

DETC2009-87517

**ARE WE THERE YET?
INVESTIGATING THE ROLE OF DESIGN PROCESS
ARCHITECTURE IN CONVERGENCE TIME**

Erich Devendorf

Graduate Research Assistant
Department of Mechanical and Aerospace Engineering
University at Buffalo – SUNY
Buffalo, NY 14260
edd4@buffalo.edu

Kemper Lewis

Professor
Department of Mechanical and Aerospace Engineering
University at Buffalo – SUNY
Buffalo, NY 14260
Corresponding Author: kelewis@buffalo.edu

ABSTRACT

In distributed design individual designers have local control over design variables and seek to minimize their own individual objectives. The amount of time required to reach equilibrium solutions in decentralized design can vary based on the design process architecture chosen. There are two primary design process architectures, sequential and parallel, and a number of possible combinations of these architectures. In this paper a game theoretic approach is developed to determine the time required for a parallel and sequential architecture to converge to a solution for a two designer case. The equations derived solve for the time required to converge to a solution in closed form without any objective function evaluations. This result is validated by analyzing a distributed design case study. In this study the equations accurately predict the convergence time for a sequential and parallel architecture. A second validation is performed by analyzing a large number of randomly generated two designer systems. The approach in this case successfully predicts convergence within 3 iterations for nearly 98% of the systems analyzed. The remaining 2% highlight one of the approach's weaknesses; it is susceptible to numerically ill conditioned problems. Understanding the rate at which distributed design problems converge is of key importance when determining design architectures. This work begins the investigation with a two designer case and lays the groundwork to expand to larger design systems with multiple design variables.

1 INTRODUCTION

The design of complex engineering systems often requires the involvement of a variety of experts. These experts may originate from different disciplines or even the same discipline.

As experts they typically fully understand only a very specific part of the aggregate design problem. In order to facilitate communication between these experts a variety of approaches have been created that broadly fall under the field of multidisciplinary design optimization (MDO).

Regardless of the approach chosen, the amount of time spent in the design process is of critical importance. It is very often the case that a design process does not end when the optimal solution is found, but when time runs out. The importance of managing time in a design process has been recently underscored by the struggles of Boeing during the design of the 787 Dreamliner. Its rival, Airbus, faced similar struggles in the design of the A-380 [1, 2] and it is estimated that delays cost Boeing at least 100 orders. Some of that business was lost to their chief competitor Airbus. The only mitigating factor for Boeing was Airbus faced the exact same problem as they did. Had Boeing been poised to seize the initiative it could have resulted in significant profits and gains in market share [3].

Even in less extreme examples time remains a critical resource in the design process, whether it be speeding time to market or meeting an important deadline [4]. Understanding the role of time to convergence in MDO is of key importance for the future success of industry. In the same way time studies are important in manufacturing a product, it is important to understand the amount of time required to design a complex system. In this paper the controlling mechanics governing the convergence rate in a non-cooperative distributed design framework are investigated. The results are applied to a distributed design case study to demonstrate their effectiveness as a tool for intelligently selecting design system architecture.

The benefits of this work include providing designers with the tools to increase design process control, to reach iterative solutions quicker and to set realistic design deadlines and time tables for product development [5]. Furthermore, this work provides direct insight into the mechanics governing distributed design. The next section explores the motivation for applying MDO frameworks to complex system design and elaborates on the reasons for investigating convergence within a non-cooperative distributed design framework.

2 MDO FRAMEWORKS

MDO problems have two primary classifications based on the process applied to complete the design [6]. From a structural perspective the simplest method to solve an MDO problem is to apply an all at once approach. In an all at once approach designers from different disciplines work as system analyzers to determine objective function and constraint values for a single optimization problem [7, 8]. There are significant advantages to organizing an MDO system to solve a single centralized optimization problem. In a centralized problem all designers are working towards the exact same objective, information about the entire system is available to all designers and any optimal solution found is optimal to the global system.

Although these advantages make centralization attractive, it is almost impossible to centralize the design of complex systems. Three major design approaches, Systematic Design, Total Design, Axiomatic Design, are all structured with the understanding that some level of system decomposition is often desirable and sometimes necessary [9-11]. In the future it may be possible to centralize a greater number of design activities, but history has demonstrated that as engineering capability increases so does the complexity of design problems. Recognizing this, it is likely decomposition will remain an important and necessary aspect of product design processes in the near future.

Fortunately for designers a wide range of approaches have been developed to aid in system decomposition. The system decomposition process can be broken into two fundamental steps:

1. Identify the necessary subsystems,
2. Create a framework for communication between subsystems.

The first step in this process is not the topic of this paper, but it is by no means a trivial task. Subsystems can be created based on object decomposition, aspect decomposition, sequential decomposition and model based decomposition [12]. A survey of the relative merits of these subsystem creation approaches was performed by Sobieski [6].

The second step is the creation of the design framework. In this paper “framework” is always be used to denote a specific MDO approach used to solve a design problem. MDO design frameworks specify the mechanics of how the design problem will be solved. These mechanics include determining the subsystem objective functions, establishing communication

protocol, assigning control of design variables and addressing the other assorted protocols required for the decomposed subsystem to effectively iterate to a solution. There are a wide range of MDO frameworks which provide for these requirements while making guarantees about system convergence and the optimality of the final converged solution. These approaches include Analytic Target Cascading [13], Concurrent Sub Space Optimization [14, 15], Bilevel Integrated System Synthesis [16] and Collaborative Optimization [17].

Using one of these frameworks has several advantages which depend on the framework chosen. For example, Analytic Target Cascading has been proven to guarantee the distributed system converges and that the converged value is a globally optimal solution [13]. Additionally, its hierarchy allows for traceability of the design process and provides for integration of marketing and business systems while establishing clear relationships between design subsystems [18]. In spite of the advantages offered by existing MDO frameworks, there are many cases when a formal framework is not used.

There are several reasons why a formal MDO framework may not be applied to a design problem. Applying a framework to a decomposed problem requires a significant level of coordination between subsystems and a high level of management expertise. Further, the designers must all “buy in” to the proposed decomposition and framework. There are also some cases that do not naturally lend themselves to formal decomposition or where the parties involved cannot agree on an appropriate framework. When no formal framework is chosen or when the framework does not specifically proscribe subsystem interactions, the design problem framework may simply become a distributed design problem. The assumptions and mechanics governing distributed design problems are discussed in Section 3.

3 DISTRIBUTED DESIGN

3.1 Problem Structure

Distributed design problems are a specific type of MDO problem, the mechanics of which are an ongoing area of research. Distributed design problems are non-hierarchical with a set of different designers, or subsystems, each seeking to optimize their own individual objectives. When distributed design problems are cooperative they are similar to the all in one approach to solving MDO problems. However, when the problems are non-cooperative they have more in common with decomposition approaches to MDO. It is important to note that the frameworks outlined for MDO decomposition approaches are not necessarily non-cooperative. For example, Concurrent Sub Space Optimization and Collaborative Optimization more closely resemble cooperative processes [19].

Non-cooperative distributed design problems are common in design, and have been widely studied. The requirements for non-cooperative behavior in an engineering design problem are [20]:

1. Designers have knowledge of only their own local objectives,
2. Designers act unilaterally to minimize their objective function,
3. Designers have complete control over specific local design variables,
4. Designers communicate by sharing the current value of their local design variables.

Distributed design problems are often sub problems that result from applying one of the design frameworks discussed in Section 2. Within a formal framework there are typically iterative steps that fulfill the assumptions of non-cooperation and behave similar to distributed design problems. For example, one approach to system decomposition is the Dedesign Managers Aid for Intelligent Decomposition (DeMAID) [21]. In DeMAID the overall system is shown using a Design Structure Matrix with the goal of eliminating iterative loops to reduce redesign [22].

There are times, however, when all iterative loops between subsystems cannot be eliminated. These iterative loops maintain the assumptions of non-cooperation for a distributed problem because the system level optimizer is responsible for setting up the design framework but not supervising its execution. Convergence equilibrium and convergence rate are the two fundamental concerns when designers iterate under non-cooperative assumptions. These concerns are discussed in the following two sections, 3.2 and 3.3.

3.2 Convergence Equilibrium

Determining if a design system converges to a solution is of critical importance to understanding the system. Convergence equilibrium in distributed problems has been a topic of research for some time with the first work being performed by Vincent [23] for a simple two designer, two design variable problems. Vincent's primary contribution was the introduction of Game Theory to model designer behavior, which was investigated further by Lewis [24]. In a game theoretic approach to design, the designers are modeled as players in an iterative game. In Vincent's work each player alternates minimizing their local objective function value and communicates the associated design variables to the other player. Each step in this alternating process is a play in a sequential game. After repeated playing of the sequential game the players either converge to a solution or diverge and continue playing indefinitely. When the players converge, they converge to a specific point called the Nash, or non-cooperative equilibrium [25]. Key to the understanding of the Nash Equilibrium is the concept of a player's rational reaction set (RRS). The rational reaction set is the partial derivative of the designer's objective function with respect to their local design variables. While determining designer RRS's is not a trivial task, work has been done developing methods to approximate them for large systems [26]. The intersection of the RRS for both players in a two person game is by definition the Nash Equilibrium. For the simple case in Eqns. 1 two designer

RRS's taken from Vincent [20] are plotted with respect to the design variables x and y in Figure 1.

$$\begin{aligned}
 F_1 &= x^2 + xy - 3x \\
 \frac{\partial F_1}{\partial x} &= 2x + y - 3 = 0 \\
 F_2 &= 0.5y^2 - xy \\
 \frac{\partial F_2}{\partial y} &= y - x = 0
 \end{aligned}
 \tag{1}$$

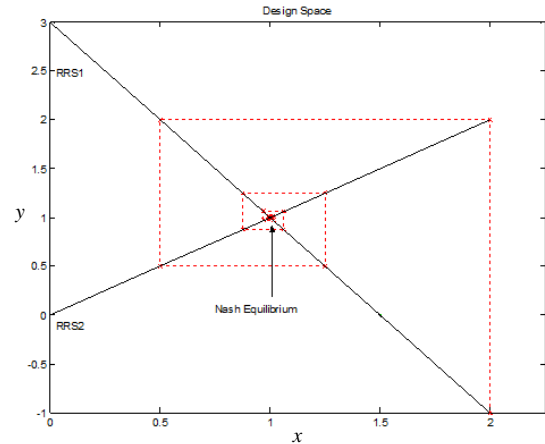


Figure 1: Nash Equilibrium

As seen in Figure 1 the repeat plays of the sequential game converges to the Nash Equilibrium at $(x,y) = (1,1)$ defined by the intersection of the two players' rational reaction sets. This work was extended by Chanron for sequential games to examine convergence more generally when there are more than two players controlling multiple design variables [27]. In these engineering examples it was shown again that convergence is a function of the relative slope of the designer's rational reaction sets and control theory was applied for large scale problem [27, 28]. A case for nonlinear rational reaction sets was also investigated [29]. While Chanron's work dealt with sequential games, work by Smith and Eppinger demonstrated a similar principle for games with simultaneous play [30].

A recent extension of this convergence work was performed by Gurnani who demonstrated that the introduction of "mistakes" into the design process could cause some non-convergent problems to converge to a solution [31]. Past work has primarily emphasized determining if a system will converge or diverge. It has also focused on determining the equilibrium point for convergence. However, convergence and equilibrium provide no information about the designer's convergence rate.

Understanding convergence rate is important in establishing distributed design architectures that converge quickly to the desired solution. It is also important in understanding how local objectives change during a design process. Determining the changes in local objectives during design to investigate the convergence shape is an important area of future research discussed in the *Conclusions* section. In the next section the techniques used to determine convergence and equilibrium for distributed design are leveraged to understand convergence rates for parallel and sequential architectures.

3.3 Convergence Rate

The second major concern for distributed design problems is determining the time required for designers to iterate to a solution. While a large amount of research has focused on the convergence characteristics of distributed design problems, much less focus has been placed on the convergence time. One of the major contributions to examining convergence time was an extension of the DeMAID method to reduce the time required for designers to converge to a solution during iterative loops. In this extension Sobieski utilized the global sensitivity equations [32] with a weighting scheme to predict an optimal ordering of designers in iterative design loops [33].

The approach taken by Sobieski succeeded in reducing the overall design time required for iterative loops in DeMAID by ordering designers sequentially. Smith and Eppinger applied a similar approach to the process used by Sobieski in ordering a design processes using Design Structure Matrices to minimize the number of feedback loops. When feedback loops are present Smith and Eppinger assume all work is conducted in parallel and apply a work transformation matrix to link design tasks based on the amount of rework one designer creates for the other. Based on the strength of these links and the strength of the coupling between systems, which is analogous to the global sensitivity equations, the relative influence of each player or group of players is determined through an eigenvalue analysis [30]. While this approach examines the basic mechanics of the distributed system, it makes no recommendation for the order in which subsystems should iterate to reach equilibrium as quickly as possible. It is shown in Section 3.3 that the rate of convergence and number of iterations required to converge depends on the design architecture chosen.

3.4 Design Process Architecture

In this paper the “architecture” refers to the ordering of the design system. It does not refer to the product architecture, which is an independent and significant area of design research [34]. Instead it refers to the structure of the design process itself. This structure includes both sequential and simultaneous solution processes. Pure sequential or pure simultaneous design architectures are the two extreme cases for design. In Figure 2 a simple diagram is shown illustrating the iterative process for a purely sequential architecture, a purely simultaneous architecture and a hybrid approach utilizing both sequential and parallel elements.

In Figure 2 Chanron defined convergence criteria for sequential design processes. The converged solution is the same regardless of the ordering of Designer 1 and Designer 2. The result is general for systems with n potential designers. In the parallel process shown there is only one potential ordering for designers and the results by Smith and Eppinger are applicable. Once again these results are general for n designers. Convergence for the hybrid architecture was not specifically proven by either Chanron or Smith and Eppinger. However, it follows naturally from their results that the system will require the same convergence criteria.

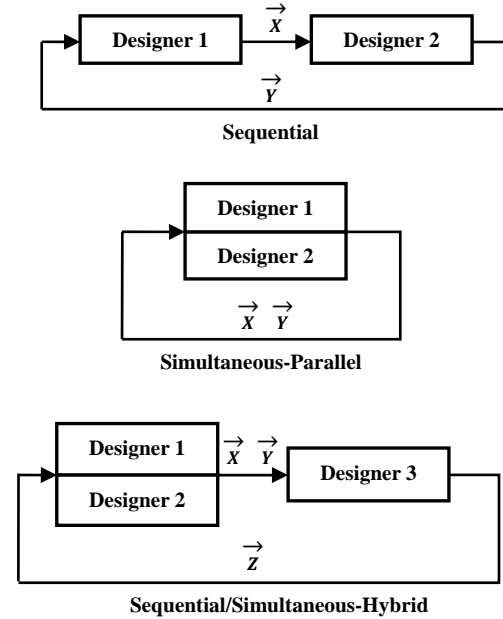


Figure 2: Potential Design Architectures

For the hybrid architecture Designer 3 interacts with Designer 1 and 2 as if they are a single subsystem controlling both \vec{X} and \vec{Y} . If Designer 3 converges individually with Designer 1 and Designer 2, then it also converges with the aggregate sequential system. Similarly, the interaction between Designer 1 and 2 can be viewed as a simple parallel architecture. If the two designers individually converge, then the aggregate system will also converge. If any of the design pairs individually diverge, then the aggregate system will diverge.

While the design architecture does not change whether a given system will converge or diverge, it does have a significant influence on the rate at which convergence occurs. A simple example of this influence can be found by simulating the two designer problem described by Eqn. 1 with two architectures. Since a two designer problem cannot be solved using a hybrid architecture, the two potential choices are a sequential and parallel architecture. The results for this simulation are summarized in Table 1.

Table 1: Two Designer Results

Architecture	Iterations	Time
Sequential	10	20
Parallel	16	16

The convergence criterion for this experiment was chosen to be a minimum change in objective function for each designer of 1% the previous value. The measure of iterations is the number of complete cycles for the whole system. For the sequential process this means Designer 1 and 2 each performed a total of 10 optimizations each with a normalized time value of

1 chosen per iteration. The parallel architecture converged 20% quicker than the sequential process. However, the cost of this increased speed was that each designer performed 16 individual optimizations, 60% more than the sequential process.

This simple example demonstrates the influence the choice of design architecture can have on the design process. It also demonstrates the inherent tradeoff between specifying sequential and parallel design architectures, with parallel architectures requiring significantly more individual iterations for a faster aggregate convergence. The simple problem also presents a challenge for current techniques used to determine design architectures.

The DeMAID genetic algorithm approach provides coupling strengths for the designers, but these strengths do not specify if a parallel or sequential process is desirable. Additionally, it will be shown in Section 4 that regardless of the architecture the eigenvalues associated with the system remain unchanged, a result that makes sense since system convergence was shown to be independent of the architecture chosen. Smith and Eppinger's approach assumed a fully parallel system to examine convergence rates. Since their method is also based on an eigenvalue analysis, it is difficult to extend it to provide useful information about architectures [30].

The inability of these methods to choose between a parallel or sequential architecture is a major shortcoming. Consider the simple hybrid system shown in Figure 2. An eigenvalue analysis would provide no new information to determine the architecture. The other technique available is sensitivity analysis. Applying the global sensitivity equations provides a quantitative measure of subsystem sensitivity.

A general approach has been to place subsystems with low relative sensitivities in parallel with one another. However, the same logic used to discuss convergence can be applied to discuss system architecture. For a simple three designer system, like the one shown in Figure 2, it may be the case that two designers can be placed in parallel based on sensitivity. Designer 1 and 2 may have relatively low joint sensitivities while Designer 3 has a high sensitivity to both Designer 1 and 2. If the two parallel designers are viewed as an aggregate subsystem again, Designer 3 can either be placed in parallel with the system or sequentially. Making this decision is difficult.

This logic can be continued for n designers as pairing designers in parallel based on sensitivity begins to determine the architecture but provides limited insight into specifying how any pairing of subsystems should interact.

In order to determine which design architecture is preferable the basic mechanics governing convergence rates for distributed systems are investigated in Section 4. In this section a general two designer, two design variable unconstrained problem with quadratic objective functions is examined for a sequential and parallel architecture. This investigation illustrates the basic differences between parallel and sequential systems that result in their different convergence rates.

4 CONVERGENCE RATE STUDY

Before proceeding with the analysis of the sequential and parallel architectures, it is appropriate to justify the choice of the case study. A two designer problem was chosen because it captures the differences in convergence rate for these architectures without becoming overly complex due to scaling. The results for the two designer problem can be extended to larger problems. Each designer is assigned a single design variable for the same reason. Dealing with two design variables simplifies the notation and the process used is similar, but more complex, as additional design variables are added.

The most important assumption made for this problem is that quadratic objective functions are used. The reason for choosing a quadratic problem is twofold. The first is that many problems can be modeled using a quadratic function [35]. Further, there is a significant volume of research and techniques that utilize quadratics in their formulation and execution [36]. The second advantage is that quadratic objective functions result in linear rational reaction sets which are either globally convergent or divergent and have a single equilibrium solution.

4.1 Process Breakdown

When examining convergence for distributed design problems the behavior is directly related to the coefficients in the designers' objective function. A general statement for the two designer's quadratic objective functions is shown in Eqn. 2.

$$\begin{aligned} F_1 &= ax^2 + by^2 + cxy + dx + ey + f \\ F_2 &= \alpha x^2 + \beta y^2 + \gamma xy + \delta x + \varepsilon y + \zeta \end{aligned} \quad (2)$$

In this system it is assumed that Designer 1 controls x and Designer 2 controls y . The coefficients of primary concern governing convergence behavior are those appearing in the designers' RRS's. For each designer the partial derivative is taken with respect to their local design variable to determine the RRS which is set equal to zero in Eqn. 3.

$$\begin{aligned} \frac{\partial F_1}{\partial x} &= 2ax + cy + d = 0 \\ \frac{\partial F_2}{\partial y} &= 2\beta y + \gamma x + \varepsilon = 0 \end{aligned} \quad (3)$$

Rearranging Eqn. 3 so that each designer can solve for their local design variable yields Eqn. 4 which is in the same form as the equations used by Chanron [28].

$$\begin{aligned} x &= -\frac{1}{2a}(cy + d) \\ y &= -\frac{1}{2\beta}(\gamma x + \varepsilon) \end{aligned} \quad (4)$$

For sequential design architectures Chanron iterated through the design process using the equations shown in Eqn. 4. The result of substitution is a series of n terms where n is the number of complete iterations between Designers 1 and 2. A similar process can be applied to parallel design architectures. The first step is to write out the several iterations for the

parallel process. In Figure 3 it is assumed the design process begins with initial estimated values for x and y of (x_1, y_1) .

In Figure 3 four iterations are shown to demonstrate the alternating pattern for the change in each designer's design variable. From inspection of the first iteration, x_1 depends upon y_0 and y_1 depends on x_0 . However, in the next iteration, x_2 could be traced back through Designer 2 to x_0 and y_2 could be traced back through Designer 1 to y_0 . The following step, Iteration 3, returns to the same dependence as Iteration 1 while Iteration 4 has the same dependency as Iteration 2. These iterations can be reorganized grouping the odd and even iterations together. The aggregate result of this grouping and simplification for the odd and even iterations of the x design variable is shown by Eqns. 5 and 6 in Table 2. To ease in the interpretation of the results the odd iteration was continued for an additional iteration.

The expressions in Eqns. 5 and 6 have been simplified for only the x variable, but an identical process can be used to express the y variable. The odd and even expressions in Eqns. 5 and 6 can be generalized as a geometric series when the repeated pattern in the terms is recognized. Eqns. 5 and 6 are written in general form for Eqns. 7 and 8 in Table 3.

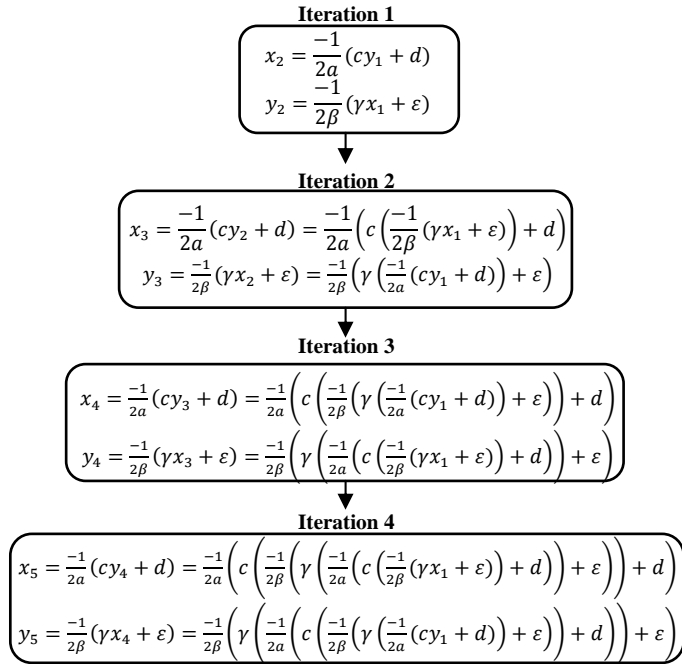


Figure 3: Parallel Iterations

In the odd iteration, Eqn. 8 in Table 2, the term $(\frac{c\varepsilon}{4a\beta})(\frac{4a\beta}{\gamma c})$ is subtracted from the expression. This accounts for the absence of the ε term in the first iteration. A separate power series could have been used to represent ε but consolidating the terms has some advantages that become apparent when manipulating the series. In fact, ε plays a key role in determining the convergence rate for a parallel architecture. The equations in Table 3 can be used to enumerate each step of a parallel design process for two designers controlling a single design variable each. The iteration numbering convention used refers to the n^{th}

iteration as using the n^{th} value of x and y to determine the $n^{\text{th}}+1$ value for x and y . For example, the $n=0$ iteration corresponds to initializing the design variables and results in $x_1 = x_{\text{init}}$. How these variables are initialized is at the discretion of the designer.

Table 2: Odd and Even Design Iterations

Even Iterations – $n = 0, 2, 4, \dots$	
$x_1 = x_{\text{init}}$	
$x_3 = \left(\frac{c\gamma}{2a2\beta}x_1 + \frac{c\varepsilon}{2a2\beta} - \frac{d}{2a}\right)$	
$x_5 = \left(\frac{c\gamma c\gamma}{2a2\beta 2a2\beta}x_1 + \left(\frac{c\gamma}{2a2\beta} + 1\right)\frac{c\varepsilon}{2a2\beta} - \left(\frac{c\gamma}{2a2\beta} + 1\right)\frac{d}{2a}\right)$	(5)
Odd Iterations – $n = 1, 3, 5, \dots$	
$x_2 = -\left(\frac{c\gamma_1}{2a} + \frac{d}{2a}\right)$	
$x_4 = -\left(\left(\frac{c\gamma}{2a2\beta}\right)\frac{c\gamma_1}{2a} + \left(\left(\frac{c\gamma}{2a2\beta}\right) + 1\right)\frac{d}{2a} + \left(\frac{c}{2a2\beta}\right)\varepsilon\right)$	
$x_6 = -\left(\left(\frac{c\gamma}{2a2\beta}\right)^2\frac{c\gamma_1}{2a} + \left(\left(\frac{c\gamma}{2a2\beta}\right)^2 + \frac{c\gamma}{2a2\beta} + 1\right)\frac{d}{2a} + \dots\right)$	(6)
$\dots\left(\left(\frac{c\gamma}{2a2\beta}\right)\left(\frac{c}{2a2\beta}\right) + \frac{c}{2a2\beta}\right)\varepsilon)$	

Table 3: Parallel Iteration in Summation Notation

Even Iteration – $n = 0, 2, 4, \dots$	
$x_{n+1} = \left(\frac{c\gamma}{4a\beta}\right)^{\frac{n}{2}}x_1 + \sum_{j=0}^{\frac{n}{2}-1}\left(\frac{c\gamma}{4a\beta}\right)^j\left(\frac{c\varepsilon}{4a\beta} - \frac{d}{2a}\right)$	(7)
Odd Iteration – $n = 1, 3, 5, \dots$	
$x_{n+1} = \left(\frac{c\gamma}{4a\beta}\right)^{\frac{n-1}{2}}\left(\frac{-c\gamma_1}{2a}\right) + \sum_{j=0}^{\frac{n-1}{2}}\left[\left(\frac{c\gamma}{4a\beta}\right)^j \dots\right]$	(8)
$\dots\left(\left(\frac{c\varepsilon}{4a\beta}\right)\left(\frac{4a\beta}{\gamma c}\right) - \frac{d}{2a}\right) - \left(\frac{c\varepsilon}{4a\beta}\right)\left(\frac{4a\beta}{\gamma c}\right)$	

If convergence is being measured based on successive iterations, then a parallel architecture will converge when the difference between the odd and even series is less than some prescribed threshold value. This is true for both x and y . The threshold required depends on what the maximum change in design variables is specified to be by the designers. These values do not need to be equal for x and y .

Parallel architectures are fundamentally different than sequential architectures when considered from a convergence rate perspective. While parallel architectures have two series describing their behavior, a sequential architecture can be described by a single power series. The sequential result derived by Chanron is shown in Eqn. 9 [28]. Only the constants in Chanron's derivation have been adjusted to match those specified in Eqn. 2.

$$x_{n+1} = \left(\frac{c\gamma}{4a\beta}\right)^{\frac{n}{2}}x_1 + \sum_{j=0}^{\frac{n}{2}-1}\left(\frac{c\gamma}{4a\beta}\right)^j\left(\frac{c\varepsilon}{4a\beta} - \frac{d}{2a}\right) \quad (9)$$

In Chanron's equation each designer iteration causes an increment in the value n . Since the designers take turns performing iterations, the designer controlling x only performs a minimization for every other value of n . The convention used for the parallel system in n is identical to that used by Chanron. In this case Designer 1 operates on even values of $n = 0, 2, 4, 6, \dots$ until convergence is achieved.

As expected the term premultiplying x_i and y_i and the constants in Eqns. 7 and 8 is the same as the term premultiplying x_i and the constants in Eqn. 9. This term was defined by Chanron as k and is equal to $(c\epsilon)/(4a\beta)$ and is critical to the global stability for the series. For the system to be stable, k must have a magnitude less than 1. Since architecture has no influence on a system's global convergence properties, the k term must be the same regardless of design process architecture.

Another advantage of adopting the same conventions used in Eqn. 9 to derive Eqns. 7 and 8 is that the similarity between Eqn. 7 and 9 is obvious. From a convergence perspective, the same series controlling part of the parallel architecture and all the sequential architecture means complete convergence of both series in a parallel system cannot occur with any fewer iterations than a sequential system. This suggests the major difference in convergence rates for the architectures is more closely related to how convergence is defined than any other factor.

Using the fully developed expressions for sequential and parallel architectures an expression can be derived to determine the expected number of iterations until convergence is achieved. This derivation is the focus of Section 5 where the results are presented along with a demonstration using the distributed design system described by Eqn. 2.

5 DETERMINING CONVERGENCE TIME

In this paper the criteria used to determine if a distributed design system has converged is a specified minimum change in design variable value. Specifying an appropriate change is not a trivial task and is problem dependant. In the equations developed to predict convergence time, the change in design variable value is specified by the term δ_{max} . For each architecture, Eqns. 7 through 9 describe how the designer's design variables will change in any given iteration. The simplest architecture from a convergence time perspective is designers sequentially iterating as it depends only on a single geometric series. Therefore, a closed form relationship for the sequential architecture is developed first in Section 5.1. Leveraging the results from studying the sequential architecture, an expression is created for the parallel process in Section 5.2.

5.1 Sequential Architecture Convergence Rate

To determine the convergence time for the sequential architecture, two discrete instances of Eqn. 9 are subtracted from one another. The current iteration is described by x_{n+1} , while the previous is described by x_{n-1} . Corresponding to these values is an equation linking each x_n value back to x_i as shown

in Eqn. 9. If the n^{th} iteration is considered the substitution can be made as shown in Eqn. 10a. Once again the convention adopted by Chanron is used to ease the algebraic manipulation and the equations are written with $k=(c\epsilon)/(4a\beta)$. The threshold value for the maximum change in design variable values between iteration $n+1$ and $n-1$ is specified by the δ_{max} as previously stated. The final solution for the subtraction of the two design variables is shown in closed form for n in Eqn. 10a.

$$\begin{aligned} \delta_{max} &\geq |x_{n+1} - x_{n-1}| \\ \delta_{max} &\geq \left| \left((k)^{\frac{n}{2}} x_1 + \sum_{j=0}^{\frac{n}{2}-1} (k)^j \left(\frac{c\epsilon}{4a\beta} - \frac{d}{2a} \right) \right) - (k)^{\frac{n-2}{2}} x_1 + \sum_{j=0}^{\frac{n-2}{2}-1} (k)^j \left(\frac{c\epsilon}{4a\beta} - \frac{d}{2a} \right) \right| \\ \delta_{max} &\geq k^{\frac{n}{2}-1} \left| (k-1)x_1 + \left(\frac{c\epsilon}{4a\beta} - \frac{d}{2a} \right) \right| \\ n &= \text{ceil} \left(2 \left(\frac{\ln \left(\frac{\delta_{max}}{(k-1)x_1 + \left(\frac{c\epsilon}{4a\beta} - \frac{d}{2a} \right)} \right)}{\ln(k)} + 1 \right) \right) \end{aligned} \quad (10a)$$

Absolute value signs are introduced into Eqn. 10a to account for the minimum move falling within the threshold as a positive or a negative value. There is a requirement on n to be an integer value, since performing partial design iterations is not logical. The *ceil* function rounds n up to the nearest integer value. Finally, since the minimum value of n is desired, Eqn. 10a is an equality rather than an inequality.

To validate Eqn. 10a, the example problem from Eqn. 1 is evaluated for both Designer 1 controlling x and Designer 2 controlling y . To modify Eqn. 10a for Designer 2 requires a slight adjustment to the constants. This adjustment is shown in Eqn. 10b and is used to calculate the value of n for Designer 2.

$$n = \text{ceil} \left(2 \left(\frac{\ln \left(\frac{\delta_{max}}{(k-1)x_1 + \left(\frac{y_d}{4a\beta} - \frac{\epsilon}{2\beta} \right)} \right)}{\ln(k)} + 1 \right) \right) \quad (10b)$$

To provide a useful frame of reference a δ_{max} of 0.003 was selected to match the 1% change in objective function specified in the problem's original solution. The evaluation of Eqn. 10a and 10b, as well as the constants used are summarized in Table 4.

The results for the analytical evaluation match those found by iteration using the two designer's RRS's. The variables n_x and n_y correspond to the number of iterations required for Designer 1 and Designer 2 to reach equilibrium respectively. Since the aggregate system does not converge until both designers converge to a solution, the system converges after 20 iterations when Designer 2 is satisfied with the overall change in design variable value. The analysis performed in this section is repeated to derive an analytical solution for the more complex parallel system.

Table 4: Sequential Convergence Calculation

Designer 1		Designer 2	
x_i	2	y_i	-1
δ_{max}	0.003	δ_{max}	0.003
k	-0.5	k	-0.5
a	1	β	0.5
c	1	γ	-1
d	-3	ε	0
n_x	20	n_y	20

5.2 Parallel Architecture Convergence Rate

Unlike the sequential system, convergence between successive design iterations in a parallel architecture depends upon convergence between two different power series, as shown in Table 3. A similar manipulation as that performed for the sequential process results in an expression for n . Although the algebra is more complicated, the solutions for n can be resolved into relatively simple closed form expressions.

In the derivation each step in the parallel design process is a single iteration for each designer. Therefore, the $n+1$ and n iterations are subtracted to determine the change in design variable value. An additional obstacle to solving this problem is there are two cases for n . When n is an even number, it corresponds to the convergence of iterations such as x_2-x_1 , x_4-x_3 , and x_6-x_5 . When n is an even number, it corresponds to the convergence of iterations such as x_3-x_2 , x_5-x_4 , and x_7-x_6 .

Because of this, two separate equations are necessary to determine the convergence time using the difference between iterations. In Eqns. 14 and 15 the final result is solved for n . However, since the process is less obvious than the sequential approach, it is explained step by step. The first analysis is performed for the case when x_{n+1} is calculated using the odd series. This corresponds to terms of the form x_2-x_1 since odd iterations produce even subscripted x values in the convention chosen. In this analysis x is considered at iteration $n+1$ and n . The same convention for k and δ_{max} as the sequential problem is applied.

$$\delta_{max}^{odd} = |x_{n+1}^{odd} - x_n^{even}|$$

$$\delta_{max}^{odd} = \left| \left((k)^{\frac{n-1}{2}} \left(\frac{-cy_1}{2a} \right) + \sum_{j=0}^{\frac{n-1}{2}} (k)^j \left(\frac{\varepsilon}{\gamma} - \frac{d}{2a} \right) - \frac{\varepsilon}{\gamma} \right) - \left((k)^{\frac{n-1}{2}} x_1 + \sum_{j=0}^{\frac{n-1}{2}-1} (k)^j \left(\frac{\varepsilon k}{\gamma} - \frac{d}{2a} \right) \right) \right| \quad (11)$$

In Eqn. 11 it is desirable to obtain a form to isolate the n term, but in order to do so the two series must be resolved. The major obstacle in this difference is resolving the ε/γ and $\varepsilon k/\gamma$ terms. The key to eliminating the terms is to realize that $\varepsilon k/\gamma$ is essentially 1 iteration ahead of ε/γ . Therefore, subtracting the two series leaves two terms remaining, ε/γ and $(\varepsilon/\gamma)k^{(n-1)/2}$. Since the constant in the odd series is equal to ε/γ , the only remaining term is $(\varepsilon/\gamma)k^{(n-1)/2}$, which can be grouped with x_1 and y_1 to yield a single expression. These manipulations are outlined in Eqn. 12.

$$\begin{aligned} \delta_{max}^{odd} &\geq \left| (k)^{\frac{n-1}{2}} \left(\frac{-cy_1}{2a} - x_1 \right) \dots \right. \\ &\quad \dots + \left(\frac{\varepsilon}{\gamma} - \frac{d}{2a} \right) \left(1 + k + k^2 + \dots + k^{\frac{n-1}{2}} \right) - \frac{\varepsilon}{\gamma} \dots \\ &\quad \left. \dots - \left(\frac{\varepsilon k}{\gamma} - \frac{d}{2a} \right) \left(1 + k + k^2 + \dots + k^{\frac{n-1}{2}} \right) \right| \\ \delta_{max}^{odd} &\geq (k)^{\frac{n-1}{2}} \left| \frac{-cy_1}{2a} - x_1 + \frac{d}{2a} \right| \quad (12) \end{aligned}$$

A notable feature of Eqn. 12 is that when y_1 lies on Designer 1's RRS, the k coefficient is exactly 0. From a convergence perspective, determining the convergence rate is simpler when the design begins on one of the designers RRS's, since only a single series needs to be evaluated per designer.

The second case of interest occurs when n has an odd value. In this case x_{n+1} is calculated by the odd series. This corresponds to terms of the form x_3-x_2 since even iterations produce odd subscripted x values in the chosen convention. The manipulation to isolate n for this series is similar to that found in Eqn. 12 with the $\varepsilon k/\gamma$ and ε/γ being resolved into a single expression dependent upon $(\varepsilon/\gamma)k^{(n-1)/2}$. The final result of the manipulation is shown in Eqn. 13.

$$\delta_{max}^{even} = |x_{n+1}^{even} - x_n^{even}|$$

$$\begin{aligned} \delta_{max}^{even} &\geq \left| (k)^{\frac{n}{2}} x_1 + \sum_{j=0}^{\frac{n}{2}-1} (k)^j \left(\frac{\varepsilon k}{\gamma} - \frac{d}{2a} \right) - \right. \\ &\quad \left. \left((k)^{\frac{n-2}{2}} \left(\frac{-cy_1}{2a} \right) + \sum_{j=0}^{\frac{n-2}{2}-1} (k)^j \left(\frac{\varepsilon}{\gamma} - \frac{d}{2a} \right) - \frac{\varepsilon}{\gamma} \right) \right| \\ \delta_{max}^{even} &\geq (k)^{\frac{n}{2}-1} \left| kx_1 + \frac{cy_1}{2a} + \frac{\varepsilon k}{\gamma} \right| \quad (13) \end{aligned}$$

The combination of Eqns. 12 and 13 can be used to determine the convergence time for a distributed design system. Similar to the sequential process they can also be manipulated into a closed form solution for n . In this manipulation the convention n_{even} is used to denote the solution for an even iteration and n_{odd} the solution for an even iteration. With this convention to differentiate between the iterations, the δ_{even} and δ_{odd} can simply be expressed as δ_{max} since it will be identical for both odd and even valued n . It is assumed that the minimal value of n is desired, which is the value corresponding to each expression equaling δ_{max} . The final expressions for n_{even} and n_{odd} are shown in Table 5.

Table 5: Closed Form Parallel Solutions

$$n_{odd} = \text{ceil} \left(\frac{2 \ln \left(\frac{\delta_{max}}{\frac{-cy_1}{2a} - x_1 + \frac{\varepsilon k}{\gamma}} \right)}{\ln |k|} + 2 \right) \quad (14)$$

$$n_{even} = \text{ceil} \left(\frac{2 \ln \left(\frac{\delta_{max}}{kx_1 + \frac{cy_1}{2a} + \frac{\varepsilon k}{\gamma}} \right)}{\ln |k|} + 1 \right) \quad (15)$$

Once again the *ceil* function is used to guarantee that the solutions for n have integer values. In spite of the more complex nature of the parallel architecture, the expression for n is very similar to the sequential case. Similar to Eqn. 10 for a sequential architecture, Eqns. 14 and 15 can be applied to the example problem in Eqn. 1. In this case direct substitution into the parallel equations for Designer 2 is possible provided a , c and d are substituted for β , γ , and ε . The results are summarized in Table 6.

Table 6: Parallel Convergence Calculation

Designer 1		Designer 2	
x_i	2	y_i	-1
δ_{max}	0.003	δ_{max}	0.003
k	-0.5	k	-0.5
a	1	β	1
c	1	γ	-1
d	-3	ε	0
n_{even}	16	n_{even}	2
n_{odd}	2	n_{odd}	20

Some interpretation of the results for this case is necessary. For each designer, Eqns. 14 and 15 predict a value of n . In this case convergence occurs when the change in design variable value is less than δ_{max} . The reason the odd and even series converge so quickly for Designer 1 and Designer 2 respectively is because the initial point (x_1, y_1) was taken to be on Designer 1's RRS. If a substitution for y_1 is made in terms of x_1 , based on Designer 1's RRS, then the denominator goes to 0, which results in the identity $\ln(\infty) = \infty$. This does not mean the overall system diverges. Rather, it suggests that any value of n will result in a solution with a minimum change in design variable values less than δ_{max} .

The predicted convergence value for this system is $n = 16$. This value corresponds to the time it takes Designer 1 and 2 to both reach a minimal change in the design variable in the same iteration. This result confirms the value found through simulation of 16 iterations in Table 1.

One of the key assumptions for each series is that the difference between the design variables is always decreasing. Based on inspection of Eqns. 12 and 13, all the terms depend upon k raised to a function of n . This indicates that for one iteration the difference between the design variables is indeed decreasing since k is less than 1.

To validate the equations for parallel and sequential convergence developed in this section a large number of combinations of quadratic objective functions are analyzed. The results for the predicted number of iterations to convergence are then verified by simulation to determine the approach's overall accuracy.

6 VALIDATION STUDY

To support the results found in Section 5, a validation study is performed for a wide range of potential distributed design problems. The design problems studied were randomly generated using uniform distributions. The bounds of these

distributions are summarized in Table 7. The only restriction placed on their value, other than the allowable range of the constants and design variables themselves, was only convergent systems were investigated. To guarantee this, after the variables were generated for a potential problem, the k value for the system was calculated. If its absolute value was less than 1, then the simulation proceeded. If it was not, then a new combination was chosen.

Table 7: Validation Parameter Range

Designer 1			Designer 2		
Variable	Min.	Max	Variable	Min.	Max
a	-3	3	α	-8	8
c	-7	7	β	-4	4
d	-6	6	γ	-2	2
x_1	-9	9	y_1	-5	5

Overall 10,000 randomly generated different convergent systems were evaluated in the study. The parallel architectures on average took 16 iterations to converge and the sequential architectures took 20 iterations to converge. A calculation of the percent error between the simulated and analytical solutions for each system was performed using Eqn. 16.

$$Error = \frac{n_{sim} - n_{analytic}}{n_{sim}} \times 100\% \quad (16)$$

In Eqn. 16, n_{sim} is the number of iterations for the simulated system to converge and $n_{analytic}$ is the predicted number of iterations required to converge from the analytic approach. For the sequential architectures, the average error across all 10,000 systems was ~20%, or approximately 4 iterations. For the parallel architectures, the average error was ~10%, or approximately 2 iterations.

The error results suggest the analytical approach provides a very good approximation for the convergence time of any distributed two designer system. However, the results also demand further exploration for why at least some of the predictions were not exact in all cases. Systems that had a prediction error greater than 20% of the mean convergence time (a predictive error of more than 4 iterations) were further studied. Approximately 300 sequential architectures and 150 parallel architectures met this criteria and further analysis of these cases using Eqns. 10, 14 and 15 found that the difference between successive iterations was accurate to at least 4 decimal places for at least the first six iterations. However, as the number of iterations increases, the disparity between the simulated and analytic prediction increased. The primary cause for this is the fact that k is raised to an exponent that is a linear function of n , the number of iterations. As n increases, so do the numerical errors in determining $k^{n/2}$.

There are two cases that can result in a large n . The first case is when k is close to 1. Since k is a function of a , c , β , and γ , certain combinations of these parameters cause k to be near 1. Therefore, numerical errors resulting from large n

corresponding to a k near 1 is one primary source of the difference between the simulated and analytical results.

The other case is the presence of a large multiplying constant within the absolute value brackets for the expressions of δ_{max} in Eqns. 10, 14 and 15. This case is less significant than the previous one because the maximum value of the multiplying constant is relatively small compared to the decay rate.

Elimination of these ill conditioned problems from the data set resulted in a dramatic decrease in average error to less than ± 1 iteration for both the sequential and parallel architectures. Overall, only 1.5% and 3% of the 10,000 parallel and sequential systems respectively had predictive errors of greater than 4 iterations. This result suggests the analytic calculation was quite accurate for the vast majority of the systems.

To investigate further, an evaluation was performed to determine if there was any correlation between the occurrence of ill conditioned problems and the size of k . 10,000 systems are analyzed for each value of k between 0.1 and 1.0 with an increment of 0.01. Figure 4 plots the percentage of systems that have a predictive error of more than 4 iterations as a function of k .

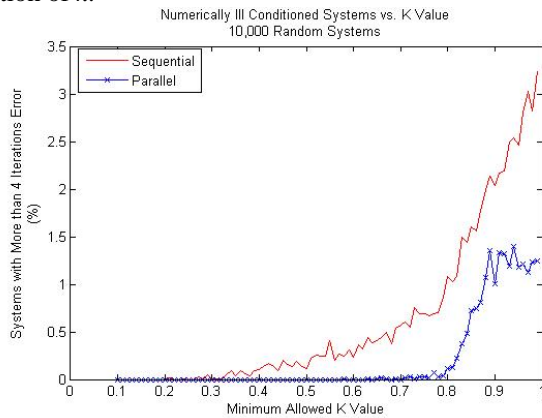


Figure 4: Prediction Error vs. Minimum K Value

From Figure 4, the trend of increasing error with an increasing value of k is clearly seen. The primary reason for this is that as k increases, it gets closer to the $k=1$ threshold where the system starts to diverge instead of converge. Also, systems with k values near 1 generally converge slower than systems with smaller k values. This increase in iterations simply means more multiplications of k , which increase the potential for predictive error.

7 CONCLUSIONS

The contribution of this paper is investigating convergence rate behavior for sequential and parallel distributed design architectures. As a starting point the investigation was restricted to simple two designers, two design variable problems. The principles derived from these simple architectures are potentially applicable to larger, more complex systems.

For the two designer, two design variables problem a geometric series expression was derived to determine the

design variable value at a prescribed iteration. The expression derived is applicable to quadratic objective functions. It was found that unlike a sequential architecture, the behavior of a parallel architecture is determined by the interaction of two geometric series.

The power series derived was used to create closed form solutions for the number of iterations to convergence for both sequential and parallel architectures. These expressions provide a closed form solution to predict the number of iterations to convergence with no objective function evaluations. During validation the analytical expressions were found to be accurate for 97% of the analyzed systems. The remaining 3% of systems corresponded to cases that led to numerical instabilities.

There are several areas of future work to expand upon the initial investigation in this paper. The first is to enhance the approach to encompass large design systems with multiple designers controlling multiple design variables. Extending this work to multiple variables requires the evaluation of additional power series. In enhancing the approach to accommodate these power series, linear control theory will be an important tool. Linear control theory provides a sound theoretical basis to manipulate the power series' algebraic expressions.

Another challenge to accommodating more than two designers is hybrid architectures become possible. The convergence rate of hybrid architectures will depend on more than one power series, similar to the parallel architecture. Additionally, the number of potential architectures increases very quickly with the addition of designers.

To manage the large number of potential architectures branch and bound optimization approaches will play a key role. Developing an effective bounding strategy can greatly reduce the number of architectures that must be evaluated. To efficiently create power series, a generalized formulation is needed that can represent parallel, sequential and hybrid architectures. It is likely this representation will be a rules based approach.

The second topic of future work is to refine the current technique to better understand the role of starting location on convergence time and final solution quality. One particular case for the parallel architecture demonstrated that choosing a starting location on either designer's RRS will result in convergence in two iterations for that specific series. When the overall system converged, the designer with the starting location on their rational reaction set converged in both the odd and even series while the other designer had converged only in one series. Examining which designer benefits from this phenomena could influence the strategy chosen in determining initial start points for design variables.

Finally, convergence is typically measured as a function of the change in objective function. By mapping the changes in design variables to specific objective function adjustments the shape of the convergence plot based on objective function can be investigated. Determining these convergence shapes to provide visualization of the design process is of critical importance to better managing distributed design problems.

ACKNOWLEDGMENTS

We would like to thank the National Science Foundation, grant DMII-0322783 and the Moog Corporation for their support of this work.

REFERENCES

- [1] Greising, D., and Johnsson, J., 2007, "Behind Boeing's 787 Delays," *Chicago Tribune*, Chicago, IL, 10 December.
- [2] Yu, R., 2008, "Boeing Again Delays Dreamliner's Debut," *USA Today*, Mclean, VA, 16 January.
- [3] Gates, D., 2009, "Strike, Supplier Problems Cause Boeing to Deliver 100 Fewer Jets Last Year Than Expected," *Seattle Times*, Seattle, WA, 9 January.
- [4] Wheelright, S. C., and Clark, K. B., 1992, *Revolutionizing Product Development, Quantum Leaps in Speed, Efficiency, and Quality*, The Free Press, NY.
- [5] Meyer, C., 1993, *Fast Cycle Time, How to Align Purpose, Strategy, and Structure for Speed*, The Free Press, New York, NY.
- [6] Sobieszczanski-Sobieski, J., and Haftka, R. T., 1997, "Multidisciplinary Aerospace Design Optimization: Survey of Recent Developments," *Structural and Multidisciplinary Optimization*, 14(1), pp. 1-24.
- [7] Khajavirad, A., and Michalek, J., 2008, "A Deterministic Lagrangian-Based Global Optimization Approach for Large Scale Decomposable Problems," *Design Automation Conference: DETC2008-50029*, Brooklyn, NY.
- [8] Cramer, E. J., Dennis, J. E., Frank, P. D., Lewis, R. M., and Rshubin, G. R., 1994, "Problem Formulation for Multidisciplinary Optimization," *SIAM Journal of Optimization*, 4(4), pp. 754-776.
- [9] Suh, N. P., 2001, *Axiomatic Design: Advances and Applications*, Oxford University Press, New York, NY.
- [10] Pahl, G., and Beitz, W., 1996, *Engineering Design: A Systematic Approach*, Springer, New York, NY.
- [11] Pugh, S., 1996, *Creating Innovative Products Using Total Design*, Addison-Wesley Pub. Co., Reading, MA.
- [12] Wiecekm, M. M., and Reneke, J. A., 2005, "Complex Systems Decomposition under Uncertainty and Risk," 6th World Congress of Structural and Multidisciplinary Optimization, Rio de Janeiro, BR.
- [13] Kim, H. M., Michelena, N. F., Papalambros, P. Y., and Jang, T., 2003, "Target Cascading in Optimal System Design," *Journal of Mechanical Design*, 125(3), pp. 474-581.
- [14] Wujek, B., Renaud, J. E., and Batill, S., 1995, "A Concurrent Engineering Approach for Multidisciplinary Design in a Distributed Computing Environment," *Multidisciplinary Design Optimization: State of the Art*, Hampton, VA, NASA, March 13-16.
- [15] Wujek, B. A., Renaud, J. E., Batill, S. M., and Brockman, J. B., 1996, "Concurrent Subspace Optimization Using Design Variable Sharing in a Distributed Computing Environment," *Concurrent Engineering: Research and Applications*, 4(4), pp. 361-377.
- [16] Sobieszczanski-Sobieski, J., Agte, J. S., and Sandusky, R. J., 1998, "Bi-Level Integrated System Synthesis (BLISS)," *NASA Paper No. 208715*.
- [17] Braun, R., 1996, "Collaborative Optimization: An Architecture for Large-Scale Distributed Design," Ph.D. Dissertation, Stanford University, Palo Alto, CA.
- [18] Michalek, J. J., Feinberg, F. M., and Papalambros, P. Y., 2004, "An Optimal Marketing and Engineering Design Model for Product Development Using Analytical Target Cascading," *Proceedings of the Tools and Methods of Competitive Engineering Conference*, Lausanne, Switzerland, April 13-17, pp. 835-846.
- [19] Lewis, K. E., 1998, "The Tradeoffs between Cooperative and Approximate Cooperative Formulations in Multidisciplinary Design," 7th AIAA/ USAF/ NASA/ ISSMO Symposium on Multidisciplinary Analysis and Optimization, AIAA-99-4923, St. Louis, MO.
- [20] Chanron, V., and Lewis, K., 2006, "A Study of Convergence in Decentralized Design Processes," *Research in Engineering Design*, 16(3), pp. 133-145.
- [21] Rogers, J. L., and Bloebaum, C. L., July 1994, "Ordering Design Tasks Based on Coupling Strengths," *NASA Paper No. 109-137*.
- [22] Denker, S., Steward, D. V., and Browning, T. R., 2001, "Planning Concurrency and Managing Iteration in Projects," *Project Management Journal*, 32(3), pp. 31-38.
- [23] Vincent, T. L., 1983, "Game Theory as a Design Tool," *Journal of Mechanisms, Transmissions and Automation in Design*, 105(June), pp. 165-170.
- [24] Lewis, K., and Mistree, F., 1997, "Collaborative, Sequential and Isolated Decisions in Design," *Design Theory and Methodology Conference: DETC1997-3883*, Sacramento, CA.
- [25] Nash, J., 1951, "Non-Cooperative Games," *The Annals of Mathematics* 54(2), pp. 286-295.
- [26] Lewis, K., and Mistree, F., 2001, "Modeling Subsystem Interactions: A Game Theoretic Approach," *Journal of Design Manufacturing and Automation*, 1(1), pp. 17-36.
- [27] Chanron, V., and Lewis, K., 2004, "Convergence and Stability in Distributed Design of Large Systems," *Design Automation Conference, DETC2004-57344*, Montreal, Canada.
- [28] Chanron, V., and Lewis, K., 2003, "A Study of Convergence in Decentralized Design," *Design Automation Conference, DETC2003-48782*, Chicago, IL.
- [29] Chanron, V., Singh, T., and Lewis, K., 2004, "An Investigation of Equilibrium Stability in Decentralized Design Using Nonlinear Control Theory," *AIAA-2004-4600*, Albany, NY.
- [30] Smith, R. P., and Eppinger, S. D., 1997, "Identifying Controlling Features of Engineering Design," *Management Science*, 43(3), pp. 276-293.
- [31] Gurnani, A., and Lewis, K., 2006, "Decentralized Design at the Edge of Rationality," *Design Theory and Methodology Conference, DETC2006-99530*, Philadelphia, PA.

- [32] Sobieszczanski-Sobieski, J., 1990, "On the Sensitivity of Complex, Internally Coupled Systems," *AIAA Journal*, 28(1), pp. 713-180.
- [33] Rogers, J. L., 1996, "DEMAID/GA: An Enhanced Design Manager's Aid for Intelligent Decomposition," 6th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, AIAA 1996-4157, Seattle, WA.
- [34] Eppinger, S. D., and Ulrich, K. T., 2008, *Product Design and Development*, McGraw Hill, New York, NY.
- [35] Perez, V. M., Renaud, J. E., and Watson, L. T., 2002, "Reduce Sampling for Construction of Quadratic Response Surface Approximations Using Adaptive Experimental Design," 43rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, AIAA 2002-1587, Denver, CO.
- [36] Wang, G., Dong, Z., and Aitchison, P., 2001, "Adaptive Response Surface Method-a Global Optimization Scheme for Computation Intensive Design Problems," *Journal of Engineering Optimization*, 33(6), pp. 707-734.