A combinatorial optimality of shear element allocation
Y. Kohama, a T. Takada a & A. Miyamura b
a Department of Architecture, Mie University, Tsu, Japan 515
b Nagoya City Women’s College, Chikusa, Nagoya, Japan 464

Abstract
One of the important subjects in AI technology is development of efficient searching technique for goals in enormous database frequently stemmed from combinatorial discrete optimality. Herein, general direction of heuristics is discussed, and comparison of efficiency is numerically made on a combinatorial discrete optimality of shear element in a 3D frame against torsion about a vertical axis.

1. Introduction
Through the development of AI technology since the late fifties the effective searching technique was obviously a main subject to approach. Particularly, when the data-base consists of combinatorial elements, its space increases so exponentially that search for optimality becomes significantly laborious because of its nondifferentiability (Polak[1]). Such combinatorial searching subjects as the traveling salesman problem (Kernighan[2]) can be found not only in AI technology but in engineering design. Several searching techniques are developed (Padberg[3], Johnson[4]), including the branch-and-bound method related to the dynamic programming as a generalized technique (Ibaraki[5]). When dealt with such nondifferentiable optimality, the exhaustive enumeration with the generate-and-test procedure is inevitably required. This is partly due to a lack of information of extrapolation on a searching space. It is laborious to describe algorithm of exhaustive enumeration in procedural language such as FORTRAN. On the contrary, declarative language including Prolog can handle it directly. The transitivity
and inheritance inferences extend the searching efficiency so strongly that combinatorial problems are more practically approached.

2. **Direction of heuristics**

Engineering design problem requires feasibility instead of the optimality partly because of lack of rigorous constraints relating to feasible sets of variables. Practically such sets may become enormous, to which the generate-and-test of constraints is implemented. Then the objective function makes an enormous data-base for the design goal. This implementation is accomplished by the exhaustive enumeration resulting in explosive generate-and-tests for a large number of either the state variable or searching breadth. The conventional scenario consists of searching approaches such as the breadth-first, the depth-first or the best-bound approach that should include ingenious technique or heuristics to decrease their searching space effectively. Apart from the system performance of the generate-and-test the direction of such heuristics can be categorized as: (a) Problem-oriented heuristics: Additional constraints or their revise are expected to prune futile alternatives. The revise or relaxation of objective function is included. It is necessary to avoid numerical operations as far as possible, which is a side-effect to searching languages such as Prolog. It is the best breakthrough that with scrutinizing the particularity of the problem some human intuition surely contributes to such constraints or evaluation functions frequently with a result of drastic effectiveness. (b) Application of branch-and-bound method including DP: Practical implementation of discrete optimality requires any of enumeration approaches such as the branch-and-bound method, the dynamic programming technique(DP) and the exhaustive enumeration. The branch-and-bound method versatile to diversified problems is summarized as: First, initialization of both the tentative value of evaluation function equal to infinitive, \( z = \infty \), and the active partial problem becomes the original problem, \( P_o \), to solve. Second, searching for a new active partial problem, \( P_i \), otherwise ending for no more active one. Third, testing \( P_i \) and a new \( z \) is obtained for a upper bound of feasible solutions, \( \{ z \} \). Lastly, branching of descendent partial problems to add the active space and searching. When the problem is translated into the optimality through a multi-stage network from combination of state variables, the dynamic programming technique(DP) can be applied with exponential decrease in the order of \( 1/p^N \) (\( p \): breadth, \( N \): depth of a network). Such decrease can be accomplished by the recurrence formulation based upon the principle of op-
timality. (c) Application of qualitative reasoning: The qualitative analysis demands to divide the over-all system into sub-systems whose formats of variable are defined by landmarks of both constraints and design variables.

3. Description of discrete optimality

General formulation of discrete optimality under crisp constraints can be summarized as follows:

\[ K_{Z,N} = \max_{L_N} \left\{ \sum_{j=1}^{N} k_{Z}(L_j) \right\} \]

subject to \( L_j \in X_j = \{L_j | g_j(L_j) \leq S_{aj} \} \) \( j = 1, 2, \ldots, N \)

where \( L_j \) means a discrete design variable, \( L_j = \{L_1, L_2, \ldots, L_j\} \), \( k_{Z,j}(L_j) \), the \( j \)-th objective component, \( X_j \), the admissible domain of \( L_j \), \( g_j(L_j) \), a function of constraints and \( S_{aj} \), the admissible vector of the \( j \)-th constraint, respectively. When crisp constraints become fuzzified, Eq.(1) is written by

\[ K_{Z,N} = \max_{L_N} \left\{ \sum_{j=1}^{N} k_{Z}(L_j) \right\} \]

subject to \( L_j \in D_j = X_1 \cap X_2 \cap \cdots \cap X_j \),

\[ X_j = \{L_j | S_j \equiv g_j(L_j) \leq S_{aj} \} \] \( j = 1, 2, \ldots, N \).

where \( D_j \) and \( X_j \) are fuzzy admissible domains and \( S_j \), the state variable vector of constraint, respectively. Furthermore, by fuzzy membership functions, \( \mu_{C_j}(S_j) \), Eq.(2) becomes the \( \alpha \)-level optimal problem as:

\[ K_{Z,N} = \max_{L_N} \left\{ \sum_{j=1}^{N} k_{Z}(L_j) \right\} \]

subject to \( L_j \in X_j(\alpha) = \{L_j | g_j(L_j) \in C_j(\alpha) \} \),

\[ C_j(\alpha) \equiv [a_{C_j}(\alpha), b_{C_j}(\alpha)] = \{S_j | \mu_{C_j}(S_j) \geq \alpha \} \] \( j = 1, 2, \ldots, N \)

where \( C_j(\alpha) \) means the \( \alpha \)-level admissible domain of state variables, \( S_j \). Eq.(3) is another crisp optimality similar to Eq.(1). Since Eqs.(1) and (3) possess common property to the resource allocation problem, the present heuristics(b) such as the branch-and-bound method or DP can be applied effectively. For DP procedure Eq.(3) becomes with lattice of \( S_j \) as:

\[ K_{Z,k}(S_k) = \max_{L_k} \left\{ \sum_{j=1}^{k} k_{Z}(L_j) | S_{k-1} \in C_{k-1}(\alpha) \right\} \]

\[ K_{Z,max} = \max_{S_N \in C_N(\alpha)} K_{Z,N}(S_N) \]
Furthermore, the above first line can be expressed in recursive form.

\[
K_{Z,0}(S_0) = F_{K_Z}
\]

\[
K_{Z,k}(S_k) = \max_{L_k} \left\{ K_{Z,k-1}(S_{k-1}) + k(Z)(L_k) \mid S_{k-1} \in C_{k-1}(a) \right\}
\]  

(5)

\(L_k\) is searched to maximize the righthand sides of Eq.(5) satisfying the given constraints of possible state variables, \(S_k\), progressively from at the first stage, \(K - 1\) until \(k = N\), in consequence with \(K_{Z,\text{max}}\).

By the branch-and-bound method Eq. (3) can be divided into the following partial problems by assigning an initial value, \(L_i^*\), to the design variable, \(L_N\):

\[
P_0 : K_{Z,N} = \max_{L_N} \left\{ \sum_{j=1}^{N} k(Z)(L_j) \mid L_j \in X_j(\alpha) \right\} = \max_{i=1,m} K_{Z,N}[i]
\]

\[
P_i : K_{Z,N}[i] = k(Z)(L_i^*) + \max_{L_{N-1}} \left\{ \sum_{j=1}^{N-1} k(Z)(L_j) \mid L_j \in X_j(\alpha) \right\} \quad i = 1, 2, \ldots, m
\]

subject to \(L_N \in X_{N,i}(\alpha) = \{X_N(\alpha) \mid L_N = L_i^* \in Z\}

(6)

where \(Z\) means the basic design variable space and \(L_i^*\), its element whose number is \(m\), respectively. Eq.(6) implies that the discrete optimality problem (original problem), \(P_0\), is finished, if all of partial problems, \(P_i, i = 1, 2, \ldots, k\) can be solved. Hence the original problem can be substituted by the \(k\)-th branch tree due to successive branch operations. Particularly, when dealt with binary-value parameter, each partial problem is derived by assigning any element of \(L\) to \(L_k = 1\) or \(0\), thus called as the binary-branch tree. The branch-and-bound method tells that when fail occurs by test for any solution generated from each partial problem, then further branch operation is not required with a result of decrease of combinatorial generate-and-test. This can be attained by the bound criteria: (a) When the optimal solution is obtained from a partial problem, \(P_i\), it is not necessary to extend further branch operation. (b) If a partial problem cannot provide optimal solution of the original problem, it is not necessary to extend further branch operation.

4. Optimal allocation of shear elements

Herein a case study is discussed on the optimal allocation of shear elements in an eccentric rigid frame subjected to horizontal loading subsequently with torsional moment. Two objective functions at individual floor and in the story direction should be optimized simultaneously under structural, architectural and construction constraints.
For the $s$-th story,

$$\begin{align*}
\mathcal{P}_0: \quad & K_{Z,N_w} = \max_{L_{N_w}} \left\{ \sum_{L_j \in L_{N_w}} k_Z(L_j) \right\} \\
\text{subject to} \quad & g(L_{N_w}) \in S_{as}, \quad L_{N_w} \subset W_s = \{L_1, L_2, \ldots, L_N\}
\end{align*}$$

(7)

where $L_j$ denotes a design variable that defines, herein, an element allocation by an binary function, $L_{N_w}$, a partial set of the allocation space, $W_s$, whose elements are $N_w$, $K_{Z,N_w}$, the objective function of optimal torsional rigidity from combination of $N_w$ elements at $s$-th story and $k_Z$, an object component or an individual torsional rigidity, respectively. $g(L_{N_w})$ means constraints from architectural, structural, constructional requirements. $S_{as}$ is the admissible space, respectively.

Branch operation: Since branch parameters, herein, stem from element allocation array, the corresponding optimality problem, $\mathcal{P}_0$, can be decomposed into $N$ partial problems, $\mathcal{P}_i$, $i = 1, 2, \ldots, N$. Accordingly,

$$\begin{align*}
\mathcal{P}_i: \quad & K_{Z,i,N_w} = k_Z(L_i^*) + \max_{L_{N_w-1}} \left\{ \sum_{L_j \in L_{N_w-1}} k_Z(L_j) \right\} \\
\text{subject to} \quad & g(L_{N_w}) \in S_{as}, \quad L_{N_w} \subset W_s,i = \{W_s| L_i = L_i^* \in W\}
\end{align*}$$

(8)

$L_i^*$ means an assigned allocation array and $W_{s,i}$, the candidate allocation set with assigned $L_i$ from $W_s$, respectively. Similarly, the problem, $\mathcal{P}_i$, can be decomposed into $(N - i)$ partial problems, $\mathcal{P}_{ij}, j > i$, subsequently into $\mathcal{P}_{ijk}, j > i$, ensures to avoid overlaps of allocation. The resulting multi-branch tree has depth of $N_w$. The problem, $\mathcal{P}_i$, without $N_w$ elements allocations should be immediately terminated.

Bound operation: Subsequently, a series of bound operations is implemented by either the lower bound test or the dominance test with effective search algorithm for admissible solutions as near as possible to the optimal solution. Since the objective function for optimality is the sum of torsional rigidity of shear elements in Eq.(7), it is preferable to implement branch operation by both selection and assignment in order of the amount of $k_Z(L_j)$. This corresponds to the dominance test whose constraints depend upon individuality of the problem.

Optimization along story: By completion of this algorithm for the $s$-th story with each number of element allocation, $N_w = 1, 2, \ldots, N$, a set of optimal allocation, $L_{N_w}^*$, is obtained as follows:

$$W_s^* = \{L_1^*, \ldots, L_{N_w}^*, \ldots, L_N^*\}$$

(9)
For M-story frame, additionally, the entire element that sums each set of elements at a story should be minimized as:

\[ Q_0 : A_{\text{opt}} = \min_{W_1, \ldots, W_M} \sum_{k=1}^{M} A(W_k | W_{k-1}) \]

subject to \( g(W_k) \in S_{ak}, \ W_k \in W_k^k; \ k = 1, \ldots, M \)

where \( A_{\text{opt}} \) means the minimum weight of elements, \( A(W_k | W_{k-1}) \), the weight of the \( k \)-th story allocation, \( W_k \), under constraint of the weight of the \((k - 1)\)-th story allocation, \( W_{k-1} \), respectively. Thus, design parameters, \( W_1, \ldots, W_M \), should be minimized by Eq.(10), which is also searched by the branch-and-bound method.

Constraints: The above technique can be applied to both objective functions under the following constraints.

strength constraint: \( Q_{si} \geq Q_{di} \) \( (11) \)

where \( Q_{si} \) means the \( i \)-th story shear resistance, which consists of strength of both shear elements and frame based on an appropriate plastic analysis, and \( Q_{di} \), the corresponding design shear force, respectively.

displacement constraints: \( (1 - \epsilon_{ki})k_{ti} \leq K_{qi}/K_{q1} \leq (1 + \epsilon_{ki})k_{ti} ; \ i \geq 2 \)

\[ K_{q1} \geq C_B W/\delta_a ; \ i = 1 \] \( (12) \)

\( \epsilon_{ki} \) means allowable deviation, \( C_B \), design base shear coefficient, \( W \), total weight of the building and \( \delta_a \), allowable drift, respectively. \( k_{ti} = C_i W_i/W \) is the prescribed goal stiffness coefficient.

torsional constraint: \( \epsilon_{ri}/\tau_{ei} \leq R_{ai} \) \( (13) \)

where \( \tau_{ei} \) means a radius of torsional stiffness, \( \sqrt{K_{qi}/K_{q1}} \).

vertical constraint of shear element: \( \sum_{i=1}^{N_W} iK_{Wj} \leq \kappa K_{q1} \); \( j = 1, \ldots, N_W \) \( (14) \)

where \( iK_{Wj} \) means element stiffness, and \( \kappa \), a constant of both dead load and element shape determined by up-lift constraint.

allocation constraints: continuity of elements along the height, constructional rules and architectural rules expressed by three categorized allocation flags of empty, removable and coerced shear walls.

5. Numerical example and discussion

On a 8-story rigid eccentric frame subject to torsion about a vertical axis, comparison between the branch-and-bound method and DP is accomplished
with emphasis on the number of generate-and-test during implementation of Prolog predicates which describe searching process for optimal allocation. Fig. 1 shows a typical floor plan with candidate allocation of elements. Fig. 2 shows a typical optimal allocation of shear elements in this 3D frame whose branch tree at the first floor is depicted in Fig. 3 for \( N_W = 8 \). Fig. 4 gives relations of the number of shear elements, \( N_W \), to the numbers of the generate-and-test by three methods; the exhaustive enumeration method that predicts theoretical upper limit; DP with two types of lattice; and the branch-and-bound method. The present DP under the breadth-first search for optimality through multi-stages corresponds to the exhaustive enumeration when through single stage. Consequently, the more coarse becomes lattice, the more decreases generation to considerable extent due to the breadth-first search. For \( N_W \leq 6 \) the branch-and-bound method gives no branch because of immediate bound due to predominance of the strength constraint by Eq. (11). Otherwise, the bound operation can sort torsional rigidities of element ensures swift search for optimality with a result in efficient pruning by the dominance test. In this example there are several optimal solutions satisfying Eqs. (7) to (14) with \( \text{opt} A_W = 101.5 \), which can be categorized into either dissimilar allocation with same \( S_{N_w} \) or similar allocation with different \( S_{N_w} \). The former allocations appear due to the common structural properties. This implies such optimal allocations can be grouped as shown in Fig. 4 by the dotted line which is considerably smaller than the exhaustive branch-and-bound.

References
5. T. Ibaraki [1991], Efficient Searches and Their Limitation, J. Japanese Soc. Artificial Intelligence, 6-1 pp.15-23
C: coerced element
[ ]: removal element

\[ \sum A_f = 49 \times 12 = 588(m^2) \]

\[ W_{\text{total}} = 588 \times 1.2 \times 8 \]

\[ C = 5645(t) \]

Fig.1 8 story frame

Fig.2 An optimal allocation

Fig.3 Branching-tree at 1-st story

Fig.4 Comparison between heuristics