Structural Eng./Earthquake Eng., JSCE, Vol. 13, No. 2, 105s—122s, 1996 October
(J. Struct. Mech. Earthquake Eng., JSCE, No. 549/I—37)

REVIEW

DISCRETE STRUCTURAL OPTIMIZATION WITH COMMERCIALLY AVAILABLE SECTIONS

Jasbir S. ARORA and Min-Wei HUANG

OPTIMAL DESIGN LABORATORY, College of Engineering, The University of Iowa
(Iowa City, Iowa 52242 U.S.A.)

Methods for mixed discrete-integer-continuous variable nonlinear optimization are reviewed for structural design applications with focus on problems having linked discrete variables. When a discrete value for such a variable is specified from an allowable set, the values for other variables linked to it must also be used in all the calculations. Optimum design of steel frames using commercially available sections is an example of this class of problems. A general formulation for this type of problems is developed. Approaches for solving such practical optimization problems are described and classified into single and multiple design variable formulations. Many approaches use two phases in their solution process before the final discrete design is obtained : In the first phase, a continuous variable optimum is usually obtained, and in the second phase, the continuous solution is somehow utilized to obtain the final discrete solution. Some of the basic optimization methods used in these approaches are also described.

Key Words : *structural, optimization, discrete, available sections, design, numerical methods*

Nomenclature

BBM	Branch and bound method
D_i	A set of discrete values for the ith Type 2 variable. Three different types of variables are defined as Variable Type 1 : Continuous variable Variable Type 2 : Discrete variables (including integer, zero-one) whose allowable values are given explicitly Variable Type 3 : Linked discrete variables (a discrete variable whose value specifies a group of other values linked to it)
d_{ij}	The jth discrete value for the ith Type 2 variable
E_i	A set of discrete values for the ith Type 3 variable
e_{ij}	The jth discrete value for the ith Type 3 variable
f	Objective function to be minimized
g_i	The ith constraint function
IP	Integer programming
LP	Linear programming
LBB	Linearized branch and bound
m	Total number of constraints
MDNLP	Mixed-discrete nonlinear programming
n	Total number of design variables of all types
n_d	Number of Type 2 variables
n_e	Number of Type 3 variables
NLP	Nonlinear programming
p	Number of equality constraints
q_i	Number of allowable discrete values in D_i
r_i	Number of properties for the ith Type 3 variable
s_i	Number of allowable discrete values in E_i
SA	Simulated annealing
SQP	Sequential quadratic programming
$\boldsymbol{T}^{(i)}$	The property table for the ith Type 3 variable. It is a matrix of s_i rows and r_i columns. Each row of $\boldsymbol{T}^{(i)}$ contains property data for one allowable design alternative for the ith Type 3 variable
$t_{jk}^{(i)}$	The kth property of $\boldsymbol{t}_j^{(i)T}$
$\boldsymbol{t}_j^{(i)T}$	The jth row of the property table $\boldsymbol{T}^{(i)}$
$\boldsymbol{x}$	A vector of design variables of dimension n
x_{iL}	Lower bound for the variable x_i
x_{iU}	Upper bound for the variable x_i
$\boldsymbol{Y}$	$= (\boldsymbol{y}^{(n_d+1)}, \ldots, \boldsymbol{y}^{(n_d+n_e)})^T$; a matrix whose each row is the chosen property vector (one for each Type 3 variable) ; these are used in function evaluations

1. INTRODUCTION

(1) Motivation

Optimization problems in engineering design often have variables that can only assume some pre-determined values, such as the plate thickness must be selected from available ones, the number of anchor bolts must be an integer, the number of reinforcing bars must be an integer, the number of teeth on a gear must be an integer, member cross-sectional areas must be selected from available ones, etc. To deal with optimization problems with mixed discrete-integer-continuous variables, several solution approaches have been developed and tested since 1960 s. A detailed review of general approaches that are applicable to a wide variety of discrete variable engineering optimization problems has been presented recently by Arora *et al.* (1994), Huang (1995), and Huang and Arora (1995, 1996a).

The need for discrete variable structural optimization has been recognized since 1968 (Toakley, 1968). However, since the continuous variable optimization algorithms were not fully developed at that time, emphasis shifted to the development of algorithms for such problems. There were very few papers on discrete variable structural optimization at that time. The integer variable linear programming (LP) methods (called integer programming (IP) or 0-1 programming), and the branch and bound methods had been developed for general discrete optimization problems, and so it was natural to apply these methods for discrete structural optimization. Toakley (1968) applied a few of these methods for optimal design of plastic and elastic structures. Design of plastic structures was formulated as an LP problem which was transformed to a mixed integer-continuous variable LP problem. The elastic design of determinate trusses subjected to displacement constraints was also formulated as a mixed integer-continuous variable LP problem. The algorithms used were : Gomory's cutting plane method (Gomory, 1960), branch and bound method (implicit enumeration) and heuristic techniques.

Reinschmidt (1971) used a branch and bound method to solve the problem of plastic design of building frames (which is a linear programming problem). The problem was transformed to an IP problem and a branch and bound method, based on Geoffrion's implicit enumeration approach (Geoffrion, 1967), was used. Elastic design of trusses subjected to stress, displacement, and member size constraints was also considered (which is a nonlinear programming (NLP) problem). The NLP problem was linearized and solved as a sequence of linear IP problems using the same branch and bound algorithm.

Cella and Logcher (1971) solved the nonlinear problem of designing trusses subjected to stress constraints using the branch and bound method. The nonlinear problem was attacked directly without linearization or introduction of integer variables. A filtered pattern search was used during the branching phase of the algorithm. The method simply evaluated each trial design and either accepted it or rejected it. Since each trial design required structural response, an approximate reanalysis approach was used to reduce the computational effort.

A direct method combining Box's algorithm and Hooke and Jeeve's method was use by Lai and Auchenbach (1973) for structural optimization. Liebman *et al.* (1981) transformed the discrete variable optimization problem to a sequence of unconstrained problems that were solved using an integer discrete gradient algorithm. A special enumeration algorithm for discrete variable optimization of trusses with stress and displacement constraints was developed by Hua (1983). The methods exploits the structure of the problem to develop heuristics that reduce the size of enumeration.

It is seen that the literature on discrete variable structural optimization is quite sparse. More recently, interest in this class of applications has resurfaced because the optimum design methodologies are beginning to be used in practical applications where discrete variables are encountered naturally. Therefore the main purpose of this paper is to review the modern literature on this topic. The main motivation is to understand features of the methods (limitations and positive aspects) so that further research can be performed to develop better methods.

(2) Linked discrete variables

Since the term linked discrete variables is used throughout the paper, it is important to understand the meaning of it. A *linked discrete variable* is defined as the one whose value specifies the values for a group of parameters related to it. As an example of linked discrete variables, consider the use of standard sections available in the American Institute of Steel Construction (AISC) manual for design of steel frames. Some of these sections are given in **Table 1**. The section number, section area, moment of inertia, or any other section property can be designated as a discrete design variable for the frame member. Once a value for such a discrete variable is specified from the AISC table, each of its linked variables (properties) must also

Table 1 A partial list of AISC standard sections

	A	d	t_w	b	t_f	I_x	S_x	r_x	I_y	S_y	r_y
W36X300	88.30	36.74	0.945	16.655	1.680	20300	1110	15.20	1300	156	3.830
W36X280	82.40	36.52	0.885	16.595	1.570	18900	1030	15.10	1200	144	3.810
W36X260	76.50	36.26	0.840	16.550	1.440	17300	953	15.00	1090	132	3.780
W36X245	72.10	36.08	0.800	16.510	1.350	16100	895	15.00	1010	123	3.750
W36X230	67.60	35.90	0.760	16.470	1.260	15000	837	14.90	940	114	3.730
W36X210	61.80	36.69	0.830	12.180	1.360	13200	719	14.60	411	67.5	2.580
W36X194	57.00	36.49	0.765	12.115	1.260	12100	664	14.60	375	61.9	2.560

A : Section area (in^2).
d : Depth (in).
t_w : Web thickness (in).
b : Flange width (in).
t_f : Flange thickness (in).
I_x : Moment of inertia about the x-x axis (in^4).
S_x : Elastic section modulus about the x-x axis (in^3).
r_x : Radius of gyration with respect to the x-x axis (in).
I_y : Moment of inertia about the y-y axis (in^4).
S_y : Elastic section modulus about the y-y axis (in^3).
r_y : Radius of gyration with respect to the y-y axis (in).

be assigned the unique value. For example, the data in the first row of **Table 1** must be used when the standard section W36×300 is selected during the optimization process. These variables affect values of the cost and constraint functions for the problem. Certain value for a particular property can only be used when appropriate values for other properties are also assigned. Relationships among such variables and their linked properties cannot be expressed analytically, and so a gradient-based optimization method may be applicable only after some approximations. It is not possible to use one of the properties as the only design variable because other section properties cannot be calculated using just that property. Also, if each property is treated as an independent design variable, the final solution would generally be unacceptable since the variables would have values that cannot co-exist.

(3) Scope of the review

The purpose of this paper is to focus on the review of methods for an important class of practical structural optimization problems with discrete variables that are linked to each other, such as the section properties of an I-section in the table of available sections. Since the previous review paper (Arora *et al.* 1994) does not discuss the treatment for linked discrete variables in any detail, the current paper is viewed as an extension of that paper.

It is important to note that the discrete structural optimization problems using available steel sections, discussed in the foregoing paragraphs, occur in many other engineering applications also. These include, selection of components from available ones ; e.g., bolt type, gear type, crank shaft type, automotive engine type, electric motor type, etc. Some of the methods and strategies described in this paper can be used for such applications as well.

Section 2 contains definitions of the mixed variable nonlinear programming problems with linked discrete variables. Section 3 describes single and multiple design variable formulations for problems with linked discrete variables. Section 4 contains an overview of the basic optimization methods used in various strategies for discrete variable optimization. A review of the recent approaches for discrete structural optimization based on different treatments for the design variables, is presented in Section 5. Finally, Section 6 contains some concluding remarks.

2. PROBLEM TYPES

To define three types of nonlinear structural design optimization problems, let n be the total number of design variables including continuous, discrete and linked discrete variables, n_d the number of discrete variables, and n_e the number of linked discrete variables. The variables types are defined as follows :

Variable Type 1 :

Continuous variables (x_i ; $i=n_d+n_e+1, \ldots, n$).

Variable Type 2:

Discrete variables (including integer, zero-one) whose allowable values are given explicitly (x_i ; $i=1, ..., n_d$). Some of the discrete variables may be treated as continuous variables during the solution process, but their final values must be specified from the available set. Some of the discrete variables cannot be treated as continuous variables during the solution process, so they must be kept discrete for all calculations.

Variable Type 3:

Linked discrete variables- discrete variables whose values specify a group of properties (x_i ; $i=n_d+1, ..., n_d+n_e$). It may or may not be possible to treat a linked discrete variable as a continuous variable during the solution process.

Now the three types of nonlinear design optimization problems are defined as follows :

a) *NLP : Nonlinear Programming Problem*

A structural design optimization problem can be expressed as a general *nonlinear programming (NLP) problem* of the following form (note that $n_d=0$ and $n_e=0$ in this case) :

$$\begin{aligned} &\text{minimize} \quad f(\boldsymbol{x}) \\ &\text{subject to} \quad g_i(\boldsymbol{x})=0 \,;\, i=1, ..., p \\ &\qquad g_i(\boldsymbol{x})\leq 0 \,;\, i=p+1, ..., m \\ &\qquad x_{iL}\leq x_i\leq x_{iU} \,;\, i=1, ..., n \end{aligned}$$

where f and g are objective and constraint functions respectively, x_{iL} and x_{iU} are lower and upper bounds for the variable x_i, and p and m are the numbers of equality constraints and the total constraints, respectively. The cost function f and each constraint function g_i are usually assumed to be twice continuously differentiable with respect to all the design variables. Each variable is assumed to be continuous which can have any value within its specified range.

b) *MD-NLP : Mixed-Discrete Nonlinear Programming Problem*

When some of the variables are *discrete* and others are *continuous*, we get the mixed-discrete nonlinear programming problem which is defined as (note that $n_e=0$ in this case):

$$\begin{aligned} &\text{minimize} \quad f(\boldsymbol{x}) \\ &\text{subject to} \quad g_i(\boldsymbol{x})=0 \,;\, i=1, ..., p \\ &\qquad g_i(\boldsymbol{x})\leq 0 \,;\, i=p+1, ..., m \\ &\qquad x_i\in D_i,\ D_i=(d_{i1}, d_{i2}, ..., d_{iq_i}) \,; \\ &\qquad i=1, ..., n_d \\ &\qquad x_{iL}\leq x_i\leq x_{iU} \,;\, i=n_d+1, ..., n \end{aligned}$$

where D_i is the set of allowable values for the ith discrete variable, q_i is the number of allowable values for the ith discrete variable, and d_{ij} is the jth allowable value for the ith discrete variable. In general, the number of available discrete values for each variable may be different.

c) *LD-NLP : Linked Discrete Nonlinear Programming Problem*

A precise mathematical formulation for the linked discrete and other design variable nonlinear programming problems has not been presented in the literature. Therefore, such a formulation is proposed and described here. Let $\boldsymbol{T}$ be a matrix whose each row contains values for the variables that are linked to a discrete design variable. Let one of the properties of Type 3 variable (such as the moment of inertia, area, section modulus, etc.) or the section number in the table of commercially available sections, be designated as a design variable x_i for the ith variable ($i=n_d+1, ..., n_d+n_e$). Let each row of a matrix $\boldsymbol{Y}$ contain the current selection of design properties for each Type 3 variable (such as the current selection of a member of the structure from the allowable sections) ; the number of rows in the matrix $\boldsymbol{Y}$ is n_e. Then a mixed variable optimization problem having continuous, discrete and linked discrete variables is defined as follows :

$$\begin{aligned} &\text{minimize} \quad f(\boldsymbol{x}, \boldsymbol{Y}) \\ &\text{subject to} \quad g_i(\boldsymbol{x}, \boldsymbol{Y})=0 \,;\, i=1, ..., p \\ &\qquad g_i(\boldsymbol{x}, \boldsymbol{Y})\leq 0 \,;\, i=p+1, ..., m \\ &\qquad x_i\in D_i,\ D_i=(d_{i1}, d_{i2}, ..., d_{iq_i}) \,; \\ &\qquad i=1, ..., n_d \\ &\qquad x_i\in E_i,\ E_i=(e_{i1}, e_{i2}, ..., e_{is_i}) \,; \\ &\qquad i=n_d+1, ..., n_d+n_e \\ &\qquad \boldsymbol{Y}=(\boldsymbol{y}^{(nd+1)}, ..., \boldsymbol{y}^{(nd+ne)})^T \\ &\qquad \boldsymbol{y}^{(i)T}\in \text{row}(\boldsymbol{T}^{(i)}) \,; \\ &\qquad i=n_d+1, ..., n_d+n_e \end{aligned}$$

(If $x_i=e_{ij}$ then the jth row of $\boldsymbol{T}^{(i)}$ is chosen as $y^{(i)T}$)

where $\boldsymbol{T}^{(i)}=(\boldsymbol{t}_1^{(i)}, ..., \boldsymbol{t}_{si}^{(i)})^T$ and $\boldsymbol{t}_j^{(i)T}=(t_{j1}^{(i)}, ..., t_{jri}^{(i)})$; $j=1, ..., s_i$

$$x_{iL}\leq x_i\leq x_{iU} \,;\, i=n_d+n_e+1, ..., n$$

where D_i and E_i are the sets of discrete values for the ith Type 2 and ith Type 3 variables, respectively, d_{ij} and e_{ij} are the jth allowable discrete values for the ith Type 2 and Type 3 variables, respectively, integers q_i and s_i are the numbers of allowable discrete values in the sets D_i and E_i, respectively, and r_i is the number of properties for the ith Type 3 variable. Matrix $\boldsymbol{T}^{(i)}$ is the allowable property table for the ith Type 3 variable. It is a matrix of s_i rows and r_i columns. Each row of $\boldsymbol{T}^{(i)}$ contains property data for one allowable design alternative for the ith Type 3 variable. An example of $\boldsymbol{T}^{(i)}$ is given in **Table 1** in which each row of data gives section properties of an allowable AISC steel section. In design of steel structures using commercial sections, if the moment of inertia is chosen as a Type 3 variable x_i for the ith steel member, then the allowable discrete values for the moment of inertia form the set E_i for the ith member. Other properties (such as the section dimensions, moduli, and area) for each section are used to construct the matrix $\boldsymbol{T}^{(i)}$ of allowable sections for the ith member of the structure. When x_i is assigned one of the values in E_i, say e_{ij}, the jth row of $\boldsymbol{T}^{(i)}$ that contains properties of the jth AISC section is also chosen to be the ith row of matrix $\boldsymbol{Y}$. The matrix $\boldsymbol{Y}$ that contains selected sections for the entire structure, is then used to analyze the structure and evaluate the cost and constraint functions.

3. DISCRETE STRUCTURAL OPTIMIZATION: PROBLEM FORMULATIONS

Design variables for problems with linked discrete variables, such as design of steel frames using commercial sections, can be defined in several ways leading to different solution approaches. Most approaches are based on defining only one design variable for each member. Some approaches have also been developed using multiple design variables for each member. These formulations are described in the following sections.

(1) Single variable formulations

a) *Single Design Variable Formulation 1*

For each steel member, one of the section properties is chosen as a Type 1 design variable and other properties are related to it via some interpolation scheme (instead of using the matrix $\boldsymbol{T}$ directly). A separate procedure is needed to select members from the available sections. All the design code constraints are difficult to impose since the interpolation scheme may not provide enough information about all the section properties.

b) *Single Design Variable Formulation 2*

For each steel member, one of the section properties is chosen as a Type 3 design variable. The problem is formulated as LD-NLP and all design variables are of Type 3. It may or may not be possible to impose all the design code constraints, depending on how the section properties are treated.

c) *Single Design Variable Formulation 3*

None of the section properties is used as design variable. Instead, for each member, an integer design variable is used (with allowable values as 1, 2, 3, ...). The value of this integer variable indicates the selected section from Matrix $\boldsymbol{T}$ (i.e., a row from **Table 1**). The problem is formulated as LD-NLP and all design variables are of Type 3. All the design code constraints can be imposed.

(2) Multiple variable formulations

a) *Multiple Design Variable Formulation 1*

For each member, multiple section properties are treated as Type 2 design variables. The problem is formulated as MD-NLP and all design variables are of Type 2. Matrix $\boldsymbol{T}$ is not used since there is no Type 3 variable. The final selection of sections from the ones available in the matrix $\boldsymbol{T}$ must be implemented separately. All the design code constraints cannot be imposed.

b) *Multiple Design Variable Formulation 2*

For each member, multiple section properties are treated as Type 1 (continuous) design variables. The problem is formulated as a standard NLP. Matrix $\boldsymbol{T}$ is not used since there is no Type 3 variable. The final selection of sections from the ones available in the matrix $\boldsymbol{T}$ must be implemented separately. Depending on the design variables chosen, it may or may not be possible to impose all the design code constraints.

4. BASIC OPTIMIZATION METHODS

In Section 2, different types of mixed-discrete-continuous variable optimization problems are defined. In Section 3, different design variables used to formulate the mixed variable nonlinear programming problem with linked discrete variables are presented. This section contains, an overview of some of the basic methods that have been used either as stand-alone, or as part of an overall solution approach for problems with linked discrete design variables. Many of the methods have been described in more detail in Arora *et al.* (1994). This section concentrates only on those method that are suitable for problems with linked discrete variables and the implementations by recent researchers.

(1) Nonlinear programming methods

Many methods have been developed in the literature for solving nonlinear optimization problems. These methods are described in Arora (1990) and in many textbooks and other references cited in there. Many solution approaches for structural optimization problems use these methods. For example, Balling and co-workers have used a hybrid generalized reduced gradient method (Hager and Balling, 1988). Arora and co-workers have used sequential quadratic programming (SQP) methods (Arora, 1989, 1990; Al-Saadoun and Arora, 1989; Huang and Arora, 1995, 1996). Still others have used generalized reduced gradient and feasible directions methods (Arora, 1990). Work continues in this area to develop still better solution approaches, especially for large scale problems.

For problems with large number of design variables, the computational effort for various structural analyses becomes massive. This is even more burdensome for mixed variable optimization since the analysis part has to be repeated numerous times. Thus the way the problem is formulated can greatly affect the time required to obtain the solution. Two approaches are often used to reduce the computational effort. One is to use approximation concepts for function evaluations (Barthelemy and Haftka, 1993). This speeds-up reanalysis of the structure. Some discrete optimization methods (e.g., branch and bound method) solve several continuous subproblems The well-known SQP methods have been used often as the subproblem solver. However, for large scale problems in which function evaluations are expensive, a sequential linear programming (SLP) method which uses linearized functions can be more suitable. Although the SLP method does not converge as well as the SQP method, the number of SLP iterations can be limited to obtain the solution faster since the accuracy of the subproblems is often not critical. Several implementations of the linearization approach have been discussed in Arora *et al.* (1994).

Another approach to reduce the computational effort is to use some known characteristics of the problem and thus obtain a better convergence rate. One such approach is called the optimality criteria methods in which an iterative procedure based on the behavior of the structure is derived and the optimum is found when this criterion is satisfied. Optimality criteria methods (OC) are often referred to as the indirect methods. Compared to the direct methods which minimize the objective function by a search procedure, the indirect methods work based on an iterative solution of the optimality conditions. For each class of the structural design problems, the optimality conditions are reduced to a simple criterion based on the structure type and the constraints considered. An iterative scheme is then developed for each class of problems to make design improvements. These are usually called the scaling procedures. For optimal design of trusses subjected to stress and displacement constraints, the optimality conditions lead to the conclusion that the strain energy density must be uniform throughout the structure (Venkayya 1971). This criterion is then used to make design improvements and obtain an optimal design. For some truss structures, the optimality conditions are interpreted to imply that the optimal design is obtained when each member is fully-stressed under at least one loading condition. Thus this condition can be used to devise a numerical procedure to obtain an optimal design. For more realistic structural design problems, other optimality criteria related to displacement, stability, natural frequency, etc., have to be used simultaneously. In these methods, the design change from one iteration to another can be large, resulting in a faster approach to a near optimal solution. The main disadvantages of the methods are their lack of generality (since the optimality criterion needs to be derived for each class of problems) and guarantee of convergence. Nevertheless, Grierson and co-workers have extensively used optimality criterion methods for design of steel frameworks using commercially available sections (Grierson and Lee, 1984, 1986; Chan *et al.* 1995).

(2) Branch and bound method (BBM)

a) *Basic BBM*

Branch and bound is perhaps the most widely known and used method for discrete variable optimization. The method was originally developed for linear problems; however, it has been applied to all types of linear and nonlinear problems with mixed variables. Many variations of the method have been implemented. When used for linear problems, the method can be implemented to converge to a global minimum point. However, for nonlinear problems, no such guarantee can be given unless an exhaustive search is performed which can be quite expensive. In this section, the basic BBM method is explained, followed by some of its variations that have been used for mixed variable nonlinear optimization problems.

The BBM is basically an *enumeration* method in which one systematically tries to reduce the number of trials to reach the minimum point (that

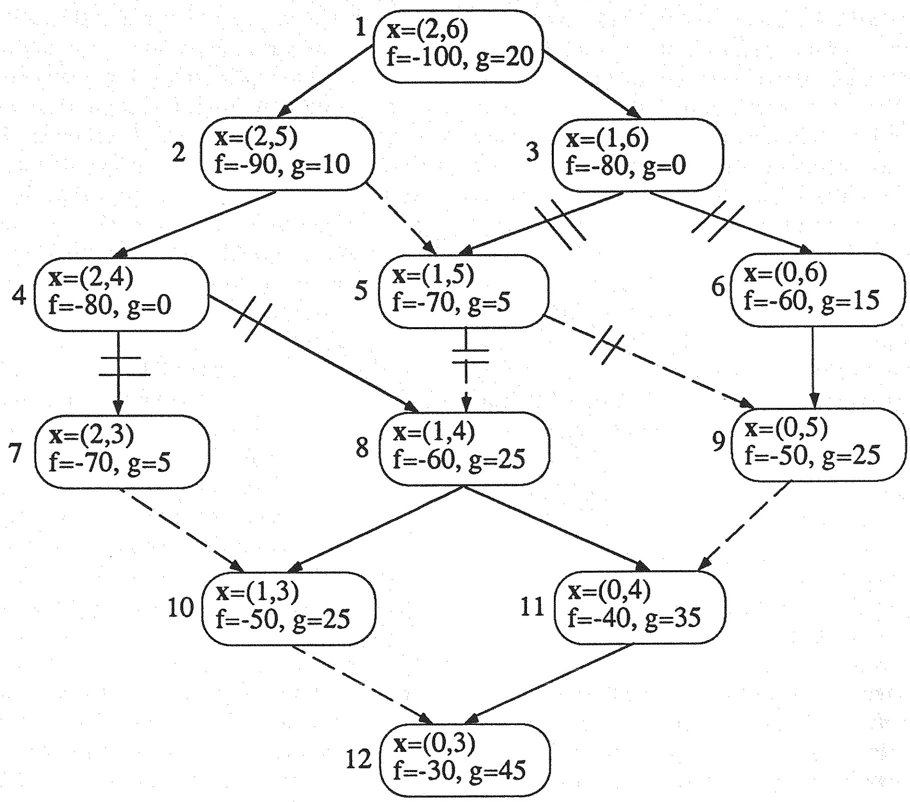

Fig.1 Branching and Fathoming Process for the Example Problem

is why it is sometimes called an *implicit enumeration* method). The concepts of *branching, bounding and fathoming* are used to achieve these objectives. To explain these concepts, consider the following linear integer programming problem :

$$\text{minimize} \quad f=-20x_1-10x_2$$

$$\text{subject to} \quad g_1=-20x_1-10x_2+75\leq 0$$

$$g_2=12x_1+7x_2-55\leq 0$$

$$g_3=25x_1+10x_2-90\leq 0$$

$$x_1\in\{0,1,2\},\ x_2\in\{3,4,5,6\}$$

x_1 and x_2 cannot have non-integer values during the solution process.

This example is the same as used by Huang and Arora (1996 a) ; however, the available discrete sets have been modified to reduce the size of enumeration. Note that the derivatives of f with respect to x_1 and x_2 are always negative for the problem. Thus one can enumerate the discrete points in the descending order of x_1 and x_2 to ensure that the cost function value is always increased when one of the variables is perturbed to the next lower discrete value. **Fig.1** shows all the possible designs for this problem. Since x_1 has three and x_2 has four possible values, there are a total of 12 combinations of designs. In general, the number of combinations would be $(\Pi_{i=1}^{n} q_i)$ for n discrete variables, each having q_i possible values. Each of the possible designs in **Fig.1** is called a node. For each node, the design variable values ($\boldsymbol{x}$), the cost function value (f) and the maximum constraint violation (g) are shown. For example, Node 5 corresponds to $\boldsymbol{x}=(1, 5)$, f$=-70$, and $g=5$. From each node, one or two nodes are branched out, shown by arrows, where the function values are evaluated. This is called *branching*. The dashed arrows indicate alternate branching paths which implies that the process of branching is not unique. As the branching process continues, a feasible design is obtained at Node 3. The cost function value for this node becomes an upper *bound* for the cost function values at other nodes. Any feasible or infeasible node with a larger value for the cost function need not be branched further because a feasible design with a smaller value for the cost function is not possible. For the present example, it is seen that a feasible design with smaller value for the cost function cannot be obtained after Node 3 or 4 ; these nodes are said to have *fathomed* ; i.e., reached their lowest point on the branch and no further branching is necessary from them. This is shown in **Fig.1** by a break in the arrows. If there is any constraint violation at a node and the cost function value is smaller than the established upper bound or an upper bound has not been established yet, further branching is necessary, as for Nodes 1 and 2 for the present example. In **Fig.1**, Node 5 need not be branched further because the design is infeasible with the cost function value higher than the established upper bound of -80. It is seen that Nodes 3 and 4 give two global minimum points for the example with cost function value of -80. The branch and bound method would find these solutions in four trials (five trials if Node 5 is branched from Node 2) instead of the 12 trials for the complete enumeration.

b) *BBM with Local Minimization*

For optimization problems where the discrete variables can have non-discrete values during the solution process and all the functions are differentiable, one can take advantage of the local minimization procedures to reduce the number of nodes. In this BBM procedure, initially an optimum point is obtained by treating all the discrete variables as continuous. If the solution is discrete, the process is terminated. If one of the variables does not have a discrete value, then its value lies between two discrete values ; e.g., $d_{ij} < x_i < d_{ij+1}$. Now two subproblems are defined, one with the constraint $x_i \leq d_{ij}$ and the other with $x_i \geq d_{ij+1}$. This process is also called *branching* which is slightly different from the one explained earlier for purely discrete problems. It basically eliminates some portion of the continuous feasible region which is not feasible for the discrete problem. However, any of the discrete feasible solutions is not eliminated. The two subproblems are solved again, and the optimum solutions are stored as nodes of the tree containing optimum values for all the variables, the cost function and the appropriate bounds on the variables. This process of branching and solving continuous problems is continued until a feasible discrete solution is obtained. Once a feasible discrete design is obtained, the cost function corresponding to this solution becomes an *upper bound* on the cost function for the subproblems (nodes) to be solved later. The solutions that have cost function values higher than the established upper bound are eliminated from further consideration (i.e., they are fathomed).

The foregoing process of branching and fathoming is repeated from each of the unfathomed nodes. The search for the optimum terminates when all the nodes have been eliminated due to one of the following reasons : (1) a discrete solution is found, (2) no feasible continuous solution can be found, or (3) a feasible solution is found but the cost function value is higher than the established upper bound. Since the BBM applied to mixed variable problems requires solution of continuous subproblems, there are two major drawbacks : (1) the number of continuous subproblems to be solved becomes very large when the number of design variables is large, and (2) for problems with linked discrete variables, the lack of differentiability of the problem functions limits the use of continuous optimization solvers. For structural optimization problems with linked discrete design variables, the first drawback can be overcome by replacing the original nonlinear problem with an approximate problem which is more efficient to solve (Barthelemy and Haftka, 1993). The second drawback can be overcome by transforming the problem to have differentiable functions. These aspects are explained later in Section 5.

There are many variations of the foregoing BBM for nonlinear problems. Many enhancements enable the method to create fewer subproblems (nodes) without neglecting subproblems with possible lower cost function values. BBMs with enhancements can be found in Tseng, Wang and Ling (1995), Huang and Arora (1995, 1996 a), and Huang (1995).

c) *BBM with Multiple Branches*

Until now, the BBM has only been used by very few researchers when dealing with linked discrete variables, such as the frame design problems using available sections. With such applications in mind, Hager and Balling (1988) have proposed a modified BBM where multiple branches are allowed from a node. The method uses the basic concepts of branching, bounding and fathoming, and solves continuous variable optimization problems at each node. The method has several levels of nodes, the total number being equal to the number of design variables for the problem. The first level consists of a number of nodes where the first design variable is assigned an allowable discrete value. Thus the number of nodes at the first level is equal to the number of allowable discrete values for design variable 1. Treating all other variables as continuous, each node is solved using a local minimization procedure. The cost function value for each node is noted, and from a node with the smallest value for the cost function, many nodes are *branched*. For each new node, the second design variable is also assigned a value from the available ones. Therefore, the number of new nodes created from a first level node is equal to the number of allowable values for the second design variable. Each new node is optimized keeping the first two variables fixed and treating the remaining variables as continuous. If any of the nodes corresponds to an infeasible problem, it is *fathomed* and no further branching is necessary from there. Now all the nodes from the first and the second level are searched for the smallest cost function value. If this node is at level one, then it is branched into many new nodes ; for each new node, the second variable is assigned an allowable value. If the node to be branched is at level two, then multiple nodes are branched from there, and the third design variable is fixed to an allowable value for each new node. This creates nodes at the third level and the number of nodes branched from that node is equal to the number of allowable values for the third discrete design variable. Once the new nodes are created, the local minimization procedure is used to solve for the remaining continuous variables and the cost function value for the node.

The foregoing procedure of creating new nodes and new levels is continued until a discrete feasible solution is obtained. Note that this will occur when some nodes are created at all the levels. The node with a discrete feasible solution and the smallest value for the cost function gives an upper *bound* for all the remaining nodes at all the levels. Now all the nodes at all levels that have cost function value higher than the established upper bound are *fathomed*. Branching is done from the unfathomed nodes until all the remaining nodes are fathomed or a discrete solution with the smaller cost function value is obtained. The new discrete solution establishes a better upper bound for the cost function. At the end of the procedure, the best discrete solution is taken as the optimum design.

The foregoing form of the BBM along with linearization of the problem has been demonstrated for steel frame optimization problems (Hager and Balling, 1988). However, the method can be applied to other discrete variable optimization problems.

(3) Simulated annealing

Simulated annealing (SA) is a simple technique that can be used to find global minimizer for continuous-discrete-integer variable nonlinear programming problems (Aarts and Korst, 1989). The approach does not require continuity or differentiability of the problem functions because it does not use any gradient or Hessian information. The basic idea of the method is to generate random points in a neighborhood of the current best point and evaluate the problem functions. If the trial point is infeasible, it is rejected right away. If the trial point is feasible and the objective function value is smaller than its current best value, then the point is accepted, and the best objective function value is updated. If the point is feasible but the objective function value is higher than the best value known thus far, then the question is whether to accept or reject the point. The answer is that it is sometimes accepted and sometimes rejected. The acceptance is based on the value of the probability density function of Bolzman-Gibbs distribution. If this probability density function has value (this is called *acceptance probability*) greater than a random number, then the trial point is accepted as the best solution even if its objective function value is higher than the known best value.

In computing the probability density function, a parameter called the *temperature* is used. For the optimization problem, this temperature can be the target value for the objective function corresponding to the global minimizer. Initially, a larger target value is selected. As the trials progress, this target value is reduced (this is called the *cooling schedule*), and the process is terminated after a fairly large number of trials.

The acceptance probability steadily decreases to zero as the temperature is reduced. Thus in the initial stages, the method is likely to accept worse designs while in the final stages, the worse designs are almost always rejected. This strategy avoids getting trapped at local minimizers. The main deficiencies of the method are the unknown rate at which the target level is to be reduced, and uncertainty in the total number of trials and in the number of trials after which the target level needs to be reduced.

The SA method has been demonstrated recently for engineering problems with discrete variables. Kincaid and Padula (1990) have used the method to determine the arrangement of manufactured members, having small errors in their lengths, for the layout of a truss structure to minimize distortion and member forces. Balling (1991) has demonstrated the method for optimization of steel frameworks with linked discrete variables. May and Balling (1992) proposed a filtered simulated annealing strategy. Based on the known gradient information, this method can probabilistically filter out many of the poorer candidates. An implementation of the simulated annealing algorithm can also be found in Huang and Arora (1995, 1996 a) and Huang (1995).

(4) Genetic algorithms

Like the simulated annealing methods, the genetic algorithms are also in the category of stochastic search methods and use implicit enumeration procedures (Goldberg, 1989). Their philosophical basis is in Darwin's theory of survival of the fittest. A set of design alternatives representing a population in a given generation are allowed to *reproduce* and *cross* among themselves, with bias allocated to the most fit members of the population. Combination of the most desirable characteristics of mating members of the population results in progenies that are more fit than their parents. Thus, if a measure which indicates the fitness of a generation is also the desired goal of a design process, successive generations produce better values of the objective function. An advantage of this approach is that no gradient information is needed, as for the SA method. Therefore, differentiability requirements–needed in gradient based methods–can be relaxed.

In a genetic algorithm, one starts with a set of feasible designs randomly generated. From this set, new and better designs are reproduced using the fittest members of the set. Each design must be represented by a finite length string. Usually binary strings have been used for this purpose. The entire process is similar to a natural population of biological creatures, where successive generations are conceived, born and raised until they are ready to reproduce. Three operators are needed to implement the algorithm : (1) reproduction ; (2) crossover ; and (3) mutation.

Reproduction is an operator where an old string is copied into the new population according to the string's fitness. Here fitness is defined according to the objective function value. More highly fit strings (those with smaller objective function values) receive higher numbers of offspring. There are many different strategies to implement this reproduction operator.

The next operator – *crossover* – corresponds to allowing selected members of the population to exchange characteristics of the design among themselves. Crossover entails selection of starting and ending positions on a pair of mating strings at random, and simply exchanging the string of 0's and 1's (for a binary string) between these positions. This is akin to transfer of genetic material in biological reproduction processes facilitated by DNA strings.

Mutation is the third step in this genetic refinement process, and is one that safeguards the process from a complete premature loss of valuable genetic material during reproduction and crossover. In terms of a binary string, this step corresponds to selection of a few members of the population, determining a location on the strings at random, and switching the 0 to 1 or vice versa.

The foregoing three steps are repeated for successive generations of the population until no further improvement in the fitness is attainable. The member in this generation with the highest level of fitness is the optimum design. Sugimoto (1992) used a genetic algorithm for discrete optimization of truss structures. An operator, called *growth*, was proposed to improve the reliability for some design problems. Lin and Hajela (1992) implemented a genetic algorithm for optimal design of structural systems with mixed variables. In their approach, the constrained minimization problem was transformed to an unconstrained problem using the exterior penalty function formulation. An implementation of the genetic algorithm can also be found in Huang and Arora (1995, 1996 a).

5. SOLUTION STRATEGIES FOR LD-NLP PROBLEMS

Solution approaches for LD-NLP problems can be divided into four broad categories based on how the problem is formulated and solved. The first two approaches use only one design variable for each member of the structure, the third one uses mixed-single and multiple design variable formulations, and the fourth one uses a continuous variable formulation along with a rounding-off procedure. These four approaches are described in the following subsections.

Most of the solution approaches use two phases to obtain the final discrete solution. In the first phase, the LD-NLP problem is somehow formulated as a standard NLP problem where all the variables are treated as continuous. This is also true for many of the solution procedures for the MD-NLP problems (Arora *et al*, 1994). The problem is then solved using any one of the many available NLP methods. In the second phase, the continuous solution is manipulated using a discrete variable optimization method to obtain the final solution.

The following issues related to the structural optimization methods using available sections have been addressed in the literature : (1) Does the method require the table of available sections to be re-arranged in any fashion? (2) Does the method need to be re-derived if a set of new sections become available? (3) Can the method impose all the constraints dictated by the design code? Comments will be made relative to these issues for the methods reviewed.

(1) Single variable approaches with approximations

Single design variable formulation for optimization of steel structures using available sections has been the most popular approach in the literature. For example, one of the section properties (e.g., area or moment of inertia) can be treated as a continuous design variable for each member and other properties are *approximately* linked to it. This corresponds to Single Design Variable Formulation 1 of Section 3.(1). It is difficult to use multiple properties, such the section area (A), moments of inertia (I_x and I_y) and section moduli (S_x and S_y) for each member, as independent design variables. With such a formulation, it is likely that the section area will reach its lower bound to minimize the weight, and the section modulus S_x for a beam element will be driven upward to satisfy the bending stress constraints. This will result in a member that is not close to any of the available sections.

Many researchers use section area as the sole design variable (continuous) and *approximate* other properties using the following relationships :

$$I_x, I_y, S_x, \text{ or } S_y = A^{r_i} \qquad (1)$$

where r_i for each property is determined using a curve fitting procedure. An advantage of this approach is that the relationships represented in Eq. (1) are continuously differentiable. Therefore a discrete variable optimization method that uses gradient or Hessian information can be used. There are a few drawbacks of this approach, however : (1) The approach cannot be used for applications where cross-sectional dimensions and the properties not represented by Eqs. (1) are also needed, such as for composite beams, and other applications where local buckling and member size constraints need to be imposed explicitly. (2) The approximation represented in Eq. (1) is not very accurate for all sections because the number of available sections is large and they vary in shape as well as size. It is impossible to distinguish between different available sections that have same or nearly same values for the independent section property. One may get around this dilemma to some extent by confining the search to the "economy sections" for beams and a similar set for columns. For such sections, the dependent section properties increase monotonically with the section area. This, however, implies that all the sections available in the AISC tables cannot be used for design. Also, in 3D frames where columns are subjected to biaxial bending and axial force, it is not possible to specify monotonic set for all the needed properties. The limitation about the accuracy of the relationships in Eqs. (1) can be overcome to some extent by dividing the available standard sections into several smaller groups. Each group contains only the sections (selected from the matrix $\boldsymbol{T}$) for which more accurate relationships of the form given in Eq. (1) can be developed for various properties through a curve fitting procedure.

Grierson and co-workers have extensively developed and demonstrated procedures for discrete variable optimization based on the foregoing philosophy. For example, Grierson and Lee (1984) formulate the problem of optimal design of 2D frameworks using the section area as the only design variable. The moment of inertia and the extreme fiber distance are related to the section area through relationships of the type given in Eq. (1). Database of available sections (wide flange, tee and double angle) is extensively analyzed and divided into several data sets. For each data set, the constants that relate moment of inertia and fiber distance to the section area are calculated in such a way to have more accurate relationship in Eqs. (1). The members of the structure are divided into three groups : axial force members, pure flexural members, and combined axial-flexural members. The data sets of the available sections are also identified for each type of the member. These data sets are re-arranged into selection tables in the ascending order of the key member property-the section area for each axial or axial- flexural member, and moment of inertia for each flexural member. The entire optimization procedure consists of two phases. Phase I uses 3 iterations of a continuous variable optimization method (based on an optimality criterion) to obtain a good starting point for Phase II. The available sections are not assigned in this phase. However, as a result of the partitioning of the available sections into data sets, the solution at the end of Phase I is likely to be quite close to a discrete solution. In Phase II, a discrete solution is sought using a dual algorithm (Fleury, 1979). A few drawbacks have been noted for the developed procedure : (1) During Phase II where discrete sections are selected, large infeasibility can occur due to the selection process. (2) The available sections must be divided into suitable economical groups and for which the analytical relationships between the primary section property and the secondary properties are accurate. This implies that any new section must be analyzed for inclusion in appropriate data sets. (3) The members must be identified as axial, flexural and axial-flexural.

Grierson and Lee (1986) extend their previous work on discrete variable optimization of steel frames to include constraints under ultimate load conditions. These constraints ensure adequate safety against collapse. Grierson and Cameron (1989) describe feature of a computer program required for practical applications in design of steel frameworks. Constraints on strength/stability of members and stiffness of the structure are imposed. Member buckling constraints are treated by adjusting the allowable stress limits. The effective length factors are automatically calculated for buckling strength calculations. A two phase procedure for optimal design of such structures using available discrete sections is presented and demonstrated for a 2D mill crane building framework. In the first phase, a member-by-member procedure is used to assign sections based on the current analysis results only, and no formal optimization is performed. This procedure is based on approximations and heuristics, but it is very fast. In the second phase, a continuous variable optimization is performed using the section area as the only design variable. The problem is linearized using Taylor's expansions for all the constraints and the cost function. The procedures developed previously by Grierson and Lee (1984) are used to relate moment of inertia and extreme fiber distance to the section area. Also the available sections are divided into several data sets, as explained previously. The two phases are executed in sequence until no changes occur in the weight of the structure. Cameron, Xu and Grierson (1991) have extended the previous work for discrete variable optimization of 2D frameworks to design of 3D frameworks.

Chan (1992), and Chan, Grierson and Sherbourne (1995) have developed a procedure for optimal design of tall building steel frameworks using available sections. The example used to demonstrate the procedure consists of a 3D unsymmetrical frame. Only the section area of members is treated as a design variable; all other section properties are related to the area through regression analyses. Members of the structure are grouped together to reduce the total number of design variables. The entire solution process, consisting of several phases, is very nicely summarized in a section called "Overall Design Procedure" (Chan *et al.* 1995). Initially all members are selected to be the largest allowable sections. Explicit approximate expressions for the two drift constraints (drift in two directions for 3D frames) are obtained, assuming the member internal forces to remain unchanged. Using these expressions, the members are re-sized to satisfy the constraints approximately. Analysis of the structure is then performed to check the member strength requirements. The lower limits on the member sizes are adjusted to satisfy the strength requirements. In the next phase, all the inter-story drift constraints are considered, and a numerical optimality criterion method is used to obtain a continuous variable optimum solution. The Gauss-Seidel iterations are used to solve for the Lagrange multipliers for the constraints while imposing non-negativity constraints on them. The last phase of the solution process involves final discrete member specification. After continuous variable optimum solution is obtained, penalty on the structural weight for each member to be specified a higher available section is calculated. A few members that have least penalty on the weight are assigned discrete available sections, and continuous variable optimization is performed again with the reduced set of design variables. The procedure is continued until all members have been assigned discrete sections. This adaptive member selection procedure works quite well and is similar to the one proposed by Arora (1989) and demonstrated for optimum design of truss structures.

Balling (1991) implemented a simulated annealing (SA) method for optimum design of 3D steel frameworks using available sections. An unsymmetric 3D 6-story frame subjected to 3 loading conditions was considered as an example problem. Members of the structure were divided into 11 groups to reduce the number of independent variables. Section number from the AISC table was used as the basic design variable for each group of members. To reduce the number of analyses required in SA, approximation concepts were used to re-analyze the structure (Vanderplaats and Salajegheh, 1989). In these approximations, the nodal displacements, member end forces, and effective length factors were approximated by a first order Taylor's series in terms of the reciprocal section properties of the members (area, strong and weak moment of inertia, and strong and weak section moduli). Complete analysis required 44 seconds, approximate re-analysis 4.7 seconds, and the initialization for approximate re-analysis (sensitivity analysis) 324 seconds of CPU time. The approximate re-analysis was used throughout the SA iterations. The exact analysis was performed for the final design to check its feasibility. The final design was found to have acceptable level of infeasibility. Two files containing tables of economical sections from the AISC table were prepared: one for the columns and the other for girders. These files were arranged according to the decreasing section area. A member-by-member search strategy based on random number generation was used to come up with a good initial design for the SA procedure. In this strategy the members were allowed to change within 4 neighboring sections. Approximate analysis procedure was used in this phase as well to evaluate designs. In the second phase, the SA strategy repeatedly generated candidate designs in a neighborhood of the current design. These designs were generated by randomly perturbing one of the 11 discrete variables at a time. A cycle was defined to consist of 11 candidate designs. The SA parameter, temperature, was held constant while each of the 11 discrete variables was perturbed. The range of perturbation of a discrete variable was between -2 and $+2$, and the perturbation was selected randomly. Thus the candidate designs may increase or decrease the total weight of the structure. If the candidate design was infeasible while the current design was either feasible or less infeasible, it was rejected right away. The final solution for the example frame was compared to the one obtained with the linearized branch and bound (LBB) method of Hager and Balling (1988) which is explained later in the paper. For small size of the neighborhood, the LBB was more efficient than the SA ; however, the cost function was higher with LBB. The result was opposite with a larger size of the neighborhoods ; actually the LBB became quite inefficient.

May and Balling (1991, 1992) have observed that the SA strategy generates many designs that are very poor because they are quite heavy or infeasible. Such designs should not be considered for analysis and acceptance in SA. Therefore, they have developed a filter for the SA strategy that blocks many of the poor deigns. After a design is generated, its "potential" is estimated. This potential is calculated using candidate design's weight (which can be calculated without any approximations) and the maximum constraint value which is estimated using the approximate analysis procedure. If the candidate is deemed feasible and the frame weight is lighter, it is passed through the filter for SA acceptance check. Otherwise, a probability of passing through the filter is calculated as $\exp(-C/S)$, where C is the normalized potential for the member and S is the filter size. The parameter C is calculated as ratio of the increase in the weight of the candidate design and a running average of the weight, and/or the ratio of the maximum constraint value and a running average for it. The filter size is calculated based on a parameter P_p (specified by the user) as $-1/\ln(P_p)$. As P_p is decreased, the filter size decreases, thus more designs are blocked. The candidate design is passed through the filter if this calculated probability is larger than a random number ; otherwise the candidate is rejected and a new candidate is generated. This strategy retains the essence of the SA method where worse design are occasionally passed through the filter. The new method is shown to be more efficient than the standard SA and slightly more efficient than the LBB.

(2) Single variable approaches without approximations

Instead of using Eq. (1), the table containing data for all the AISC sections can be used directly in structural optimization. For example, the section area A can be used as the sole discrete design variable (Type 2) and, when structural analysis is needed, the table can be searched to obtained proper values of other section properties corresponding to the current value of A. Therefore, the mixed variable optimization for the steel structural design problem becomes a more general form represented in the problem LD-NLP in which other properties are related to the sole design variable via the matrix $\boldsymbol{T}$. This corresponds to the Single Design Variable Formulation 2 of Section 3.(1). However, this relationship (via matrix $\boldsymbol{T}$) is not continuously differentiable and hence a gradient-based method cannot be used. Huang (1995) and Huang and Arora (1996 b) have used this approach during Phase II of a solution process for optimization of steel frames with available sections. The procedure, explained later in the paper, uses a branch and bound method during Phase II.

Another approach would be to use the available section number (i.e., the row number of the matrix $\boldsymbol{T}$) as the integer design variable for each member of the structure. This corresponds to the Single Design Variable Formulation 3 of Section 3.(1). Once the section number is specified, all its properties can be obtained from the appropriate row of the table and used in all the calculations. Liebman, Khachaturian and Chanaratna (1981) have used this approach for optimal design of steel frames. The constrained optimization problem is transformed to an unconstrained one using the interior penalty functions. The unconstrained problem is then solved using the integer gradient direction method of Glankwahmdee *et al.* (1979). Three example problems are solved : a reinforced concrete beam and two framed structures. The table of available sections needs to be re-arranged such that the section areas are in an ascending order. All the design code constraints can be explicitly imposed. An initial feasible point is needed to start the search process which may be difficult to obtain in some applications (Elwakeil and Arora, 1995). The method is quite simple to implement; however, it can be quite time consuming on the computer because integer gradient evaluation as well as step size calculation can require a large number of analyses. Also, the penalty function approach to impose the constraints can be time consuming.

For mixed variable optimization problems, Amir and Hasegawa (1989) have also used an approach that is quite similar to the one used by Liebman *et al.* (1981). In their approach, any continuous variable is also transformed to a discrete variable. Some modifications of the previous approach are suggested to improve the search process; i.e., if the process fails along the calculated discrete search direction, then some neighborhood points are searched for improved solutions. Three example problems are solved : a hollow rectangular simply supported beam, a reinforced concrete beam, and a mill building structure. Weight is minimized in each case, except for the reinforced concrete beam where the cost is minimized. For the building problem, available section number is chosen as discrete design variable. The members of the structure are divided into six groups, so there are six design variables. All the available sections must be re-arranged in an ascending order for the section area. All the design code constraints can be imposed explicitly.

Two stochastic methods, simulated annealing and genetic algorithm, can also be used to solve problems with linked discrete variables, formulated as LD-NLP. The methods are known to be slow; however, an advantage is that the gradients of functions are not required. Therefore analytical relationships (such as Eq. (1)) among the properties need not be provided and the matrix $\boldsymbol{T}$ (such as **Table 1**) can be used directly. It is not necessary to use any one of the properties as the only design variable; instead, an integer design variable (the section number) can be used; i.e., Single Design Variable Formulation 3 is used. During the solution process, values for all other properties are obtained from the matrix $\boldsymbol{T}$ according to the integer design variable which is determined by the stochastic method. All the design code constraints can be checked since all the section properties are precisely known. In their pure form, these approaches have not been used for design of steel frames because they are extremely time consuming when the number of design variables is large and the number of available sections for each member is large. The techniques, however, have been combined with other methods to reduce the computational burden. Such two phase approaches using the SA and GA algorithms during Phase II, are explained in Sections 5.(1) and 5.(3).

(3) Mixed single and multiple variable approaches

Another approach to deal with dependent design properties is to treat some of them as independent design variables. For the planar steel frames, the design variables could be the section area, moments of inertia and section moduli. As noted earlier, however, treating dependent variables as independent design variables often results in an impractical design.

Hager and Balling (1988) have developed a two phase procedure for optimum design of planar steel frames using multiple section properties as design variables along with a branch and bound method; i.e., Multiple Design Variable Formulation 1 is used. The procedure is demonstrated on a 3-bay, 8 story frame, adapted from the one used by Liebman *et al.* (1981). The frame is subjected to uniform vertical loads including the self weight and a side load at each story. In Phase I of the procedure, a continuous variable optimization problem is formulated using the section area, strong axis moment of inertia and section modulus as the design variables. A hybrid-generalized reduced gradient method is used to determine the continuous optimum. It is noted that since the final continuous solution will not be close to any of the discrete sections, it will not be very useful. This difficulty is mitigated by adding enveloping constraints to the continuous optimization problem. Using all the "economy sections", a convex hull is constructed which essentially defines new linear constraints for the problem, forcing the final solution to be close to the available discrete sections. In Phase II, a modified branch and bound method (BBM), explained previously, is used to determine the discrete solution. To reduce the size of enumeration in BBM, Hager and Balling propose to define small neighborhoods consisting of 3 or 4 sections around the continuous optimum design for each member. During BBM, the solution is searched only in these neighborhoods. Even with this strategy, the number of trial designs is quite large requiring enormous computational effort for structural analysis. To overcome this difficulty, the problem is linearized about the continuous solution using the three design variables of Phase I. With this approach, the problem to be solved in Phase II becomes a linear programming (LP) problem, and so the method is called linear branch and bound (LBB) method. As explained in Section 4.(2), some members are assigned discrete sections while others are represented by the continuous variables. The resulting problem is solved using the Simplex method of linear programming. Then some more members are assigned discrete sections, and the procedure is continued until all the members have been assigned available sections. Another procedure is developed to further improve efficiency of the BBM method by specifying discrete sections to the members with largest change in the weight at initial stages of the search. This procedure substantially reduces the computational times. The difficulty of infeasibility of linearized problems during BBM is encountered and a procedure to overcome it is discussed. It is noted that all the design code constraints cannot be imposed with this procedure due to the selection of the linearization variables for the problem. The approach, however, does not require the standard sections to be ordered in any way, nor does it require any approximate relationships between the section properties.

Balling and Fonseca (1989) extend LBB strategy of Hager and Balling (1988) to discrete variable optimization of 3D steel frames. Five section properties are used as design variables instead of the three for 2D frames. They are : area, two moments of inertia and two section moduli.

Three procedures have been recently developed by Huang and Arora (1996 b) for design of planar steel frames using the AISC standard sections. The procedures consist of two phases. In Phase I, multiple design variables (dimensions of the cross section) are used and the problem is formulated as a standard NLP; i.e., Multiple Design Variable Formulation 2 of Section 3.(2) is used. In the first strategy, an SQP method is first used to find a continuous solution. A candidate section set (a subset of the AISC table) for each member is created based on the continuous solution. This is done to reduce the candidate discrete designs for Phase II. In Phase II, a genetic algorithm (based on Huang and Arora 1996 a) is used to solve the discrete variable optimization problem. An integer variable (the section number) is used as the sole design variable for each steel member; i.e., Single Design Variable Approach 3 of Section 3.(1) is used.

The second procedure is similar to the first one except that simulated annealing (based on Huang and Arora 1996 a) is used for discrete variable optimization in Phase II.

In the third procedure, an SQP method is also used to find a continuous solution for the four design variable formulation, as for the foregoing two procedures. Then a candidate section set is created for each member such that the moment of inertia, section area and section modulus are within 5% of the ones for the continuous solution. In Phase II, the problem is formulated using the moment of inertia as the sole discrete design variable for a member; i.e., Single Design Variable Formulation 2 of Section 3.(1) is used. Relationships of the form given in Eqs. (1) are used at this stage. In these relationships, the values for the constant r_i are calculated using the section dimensions corresponding to the continuous solution of Phase I. The allowable discrete values for the moment of inertia for each member are specified by the selected set of sections for the member. Each selected section set is arranged in the ascending order of values for the moment of inertia. The problem is now solved using a branch and bound method (based on Huang and Arora 1996 a) where the discrete variables can have non-discrete values during the solution process, as explained in Section 4.(2). After each local minimization, members are assigned discrete sections from the selected set according to the values of the moment of inertia. At the end of the branch and bound method, when a discrete solution has been obtained, the allowable section set is updated based on the final values for the moment of inertia, section modulus and the section area of each member. Then, the BBM is repeated to obtain a new discrete solution. The process is continued until the solution cannot be improved further. One drawback of this approach is that all the design code constraints cannot be checked. A remedy for this drawback would be to use SA or GA during Phase II along with Single Design Variable Formulation 2 of Section 3.(1).

(4) Rounding-off methods

The simplest and fastest way to obtain a discrete solution is to round-off or round-up all of the values of the design variables obtained at the continuous optimum. The problem must be formulated and solved with continuous design variables, such as the Single Design Variable Formulation 1 or the Multiple Design Variable Formulation 2 of Section 3. Then the optimum design variables are rounded to their nearest discrete values. The rounded-up solution often produces a more conservative design, especially for structural design problems. However, in some cases, the resulting discrete solution may violate some of the constraints.

Another approach would be to increase only some variables to their upper discrete neighbors and decreased others to their lower neighbors. The main difficulty with this approach would be the selection of the variables that can be increased or the variables that can be decreased.

Huang and Arora (1995, 1996 a) describe a dynamic rounding-up method which increases only one variable to its upper discrete neighbor at a time. The selected variable is then fixed at the discrete value and the problem is optimized again, allowing other variables to change. This process is repeated until all variables are selected and fixed to discrete values. This method still does not guarantee a feasible discrete solution but the cost function value is usually smaller than a simple rounding-off method.

Al-Saadoun and Arora (1989) have also presented an approach to optimize framed structures using AISC sections. In their approach, four design variables are chosen for each I-section-flange width and thickness, height of the section and web thickness. The problem is formulated as a continuous variable optimization problem to minimize the weight. Thus Multiple Design Variable Approach 2 is used. The problem is solved using a standard nonlinear programming algorithm, such as the SQP method. Once the final solution is obtained, each member is selected from the available ones using one of the following two criteria : (1) selection based on optimum depth and section modulus, and (2) selection based on optimum section modulus and minimum area of cross-section.

A similar approach is suggested in Arora (1989) for optimal design of trusses. The discrete member selection process is dynamic and works as follows : Once a continuous solution is obtained, a member that gives the least penalty for the cost function due to discrete specification, is selected from the available sections. The problem is then re-optimized using the continuous variable method while keeping the selected members as fixed. The process is continued until all members have been selected from available sections. The sensitivity of the cost function to design variables is used to calculate this penalty for the cost function. Chan *et al.* (1995) also use a similar procedure for optimal design of tall steel building frameworks. The Single Design Variable Formulation 1 of Section 3.(1) is used throughout. Initially, a continuous optimal solution is obtained using an optimality criterion method. Then, the dynamic rounding-up procedure along with the optimality criterion approach are used to obtain a discrete solution.

6. DISCUSSION AND CONCLUTIONS

This paper contains a review of optimization methods for mixed variable nonlinear programming problems with linked discrete variables. Design of steel structures using commercially available sections is one of the major application areas where such problems are encountered. These problems are more difficult to solve due to a large number of design variables and the large CPU time required for many structural analyses. While it is difficult to recommend any one method over the others, procedures with the following restrictions are often acceptable : (1) the solution found may not be a true optimal solution, (2) small constraint violation is acceptable, (3) user may have to spend extra effort to simplify the analysis part, (4) the method may be suitable for only a restricted class of problems, and (5) large CPU time may be needed.

There are a few approaches that use the section number (an integer variable) as the sole design variable for each member. These are : integer gradient method, simulated annealing and genetic algorithms. These are the most general approaches because all the design code constraints can be imposed explicitly. However, they are also the most time consuming ones because they require a very large number of structural analyses. Approximations can be used to speed-up the computational process but that will loose robustness of the methods. Thus the approaches appear to be suitable for small scale problems only. One advantage of the approaches is that they are highly suitable for parallel computers which may alleviate the limitation of large CPU times.

Many approaches use two phases in their solution process. In Phase I, the problem is formulated and treated as a continuous variable problem. Single or multiple design variables for each member are possible in these formulations. Depending on the formulation, it may or may not be possible to impose all the design code constraints. The formulation in this phase results into a standard NLP problem which can be solved using any one of the many available algorithms. In Phase II, the solution from Phase I is some how used to define a manageable discrete variable problem. This has been usually done by restricting the number of allowable discrete sections for each member. The problem is then solved using one of the discrete variable optimization methods, such as the branch and bound method, simulated annealing, genetic algorithms, etc.

Another promising approach is to use a dynamic member assignment process while performing several continuous variable optimizations. In such a process, the problem is formulated with continuous design variables, such the section dimensions. All the design code constraints can be imposed in such a formulation. The problem is solved using a standard NLP algorithm. Then, some members are assigned available sections that give least penalty for the cost function. The problem is re-optimized for the remaining design variables using the standard NLP algorithm.The process is thus repeated until all the members have been assigned available sections.

In conclusion, it appears that no one approach can be declared as ideal at the present time for all practical applications. Research needs to be continued to develop and evaluate computational strategies that are effective for optimal design of this important class of problems.

ACKNOWLEDGEMENTS : We would like to acknowledge partial support for this research under the project, "Design Sensitivity Analysis and Optimization of Nonlinear Structures," from the NSF of USA, Grant No.CMS-93-01580. We would also like to thank Professor Rick Balling of Brigham Young University, Professor Kunihito Matsui of Tokyo Denki University, and Professor Don Grierson of University of Waterloo for reading a draft of the paper and providing us with their suggestions to improve the presentation.

REFERENCES

1) Aarts, E. and Korst, J. (1989), *Simulated Annealing and Boltzmann Machines : A Stochastic Approach to Combinatorial Optimization and Neural Computing*, Wiley-Interscience Series in Discrete Mathematics and Optimization, John Wiley and Sons, New York.
2) Amir, H. M. and Hasegawa, T. (1989), "Nonlinear Mixed-Discrete Structural Optimization," *Journal of Structural Engineering*, ASCE, Vol.115, No.3, pp.626-646.
3) Al-Saadoun, S. S. and Arora, J. S. (1989), "Interactive Design Optimization of Framed Structures," *Journal of Computing in Civil Engineering*, ASCE, Vol.3, No.1, pp.60-74.
4) Arora, J. S. (1989), *Introduction to Optimum Design*, McGraw Hill Book Co., New York, pp.491-495.
5) Arora, J. S. (1990), "Computational Design Optimization : A Review and Future Directions," *Structural Safety*, An International Journal, Vol.7, pp.131-148.
6) Arora, J. S., Huang, M. W. and Hsieh, C. C. (1994), "Methods for Optimization of Nonlinear Problems with Discrete Variables : A Review," *Structural Optimization*, Vol.8, pp.69-85.
7) Barthelemy, J.-F. M. and Haftka, R. T. (1993), "Approximation Concepts for Optimum Structural Design-A Review," *Structural Optimization*, Vol.5, No.3, pp.129-144.
8) Balling, R. J. (1991), "Optimal Steel Frame Design by Simulated Annealing," *Journal of Structural Engineering*, ASCE, Vol.117, No.6, pp.1780-1795.
9) Balling, R. J. and Fonseca, F. (1989), "Discrete Optimization of 3D Steel Frames," *Computer Utilization in Structural Engineering*, ASCE, New York, N. Y., pp.458-467.
10) Cameron, G. E., Xu, L., and Grierson, D. E. (1991), "Discrete Optimal Design of 3D Frameworks," in *Electronic Computation*, Proceedings of the Tenth Conference, O. Ural and T-L. Wang, Eds., American Society of Civil Engineers, New York, pp.181-188.
11) Cella, A. and Logher, R. D. (1971), "Automated Optimum Design from Discrete Components," *Journal of Structural Engineering*, ASCE, Vol.97, No. 1, pp.175-189.
12) Chan, C.-M. (1992), "An Optimality Criteria Algorithm for Tall Steel Building Design Using Commercial Standard Sections," *Structural Optimization*, Vol.5, pp.26-29.
13) Chan, C.-M., Grierson, D. E. and Sherbourne, A. N. (1995), "Automatic Optimal Design of Tall Steel Building Frameworks," *Journal of Structural Engineering*, ASCE, Vol.121, No.5, pp.838-847.
14) Elwakeil, O. and Arora, J.S. (1995), "Methods for Finding Feasible Points in Constrained Optimization," *AIAA Journal*, Vol.33, No.9, pp.1715-1719.
15) Fleury, C. (1979), "Structural Weight Optimization by Dual Method of Convex Programming," *International Journal for Numerical Methods in Engineering*, Vol.14, pp.1761-1783.
16) Geoffrion, A. M. (1967), "Integer Programming by Implicit Enumeration and Balas' Method," *Society of Industrial and Applied Mathematics Reviews*, Vol.9, No.2, pp.178-190.
17) Glankwahmdee, A., Liebman, J. S. and Hogg, G. L. (1979), "Unconstrained Discrete Nonlinear Programming," *Engineering Optimization*, Vol.4, pp.95-107.
18) Goldberg, D. E. (1989), *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA, U.S.A.
19) Gomory, R. E. (1960), "An Algorithm for the mixed Integer Problem," *Report No. P-1885*, The Rand Corporation, Santa Monica, CA.
20) Grierson, D. E. and Lee, W. H. (1984), "Optimal Synthesis of Steel Frameworks Using Standard Sections," *Journal of Structural Mechanics*, Vol.12, No.3, pp.335-370.
21) Grierson, D. E. and Lee, W. H. (1986), "Optimal Synthesis of Frameworks Under Elastic and Plastic Performance Constraints Using Discrete Sections," *Journal of Structural Mechanics*, Vol.14, No.4, pp.401-420.
22) Grierson, D. E. and Cameron, G. E. (1989), "Microcomputer-Based Optimization of Steel Structures in Professional Practice," *Microcomputers in Civil Engineering*, Vol.4, pp.289-296.
23) Hager, K. and Balling, R. J. (1988), "New Approach for Discrete Structural Optimization," *Journal of Structural Engineering*, ASCE, Vol.114, No.5, 1120-1134.
24) Hua, H. M. (1983), "Optimization of Structures of Discrete-Sized Elements," *Computers and Structures*, Vol.17, No.3, pp.327-333.
25) Huang, M. W. (1995), "*Algorithms for Mixed Continuous-Discrete Variable Problems in Structural Optimization*", Ph. D Dissertation, Civil and Environmental Engineering, The University of Iowa, Iowa City, IA, U.S.A.
26) Huang, M. W. and Arora, J. S. (1995), "Engineering

Optimization with Discrete Variables," *Proceedings of the 36th AIAA SDM Conference*, New Orleans, LA, April 10-12, pp.1475-1485.

27) Huang, M. W. and Arora, J. S. (1996 a), "Optimal Design with Discrete Variables : Some Numerical Experiments," *International Journal for Numerical Methods in Engineering*, to appear.

28) Huang, M. W. and Arora, J. S. (1996 b), "Design of Steel Structures Using Standard Sections," *Technical Report* No. ODL-96.04, Department of Civil and Environmental Engineering, The University of Iowa, Iowa City, IA.

29) Kincaid, R.K. and Padula, S.L. (1990), Minimizing Distortion and Internal Forces in Truss Structures by Simulated Annealing, *Proceedings of the 31st AIAA SDM Conference*, Long Beach, CA, American Institute of Aeronautics and Astronautics, Washington, D. C., pp.327-333.

30) Lai, Y-S. and Achenbach, J.D. (1973), "Direct Search Optimization Method," *Journal of Structural Engineering*, ASCE, Vol.98, No.1, pp.19-31.

31) Liebman, J. S., Khachaturian, N. and Chanaratna, V. (1981), "Discrete Structural Optimization," *Journal of Structural Engineering, ASCE*, Vol.107, No.ST11, pp.2177-2197.

32) Lin, C.-Y. and Hajela, P. (1992), "Genetic Algorithms in Optimization Problems with Discrete and Integer Design Variables," *Engineering Optimization*, Vol.19, pp.309-327.

33) May, S. A. and Balling, R. J. (1991), "Strategies which Permit Multiple Discrete Section Properties per Member in 3D Frameworks," in *Electronic Computation*, Proceedings of the Tenth Conference, O. Ural and T-L. Wang, Eds., American Society of Civil Engineers, New York, pp.189-196.

34) May, S. A. and Balling, R. J. (1992), "A Filtered Simulated Annealing Strategy for Discrete Optimization of 3D Steel frameworks," *Structural Optimization*, Vol.4, pp.142-148.

35) Reinschmidt, K. F. (1971), "Discrete Structural Optimization," *Journal of Structural Engineering*, ASCE, Vol.97, No.1, pp.133-156.

36) Sugimoto, H. (1992), "Discrete Optimization of Truss Structures and Genetic Algorithms," *Proceedings of the Korea-Japan Joint Seminar on STRUCTURAL OPTIMIZATION, Seoul, Korea*, pp.1-10.

37) Toakley, R. (1968), "Optimum Design Using Available Sections," *Journal of Structural Engineering*, ASCE, Vol.94, No.5, pp.1219-1241.

38) Tseng, C.H., Wang, L.W., and Ling, S.F. (1995), "Enhancing Branch and Bound Method for Structural Optimization," *Journal of Structural Engineering*, ASCE, Vol.121, pp.301-306.

39) Vanderplaats, G. N. and Salajegheh, E. (1989), New Approximation Method for Stress Constraints in Structural Synthesis," *AIAA Journal*, Vol.27, No.3, pp.352-358.

40) Venkayya, V. B. (1971), "Design of Optimum Structure," *Computer and Structures*, Vol.1, Nos.1/2, pp.265-309.

(Received May 10, 1996)