Division of Research
Graduate School of Business Administration
The University of Michigan

September 1981

LINEARIZED NONLINEAR MIP FORMULATIONS FOR LOADING A FLEXIBLE MANUFACTURING SYSTEM

Working Paper No. 278

Kathryn E. Stecke

The University of Michigan

FOR DISCUSSION PURPOSES ONLY

None of this material is to be quoted or
reproduced without the express permission
of the Division of Research.

ABSTRACT

This paper addresses production planning problems for flexible manufacturing systems (FMS's). Specifically, the grouping and loading problems are formulated as nonlinear 0-1 mixed integer programs. Several linearization methods are examined and applied to data from an existing FMS. Several problems are solved using the linearization that results in the fewest additional constraints and/or variables. We also discuss when each linearization is best and the use of the linearized models in the solution of actual planning problems.

Presented at the Joint ORSA/TIMS National meeting in Houston in October, 1981.

1. Introduction

This paper addresses the grouping and loading problems, two of a series of five production planning problems that are defined here for flexible manufacturing systems (FMS's). An FMS is an automated batch manufacturing system consisting of numerically controlled machines, linked by automated material handling, that perform the operations required to manufacture parts. Each operation requires one or more cutting tools. The tools for all operations that can be performed by a machine are stored in its limited capacity tool magazine. Each machine has an automatic tool interchanging device that can interchange two cutting tools in seconds. This rapid interchange capability allows several consecutive operations to be machined with virtually no set-up time between operations. One or more computers control most activity such as machining operations, part movements, and tool interchanges. The computer cannot route a particular part to a machine unless all tools required for the next operation were previously placed in the magazine. This last requirement indicates the need for planning prior to production. For additional information concerning FMS's, see Stecke and Solberg [1981a].

Managing production for an FMS is more difficult than for production lines and job shops because: each machine is quite versatile and capable of performing many different operations, the system can machine several part types simultaneously, and each part may have alternative routes through the system. These additional capabilities and planning options increase both the number of decision variables and constraints associated with setting up an FMS.

To best utilize an FMS's capabilities, a careful system set-up is required prior to production. Set-up decisions for batch manufacturing are made frequently to meet new or altered production requirements, for example, whenever production requirements for a part type are met. This contrasts with a mass production system where set-up is part of the design process and set-up changes are few.

The decision variables of the set-up problem of an FMS are: part types to be produced next, relative numbers of parts of each type to be machined simultaneously, number of pallets and fixtures of each fixture type to be reserved for each part type, and allocation of operations (and tools) to machines. The objective of a set-up is system dependent, but commonly is to maximize expected production.

Since the problem in its general form is intractable, the following approach is suggested. A set of five production planning problems are defined, the solutions to which are a system set-up. The problems can be solved sequentially, or alternatively, candidate solutions to the problems can be generated iteratively until a suitable final solution is found. In addition surrogate objectives are used for each problem rather than attempting to maximize production directly. The problems are:

1. Part type selection problem:

   From a set of part types that have production requirements, determine a subset for immediate and simultaneous processing.

2. Machine grouping problem:

   Partition the machines into machine groups in such a way that each machine in a particular group is able to perform the same set of operations.

3. Production ratio problem:

   Determine the relative ratios at which the part types selected in problem (1) will be produced.

4. Resource allocation problem:

   Allocate the limited number of pallets and fixtures among the selected part types.

5. Loading problem:

   Allocate the operations and required tools of the selected part types among the machine groups subject to technological and capacity constraints of the FMS.

For additional information concerning these problems, see Stecke [1981].

The plan of the paper is as follows. The machine grouping (2) and loading (5) problems are formulated in Section 2 as 0-1 nonlinear mixed integer programs (MIP's). In addition different methods of solving nonlinear MIP's are surveyed. Section 3 presents several linearization methods. In Section 4 these are applied to data from an existing FMS. Numerical solutions are obtained using the linearization resulting in the fewest additional constraints and/or variables. In addition conclusions are drawn concerning conditions under which each linearization is best. Computational experience, which shows advantages to considering the nonlinear terms, is discussed in Section 4. The paper concludes with a discussion in Section 5.

2. Mathematical Programming Formulations

After defining required notation, this section begins by developing the constraint formulations necessary for the grouping and loading problems. The grouping problem is then defined and formulated. Finally, several loading objectives are developed.

Parameters and Variables

The subscripts, input parameters, and decision variables are given in Table 1. Several subscript conventions and parameters require further explanation.

TABLE 1

Notation

Subscripts:

operation	$i=1,\ldots, b$
machine	$j=1,\ldots, m$
machine group	$\ell=1,\ldots, M$
machine type	$n=1,\ldots, \overline{m}$
set of operations	$k=1,\ldots, K$

Parameters (Input):

$p_{i\ell}$ = processing time of operation i on one of the machines in machine group ℓ;

q_i = maximum number of times that operation i can be assigned;

d_i = number of slots required in a tool magazine by operation i;

t_ℓ = capacity of the tool magazine for each machine in group ℓ;

w_{ik} = number of slots saved due to common tools when operations i and k are assigned to the same machine;

= count of the number of spaces (slots) occupied by the tools contained in the intersection of the sets of tools required by operations i and k;

B_k = index set of sets of operations;

w_{B_k} = number of slots saved when the operations in B_K are assigned to the same machine;

P = index set of compatible part types that are to be produced simultaneously on the system of machines;

a_i = production ratio (relative to the remaining part types P-{i}) at which part type i will be produced;

m_n = {j|machine j is of machine type n};

Decision Variables (Output):

M_ℓ = {j|machine j is in machine group ℓ};

$$x_{i\ell} = \begin{cases} 1, & \text{if operation i is assigned to each machine in group } \ell; \\ 0, & \text{otherwise.} \end{cases}$$

In the formulations, several parameters of Table 1 might range over either machines (j) or machine groups (ℓ). In those formulations for which there is only one machine in each group, the machine subscript j is used.

The differences between machine types and machine groups should be clarified. All machines of the same machine type are physically identical (i.e., same axes of motion, dimensions, horsepower, capabilities). Each set of machine types, m_n, might be partitioned into several machine groups. Machines that are identically tooled comprise a group and are said to be pooled. Hence each machine in a particular group will be able to perform the same operations.

Constraint Formulations

The constraints of the grouping and loading problems are now given.

First, each operation must be assigned to at least one machine of the machine type required by the operation. In addition there is a limited number of duplicate assignments allowed:

$$1 \leq \sum_{\ell=1}^{M} x_{i\ell} \leq q_i, \qquad i = 1,\ldots,b. \tag{1}$$

It is understood that $x_{ij}(x_{i\ell}) = 0$ if operation i cannot be performed by the machine type corresponding to machine (group) j or ℓ.

Second, the tool magazine capacity constraint which relates the number of tool slots required by the operations assigned to a machine (group) to the total number of slots contained in the machine (group)'s tool magazine, in its simplest form, is

$$\sum_{i=1}^{b} d_i \, x_{i\ell} \leq t_\ell, \qquad \ell = 1,\ldots,m.$$

Since only one tool can be used at a time, however, it is unnecessary to assign any tool more than once to the same machine. Also the actual number of slots used depends on the physical placement of the tools in the tool magazine. In the example shown in Figure 1, two three-slot tools placed side by side require only five slots rather than six. Another complicating consideration is that since larger tools are heavier, tool magazines must be weight balanced. In addition, several operations may require some of the same tools. Space in the tool magazine can be saved by eliminating tool duplication and considering overlap and weight balancing. These savings are measured by w_{B_k}.

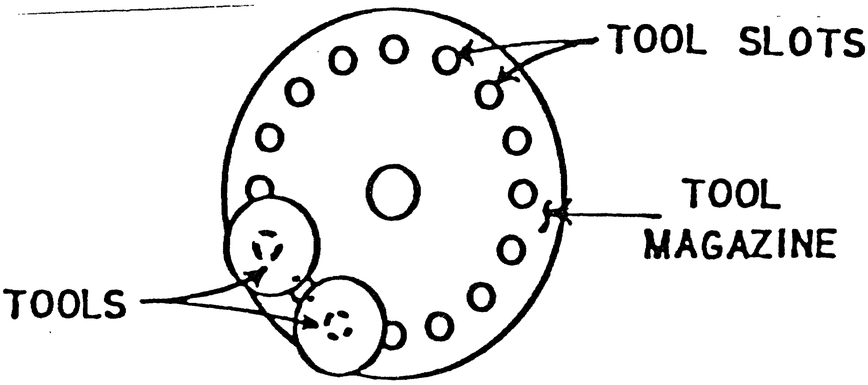

Figure 1. Tool Magazine.

The tool magazine capacity constraint then becomes:

$$\sum_{i=1}^{b} d_i x_{i\ell} - \sum_{i_1=1}^{b-1} \sum_{i_2=i_1+1}^{b} w_{i_1 i_2} x_{i_1 \ell} x_{i_2 \ell} +$$

$$+ \sum_{i_1=1}^{b-2} \sum_{i_2=i_1+1}^{b-1} \sum_{i_3=i_2+1}^{b} w_{i_1 i_2 i_3} x_{i_1 \ell} x_{i_2 \ell} x_{i_3 \ell} + \dots$$

$$+ (-1)^{p+1} \sum_{i_1=1}^{b-p+1} \sum_{i_2=i_1+1}^{b-p+2} \cdots \sum_{i_p=i_{p-1}+1}^{b} w_{i_1 i_2 \dots i_p} x_{i_1 \ell} x_{i_2 \ell} \dots x_{i_p \ell} \leq t_\ell,$$

or, in more compact form,

$$\sum_{i=1}^{b} d_i x_{i\ell} + \sum_{p=2}^{b} (-1)^{p+1} \sum_{\substack{\forall \bar{B} \subseteq B_k \\ |\bar{B}|=p}} w_{\bar{B}} \prod_{i_k \in \bar{B}} x_{i_k \ell} \leq t_\ell, \quad \ell=1,\dots, M. \tag{2}$$

Finally there is the integrality constraint:

$$x_{i\ell} = 0 \text{ or } 1, \text{ for all } i,\ell. \tag{3}$$

Machine Grouping Formulation

Pooling (see Kleinrock [1976], Stecke and Solberg [1981a]) increases system performance by decreasing the probability that a part is blocked by having no machine available for the next operation. Having more than one machine in a group is one way to allow alternate routes for some part types.

Stecke [1981] considered the best partitions of m items (servers, machines) into M (machine) groups to maximize expected production using a closed queueing network model. In particular, the results include:

i. Fewer groups are better, i.e., pool as much as possible;

ii. The maximum expected production is obtained from systems with the most unequal sized groups. More generally, all possible partitions are ordered according to expected production.

These results are used here.

This paper considers a more detailed treatment of the grouping problem than that of Stecke [1981] by adding constraints on tool requirements and tool magazine capacity that use actual operation times.

Maximum pooling of all machines of the same type into one group may not be possible because of some technological constraints such as tool magazine capacity. The approach we take to maximize pooling is as follows:

1. Set M equal to the minimum number of machines (or, machine groups)

required to perform all operations of the part types in P.

2. Use the optimal pooling for M groups from Stecke [1981].

Since all possible partitions of machines are ordered in Stecke [1981], the solution to the more detailed model is then the best feasible partition.

Step 1, which determines M, is now described. The problem is to find $\{x_{ij} | i=1,\ldots,b,\ j=1,\ldots,m\}$ to

$$\text{maximize} \quad \sum_{j=1}^{m} \gamma^{j} s\ell_{j}$$

subject to

$$s\ell_{j} = t_{j} - \Big(\sum_{i=1}^{b} d_{i} x_{ij} + \sum_{p=2}^{b} (-1)^{p+1} \sum_{\substack{\forall \bar{B} \subseteq B_K \\ |\bar{B}| = p}} w_{\bar{B}} \prod_{i_k \in \bar{B}} x_{i_k j} \Big),$$

and equations (1), (3), and $q_i = 1$,

where $\gamma = \sum_{i=1}^{b} d_i$. Note that $s\ell_j$ is the slack in the tool magazine capacity constraint of machine j.

Let m_{on} be the maximum number of machines of machine type n required to perform the operations of the part types in P, if common tooling is not taken into consideration, n = 1, ..., $\bar{m}$. The m_{on} are obtained by adding the number of tool slots required of machine type n, dividing by the capacity of each machine's tool magazine, t_n, and rounding to the next highest integer. Then an upper bound on the total number of required machines is m_o machines, where

$$m_o = \sum_{n=1}^{\bar{m}} m_{on}.$$

The objective function formulation allows for the values of the slack ($s\ell_j$) in the $|m_n|$ machines to monotonically increase. For every machine type, the initial machines will be filled first; if there is insufficient tool slot capacity and another machine is required, machine i will tend to be filled before an operation is assigned to machine j, for i<j. The result is the minimum number of machines of each type that are needed to perform the required operations.

An example will demonstrate the problem. Consider a 15-machine system of four machine types with $m_{o1}=4$, $m_{o2}=3$, $m_{o3}=4$, and $m_{o4}=3$. Then 14 machines are required if overlap is not considered. The machines, j, and their machine types, n, are as follows:

n:	1	2	3	4
machine j:	(1 2 3 4)	(5 6 7)	(8 9 10 11 12)	(13 14 15)

Suppose that the solution to Step 1 was that M=10, and that 3 machines were required of the first 3 types, and only one of the fourth type. Then the optimal pooling into machine groups according to Step 2 is:

machine j: (1 2) (3) (4) (5) (6) (7) (8 9 10) (11) (12) (13 14 15)

Notice that: all machines of the fourth type could be pooled, none of the second type could be, and there are 10 machine groups.

Loading Formulations

A usual loading procedure for both conventional systems and FMS's attempts to balance the assigned workload on each machine; the aim is to equalize the total weighted processing time, or workload, of the operations assigned to each machine. The processing time of each operation is weighted by the production ratio (a_i) of the corresponding part type i as calculated in the fourth production planning problem. In addition each operation is often assigned to only one machine. The consequence is that each part type has a fixed route through the shop.

However, the flexibility and capabilities of an FMS indicate that perhaps new planning and control procedures should be developed for FMS's, which could also perhaps be applicable to other types of manufacturing systems. In a previous study (Stecke [1977], Stecke and Solberg [1981b]), alternative loading objectives are defined. Application of any of several objectives can result in better system performance than that of a balanced assigned workload.

Stecke [1981] has shown that the practice of balancing is too restrictive for most FMS's, since the inherent flexibility can often be utilized for better system performance. Several alternative loading objectives are listed in Table 2. The decision concerning which to apply is problem dependent. Each may be best under certain circumstances. Some are contradictory in some situations; for other systems, several objectives may apply.

TABLE 2

Loading Objectives

1. Balance the assigned machine processing times;
2. Minimize the number of movements from machine to machine, or equivalently, maximize the number of consecutive operations on each machine;
3. Balance the workload per machine for a system of groups of pooled machines of equal sizes;
4. Unbalance the workload per machine for a system of groups of pooled machines of unequal sizes;
5. Fill the tool magazines as densely as possible;
6. Maximize the sum of operation priorities.

Consider the first objective function. Let r_j be the relative workload assigned to machine j:

$$r_j = \sum_{i=1}^{b} a_i p_{ij} x_{ij}, \qquad j=1,\ldots,m. \tag{5}$$

Then r_j is also a measure of the relative utilization of machine j. If all of the r_j are equal, the system is perfectly balanced. This is usually not possible because of the discrete values of processing times. The following formulations optimally balance the workloads assigned to machines as much as possible. Each objective function is a measure of system imbalance.

The problem is to find $\{x_{ij} | \ i=1,\ldots,b,\ j=1,\ldots,m\}$ to minimize $h_i(x)$, where $h_i(x)$ is one of the following four:

1. $\displaystyle \underset{\substack{j=1,\ldots,\ m-1 \\ h=1,\ldots,\ m;}}{\text{maximum}} |r_j - r_h|$

2. $\displaystyle \sum_{j=1}^{m-1} \sum_{h=j+1}^{m} |r_j - r_h|^{\gamma}, \quad \gamma>0$

3. $\displaystyle \sum_{j=1}^{m-1} \sum_{h=j+1}^{m} (r_j - r_h)^2$

4. $\beta-\alpha$

   subject to $0 \le \alpha \le r_j \le \beta, \qquad j=1,\ldots,m.$

The constraints are: (1), (2), (3), (5), and $q_i = 1$.

The second objective, minimize the number of movements, is quite different from the first. The second objective is relevant, for example, when transportation time or distance from machine to machine is large relative to average operation time. There are manufacturing systems for which minimizing movements from machine to machine is best even at the expense of balancing

(Stecke and Solberg [1981b]). It can be more advantageous for a part type to remain on a machine for several consecutive operations rather than to move for the sake of balancing. Furthermore when several consecutive operations require the same machine type, time may be saved by processing all of them on the same machine, if this is technologically possible. Both travel time (from machine to machine) and waiting time (for a possibly busy next machine to become idle) may be saved.

The first two objectives of Table 2 are often incompatible. When allocating operations in large sets, the potential for balance decreases: if the operation times to allocate are smaller, a better balance is likely.

We now formulate the second loading objective. Notice that if i and i+1 represent consecutive operations, then

$$x_{ij}-x_{i+1,j} = \begin{cases} 0, & \text{if operations i and i+1 are on the same machine j;} \\ \pm 1, & \text{if operations i and i+1 are assigned to different machines.} \end{cases}$$

If N is defined as twice the number of excess movements, then

$$N = \sum_{i=1}^{b-1} \sum_{j=1}^{m} |x_{ij}-x_{i+1,j}| = \sum_{i=1}^{b-1} \sum_{j=1}^{m} (x_{ij}-x_{i+1,j})^2 .$$

Some of the differences $(x_{ij}-x_{i+1,j})$ need not be included in the calculation of N. For example for some machine j, operation i may require a machine type other than that of machine j; in this case, $x_{ij} = 0$. In particular, if $(x_{ij} - x_{i+1,j}) = x_{ij}$ or $-x_{i+1,j}$, then this term may be excluded from N. Inclusion is not incorrect, merely unnecessary and inefficient. The second objective, then, is to

$$\text{minimize} \quad \sum_{i=1}^{b-1} \sum_{j=1}^{m} (x_{ij}-x_{i+1,j})^2 \text{ (or)}$$

$$= \sum_{i=1}^{b-1} \sum_{j=1}^{m} |x_{ij} - x_{i+1,j}|,$$

subject to (1), (2), (3), and $q_i = 1$.

The advantages from utilizing flexibility by allowing pooling and hence alternative part routes provide motivation for the remaining loading objectives. The third (and fourth) objectives (un)balance the workload per machine for a system of groups of pooled machines. The applicability of either depends on the configuration of the manufacturing system, or how the machines are grouped (see Stecke [1981]). For the third objective, the problem is to minimize $h_i(x)$, where $h_i(x)$ is one of the following four:

1. $$\underset{\substack{\ell=1,\ldots,\ M-1\\ k=\ell+1,\ldots,\ M}}{\text{maximum}} \left| \frac{r_\ell}{s_\ell} - \frac{r_k}{s_k} \right|$$

2. $$\sum_{\ell=1}^{M-1} \sum_{k=\ell+1}^{M} \left| \frac{r_\ell}{s_\ell} - \frac{r_k}{s_k} \right|^\gamma \quad , \quad \gamma > 0$$

3. $$\sum_{\ell=1}^{M-1} \sum_{k=\ell+1}^{M} \left(\frac{r_\ell}{s_\ell} - \frac{r_k}{s_k} \right)^2$$

4. $\beta - \alpha$

   subject to $0 \leq \alpha \leq r_\ell/s_\ell \leq \beta$, $\quad \ell=1,\ldots,M,$

subject to (1), (2), (3), (5), and $q_i = 1$.

Notice that r_ℓ/s_ℓ is the workload per machine in machine group ℓ.

The fourth problem is to minimize $g_i(x)$, where $g_i(x)$ is one of the following four:

1. $$\underset{\ell=1,\ldots,\ M}{\text{maximum}} \ |r_\ell - X_\ell|$$

2. $$\underset{\ell=1,\ldots,\ M}{\text{maximum}} \ |r_\ell - X_\ell|^\gamma, \quad \gamma > 0$$

3. $\sum_{\ell=1}^{M} (r_\ell - X_\ell)^2$

4. $\beta - \alpha$

$$\text{subject to } 0 \leq \alpha \leq r_\ell - X_\ell \leq \beta, \qquad \ell = 1, \ldots, M,$$

subject to (1), (2), (3), (5), and $q_i = 1$, and where X_ℓ is the theoretical optimal workload that should be assigned to machine group ℓ to maximize expected production (see Stecke and Solberg [1981b]).

The rationale for the fifth objective of Table 2 is that when tool magazines are filled, perhaps several operation assignments are duplicated to result in alternative part routes, which should increase machine utilization and production, and decrease waiting time. No single tool should be assigned to any particular machine more then once. In addition the maximum number of times that an operation could be duplicated can be specified. One formulation of this objective minimizes slack in the capacity constraints for all machines. Then the problem is to

$$\text{minimize } \sum_{j=1}^{m} s\ell_j$$

subject to (1), (3), $q_i \geq 1$, and

$$s\ell_j = t_j - \Big(\sum_{i=1}^{b} d_i x_{ij} + \sum_{p=2}^{b} (-1)^{p+1} \sum_{\substack{\forall \bar{B} \subseteq B_k \\ |\bar{B}| = p}} w_{\bar{B}} \prod_{i_k \in \bar{B}} x_{i_k j} \Big).$$

The aim of the sixth objective in Table 2 is similar to the aim of the fifth: to duplicate assignments of some operations. Operation assignments should not be duplicated arbitrarily. Some operations are more critical than other operations, such as bottleneck operations. In such cases, weights could be assigned to prejudice operation assignments. If w_i is the weight assigned to operation i, then the problem is to

$$\text{maximize} \sum_{i=1}^{b} \sum_{j=1}^{m} w_i x_{ij}$$

subject to (1), (2), (3), and $q_i \geq 1$.

Almost all of the objective functions and some of the constraints are nonlinear. There is a variety of approaches to solve nonlinear MIP's. They can be solved directly (Cooper [1981], Ginsburgh and Van Peetersen [1969], Hammer [1969], Hansen [1971,1979], Marsten and Morin [1978], Morin [1978], Lawler and Bell [1967]). Alternatively, piecewise linear approximations (Dantzig [1962], Hu [1969], Watters [1967]), or heuristic algorithms (Stecke and Solberg [1981a]) can be used. Another approximate approach is to ignore some tooling considerations that result in the nonlinear terms in the capacity constraints. Ignoring these factors results in feasible, but worse solutions. An exact approach is to linearize the nonlinear terms (Balas [1964], Glover and Woolsey [1973, 1974], Glover [1975]). The resultant linearized 0-1 problems could either be solved directly, or relaxed. The applicability of a direct nonlinear approach versus a transformed linear approach is problem-dependent (Taha [1970]). The direct approach is more difficult while linearization results in much larger problems.

In the following section, we present several methods to linearize the nonlinear terms, and apply combinations of these methods to data in Section 4.

3. Linearization of the Product Terms

Fortunately the nonlinear terms in the formulations are products of 0-1 integer variables, for which there are several methods to linearize the terms.

The standard procedure is to replace each cross product term with a new variable. Additional constraints are required to insure that the new variables take on the correct values.

In this section we survey five linearization methods, which differ in both the numbers of additional variables (either integer or continuous) and constraints generated. The difficulty of integer problems depends primarily on the number of integer, rather than continuous, variables. Additional details concerning the linearizations of the formulations in Section 2, in particular, the generated variables and constraints, can be found in Stecke [1981].

The first method was developed by Balas [1964]. Each product term $\prod_{i_k \in \bar{S}} x_{i_k j}$ is replaced by a new variable $x_{\bar{S}}$

$$\sum_{i_k \in \bar{S}} x_{i_k j} - x_{\bar{S}} \leq p-1, \qquad p = |\bar{S}| \tag{6}$$

$$\sum_{i_k \in \bar{S}} x_{i_k j} + p\, x_{\bar{S}} \leq 0, \qquad j=1, \ldots, m \tag{7}$$

For each product term, there are two new constraints and one new integer variable.

The second method, described in Glover [1975] for quadratic terms, can be used for higher order terms by recursive application (Stecke [1981]). For m x m quadratic terms, Balas's approach (method 1) introduces m(m-1)/2 new integer variables (one for each cross product term) and m(m-1) additional constraints. Glover's approach adds 4m constraints and m continuous variables, which are automatically 0-1 without requiring an integer restriction. The second method has the advantage that the transformed linear integer program has the same number of integer variables as the original nonlinear program.

The third method (Glover and Woolsey [1974]) allows the new variables $x_{\bar{S}}$ (of equations (6) and (7)) to be continuous by replacing the second inequality of Balas' method (equation (7)) with $|\bar{S}|$ additional constraints. Despite the additional constraints method 3 can be better than the first method since the additional constraints are simpler and the new variables are continuous.

The fourth method (Glover and Woolsey [1974]) also allows $x_{\bar{S}}$ to be continuous by replacing several of the constraints (7) that contain terms with common variables, by a single constraint. If there are no variables in common, there is no reduction.

The final method (Glover and Woolsey [1973]) consists of a series of three rules that reduces the number of required constraints of the first of the pair of constraints (equation (6)) generated by Balas's method. The new variables are continuous.

4. Application and Computational Results

Combinations of the five linearizing methods are applied to data from the Sundstrand DNC (Direct Numerical Control) line at the Caterpillar Tractor Company in Peoria, Illinois. The Caterpillar FMS consists of nine metal-cutting machines plus an inspection station. This set of machines includes four 5-axis Omnimills, three 4-axis Omnidrills, and two vertical turrent lathes (VTL's). In addition there are two automatically controlled transporters, which provide in-process material handling and also deliver parts to the inspection station. The 16-station load/unload area is located midway along the line's length. These stations also provide some in-process inventory. A remotely located DEC PDP 11/20 computer and supporting equipment control the entire system on a real-time basis.

The parts machined on this line are two sizes of housings for automatic transmissions. Each type of housing is composed of two parts, a transmission case and a cover. The parts arrive at the facility in rough casting form and leave as an assembled matched pair. There are three part types: transmission cases, covers, and assemblies.

Caterpillar's loading objective was to balance the assigned workload per machine as much as possible; in addition each operation was assigned to only one machine.

The machines have different capabilities. Some operations require a mill; others can be performed on either a mill or a drill (mills can do drilling operations, but not vice versa). The two VTL's could be pooled although they were not in management's original set-up strategies. After certain operations, a proportion of the parts will visit the inspection station. For additional information concerning the management and control of the DNC line, see Stecke [1977].

4.1 Input

Some operations are collected in advance into operation sets. For example, a large case requires 49 operations (Stecke [1977]), which Caterpillar had aggregated into nine operation sets. These operation sets, along with those of covers and assemblies, are allocated among machines according to various loading objectives.

The input data includes for each operation set: the machine type required, the total number of tool slots required, the tool number and number of slots for any tool of that operation set which is required by at least one other operation set, and the processing time.

Initial calculations include a table of the number of tool slots saved (w_{B_k}). This table, as well as the constraint formulations, are found in Stecke and Solberg [1981a].

4.2 Constraints Linearized and Compared

The tool magazine capacity constraint is formulated, and then linearized, according to the different methods. The basic nonlinear formulation consists of 48 integer variables and 25 constraints. See Table 3, which also contains the number of additional (continuous) integer variables and constraints that are generated by each of six combinations of the five linearizing methods. Application of each of the first two methods results in different constraint linearizations. Methods 3 and 4 replace the second of each pair of constraints generated by method 1, while the fifth method replaces the first constraint of each pair of method 1.

The best combination of methods, with respect to fewest additional variables and/or additional constraints, was to be chosen to run on a CDC 6600 in conjunction with each loading objective.

TABLE 3

Number of Variables and Constraints Generated by Linearizing

			Linearization Methods					
	Nonlinear Formulations		1	2	3	4	5	4,5
Variables Integer (Continuous)	48	+	113	(76)	(113)	(113)	113	(113)
Constraints	25	+	218	228	373	157	152	91

The variables in parentheses are automatically 0-1 when the original variables are constrained to be so. Then the two sets of linearized constraints that are candidates for selection to use for solving the MIP's are method 2, and methods 4 and 5. Method 2 generates fewer variables, methods 4 and 5 fewer constraints. Since the difference in the number of constraints is large, the constraint set chosen to run the MIP's is that generated by the fourth and fifth methods.

4.3 Comparison of Linearizations

Each linearization method is applicable to different types of nonlinear problems. If the product of nonlinear 0-1 variables is quadratic, then the second method (Glover [1975]) may be best. Since higher order product terms are present for our problem, the second linearizing procedure must be applied iteratively to result in an increasing number of generated constraints. In similar situations, methods 4 and 5 would usually be better than method 2.

4.4 Objective Functions Linearized

The first objective function formulated is the grouping objective, which maximizes pooling. To accomplish this, the number of machines required is minimized. The remaining machines can then be pooled as indicated in Stecke and Solberg [1981a].

The two machine types, n, are mills and drills. Let $m_{oM}(m_{oD})$ be the upper bound on the number of mills (drills) required when overlap is not considered. The m_{on} are obtained by rounding to the next highest integer, the ratio of the number of slots of all operations that require machine type n (M or D) divided by t_n. Then

$$m_{oM} = \lceil 169/60 \rceil = \lceil 2.8 \rceil = 3 \text{ mills}$$

$$m_{oD} = \lceil 68/60 \rceil = \lceil 1.113 \rceil = 2 \text{ drills},$$

where $\lceil x \rceil$ denotes the least integer greater than or equal to x.

The grouping objective maximizes a linear combination of the slack variables to find the minimum number of required machines of each type. There are no additional constraints and variables that have not already been linearized for the capacity constraints in Section 4.2.

The second objective formulated minimizes movements. The number of additional constraints required by an application of the second method is 48, and there are 16 new continuous variables. The first method introduces 16 integer variables and 32 constraints. However, several of the variables have previously been defined and linearized for the capacity constraints. Hence the first method introduces only 9 new integer variables and 16 new constraints. This information is summarized in Table 4. The constraint set cannot be reduced further by methods 4 or 5 because no two constraints contain any common variables.

TABLE 4

Objective Function Linearizations

		Nonlinear		Linearization Methods		
		Formulations		1	2	
Minimize Movements	Variables	-	+	9	(16)	Constraint set cannot be reduced further
	Constraints	-	+	16	48	
Balancing	Variables	(7)	+	(12)	-	
	Constraints	7	+	29	-	

In general, the second method can require additional constraints using variables that previously have been linearized. The other methods do not require additional constraints for these variables.

Finally, note that a composite objective can be defined that minimizes both movements and the number of required machines. This is achieved using a linear combination of the two objectives.

The next loading objective that is linearized balances the assigned machine processing times. Some inequalities were used to reduce the number of new variables and constraints. Summarized in Table 4, the basic formulation introduces 7 continuous variables (the r_j) and 7 constraints. Linearization by method 1 adds 12 continuous variables and 29 constraints.

Formulations of the third (and fourth) loading objectives, (un)balancing, are similar to that of the first objective. In addition the resultant formulations are smaller than the first. Since machines are partitioned into groups and assignments are identical for each machine in a group, a capacity constraint is required only for each group rather than for each machine. The resultant smaller formulations are not linearized and solved here.

4.5 Problem Sizes

A summary of the sizes of the linearized MIP formulations of the grouping problem, and the two representations of the loading problem, is given in Table 5.

TABLE 5

Problem Sizes

	OBJECTIVES		
	Maximize Pooling	Minimize Movements	Balancing
Variables	48 + (113) = 161	57 + (113) = 170	48 + (132) = 180
Constraints	116	132	152

The computer code used to solve the three linearized mixed integer programming problems, MIPZ1, is described in McCarl, Barton and Schrage [1973]. Solution times ranged from about 1.5 to 2.5 minutes on a CDC 6600. The code is an adaptation of the code developed by Bravo, et al. [1970] and requires integer variables to be either zero or one. The algorithm is a modification of Balas's Additive Algorithm [1965] along the lines suggested by Glover [1968] and Salkin [1970]. Details are described in McCarl, et al. [1973].

4.6 Solutions

The nonlinear magazine capacity constraints result in larger linear MIP's, but also in better solutions. For the example, the solution to the grouping problem was that all three drills could be pooled. If overlap were ignored, the solution is that two drills are needed to hold all required tools. Consideration of tool duplication also allowed more pooling of mills than otherwise.

The optimal solutions of the three MIP's are identical to those in Stecke and Solberg [1981b], which were obtained by heuristic means according to the same balancing/minimizing movements/pooling objectives.

5. Discussion

The detailed nonlinear MIP formulations of the grouping and loading problems discussed in this paper provide optimal solutions that are feasible in actual applications. Linearizing methods have been suggested that are appropriate for different problems. For common problem sizes, the linearized MIP's can be solved. For larger systems, additional research needs to be done. Perhaps heuristic procedures, piecewise linear approximations, or linear relaxations of the MIP's could be used if optimal loadings are not required.[1]

[1] The author would like to thank Bruce W. Schmeiser and James J. Solberg of Purdue University for their helpful comments during the preparation of this manuscript. The research was supported in part by the National Science Foundation Grant No. APR 74 15256.

REFERENCES

1. BALAS, EGON, "Extension de l'algorithme additif à la programmation en nombres entiers et à la programmation non linéaire," *C.R. Acad. Sc. Paris* (May 1964).

2. BALAS, EGON, "An Additive Algorithm for Solving Linear Programs with Zero-One Variables," *Operations Research*, Vol. 13 (1965), pp. 517-545.

3. BRAVO, A., GOMEZ, J.G., LUSTOSA, L., SCHRAGE, L. and PIZZOLATO, N.D., *A Mixed Integer Programming Code*, Report 7043, University of Chicago, Chicago IL, September 1970.

4. COOPER, MARY W., "A Survey of Methods for Pure Nonlinear Integer Programming," *Management Science*, Vol. 27 (1981), pp. 353-361.

5. DANTZIG, GEORGE, B., *Linear Programming and Extensions*, Princeton University Press, Princeton NJ, 1962.

6. GINSBURGH, V. and VAN PEETERSEN, A., "Un algorithme de programmation quadratique en variables binares," *Réview Françoise d'Informatique et de Recherche Operationnelle*, Vol. 3 (1969), pp. 57-64.

7. GLOVER, FRED, "Surrogate Constraints," *Operations Research*, Vol. 16 (1968), pp. 741-749.

8. GLOVER, FRED, "Improved Linear Integer Programming Formulations of Non-Linear Integer Problems," *Management Science*, Vol. 22 (1975), pp. 455-460.

9. GLOVER, FRED, "Heuristics for Integer Programming Using Surrogate Constraints," *Decision Sciences*, Vol. 8 (1977), pp. 156-166.

10. GLOVER, FRED and WOOLSEY, EUGENE, "Further Reduction of Zero-One Polynomial Programming Problems to Zero-One Linear Programming Problems," Operations Research, Vol. 21 (1973), pp. 156-161.

11. GLOVER, FRED and WOOLSEY, R.E., "Converting the 0-1 Polynomial Programming Problem to a 0-1 Linear Program," Operations Research, Vol. 22 (1974), pp. 180-182.

12. HAMMER, PETER L., "A B-B-B Method for Linear and Nonlinear Bivalent Programming," Operations Research, Statistics and Economics Mimeograph Series No. 48, Technion, May 1969.

13. HANSEN, P., "Nonlinear 0-1 Programming by Implicit Enumeration," paper presented at the 7th Mathematical Programming Symposium, The Hague, September 1971.

14. HANSEN, P., "Methods of Nonlinear 0-1 Programming," Annals of Discrete Mathematics, Vol. 5 (1979), pp. 53-70.

15. HU, T.C., Integer Programming and Network Flows, Addison-Wesley, Reading, Massachusetts, 1969.

16. LAWLER, EUGENE L. and BELL, M.D., "A Method for Solving Discrete Optimization Problems," Operations Research, Vol. 15 (1967), pp. 1098-1112.

17. MARSTEN, ROY, E. and MORIN, THOMAS L., "A Hybrid Approach to Discrete Mathematical Programming," Mathematical Programming, Vol. 14 (1978), pp. 21-40.

18. McCARL, BRUCE, BARTON, DAVID and SCHRAGE, LINUS, MIPZ1 -- Documentation on a Zero-One Mixed Integer Programming Code, Dept. of Agricultural Economics, Purdue University, W. Lafayette IN, September 1973.

19. MORIN, THOMAS L., "Computational Advances in Dynamic Programming," Dynamic Programming and Its Applications, Martin L. Puterman, (ed.), Academic Press, New York, 1978.

20. SALKIN, HARVEY M., "On the Merit of Generalized Origin and Restarts in Implicit Enumeration," Operations Research, Vol. 18 (1970), pp. 549-555.

21. STECKE, KATHRYN E., "Experimental Investigation of a Computerized Manufacturing System," Master's Thesis, Purdue University, W. Lafayette, IN 47907, December 1977.

22. STECKE, KATHRYN E., "Production Planning Problems for Flexible Manufacturing Systems," Ph.D. Dissertation, Purdue University, W. Lafayette, IN 47907, August 1981.

23. STECKE, KATHRYN E. and SOLBERG, JAMES J., "The CMS Loading Problem," Report No. 20, NSF GRANT No. APR74 15256, School of Industrial Engineering, Purdue University, W. Lafayette, IN 47907, February 1981a.

24. STECKE, KATHRYN E. and SOLBERG, JAMES J., "Loading and Control Policies for a Flexible Manufacturing System," International Journal of Production Research, forthcoming in 1981b.

25. TAHA, HAMDY A., "A Balasian-Based Algorithm for 0-1 Polynomial Programming," Research Report No. 70-2, University of Arkansas, May 1970.

26. WATTERS, LAWRENCE J., "Reduction of Integer Polynomial Programming Problems to Zero-One Linear Programming Problems," Operations Research, Vol. 15 (1967), pp. 1171-1174.