHINING OPERATIONS

The basic turning operation is shown in Figure A-1.

D = workpiece diameter (in)
L = length of cut (in)
d = depth of cut (in)
ω = rotational speed of workpiece (rev/min)
f = feedrate (in/rev)

v = cutting speed (surface ft/min)
$= \omega \pi D/12$

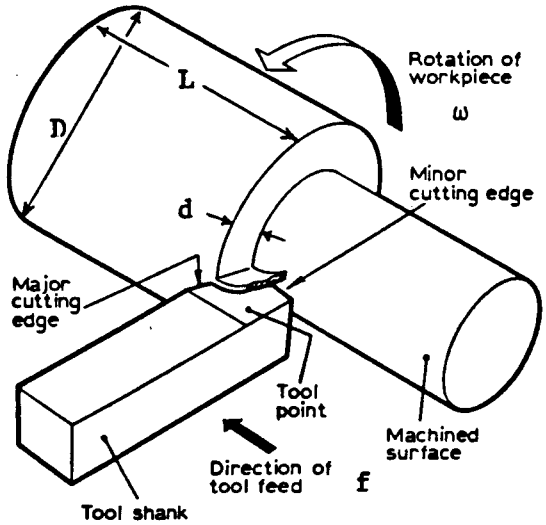


Figure A-1 Control Details for Turning

The following equations and numerical data for the MEP were adapted from [20,21,22]. Tm is the machining time (min), T is the tool life (min), and Tr is the tool replacement time (min).

$$Tm = L\omega^{-1} f^{-1} = L\,(\,\pi\ D\ /\ 12\,)\ v^{-1} f^{-1} \qquad (A.1)$$

$$T = (\ 7.5E3\ /\ d\ )\ v^{-5}\ f^{-2.15} \qquad (A.2)$$

with D = 3.0 in
L = 10.0 in
d = 0.1 in
X = 0.351 $/min
Ct = $0.487 (throw-away type)
Tr = 1.0 min

The four constraints for the MEP are as follows:

$$\text{Cutting power} = 23\ d\ v\ f^{0.76} <= 5\ HP \qquad (A.3)$$

$$\text{Surface finish} = f <= 0.014 \qquad (A.4)$$

vmax = 600 sft/min fmax = 0.02 in/rev

The basic drilling operation is shown in Figure A-2.

D = drill diameter (in)
L = length of cut (in)
ℓ = tool length (in)
ω = spindle speed (rev/min)
f = feedrate (in/rev)

v = cutting speed (ft/min)
= $\omega\pi D/12$

Here, the following equations and numerical data were adapted from [21,2]:

$$Tm = L\,\omega^{-1} f^{-1} = L\,(\,\pi\,D\,/\,12\,)\,v^{-1} f^{-1} \qquad (A.5)$$

$$T = 1.324E9\ v^{-9.8}\ f^{-4.9} \qquad (A.6)$$

with D = 0.5 in
L = 1.5 in
ℓ = 3.25 in
X = 0.565 $/min
Tr = 6.0 min

and Ct = re-sharpening cost
+ new too cost / # possible re-sharpenings
= 3.90 + 97.44/10
= $13.64

The four constraints are as follows:

$$\text{Cutting power} = (\ 0.174\ BHN\ D^{0.8}/33000\ )\ v\ \ f^{0.8} \qquad (A.7)$$
$$<= 40\ HP$$

where BHN = Brinnell Hardness Number of the workpiece
= 165.0 (cast iron)

$$\text{Surface finish} = f <= 0.0175\ (\ \ell / D\ )\ D^{0.6} \qquad (A.8)$$

vmax = 88 ft/min fmax = 0.30 in/rev

The third metal-cutting operation considered in this thesis is milling, shown in Figure A-3.

D = tool diameter (mm)
L = length of cut i.e. workpiece length (mm)
d = depth of cut (mm)
N = number of teeth on cutting tool
W = width of tool (mm)
ω = spindle speed (rev/min)
f = feedrate (mm/rev)

v = cutting speed (m/min)
= $\omega\,\pi D\,/\,1000$

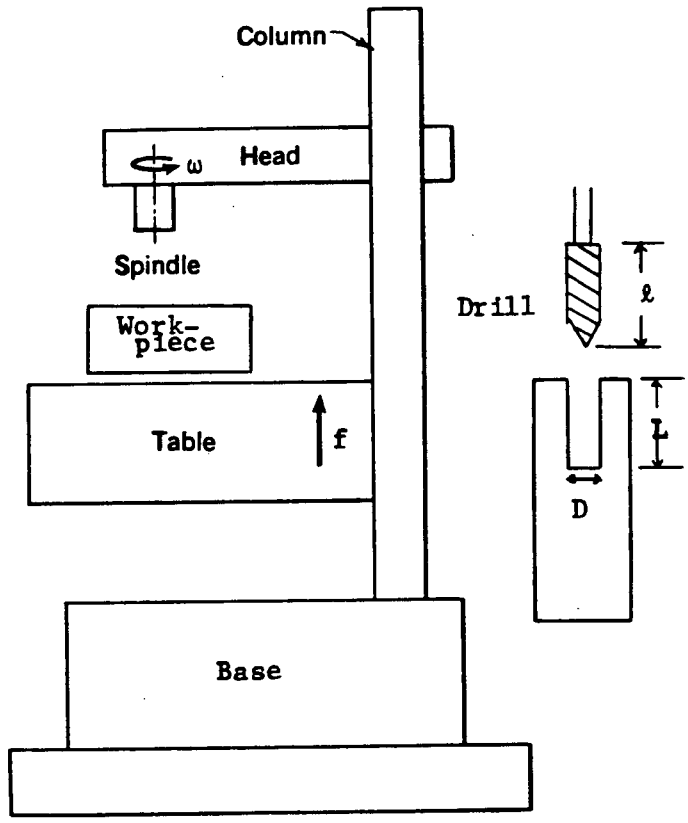
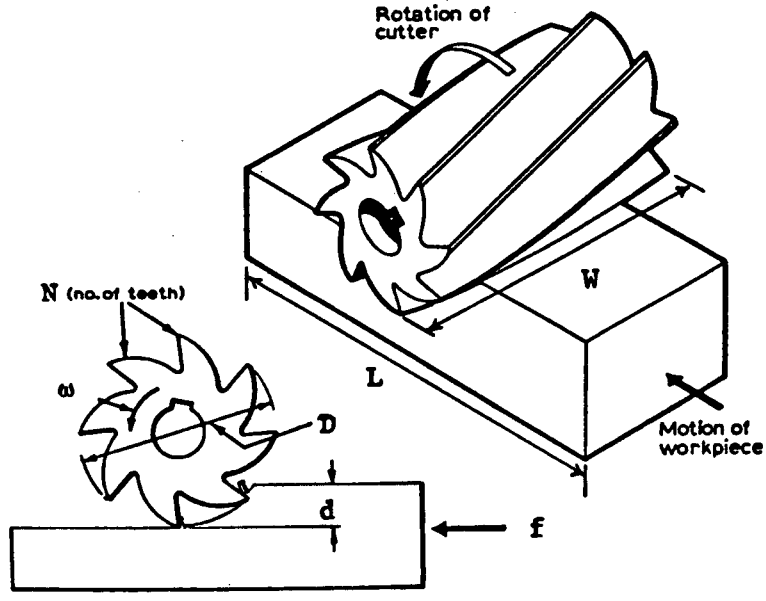


Figure A-2 Control Details for Drilling

Figure A-3 Control Details for Milling



The following equations and numerical data were adapted from [21,23]:

$$Tm = L\, f^{-1} \qquad \text{(A.9)}$$

$$T = 3.55E4/N^{0.757}\,(\,\pi D/1000\,)^{1.818}\, v^{-1.818}\, f^{-1.212} \qquad \text{(A.10)}$$

with D = 38.4 mm
L = 200 mm
d = 1.0 mm
N = 3
W = 30 mm

X = 0.480 $/min
Ct = $38.974 (throw-away type)
Tr = 1.33 min

The constraints are as follows:

$$\text{Cutting power} = \frac{146.6\, W^{1.1786}\, d^{0.9128}\, 60}{D^{1.1713}\, N\, 6010.2}\left(\frac{\pi D}{1000}\right) v^{-0.556}\, f^{0.7509} \qquad \text{(A.11)}$$

$$\leq 7.5 \text{ KW}$$

$$\text{Surface finish} = (\,\pi\, D\,/\,1000\,)\, v^{-1} <= 0.1 \qquad \text{(A.12)}$$

vmax = 145 m/min fmax = 173 mm/min

APPENDIX B: NOTES ON THE COMPUTER PROGRAM

COMPUTER PROGRAMMING NOTES

This thesis work has two main themes: analysis and simulation. It was therefore important to select a programming language that could properly serve both needs. SIMULA is a general purpose simulation language for discrete time event modelling. It is a super-set of ALGOL 60, and thus encourages structured programming. Another important feature is the CLASS declaration which is used to create identical 'processes' via a single template. Finally, SIMULA is transportable since there exists just one licensor: The Norwegian Computing Centre of Oslo, Norway. For these reasons, SIMULA was selected as the target language for this thesis. More information about programming in SIMULA can be found in [39,40].

The program CLASS hierarchy is shown in Figure B-1. The FACTORY comprises a Supervisory Controller and three kinds of Local Controllers, each associated with a different metal-cutting operation. The fourth class, MACHINE, will be discussed later.

The Local Controllers use specific workpiece data (eg. length of cut) with fixed machining data (eg. tool size) to define the MEP's of specific work stations, as shown in Figure B.2. The Supervisory Controller then analyzes each work station separately. First, the MEP is solved using the GP decomposition method. The dual set of simultaneous equations is solved using an MTS program called FSLE [41]; this yields the minimum cost and optimal cycle time. Next, a lower bound on cycle time is determined from the constraints and an upper bound from the optimal cycle times of the individual work stations. Finally, (cycle time,cost) data are generated for the region of interest and the data is fit to a polynomial curve. The curve fitting uses an MTS program called DOLSF [42].

At this point, the curves of selected stations are summed to represent their combined cost, and the minimum is obtained by setting the first derivative to zero. This requires another MTS program RPOLY2 [43] that calculates the zeros of a polynomial (since the derivative of a polynomial is itself a polynomial). Once minimum cost and optimal cycle time data are obtained for all appropriate sub-sets of the ATL, the analysis is complete (Figure B.3).

The CLASS MACHINE is used to generate a specific transfer line composed of several work stations with common failure and repair distributions, a set of buffers, and one Supervisory Controller (Figure B.4). The status of a station is modelled as UP (working), DOWN (failed), STARVED (upstream buffer empty), BLOCKED (downstream buffer full), or JUST REPAIRED.

The interaction of these CLASS's within the two level hierarchy is outlined in Figure B.5 and Figure B.6. Note that the Supervisory Controller does not monitor the ATL. When a Machine is ready to start on a new workpiece, it signals the Supervisory Controller to review the status of the ATL and assign the proper cycle time (and cost data) to that Machine.

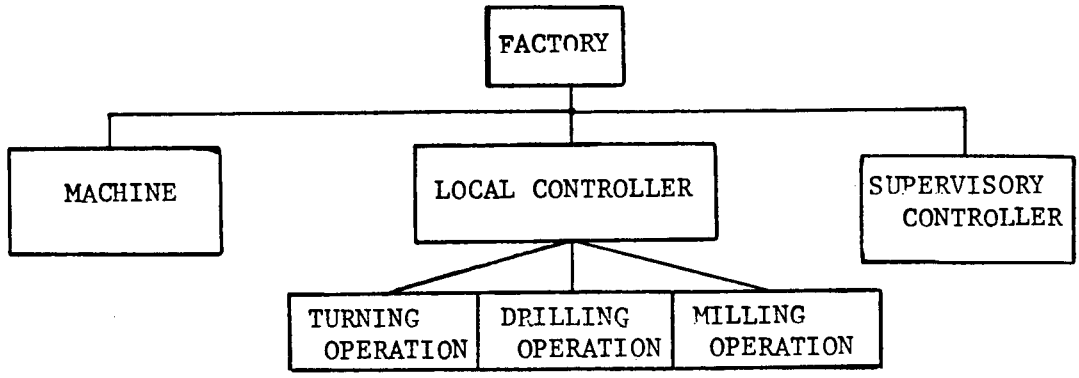


Figure B-1 Program Class Hierarchy

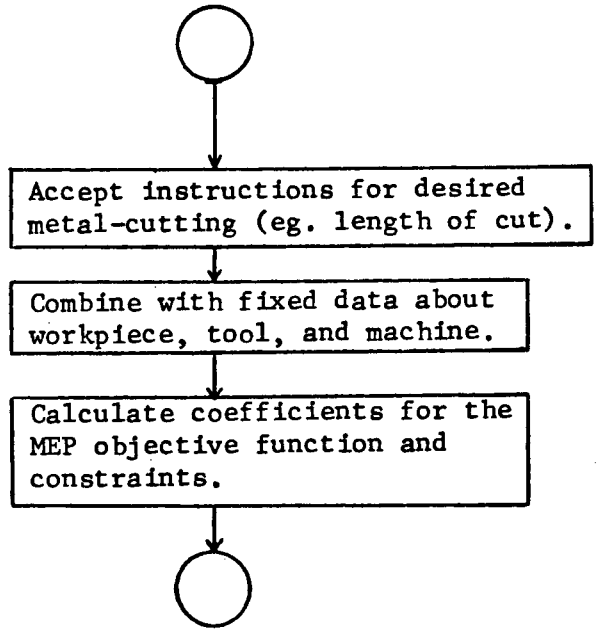


Figure B-2

Analysis Flow Chart for Class: Local Controller

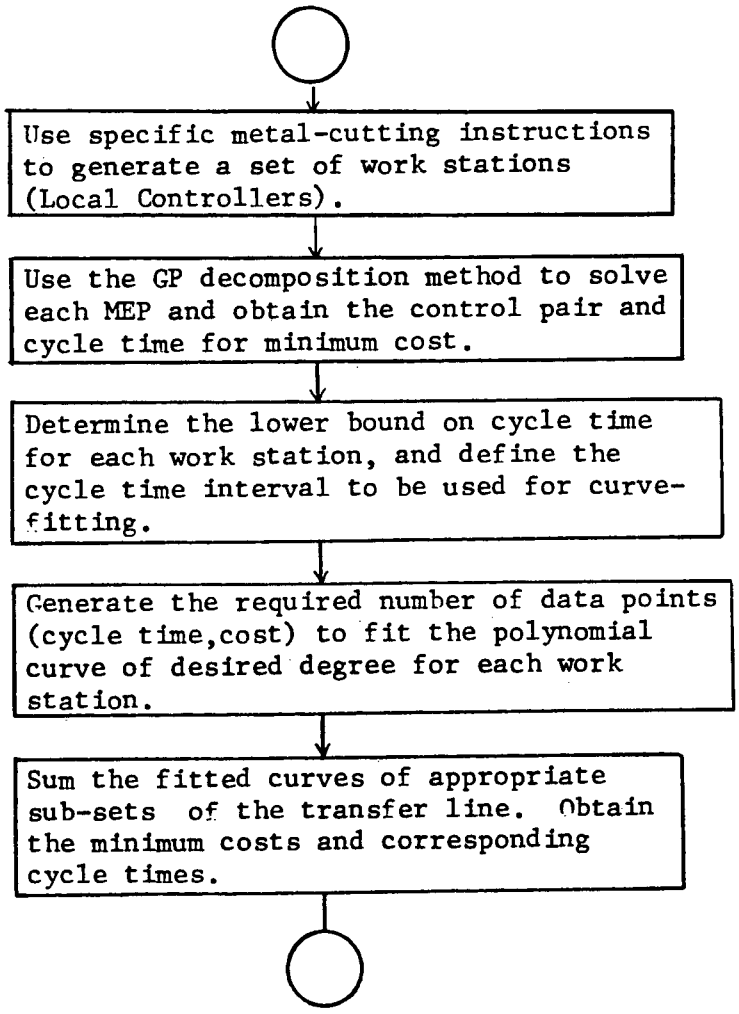


Figure B-3 Analysis Flow Chart for Class: Supervisory Controller

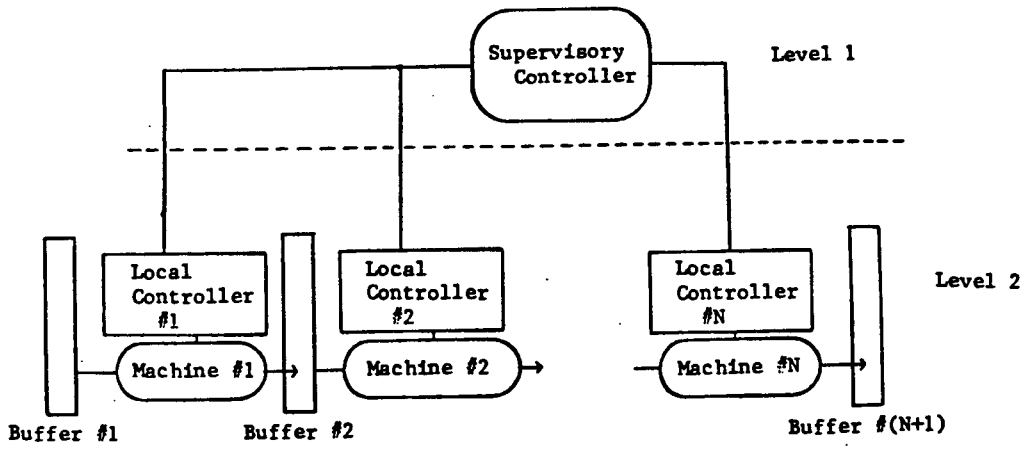


Figure B-4 Proposed Control Architecture

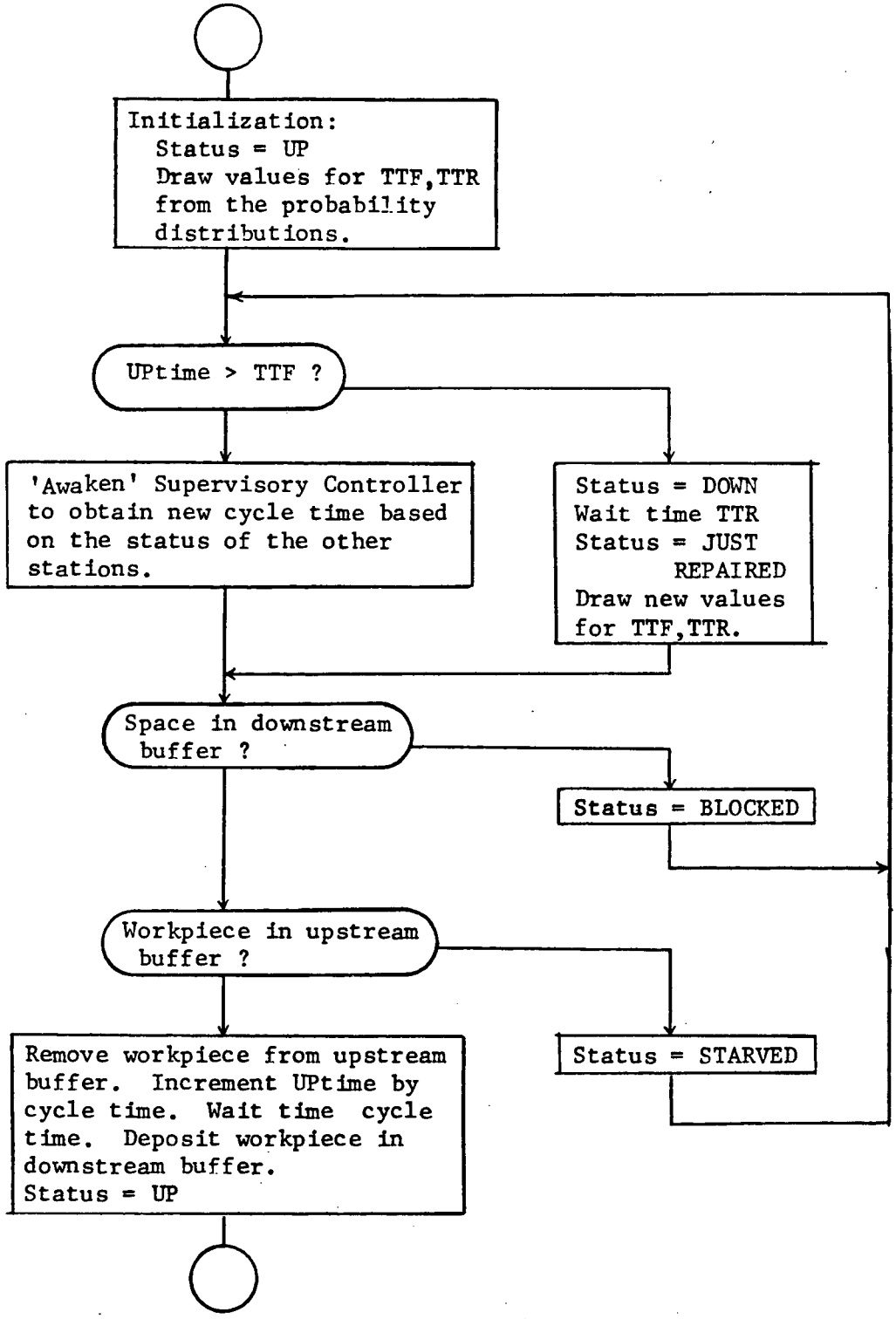


Figure B-5 Simulation Flow Chart for Class: Machine

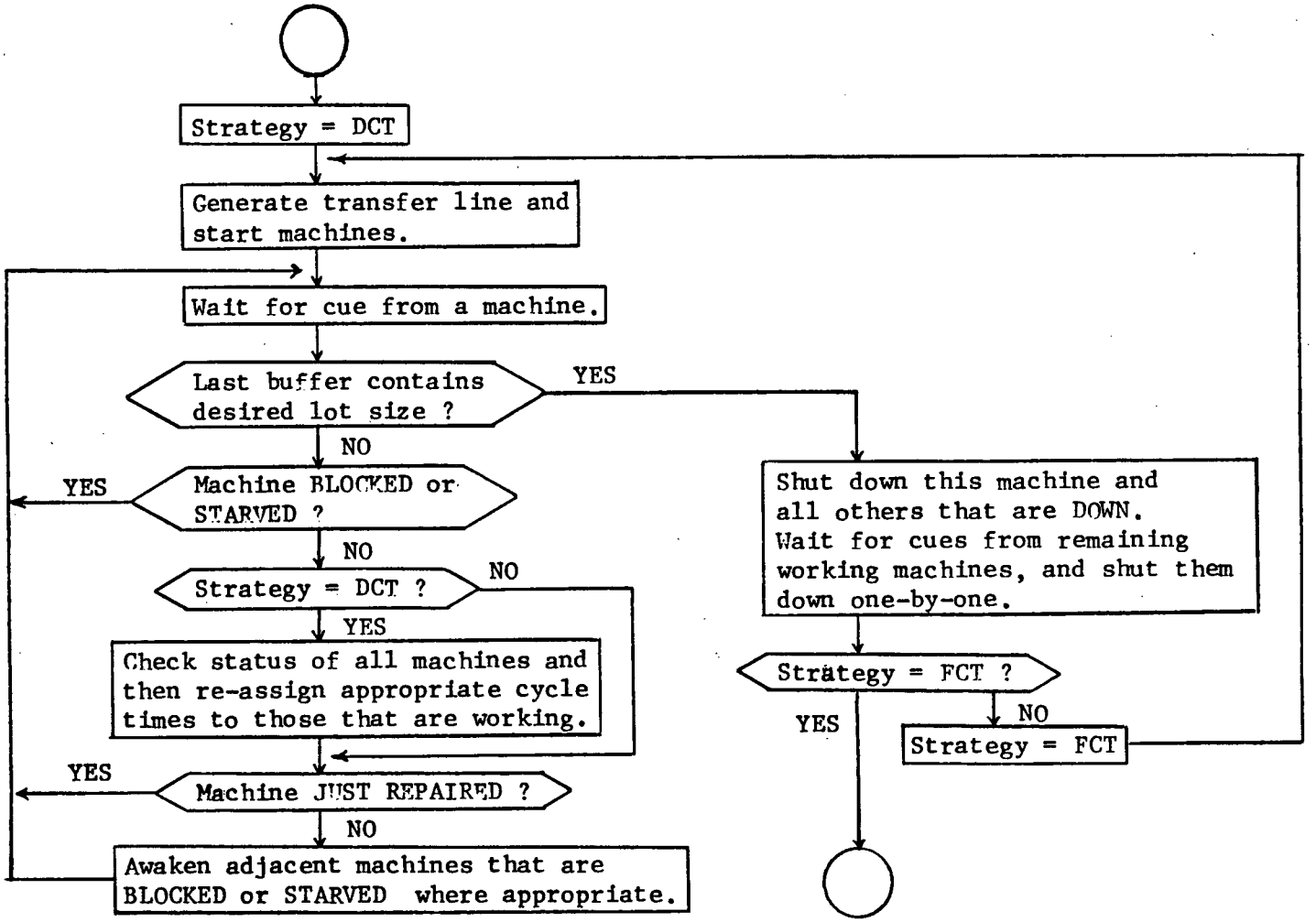


Figure B-6 Simulation Flow Chart for Class: Supervisory Controller

This permits a more efficient simulation, and also mimics the interrupt-driven communications scheme envisioned for the control hierarchy.

In Chapter 4, the length of a simulation trial was measured in terms of the number of failures expected F, which is related to MTTF. However, from a programming point of view, it was more convenient to use the number of workpieces N in the last buffer. If we associate one workpiece with one machine cycle, then

N (pieces) = MTTF (pieces/failure) x F (failures) (B.1)

To simplify the program further, the MTTF and MTTR values were converted from cycles to minutes using a 'mean' cycle time $\bar{t}$; $\bar{t}$ is the arithmetic average of the optimal cycle times for the sub-sets of the ATL. For example, for a two station ATL:

$$\bar{t} = 1/3 \left(t_{12}^{*} + t_{1}^{*} + t_{2}^{*} \right) \quad \text{(B.2)}$$

As a final note, the computer plots in this thesis were obtained using the MTS program package ALGRAF [44].

APPENDIX C: STATISTICAL ACCURACY IN SIMULATION

In this thesis, transfer line behavior was modelled as a stochastic process. Two of the measures used to characterize production were mean in-process buffer level and mean cost/workpiece. Because these are statistics of means, we must take care when the simulation is performed.

Statistics can either be obtained from one long trial, or from averaging many shorter trials. The main advantage of multiple trials is that statistical accuracy can be easily increased by adding more trials; in this way, the 'new' data is simply added to the 'old'. If we want to have just one trial, we must re-run the simulation over and over again with larger L's; this means that we generate all new data each time, which is wasteful. To make each trial statistically independent, we must use unique seeds for the random draws from the probability distributions. The student's t distribution can be used to determine the required length L of a trial and the number of trials N by using estimates for the mean and standard deviation of the random variable x [45,46]. Consider a sample of N trials, each of length L.

$\bar{x}$ = sample mean = $(\sum_i x_i)/N$
u = true mean
s = sample standard deviation = $\sqrt{(\sum_i (x_i-\bar{x})^2)/(N-1)}$

We can now use the following procedure to determine whether $\bar{x}$ is a reliable measure of u or not.

1. Select an acceptable confidence interval CI eg. 95%
2. Determine the t value t^* for (N-1) degrees of freedom from a set of statistical tables.
3. Compute $t = (\bar{x}-u)/(s/\sqrt{N})$ (C.1)
4. if $t>t^*$, then accept the hypothesis that $\bar{x}$ is an acceptable measure of u with certainty CI; otherwise, L or N or both must be increased.

When the true mean u is unknown, the numerator of (C.1) can be replaced by $k\bar{x}$, where k indicates how far away u can be from $\bar{x}$. For example, if k=0.10, then $t>t^*_{0.95}$ means that we are 95% certain that the true mean u is within 10% of the sample mean $\bar{x}$.

For the simulation work of Chapter 4,

CI = 90 N = 4
k = 0.10 L = 150 failures (on average)
$t^*_{0.90}(3) = 1.64$

APPENDIX D: THE DUAL FUNCTION FOR CONSTRAINTS g>=1

Let's start with the derivation of (2.20) for constraints of the traditional kind [17]. Consider the following problem:

$$\min g_0(x) = \sum_{i=1}^{m} u_i \quad \text{subject to} \quad g_1(x) = \sum_{i=m+1}^{p} u_i <= 1 \qquad (D.1)$$

The generalized Arithmetic Mean/Geometric Mean inequality states that if u and λ are vectors in $\mathbb{R}^n$ with u>0 and λ >=0, then

$$\sum_i u_i^{\lambda} >= \prod_i (c_i/\delta_i)^{\delta_i} \lambda^{\lambda} \qquad (D.2)$$

where $\lambda = \Sigma\delta_i$ and $A\delta = 0$ (orthogonality conditions). Now applying this to the objective function of (D.1), we obtain

$$\lambda_0 = \sum_{i=1}^{m} \delta_i = 1 \qquad \text{(normality condition)} \qquad (D.3)$$

$$\text{and} \quad g_0(x) = \sum_{i=1}^{m} u_i >= \prod_{i=1}^{m} (u_i/\delta_i)^{\delta_i} \qquad (D.4)$$

For the constraint g_1, we have

$$\lambda_1 = \sum_{i=m+1}^{p} \delta_i >= 0 \qquad \text{and} \qquad (D.5)$$

$$1 >= g_1^{\lambda_1} = \sum_i u_i^{\lambda_1} >= \prod_i (c_i/\delta_i)^{\delta_i} \lambda_1^{\lambda_1} = \prod_i (c_i \lambda_1/\delta_i)^{\delta_i} \qquad (D.6)$$

If we multiply the two inequalities (D.4) and (D.6), we obtain

$$g_0(x) >= \prod_{i=1}^{m} (c_i/\delta_i)^{\delta_i} \prod_{i=m+1}^{p} (c_i\lambda_1/\delta_i)^{\delta_i} \qquad (D.7)$$

which is just (2.20). Now replace g_1 in (D.1) by the following constraint:

$$g_2 = \sum_{i=m+1}^{p} u_i >= 1 \quad \text{with } \lambda_2 <= 0 \qquad (D.8)$$

Rather than dealing with g_2 directly, let us define

$$g_3 = g_2^{-1} \quad \text{with} \quad \lambda_3 = -\lambda_2 >= 0 \qquad (D.9)$$

Then using (D.2), we have

$$1 >= g_3^{\lambda_3} = (g_2^{-1})^{\lambda_3} = (g_2^{\lambda_3})^{-1} >= \prod_i (c_i\lambda_3/\delta_i)^{-\delta_i} \qquad (D.10)$$

Finally, we obtain (2.26) by multiplying (D.4) and (D.10).

REFERENCES

1. Pressman,R.,Williams,J., Numerical Control and Computer-Aided Manufacturing, Wiley, 1977.

2. Batra,J.,Barash,M.,"Automatic Computerized Optimization of Multi-spindle Drilling with Probabilistic Tool Life", Mac. Tool Des. Res., V.14, 1973.

3. Chang,T.,Wysk,R.,Davis,R.,Choi,B.,"Milling Parameter optimization through a Discrete Variable Transformation", Int. J. Prod. Res., V.20 #4, 1981.

4. Boer,C.,De Malherbe,M.,"Adaptive Control optimization for a Numerically Controller Milling Process", Mac. Tool Des. Res., V.18, 1977.

5. Mayne,R.,Malkin,S.,"Parameter Optimization of the Steel Grinding Process, Trans. ASME, J. Eng. Ind., August 1976.

6. Stelson,T.,Komanduri,R.,Shaw,M.,"Manual Adaptive Control of a Continuous Grinding Operation", Int. J. Prod. Res., V.17 #2, 1979.

7. Amitay,G.,Malkin,S.,Koren,Y.,"Adaptive Control Optimization of Grinding", Trans. ASME, J. Eng. Ind., February 1981.

8. Masory,O.,Koren,Y.,"Adaptive Control System for Turning", Annals of the CIRP, V.29 #1, 1980.

9. El-Karamany,Y.,Papai,F.,"Determination of Turning Machine Performance by Non-Linear Programming", Int. J. Mach. Tool Des. Res., V.18, 1978.

10. Wilson,G.,Wilkinson,J.,"Adaptive Control for a C.N.C. Lathe", Proc. 22nd Mach. Tool Des. Res. Conf., 1981.

11. Bedini,R.,Lisini,G.,Pinotti,P.,"Experiments on Adaptive Control of a Milling Machine", Trans. ASME, J. Eng. Ind, February 1976.

12. Lewis,A.,Nagpal,B.,"Economic Considerations in Adaptive Control", Proc. 22nd Mach. Tool Des. Res. Conf., 1981.

13. Amarego,E.,Brown,R., The Machining of Metals, Prentice-Hall, 1969.

14. Duffin.R.,Peterson,E.,Zener,C., Geometric Programming, Wiley, 1967.

15. Zener,C., Engineering Design by Geometric Programming, Wiley-Interscience, 1971.

16.Beightler,C.,Phillips,D., Applied Geometric Programming, Wiley, 1976.

17.Ecker,J.,"Geometric Programming: methods, computations, and applications", SIAM Review, V.22 #3, 1980.

18.Bohn,E.,"Optimum Design of Control System Compensators by Geometric Programming", Trans. ASME, J. Dyn. Sys. Meas. Cont., June 1980.

19.Bohn,E.,Private Correspondence,1982-1983.

20.Phillips,D.,Beightler,C.,"Optimization in Tool Engineering using Geometric Programming", AIIE Trans., V.2 #4, 1970.

21.Boothroyd,G., Fundamentals of Metal-Machining and Machine Tools, Scripta, 1975.

22.Walvekar,A.,Lambert,B.,"An Application of Geometric Programming to Machining Variable Selection", Int. J. Prod. Res., V.8 #3, 1970.

23.Petropoulos,P.,"Optimal Selection of Machining Rate Variables by Geometric Programming", Int. J. Prod. Res., V.11 #4, 1973.

24.Ermer,D.,"Optimization of the Constrained Machining Economics Problem by Geometric Programming", Trans. ASME, J. Eng. Ind., November 1971.

25.Ermer,D.,Kromdihardjo,S.,"Optimization of Multi-pass Turning with Constraints", Trans. ASME, J. Eng. Ind., November 1981.

26.Ohmi,T.,"An Approximation for the Production Efficiency of Automatic Transfer Lines with In-Process Storage", AIIE Trans., V.13 #1, 1981.

27.Yang,H.,Chen,K.,Chang,W.,Wang,Y.,"An Investigation on Reliablity and Optimum Buffer Stocks of Machine Tool, Transfer Lines and their Simulations on Digital Computer", Mach. Tool Des. Res., V.21, 1981.

28.Buzacott,"Automatic Transfer Lines with Buffer Stocks", Int. J. Prod. Res., V.5 #3, 1967.

29.Law,S.,"A Generalized Simulator for Automatic Transfer Line Systems", Annals of the CIRP, V.3 #1, 1982.

30.Masso,J.,Smith,M.,"Interstage Storages for Three Stage Lines subject to Stochastic Failures", AIIE Trans., V.6 #4, 1974.

31.Hatcher,J.,"The Effect of Internal Storage on the Production Rate of a Series of Stages having Exponential Service Times", AIIE Trans., V.1 #2, 1969.

32.Muth,E.,"The Production Rate of a Series of Work Stations with Variable Service Times", Int. J. Prod. Res., V.11 #2, 1973.

33.Canuto,E.,Villa,A.,Rossetto,S.,"Transfer Lines: a deterministic model for buffer capacity selection", Trans. ASME, J. Eng. Ind., May 1982.

34.Frost-Smith,E.,Marten,H.,Yeates,A.,"Optimizing Control of a Batch Machine Shop", Proc. IEE, V.118 #12, 1971.

35.Duffie,N.,Bollinger,J.,"Distributed Computing Systems for Multiple-Processor Industrial Control", Annals of the CIRP, V.29 #1, 1980.

36.Wadsworth,G.,Bryan,J., Introduction to Probability and Random Variables, McGraw-Hill, 1960.

37.Arndt,G.,"A Survey of Flexible Manufacturing System Layouts", Inst. of Eng. Aust., Mech. Eng. Trans., 1982.

38.Boothroyd,G.,Poli,C.,Murch,L., Automatic Assembly, Dekker, 1982.

39.SIMULA Documentation from the Norwegian Computing Center:
Programmer's Guide, IBM System 360, #S-23-0, 1973.
User's Guide, IBM System 360, #S-24-1, 1975.
360/370 External Procedure Library, #S-56, 1981.

40.Birtwistle,G., SIMULA Begin, Studentlitteratur, 1974 (Eng.Trans.).

41.UBC MATRIX, UBC Computing Centre, March 1982.

42.UBC CURVE, UBC Computing Centre, September 1981.

43.UBC ROOT, UBC Computing Centre, June 1980.

44.UBC GRAPHICS, UBC Computing Centre, April 1981.

45.Walpole,R.,Myers,R., Probability and Statistics for Engineers and Scientists, Macmillan, 1972.

46.Kleijnen,J.,"Design and Analysis of Simulations: practical statistical techniques", Tech. J. Soc. Comp. Sim., V.28 #3, 1977.