Development and Application of the Collaborative Optimization Architecture in a Multidisciplinary Design Environment

R. D. Braun* I. M. Kroo†

August 14, 1995

Abstract

Collaborative optimization is a design architecture applicable in any multidisciplinary analysis environment but specifically intended for large-scale distributed analysis applications. In this approach, a complex problem is hierarchically decomposed along disciplinary boundaries into a number of subproblems which are brought into multidisciplinary agreement by a system-level coordination process. When applied to problems in a multidisciplinary design environment, this scheme has several advantages over traditional solution strategies. These advantageous features include reducing the amount of information transferred between disciplines, the removal of large iteration-loops, allowing the use of different subspace optimizers among the various analysis groups, an analysis framework which is easily parallelized and can operate on heterogenous equipment, and a structural framework that is well-suited for conventional disciplinary organizations. In this article, the collaborative architecture is developed and its mathematical foundation is presented. An example application is also presented which highlights the potential of this method for use in large-scale design applications.

1 Motivation and General Description

Numerous design problems exist in which the product is so complex that a coupled analysis driven by a single optimizer is not practical. This situation is often the result of the organizational philosophy found in many large, design groups, where specialists are typically separated by discipline and multidisciplinary interaction is difficult. For example, most large aerospace companies include aerodynamics, structures, and dynamics and control divisions. Typically, when beginning a multidisciplinary project, such as design of a new airplane, a project-leader must decompose the original problem and distribute the relevant parts among the existing organizational groups. Armed

*Aerospace Engineer, NASA Langley Research Center, Hampton VA
†Associate Professor, Stanford University, Stanford CA

only with the current set of multidisciplinary design optimization (MDO) tools, a large-scale design problem may be cumbersome to manage, time-ineffective, and even non-convergent. One problem which often occurs in this situation is disciplinary sequencing, where one analysis group must wait for data computed by another group. In other cases, where an integrated analysis approach is possible, significant time must usually be invested *a priori* to integrate the analysis codes required for a solution. Even in an integrated approach, time degradation as a result of disciplinary sequencing may occur [7]. As a result, the solution of large, coupled, multidisciplinary problems remains a challenging task in need of more flexible solution methodologies.

Collaborative optimization is a design architecture applicable in any multidisciplinary analysis environment but specifically intended for large-scale distributed analysis applications. In this approach, a complex problem is hierarchically decomposed along disciplinary boundaries into a number of subproblems. With the use of local subspace optimizers, each discipline is given control over its own set of local design variables and is charged with satisfying its own disciplinary constraints. The goal of each local optimizer is to agree with the other groups on values of the multidisciplinary variables, while a system-level optimizer provides coordination and minimizes the overall objective. This design strategy is analagous to that found in most design teams where a team leader (system-level optimizer) is responsible for minimizing the overall objective while guiding a set of disciplinary experts (subspace optimizers) into agreement. As a result, this method should be well-suited for use in conventional design organizations. Although the disciplinary decomposition increases the total number of variables, for many problems the communication requirements and the size of the system-level optimization problem are reduced in comparison to non-hierarchically decomposed strategies [18]. These reductions are a direct result of the collaborative architecture's reliance on subspace optimizers to handle the disciplinary decisions.

The characteristics of this approach to optimal design are quite different from those of the standard optimization procedure in use today; several of these distinctions include: (1) a natural fit to the current disciplinary expertise structure found in most design organizations, (2) no analysis integration requirements, (3) the potential selection of each subspace optimizer to best fit the given disciplinary model (large/small, sparse/full, constrained/unconstrained), (4) an analysis framework which is easily parallelized and can be operated on heterogenous equipment, and (5) a synchronization of the design process. (For instance, the structures group does not have to wait a week for the aerodynamics group to provide a starting solution. Instead, each discipline may begin work on their local design simultaneously and convergence to an overall solution is achieved through collaboration.)

This article focuses on development of the collaborative architecture. The niche that this architecture fills in the spectrum of MDO methods is discussed and the mathematical formulation is presented. A highly constrained sample application is also presented which demonstrates the solution characteristics of this architecture.

Figure 1: Standard Optimization Approach

Consider the approach to optimization illustrated in Fig. 1, where we are concerned with a problem in n variables and m constraints. Here

$$x \in \Re^n$$

$$J : \Re^n \rightarrow \Re$$

$$c : \Re^n \rightarrow \Re^m, m \leq n.$$

Within the present article, this formulation will be referred to as the standard optimization approach. At present, this formulation has the broadest use in engineering design. For a multidisciplinary system, use of this approach requires an integrated set of analysis models such that for a given set of design variables (x), the analysis returns the values of each constraint (c), and the objective function (J). Hence, in this formulation, the role of each discipline is limited to one of function evaluation only (i.e., the analysis-block has no explicit decision-making power).

Since we are concerned with decomposition of the optimization problem in an "analysis-convenient" manner, we must first examine the analysis block of Fig. 1 in more detail. Note that instead of the single analysis-block shown in Fig. 1, the multidisciplinary analysis actually looks more like that shown in Fig. 2, where

$$\begin{array}{ll} x_1 \in \Re^{n_1} & c_1 : \Re^{n_1} \rightarrow \Re^{m_1} \\ x_2 \in \Re^{n_2} & c_2 : \Re^{n_2} \rightarrow \Re^{m_2} \\ \vdots & \vdots \\ x_N \in \Re^{n_N} & c_N : \Re^{n_N} \rightarrow \Re^{m_N} \end{array}$$

$$\sum_{i=1}^{N} n_i \geq n \qquad \sum_{i=1}^{N} m_i = m$$

Here, the individual analysis-blocks may be thought of as multiple subroutines in a single program or multiple programs in a single analysis. According to the requirements of the individual analysis-blocks, the original design variable vector x is

Figure 3: Simultaneous Analysis and Design [1] or All-At-Once [9] Optimization Formulation.

partitioned into N subsets x_i which, in general, are not mutually disjoint. The original set of constraints is also partitioned into N pieces, c_i, where the c_i vectors are mutually disjoint.

From Fig. 2, the linking among the analysis blocks is evident, appearing in the form of intermediate or coupling variables, y_{ij}. In general, there may be any number of these coupling variables. Note that in this article, y_{ij} refers to a variable computed in analysis block j, and required as input to analysis block i.

As shown in Fig. 2, in the standard optimization approach, the original constraint vector is reassembled and along with the objective function (which is assumed to be computed in analysis block N) is returned to the optimization block. Hence, in this approach, this design variable and constraint decomposition occurs implicitly, unbeknownst to the optimization package.

Research in decomposition analysis [14, 18, 20] has led to a class of alternative formulations, known as either "simultaneous analysis and design" or "all-at-once"

Figure 4: Collaborative Optimization Architecture

approaches [1, 9]. As sketched in Fig. 3, in this type of formulation, the N analysis-blocks are executed in parallel. Each analysis-block is responsible for computing its own set of the originally partitioned constraints, c_i. Furthermore, for each coupling variable, y_{ij} shown in Fig. 2, a design variable, y_{ij}, and an equality constraint, d_{ij}, are added to the optimization problem set. These auxillary constraints may be referred to as compatibility constraints since their purpose is to ensure that multidisciplinary feasibility is achieved by the parallel analyses at the problem solution. When satisfied, these equality constraints, d_{ij}, require that the value of a variable computed in analysis block j match the value of the equivalent variable input to analysis block i.

In comparison to the standard formulation, the solution strategy depicted in Fig. 3 avoids disciplinary sequencing through the use of a parallel analysis strategy. Furthermore, the requirement of producing a compatible multidisciplinary model (often termed multidisciplinary feasibility) is removed from the analysis block. Instead, this feasibility requirement is an added responsibility of the optimizer. In this manner, consistency across the disciplinary models (the parallel analysis-blocks) is only required at the solution, where $d_{ij} = 0$. In some cases, these formulative changes have been shown to produce computational savings by removing implicit iteration loops from the original analysis-block [7, 15].

Although the approach of Fig. 3 may be computationally faster than the use of the standard optimization approach, the analysis groups are still removed from the design decision-process, acting as mere function evaluators. Furthermore, in large-scale applications, with thousands of design variables and constraints, the use of a single optimizer may lead to two problems: (1) an inefficient solution strategy where too much time is spent on communications, and (2) the posing of a larger optimization problem than necessary since all design decisions (no matter how small) are made by the optimization routine.

In the collaborative architecture, each of these issues is mitigated through a reliance on subspace optimizers to handle the local (single analysis-block) decisions. As sketched in Fig. 4, the problem is hierarchically decomposed along analysis-block boundaries into N subproblems. The original problem constraints and design variables are partitioned among these subproblems as presented in Fig. 2. Because the design variable partitioning is not mutually disjoint, this approach to decomposition permits the construction of subspace optimization problems in which a feasible point is guaranteed to exist (assuming a feasible point existed in the original problem statement).

Numerous multilevel optimization approaches have been documented in the literature [2, 3, 4, 16, 17, 19, 21]. The most significant distinctions among these multilevel methods include (1) the way in which the disciplinary constraints are handled and (2) the manner and extent of coordination among the different levels. In the collaborative approach the disciplinary constraints are treated at the subspace level. Additionally, the subspace problems are not burdened by the requirement of having to help satisfy the constraint set of the other analysis-blocks. As a result, in the collaborative approach, (1) the workload and communication requirements of the system-level coordination process are significantly reduced and (2) the computational burden and the local design responsibility remains in the subspaces (where the disciplinary analyses are located). This is in contrast to other multilevel methods which maintain a majority of the design decision responsibility at the system-level. The price for providing the subspace optimizers with the freedom to make local design decisions in the collaborative architecture is an increase in the overall problem size through the addition of an extra set of auxillary variables (referred to as system-level targets, z, or subspace parameters, q).

In contrast to other multilevel methods [3, 17, 21], the collaborative architecture allows the subspaces to disagree on the appropriate values for these multidisciplinary variables (z). This results in a less-restricted sub-level design space and eliminates the need for multiple equality constraints, often found in multilevel methods. At the solution, these multidisciplinary variables are used to enforce compatibility among the analysis blocks. An element of the system-level target vector, z_{ij}, is created for (1) each variable which is input to one analysis-block and computed in another and (2) each variable which requires input to multiple analysis-blocks.

Of the multilevel optimization methods cited previously, the collaborative architecture is most closely related to the strategies presented in Refs. [2, 19]. In each approach, the subspace optimization problems are formulated to guarantee the existence of a feasible subspace point. In the collaborative architecture this is accomplished by providing design freedom within the subspace optimization processes and enforcing multidisciplinary compatibility at the system-level. In Refs. [2, 19], the subspace constraints are augmented into the subspace objective function and dealt with cumulatively at the system-level. This results in smaller subspace optimization problems, since there is no need for local variables to augment each system-level target, but a

more complex set of system-level constraints.

Another distinction between the collaborative architecture and the multilevel methods cited previously is the manner in which the system-level gradient information is obtained. As outlined in Section 3.4, as a result of structure of the collaborative formulation, estimation of the system-level gradients is analytic and does not even require accurate subproblem Lagrange multiplier estimates. This is in contrast to the approaches discussed in Refs. [2, 4, 16, 19].

In the terminology of Ref. [9], the collaborative architecture is an individual discipline-feasible approach. Multidisciplinary feasibility is handled by the system-level optimizer much like the approach of Fig. 3. However, in contrast to this approach, insuring multidisciplinary feasibility at the solution is the only task of the collaborative architecture's system-level constraints, d_i. In this formulation, the system-level optimizer is not responsible for any analysis evaluation and exists solely to provide multidisciplinary coordination across the parallel analysis groups.

As outlined in Fig. 4, the objective of each subspace optimizer is to minimize the discrepancy between the subset of subspace variables which are multidisciplinary (both inputs, x and outputs, y) and the target values of these variables computed by the system-level optimizer, $q = z$ while satisfying the set of subspace constraints. Knowledge of a particular set of subspace constraints is not required outside the relevant subproblem. To enforce multidisciplinary feasibility at the solution, the subspace objectives, d_i, are treated as constraints by the system-level optimizer. Details of the mathematical formulation are provided in the following section.

3 Mathematical Formulation

3.1 System-Level Coordination Problem

The collaborative system-level coordination problem may be expressed as,

$$\min \; J_N(z) \tag{1}$$

$$s.t. \; d_i(z,p) \;=\; \sum_{j=1}^{h_i} (p_{ij} - z_{ij})^2, \; i = 1, N \tag{2}$$

where,

J_N : system-level objective function; computed in subspace N

z : system-level design variable vector of length k

p : system-level parameter vector of length l

d : system-level nonlinear constraint vector of length N

h_i : number of system-level design variables of significance within subspace i

Because each subproblem is only required to match a subset of the system-level targets, there are at least twice as many system-level parameters as system-level design variables. Furthermore, in the notation of eq. (2), the system-level design variable vector, z, is not partitioned into mutually disjoint sets. For example,

$$\begin{array}{ccc} z_1 = z_{11} & z_1 = z_{21} & \\ z_2 = z_{12} & \vdots & z_2 = z_{N1} \\ \vdots & z_3 = z_{22} & z_3 = z_{N2} \\ \vdots & \vdots & \vdots \\ z_k = z_{1h_1} & z_k = z_{2h_2} & z_k = z_{Nh_N} \end{array}$$

As a result,

$$\sum_{i=1}^{N} h_i = l \geq 2k$$

The system-level constraint vector, d, and parameter vector, p are obtained from the optimal solution of a set of N subproblems, each of the form described in the following section.

3.2 i^{th} Subspace Optimization Problem

Partitioning the h_i system-level targets, q, into the $h_i^{'}$ multidisciplinary inputs of analysis block i and $h_i^{''}$ multidisciplinary outputs computed in analysis block i, each subspace optimization problem may be expressed as,

$$\min\ d_i(x,q) = \sum_{j=1}^{h_i^{'}}(x_{ij}-q_{ij})^2 + \sum_{j=1}^{h_i^{''}}(y_{ij}-q_{ij})^2 \quad (3)$$

$$s.t.\ c_i(x) \geq 0 \quad (4)$$

where,

q : subproblem parameter vector of length h_i, equal to system-level targets, z
x : subproblem design variable vector of length n_i, where $n_i \geq h_i^{'}$
c : subproblem nonlinear constraint vector of length m_i
y : subproblem multidisciplinary output vector of length $h_i^{''}$

Note that only a subset of the subspace design variables are represented in the subspace objective function, eq. (3). In particular, for analyses with less multidisciplinary coupling, the difference between n_i and $h_i^{'}$ will be larger denoting the increased degrees-of-freedom for the analysis block. Note that the system-level targets appear as parameters in the subspace optimization problem. As a result, the analysis block constraints are explicitly dependent on the subspace design variables, x, only.

3.3 System-Level, Subspace Relationship

The system-level and subspace optimization processes are related as,

$$p_{ij} = {x_{ij}}^* \tag{5}$$

$$q_{ij} = z_{ij} \tag{6}$$

where, $x_{ij} \longrightarrow x_{ij}^*$ at the subspace solution.

Hence, the subspace parameters, q, are represented at the system-level as design variables, z, and the system-level parameter values, p, are determined by the subspace optimization processes.

3.4 System-Level Gradient Information, Formulation Refinements

Constraint and objective gradients may be obtained by finite-differencing the subproblems. However, this approach is computationally expensive and requires tight convergence of each subspace optimization [4]. Alternatively, an estimate of the system-level constraint Jacobian can be obtained through post-optimality analysis of each subproblem. In fact, the collaborative formulation has been specifically posed to make use of this information which is readily available at the solution of each subspace optimization problem [6].

$$\left.\frac{\partial d_i}{\partial z_j}\right|_{system} \equiv \left.\frac{d^* d_i}{dq_{ij}}\right|_{subspace} = -2.0({x_{ij}}^* - q_{ij}) \tag{7}$$

The equivalence statement made in eq. (7) is a result of the system-level targets being treated as parameters within each subspace optimization problem. The simple algebraic form of this equation results from (1) the subspace constraints being posed as explicit functions of the subspace design variables only, and (2) a subspace objective function which is simply the sum of squared discrepancy terms. With use of eq. (7), each time the system-level constraints are evaluated, the system-level Jacobian can also be provided.

With the addition of an extra system-level target, z_{k+1}, and augmentation of the system-level parameter vector with the original objective function, $p_{N(h_N+1)}$, the problem can be re-formulated (assuming the system-level objective is computed within the N^{th} subproblem) as,

System-level:

$$\min\ J(z) = z_{k+1} \tag{8}$$

$$s.t.\ d_i(z,p) = \sum_{j=1}^{h_i}(p_{ij} - z_{ij})^2 + \delta_{iN}(p_{N(h_N+1)} - z_{k+1})^2,\ i = 1, N \tag{9}$$

where,

$$p_{N(h_N+1)} = J_N \tag{10}$$

i^{th} **Subspace:**

$$\min\ d_i(x,q) = \sum_{j=1}^{h_i'}(x_{ij}-q_{ij})^2 + \sum_{j=1}^{h_i''}(y_{ij}-q_{ij})^2 + \delta_{iN}(J_N - q_{N(h_N+1)})^2 \tag{11}$$

$$s.t.\ c_i(x) \geq 0 \tag{12}$$

where,

$$\delta_{iN} = \begin{cases} 1 & \text{if } i = N \\ 0 & \text{if } i \neq N \end{cases} \tag{13}$$

With this refinement in the collaborative architecture, all of the required system-level derivative information is available at no additional computational cost over that required for constraint evaluation.

3.5 Formulation Refinement to Accommodate Multiple Feasible Regions in the Subspaces

In general, an optimization problem posed with multiple nonlinear constraints, may have multiple feasible regions; whereas, standard calculus-based optimization approaches only guarantee convergence to a local solution [11]. For the collaborative architecture, where the original-problem constraints are accommodated at the subspace level, multiple feasible regions can pose an additional convergence difficulty. In the collaborative approach, system-level variable perturbations translate directly into subspace parameter variations. As the system-level converges upon a solution, the subspace parameters may vary over a relatively large range. As a result, the subspace optimizer may discontinuously jump from one feasible solution region to another. This subspace solution inconsistency leads to non-smoothness in the subspace objective function which produces erroneous system-level constraint gradient estimates.

A solution to this difficulty is to limit the domain of each subspace optimization process to a single feasible region. This restriction which is analagous to the local convergence restriction of standard calculus-based optimizers may be enforced in several ways. One possibility is to add a small penalty-term to the subspace objective functions. Hence, eqs. (3) and (4) become,

$$\min\ d_i(x,q) = \sum_{j=1}^{h_i'}(x_{ij}-q_{ij})^2 + \sum_{j=1}^{h_i''}(y_{ij}-q_{ij})^2 + \epsilon\sum_{j=1}^{h_i'}(x_{ij}-X_{ij}^*)^2 \tag{14}$$

$$s.t.\ c_i(x) \geq 0 \tag{15}$$

where,

X^* : optimum subproblem design variable vector from previous subproblem call
ϵ : penalty magnitude (e.g., 10^{-6})

With this approach, an improper choice of ϵ will affect problem convergence. An alternate approach (which is applied in Section 4) is to replace each inequality constraint with a slack-variable and equality constraint. As,

$$c_i(x) \geq 0 \quad \longrightarrow \quad \begin{cases} c_i(x) - s_i & = 0 \\ s_i & \geq 0 \end{cases} \tag{16}$$

This approach does not eliminate the inequality; it is still present as a bound on the auxillary subspace design-variable, s_i. Furthermore, the subspace problem size is increased. However, because the inequality is present in the form of a simple bound, an optimization algorithm may not include its effect in determination of the search direction. As a result, use of the slack-variable formulation refinement restricts the subspace optimizer to a single feasible region (e.g., where the bound is always active). This algorithmic refinement yields a smooth variation in the subspace objective function with respect to parameter variations even in the presence of nonlinear constraints with multiple feasible regions.

4 Application To Aerospace Vehicle Design

4.1 Lunar Ascent Trajectory Optimization, 24 collocation segments

To demonstrate application of the collaborative architecture, a lunar ascent trajectory optimization problem was selected. The particular problem chosen has a well-known variational solution [8]. The objective is to find the optimal thrust-angle profile such that a minimum-time ascent flight path from the lunar surface to a prescribed orbit (120 nm, circular) is achieved. In this case, minimum-time is equivalent to minimum-fuel since a constant propellant-mass-flowrate is assumed.

The problem characteristics are sketched in Fig. 5. Note that the constraint set includes initial and terminal conditions as well as the equation-of-motion requirements. In the present study, this problem is modeled with the collocation technique in which polynomial expressions are used to approximate both the state (position and velocity) and control (thrust-angle) profiles. The equations of motion (F=ma) are then imposed at discrete points along the flight path such that the appropriate physics is satisfied [5, 10, 13]. This modeling technique was selected because the number of variables and constraints could be simply varied; thereby, demonstrating the collaborative architecture on problems of small, moderate, or large size.

Figure 6: Collaborative Optimization Decomposition Structure; 3 equal-time arcs

In the present investigation, optimization is performed with the sequential quadratic programming algorithm, NPSOL [12]. NPSOL uses a quasi-Newton method to approximate the Hessian of the Lagrangian. Gradients are obtained by forward finite-differencing. This type of algorithm is known to converge to local minimum for problems which are scaled properly and are twice-continuously differentiable.

Although this problem could be solved by standard optimization techniques using a single optimizer (i.e., Fig. 1), for demonstration purposes we choose to decompose the analyses into three equal-time, arcs ($N = 3$) as illustrated in Fig. 6. The modeling of these three arcs comprises the analysis portion of each of three subspaces optimization problems. Coupling requirements on the state and control profiles at the arc boundaries yield the system-level targets. In this case, we only require value-continuity across the arc boundaries for each state and control variable. Since we are concerned with a planar solution, this arc-boundary compatibility requirement yields 10 system-level targets (two-position, two-velocity, and one-control at each of the two

Figure 7: Illustration of The Collaborative Optimization Process

arc-boundaries). An 11^{th} system-level variable (arc time, Δt) is added such that the system-level objective gradient becomes analytic, as described in Section 3.4. Based on the need of each of the subspaces, these 11 variables are partitioned into vector sets such that, $h_i = \{6, 11, 6\}$. Finally, since there are three subspaces, there are also three system-level constraints.

Each subspace must satisfy its own set of equation-of-motion constraints. Satisfaction of the initial and terminal state constraints are added responsibilities of the first and third subspaces, respectively. Since the six system-level targets of significance to the first subspace represent target conditions for the endpoint of the analysis arc, $h_1^{'} = 0; h_1^{''} = 6$. Similarly, $h_2^{'} = 5; h_2^{''} = 6$, and $h_3^{'} = 5; h_3^{''} = 1$. Hence, in this example, analysis-block coupling exists on a subset of both input and output variables.

The actual number of subspace variables and constraints depends on how fine a discretization is performed. Assuming each subspace arc is decomposed into s segments over which linear state and control profiles are specified, the subspaces will be characterized by $5s + 2$, $5s + 6$, and $5s + 6$ design variables and $4s$, $4s$, and $4s + 3$, constraints, respectively. For a first example, $s = 8$ is chosen. This results in a total of 145 variables (including the 11 system-level targets) and 102 constraints (including three at the system-level).

The collaborative solution architecture begins with the system-level optimizer partitioning its initial guess of the system-level design variables and sending this information to the relevant subspaces. Within each subspace, these system-level design variables are treated as parameters (or targets). Each subspace optimization process produces a result which satisfies its respective set of constraints while trying to match the system-level targets as closely as possible. When discrepancies occur over the proper value of a multidisciplinary variable, the system-level optimizer co-

Figure 8: Collaborative Optimization Convergence History

ordinates with each subspace optimizer to resolve the difference. This coordination appears through selection of the system-level variables such that the system-level constraints, d_i, become satisfied. As an example of this collaborative process, consider the negotiation that must occur between the system-level optimizer, and the first two subspace optimizers over the appropriate value of the thrust-angle at the interface of these two arcs. This collaboration is highlighted in Fig. 7 which depicts the first three system-level iterations. As shown in this figure, neither subspace is able to match this system-level target during the first iteration. In fact, the first subspace optimization process terminates higher and the second subspace optimizer terminates lower than the target. Based on each subspace's ability to match the complete set of system-level targets (measured by the subspace objective functions), a new set of targets is produced. With these new targets in the Fig. 7 example, much better agreement is reached during the second system-level iteration. With a good match in this particular system-level target, a step is taken to reduce the system-level objective function (arc time, $z(11)$). This results in the bigger change in the system-level target between system-level iteration 2 and 3.

The convergence history of the collaborative solution to this problem with $s = 8$ is shown in Fig. 8. The sharp initial decrease in the system-level objective function results from the use of a linear system-level objective and is characteristic of solutions obtained with this approach [7]. As multidisciplinary feasibility is achieved (denoted by the value of Log $\|d_i\|$ in Fig. 8), the system-level objective returns towards the appropriate value. Upon convergence, the objective function and control profile were found to agree quite well with the published solution (to within 0.16% on the objective function). Even closer agreement could have been achieved through specification of a smaller system-level nonlinear feasibility tolerance.

To obtain the convergence history described in Fig. 8, the basic collaborative formulation (Section 3.1 and 3.2) was modified with the refinements presented in Sections

Figure 9: One-dimensional Slice of the Subspace 1 Design Space

3.4 and 3.5. As suggested by Fig. 9, without the Section 3.5 refinement, system-level convergence could not be achieved. For nonlinearly constrained subspace problems, this lack of convergence is a result of non-smoothness in the subspace objective function yielding bad system-level sensitivities. This degree of subspace non-smoothness had been observed in a previous problem in which this same type of convergence difficulty was noted [7]. As shown in Fig. 9, use of the slack-variable formulation refinement restricts the subspace optimizer to a single feasible region. This restriction yields a smooth variation in the subspace objective function with respect to parameter variations (i.e., a smooth system-level constraint gradient) enabling system-level convergence.

4.2 Problem Size and Analysis Coupling Effects on the Relative Performance of the Collaborative Architecture

By varying s, the problem size may be modified; however, the degree of coupling among the analysis groups remains the same. In this manner, the performance of the collaborative architecture on problems of different size with various degrees of analysis-coupling is assessed.

The lunar ascent problem was posed for $s = 2, 3, 8, 16, 32$, and 64. In each case, the problem was solved using the standard optimization approach (Fig. 1) and using the collaborative architecture (Fig. 4). As shown in Table 1, with use of the standard approach, the number of variables and constraints varied from 32 to 962 and from 27 to 771, respectively. For these same problems, the collaborative architecture required from 55 to 985 and 30 to 774 design variables and constraints, respectively. In each case, the collaborative posing of the problem requires 3 additional constraints (the $N = 3$ system-level constraints) and 23 additional design variables to accommodate analysis coupling. The suitability of the collaborative architecture for a given problem

Figure 10: Computational Performance of the Collaborative Architecture

depends on the relative proportion of multidisciplinary coupling, (which decreases as s increases). For small problems with a relatively large amount of coupling, use of the collaborative architecture is computationally inefficient as a result of the strategy's large amount of optimization overhead. However, as one moves from small problems with significant coupling to large, loosely-coupled problems, the decomposition characteristics of the collaborative architecture become computationally tractable. This trend is shown in Fig. 10.

Table 1 also shows that as a result of the convergence character of the collaborative solution, the final objective achieved is always observed to be below that obtained by the standard optimization approach (within the system-level nonlinear feasibility tolerance). This is a consequence of the architecture's individual discipline feasible approach where multidisciplinary feasibility is only achieved through system-level constraint satisfaction.

Figure 10 shows the number of analysis calls required by the collaborative architecture to reach the solution relative to the number of analysis calls required by a standard optimization approach. Comparison of the two curves shown in this figure illustrates the effect of warm-starting the subspace optimization problems on the computational solution requirements. Here, a warm-start refers to restarting each of the subspace optimizers from their prior solution with the prior solution estimates

of the active constraint set, Lagrange multipliers, and Hessian of the Lagrangian. Because the subspace optimization problems converge as the system-level solution is approached, warm-starting helps to significantly reduce the method's computational requirements.

As shown in Fig. 10, the collaborative architecture begins to become computationally competitive with the standard optimization approach as s increases. Note that as the overall problem size increases, the number of system-level iterations (Table 1, column 9) required by the collaborative architecture to reach the solution remains approximately constant. This is a result of the architecture's formulation in which the subspace analyses are required to work harder as the problem size is increased; while, the system-level is driven by the number of coupling variables. For the largest problem considered here ($s = 64$), the collaborative solution still requires more analysis calls than the standard optimization approach to reach the solution. However, in the present comparison, the subspace optimization problems were solved sequentially (neglecting the parallel processing capability of the collaborative scheme). Furthermore, if the analysis-integration and communication requirements (*a priori* setup time) were included, the collaborative architecture would prove to be more efficient than standard optimization approaches when applied to large, loosely-coupled multidisciplinary problems. For a multidisciplinary problems of this size (on the order of a thousand variables and constraints), these *a priori* requirements are often substantial.

5 Summary

Collaborative optimization is a design architecture applicable in any multidisciplinary analysis environment but specifically composed for large-scale, distributed analysis applications. In this approach, a complex problem is hierarchically decomposed along disciplinary boundaries into numerous subproblems which are guided towards multidisciplinary convergence by a system-level coordination process. When applied to problems in a multidisciplinary design environment, this scheme has several advantages over traditional solution strategies. These advantageous features include minimizing the amount of information transferred between disciplines, the removal of large iteration-loops between disciplines, allowing the use of different subspace optimizers among the various analysis groups, an analysis framework which is easily parallelized and operable on heterogenous equipment, and a structural framework that is well-suited for conventional disciplinary organizations.

In the present article, the collaborative architecture is developed and its mathematical foundation is presented. Refinements to the basic formulation are presented which demonstrate that the system-level gradient information may be obtained with no additional computational cost over that required for constraint evaluation and multiple subspace feasible regions may be accommodated. The niche which this architecture fills in the spectrum of MDO methods is also discussed.

A lunar ascent trajectory problem is presented which demonstrates the ability of this architecture to reach the appropriate, highly constrained solution. Size variations are used to infer the general characteristics of problems for which the collaborative architecture would be well-suited. In addition to becoming computationally competitive with standard optimization approaches as the problem size increases, the organizational advantages of the collaborative architecture make it well-suited for large, loosely-coupled, multidisciplinary design problems.

References

[1] Balling, R.J., and Sobieszczanski-Sobieski, J., *Optimization of Coupled Systems: A Critical Overview of Approaches,* AIAA Paper 94-4330, Sept. 1994.

[2] Balling, R.J., and Sobieszczanski-Sobieski, J., *An Algorithm for Solving the System-Level Problem in Multilevel Optimization,* AIAA Paper 94-4333, Sept. 1994.

[3] Barthelemy, J.M., and Riley, M.F., *Improved Multilevel Optimization Approach for the Design of Complex Engineering Systems,* AIAA Journal, Vol. 26, No. 3, pp. 353-360, March 1988.

[4] Beltracchi, T.J., *Decomposition Approach to Solving the All-Up Trajectory Optimization Problem,* Journal of Guidance, Control, and Dynamics, Vol. 15, No. 3, pp. 707-716, May-June 1992.

[5] Betts, J.T., and Cramer, E.J., *The Application of Direct Transcription to Commercial Aircraft Trajectory Optimization,* AIAA Paper 92-4528, May 1992.

[6] Braun, R.D., Gage, P.J., and Kroo, I.M., *Post-Optimality Analysis in Aerospace Vehicle Design,* AIAA Paper 93-3932, Aug. 1993.

[7] Braun, R.D., Powell, R.W., Lepsch, R.A., Stanley, D.O., and Kroo, I.M., *Multidisciplinary Optimization Strategies for Launch Vehicle Design,* AIAA Paper 94-4341, Sept. 1994.

[8] Bryson, A.E., and Ho, Y., *Applied Optimal Control,* Hemisphere Publishing Corp., 1975.

[9] Cramer, E.J., Dennis, J.E., Frank, P.D., Lewis, R.M., and Shubin, G.R., *Problem Formulation For Multidisciplinary Optimization,* SIAM Journal of Optimization, Vol. 4, No. 4, pp. 754-776, Nov. 1994.

[10] Enright, P.J., and Conway, B.A., *Optimal Finite-Thrust Spacecraft Trajectories Using Collocation and Nonlinear Programming,* Journal of Guidance, Control, and Dynamics, Vol. 14, No. 5, pp. 981-985, Sept-Oct 1991.

[11] Gill, P.E., Murray, W., and Wright, M.H., *Practical Optimization,* Academic Press, Inc., 1981.

[12] Gill, P.E., Murray, W., Saunders, M.A., and Wright, M.H., *User's Guide for NPSOL (Version 4.0): A Fortran Package for Nonlinear Programming,* Technical Report SOL 86-2, Dept. of Operations Research, Stanford University, January 1986.

[13] Hargraves, C.R., and Paris, S.W., *Direct Trajectory Optimization Using Nonlinear Programming and Collocation,* Journal of Guidance, Control, and Dynamics, Vol. 10, No. 4, pp. 338-342, July-Aug 1987.

[14] Himmelblau, D.M. ed., *Decomposition of Large-Scale Problems,* North-Holland Publishing Co., 1973.

[15] Kroo, I., Altus, S., Braun, R., Gage, P., and Sobieski, I., *Multidisciplinary Optimization Methods for Aircraft Preliminary Design,* AIAA Paper 94-4325, Sept. 1994.

[16] Peterson, F.M., Cornick, D.E., Brauer, G.L., and Rehder, J.J., *A Two-Level Trajectory Decomposition Algorithm Featuring Optimal Intermediate Target Selection,* Journal of Spacecraft & Rockets, Vol. 14, No. 11, pp. 676-682, Nov. 1977.

[17] Sobieszczanski-Sobieski, J., James, B., and Riley, M., *Structural Optimization by Generalized, Multilevel Decomposition,* AIAA Journal, Vol. 25, No. 1, pp. 139-145, Jan. 1987.

[18] Sobieszczanski-Sobieski, J., *Optimization by Decomposition: A Step from Hierarchic to Non-Hierarchic Systems,* NASA CP-3031, Sept. 1989.

[19] Sobieszczanski-Sobieski, J., *Two Alternative Ways for Solving the Coordination Problem in Multilevel Optimization,* Structural Optimization, Vol. 6, pp. 205-215, Dec. 1993.

[20] Wagner, T.C., *A General Decomposition Methodology for Optimal System Design,* Ph.D. Dissertation, University of Michigan, 9319649, 1993.

[21] Walsh, J.L., Young, K.C., Pritchard, J.I., Adelman, H.M., and Mantay, W.R., *Integrated Aerodynamic/Dynamic/Structural Optimization of Helicopter Rotor Blades Using Multilevel Decomposition,* NASA TP-3465, Jan. 1995.