Improved Multilevel Optimization Approach for the Design of Complex Engineering Systems

Jean-Francois M. Barthelemy*
Virgina Polytechnic Institute and State University, Blacksburg, Virginia
and
Michael F. Riley†
PRC Kentron, Hampton, Virginia

Multilevel optimization methods are being considered for the design of complex systems on distributed networks of computers or even parallel processors. An obstacle to the use of multilevel methods is that they can be computationally expensive because of the cycling necessary to account for the coupling between the subproblems. This research effort aims at increasing the efficiency of multilevel optimization by adapting two techniques widely used in one-level optimization: constraint approximation and temporary constraint deletion. These improvements are tested on 3, 10, and 52 bar planar trusses. The results show that for larger problems (approximately 100 variables and larger), the cost of analysis dominates the total cost so that multilevel optimization is no more expensive than one-level optimization. If parallel processing is used or the analysis process itself is decomposed, then multilevel optimization stands to become more economical than one-level optimization.

Introduction

MULTILEVEL optimization is a promising approach to solving today's large design problems. In this approach, large optimization problems are broken in smaller problems that are solved in an iterative fashion.

A general approach for applying multilevel optimization to a large class of nonlinear design problems is described in Ref. 1. This approach is known as the linear decomposition approach. In it, the design problem is decomposed into a hierarchy of subproblems. At the top level, a subproblem optimizes a simplified model that describes the overall behavior of the system. At the lower levels, subproblems optimize increasingly detailed representations of subsystems. The method effectiveness has been demonstrated on two- and three-level structural framework design problems.[2,3] Its applicability of multidisciplinary design problems, as well as the design of a composite multicell beam, has been examined. The method has been adapted to penalty function optimization techniques[5] where improved efficiency was demonstrated by limiting the optimization to one minimization step for each subproblem within each cycle.

Among the advantages of the linear decomposition method are that 1) it respects the design process modularity and fits very well the typical design group organization; 2) it accounts rationally for the coupling between the different subsystems being designed and therefore facilitates the tradeoffs in the design process; and 3) it can be used on networks of distributed computers or parallel processors. It can be argued that one would consent to losing some computer efficiency in order to gain the additional design freedom. However, this is not to say that the issue of computer efficiency is to be ignored altogether. The only attempt to look at the linear decomposition method efficiency is Ref. 5 where the author showed that by careful implementation he could reduce the computing time requirements by a factor of two or three with respect to one-level solutions.

One of the elements that influence the cost of multilevel optimization methods is the necessity to cycle through the optimization of the various subproblems in order to account for the coupling. In the limit, if a problem is decomposed into a number of subproblems that are totally uncoupled, there is no need to cycle through the optimization of the various subproblems to coverage to the optimum design of the system and it is very likely that the multilevel optimization approach will require less computer time than the one-level counterpart (for reasonably sized problems). On the other hand, if a problem is decomposed in a number of highly coupled subproblems, a lot of cycling may be required in order to converge to the optimum design; conceivably, convergence may be practically impossible if an inappropriate decomposition is chosen. Therefore, the benefit obtained by replacing a larger optimization problem by sequences of smaller subproblems solved in an iterative fashion can easily be offset by the requirements to repeatedly reoptimize the subsystems.

This paper examines two approximation techniques commonly used for large one-level optimization problems and investigates their applicability to the multilevel optimization approach. The first technique entails replacing the initial generally implicit design problem by a sequence of inexpensive explicit design problems. In this effort, the convex approximation[6,7] will be used. Using approximations, the cost of analyzing the system being designed becomes independent on the number of multilevel cycles required to converge the design and should be the same as for a one-level approach also using approximations. The second technique consists of reoptimizing only those subproblems that have violated critical or nearly critical constraints. This technique is somewhat analogous to the constraint deletion technique[8] used in one-level optimization.

The proposed algorithm is presented first and one- and two-level solutions to sequences of approximating subproblems are discussed. The algorithm is applied to the minimization of the volume of planar trusses under multiple-load cases and constraints on stresses, displacements, local buckling, local crippling, as well as various constraints on geometry. Results are given for 3, 10, and 52 bar trusses.

Presented as Paper 86-0950 at the AIAA/ASME/ASCE/AHS 27th Structures, Structural Dynamics and Materials Conference, San Antonio, TX, May 19–21, 1986; received Oct. 6, 1986; revision received June 26, 1987.

*Research Associate (currently, Aerospace Engineer, Interdisciplinary Research Office, NASA Langley Research Center, Hampton, VA). Member AIAA.

†Programmer/Analyst.

Development

Functional Form for the Problem Solved

The design problems considered belong to the following class of *nonlinear programming problems* (NP):

Find $Y, X_1, X_2, \ldots, X_{n_s}$ so that $F(Y)$ is minimum and

$$g(Y) \leq 0$$

$$g_s(Y, X_s) \leq 0, \quad s = 1, n_s$$

$$Y^l \leq Y \leq Y^u$$

$$X_s^l \leq X_s \leq X_s^u, \quad s = 1, n_s \tag{1}$$

This functional form is common.[1-5] In the formulation, the objective function F and the vector of constraints g depend exclusively on a subset Y of the design variables of the problem (Y and g may be referred to as global variables and constraints). Also, the vector of constraints g_s depends exclusively on the variable vectors Y and X_s (X_s and g_s may be referred to as local variables and constraints). All the functions F, g, and g_s are nonlinear in general. The superscripts l and u identify lower and upper bounds on the variables. The variables are concatenated in one vector as

$$Z = [Y, X_1, X_2, \ldots, X_{n_s}] \tag{2}$$

The initial design for NP is denoted Z^o and the final design Z^*.

The type of problem described by Eq. (1) may have many valid solutions. Indeed, imagine that none of the constraints in g_s is active (i.e., satisfied as equalities) at the final design. All values of X_s that satisfy the strict inequality constraints $g_s(Y, X_s) < 0$ will be solutions of the problem yielding the same final objective function value. If at the optimum design a subproblem is such that there is one l so that $g_{sl} = 0$, it will be called locally constrained; if $g_{sl} < 0$ for all l, it will be called globally constrained.

Convex Approximation

In most cases, NP is an implicit and nonlinear problem. To reduce the cost associated with repetitive calculation of the objective and the constraints, NP is replaced by a sequence of *convex approximating problems* (CP_i):

Find $Y, X_1, X_2, \ldots, X_{n_s}$ so that $\bar{F}_i(Y)$ is minimum and

$$\bar{g}_i(Y) \leq 0$$

$$\bar{g}_{si}(Y, X_s) \leq 0, \quad s = 1, n_s$$

$$Y^l \leq Y \leq Y^u$$

$$X_s^l \leq X_s \leq X_s^u, \quad s = 1, n_s \tag{3}$$

The overbars indicate that the functions are approximations. The initial design for CP_i is Z_i^o and the final one Z_i^*. In this application, a convex approximation[6] has been chosen. For a function $g(X)$, which is at least once differentiable, the convex approximation at X^o reads

$$\bar{g}(X) = g(X^o) + \sum_{+} \frac{\mathrm{d}g}{\mathrm{d}X_m}(X_m - X_m^o) + \sum_{-} \frac{\mathrm{d}g}{\mathrm{d}X_m} \frac{X_m^o}{X_m}(X_m - X_m^o) \tag{4}$$

where $\sum_{+}$ represents the summation of all the terms with positive $\mathrm{d}g/\mathrm{d}X_m$, while $\sum_{-}$ represents the summation of all the terms with negative $\mathrm{d}g/\mathrm{d}X_m$. For positive design variable values, this approximation is convex[7] and also the most conservative[6] of all those based on first-order derivative information and using terms linear in the variables or their reciprocal. Returning to CP_i, the approximations are generated at the initial design point Z_i^o, which is the solution Z_{i-1}^* of CP_{i-1}.

One-Level Solution

The traditional approach to solving NP by sequences of approximations proceeds then as follows. An initial design Z_1^o must be given. At each pass i, a problem analysis is performed where the objective function and the constraints, as well as their first derivatives with respect to Z, are calculated at the initial point Z_i^o. The approximations of Eq. (3) are constructed and the resulting nonlinear programming problem is solved using an appropriate algorithm yielding Z_i^*. The convergence of the process is checked and, if it is not achieved, the process is restarted taking as a new initial point the previous optimum design. If convergence occurs at pass I, then $Z^* = Z_I^*$.

Note that it may be necessary to start the design process at a point that is infeasible. Since the approximation used is conservative, it may reduce the size of the feasible domain and it is possible that the initial approximation has no feasible solution if generated at a strongly infeasible initial design. If it is desired to use a nonlinear programming algorithm that is not designed to handle infeasible designs, it may become necessary to use a constraint relaxation technique, as outlined in Appendix C.

Two-Level Solution

If it is desired to solve NP using decomposition, then, at each pass i, CP_i may be solved using decomposition.[1] The local vectors of constraints $\bar{g}_{si}$ are first replaced by scalar cumulative measures of constraint violation[9] (*cumulative constraint*), as

$$C_{si} = \frac{1}{\rho} \ln\left[\sum_{l=1}^{n_s} \exp(\rho \bar{g}_{sli})\right] \tag{5}$$

This transformation reduces the amount of constraints to be dealt with by the optimizer. In general, however, all of the constraints must be evaluated by the analysis program. As shown in Appendix B, it is conservative with respect to the constraints $\bar{g}_{sli}$ and it preserves the convexity of subproblems CP_i. Also given in Appendix B are recommendations as to the choice of the user-selected parameter ρ.

Subproblem CP_i is decomposed in $n_s + 1$ smaller subproblems that are solved in repetitive cycles. At pass i, cycle j, there are n_s *bottom level* (or local level) *convex subproblems* BC_{sij} solved for fixed $Y = Y_{ij}^o$:

Find X_s so that $C_{sij}(Y_{ij}^o, X_s)$ is minimum and

$$X_s^l \leq X_s \leq X_s^u, \quad s = 1, n_s \tag{6}$$

Subproblem BC_{sij} is unconstrained and is devoted to finding the value of the local variable X_s that globally minimizes the violation of those local constraints exclusively depending on X_s.

The solution of this nonlinear programming problem is X_{sij}^* and the optimum constraint violation is C_{sij}^*, both of which remain functions of Y_{ij}^o. The optimum constraint violation for subproblem s can be estimated in terms of Y by

$$\tilde{C}_{sij}(Y) = C_{sij}^*(Y_{ij}^o) + \sum_{+} \frac{\mathrm{d}C_{sij}^*}{\mathrm{d}Y_m}(Y_m - Y_{mij}^o) + \sum_{-} \frac{\mathrm{d}C_{sij}^*}{\mathrm{d}Y_m} \frac{Y_{mij}^o}{Y_m}(Y_m - Y_{mij}^o) \tag{7}$$

where $\mathrm{d}C_{sij}^*/\mathrm{d}Y_m$ is obtained by performing the sensitivity analysis of the optimum design of BC_{ij} with respect to the components of $Y = Y^o_{ij}$.[10,11] Note that Eq. (7) is of the same form as the convex approximation of Eq. (4); however, the reference is Y^o_{ij}, the initial design for cycle j as opposed to Y^o_i, the initial design for pass i. Finally, there is a *top level* (or global level) *convex subproblem* TC_{ij} defined as:

Find Y so that $\bar{F}_i(Y)$ is minimum and

$$\bar{g}_i(Y) \leq 0$$

$$\tilde{C}_{sij}(Y) \leq 0, \quad s = 1, n_s$$

$$Y^l \leq Y \leq Y^u$$

$$(1-\beta) Y^o_{ij} \leq Y \leq (1+\beta) Y^o_{ij}) \tag{8}$$

In this subproblem, a global design is chosen that minimizes the objective function while satisfying the global constraints and, to the first-order, the local constraints. Move limits are added (β is the move limit parameter) to preserve the validity of the approximations built in Eq. (7) and to control the oscillations resulting from the discontinuities of dC^*_{sij}/dY_m should such oscillations develop.[4]

The algorithm proceeds as follows. An initial design $Z^o_1 = Z^o_{11}$ must be given. Each pass i starts by analysis of the top and bottom level subproblems. This entails calculation of functions F, g, and g_s ($s = 1, n_s$) as well as their derivatives with respect to Z at Z^o_i. Then begins a series of multilevel cycles within pass i. Within cycle j, each BC_{sij} is optimized for fixed value of Y^o_{ij}, yielding X^*_{sij}, C^*_{sij} [see Eq. (6)], and dC^*_{sij}/dY_m; the sensitivity of the bottom level design to the top level design is thus obtained.[11] Then TC_{ij} is optimized and Y^*_{ij} found. Unless convergence is obtained, the cycle is restarted taking $Z^o_{ij} = Z^*_{ij-1}$. Should convergence occur at cycle J, the optimum for the current pass is $Z^*_i = Z^*_{iJ}$, the optimum constraint violation for the various subproblems are $C^*_{si} = C^*_{siJ}$ ($s = 1, n_s$); then a new pass is started with $Z^o_i = Z^*_{i-1}$. The process continues until the design has converged at pass I. The final design is given by $Z^* = Z^*_I$, the optimum design of the last pass.

To further reduce the computational cost, a subproblem disconnect feature is added. After the first cycle within one pass, those subproblems whose constraints have remained far from being violated ($C^*_{sij} \ll 0$) are ignored from the remainder of the pass, thereby avoiding unnecessary optimization and optimum sensitivity analysis. This approach is similar to the constraint deletion technique used in one-level optimization.

As explained at the end of the previous section, if its initial design is infeasible, subproblem TC_{ij} may have to be solved using the constraint relaxation technique of Appendix C.

Applications

Planar Truss Design

The algorithm described above has been applied to the problem of minimization of the volume of planar trusses built of cylindrical members. Several load cases are considered and there are constraints on nodal displacements, member stresses, and local buckling and crippling, as well as some geometrical constraints. The design variables include member cross-sectional area and mean diameter. Several members may be linked and have the same cross section. Details on the constraint formulation are given in Appendix A. The solution of the one-level approach is obtained using ADS[12] and its combination of a usable-feasible direction algorithm and polynomial interpolation with a bounds one-dimensional search.

The decomposition used includes a top level subproblem dealing with the minimization of the truss volume with respect to the member cross-sectional areas under constraints on member stresses and nodal displacements. Stresses and displacements are obtained from a custom finite-element package; the calculation of the derivatives is based on the virtual load method. Top level optimization is carried out using the same algorithm as in the one-level approach.

There is one bottom level subproblem for each different cross section in the model and several members can be designed by the same subproblem. Each subproblem has one design variable. The bottom level constraints guard against local buckling and crippling, assure a minimum member thickness-to-diameter ratio, as well as establish the upper and lower bounds on the cross-sectional diameter and thickness. Bottom level optimization is carried out using the Golden-section search subroutine ZXGSN in the IMSL library.[13]

The calculation of the derivatives for Eq. (7) warrants some explanation. As shown in Appendix A, the optimum objective function for subproblem s (C^*_s) is an explicit function of the area of the elements designed by subproblem s (A_s), but also an implicit function of all the areas of the truss A through its dependency on the stresses applied in the various load cases on the members designed by subproblem s (σ_s) as

$$C^*_s = C^*_s[A_s, \sigma_s(A)] \tag{9}$$

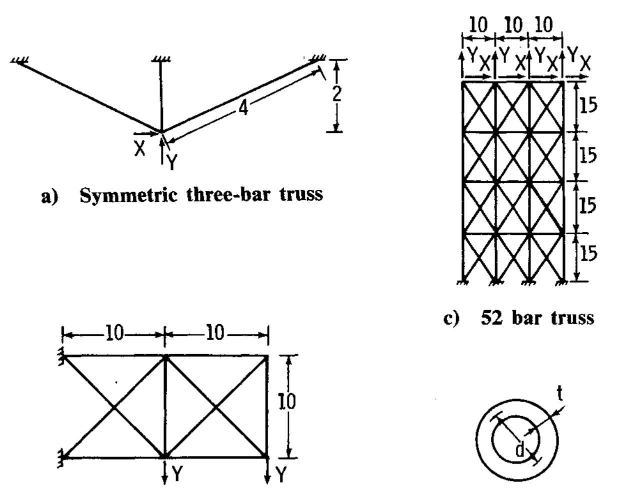
The sensitivity of that optimum to the change in the area of subproblem t is obtained by the chain rule of differentiation,

$$\frac{dC^*_s}{dA_t} = \frac{\partial C^*_s}{\partial A_t}\delta_{st} + \sum_i^{n_s} \frac{\partial C^*_s}{\partial \sigma_{si}} \frac{\partial \sigma_{si}}{\partial A_t} \tag{10}$$

where δ_{st} is the Kronecker delta, $\partial C^*_s/\partial A_t$ and $\partial C^*_s/\partial \sigma_{si}$ are the sensitivity derivatives of the optimum of subproblem s with respect to the top level parameters that directly affect it, while $\partial \sigma_{si}/\partial A_t$ are the sensitivity derivatives of the structure stresses with respect to area A_t.

Example Problems

The truss models selected are depicted in Fig. 1. The problem formulation is detailed in Appendix A. The following values of Young's modulus, crippling constant, and maximum thickness-to-diameter ratio are used for all three examples: $E_i = 2.2 \times 10^{11}$ N/m^2, $K_i = 0.2$, $D_s = 0.2$. All the initial design points have identical members defined by initial diameter d^o and initial thickness t^o. The lower and upper limits of all diameters and thicknesses are, respectively, $d_{min} = t_{min} = 1.0 \times 10^{-3}$ m and $d_{max} = t_{max} = 1.0$ m. The various runs are summarized in Table 1, in which the run numbers clearly identify the examples considered. For each



a) Symmetric three-bar truss

b) 10 bar truss

c) 52 bar truss

d) Typical cross section

Fig. 1 Example models.

Table 1 Test case description

Run	d^o, m	t^o, m	σ_{max}, N/m,	u_{max}, m
03A	1.0×10^{-1}	5.0×10^{-3}	1.0×10^{9}	—
03B	5.0×10^{-2}	1.0×10^{-3}	1.0×10^{9}	—
03C	1.0×10^{-1}	5.0×10^{-3}	3.0×10^{8}	—
03D	5.0×10^{-2}	1.0×10^{-3}	3.0×10^{8}	—
10A	1.0×10^{-1}	5.0×10^{-3}	3.0×10^{8}	—
10B	5.0×10^{-1}	2.0×10^{-2}	3.0×10^{8}	—
10C	1.0×10^{-1}	5.0×10^{-3}	3.0×10^{8}	3.0×10^{-2}
10D	5.0×10^{-1}	2.0×10^{-2}	3.0×10^{8}	3.0×10^{-2}
52A	1.0×10^{-1}	5.0×10^{-3}	3.0×10^{8}	1.0×10^{-1}
52B	5.0×10^{-2}	2.0×10^{-3}	3.0×10^{8}	1.0×10^{-1}

Table 2 Optimal volumes, m^3

Example truss	A and B	C and D
3 bar ($\times10^3$)	4.628	5.472
10 bar ($\times10^1$)	2.068	5.852
52 bar ($\times10^1$)	5.179	—

model, runs A and B (as well as runs C and D) are identical except that they originate from different starting points.

The symmetric three-bar truss of Fig. 1a is optimized under two simultaneous load cases with $(X,Y)=(2.0\times10^5, 2.0\times10^5$ N) and (0.0, 3.0×10^5 N). As shown in Table 1, runs 03A and 03B were made for high-strength material, while runs 03C and 03D were made for low-strength material. This problem has 4 variables and 20 constraints; after removing the constrained displacements, its finite-element model has only 2 degrees of freedom.

The 10 bar truss of Fig. 1b is optimized under loads $Y=5.0\times10^5$ N. Runs 10C and 10D are respectively identical to 10A and 10B, except for the addition of a limit (u_{max}, Table 1) on the vertical displacement at the lower right-hand corner. This problem has 20 variables and 60 or 61 constraints; the finite-element model has 8 degrees of freedom.

The 52 bar truss of Fig. 1c is optimized under loads of $(X,Y)=(5.0\times10^5, 1.0\times10^6$ N). The horizontal displacement of the truss upper left corner is limited to u_{max} as per Table 1. This problem has 104 variables and 313 constraints; the finite-element model has 32 degrees of freedom.

Results

The 10 examples of Table 1 were run on a CDC Cyber 170-855 computer with 377700_8 60-bit words of core memory using the NOS 2.3 operating system. A limited amount of working storage overlay and temporary storage of data on tape were required to load the necessary codes in this relatively small machine. All the optimizations reported below were terminated when the relative improvement in the objective function had been less than 1% in the last three passes. Unless otherwise specified, all the problems were run with the constraint relaxation technique of Appendix C using a ratio of 0.1:0.01 between the objective function and the penalty term. For the multilevel procedure, the convergence of the inner cycles used an adaptive tolerance varying between 10% at the beginning of the process and 0.1% toward the end to decide on the convergence of cycles within a given pass. Also, the move limit parameter β was adaptive and varied between 30% at the beginning of the process and 5% at the end; if oscillations were detected, β was initially reduced to 5% and then the pass was terminated. Finally, the Golden-section search of the local level subproblems was continued to the point where the optimum diameter was known rather accurately to within 0.00001 m, except run 52B where 0.0000001 m was necessary. This was in order to have high-accuracy sensitivity derivatives to construct Eq. (7). It must be pointed out that the determination of the appropriate combination of parameters for the algorithm (tolerances, move limits, etc.) took some experimentation

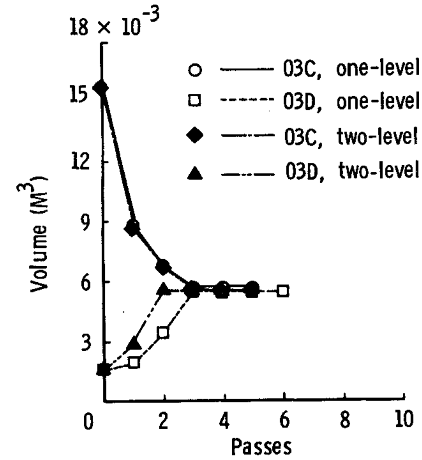


Fig. 2 Convergence history, examples 03C and 03D.

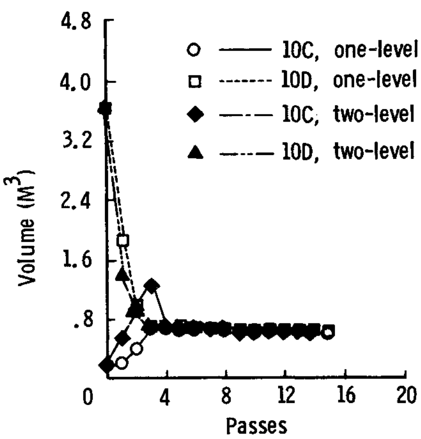


Fig. 3 Convergence history, examples 10C and 10D.

and it is expected that the proper combination remains very much problem dependent.

Table 2 gives the lowest volumes obtained for each of the five different examples. These are chosen among all solutions to the same problem, regardless of the starting point or the optimization approach, provided that they correspond to a feasible design.

Figures 2–4 depict the convergence history for one of each model, solving the same problem starting from the two different starting points and using both the one- and the two-level approaches. Figure 2 gives the convergence history for examples 03C and 03D. Despite the fact that the designs are started relatively far from the final design, both approaches converge within, at most, six passes. The convergence for the designs started from the feasible starting point (03C) is the same regardless of the approach. This is expected since the original problem is replaced by a sequence of convex subproblems, the only difference being that the inner optimization

Table 3 Typical member designs

	One level		Two level	
Solution example	10A	10B	10A	10B
	Subproblem 4—Locally constrained			
Area, $m^2 \times 10^3$	1.79	1.82	1.80	1.73
Diameter, $m \times 10$	3.09	3.11	3.10	3.08
	Subproblem 1—Globally constrained			
Area, $m^2 \times 10^3$	3.26	3.33	3.36	3.26
Diameter, $m \times 10$	2.90	1.00	5.34	5.22

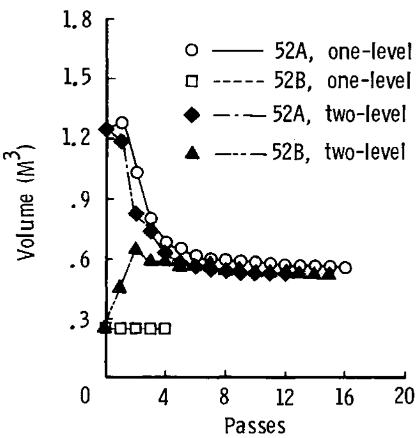


Fig. 4 Convergence history, examples 52A and 52B.

problem is solved differently. Since the constraint relaxation technique of Appendix C transforms the one- and two-level approaches differently, the designs started from the infeasible starting point should not generally follow the same path. Note that the one-level solutions appear to be slightly conservative.

Figure 3 shows similar results for examples 10C and 10D. Here again, the designs are started from very different starting points and converge within, at most, 15 passes. The one-level solution to 10D is also somewhat conservative.

Finally, Fig. 4 gives results for examples 52A and 52B where both two-level solutions converge within, at most, 15 passes; however, only one one-level solution converges in 16 passes and is still relatively far from the final solution. The one-level run started deep in the infeasible region terminates after four passes without any appreciable improvement, as each inner optimization makes very little progress and appears impeded by the nonlinearity of the convex approximations.

Table 3 compares detailed designs for two different subproblems in examples 10A and 10B. Subproblem 4 is locally constrained; that is, at the final design, at least one of its local constraints is active. Both areas and diameters are very close regardless of the initial starting point or approach used. On the other hand, subproblem 1 is globally constrained, that is, none of its local constraints is active at the optimum design. As pointed out above in the discussion on the functional form for the problem solved, there are an infinite number of local solutions for this subproblem that all satisfy the local constraints. This is clear in Table 3 where the diameter of number 1 varies from 1.00×10^{-1} to 5.34×10^{-1}. Note that the designs generated by the multilevel approach are the same. This is because the final design minimizes the constraint violation for that particular subproblem. In other words, for a globally constrained subproblem, the multilevel algorithm chooses a design that not only satisfies the local constraints, but also maximizes the margin with respect to those constraints in a certain sense.

Table 4 Average timing results, three-bar truss

Solution	One level	Two level
Number of passes	6	5
CPU time per pass, s		
Analysis	0.7	0.7
Optimization	0.7	1.7
Total	1.4	2.4
Error,[a] %	1.7	0.8

[a]The references for the errors are given in Table 2.

Table 5 Average timing results, 10 bar truss

Solution	One level	Two level unaccelerated	Two level
Number of passes	14	12	12
CPU time per pass, s			
Analysis	7.5	7.6	7.6
Optimization	2.8	19.6	8.6
Total	10.3	27.2	16.2
Error,[a] %	2.9	0.3	0.6

[a]The references for the errors are given in Table 2.

Table 6 Average timing results, 52 bar truss

Solution	One level[a]	Two level
Number of passes	16	14
CPU time per pass, s		
Analysis	262	256
Optimization	122	131
Total	384	387
Error,[b] %	7.0	0.8

[a]Includes only run 52A results.
[b]The references for the errors are given in Table 2.

Tables 4–6 show some summary statistics that help compare the approaches and indicate how this comparison is affected by problem size. For each problem size and approach, the data are an average between all the available runs. The number of passes necessary seems to be about 15, regardless of the problem size or the algorithm used (beyond the rather small three-bar truss example). The computation times are for one pass and are broken down in the time devoted to analysis (exact constraint values and their derivatives) and to optimization (optimization per se and approximate, sensitivity, and optimum sensitivity analyses). For the three-bar truss, it is seen that, as expected, there is quite a bit more effort expended in the multilevel optimization than in the one-level one. However, as the problem size increases, this situation changes so that, for the 52 bar truss, both approaches use about as much computation time for the optimization part of the process. On the other hand, as the model size increases, analysis takes a progressively more important part of the computation time. Therefore, it appears that for large problems both approaches, as implemented here, are equivalent because they are dominated by analysis cost. This observation is probably valid for most structural optimization applications, as planar truss analysis is among the least expensive analyses available.

Table 5 shows two types of two-level runs. The unaccelerated runs do not make use of adaptive convergence criterion or subproblem disconnect. It appears that these two features alone reduce the cost of optimization by a factor of two.

Finally, the error entry in Tables 4–6 is obtained as the average relative errors observed by comparison with the references of Table 2. It shows that the multilevel solution is consistently more accurate than the one-level one. The same observation can be made from Figs. 2–4. Close analysis of these results indicates that the usable-feasible direction algorithm used to perform optimization in the one-level approach (and at the top level for the two-level approaches) has considerable difficulties handling the curvature of the convex approximations to the constraints. Actually, one-level run 52B never even converges. By decomposing the problem in smaller subproblems, it appears that better mathematical conditioning is obtained and that the two-level algorithm is not affected (or less affected) by this difficulty. A similar observation has been made before.[3] This problem would probably be alleviated if an algorithm specifically designed for this type of approximation were used, i.e., the dual algorithm of Ref. 7. Doing so would yield similar levels of accuracy for both one-level and multilevel schemes. However, this would not significantly affect the conclusions on optimization cost. Indeed, for reasonably sized problems, the cost is dominated by that of analysis, as indicated above.

Summary

This paper addresses the problem of the efficiency of multilevel optimization algorithms. As a large design problem is decomposed into smaller ones, which in turn are optimized separately, an interative procedure must be devised that accounts for the coupling between the subproblems. These repeated reanalyses and reoptimizations of the various subproblems drive up the cost of optimization. The proposed algorithm tends to alleviate that effect by resorting to approximations and subproblem disconnect.

One component of the cost of optimization is the cost of the analysis of the system that is required to obtain the values of the constraints and their derivatives. In this application, the initial design problem is replaced by a sequence of convex approximations that are the most conservative of the approximations linear in the variables or their reciprocal. The conservatism introduces the need for constraint relaxation and the approximations may be harder for gradient-based primal algorithms to handle, as our one-level results seem to indicate. On the other hand, the use of approximations takes the analysis of the system out of the iterations required to converge the multilevel procedure. Therefore, the cost of analysis is the same whether multilevel optimization is used or not.

The second component in the cost of the optimization is that of optimizing the explicit approximate subproblems. This cost is kept down here by using subproblem disconnect, a feature by which subproblems that have no violated, critical, or near-critical constraints are not reoptimized after the first cycle in one approximation. This feature, along with an adaptive convergence criterion, helped reduce the cost of optimization by a factor of two for the 10 bar truss example. As the size of the design problem increases (to approximately 100 design variables), the cost of optimization for the multilevel approach comes back in line with that of the one-level approach.

However, as the size of the problem increases, the total cost of optimization becomes dominated by that of analysis, so that one-level and multilevel implementations of the algorithm become equivalent.

For the planar truss design examples solved here, the multilevel approach has proved more robust and more accurate than the one-level one. This difference would probably be eliminated if an algorithm specifically designed for the convex approximations were used to solve the approximate subproblems. Some experimentation has been necessary in order to select the appropriate set of algorithm-related parameters.

It must be noted that this implementation did not take advantage of the benefits of analysis decomposition or distributed processing. Should this be done, multilevel optimization would prove more economical than one-level optimization.

Appendix A: Detailed Problem Formulation

This appendix contains the expressions used for the various constraints in the problem. The problem global constraints involve stresses and displacements of the finite-element model. If σ_{ij} is the stress in member i under load case j and $\sigma_{\max}$ the maximum allowable stress, then a stress constraint reads

$$|\sigma_{ij}|/\sigma_{\max} - 1 \le 0 \tag{A1}$$

With u_{kj} as the displacement in degree-of-freedom k of the structure and under load case j as well as $u_{k\max}$, the maximum allowable displacement in degree-of-freedom k, a displacement constraint reads

$$|u_{kj}|/u_{k\max} - 1 \le 0 \tag{A2}$$

The local level constraints are on buckling, crippling, maximum thickness-to-diameter ratio, and upper and lower bounds for the thicknesses. Assuming that member i is designed by subproblem s, then the constraint guarding member i against buckling under load case j is

$$-8L_i^2\sigma_{ij}/\pi^2 E_i d_s^2 - 1 \le 0 \tag{A3}$$

where L_i is the length of member i, E_i the Young's modulus for the material used in member i, and d_s the diameter for subproblem s. The crippling constraints for member i under load case j is

$$-\pi d_s^2 \sigma_{ij}/K_i E_i A_s - 1 \le 0 \tag{A4}$$

where K_i is the crippling constant for element i and A_s the area for subproblem s. The geometrical constraint defining a maximum value D_s for the thickness-to-diameter ratio is

$$A_s/\pi D_s d_s^2 - 1 \le 0 \tag{A5}$$

With $t_{\min}$ and $t_{\max}$ the minimum and maximum thicknesses for the design, respectively, the constraints enforcing these limits read

$$1 - A_s/\pi d_s t_{\min} \le 0 \tag{A6}$$

$$A_s/\pi d\hat{s} t_{\max} - 1 \le 0 \tag{A7}$$

Appendix B: Kreisselmeier-Steinhauser Function

A few points are made here about the Kreisselmeier-Steinhauser function.[9] Given m constraints that are twice-differentiable functions of a n-dimensional vector of variable X:

$$g_l(X) \le 0, \quad l = 1,m \tag{B1}$$

In what follows, the dependency on X will be implied for the sake of making the notation more concise. A cumulative measure of the satisfaction or the violation of these constraints is given by

$$C=\frac{1}{\rho}\,\ell n\left[\sum_{l=1}^{m}\exp(\rho g_l)\right] \tag{B2}$$

where ρ is a user-selected positive number. If $g_{max}=\max(g_1,g_2,\ldots,g_m)$, then

$$C=g_{max}+\frac{1}{\rho}\,\ell n\left\{\sum_{l=1}^{m}\exp[\rho(g_l-g_{max})]\right\} \tag{B3}$$

From Eq. (B3),

$$g_{max}\leq C\leq g_{max}+(1/\rho)\ell n(m) \tag{B4}$$

From Eq. (B4), it is clear that the Kreisselmeier-Steinhauser function is a conservative envelope function following the maximum constraint. The larger the value of ρ, the closer C follows g_{max}.

The gradient of the cumulative constraint is easily obtained from that of the participating constraints. If g^x is the column vector whose rth entry is dg/dX_r,

$$C^x=\sum_{l=1}^{m}\exp[\rho(g_l-C)]g_l^x \tag{B5}$$

Equation (B5) shows that, for large values of ρ, C^x is dominated by g^x_{max}. This can be a problem for gradient-based optimization procedures as g^x_{max} is a discontinuous function.

The choice of ρ must be a compromise between the desire to closely follow g_{max} on the one hand and the concern for discontinuous gradients on the other hand. It has proved satisfactory to choose ρ as follows. If the optimization program constraint tolerance is ϵ (that is, if $|g|\leq\epsilon$ indicates that constraint g is active), then choose ρ so that

$$\rho=\ell n(m)/\epsilon \tag{B6}$$

With such a choice, if one of the cumulative constraints becomes active $[C(X)=0]$ during optimization, the largest of the participating constraints will be within ϵ of being active itself.

The matrix of second derivatives of $C(X)$ is derived from Eq. (B5). If g^{xx} is the square matrix whose r,t entry is $d^2g/dX_r dX_t$,

$$C^{xx}=\sum_{l=1}^{m}\exp[\rho(g_l-C)](\rho g_l^x g_l^{xT}+g_l^{xx})-\rho C^x C^{xT} \tag{B7}$$

After using Eq. (B4) as well as some algebraic manipulations, Eq. (B7) may be rewritten:

$$C^{xx}=\sum_{l=1}^{m}\exp[\rho(g_l-C)]g_l^{xx}+\sum_{l=1}^{m}\sum_{n>l}^{m}\exp[\rho(g_l-C)]$$

$$\times\exp[\rho(g_n-C)](g_l^x-g_n^x)(g_l^x-g_n^x)^T \tag{B8}$$

If the g_l constraints are convex functions, the g_l^{xx} matrices are positive semidefinite. Further, $(g_l^x-g_n^x)(g_l^x-g_n^x)^T$ is positive semidefinite as matrix product of a vector onto itself. Finally, the coefficients of all the matrices in Eq. (B8) are strictly positive. Therefore, C^{xx} is positive semidefinite as the sum of positive semidefinite matrices and C is convex as long as the g_l are convex themselves.

Appendix C: Constraint Relaxation

Given a general nonlinear programming problem:
Find X so that $F(X)$ is minimum, and

$$g_l(X)\leq 0,\quad l=1,m$$

$$X^l\leq X\leq X^u \tag{C1}$$

It may be necessary to solve this problem by beginning at an infeasible starting point using a nonlinear programming algorithm not specifically designed to handle strongly violated constraints. In such a case, the following transformed problem may be solved for the solution of Eq. (C1), provided that it has a solution:

Find X, ω so that $F(X)+K\omega$ is minimum and

$$g_l(X)+1-\omega\leq 0,\quad l=1,m$$

$$X_s^l\leq X_s\leq X_s^u$$

$$1\leq\omega \tag{C2}$$

If X^o is a starting design that is infeasible for Eq. (C1), an initial ω^o can be found such that X^o is feasible for Eq. (C2),

$$\omega^o=1+\max_l[g_l(X^o)] \tag{C3}$$

Minimization of the new objective function will tend to drive the *relaxation variable* ω toward its lower bound. If ω reaches its lower bound, Eq. (C2) becomes identical to Eq. (C1), except for a constant offset of the objective function. The user-defined parameter K may be chosen so that a predetermined ratio exists at the initial design between the true objective function $F(X^o)$ and the penalty from $K\omega^o$. Should the solution of Eq. (C2) still be infeasible for Eq. (C1), it may be necessary to increase that ratio.

In general, there is no guarantee that Eq. (C1) has a solution. If Eq. (C1) does not, a final value of ω that is greater than one will result, regardless of the value of K. Solving Eq. (C2) is a rational approach to finding a design that, in a certain sense, "minimizes" the violation of the constraints of Eq. (C1).

Acknowledgment

The support provided by NASA Research Grant NAG-1-145 and Contract NAS1-18106 is gratefully acknowledged. This work was performed while the first author was Research Associate at Virginia Polytechnic Institute and State University in Blacksburg.

References

[1]Sobieszczanski-Sobieski, J., "A Linear Decomposition Method for Large Optimization Problems—Blueprint for Development," NASA TM-83248, Feb. 1982.

[2]Sobieszczanski-Sobieski, J., James, B., and Dovi, A., "Structural Optimization by Multilevel Decomposition," *AIAA Journal*, Vol. 23, Nov. 1985, pp. 1775–1782.

[3]Sobieszczanski-Sobieski, J., James, B., and Riley, M., "Structural Optimization by Generalized, Multilevel Decomposition," *AIAA Journal*, Vol. 25, Jan. 1987, pp. 139–145.

[4]Barthelemy, J.-F. M., "Development of a Multilevel Optimization Approach to the Design of Modern Engineering Systems," NASA CR-172184, Aug. 1983.

[5]Haftka, R. T., "An Improved Computational Approach for Multilevel Optimum Design," *Journal of Structural Mechanics*, Vol. 12, No. 2, 1984, pp. 245–261.

[6]Starnes, J. H. Jr. and Haftka, R. T., "Preliminary Design of Composite Wings for Buckling, Strength, and Displacement Constraints," *Journal of Aircraft*, Vol. 16, Aug. 1979, pp. 564–570.

[7]Fleury, C. and Braibant, V., "Structural Optimization, A New Dual Method Using Mixed Variables," Aerospace Laboratory, Univ. of Liege, Belgium, Rept. SA-115, March 1984.

[8]Schmit, L. A. Jr. and Farshi, B., "Some Approximation Concepts for Structural Synthesis," *AIAA Journal*, Vol. 12, May 1974, pp. 692–699.

[9]Kreisselmeier, G. and Steinhauser, R., "Systematic Control Design by Optimizing a Vector Performance Index," *Proceedings of IFAC Symposium on Computer Aided Design of Control Systems*, Zurich, 1979, pp. 113–117.

[10]Sobieszczanski-Sobieski, J., Barthelemy, J.-F. M., and Riley, K. M., "Sensitivity of Optimum Solutions to Problem Parameters," *AIAA Journal*, Vol. 20, Sept. 1982, pp. 1291–1299.

[11]Barthelemy, J.-F. M. and Sobieszczanski-Sobieski, J., "Optimum Sensitivity Derivatives of Objective Functions in Nonlinear Programming," *AIAA Journal*, Vol. 21, June 1983, pp. 913–915.

[12]Vanderplaats, G. N., "ADS—A FORTRAN Program for Automated Design Synthesis—Version 1.10," NASA CR-177985, Sept. 1985.

[13]"IMSL Users Manual," ed. 9.2, IMSL, Houston, TX, Nov. 1984.