NASA Technical Memorandum 83248

(NASA-TM-83248) A LINEAR DECOMPOSITION METHOD FOR LARGE OPTIMIZATION PROBLEMS. BLUEPRINT FOR DEVELOPMENT (NASA) 61 p HC A04/MF A01 CSCL 01C

N82-22245

Unclas
G3/05 09577

A LINEAR DECOMPOSITION METHOD FOR LARGE OPTIMIZATION PROBLEMS —BLUEPRINT FOR DEVELOPMENT

JAROSLAW SOBIESZCZANSKI-SOBIESKI

FEBRUARY 1982

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

TABLE OF CONTENTS

SUMMARY

The paper proposes a method for decomposing large optimization problems encountered in the design of engineering systems such as an aircraft into a number of smaller subproblems. The decomposition is achieved by organizing the problem and the subordinated subproblems in a tree hierarchy and: (1) minimizing the constraint violation in each subproblem using its local design variables while holding constant the higher level variables which figure as parameters in the subproblem minimization, (2) calculating the sensitivity derivatives of the subproblem minimum solution to the parameters, (3) using the derivatives to form a linear extrapolation of each subproblem minimum with respect to the higher level variables, (4) optimizing the system for its objective function and constraints, including the linear extrapolation of each subsystem constraint violation minimum in lieu of repeated subsystem minimizations, (5) repeating operations (1) through (4) until convergence is attained.

The decomposition is introduced for a two-level system and generalized to a multilevel case. A formalization of the procedure suitable for computer implementation is developed and the state of readiness of the implementation building blocks is reviewed showing that the ingredients for the development are on the shelf. The decomposition method is also shown to be compatible with the "natural" human organization of the design process of engineering systems. It is, therefore, viewed as an opportunity to bring mathematical rigor to that process without overturning its time-honored organizational structure. The method is also examined with respect to the trends in computer hardware and software progress to point out that its efficiency can be amplified by network computing using parallel processors. A few numerical examples drawn from the areas of structures and aircraft design are given to illustrate the salient conceptual points of the method. While the full validation of the method still remains, its presentation here provides a guide to its development and test applications, and an opportunity to elicit comments, critiques, and improvements to a basic concept.

LIST OF SYMBOLS

Variables

For some variables, notation is given in shorthand and longhand forms and upper and lower case, e.g., B, B(NID); G_i, g_i.

A	cross-sectional area
B, B(NID)	vector of behavior variables at node NID
CID	identification of a constraint vector
E, E(CID)	vector of equality constraints
E_j, e_j	element of $\{E\}$
F, F(V,H)	objective function
f	function in a general mathematical sense, also objective function
G, G(CID)	inequality constraint
G_j, g_j	element of $\{G\}$
H, H(NID)	vector of parameters
H_j, h_j	element of $\{H\}$
I	cross-sectional moment of inertia
K	stiffness matrix
NID	node identification in general, NID is a vector NID = $\{u,v\}$, where u is the node level number (u=1 is system level), v is the node position in its level counted from left (.e.g, fig. 5 and Table II)
$NQI(SSID_1,SSID_2)$	network interaction quantity representing influence of subsystem $SSID_1$ on subsystem $SSID_2$
P	cumulative measure of constraint violation (cumulative constraint, penalty term)
P_s	P for a node playing the role of a system
P_i	P for a node $SSID_i$ playing the role of subsystem

PID	same as NID, used whenever the node is discussed in its parent role
QI, QI(PID,SSID)	vector of interaction quantities representing influence of parent system PID on its subsystem SSID
RQI, RQI(PID,SSID)	vector of interaction quantities representing influence of a subsystem SSID on its parent system PID
SSID	same as NID, used whenever the node is discussed in its subsystem role. A subscripted form, e.g., $SSID_1$, $SSID_2$ is used to distinguish two different SSID vectors corresponding to two subsystems
V	vector of design variables in general, may denote X or Y or both
v_i	element of vector V
X, X(PID)	1. vector of design variables of a node playing the role of parent to its subsystems 2. vector of parameters in a subsystem optimization (see definition of H)
X_i	subscripted form of X, e.g. X_1 and X_2 used to distinguish vectors X for different nodes
x_i	element of vector X
x_r^i	element of vector X_i
Y, Y(SSID)	vector of design variables of a node playing the role of a subsystem
Y_i	subscripted form of Y, e.g., Y_1 and Y_2, used to distinguish vectors Y for different nodes
y_i	element of vector Y

Constants

k	number of the system level constraints
n	number of design variables
m	number of constraints in general
q	number of elements in vector $\{x_i\}$
s_2	number of subsystems in a multilevel system

δ denotes an increment of a variable, e.g.: δx_i

ϵ a small positive constant

Subscripts, Superscripts

B	initial value of P
e	extrapolated value
L	lower bound
i	as a subscript: position of design variable in a vector of variables; a general purpose subscript; as a superscript: identifies a subsystem to which the variable pertains
j	position of a constraint in a vector of constraints; a general purpose subscript
o	initial value from which one extrapolates
p	perturbed value
r	general purpose subscript for a vector element
q	value of P_t in iteration q
Q	number of decrements for reducing P_B to zero
s	system quantity
t	target value to which P is to be reduced

ronyms, Special Symbols

ØAC(ØAQ), ØAC	constraint operator (see subsection Operators, section FORMALIZATION...)
ØAQ((X,H),(QI,RQI,NQI)),ØAQ	interacton analysis operator, first parenthesis-input, second parenthesis-output
ØASW(OAW),ØASW	interaction analysis sweep operator
ØOBJ((X),(H,F)),ØOBJ	objective function operator
$ØOPT_m((H),(v,\phi)),ØOPT_m$	optimization operator (m=1, or m=2)
$ØSEN((H,V,\phi),(\partial V/\partial H_j,\partial\phi/\partial H_j)),ØSEN$	optimum sensitivity analysis operator
ØSUBSW(OOPT1,OSEN),ØSUBSW	subsystem elimination sweep operator

ØTER	procedure termination operator
PS	parent system
SS	subsystem
$SS_{u,v}$	subsystem identified by subscripts defined in SSID
STO	subject to
$\overline{\text{character}}$	overbar indicates the optimum value (constrained minimum solution)
$\widetilde{\text{character}}$	tilde indicates the system quantity
*,**	before and after one iteration

INTRODUCTION

The purpose of this report is to propose a method for decomposing a large optimization problem into a hierarchy of much smaller subproblems, and to provide a blueprint for development of a computer implementation for the method. Several, general purpose-oriented, decomposition schemes based on various sets of restrictive assumptions have been proposed. Some of them are described in ref. 1 in which numerous sources are also quoted. In addition, decomposition schemes specialized for structural optimization have also been introduced (e.g., ref. 2 and 3). However, there seems to be no scheme available that would be adequately general and efficient to meet the need for optimization of large multidisciplinary engineering systems. In absence of such a scheme, an attempt to perform such optimization using disciplinary analyses of a significant depth would saturate even the most advanced optimization software-hardware systems available today.

There is ample motivation for undertaking development of a scheme for optimization of large multidisciplinary engineering systems at this time. On one hand there is a need to bring mathematical optimization methods to bear on truly large engineering design problems, for example synthesis of aircraft as multidisciplinary engineering systems. On the other hand, there is an opportunity brought about by recent theoretical and computer technology developments. Some of these developments are: (1) approximate analysis and variable linking (e.g., ref. 4), (2) analytical generation of gradient information (e.g., refs. 4, 5, 6, 7, 8), (3) piecewise-linear optimization of nonlinear problems (e.g., refs. 9, 10), (4) reduction of the multitude of constraints to one cumulative constraint (e.g., ref. 8), (5) sensitivity analysis of optimum solutions (ref. 11), (6) development of distributed (network) computing involving relatively inexpensive mini and microcomputers linked with mainframe computers (ref. 12), and employing comprehensive data management systems (ref. 13).

It appears that the desired capability can be developed in a way described in this report. The report begins with an introduction of the linear decomposition principle in a narrative form, with a minimum of mathematics, to establish an understanding of the basic idea which is very simple. Next, a formalization of the basic idea into a procedure suitable for computer implementation is given including identification of all the building blocks, their state of readiness, and discussion of the development work required to make each block ready for integration in the procedure. Specifications for the verification testing are presented to establish testing as one of the critical stages of the development. Finally, compatibility of the proposed method with the organization of a typical design office is discussed to show that the method supports the natural tendency of engineers to work in concurrently operating specialty groups, each dealing with a limited part of the problem, and that it can exploit the parallel processing capability of a distributed (network) computing.

The paper offers no numerical examples for execution of the entire method because such complete examples will require completion of the very development outlined herein. However, partial examples are used to illustrate salient points of the approach, and execution of complete examples is specified as a verification requirement. In particular, the proposed approach is documented here to provide a blueprint for development and evaluation, and to serve as a baseline for critique, modifications, and improvements.

KEY IDEA FOR THE DECOMPOSITION

The optimization problem under consideration is a general nonlinear mathematical programing problem of finding a vector of design variables $\{V\}$ that minimizes an objective function $F(V)$ and satisfies constraints $g_j(V)$. In a standard notation:

$$\min_{\{V\}} F(V)$$

$$\text{STO (subject to)} \quad g_j(V) \leq 0, \; j = 1 \rightarrow m; \tag{1}$$

Recognizing that the number of design variables and constraints is large, one asserts that it may be advantageous to make the problem more tractable by solving it as an assembly of separate but coupled smaller problems, that is to solve it using a decomposition approach, instead of the "all-in-one" approach.

The basic idea for decomposition, termed a linear decomposition for reasons that will soon become apparent, will be first introduced using a framework structure as an example. Next, it will be presented at a completely general level of abstraction beginning with a system whose design variables form a two-level hierarchy and, subsequently, generalizing to a multilevel hierarchy.

Introductory Example

A framework structure shown in figure 1 is a particular example of a two-level system. The framework is made up of three I-beams, each defined by 6 cross-section design variables (see section A-A in fig. 1) hence 18 variables need to be considered in optimization of the entire structure for minimum mass under a static load. However, the problem can be decomposed into smaller subproblems because analysis of the framework for displacements, and internal forces shown in fig. 2, can be carried out with sufficient accuracy for engineering purposes knowing only the cross-sectional area A and moment of inertia I_y for each beam; in fact only I_y is required if beams are slender. The detailed cross-sectional dimensions (design variables) shown in the inset in figure 1, do not enter that analysis.

This observation suggests a decomposition of the "all-in-one" optimization problem of 18 variables into 4 smaller problems: 3 problems at a component (subsystem) level each entailing 6 design variables to determine the detailed cross-sectional dimensions, and 1 problem of 6 (or 3) system level variables representing A and I_y (or I_y only) for each beam. The resulting iterative procedure may be summarized in the following sequence of steps shown separately for each of the two levels:

System level - whole framework	Component (subsystem) level - each separate beam
1. Define loads. 2. Define displacement constraints.	1. Initialize detailed dimensions. 2. Compute A, I_y for each beam. 3. Move to the system level.
4. Analyze the framework to compute its displacements and the end forces (N, M, T, fig. 2) on each beam. Compute derivatives of these quantities with respect to the A, I_y of each beam. 5. Move to the component level.	
	6. For beam 1, hold constant the end forces N, M, T and the values A and I_y. Analyze the beam to evaluate its constraints such as stress and local buckling. Form a single measure of the constraint violation using, for example, an exterior penalty function. Optimize 6 cross-sectional dimensions as subsystem design variables to minimize the measure of constraint violation as an objective function subject to minimum gage and other side constraints, including equality constraints on A and I_y. The equality constraints assure that the beams A and I_y computed from the cross-sectional dimensions are equal to those prescribed at the system level. 7. For optimized beam, compute derivatives of the minimized measure of the constraint violation and the subsystem design variables with respect to the constants: N, M, T and A, I_y.

8. Repeat from 6 for ~~beams~~ 2 and 3.

9. Move to the system level.

10. Approximate the minimized measures of the constraint violation in each beam as linear functions of 6 quantities A and I_y by a linear Taylor expansion, using the derivatives computed in step 7. In this expansion each of the end forces N, M, T is also approximated as a linear function of all 6 quantities A, and I_y using derivatives computed in step 4.

11. Optimize 6 system level variables A, I_y, to minimize structural mass subject to:

    a) framework displacement constraints approximated as functions of A and I_y by linear Taylor expansion using derivatives computed in step 4.

    b) constraints requiring that the minimized measure of the constraints violation in each beam be reduced by a predetermined decrement.

    c) move limits on the variables A and I_y to protect accuracy of the linear Taylor expansions and to account for side constraints of the subsystem design variables. The latter are approximated as functions of A and I_y by a linear Taylor expansion using derivatives computed in step 7.

    d) side constaints on A and I_y.

12. Repeat from 4 with the system level design variables A and I_y obtained in step 11, and the corresponding approximate sub-system design variables estimated by a linear extrapolation as described in step 11c. Terminate when: a) the framework displacements are within constraints. b) the minimized measure of constraint violation for each beam is reduced to at least zero. c) no further reduction of the framework mass appears possible.	

The procedure differs from refs. 2 and 3 in the formulations of the subsystem and system level optimizations (steps 6 and 11) and the way the two levels are linked by means of the optimum sensitivity analysis (steps 7 and 10).

The framework example given here in a purely descriptive manner is repeated in Appendix A, cast in the formulation of the general decomposition method presented herein. To provide more examples for the salient points of the method, Appendix A describes also two other applications, including a case of an aircraft viewed as a multidisciplinary system. The method will now be introduced at an abstract level and frequent reference to Appendix A is recommended as the discussion unfolds.

Two-Level System Optimization

A two-level system, that does not have to be a structure, is depicted in figure 3 by a Venn diagram showing a system with a number of subsystems connected to it, so that the variables $X = \{X_1, X_2, \ldots X_i \ldots\}$ are system design variables and variables Y_i are subsystem design variables. By definition, X_i are those variables whose values must be known to analyze the system to obtain the relevant behavior variables and the objective function, and Y_i are those variables whose values are not needed to perform that analysis, although they are needed for analysis of subsystem i.

An optimization problem posed for the system together with its subsystems calls for finding values of the design variables $\{X\}$ and $\{Y\}$ that minimize an objective function subject to constraints on the behavior variables in the system and all the subsystems. In principle, the problem can be solved by collecting all elements of $\{X\}$ and $\{Y\}$ in one vector of design variables such as vector $\{V\}$ in eq. 1, to be manipulated by an appropriate optimization algorithm coupled in an iterative loop with an analysis algorithm which evaluates the objective function and all the constraints.

In contrast to this "all-in-one" approach, a decomposition approach separates the system and subsystem optimizations into an iteratively executed sequence of steps, each described in one of the following subsections.

<u>System analysis</u>.- For the values of system variables X_i prescribed by an optimization algorithm and other required inputs, the system is analyzed. The analysis outputs the system constraints, objective function, and the information needed as inputs for analyses of each subsystem.

This implies a simplifying assumption that the system is strictly hierarchical so that information generated by system analysis satisfies input requirements for the subsystems below but the system analysis input does not depend on the outputs from the subsystem analyses, and, similarly, output from analysis of one subsystem is not needed as input for analysis of another subsystem. In other words, the influence of one component of the system on another is restricted to a top-down influence with no reverse and lateral influences allowed.

It is recognized that most engineering systems are networks and, consequently, the reverse and lateral influences do exist. Therefore, the simplifying assumption defined above is only temporarily used to keep the introduction of the basic concept concise. In the next section, the means to accommodate the reverse and lateral influences are defined, and it is explained in Appendix B how these influences can be included in the basic concept.

Subsystem optimizations.- When the system's analysis is completed, each subsystem "i" is optimized as a separate subproblem "i." In each subproblem "i," the variables X_i are kept constant and the design variables Y_i are sought to minimize an objective function. The subsystem objective function is not the same as the system's one. Instead, it is defined as a single measure of unsatisfaction of the constraints local to the subsystem "i." One such measure is a well-known formulation for the penalty term in the exterior penalty function (ref. 5):

$$P_i = \sum_j (\langle g_j \rangle)^2$$

$$\langle g_j \rangle = \begin{cases} g_j, \text{ if } g_j > 0. \\ 0.0, \text{ if } g_j \leq 0. \end{cases} \tag{2}$$

It represents the sum of the squares of violated constraints g_j^i in subsystem i and will be referred to as a cumulative constraint. Minimization of P_i is subject to upper and lower bound limits on elements of $\{Y_i\}$ and equality constraints are required to maintain a constant $\{X_i\}$. These equality constraints arise from the functional relationship

$$\{X_i\} = f(\{Y_i\}), \text{ or } f(\{X_i\}, \{Y_i\}) = 0 \tag{2a}$$

which is assumed to exist between the two levels of the design variables (for an example, see beam cross-sectional area and moment of inertia as a function of the cross-sectional dimensions in a framework, eqs. A1,A2, in Appendix A). Thus, the subsystem optimization is

$$\min_{\{Y_i\}} P_i(\{Y_i\},\{X_i\})$$

$$\text{STO:} \quad \{Y_i\}_L \leqslant \{Y_i\} \leqslant \{Y_i\}_U \tag{3}$$

$$e^i_j(\{Y_i\}, \{X_i\}) = 0, \; j = 1 \rightarrow m_i$$

Sensitivity analysis of optimal subsystems.- Next, the optimum solution for each subsystem "i" is analyzed for its sensitivity to elements of $\{X_i\}$. This analysis (ref. 11) yields derivatives of the optimum objective function and design variables with respect to the constant parameters of the optimization problem. Specifically, the partial derivatives of P_i and $\{Y_i\}$ with respect to element x^i_r of $\{X_i\}$ are obtained. These partial derivatives are needed to express the optimal values of P_i and $\{Y_i\}$ as functions of the finite increments δx^i_r of variables x^i_r by means of linear extrapolations:

$$(\overline{P}_i)_e = (\overline{P}_i)_o + \nabla P_i^T\{\delta X_i\} \tag{4a}$$

$$(\overline{Y}_i)_e = (\overline{Y}_i)_o + \sum_{r=1}^{q} \nabla Y_i^T \delta x^i_r \tag{4b}$$

The extrapolations (eq. 4) in effect turn the optimal subsystem solution P_i and $\{Y_i\}$ into functions of the system level variables which are perturbed by $\{\delta X\}$ so that $\{X\}_p = \{X\}_o + \{\delta X\}$. As explained in the next section, the increments δx_i are variables in a consecutive stage of the system level optimization whose starting point is defined by $\{X_o\}$. The functions shown in eq. 4 will be referred to as linear representation of subsystem i, hence, the name of the linear decomposition method.

System optimization.- After all the subsystems have been optimized and their linear representations established, it is the system's turn to be optimized. In this optimization, one seeks values of the elements of $\{\delta X\}$ to minimize an objective function $F(X)$ subject to constraints.

The constraints are:

1. upper and lower limits on elements of $\{X\}$

2. upper and lower limits on elements of $\{Y_i\}$ vectors for all subsystems

3. requirement that the sum of the measures of unsatisfaction of the system and subsystem constraints be reduced by a predetermined amount.

Constraints 1 are ordinary side constraints. Constraints 2 are similar to move limits used in optimization by means of sequential piecewise linearization (e.g., ref. 10) and are required to prevent deterioration of the extrapolation accuracy (eq. 4) beyond an acceptable limit. Constraint 3 is a cumulative constraint defined in eq. 2 but generalized to encompass the system and all of its subsystems, so that it becomes

$$P = P_s + \sum_i P_i \tag{5}$$

where $P_s = \sum(\langle g_j \rangle)^2$ represents the system constraints. Specifically, constraint 3 is written as a single constraint in terms of P:

$$G(X) = P - P_t \leq 0 \tag{6}$$

where P is computable as a function of $\{\delta X\}$ because P_s in eq. 5 can be obtained from analysis of the system which yields the values of g_j, and the term $\sum P_i$ in that equation is a linear function of $\{\delta X\}$ via eq. 4. The P_t quantity is a target reduction of the constraint unsatisfaction established by prescribing a reduction schedule for P at the beginning of the whole process. For example, if a reduction schedule were set to decrease P from an initial value of P_B to zero in Q equal decrements, then a $P_t{}^q$ for iteration q would be:

$$P_t^q = P_B - q(P_B/Q) = P_B(1-q/Q) \tag{7}$$

Thus, the system optimization has the meaning of finding $\{\delta X\}$ that reduce the total constraint unsatisfaction by a predetermined amount while minimizing the objective function.

In a concise notation, it is

$$\min_{\{\delta X\}} F(\delta X) \tag{8a}$$

$$\text{STO:} \quad \{X\}_L \leqslant \{X\} \leqslant \{X\}_U \tag{8b}$$

$$\{Y_i\}_L \leqslant \{Y_i\} \leqslant \{Y_i\}_U, \quad i = 1 \to s_2 \tag{8c}$$

$$G(\delta X) = P - P_t \leqslant 0 \tag{8d}$$

When a solution $\{\delta X\}$ is found, one increments variables $\{X\}$ and $\{Y_i\}$, the latter via eq. 4b.

Iterative process and its termination.- The optimization procedure terminates when the system optimization yields $F = F_{min}$, within constraints 1 and 2, with $P \leqslant \varepsilon$ in constraint 3, (eq. 8d). Otherwise, the analysis of the system with new $\{X\}$ and $\{Y_i\}$ obtained in the most recent subsystems and system optimizations is repeated and the procedure continues until convergence. The convergence considerations will be discussed further in the following sections.

Consistent with the exterior penalty approach used in eq. 2, the procedure cannot progress unless a nonzero cumulative constraint value exists to be reduced in eq. 3 and 8. Consequently, the initial design must be infeasible with respect to at least one local or system constraint.

Main features of two-level system optimization.- The decomposition approach then, leads to an iterative process in which:

1. The system and subsystems are initialized in the infeasible domain.
2. System level optimization are interspersed between the series of subsystem optimizations. Each optimization involves a number of design variables smaller than that required by the "all-in-one" approach.
3. The objective function is entirely controlled by variables $\{X\}$ at the system level and never figures in the subsystem level optimizations directly. However, each optimal subsystem participates in the system level optimization by means of its linear representation.
4. The procedure terminates when all constraints in the system and subsystems are satisfied and the objective function is a minimum.
5. The decomposition is achieved by optimizing subsystems to obtain the best possible satisfaction of their local constraints consistent with the parameters imposed by the system, and then constructing their linear representations to be included in optimization of the system.

Generalization to Multilevel Systems

In many engineering systems, the number of levels required to reduce the suboptimization problems to manageable sizes may be greater than two. However, it is straightforward to generalize the idea to more than two levels. A three-level system is illustrated by the Venn diagrams shown in fig. 4. The cluster drawn with a heavy line is the two-level system introduced in fig. 3 and discussed in the previous section. That cluster is now a part of a more than two-level system whose description requires a suitably modified notation. In that system $S_{2,1}$ relates to $S_{1,1}$ in the same way that $S_{3,1}$, $S_{3,2}$, $S_{3,3}$ relate to $S_{2,1}$ in the two level scheme where $S_{2,1}$ plays the role of a system and $S_{3,1}$, $S_{3,2}$, and $S_{3,3}$ are subsystems.

To proceed with the method one must:

1. optimize the cluster composed of subsystem $S_{2,1}$, and subordinated subsystems $S_{3,1}$, $S_{3,2}$, and $S_{3,3}$, as if it were a two-level system. The previously discussed procedure applies here with one important difference: the objective function to be minimized for $S_{2,1}$ is taken to be the cumulative constraint (eq. 5) representing constraint unsatisfaction in the entire cluster and, consequently, the optimization of $S_{2,1}$ is defined by eq. 3.
2. repeat the above for other clusters in which the top subsystem are $S_{2,2}$, $S_{2,3}$, ...$S_{2,j}$, etc.
3. perform sensitivity analysis of each optimum $S_{2,j}$ to construct their linear representations to be used in optimization of $S_{1,1}$. Embedded in that linear representation of the $S_{2,j}$ subsystem are the linear representations of all the subsystems in the cluster subordinated to it.
4. optimize the $S_{1,1}$ system in a way analogous to the system optimization in the two-level scheme, defined by eq. 8 which includes the system objective function.

The generalization is recursive and applicable to a hierarchy of unlimited number of levels because in the process of decomposition all subsystems at each level are "swept" out and replaced by their linear representations. Each of these representations (eq. 4) and the associated move limits (eq. 8c) are carried upward to be used in optimizations of the higher level subsystems. This horizontal sweeping out of the subsystems and upward percolation of their linear representations together with the move limits continues, until only one system remains to be optimized on the top of the pyramid.

Thus, the main features of the multilevel optimization are the same as those given in the preceeding subsection for two-level optimization with the last of them reformulated as follows:

"5. The decomposition is achieved by optimizing subsystems at each level to obtain the best possible satisfaction of their local constraints consistent with the parameters imposed by their corresponding parent systems of the next higher level, and then constructing their linear representations to be included in optimizations of these parent systems."

KEY ELEMENTS OF THE METHOD IMPLEMENTATION

The basic concept of optimization with decomposition can be formalized in a way suitable for computer implementation. The formalization described in this section is given in terms of:

- hierarchial structure
- nodes, and clusters of nodes
- design and behavior variables
- parameters
- interaction quantities
- objective function
- operators
- iterative procedure

Each of these entities is defined and assigned a FORTRAN-like notation (also restated in the List of Symbols).

Hierarchial System

A system hierarchial structure is given in fig. 5. In that structure, one may identify a pair of nodes composed of a parent system, PS, and subsystem, SS. The identification is recursive so that any node can be designated PS for an SS at a lower level and, conversely, it may also be designated an SS subordinated to a node at a higher level. This recursivity is essential to assure flexibility of expanding the hierarchial structure downward and upward as needed. Each node in the hierarchy is identified by two-element integer position vector, NID, stating a level number from top (top level is designated as 1) and horizontal position from left. The notation PID ≡ NID and SSID ≡ NID is used when a given node is referred to as a parent and a subsystem, respectively. Referring to fig. 5 for an example, the node marked with PS has PID ≡ NID = 2,2 , and the one marked with SS has SSID ≡ NID = 3,4 .

Two examples of clusters of nodes are shown in figure 5 in the dashed-line envelopes. Nodes in a cluster are subordinated to a single cluster parent node on top of the cluster hierarchy. In the extreme, the entire system is a cluster subordinated to node (1,1). In the other extreme, a cluster may be degenerate and consist of only one node (e.g., node (3,2)). The entire graph shown in fig. 5 can be described in a tabular form shown in Table I, in which each unity entry represents a link between the nodes designated on top and left edges of the table and indicates existence of parent-subsystem interactions.

Design Variables, Parameters, and Behavior Variables

The design variables of a parent system are defined as the quantities which are being changed by an optimization algorithm in the optimization of parent system. The design variables are collected in a vector denoted X, and, optionally X(PID).

Design variables of a subsystem are defined as the quantities which are being changed by the optimization algorithm in the optimization of a subsystem. They are collected in a vector denoted Y, and, optionally, Y(SSID). Symbol V is used to denote both X and Y.

The parameters of optimization are defined as the physical quantities that remain constant in the optimization of parent system or subsystem. They are elements in vectors denoted H, and, optionally, H(PID) and H(SSID) for the parent system and subsystem, respectively.

Behavior (state) variables are physical quantities that characterize the state of a particular node in the system and are output by the node analysis. These variables are used to formulate constraints, an objective function, and to describe mutual influences among the nodes. The behavior variables are elements of a vector denoted B, and, optionally, B(PID) and B(SSID), for parent system and subsystem, respectively, recognizing that B = B(V,H).

Parent-Subsystem Interactions

Interaction quantities form a subset of the set of behavior variables of a node and are defined as these behavior variables which represent influence--of a parent system on each of its subsystem by being a part of the subsystem analysis input. They are collected in a vector denoted QI, and optionally, QI (PID,SSID).

Occasionally, some of the data that go as input into analysis of a parent system will also go without any change as input into analyses of one, or more, subsystems of that parent. For conceptual uniformity of the approach, such data will be regarded as having passed through the parent system analysis and, therefore, will be categorized as QI, as illustrated by an aircraft example in Table IV, entries QI and X.

Many systems of practical interest are organized as networks so that output of a particular node analysis may enter as input the analysis of its parent node and, possibly, analysis of another node bypassing their common parent node. Applicability of the decomposition is extended to such systems by introducing two additional types of interaction quantities: a Reverse Interaction Quantity and a Network Interaction Quantity.

Reverse Interaction Quantities form a subset of the set of behavior variables of a node and are defined as the physical quantities which are the output of a subsystem analysis and represent the influence of that subsystem on its parent system by being a part of the input into the parent system analysis. They are collected in a vector denoted RQI and, optionally, RQI(PID,SSID), analogously to QI(PID,SSID).

Network Interaction Quantities form a subset of the set of behavior variables of a node and are defined as the physical quantities which are the output of a subsystem by constituting a part of the other subsystem's analysis input directly, bypassing the parents of the two subsystems involved.

These quntities are collected in a vector denoted NQI and, optionally, $NQI(SSID_1,SSID_2)$ to indicate the direction of the influence from subsystem identified by $SSID_1$ to subsystem identified by $SSID_2$.

The definitions of RQI and NQI are included in the body of the report for completeness. However, explanation of how they can be included in the procedure is deferred to Appendix B.

Objective Function

The objective function is defined in two different ways: For the parent sytem of the highest level (PID = 1,1), the objective function is the measure of goodness (figure of merit) for the entire hierarchical system. In other words, this objective function is the same one that would have been chosen if the optimization without decomposition were to be performed. For each of the subsystems, the objective function is the measure of constraint unsatisfaction defined by eq. 5 for the given subsystem and all subsystems subordinated to it (an entire cluster). The objective function is denoted by F, and, optionally, F(PID) and F(SSID) recognizing that $F = F(V,H)$.

Constraints

Constraints of optimization are stated in the form of inequalities and equalities. The constraints are denoted by:

$$G \leqslant 0,\ E = 0, \text{ and, optionally}$$

$$G(NID) \leqslant 0,\ E(NID) = 0$$

where

$$G = G(V,H),$$

and

$$E = E(V,H).$$

Operators

In the optimization procedure, the quantities defined in the foregoing constitute input and output of several algorithms which will be referred to as operators in a generic sense. These operators are computer program subroutines, programs or program systems in which the actual content is problem dependent. They are defined below by names, always beginning with a character Ø, followed by a list of input and output variables, in that order, grouped in parentheses.

Interaction analysis operator.- This operator computes the interaction quantities: QI(PID,SSID), RQI(PID,SSID), and NQI($SSID_i$,$SSID_j$) when applied to a node of a system. There may be several vectors QI, as many of them as many subsystems SSID there are in a cluster directly subordinated to the parent node PID, and there may also be several NQI vectors per node. However, for each node there is only one vector RQI.

The operator is denoted by ØAQ, and, optionally ØAQ((V,H), (QI,RQI,NQI)), where QI, RQI and NQI stand for all vectors QI(PID,SSID), RQI(PID,$SSID_i$), and NQI($SSID_i$,$SSID_j$), respectively.

Interaction analysis sweep operator.- This operator carries out executions of the appropriate ØAQ operator for each node of the hierarchical system. In a system of physically dissimilar nodes, many operators ØAQ of different analytical capabilities are required, therefore, the ØASW operator must be capable of matching the ØAQ's with the appropriate nodes.

Constraint operator.- This operator computes those behavior variables whose computation is not included in the node's ØAQ operator and evaluates the node constraints. The operator is denoted by ØAC, and, optionally, ØAC((V,H,QI),(G,E)).

Similarly to the ØAQ's, there may be many ØAC's of different analytical capabilities to be matched with the particular nodes of the system. Logic to do the matching must be provided for in the overall optimization procedure itself.

Objective function operator.- This operator applies to the top node only and evaluates the objective function F for the entire hierarchical system. The operator is denoted by ØØBJ, or optionally, ØØBJ((X,H),(F)).

Optimization operator.- This operator solves an optimization problem for a node in the hierarchy graph shown in fig. 5. In a standard notation, the problem is:

$$\min_{\{V\}} \phi(V,H) \quad (9a)$$

$$\text{STO:} \quad \{G\} \leqslant 0 \quad (9b)$$

$$\{E\} = 0 \quad (9c)$$

The meaning of V and H, and the definition of the objective function ϕ depend on the position of the node being optimized in the system erarchy. Two types of position are distinguished as follows: 1. A node positioned anywhere except on the top of the hierarchy, and 2. The node positioned on top (e.g. node 1,1 in fig. 5).

Case 1: For node i subordinated to parent node j the variables $V = Y_i$, and parameters $H \equiv X_j$ of the parent node. The objective function is a penalty function representing the constraint violation, $\phi \equiv P(Y,X)$, eq. 2, in the node. If the node being optimized is a parent of a nondegenerate cluster, then its penalty function includes the constraint violations in all the nodes in the cluster subordinated to it (eq. 5). These violations are evaluated by linear representations of the subordinated nodes (eq. 4).

The inequality constraints in eq. (9b) are the same as in eqs. (8b) and (8c), except that the vectors $\{Y_i\}$ are included from all subsystems in the cluster. The equality constraints in eq. (9c) arise from the relationship stated in eq. (2a) which exists between the variables of the node and its parent.

The optimization operator executing with the aforementioned definitions of V,H and ϕ will be referred to as executing in mode 1.

Case 2: For the parent system on top of the pyramidal hierarchy, the objective function is the one for the entire hierarchical system, $\phi \equiv F$. The design variables $V \equiv X$ and $H \equiv H$, where X and H are design variables and parameters of the top node of the hierarchy. The inequality constraints in eq. (9b) are the same as in eqs. (8b) and (8c), except that the vectors $\{Y_i\}$ are included from all subsystems of the entire system. The equality constraints in eq. (9c) exist only, if required by the system level optimization problem at hand; they do not arise from eq. (2a).

The optimization operator executing with these definitions of ϕ, V, and H will be referred to as executing in mode 2.
The optimization operator is denoted by ØØPTm or, optionally, ØØPTm((H),(V,ϕ)) where V and ϕ denote optimum values of the design variables and objective function, and m = 1 or m = 2 indicate the execution mode. In its execution the optimization operator calls the operators ØØBJ (in mode 2 only) and ØAC.

The operator is defined as a "black box," therefore the optimization procedure it uses to solve eq. 9 can be freely selected and need not be specified in detail here. It is conceivable that several optimization operators would be used, each employing an optimization technique specialized for a particular node or group of nodes. A logic necessary to match the operators with the nodes must be provided for in the overall optimization procedure itself.

Optimum sensitivity analysis operator.- This operator calculates derivatives of V and ϕ with respect to parameters H. Specifically, for a node i subordinated to parent node j the operator will output the partial derivatives of ϕ_i and Y_i with respect to the elements of X_j. The operator is denoted by ØSEN((H,V,ϕ), ($\partial V/\partial H_j, \partial\phi/\partial H_j$)).

Subsystem elimination sweep operator.- This operator consists of a repetition of the operators ØØPT1 and ØSEN for each subsystem in a row to produce their linear representation to their parents. Notation for the operator is ØSUBSW, or, optionally, ØSUBSW(ØØPT1,ØSEN) to indicate that this operator calls for application of ØØPT1 and ØSEN, in that order, to each node in the given row.

Procedure termination operator.- This operator performs the following tests:

1. P(PID = 1,I) $\leqslant$ TOL1
2. For each node:
   $G_j \leqslant$ TOL2 and $|E_j| \leqslant$ TOL3 where TOL2 and TOL3 can be prescribed for each constraint of each node.
3. Increment of F $\leqslant$ TOL4 during last n executions of the procedure.

When all tests are satisfied, the operator makes a decision to stop further repetitions of the procedure. The operator is denoted by ØTER.

Overall Iterative Procedure

With the foregoing definitions, the entire iterative procedure of optimization with decomposition can be presented in a compact step-by-step form.

1. Initialization of {X} and {Y} for all nodes.
2. Execute ØASW(ØAQ)
3. Row number set to the bottom row number
4. Execute ØSUBSW(ØØPT1,ØSEN)
5. Row number reset to next higher level
6. If the row number is not 1 (the highest level), repeat from 4, otherwise continue
7. Identify $\phi \equiv F$
8. Execute ØØPT2
9. Execute ØTER
10. Repeat from 2, unless ØTER indicates end of the procedure, then continue
11. Final analysis:
    - 11.1 Execute ØØBJ
    - 11.2 Execute ØASW
    - 11.3 Execute ØAC for all nodes of the system

    STOP

A block diagram in fig. 6 shows the calling hierarchy of the operators in the procedure.

Concluding the procedure description, it should be emphasized that the definitions of the operators given in the foregoing are introduced primarily for the conceptual purposes. These definitions are by no means meant to prescribe rigidly the detailed allocation of functions to the code modules, that allocation being clearly an implementation decision to be made in consideration of a number of computer software and hardware factors not included in the scope of this report. On the aggregate, however, the method implementation should have all the functional capabilities called for by the operator definitions.

STATUS OF KEY ELEMENTS OF THE METHOD

The previous discussion referred several times to self-contained operations which constitute the key elements for the proposed optimization procedure. This section summarizes the state-of-readiness for these key elements.

System and Subsystem Analyses

In the context of the decomposition scheme, the purpose of the analyses is to calculate the behavior variables which, in turn, are used to compute the interaction quantities (QI, RQI and NQI), the equality and inequality constraints (G and E), and the objective function. Elements of such analyses are embedded in the operators ØAQ, ØAC, and ØØBJ; they are entirely problem dependent and not a part of the proposed development, except where needed to carry out the test cases. Examples relevant to aircraft optimization are the generally available programs for finite-element structural analysis, including substructuring, computational aerodynamics, and aircraft performance analysis. Programs described in ref. 14 constitute an example of the latter.

Although it is preferable, for obvious reasons, to obtain the needed behavior variables by analysis, intrinsic modularity of the decomposition scheme allows substitution of experiment for analysis wherever necessary.

It is desirable that the analyses be capable of generating derivatives of the behavior variables, in addition to the variables themselves, preferably by efficient analytical techniques. Example of algorithms for such derivatives are computation of derivatives of stress with respect to cross-section areas described in ref. 5 and a technique for obtaining derivatives of the pressure coefficients at a given wing location with respect to the wing planform shape variables (e.g.: aspect ratio) described in ref. 15.

Optimization

Numerous mathematical programing techniques embodied in existing computer programs are available to perform optimization which is the task of operator ØØPT in mode 1, ($m = 1$), and mode 2, ($m = 2$). The program CONMIN (ref. 16) based on the usable-feasible directions algorithm is one natural candidate for a major building black of the ØØPT operator. Implementation of CONMIN in ØØPT may be facilitated by embedding it in program FRANOPP described in ref. 17.

The effectiveness of the use of penalty function defined in eq. 2 has been well established in the numerous reports on the Sequential Unconstrained Minimization Technique (SUMT) with an exterior penalty function. Favorable experience with the use of such a penalty function instead of many individual constraints in the usable-feasible direction algorithm (program CONMIN) was reported in ref. 8. An example of an optimization in mode 1 was given for a thin-walled beam in ref. 11, using numerous local design variables, and constraints on stress, local buckling, and cross-section geometry.

However, by its very nature the formulation based on an exterior penalty function is nonconservative, and it is also inefficient when the initial trial design is a feasible one. Efficiency improvement and preservation of the design feasibility in such cases is one area that requires further work. It may require exploration of alternative formulations for a single measure of the constraint violations; one such measure appears to be suggested by the well-known interior, and extended-interior penalty function formulations which, when started with a feasible design, efficiently generate a series of steadily improving feasible designs, in a conservative fashion that many engineers prefer in conducting a design process.

At the subsystem level, the optimization operator must account for the equality constraints. The equality constraints (eq. (2a)) may be satisfied exactly, preferably by a design variable elimination technique whenever analytically possible or by one of the specialized optimization procedures. An example of the former is given in ref. 11 (the case of a thin-walled beam), a few techniques of the latter category are described in refs. 5 and 18. As an alternative to the exact satisfaction of the equality constraints in every execution of the optimization operator (referred to as a strict equality optimization mode), it is possible to relax the strict equality requirement and to include the squared residuals of the equality constraints in the penalty function (cumulative constraint, eq. 2). This alternative (referred to as a relaxed equality optimization mode) will lead to a satisfaction of the equality constraints when, but not necessarily before, the whole procedure converges. A satisfactory experience with this alternative has been reported in ref. 2 for a two-level optimization of structures built up of tension-compression components.

In case the reverse interaction quantities (RQI) exist between a subsystem and its parent node, they will give rise to equality constraints at the parent node level. These constraints represent a requirement of vanishing difference between the RQI values assumed before the given subsystem analysis is accomplished and the values obtained from that analysis. Example of such equality constraints and a demonstration that they can be satisfied iteratively are given in refs. 19 and 20 for the problem of aeroelastic loads. In this problem, the RQI is the shape of an elastic aircraft deformed due to the aeroelastic loads which themselves are influenced by that shape. To account for this coupling, two aerodynamic shapes, one used in the aerodynamic load analysis and the other obtained from structural analysis due to these loads, are being iteratively brought into a match.

Conceptually, an RQI such as the one discussed above could as well be identified as an NQI. Consequently, the results reported in refs. 19 and 20 appear to indicate that inclusion of the network influences, NQI, will not keep the overall procedure from converging.

The proposed decomposition procedure calls for use of the piecewise linear approximations (e.g., eq. 4) which implies that te operator ØØPT will carry out its optimization task (in either mode 1 or mode 2) entirely, or at least partially, in a piecewise-linear manner whose effectiveness is well documented in refs. 4, 6, 10, and 21.

Optimum Sensitivity Analysis

This analysis performed in the operator ØSEN has been described in ref. 11, and has been implemented in a general purpose experimental computer code which was tested on a number of examples with satisfactory results. To function as a part of the operator ØSEN, the program required the following improvements:

1. automating selection of the active constraints
2. automatic elimination of linearly dependent constraint gradient vectors
3. implementation of of a linearized version of the optimum sensitivity analysis whose input does not require the costly-to-compute second derivatives of the behavior variables.

Termination and Sweep Operators

These operators (ØTER, ØASW and ØSUBSW) are conceptually straightforward. They may take form of high-level language codes or, optionally, may be written in an operating system control language, dependently on the organization of the entire procedure (see section Execution Control, Data Management and Hardware Implementation).

Overall Convergence of the Optimization Procedure

The sensitivity analysis provides a very powerful extrapolation tool for predicting changes to the optimum solution caused by changes to the problem parameters. For highly nonlinear structural optimization cases, the accuracy of extrapolation represented by eq. 4 was demonstrated in ref. 11 to be very good throughout the parameter changes of the order of 20 percent.

In view of that finding and a success with a two-level approach reported in ref. 2, prospect for fast overall convergence of the proposed multilevel optimization procedure appears to be very good. However, no proof of convergence is available and it is doubtful that one can be developed for an entirely general case, in which local minimum and disjoint design spaces would have to be considered since the design experience indicates that they do occur in optimization of engineering systems. Nevertheless, if the proposed method can be shown experimentally to produce a sequence of "improved" designs, then there should be no difficulty in formulating "practical" convergence criteria by which to decide when to terminate the process.

Execution Control, Data Management and Hardware Implementation

Since the decomposition procedure calls for iterative execution of computer programs organized in the afore discussed operators, numerous loops, branches, and other standard software constructs will have to be coded. In the general case, the computer programs will mostly be stand-alone, existing codes too large to be turned into subroutines to a main program. Consequently, to build the desired constructs, one can choose among the following software executive capabilities currently available:

1. control language embedded in an operating system, e.g., CDC Cyber Control Language, whose use as an executive software is described in refs. 22, 23, and 24.

2. capability to command the selected operating system functions, including execution of stand-along programs, from inside of a FORTRAN program. This capability, intrinsic in some minicomputers (such as a PRIME 750) is now also available on mainframe computers (such as the CDC Cyber computer series under NOS 1.4).

3. a high-level language called Engineering Analysis Language (EAL), reference 25, which is a utility provided with a system of programs and data of the same name. The system is open ended and permits addition of new, heretofore stand-alone codes.

4. A combination of alternative 1, 2, and 3.

It is conceivable that for the decomposition concept demonstration, a pilot program can be built using a simple main program-subroutine organization.

Regardless of the type of executive capability, one has to be prepared, in a general case, for handling a large volume of data and to provide means for visibility and tracking of the final and intermediate results. The minimum of the data handling capability appears to be assured in the file management typical of a modern operating system such as NOS 1.4, used in a way described in ref. 24, with the more advanced data utility being available in EAL (ref. 25). Ultimately, the IPAD (ref. 13) data management software such as RIM (ref. 28) and IPIP (ref. 29) may prove to be the solution. Regarding the intermediate and final result visibility, it appears that IPAD- offered graphics utilities, together with the vendor provided operating system graphics software, should be adequate.

Hardware implementation of the decomposition procedure may be done solely on a mainframe computer, or a minicomputer, or it may be distributed over a network of computers of various types as suggested in ref. 12. In view of the parallelism inherent in both the decomposition concept and network computing, the latter may emerge as a preferred alternative.

In summary, all the main building blocks appear to be available in forms adequate for immediately commencing development of the proposed multilevel optimization procedure. Improvements needed in some of the building blocks may be accomplished in parallel with that development, and those aspects of the method which cannot be ascertained theoretically; e.g., the overall convergence, can be explored experimentally as postulated in the section on testing.

DECOMPOSITION METHOD AND ORGANIZATION OF A DESIGN PROCESS

The acceptance of the proposed method, with its wide range of applicability will depend not only on its use as a new tool by designers but even more importantly on its use by managers of design organizations. A few factors likely to affect that acceptance are examined below.

Compatibility with a Design Office Structure

In contrast to "all-the-variables-in-one-basket" approach which would hide the optimization mechanism in one mysterious "black box," the multilevel decomposition makes contributions of all the disciplines clearly visible and therefore should fit very well the natural division of the design functions among many specialty groups cooperating with each other. The obvious similarity between the system hierarchy diagram, such as the one in fig. A5(b), and a typical design office organization chart supports that assertion. Each node of the hierarchy diagram can be equated with a group of specialists equipped with their tools, intuitive judgment, methods, computers and, yes, experimental facilities.

Along with that similarity, the approach brings in some new elements which are likely to have a profound and positive impact on the way the design decisions are made and compromises reached in the inevitable interdisciplinary conflicts. These new elements are as follows:

1. Each group designs their subsystem to specific requirements represented by parameters $H = X$ prescribed from the next higher level, with the objective of minimizing the measure of violation of the subsystem constraints (the cumulative constraints). That objective, which replaces a traditional, disciplinary measure of goodness (e.g., minimum weight), is pursued through repetitions of the entire procedure, reducing its value to a new, lower minimum in each repetition until it reaches zero--an indication that all constraints are satisfied. Example of this might be a structures group where one iteration task might be to design an airframe of a prescribed mass, loads, and external shape to minimize cumulative measure of violation of constraints such as stress, deformation, and flutter.

2. Each group analyzes its optimal design for sensitivity with respect to the prescribed parameters.

3. At the next higher level node (parent system), the amounts of violation and the sensitivity derivatives from all the subsystems subordinated to that node are used to perform the parent system optimization. This optimization resolves the tradeoffs between the subordinate modes in an entirely objective manner on the basis of the optimum sensitivity derivatives.

4. All the sensitivity information needed to resolve all the trade-off problems posed in the entire system design rises to the very top of the hierarchy where the objective function and constraints for the entire system are considered.

In the process, engineers at each level have full visibility of their results and full knowledge of the sensitivity information concerning their subsystem and its cluster of subsystems at the levels below. Along with responsibility for their results, they also retain choice and control of their tools and methods to perform aforementioned tasks 1, 2, and 3. Remarkably, this choice may include also the use of experimental methods.

Efficiency of the hierarchial organization requires, logically, that data be transferred quickly and that work at various nodes be performed in parallel to the largest possible extent. These requirements, in turn, call for integrated management of data and for parallel computer processing. In addition, the latter brings about the issue of synchronous vs. asynchronous (chaotic) modes of operation.

Synchronous Operation Mode

In a synchronous mode, which seems "natural" to anyone used to the conventional iterative algorithms, optimization of a given parent system is not attempted before optimizations and sensitivity analyses are completed for each of the subsystems in the subordinate cluster. This is a preferred mode, but only if all the subsystems require approximately the same time for their optimizations and sensitivity analyses, or if these operations take so little time that waiting for the "slowest" subsystem results does not matter, otherwise one may consider an asynchronous mode.

Asynchronous Operation Mode

In this mode, optimization and sensitivity analysis of a parent system proceed when the optimization and sensitivity results from most, but not necessarily all, of the subordinate subsystems are available. The few subsystems which are late with their results are temporarily ignored until the next opportunity to include their information arises. It is easy to predict, that those subsystems in which experimental work is involved would be especially, but not solely, prone to fall into the "slow" category which would contribute intermittently instead of consecutively to the successive iterations in the process.

An encouraging precedent for the notion that iterations based on such intermittent contributions do converge is being found in the concept of the so-called chaotic iterative methods recently proposed (ref. 26) for solution of large sets of simultaneous equations by parallel computing in a network of microprocessors. Furthermore, one may argue that although the optimum obtained by an asynchronous optimization will probably be different from one obtained by a synchronous optimization, it will not necessarily be worse.

This assertion is based on the following reasoning: In general, the original (nondecomposed) problem may be nonlinear and nonconvex, therefore, local minima may occur. Hence, the optimization result (discovery of one local minimum or another) is search-path dependent, the path taken being affected by the choice of the synchronous or asynchronous mode. However, although different paths may lead to a different local minima, there is no reason to expect that, intrinsically, the path corresponding to one particular mode will always lead to a superior local minimum.

Summary of Advantages

In summary, the multilevel decomposition viewed in context of the organization of the design offers the following advantages:

1. The engineering specialty groups retain control over their tasks, including the methods and tools. Consequently, creativity, insight, and responsibility for results are promoted rather than being impeded, as would be the case in the all-in-one optimization which would be perceived as a "black box" taking over the engineer's task.

2. Because of the independence of the choice of solution methods and the possibility of using the asynchronous mode, experimental methods may be included.

3. Visibility of intermediate results at all levels and a clear picture of the trade-off trends is assured.

4. Advantage is taken of the opportunities provided by computer technology such as the parallel processing and the organized means of data management.

Implementation of the multilevel decomposition method may be visualized as a network of computers of various, judiciously chosen, types organized in a hierarchial structure underlying a similar structure of a human organization which uses the method as a tool in design of a complex engineering system.

TESTING THE METHOD

For a successful implementation of the proposed multilevel optimization method, two types of testing are needed. One of them will be a routine module-by-module verification of the code correctness, mandatory in any software system development. The other type of testing should address the conceptual uncertainties and generate experience to guide the implementation. For this type of testing, one may propose working out a few physical examples as well as a more abstract simulation which are described below.

Physical Examples

A portal framework used as example 1 in Appendix A would provide a good development test case. Its testing should consist at least of the following:

1. Optimization by the all-in-one approach starting from a few different initial designs to establish a benchmark optimal design (or designs if local minima are found).

2. If a usable-feasible directions optimization technique is to be used, then the optimization should be repeated with a cumulative constraint (eq. 2) to ascertain its influence on the benchmark results.

3. Repetition of the optimization using a piecewise-linear procedure to evaluate its influence on the benchmark results.

4. Finally, the multilevel optimization to compare its convergence, efficiency, and results with the benchmark case. This comparison in conjunction with the data available from tests 2 and 3 will reveal the extent to which the inevitable discrepancies are caused by the multilevel approach, rather than by the use of cumulative constraint and piecewise-linear procedure which are auxiliary tools incidental to the very concept of the multilevel decomposition.

A simplified aircraft optimization described as example 2 in Appendix A would provide an opportunity for testing multidisciplinary influences. A benchmark result could be obtained by one of the existing aircraft synthesis codes, e.g., program described in ref. 14. For consistency of comparison, the level of detail of the aerodynamic and structural analyses in the multilevel optimization should be the same as in the code for generating the benchmark result.

Simulation of a Multilevel Optimization

The primary purpose of the simulation will be generation of operational experience with various configurations of the system to be optimized in order to explore the new, previously untested aspects of the entire method and guide its implementation. The major untested aspects that cannot be ascertained theoretically and, therefore, need to be investigated experimentally are:

1. Overall convergence of the method

2. Influence of the interactions RQI and NQI on the convergence

3. Computational costs savings of the multilevel approach relative to the all-in-one approach

4. Relative efficiencies of various computer organizations of the procedure and user-oriented features of its input and output

5. Feasibility, accuracy and efficiency of the parallel synchronous and asynchronous modes of operation

To perform these tests quickly and inexpensively, only analytical functions should be used in the nodes of the simulator system. Initially, the functions for simulation of the node constraints and the system objective should be chosen to be convex in the X and Y domains in order to avoid the multiple local minima problem. To assure ease of code modifications, the entire simulation program should be coded in-core in the form of a main program calling subroutines, even for simulation of the case of parallel computing on distributed processors.

Decisions on the details of the simulation program organization and selection of the analytical functions as well as details of the physical test examples are part of the development program beyond the scope of this report.

CONCLUSIONS

A method for decomposing a large optimization problem into a hierarchy of separate-but-coupled subproblems is introduced. The method is based on a linear decomposition principle, using the concept of sensitivity of each optimal subsystem to the variables of the system of next higher level which are treated as parameters in that subsystem optimization.

It is shown that the method can be implemented using as building blocks the algorithms and computer programs which already exist. The state of readiness of each building block is discussed, and the formulation of the entire integrated procedure for optimization with decomposition is described to the degree of detail needed to guide the development work, including testing.

The main benefits of the method include: (1) compatibility with the natural organization of a design process under which the work is divided among the specialty groups, (2) ability to take advantage of the computer technology progress in the organized data handling and parallel computing, (3) ability to absorb the disciplinary experimental data in the process of system optimization, and (4) the mathematical rigor introduced in the process of making the design decisions for practical engineering problems.

APPENDIX A

EXAMPLES OF DECOMPOSITION

Three examples provided in this appendix illustrate decomposition for optimization purposes. The examples are minimum mass optimization of a simple framework (which was discussed in an abridged form in section "Key Idea for the Decomposition"), minimum mass optimization of a wing-fuselage airframe, and an optimization of an aircraft for a minimum of fuel consumed to deliver a given payload over a prescribed range. Discussion of the examples is limited to the features directly relevant to the decomposition concept with the disciplinary details of analysis omitted. General familiarity of the reader with the disciplinary technologies involved in the examples is assumed.

Example 1: A Portal Framework

A portal framework shown in figure 1 is an example of a hierarchial system that can be optimized for minimum mass subject to strength and displacement constraints using the linear decomposition approach. The decomposition is two-level and results directly from the fact that one can use an engineering beam theory to analyze the framework for internal forces (end forces on each beam) and displacements, assuming that A(cm^2) and Iy(cm^4) for each beam (or even Iy alone, if beams are slender) are given but without knowing the detailed cross-sectional dimensions (b_1, t_1...etc.). These dimensions can be optimized separately for each beam as long as the end forces (fig. 2) in each beam are known and assumed fixed.

Consequently, the correspondence of the basic elements of the decomposition approach to the specifics of the framework example is immediately obvious as shown in a Venn diagram in figure A1; a hierarchy diagram in figure A2; and Table II, which displays the one-to-one equivalences of entities involved. Example of a particular form of eq. (2a) can be given by cross-sectional area and moment of inertia as functions of cross-sectional dimensions for an I-beam:

$$z_2 = h-(t_1+t_2)/2; \; z_3 = (h-t_2)/2$$

$$A_1 = b_1t_1; \; A_2 = b_2t_2; \; A_3 = (h-t_1-t_2)t_3$$

$$A = A_1+A_2+A_3 \qquad \text{(A1)}$$

$$c = (A_2z_2+A_3z_3)/A$$

$$I_y = (b_1t_1^3+b_2t_2^3+t_3(h-t_1-t_2)^3)/12 + A_1c^2 + A_2(z_2-c)^2 + A_3(z_3-c)^2 \qquad \text{(A2)}$$

Example 2: A Wing-Fuselage Airframe

An airframe composed of an arrow-wing and fuselage (figure A3(a)) provides an example of a hierarchical structural system whose strongly coupled subsystems are substructures. As shown in figure A3, the relatively long root chord of the wing in the arrow wing configuration chosen for the example makes the chordwise plate bending of the wing couple with the fuselage bending and gives rise to interaction forces (F_1 through F_5 in figure A3(b)) which can be computed using a stiffness matrix K shown topologically in figure A3(c)). Each element K_{ij} of matrix K is a sum of stiffness contributions from wing, $K_{ij}^{(w)}$, and fuselage, $K_{ij}^{(f)}$: $K_{ij} = K_{ij}^{(w)} + K_{ij}^{(f)}$. Consequently, the stiffness coefficients $K_{ij}^{(w)}$ and $K_{ij}^{(f)}$ play the same role in the airframe substructuring analysis as the beam cross-sectional properties A and I_y in the framework analysis and become the system level design variables. Assuming the airframe structural mass as an objective function to be controlled at the system level, the additional design variables are structural masses, $W^{(w)}$ and $W^{(f)}$, of the wing and the fuselage.

Similarly as in the framework example, the correspondence of basic elements of the decomposition approach to the particular elements of the airframe example is shown by means of a Venn diagram in figure A4(a), a hierarchy diagram in figure A4(b), and Table III. Examples of local variables are shown in figure A4(c).

The choice of stiffness coefficients as design variables is highly unconventional and experience is limited relative to its practicality. The ultimate proof of the proposed approach will lie in the test results.

Example 3: A Transport Aircraft

Design of aircraft provides an example for multilevel optimization of a multidisciplinary engineering system. For this purpose, one may isolate one of the many aspects of aircraft optimization and present it here in a manner intentionally narrowed and simplified for conciseness. Suppose that the optimization task calls for minimization of the fuel consumed to deliver a given payload over a given range. One possible choice of the design variables is listed in two groups: (1) the wing area S, aspect ratio AR, airframe structural mass $W^{(s)}$, and (2) takeoff thrust T_o, propulsion system mass $W^{(p)}$, fuel consumed $W^{(j)}$, drag coefficient $c_D^{(p)}$, and the mission parameters of cruise altitude h_{cr} and Mach number M_{cr}. Assuming the above as the system level variables, a Venn diagram and a hierarchy diagram develop as shown in figure A5 (a and b).

The variables of group 1 (S, AR and $W^{(s)}$) govern the airframe which is a parent system to the wing and fuselage subsystems, and which was examined in the previous example. The diagram shows propulsion as a subsystem directly connected to the aircraft system and governed by group 2 of the system level variables. The propulsion subsystem design variables $\{y_1^{(2,2)} \ldots y_i^{(2,2)}\}$ need not be specified for the purpose of this discussion beyond mentioning, say, a compressor diameter and a turbofan bypass ratio as examples.

Correspondence of the generic elements of the decomposition approach to the particular elements of this example is given in Table IV. Notice that, unlike in the previous two examples, there exists a reverse interaction quantity (RQI) between the nodes 2.1 and 1.1. This RQI represents influence of the elastic deformation of the airframe on the aerodynamic loads computed at the system level and calls for a special system level constraint. The function of that constraint will be to bring the aerodynamic loads assumed as input to the structural analysis into equality with the same loads corrected to reflect the deformed shape which is obtained as output from that analysis.

The influence of an NQI type also exists in this example between nodes 2.2 and 3.1 in the form of the engine weight influence on the wing structure stress.

APPENDIX B

HANDLING THE REVERSE AND NETWORK INTERACTIONS

This appendix describes one possible approach to the problem of including the reverse and lateral interaction (RQI's and NQI's) among the nodes of a hierarchical, top-down system.

An example of a system hierarchial diagram with the RQI and NQI existing in addition to the QI is shown in figure B1. Problems caused by existence of the RQI and NQI are defined and their solution proposed.

Problem 1: Vertical Reverse Coupling

The problem is caused by existence of the RQI and can be described as follows:

For solution of, say, node 3.2 (diagram, fig. B1), one needs to know QI from solution of node 2.1, but that node cannot be solved unless the RQI from node 3.2 is available.

The solution to the problem is to assume an, as yet unknown, RQI = RQI*, solve node 2.1 obtaining QI = QI* and then solve node 3.2, using QI* as an input, to obtain a corrected RQI = RQI** which in turn will result in a corrected value of QI = QI**. Convergence is obtained when the differences between the elements of the vectors RQI* and RQI**, and QI* and QI** diminish below an assumed tolerance.

This convergence could be achieved by a separate dedicated iteration embedded in the overall optimization procedure. However, there is no need for such dedicated iteration, because the task can be delegated to the optimization procedure by simply adding to the set of constraints of the parent node (node 2.1 in the example in figure B1) a constraint enforcing the convergence of, say, RQI* and RQI**, i.e.,

$$g_r = \|RQI^{**}\| - \|RQI^{*}\| \leq \varepsilon \, \|RQI^{**}\| \qquad (B1)$$

where ε is a small tolerance parameter.

The iteration toward converging the RQI** and RQI*-vectors is clearly a part of the overall system analysis. Therefore, its proposed blending with the optimization itself can be recognized as the so-called integrated analysis-optimization whose conceptual and numerical validity is established in the literature, e.g., refs. 5 and 27, and refs. 19 and 20 in which an elastic aircraft shape deformed by aerodynamic loads played implicitly a role of an RQI.

Problem 2: Lateral Coupling

The problem is caused by existence of the NQI and, basically resembles that of RQI, therefore, a similar approach can be taken. Two subproblems need to be distinguished here.

Subproblem A - a one-way lateral interaction.- For solution of, say node (p,q), one needs to know a result of solution of node (m,n) (an influence marked by a dotted arrow in figure B1) which can be solved without inputs from node (p,q).

Solution is simply that of timing, if m < p. Node (m,n) (which is at a level higher than node (p,q)) is solved first and the solution of node (p,q) follows with the NQI information routed directly from node (m,n) to node (p,q), e.g., from node (3.2) to node (4.1) in figure B1.

However, if m > p (a node at a lower level influences a node at a higher level, e.g., (4.1) to (3.3)), the situation becomes similar to the one involving an RQI and calls for a similar solution. First time, one can solve the (p,q) node for assumed vector NQI* ((m,n),(p,q)) and then solve the (m,n) node to produce a corrected value of NQI** ((m,n),(p,q))to be used in the next analysis of the (p,q) node, etc. Convergence can be enforced by introducing a constraint analogous to equation B1:

$$g_r = \| NQI^{**} \| - \| NQI^{*} \| \leqslant \varepsilon \| NQI^{**} \| \qquad (B2)$$

in the optimization of the node which is the lowest level parent to both nodes (m,n) and (p,q). In the example in figure B1, such node is node 2.2.

Subproblem b - a two-way lateral interaction.- In this subproblem, each node in a pair (m,n) and (p,q) needs as input for its analysis the NQI from the other node in the pair. The solution is to include in the optimization of the node which is the lowest level common parent of the pair two additional constraints to account for discrepancy between the NQI** and NQI* acting in both directions.

$$g_r = \| NQI^{**}((m,n),(p,q)) \| - \| NQI^{*}((m,n),(p,q)) \| \leqslant \varepsilon \| NQI^{**}((m,n),(p,q)) \| \qquad (B3a)$$

$$g_{r+1} = \| NQI^{**}((p,q),(m,n)) \| - \| NQI^{*}((p,q),(m,n)) \| \leqslant \varepsilon \| NQI^{**}((m,n),(p,q)) \| \qquad (B3b)$$

and to assume the NQI* values in the first iteration.

An example of such interaction is illustrated in figure B1 for a pair of nodes (3,5) and (4,2) whose lowest level common parent node is (2.2).

REFERENCES

1. Lasdon, L. S.: Optimization Theory for Large Systems. The McMillan Co., New York, 1970.

2. Schmit, L. A.; and Ramanathan, R. K.: Multilevel Approach to Minimum Weight Design Including Buckling Constraints. AIAA J., Vol. 16, No. 2, February 1978, pp. 97-104.

3. Kirsch, U.; and Moses, F.: Decomposition in Optimum Structural Design. J. of Structural Division ASCE, STI, January 1979, p. 85.

4. Schmit, L. A.; and Miura, H.: Approximation Concepts for Efficient Structural Synthesis. NASA CR-2552, March 1976.

5. Fox, Richard L.: Optimization Methods for Engineering Design. Addison-Wesley Publ. Co., Reading, MA, 1971.

6. Storaasli, O. O.; Sobieszczanski, J.: On the Accuracy of the Taylor Approximation for Structure Resizing. AIAA J., Vol. 12, No. 2, February 1974, pp. 231-233.

7. Haftka, R.: Sensitivity Analysis. Proceedings of the NASA-ASI Session on Modern Structural-Optimization, Liege, Belgium, August 1980.

8. Sobieszczanski-Sobieski, Jaroslaw: From a "Black Box" to a Programing System: Remarks on Implementation and Application of Optimization Methods. Proceedings of a NATO Advanced Study Institute Session on Structural Optimization, University of Liege, Sart-Tilman, Belgium, August 4-15, 1980.

9. Pope, G. G.; and Schmit, L. A.: Structural Design Applications of Mathematical Programing Techniques. AGARDOGRAPH No. 149, Ch. 5, pp. 48-54, February 1971.

10. Anderson, M. S.; and Stroud, W. J.: A General Panel Sizing Computer Code and Its Application to Composite Structural Panels. AIAA J., Vol. 17, No. 8, August 1979, pp. 892-897.

11. Sobieszczanski-Sobieski, J.; Barthelemy, Jean-Francois; and Riley, Kathleen M.: Sensitivty of Optimum Solutions to Problem Parameters. Presented at AIAA/ASME/ASCE/AHS 22nd Structures, Structural Dynamics and Materials Conference, Atlanta, GA, April 6-8, 1981. AIAA Paper No. 81-0548, also published as NASA TM 83134, May 1981

12. Rogers, J. L., Jr.; Dovi, A. R.; and Riley, K. M.: Distributing Structural Optimization Software Between a Mainframe and a Minicomputer. Proceedings of the Engineering Software Second International Conference and Exhibition, London, England, March 24-26, 1981.

13. Fulton, R. E.: Overview of Integrated Programs for Aerospace-Vehicle Design (IPAD). NASA TM-81874. September 1980.

14. Sliwa, S. M.: Sensitivity of the Optimal Design Process to Design Constraints and Performance Index for a Transport Airplane. AIAA Paper No. 80-1895, AIAA Aircraft Systems and Technology Meeting, Anaheim, CA, August 4-6, 1980.

15. Mercer, J. E.; and Geller, E. W.: Development of an Efficient Procedure for Calculating the Aerodynamic Effects of Planform Variation. NASA CR-3489, November 1981.

16. Vanderplaats, Garret N.: The Computer for Design and Optimization. Computing in Applied Mechanics. R. F. Hartung, ed., AMD - Vol. 18, American Soc. Mech. Eng., c. 1976, pp. 25-48.

17. Riley, K. M.: FRANOPP - Framework for Analysis and Optimization Problems. NASA CR-165653, Kentron International, Inc., Hampton, VA, January 1981.

18. Vanderplaats, G.: Optimization with Equality Constraints Using Program CONMIN. Private Communication, October 1979.

19. Giles, G. L.; and McCullers, L. A.: Simultaneous Calculation of Aircraft Design Loads and Structural Member Sizes. AIAA Paper No. 75-965, AIAA Aircraft Systems and Technology Meeting, Los Angeles, CA, August 4-7, 1975.

20. Sobieszczanski, J.; McCullers, L. A.; Ricketts, R. H.; Santoro, N. J.; Beskenis, S. D.; and Kurtze, W. L.: Structural Design Studies of a Supersonic Cruise Arrow Wing Configuration. NASA CP-001, Hampton, VA, November 1976, pp. 659-684.

21. Sobieszczanski-Sobieski, J.; and Bhat, R. B.: Adaptable Structural Synthesis Using Advanced Analysis and Optimization Coupled by a Computer Operating System. J. of Aircraft, Vol. 18, No. 2, February 1981, pp. 142-149.

22. Sobieszczanski, J.: Building a Computer Aided Design Capability Using a Standard Time Shared Operating System. Proceedings of the ASME Winter Annual Meeting, Integrated Design and Analysis of Aerospace Structures, Houston, November 30-December 5, 1975, pp. 93-112.

23. Sobieszczanski-Sobieski, J.; and Goetz, R. C.: Synthesis of Aircraft Structures Using Integrated Design and Analysis Methods - Status Report. NASA CP-2049, Oct. 1978. pp. 63-76.

24. Dovi, A. R.: ISSYS - An Integrated Synergistic Synthesis System. NASA Contractor Report 159221, Kentron International, Inc., Hampton Technical Center, Hampton, VA, February 1980.

25. Whetstone, W. D.: EISI-EAL: Engineering Analysis Language. Proceedings of the ASCE Conference on Computing and Civil Engineering, Baltimore, MD June 1980.

26. Baudet, G. M.: The Design and Analysis of Algorithms for Asynchronous Multiprocessors. Ph. D. Thesis, Department of Computer Science, Carnegie-Mellon University, 1978.

27. Schmit, L. A.; and Fox, R. L.: An Integrated Approach to Structural Synthesis and Analysis. AIAA J., Vol. 3, No. 6, pp. 1004-1112, June 1965.

28. User Guide: Relational Information Management (RIM), Report DG-IPAD-70023M, Contract NAS1-14700, Boeing Commercial Aircraft Co., 1981.

29. IPAD: Integrated Program for Aerospace Vehicle Design, Proceedings of a National Symposium, Denver, CO, September 17-19, 1980, NASA CP 2143.

TABLE I.- TABULAR REPRESENTATION OF THE HIERARCHICAL SYSTEM SHOWN IN FIGURE 5

	1,1	2,1	2,2	2,3	3,1	3,2	3,3	3,4	3,5	3,6	3,7
1,1											
2,1	1										
2,2	1										
2,3	1										
3,1		1									
3,2		1									
3,3		1									
3,4			1								
3,5				1							
3,6				1							
3,7				1							
4,1					1						
4,2					1						
4,3							1				
4,4							1				
4,5								1			
4,6									1		
4,7									1		
4,8											1
4,9											1
4,10											1

TABLE II.- CORRESPONDENCE OF THE GENERIC ENTITIES TO THE FRAMEWORK EXAMPLE SPECIFICS

	Generic entities	Particular to framework example
S	System	Framework
SS	Subsystem	Beam
QI	Interaction Quantities	End forces, M, N, T shown in Fig. 2
X	Parent system design variables	A_1, I_{y1}, A_2, I_2, A_3, I_3
Y	Subsystem design variables	detailed cross-sectional dimensions b_1, t_1, ...h
H	Parameters for the system	not included in the example
X	Parameters for subsystem i	A_i, I_{yi}
B_s	Behavior variables for the system	displacements, internal forces (end forces on beams)
B_i	Behavior variables for subsystem i	stresses, buckling critical stresses
F	Objective function for the system	mass
P_i	Objective function for subsystem i	a measure of violation of constraints in beam i
P	Constraints for the system	displacements and a measure of constraint violation in all beams, and move limits on the system design variables.
G_i	Inequality constraints for subsystem i	stress and buckling constraints in beam i
E_i	Equality constraints for subsystem i	system prescribed A_i, I_{yi} for beam i
ØAQ	Interaction analysis operator	finite-element analysis to yield displacements and internal forces (end forces) for framework
ØASW	Interaction analysis sweep operator	one application of a finite-element analysis to the framework
ØAC	Constraint operator	for framework, it computes displacement constraints; for beam i, it computes beam constraints G and E for given A_i, I_{yi} and end forces
ØOBJ	Objective function operator	computes the framework mass

TABLE II.- CONCLUDED

ØOPT2	Optimization operator for beam i	⟷	any optimization procedure capable of solving for beam i $\min_{\{b_1, t_1 \ldots h\}} P_i(b_1, t_1, \ldots h, A_i, I_{yi})$ STO $I_{yi}(\{b_1, t_1 \ldots h\}) = I_{yi}$ $A_i(\text{---''---}) = A_i$
ØOPT1	Optimization operator for system	⟷	any optimization procedure capable of solving for framework: $\min_{\{A_1 \to I_{y3}\}} F(A_1, I_{y1} \ldots)$ STO $P = P_s + \Sigma P_i = 0$
ØSEN	Optimum sensitivity analysis operator for beam i	⟷	computes $\partial/\partial A_i$, $\partial/\partial I_{yi}$ for $\bar{P}_i$, and $\{\bar{b}_1 \ldots \bar{h}\}$
ØSUBSW	Subsystem elimination sweep operator	⟷	repeats optimization and optimum sensitivity operators for each of the three beams of the framework
ØTER	Termination operator	⟷	terminates when: 1. the framework mass converged 2. all constraints for the framework and each beam are satisfied

TABLE III.- CORRESPONDENCE OF THE GENERIC ENTITIES TO THE AIRFRAME EXAMPLE SPECIFICS

	Generic entities		Specifics of the airframe example
S	System	⟷	Airframe
SS	Subsystem	⟷	Wing; fuselage
QI	Interaction Quantities	⟷	Forces ($F_1 \to F_5$) at the wing-fuselage joints (Fig. 3b)
X	Parent system design variables	⟷	$K_{ij}^{(w)}$, $W^{(w)}$, $K_{ij}^{(f)}$, $W^{(f)}$
Y	Subsystem design variables	⟷	detailed cross-sectional dimensions (Fig. A4(c))
H	Parameters for the system	⟷	not included in the example
X	Parameters for a subsystem	⟷	$K_{ij}^{(w)}$, $W^{(w)}$, $K_{ij}^{(f)}$, $W^{(f)}$
B	Behavior variables	⟷	for the system: flutter speed; for each subsystem: stress, local buckling stress
F	Objective function for the system	⟷	structural mass
P_i	Objective function for subsystem i	⟷	a measure of violation of constraints in wing (i=1), fuselage (i=2)
P	Constraints for the system	⟷	displacements and first natural frequency, a measure of violation of constraints in wing and fuselage, move limits on the system design variables
G_i	Inequality constraints for subsystem i	⟷	stress and buckling constraints
E_i	Equality constraints for subsystem i	⟷	system prescribes $K_{ij}^{(w)}$, $W^{(w)}$, and $K_{ij}^{(f)}$, $W^{(f)}$ for wing and fuselage, respectively
ØAQ	Analysis operator	⟷	substructuring analysis to yield forces on the joints (Fig. A3(b))
ØASW	Interaction analysis sweep operator	⟷	as above (in this case ØASW is the same as ØAQ)
ØAC	Constraint operator	⟷	for airframe it computes displacement and frequency constraints; for wing and fuselage, it computes G and E for given $K_{ij}^{(w)}$, $K_{ij}^{(f)}$ and the joint forces $F_1 \to F_5$

TABLE III.- CONCLUDED

ØOBJ	Objective function operator	computes structural mass of airframe
ØOPT2	Optimization operator for subsystem i	any optimization procedure capable of solving for wing (i=1), fuselage (i=2) $\min_{\{y^{(i)}\}} P(\{y^i\},\{K_{pq}^{(i)}\}, W^{(i)})$ STO $K_{pq}^{(i)}(\{y^{(i)}\}) = K_{pq}^{(i)}$ $W^{(i)}(\{y^{(i)}\}) = W^{(i)}$
ØOPT1	Optimization operator for system	any optimization procedure capable of solving $\min_{\{W^{(w)},W^{(f)}\}} F(W^{(w)}, W^{(f)})$ STO $P = P_s + \Sigma P_i = 0$
ØSEN	Optimum sensitivity analysis operator for subsystem i	computes $\partial/\partial K_{pq}^{(i)}$, $\partial/\partial W^{(i)}$ for $\bar{P}_i$ and $\{\bar{y}^{(i)}\}$ for wing (i=1) and fuselage (i=2)
ØSUBSW	Subsystem elimination sweep operator	executes optimum sensitivity operators for wing and fuselage
ØTER	Termination operator	terminates when: 1. the airframe mass converged 2. all constraints for airframe and each beam are satisfied

TABLE IV.- CORRESPONDENCE OF THE GENERIC ENTITIES FOR THE AIRCRAFT EXAMPLE SPECIFICS

	Generic entities		Entities particular to the aircraft example
S	System	⟷	1.1 Aircraft
SS	Subsystems	⟷	2.1 Airframe, 3.1 Wing, 3.2 Fuselage 2.2 Propulsion (engines)
QI	Interaction Quantities	⟷	1. Aircraft/Airframe: Wing area S, aspect ratio AR, structural mass of airframe, $W^{(s)}$ QI between airframe and wing and fuselage; see the previous example 2. Aircraft/Propulsion: takeoff thrust T_o, engine mass $W^{(p)}$, engine-in-nacelle drag coefficient $C_D^{(p)}$, cruise altitude h_{cr}, cruise Mach number M_{cr}, fuel consumed in cruise $W^{(j)}$
RQI	Reverse interaction quantities	⟷	Aircraft/Airframe: change of the aerodynamic shape due to the elastic deformation of airframe
X	System design variables	⟷	for aircraft: S, AR, $W^{(s)}$, T_o, $W^{(p)}$, $W^{(j)}$, $C_D^{(p)}$, h_{cr}, M_{cr}
Y	Subsystem design variables	⟷	for airframe: wing and fuselage structural masses $W^{(w)}$ and $W^{(f)}$, elements of wing-fuselage interaction stiffness matrix $K_{ij}^{(w)}$ and $K_{ij}^{(f)}$, detailed dimensions of wing and fuselage structures as in the previous example. for propulsion: detailed design variables of engine, e.g., number of the compressor stages
H	Parameters for the system	⟷	prescribed range
X	Parameters for subsystem	⟷	2.1 Airframe: S, AR, $W^{(s)}$ 2.2 Propulsion: T_o, $W^{(p)}$, $W^{(j)}$, $C_D^{(p)}$, h_{cr}, M_{cr}

TABLE IV.- CONTINUED

B	Behavior variables	for the system payload and range for subsystem 2.1 Airframe: see the previous example for subsystem 2.2 Propulsion; one example is turbine flow temperature
F	Objective function for the system	maximum payload
P_i	Objective function	for subsystem 2.1 Airframe: a measure of violation of constraints imposed on the subsystem behavior variables (see the previous example for subsystem 2.2 Propulsion; a measure of violation of constraints imposed on the subsystem behavior variables
P	Constraints for the system	one example: takeoff field length
G	Inequality constraints	for subsystem 2.1 Airframe: see the previous example for subsystem 2.2 Propulsion; one example is the temperature of the turbine blades
E	Equality constraints	for subsystem 2.1 Airframe; one example is the system-prescribed wing area for subsystem 2.2 Propulsion: one example is the system-prescribed takeoff thrust
ØAQ	Interaction analysis operator	for the system: aircraft analysis for performance and loads for airframe subsystem: substructuring analysis including aerodynamic loads
ØASW	Interaction analysis sweep operator	one application of each of the above two operators
ØAC	Constraint operator	for the aircraft system: computation of constraints such as takeoff field length for the airframe subsystem; see the previous example for engine, one example is the engine thermodynamic analysis
ØOBJ	Objective function operator	computes the fuel consumed for a given mission

TABLE IV.- CONCLUDED

ØOPT2	Optimization operator for subsystems ⟷	for airframe subsystem 2.1: any optimization procedure capable of minimizing airframe constraint violation subject to the airframe equality constraint (See the previous example for this operator for wing and fuselage.) for propulsion subsystem 2.2: any optimization procedure capable of minimizing the engine constraint violation subject to the engine equality constraints
ØOPT1	Optimization operator for the system ⟷	any optimization procedure capable of solving $\min F(S, AR, W^{(s)}, T_o, W^{(p)}, W^{(j)}, C_D^{(p)}, h_{cr}, M_{cr})$, STO $P = P_s + \sum_i P_i$
ØSEN	Optimum sensitivity analysis operator for subsystem i ⟷	for airframe subsystem 2.1: computes partial derivatives such as $\partial/\partial S$ $\partial/\partial AR$ for the constraint violation P_i and the airframe system variables $k_{ij}^{(f)}$ and $K_{ij}^{(w)}$ (See the previous examples for this operator for wing and fuselage.) for propulsion subsystem 2.2: computes partial derivatives such as $\partial/\partial T_o$ and $\partial/\partial h_{cr}$ of the engine constraint violation P_i
ØSUBW	Subsystem elimination sweep operator ⟷	executes ØSEN operator for wing, fuselage, airframe, and propulsion subsystems
ØTER	Termination operator ⟷	terminates when: 1. the payload converged to a maximum 2. all constraints for the aircraft system and its subsystems are satisfied

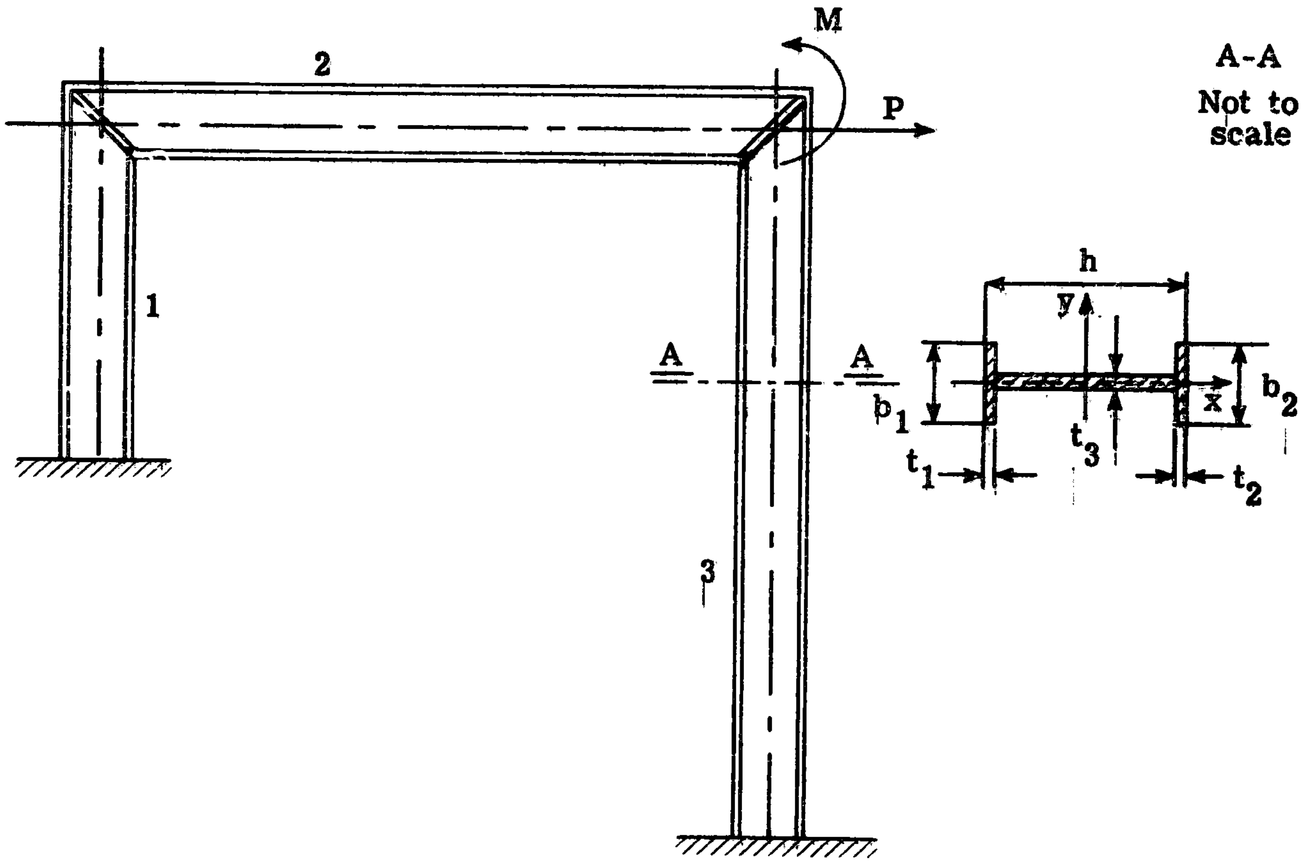

Figure 1.- A portal framework.

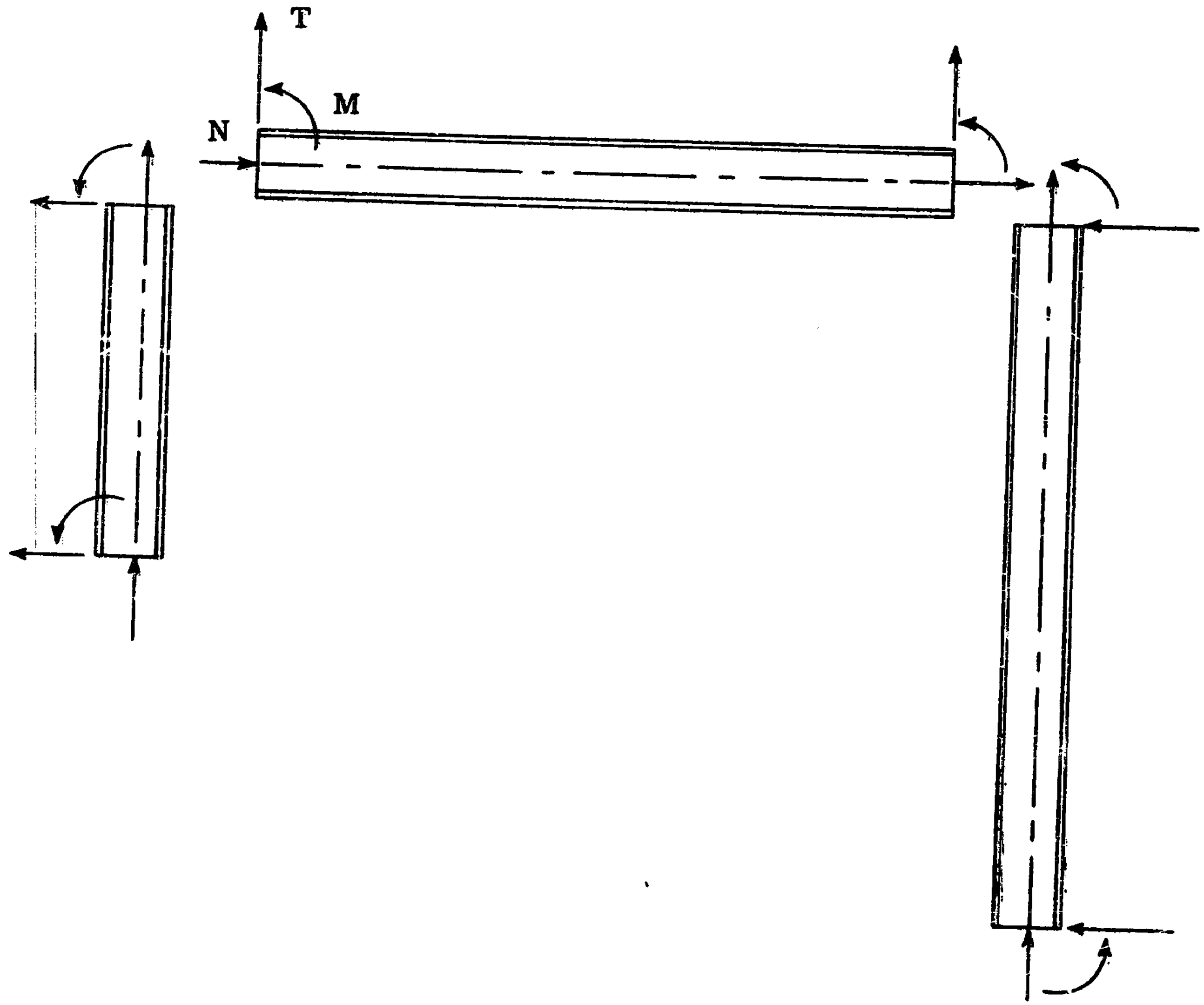

Figure 2.- Interaction quantities in a framework system are the beam-end forces (internal forces) N, T, and M.

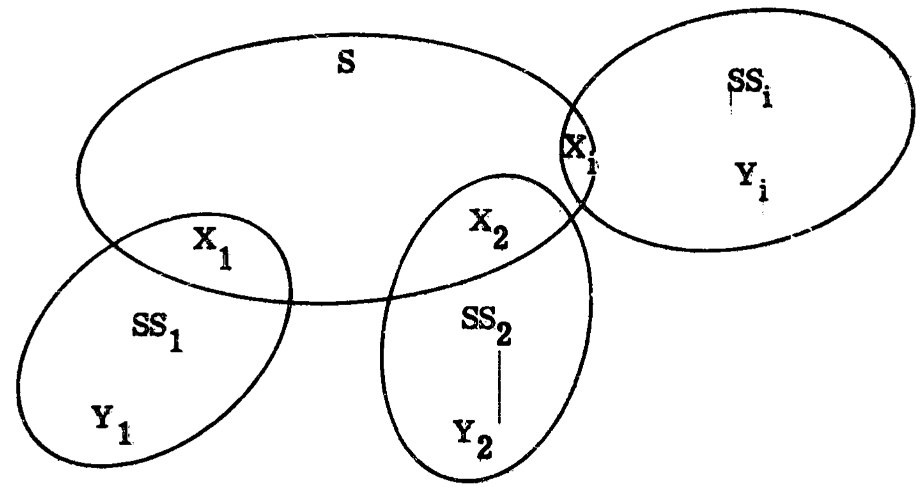

Figure 3.- A Venn diagram for a two-level system hierarchy of design variables.

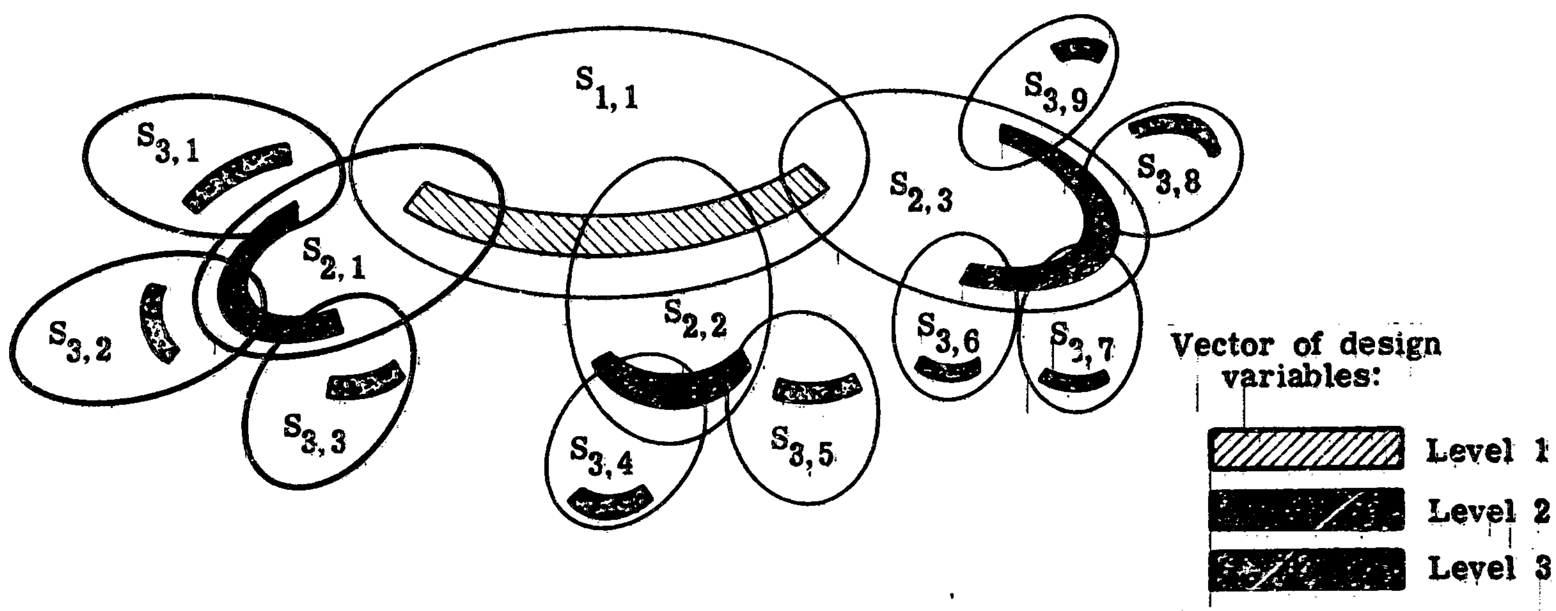

Figure 4.- A Venn diagram for a three-level system hierarchy of design variables.

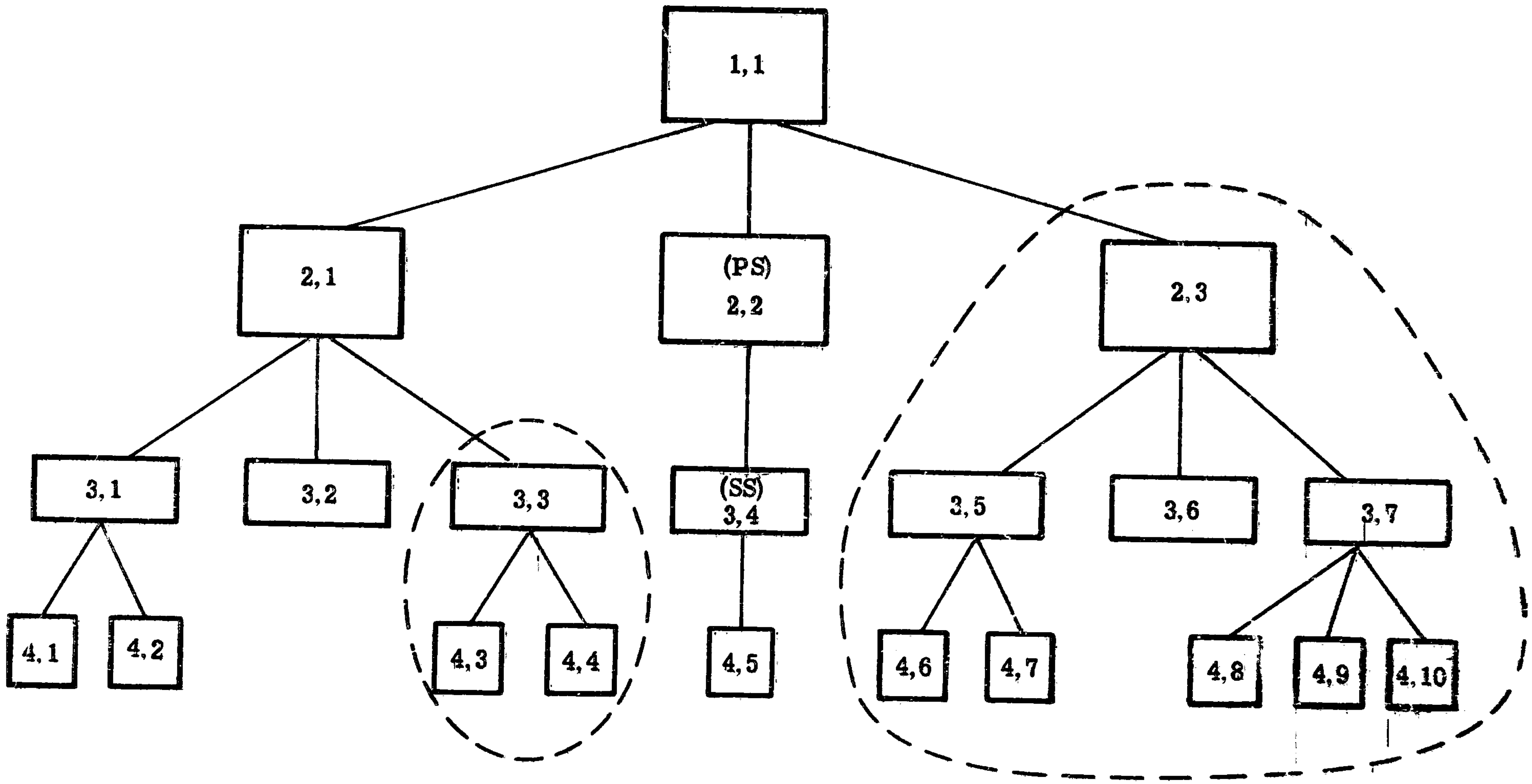

Figure 5.- A hierarchy diagram for an example of a four-level system.

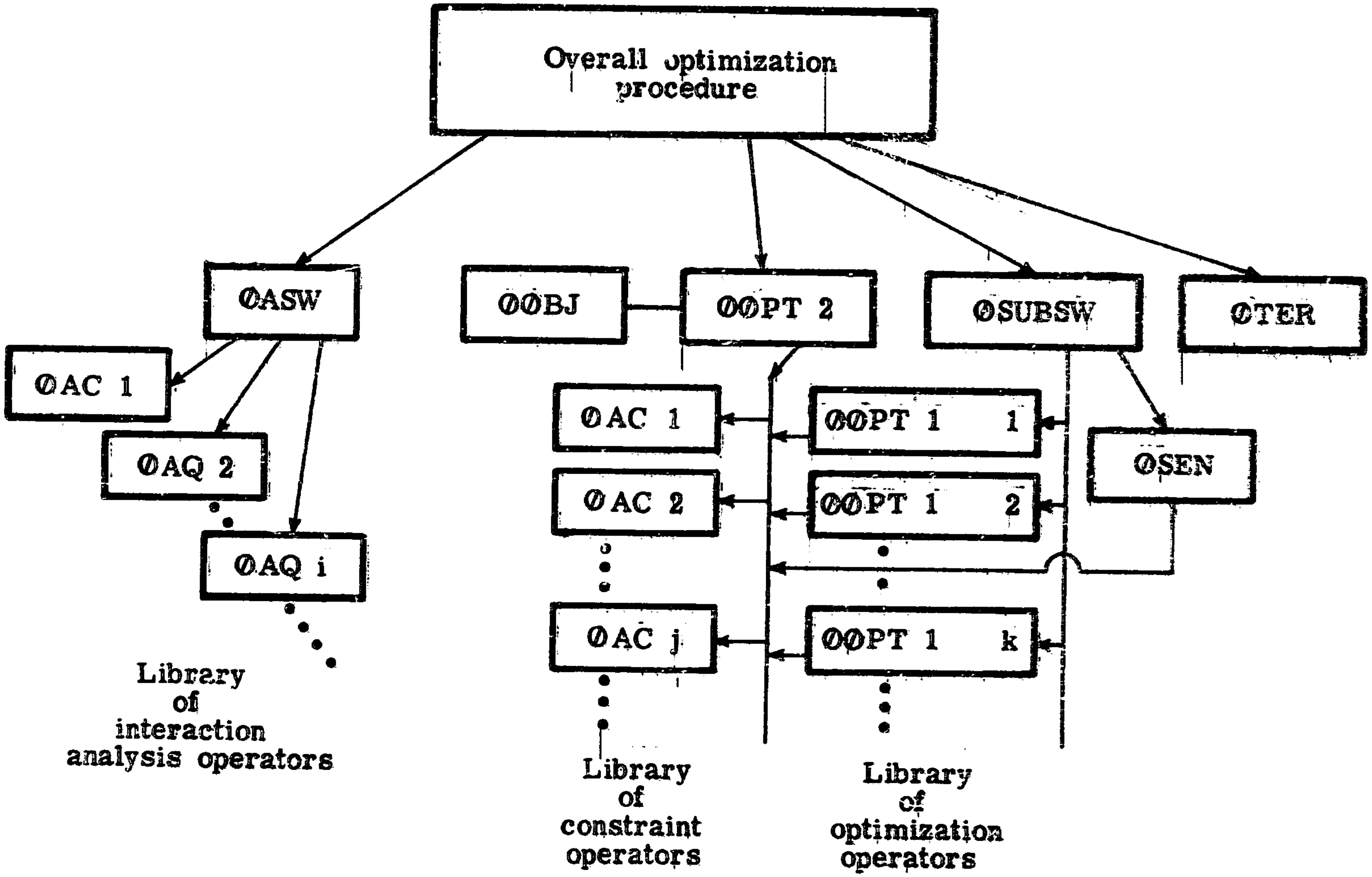

Figure 6.- Block diagram of the overall optimization procedure. Arrows indicate calling hierarchy, e.g., operator ØSEN which calls, selectively, operators ØAC j.

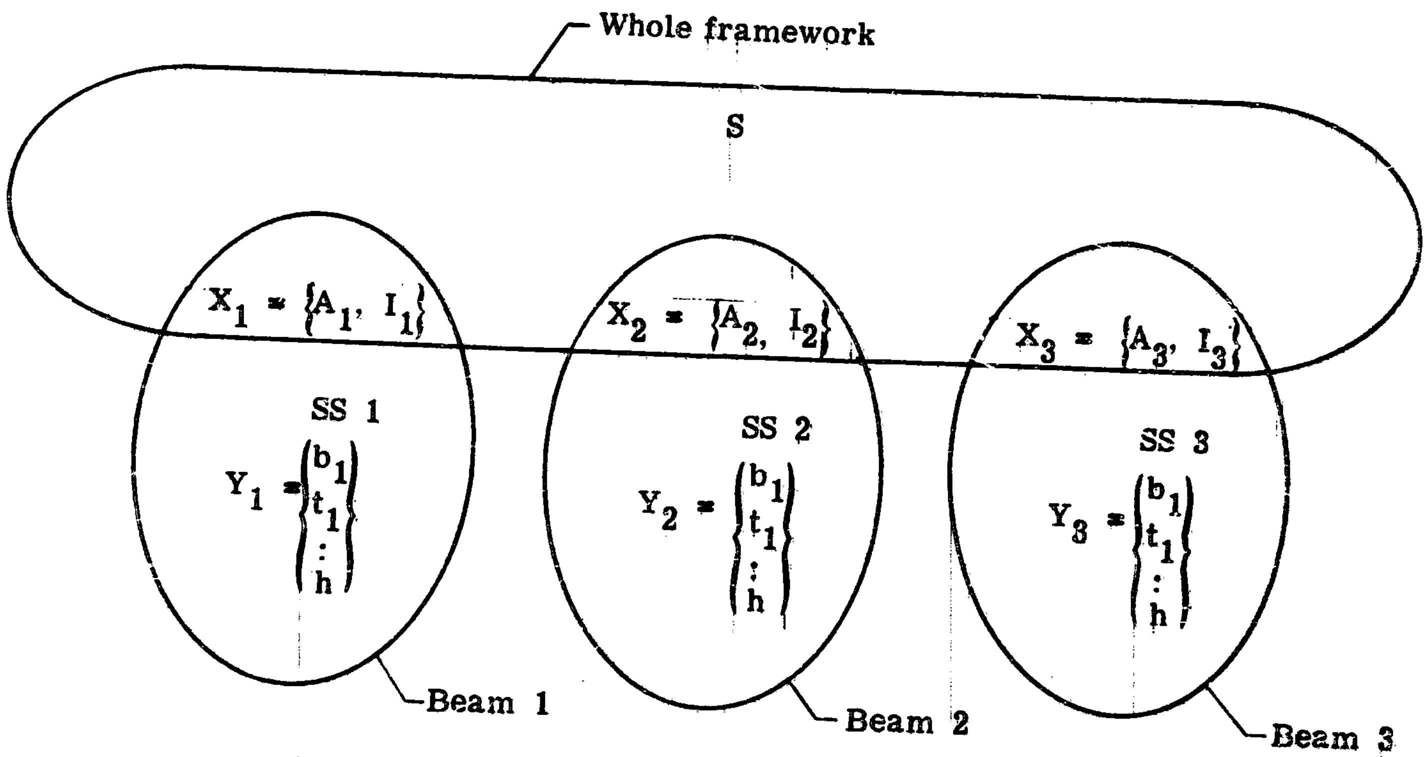

Figure A1.- Venn diagram for a framework system.

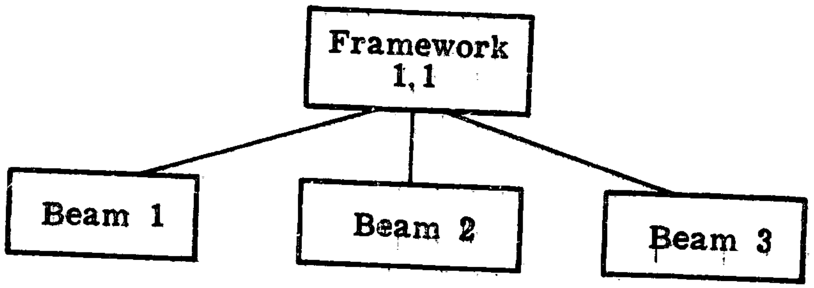

Figure A2.- Hierarchy diagram for a framework system.

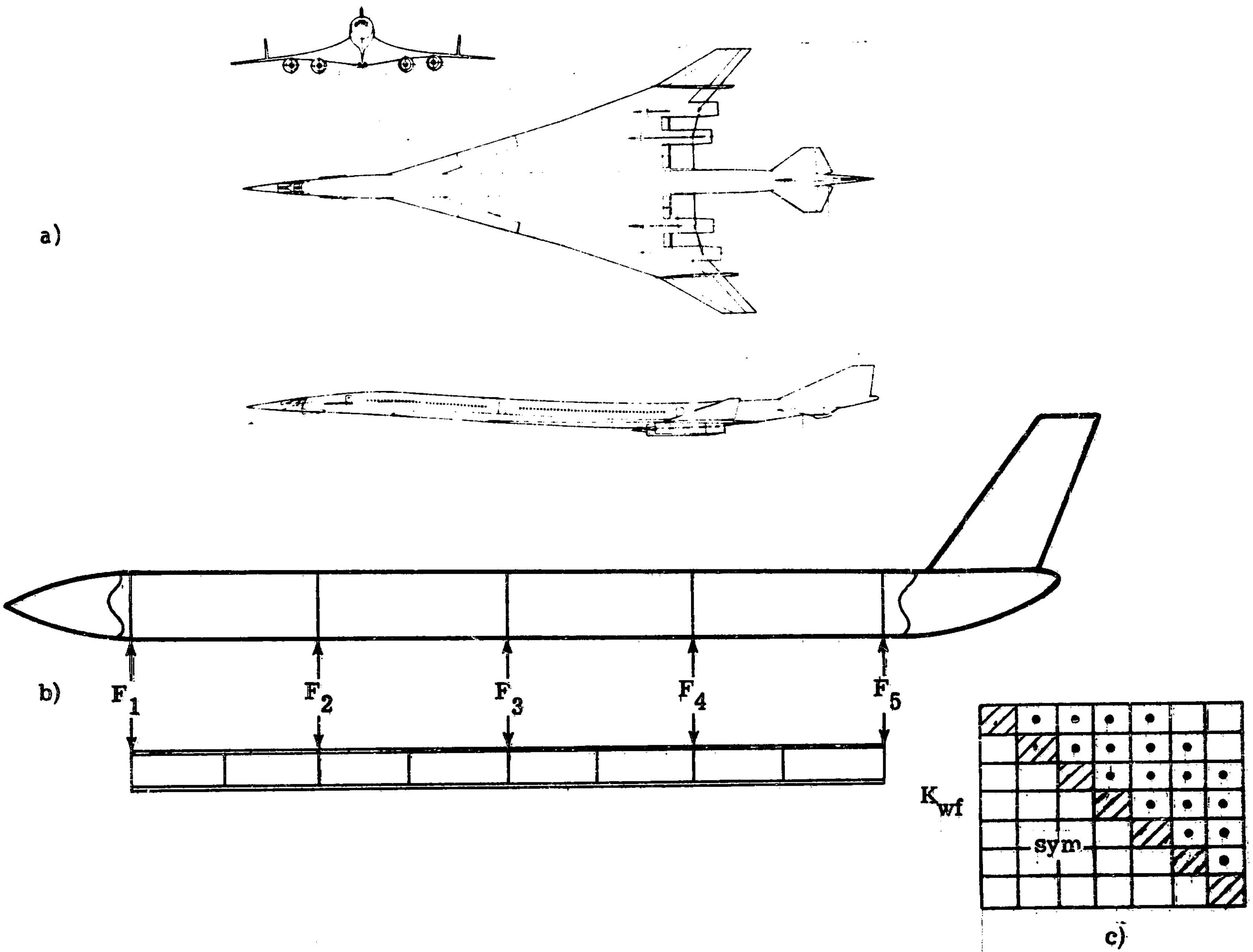

Figure A3.- Airframe example: (a) airframe, (b) interaction forces between wing and fuselage, (c) interaction stiffness matrix topology.

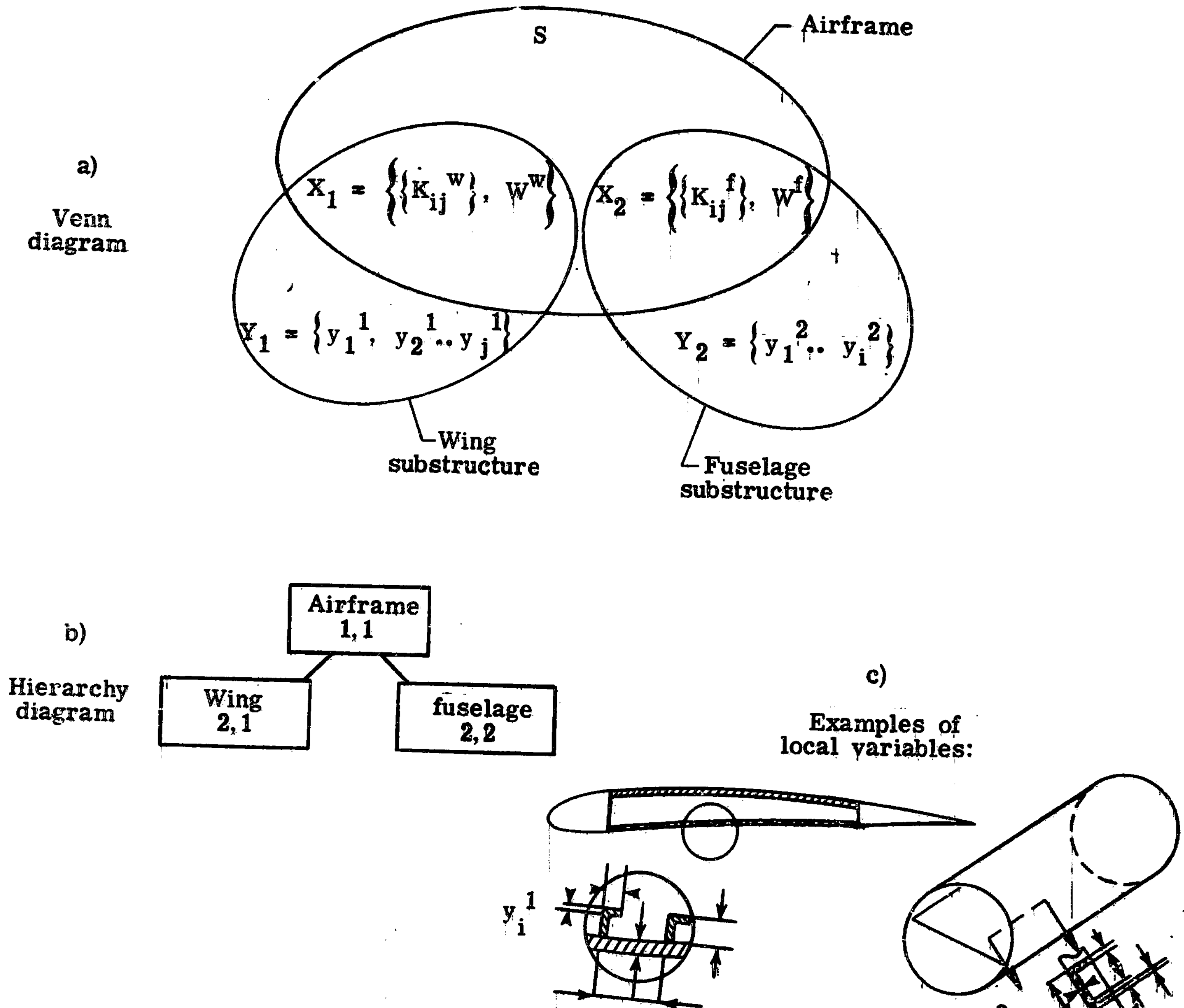

Figure A4.- Airframe diagrams and variables.

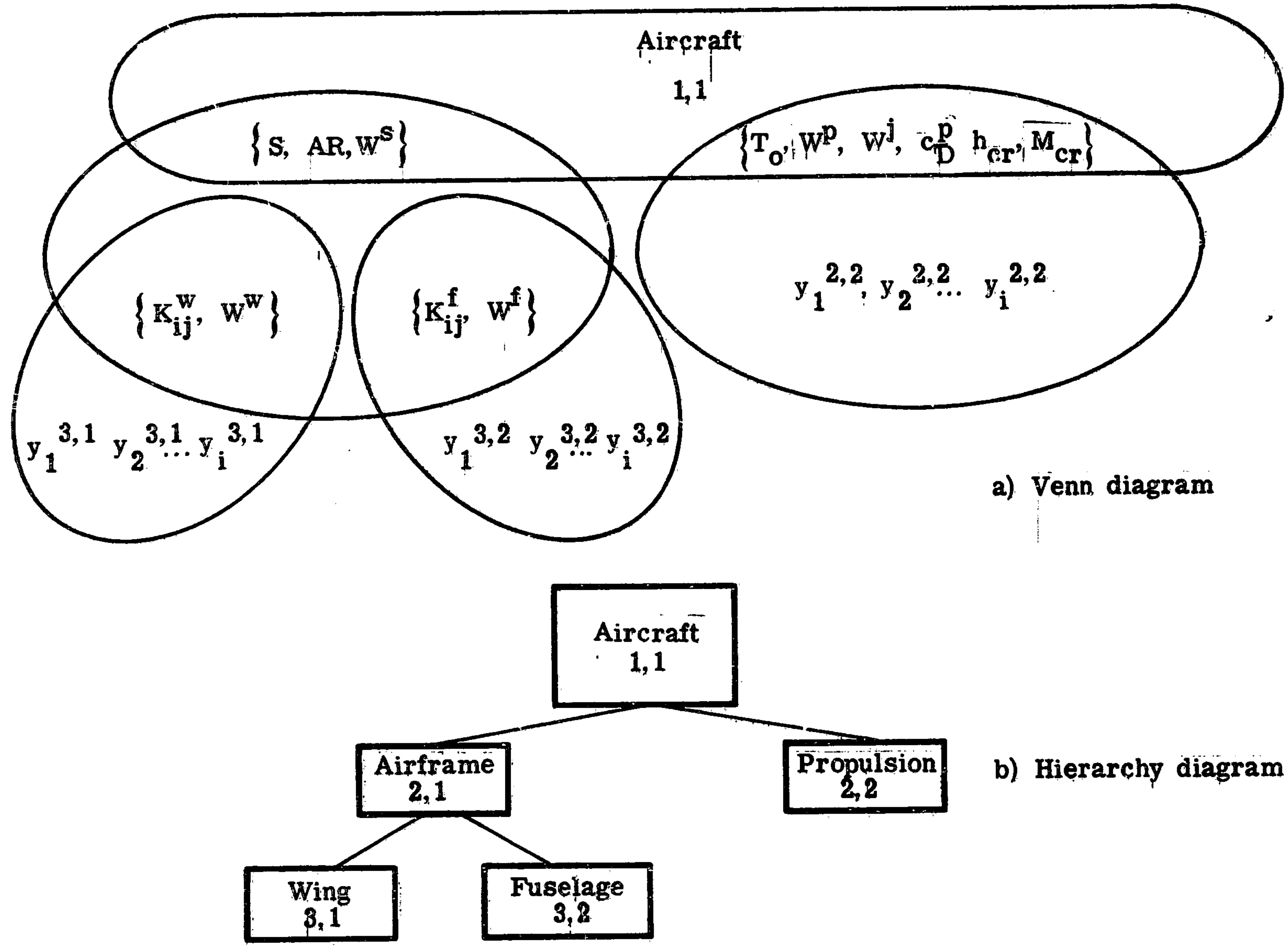

Figure A5.- Venn and hierarchy diagrams for an example of aircraft viewed as an engineering system.

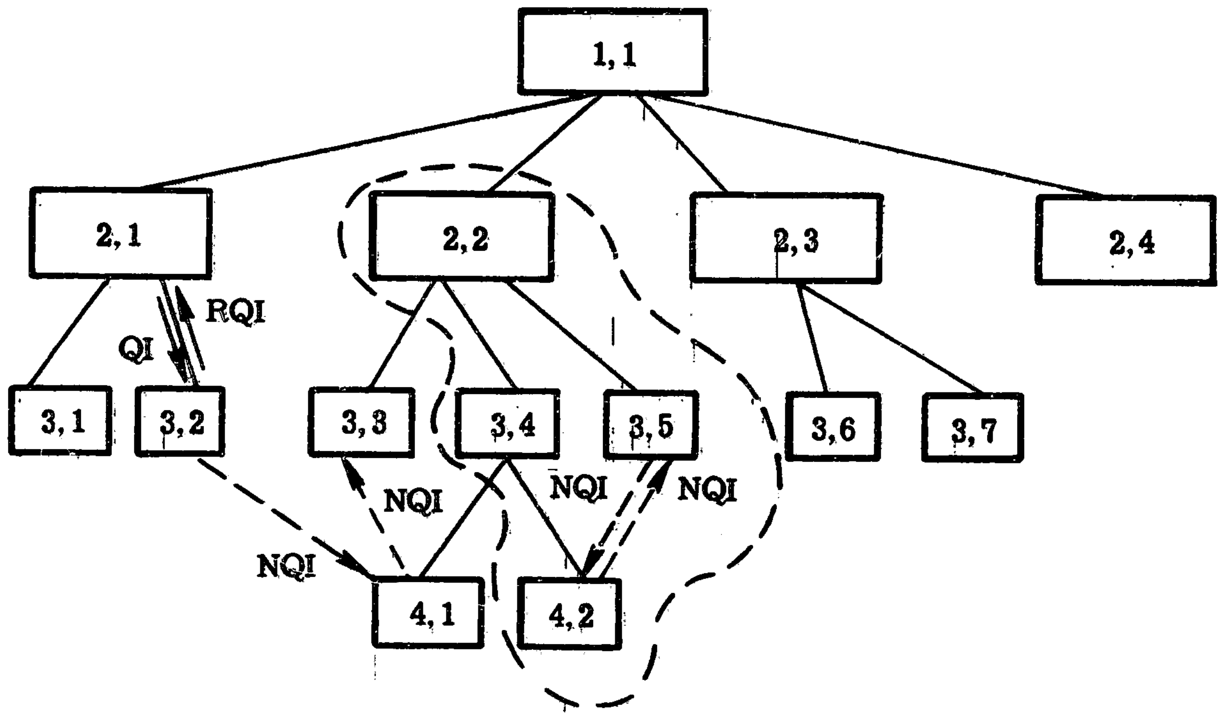

Figure B1.- An example of a network system: Interactions QI, RQI, and NQI exist among the nodes.