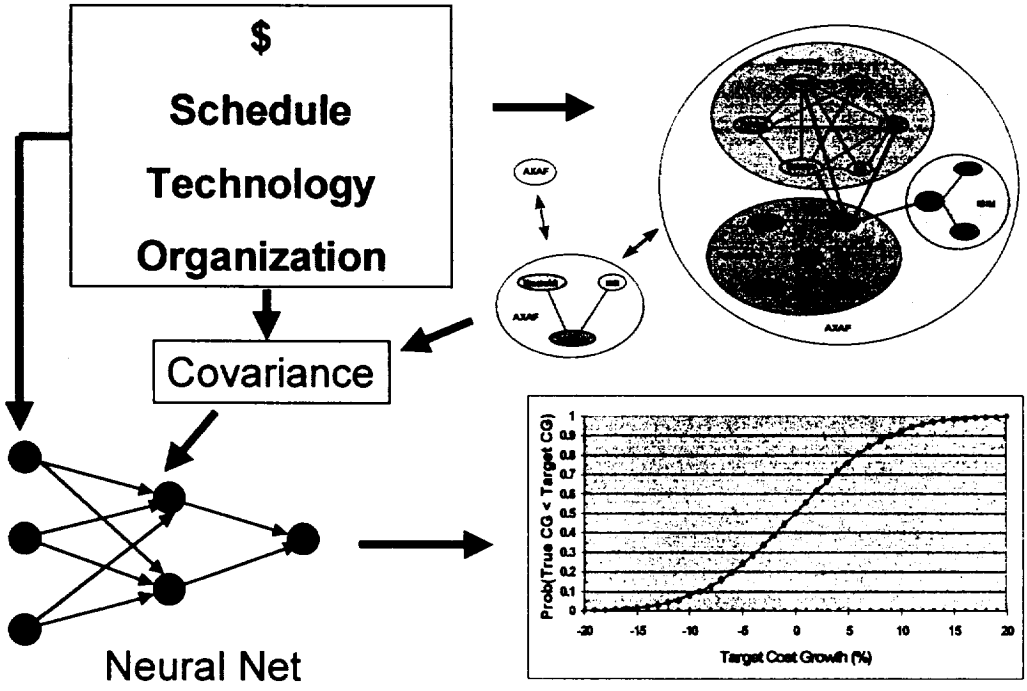
OR **APPLICATIONS**

*Innovative and Cost-Effective Operations Research Solutions*

# Systems Engineering Design Via Experimental Operations Research

**(H-27473D Final Report)**

## Complex Organizational Metric for Programmatic Risk Environments (COMPRÉ)




**For**

**NASA-MSFC**

**by**

**OR Applications**

**April 1999**

7811 Lent Drive • Huntsville, Alabama 35802 • (256) 880-1132

# Acknowledgments

The author wishes to thank **Dr. Dale Thomas**, EL61, for his role as architect of the COMPRÉ program management tool, while serving as COTR for this effort. I also wish to thank Dr. Thomas for freely allowing me to utilize his technical concepts and systems complexity terminology. I wish to thank Tara Claborn, Sharon Czarnecki, Wayne Johnson, and Spencer Hill of SAIC for their exhaustive independent verification of the concepts described in this report, for their cost and COMPRÉ data collection efforts while exercising the tool, and for their successful initial deployment of the tool during the initial pilot studies. Thanks are also due to Dr. Tze-San Lee from Western Illinois University for his independent V&V of the COMPRÉ statistics. Finally, the author thanks the numerous NASA-MSFC engineers and program managers for five successful peer reviews of this concept.

# TABLE OF CONTENTS

# Systems Engineering Design Via Experimental Operations Research


## Abstract

Unique and innovative graph theory, neural network, organizational modeling, and genetic algorithms are applied to the design and evolution of programmatic and organizational architectures. Graph theory representations of programs and organizations increase modeling capabilities and flexibility, while illuminating preferable programmatic/organizational design features. Treating programs and organizations as neural networks results in better system synthesis, and more robust data modeling. Organizational modeling using covariance structures enhances the determination of organizational risk factors. Genetic algorithms improve programmatic evolution characteristics, while shedding light on rulebase requirements for achieving specified technological readiness levels, given budget and schedule resources. This program of research improves the robustness and verifiability of systems synthesis tools, including the Complex Organizational Metric for Programmatic Risk Environments (COMPRÉ).


## Executive Summary: Findings

1. Chromatic number (a graph metric) is a statistically significant measure of programmatic complexity, and, therefore, is included as a variable in the current COMPRÉ model. Chromatic number also appears to work well in capturing organizational complexity through covariance.
2. There are tradeoffs among user-friendliness, model realism, and input reliability for various levels of architectural decomposition. Level III COMPRÉ decompositions appear to be a good balance of these factors.
3. Artificial neural networks significantly outperform sophisticated nonlinear regressions in modeling system complexity for NASA programs. An artificial neural network performs a macro-assessment on global variable inputs in COMPRÉ.
4. Component-level cost growth forecasting addresses program evolution, and is employed in COMPRÉ as a generator for cost growth risk.
5. COMPRÉ includes a micro-assessment of programmatic complexity through interaction and a macro-assessment of global parameters through synthesis. These two types of assessments simultaneously complement and compete with each other to determine the metric of system complexity.

### 1.0 Introduction

Systems engineering (SE) is a field that is generally viewed as the process of formulating and solving problems at a very high conceptual level. The efficacy of SE tools is determined by their ability to be broadly applied across engineering disciplines. However, broad application is a characteristic which often competes with meaningfulness of results. In other words, the features of an SE tool which make it broadly applicable to

numerous disciplines may not be consistent with obtaining highly meaningful results in a specific engineering discipline. **Generality competes with specificity.**

Operations research (OR) is sometimes regarded as a subset of SE, with particular emphasis on process and operational problems. Applications of OR tend to be fairly problem specific and technical in nature. While OR methods can often be broadly applied, significant parameter tuning is nearly always required, and this tuning is quite dependent on the characteristics of the problem at hand. For these reasons, OR usage requires a thorough knowledge of mathematics and algorithms. OR practitioners also tend to have their favorite methods, familiarity being what it is. **Because of this, practitioners tend to stick with a relatively small "bag of tricks", despite the fact that their knowledge accommodates a much larger set of OR methods.**

The author believes that a relevant subfield of operations research, called **experimental OR**, would be useful to the practicing and academic communities. By experimental OR, the author is referring to the combination of OR methods in unique and innovative ways, with the output being a broadly applicable method with desirable problem specific features. **The efficacy of experimental OR is highly dependent on the ability of the experimenter to rapidly turn around small failures in pursuit of a greater success.** This means that the experimenter cannot get "locked-in" to his or her favorite method. The approach is analogous to the mixing of ingredients to produce a desired recipe. **Like the recipe process, it is more important to obtain the right mixture of ingredients (methods) than it is to use only the best, or most sophisticated, ingredients.**

## 1.1 Study Objectives

The objective of this study is to apply the experimental OR approach using the ingredients of graph theory, neural networks, organizational covariance, and genetic algorithms to arrive at a significant improvement in systems engineering, as it applies to NASA's programmatic controls. These efforts help to answer the following key questions:

1. To what extent can the related fields of graph theory and neural networks shed light on the analysis, control, and design of programmatic architectures?

2. To what extent can organizational covariance be used to improve the development of organizational risk factors in risk tools such as COMPRÉ?

3. To what extent can genetic algorithms be used to evolve the analysis structures and subtle features of programmatic architectures, as in COMPRÉ?

4. What is a suitable mix of graph theory, neural networks, organizational covariance, and genetic algorithms in this context?

### 1.2 Significance

The significance of this research and development effort includes increased modeling capabilities and flexibility, preferable programmatic/organizational design features, better system synthesis, more robust data modeling, enhanced determination of organizational risk factors, improved programmatic evolution characteristics, and better understanding of rulebase requirements for achieving specified technological readiness levels, given budget and schedule resources.

### 1.3 Statement of Uniqueness

The experimental OR approach is a unique and innovative methodology for this field. The combination of graph theory, neural networks, organizational covariance, and genetic algorithms have not been previously applied to large complex projects of the type that NASA frequently encounters.

## 2.0 Results

In this section, the results of the efforts are given. These results include the products of innovative research, development, and application of graph theory, neural network, organizational covariance, and genetic algorithm methodologies as applied to the design of programmatic and organizational architectures. While the results of these developments are standalone, they also support certain features of the Complex Organizational Metric for Programmatic Risk Environments (COMPRÉ).

### *2.1 Task 1: Graph Theory Approach to Programmatic Hierarchies*

*"The fact that a simplex has a natural geometric dimension in which it "lives" suggests that trying to force anything associated with this object into a lower dimension is going to introduce some stress into the system."* - John L. Casti, Complexification

Task 1 involves the investigation of the use of graph theory concepts and applications to the design of programmatic hierarchies. This investigation includes the determination of preferable characteristics for programmatic representation and robust operations.

#### 2.1.1 Graph Theory Fundamentals

A **graph**, G, is a finite nonempty set V together with an irreflexive, symmetric relation R on V. As an example, define $V = \{v_1, v_2, v_3, v_4\}$ and $R = \{(v_1,v_2), (v_2, v_4), (v_3, v_2), (v_4,v_1)\}$ as the following graph:

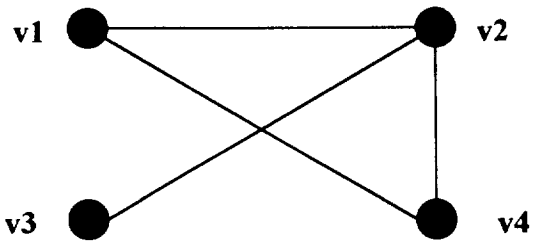



**Figure 2.1.1-1. A Simple Graph Representation**

Figure 2.1.1-2 provides a graph of the Solid Rocket Booster (SRB) subsystems at delivery (DEL). Note that this graph is "fully connected," in that each subsystem is connected to every other subsystem. For each subsystem listed, the first number following the subsystem name is the NASA Technology Readiness Level (TRL). This is followed by the proportion of the budget allocated to the subsystem, and finally, the schedule duration in years.

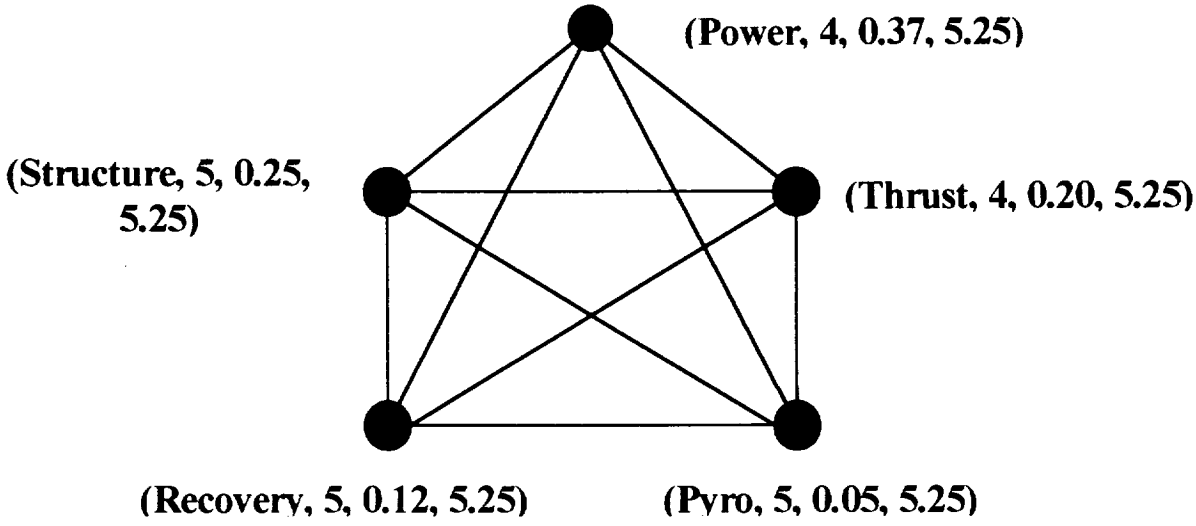



**Figure 2.1.1-2. A Simple Graph Representation of the SRB at DEL**

Table 2.1.1-1 shows the COMPRÉ input matrix for the Mars Pathfinder (MARS) at delivery. The individual subsystems are established, along with their respective durations (t), technology readiness levels (TRL), and organizational interface factors, represented by the integers in the matrix. A nonzero value in the matrix indicates an existing connection between subsystems represented by the corresponding row and column intersection. Figure 2.1.1-3 shows the graph resulting from the interactions represented by the input matrix.

**Table 2.1.1-1. COMPRÉ Inputs for MARS at DEL**

| | Subsystem | $ | t | TRL | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Mech. Subs | | 4.1 | 6 | S | | | | | | | | | | | | |
| 2 | Thermal Control | | 4.1 | 6 | 2 | Y | | | | | | | | | | | |
| 3 | Entry & Descent | | 4.1 | 4 | 2 | 2 | M | | | | | | | | | | |
| 4 | Elect. Power | | 4.1 | 6 | 2 | 2 | 2 | M | | | | | | | | | |
| 5 | Telecomm. | | 4.1 | 5 | 2 | 2 | 2 | 2 | E | | | | | | | | |
| 6 | Attitude & Info. | | 4.1 | 5 | 2 | 2 | 2 | 2 | 2 | T | | | | | | | |
| 7 | Flt. SW | | 4.1 | 5 | | | | | 2 | 2 | R | | | | | | |
| 8 | Reaction Control | | 4.1 | 6 | 2 | 2 | 2 | 2 | 2 | 2 | | I | | | | | |
| 9 | Rocket Assist | | 4.1 | 6 | 2 | 2 | 2 | 2 | 2 | 2 | | | C | | | | |
| 10 | Imager for Path | | 4.1 | 4 | 3 | 2 | | 2 | 2 | 2 | | | | & | | | |
| 11 | Atmos. Str. | | 4.1 | 4 | 3 | 2 | | 2 | 2 | 2 | | | | 2 | & | | |
| 12 | APX Spectrom. | | 4.1 | 4 | 3 | 2 | | 2 | 2 | 2 | | | | 2 | 2 | & | |
| 13 | MicroRove | | 4.1 | 4 | 2 | 2 | | 2 | 2 | 2 | | | | 2 | 2 | 2 | & |

Figure 2.1.1-4 shows the programmatic architecture of AXAF at delivery for various levels of decomposition. The Level I decomposition is shown in the upper left portion of the figure, and is merely the designation for the system itself, AXAF. The Level II decomposition is shown in the lower left portion of the figure. This level of decomposition is an initial breakdown of the program, AXAF, into its three major subsystems, Spacecraft, Telescope, and ISIM. Finally, the Level III decomposition (right portion of figure) further reduces the three major subsystems into their major components. Of particular note is the refinement of interfaces as the decomposition progresses. For instance, Spacecraft and Telescope interface at Level II, but only the OBA component of the Telescope subsystem interfaces with components in the Spacecraft at Level III. Thus, there is a decomposition level tradeoff involved in analyzing the program using COMPRÉ. Lower levels of decomposition require greater data collection efforts, but provide better interface modeling and more detailed analyses. Because of this balancing among user-friendliness, model realism, and input reliability, the author believes that the optimal level of decomposition for most programs is Level III.

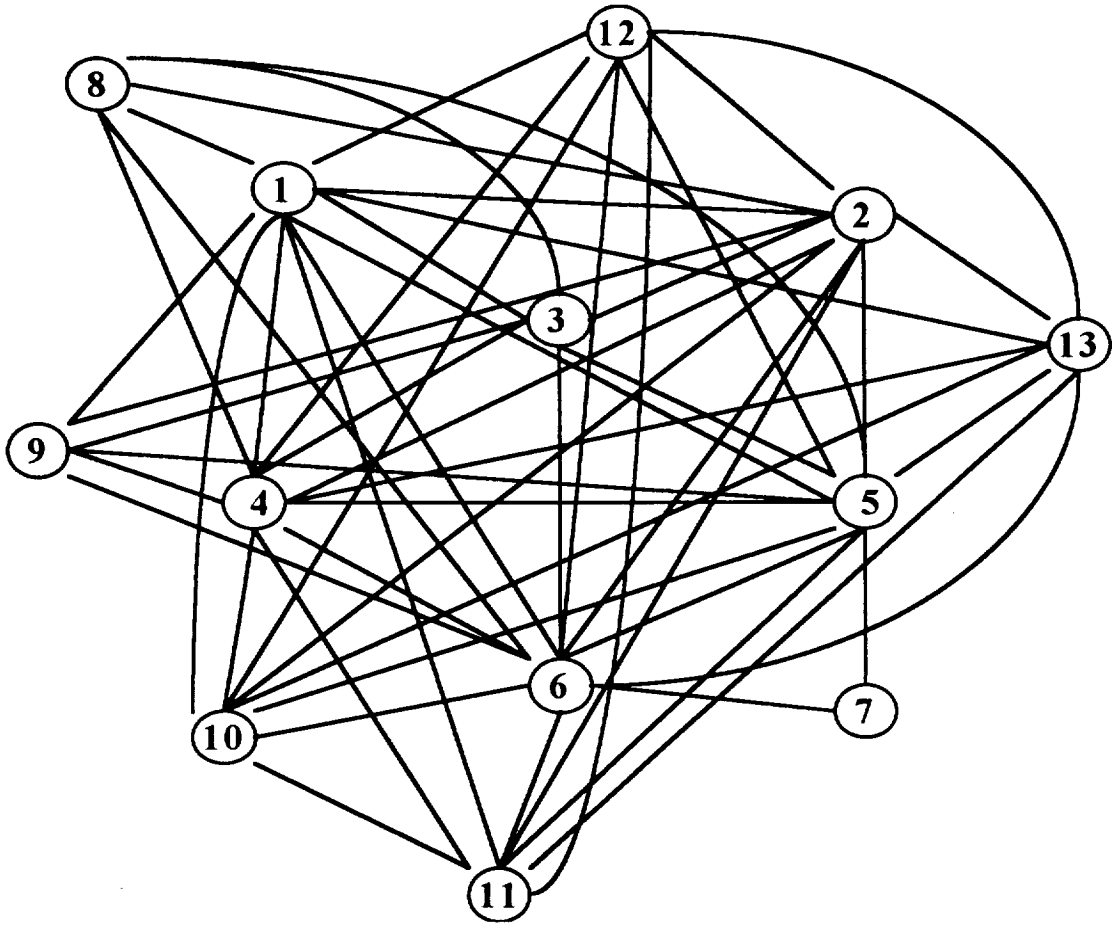



**Figure 2.1.1-3. A Simple Graph Representation of MARS at DEL**

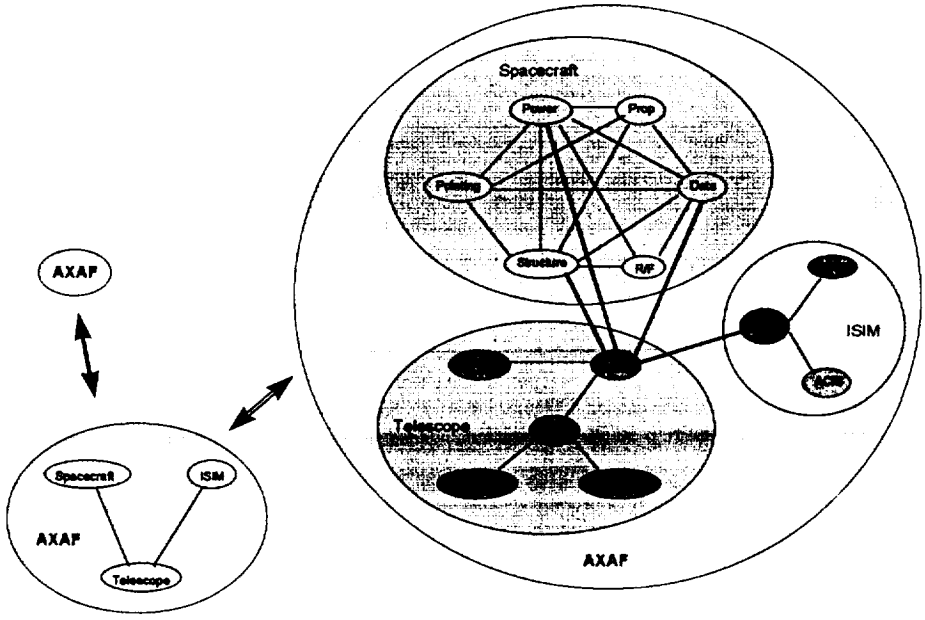



**Figure 2.1.1-4. An Architectural Decomposition of AXAF at DEL**

The vertices, $v_1$, $v_2$, $v_3$, and $v_4$, of V help form the edges of the relation R. Adjacent vertices are connected in a graph, e.g., $v_1$ and $v_2$ are adjacent, but $v_3$ and $v_4$ are nonadjacent vertices. Directed graphs, or **digraphs**, consist of edges that have associated directions with them. Figure 2.1.1-5 shows a directed graph.

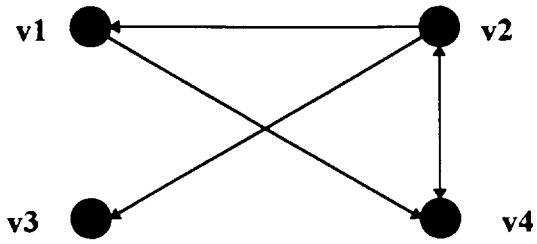



**Figure 2.1.1-5. A Directed Graph**

A **network** is a graph or digraph which assigns a value to each of the edges. A network resulting from a graph is called an undirected network, while a network resulting from a digraph is called a directed network. Figure 2.1.1-6 shows a directed network.

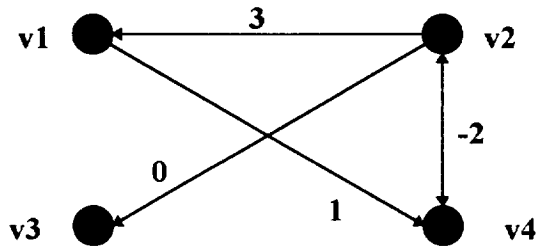



**Figure 2.1.1-6. A Directed Network**

For any vertex, v, the number of edges in the graph incident to v is known as the **degree** of v. The degrees of $v_1$, $v_2$, $v_3$, and $v_4$, from Figure 2.1.1-1 are 2, 3, 2, and 1, respectively. If p is the number of vertices and q is the number of edges for a graph, G, a well-known graph theory theorem is that:

$$\sum_{i=1}^{p} \deg(v_i) = 2q$$

A vertex is called even or odd if its degree is even or odd. Another well-known theorem is that: *Every graph contains an even number of odd vertices.*

A **cycle** is a path through the graph beginning and ending at the same vertex and passing through each vertex in the path only once. It need not include all vertices of the graph. For the graph of Figure 2.1.1-1, the following are examples of cycles: $v_1$, $v_2$, $v_4$, $v_1$ and $v_2$, $v_4$, $v_1$, $v_2$. A **circuit** is a relaxed form of cycle in which some vertices may be

repeated. A graph is connected if every two vertices of the graph is connected by some path or trail. A connected graph with no cycles is called a **tree**.

A graph containing a circuit which includes all vertices and edges of the graph is called **eulerian**. This brings us to another theorem: *A graph is eulerian if and only if it is connected with every vertex even.*

If a graph has a path (not a circuit) containing all vertices and edges, then it is called **traversable**, and the path is called an eulerian path, or eulerian trail. Another theorem states: *A graph is traversable if and only if it is connected with two odd vertices. Also, any eulerian trail of such a graph begins at one of the odd vertices and ends at the other odd vertex.* Note that while the graph of Figure 2.1.1-1 is not eulerian, it is traversable, with (one) eulerian path $v_3$, $v_2$, $v_4$, $v_1$. The two odd vertices are $v_2$ and $v_3$.

A graph is called **hamiltonian** if there is a cycle in it that contains every vertex. The graph of Figure 2.1.1-1 is not hamiltonian. The classical "salesman" problem consists of trying to find a hamiltonian graph through a set of cities. The "traveling salesman" problem (TSP) is the problem of finding the least expensive hamiltonian graph through a set of cities.

### 2.1.2 Some Advanced Graph Theory Concepts

The **complement**, C, of a graph, G, is a graph with the same vertex set as G, but with the property that two vertices of C are adjacent if and only if the same two vertices of G are not adjacent. The complement of the graph in Figure 2.1.1-1 is:


**Figure 2.1.2-1. The Complement of the Graph of Figure 2.1.1-1**

A **signed** graph is an undirected network whose functional values are +1 or -1. The value +1 is sometimes represented by a solid line, while the value -1 is sometimes represented by a dashed line. Figure 2.1.2-2 shows a signed graph. A signed graph is called **balanced** if its vertex set may be partitioned into two subsets (one of which might be empty) so that each edge joining two vertices in the same subset is positive, while each edge joining vertices in different subsets is negative. The graph of Figure 2.1.2-2 is balanced with subsets $\{v_1,v_2,v_4\}$ and $\{v_3\}$. Theorem: *A signed graph is balanced if and only if for every two vertices of it, all paths joining them have the same sign.* Balanced and unbalanced graphs have important implications to social systems.

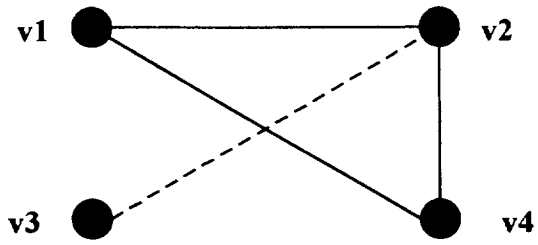



**Figure 2.1.2-2. A Signed Graph Which is Balanced**

A signed graph is **clusterable** if its vertex set can be partitioned into subsets (called clusters) so that every positive edge joins vertices within the same subset and every negative edge joins vertices in different subsets. Thus, a clusterable graph is a generalization of a balanced graph where the number of partitions (clusters) can exceed two. Theorem: *A signed graph is clusterable if and only if it contains no cycle with exactly one negative edge.* Thus, the graph of Figure 2.1.2-2 is both balanced and clusterable. A balanced graph is, by definition, clusterable.

A **planar** graph is a graph which can be drawn in a plane such that no two edges intersect except at a vertex. A planar graph already drawn in a plane so that no two edges intersect is called a **plane** graph. The connected pieces of the plane are called **regions** of the graph. Theorem: *Given a connected plane graph with p vertices, q edges, and r regions, $p - q + r = 2$.* Also: *Given a planar graph with p vertices and q edges, and p greater than or equal to three, then q is less than or equal to $3p - 6$.* Planar graphs are important in coloring problems. As is shown in Figure 2.1.2-3, the graph of Figure 2.1.1-1 is planar with two regions.

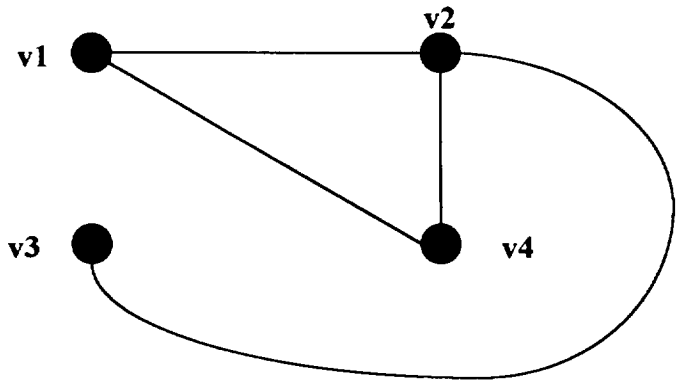



**Figure 2.1.2-3. Representing the Graph of Figure 2.1.1-1 as a Planar Graph**

A **coloring** of a graph is the assignment of a different color to each vertex such that adjacent vertices are assigned different colors. For an n-coloring of the graph in Figure 2.1.2-3, what is the minimum value of n? We see that n must be at least three. This is called the graph's **chromatic number**. An example 3-coloring for this graph is shown in Figure 2.1.2-4. A proper 9-coloring of the MARS graph is shown in Figure 2.1.2-5. An important theorem states: *The chromatic number of a graph is less than or equal to one plus the maximum degree among its vertices.* The Four Color Theorem states: *The chromatic number for any planar graph is less than or equal to four.*

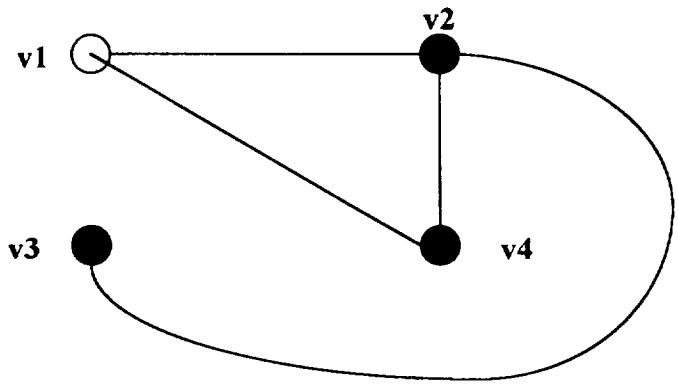



**Figure 2.1.2-4. A 3-Coloring of the Graph of Figure 2.1.1-1**

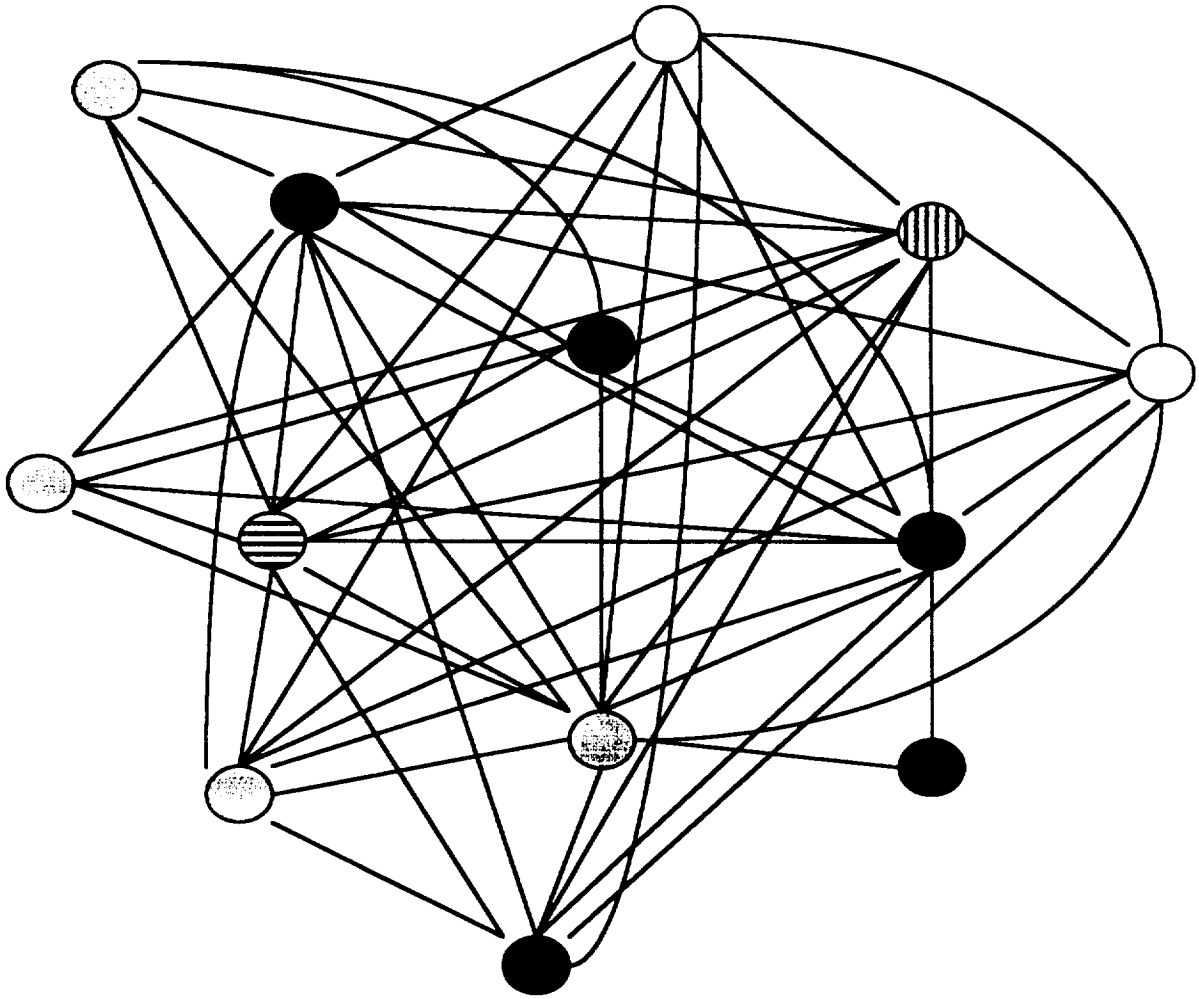


**Figure 2.1.2-5. A Proper 9-Coloring of MARS at DEL**

Numerous models were developed to evaluate the correlative effect of roughly 15 different graph metrics discussed in this section. Table 2.1.2-1 provides the results of the major regression experiments conducted. This table provides the results following a significant filtering of the graph metrics. The data used for filtering and regressions is provided in Appendix A. These experiments were conducted following an exhaustive comparison of the inter-correlation of a larger set of graph metrics. It is seen that, in addition to the initial cost estimate, both minimum degree and chromatic number are statistically significant predictors of cost growth. For actual duration prediction, no graph metrics were found to be significant predictors.

**Table 2.1.2-1. Regression Statistics for Graph Metric Models**

| Depend. Variable | Cost Est. | Dura-tion Est. | Min. Deg. ($\delta$) | Chrom. No. ($\chi$) | Connec -tivity (K) | Corre-lation (R) | Signifi-cance Level ($\alpha$) |
|---|---|---|---|---|---|---|---|
| Cost Growth | 5.7E-07 | | 9.1E-03 | 4.7E-02 | | 0.934 | 4.4E-06 |
| Cost Growth | 1.4E-06 | | 7.9E-02 | | | 0.909 | 4.9E-06 |
| Cost Growth | 9.5E-06 | | | 0.945 | | 0.884 | 2.5E-05 |
| Cost Growth | 2.6E-06 | | | | | 0.883 | 2.6E-06 |
| Cost Growth | | | 0.965 | 0.433 | | 0.262 | 0.608 |
| Cost Growth | 1.7E-06 | | | 0.245 | 2.9E-02 | 0.922 | 1.2E-05 |
| Cost Growth | 8.3E-07 | | | | 5.5E-02 | 0.913 | 3.6E-06 |
| Actual Duration | 3.2E-02 | 4.6E-02 | | | | 0.858 | 9.0E-05 |
| Actual Duration | 2.1E-02 | 6.5E-02 | 0.22 | 0.31 | | 0.877 | 8.5E-04 |
| Actual Duration | 2.1E-04 | | 0.199 | 0.409 | | 0.830 | 1.3E-03 |
| Actual Duration | 1.1E-04 | | 0.304 | | | 0.820 | 4.0E-04 |

**Significant Finding:**

An exhaustive investigation of approximately 15 different graph metrics was conducted. This investigation included several hundred different nonlinear regressions and neural network examinations. The two graph metrics which survived the cut were minimum degree and chromatic number. **Ultimately, a variable representing chromatic number was added to the COMPRÉ model.**

## 2.2 *Task 2: Organizational Design Using Covariance Structures*

This task involves the investigation of the use of statistical covariance concepts and applications to the design and analysis of organizations. This investigation includes concepts resulting from Task 1, particularly those involving balanced vs. unbalanced graphs.

### 2.2.1 Organizational Covariance Model and Input Requirements

The baseline **organizational covariance** model is isomorphic to the programmatic covariance model found in Complex Organizational Metric for Programmatic Risk Environments (COMPRÉ):

$$\lambda(t) = \sum_{i=1}^{n} w_i(t) E[r_i(t)]$$

$$\sigma(t) = [\sum_{i=1}^{n} w_i(t) \sum_{j=1}^{n} w_j(t) C_{ij}(t)]^{1/2}$$

$$C_{ij}(t) = \frac{\int_0^t \{[r_i(t) - E[r_i(t)]][r_j(t) - E[r_j(t)]]\} dt}{\int_0^t dt}$$

$$E[r_i(t)] = \frac{1}{t} \int_0^t r_i(t) dt$$

$$\sum_{i=1}^{n} w_i(t) = 1, w_i(t) \geq 0$$

where

$\lambda(t)$ is a (time-dependent) organizational development maturity function, representing a rate of payoff for investing in a certain level of organizational development and technical capability,

n is the number of personnel investments made,

$w_i$ (t) is the relative investment weight for personnel i,

$E[r_i$ (t)] is the expected return-on-investment for personnel i over the organizational development schedule represented by t,

$r_i$ (t) is the return-on-investment for personnel i over the schedule duration, t, and is dependent on personnel education, experience, and skill levels,

$\sigma(t)$ represents total organizational risk, and

$C_{ij}$ (t) is the covariance between personnel i and j over the organizational development duration.

The organizational development maturity function, $\lambda$, accounts for personnel technical capability, schedule, and budgetary distribution decisions, as well as organizational size. The organizational risk function, $\sigma$, captures the same issues, as well as architectural design of the organization. It is anticipated that the ratio of these two functions may provide a useful organizational complexity measure for decision-making and control purposes.

### 2.2.2 Integration of Organizational Covariance With Graph Theory Concepts

The organizational **architecture** is an important determinant in the evaluation of the organizational covariance structure. This architecture also is very important from a graph theory perspective. Figure 2.2.2-1 show a graph theory representation of a four person organization. The solid lines between personnel indicate a positive relationship, the dashed lines a negative relationship, and the lack of any line represents a neutral relationship. In this particular example, the graph is balanced and can be partitioned into two groups of personnel each of which has only positive relationships within and negative relationships between. The groups are {1, 2, 4} and {3}. Some organizational research has indicated the importance of implementing and sustaining a balanced organization.

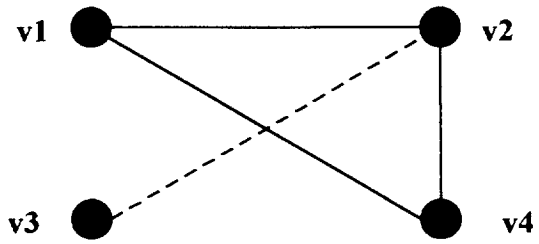



**Figure 2.2.2-1. Graph Theory Representation of One Type of Balanced Organization Consisting of Four Personnel**

**Significant Finding:**

**The chromatic number variable, in combination with the organizational interface factors already present in COMPRÉ appears to satisfactorily capture organizational complexity through covariance.**

## *2.3 Task 3: Neural Network Approach to Programmatic Hierarchies*

This task involves the investigation of the use of neural network concepts and applications to the design and evaluation of programmatic hierarchies. There is an attempt to treat a programmatic hierarchy as a neural network. The task includes the selection of learning, training, transfer functions, and associated parameters. Information

concerning "optimal" design of neural networks is considered for programmatic design. Task 1 results are integrated with the findings of this task.

### 2.3.1 Network Geometry, Learning, and Propagation

**Artificial neural networks** were developed based on the concept that the human brain is one of the best pattern recognition tools available. Thus, artificial neural networks are an attempt to simulate the processes that biological neurons perform in the human brain. The representation for an artificial neuron is characterized by a number, n, of inputs, each weighted and leading to a sum, which is then activated by a **transfer function**, producing an output on a path leading to other neurons or the final output. Figure 2.3.1-1 shows a schematic for an artificial neuron. A common activation or transfer function is the **sigmoidal** function. The sum of weighted parts and logistic (most widely used sigmoidal) activation functions are denoted by:

$$I = \sum_{i=1}^{n} w_i x_i$$

$$y = \phi(I)$$

$$\phi(I) = \frac{1}{1 + e^{-\alpha I}}$$

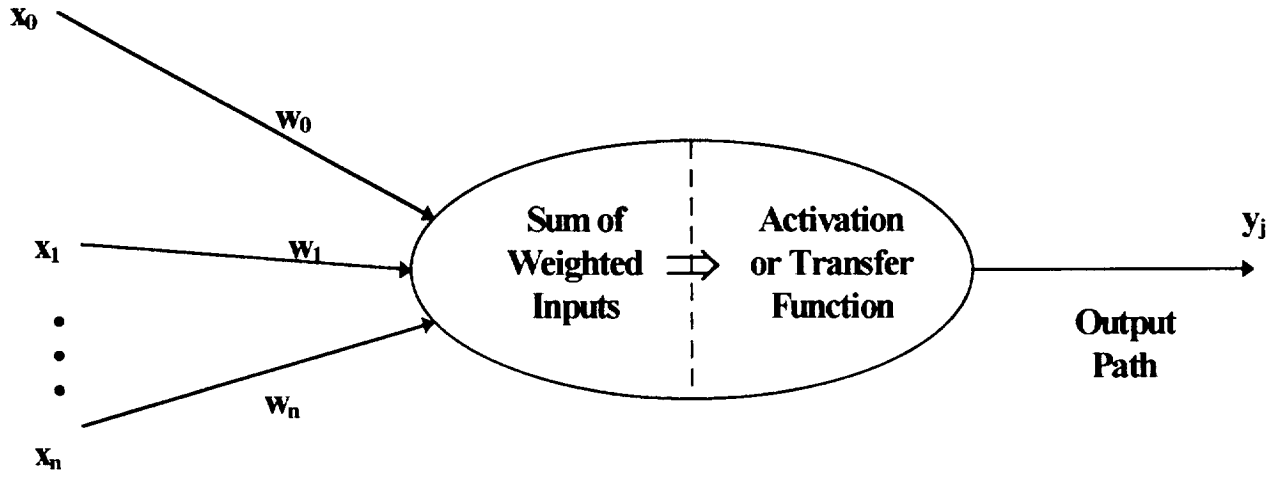



**Figure 2.3.1-1. An Artificial Neuron**

A neural network is composed of a number of layers, each consisting of a certain number of artificial neurons. The first layer of neurons is called the **input layer**, the next layers are called middle or **hidden layers**, and the final layer is called the **output layer**. Figure 2.3.1-2 shows a simple neural network, called a **feedforward network**.

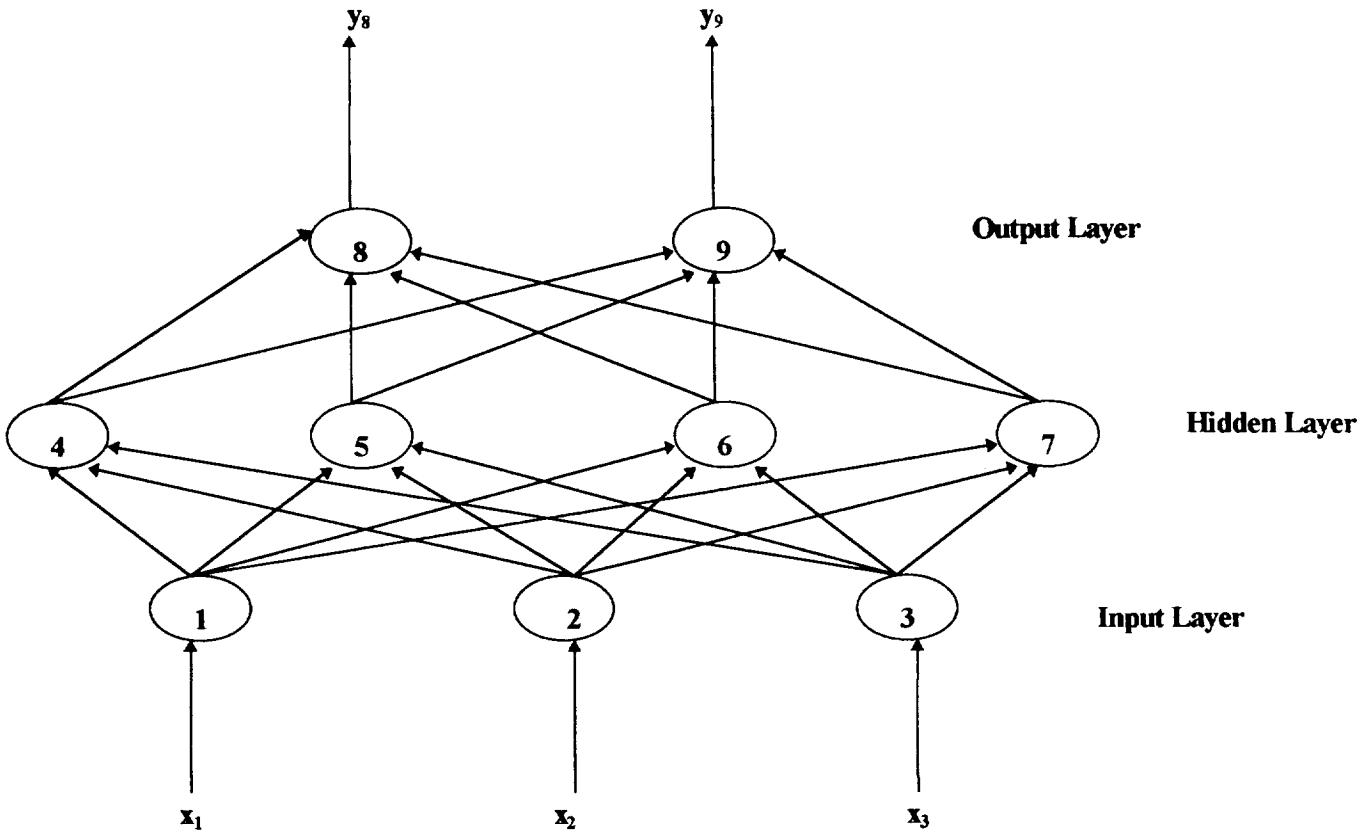



**Figure 2.3.1-2. A Simple Feedforward Neural Network**

### 2.3.2 Network Control and Strategy

Neural network operations are controlled by the type of **learning** (supervised vs. unsupervised), **training** (e.g., backpropagation), and the network **design** (number of layers, number of neurons in each layer). Training algorithms are further characterized by tuning parameters such as momentum, transfer function parameters, learning constants, etc.

Approximately 12 different neural network model configurations were experimented with in this task. Each of these were trained over several hundred to several thousand combinations of network weights. Some networks performed well, while others did not. The networks that did not perform well resisted all attempts at training, indicating a probable lack of causality. Of course, even those networks that did train well do not guarantee causality, only make it more likely. However, some of the networks with good fit were also discarded, due to an unhealthy overdependence on a single variable or an indication that the network was memorizing vs. learning the database. The final network chosen is balanced (in terms of predictor variables), does not appear to be memorizing the database, and has an excellent fit. This neural network is discussed in Section 2.5 of the report.

**Significant Finding:**

Several hundred neural network configurations with different variable combinations were investigated during this study. Neural networks were found to significantly outperform sophisticated nonlinear regression methods, on a consistent basis. **The final neural network implemented in COMPRÉ is shown in Section 2.5 of this report.**

## *2.4 Task 4: Use of Genetic Algorithms to Evolve Programs*

This task involves the investigation of the use of genetic algorithm (GA) concepts and applications to the evolution of programmatic hierarchies. This investigation includes the selection of appropriate GA formulations and parameters. This task focuses on the ability or inability of GA's to replace or complement the use of basis functions in evolving programmatic hierarchies. The task utilizes data from previous efforts, as well as complementary on-going efforts.

### 2.4.1 Genetic Algorithm Setup: Population, Fitness, Mating, and Mutation Parameters

**Genetic algorithms** (GA's) were developed from observations of biological processes and species population evolution processes. The evolution processes modeled include mating, mutation, and survival of the fittest. Typical GA parameters of interest include: population size, chromosome length, crossover rate, mutation rate, generation gap, elitism, and diversity. Also important in any GA are typical algorithm parameters, such as initial conditions, measures of performance, and termination criteria.

### 2.4.2 Programmatic Evolution Using Genetic Algorithms

Genetic algorithms may be useful in evolving programmatic characteristics over a duration. Specifically, GA's might be applicable to the prediction of **excessive optimism** and/or pessimism concerning programmatic features such as budget, schedule, and technological readiness. GA's might also be useful in determining the **interactions** among these three programmatic elements. For instance, how much budget does it take to increase technological readiness from one level to another over a stated period of time? Questions such as these may be answered using a combination of GA's, neural nets, and appropriate historical databases.

Figure 2.4.2-1 shows how estimates of the program duration evolve over milestones for some programs in the database. In nearly every case shown, duration estimates grew as the program evolved. In certain cases, such as OTA, SSM, and HST, initial duration estimates were wildly optimistic.

Figure 2.4.2-2 shows how estimates of the overall program TRL evolve over milestones for some programs in the database. In nearly every case shown, TRL estimates shrank as the program evolved, indicating that the initially estimated technology readiness was higher than the actual readiness. In certain cases, such as OTA, SSM, and HST, initial TRL estimates were wildly optimistic.

Figure 2.4.2-3 shows how estimates of the subsystem TRL's evolve over milestones for GRO. In nearly every case shown, TRL estimates shrank as GRO evolved, indicating

that the initially estimated technology readiness was higher than the actual readiness. In the case of thermal control, the initial TRL estimate was wildly optimistic.

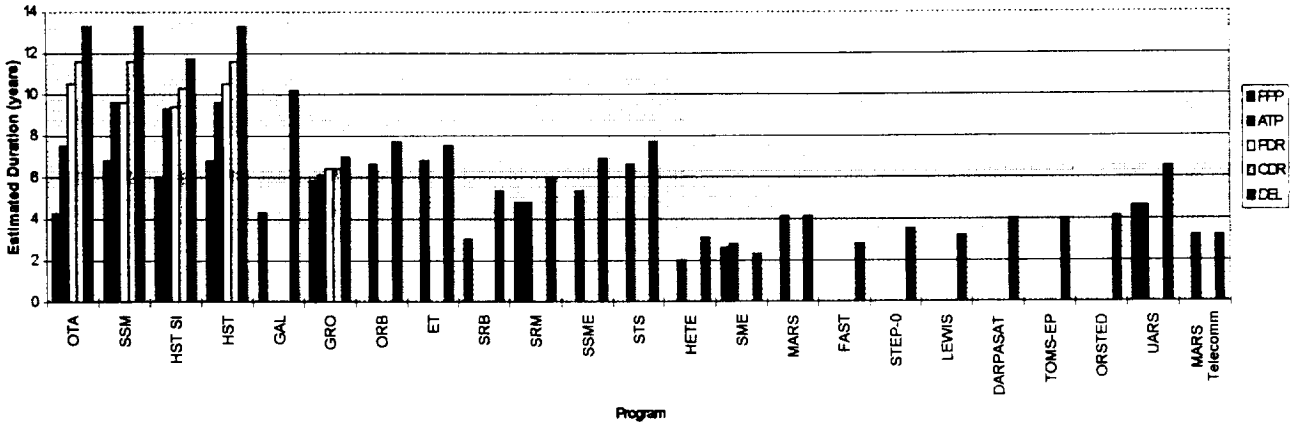



**Figure 2.4.2-1. Evolution of Duration Estimates for Some Programs**

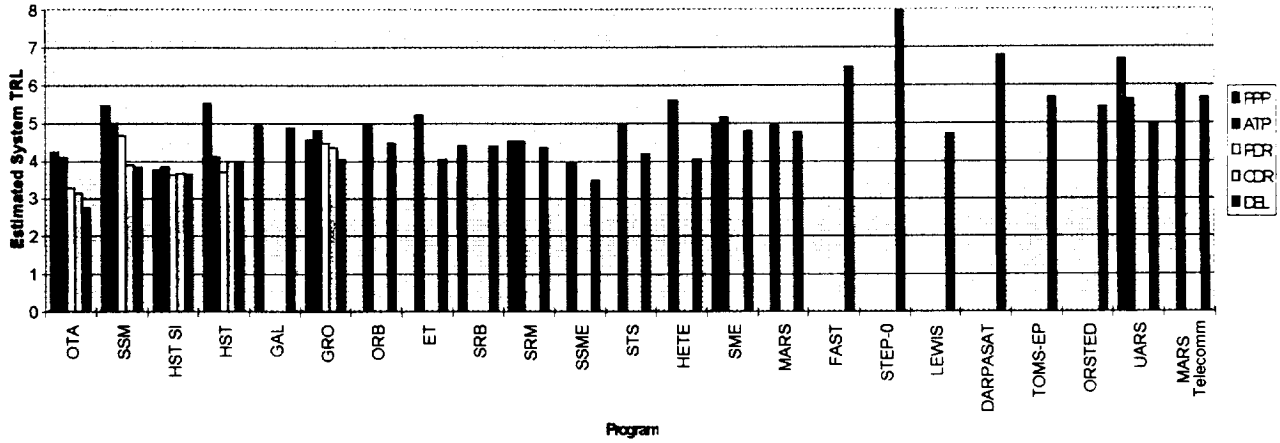



**Figure 2.4.2-2. Evolution of Overall TRL Estimates for Some Programs**

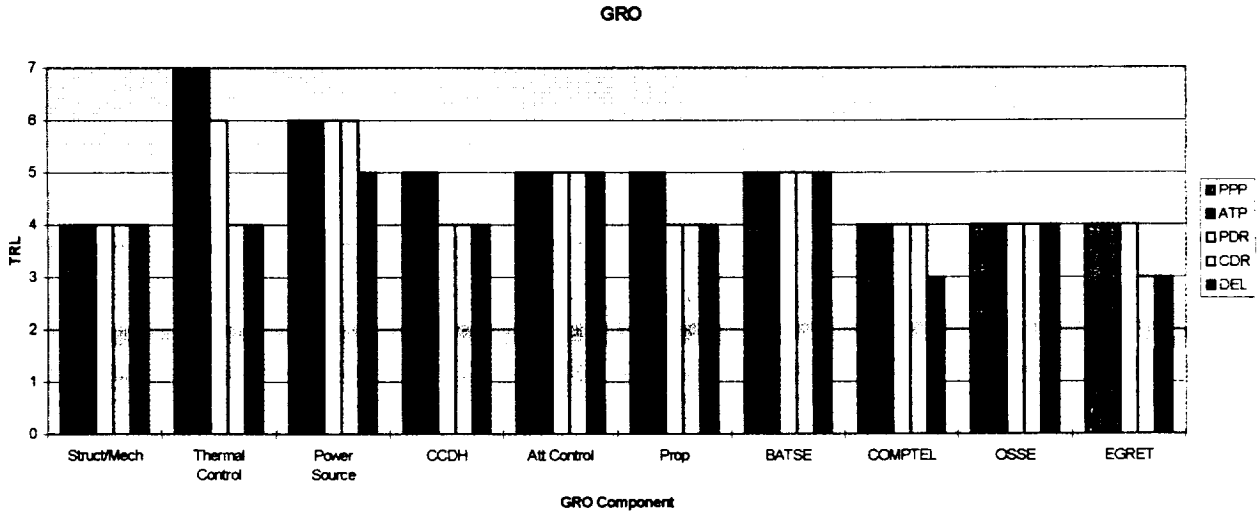



**Figure 2.4.2-3. Evolution of TRL Estimates for GRO Subsystems**

The preceding graphs make it clear that there is a bias present in estimating TRL and schedule duration at the beginning of programs. Furthermore, this bias is firmly in the direction of excessive optimism regarding the programmatic requirements. To what extent does this bias reflect itself in the experience cost growth of the program?

Figure 2.4.2-4 shows the trend in the number of cost growth "events" with increasing TRL. A cost growth event is defined as a 10% increase in subsystem or component cost for a given program. Thus, a subsystem with a cost growth of 53% is associated with 5.3 cost growth events. All components and subsystems for the entire database were evaluated and categorized by TRL. This trend follows intuition, since more risky technology (lower TRL components) would be expected to have a greater number of cost growth events, and thus, higher cost growth than more mature technology subsystems. The regressed line in the chart is derived using an exponential trend. This regression serves as the basis for establishing expected programmatic cost growth in COMPRÉ.

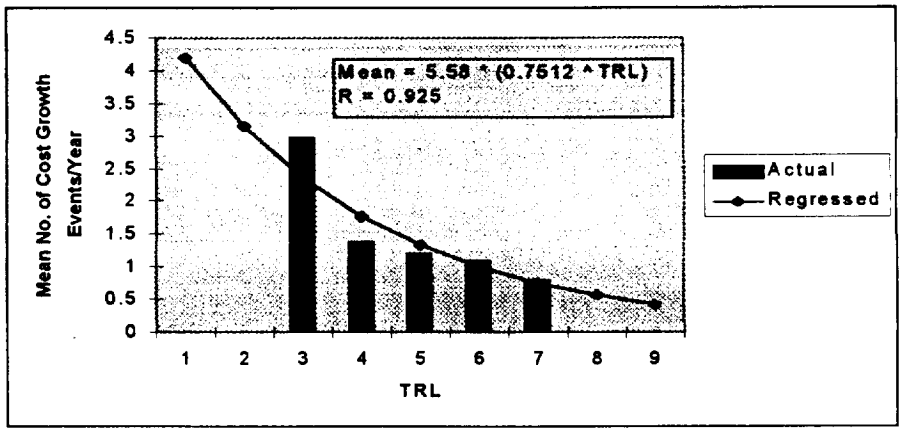



**Figure 2.4.2-4. Mean No. of CG Events vs. TRL**

Figure 2.4.2-5 shows the trend in the standard deviation of cost growth "events" with increasing TRL. All components and subsystems for the entire database were evaluated and categorized by TRL. This trend also follows intuition, since more risky technology (lower TRL components) would be expected to have a greater variance in the number of cost growth events, and thus, higher cost growth than more mature technology subsystems. The regressed line in the chart is derived using an exponential trend. This regression serves in the covariance matrix of COMPRÉ.

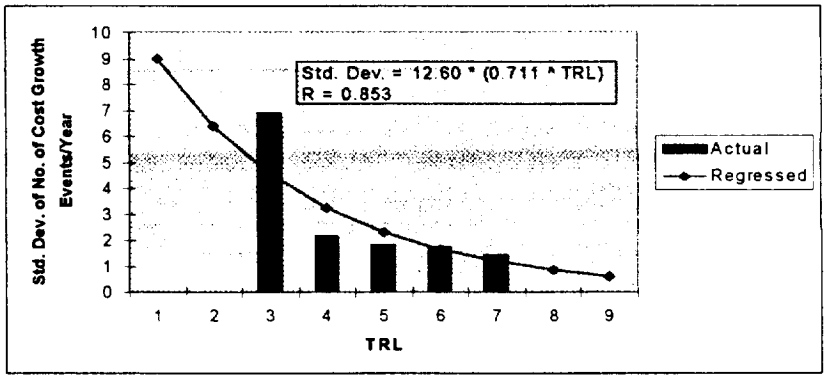



**Figure 2.4.2-5. Standard Deviation of No. of CG Events vs. TRL**

**Significant Finding:**

**The cost growth regressions were included in the COMPRÉ model. These regressions represent a novel, intuitive, and data-supported methodology for understanding the stability (through cost growth) of a program. Although these regressions naturally evolve from the optimism bias, the excessive optimism in estimation programmatic duration and TRL is not currently compensated for in COMPRÉ.**

## *2.5 Integration of Research Results into COMPRÉ Configuration*

This section of the report provides the details of how the detailed research from previous sections was used in arriving at the current COMPRÉ model.

Table 2.5-1 provides the connection weights for the artificial neural network model in COMPRÉ. A value in this matrix represents the connection weight between the neuron corresponding to the row and the neuron associated with the column of the matrix.

Figure 2.5-1 shows the fit of the neural network to the actual database for cost growth (%). This model has an average error of 7% per program. The fit is reasonably good for all programs except PV Large and DE. Some of the smaller cost growth programs are also underfitted. Nevertheless, the neural network fits the cost growth data significantly better than regression.

Figure 2.5-2 provides the current COMPRÉ architecture. Information on cost, schedule duration, technology readiness levels, organizational interface factors, and component interfaces are used to create an architecture diagram like that shown in the upper right portion of the figure. A covariance analysis, which serves as a micro-assessment, results in risk measures that account for how the program is put together, the technological risk, and the schedule and cost risk components. These risk parameters are then fed into a neural network, along with global features of the program such as overall cost, duration, and chromatic number.

The neural network (lower left portion of the figure) performs a macro-assessment by assimilating these global features into a final measure of system complexity, namely, cost growth. Finally, an s-curve (lower right) which provides the probability that the actual cost growth is less than a given target value is produced. Program managers may then use such a curve to state various confidences in achieving budget, or to redesign the program to produce a more satisfactory cost growth profile.

The s-curve is arrived at by using the expected cost growth output from the neural net model combined with the average error of the neural net itself to arrive at a normal distribution for cost growth. The mean of this normal distribution is the expected cost growth (from the neural net) of the program of interest. The standard deviation of the normal distribution is the average error (nominally 7%) of the model. This is not the same standard deviation as that resulting from the basis functions. However, it is influenced through training by the basis function standard deviation, as well as the other parameters utilized in the neural network. The cumulative probability resulting from

application of the normal distribution is the point on the s-curve corresponding to the target value. Thus,

$$P(CG < TCG) = NORMS(\frac{TCG - \mu}{\sigma})$$

where,

CG = actual (unknown) programmatic cost growth (%),
NORMS = standard normal distribution,
P = probability,
TCG = target cost growth (%),
$\mu$ = expected cost growth (%) from neural net,
$\sigma$ = neural net average error (%) in predicting cost growth.

Figure 2.5-3 shows the COMPRÉ cost growth results for the Materials Science Research Rack No. 1 (MSRR-1). The 95% confidence point corresponds to a cost growth of 12%, which is shown in comparison with the other programs in the database. From a comparative standpoint, this would appear to be a very stable program.

Figure 2.5-4 shows the COMPRÉ schedule growth results for the Materials Science Research Rack No. 1 (MSRR-1). The 95% confidence point corresponds to a schedule growth of 10%, which is shown in comparison with the other programs in the database. Again, from a comparative standpoint, this would appear to be a very stable program.

Figure 2.5-5 shows a comparative feature analysis for 4 other programs with similar global characteristics as MSRR-1. The resulting cost growth values range from 0% (MARS) to 42% (ERBS). This is another sanity check on the restrained cost growth values achieved by COMPRÉ.

Figure 2.5-6 provides a comparison of the cost-risk for the subsystems of MSRR-1. Cost-risk is the compound value arising from the investment in a subsystem and its relative risk produced by the micro-assessment of covariance analysis. Here, it is seen that the ESA Module holds an order of magnitude greater cost-risk than any other MSRR-1 subsystem. This seems to track well with intuition.

**Table 2.5-1. COMPRÉ Neural Net Connection Weights**

| Hidden Layer or Output Node | Cost Variable | Duration Variable | COMPRÉ Risk Variable (sigma) | COMPRÉ Return Variable (lambda) | Chromatic Number Variable (chi) | Bias Node | Cost Growth (output node) |
|---|---|---|---|---|---|---|---|
| 1 | 3.16 | 0.88 | -81.1 | 6.33 | -3.65 | 1.26 | 53.9 |
| 2 | 7.44 | -1.75 | 102.8 | 9.83 | -0.27 | 0.30 | -43.1 |
| 3 | 0.64 | -14.3 | -9.5 | -27.3 | 11.8 | 4.08 | -5.49 |
| 4 | -3.42 | 3.05 | -2.3 | -4.07 | -9.30 | 1.22 | -14.3 |
| Cost Growth (output) | N/A | N/A | N/A | N/A | N/A | -7.58 | N/A |

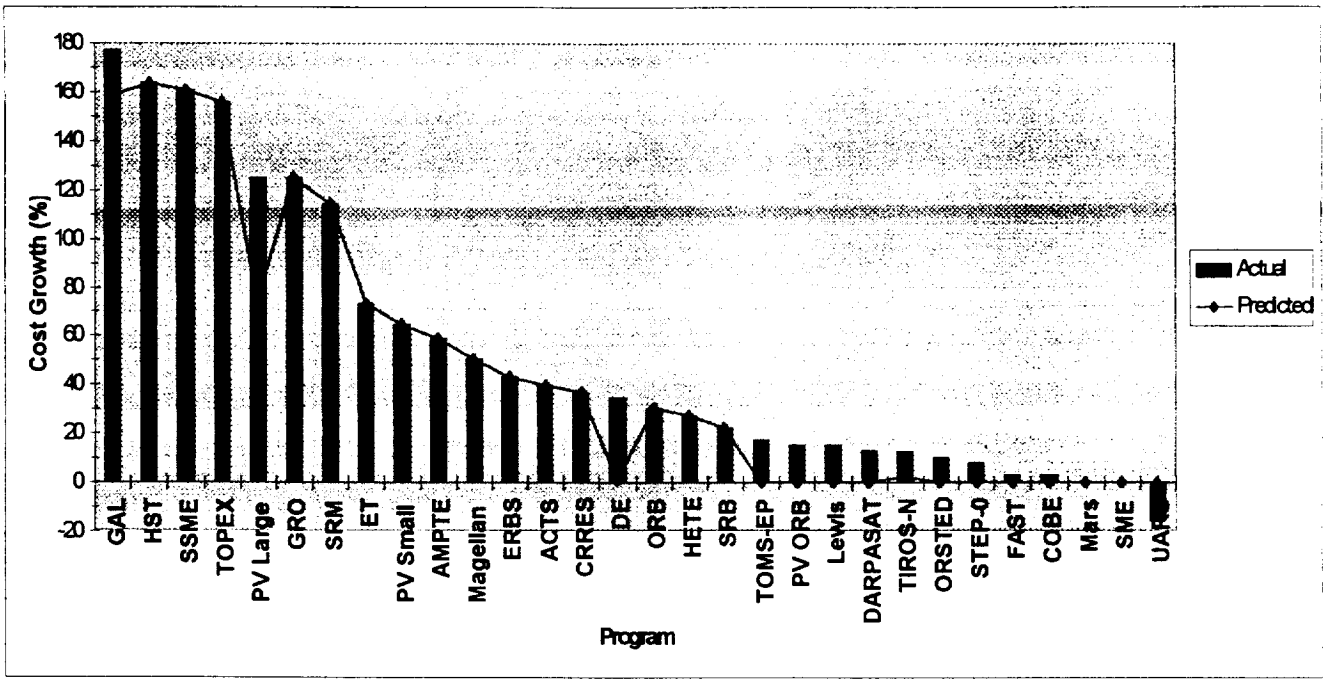



**Figure 2.5-1. Neural Network Fit to Actual Database**

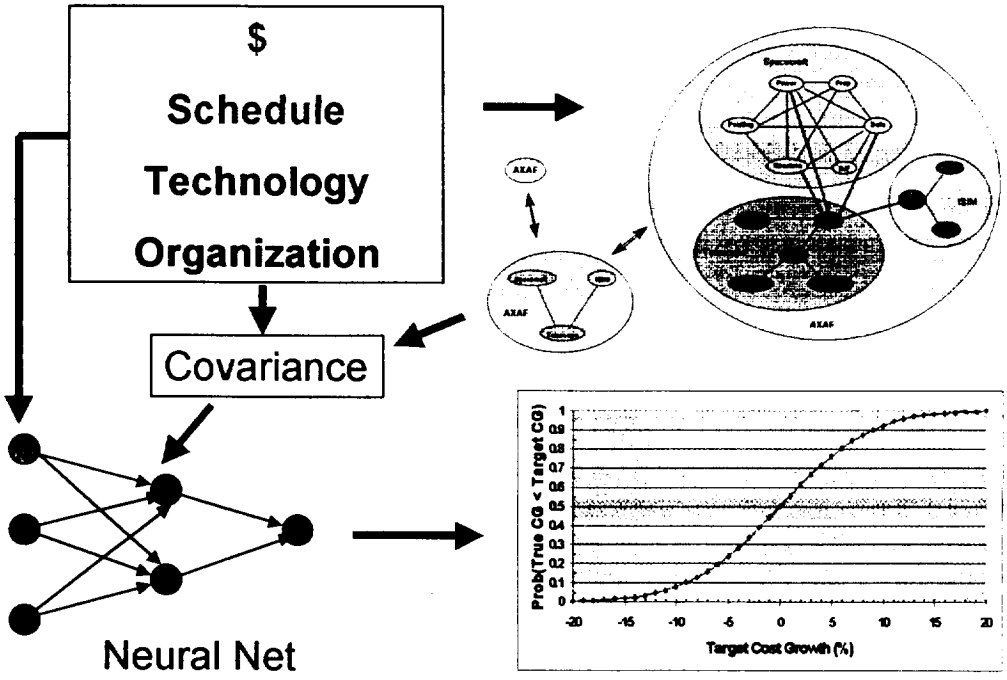



**Figure 2.5-2. Current COMPRÉ Model Architecture**

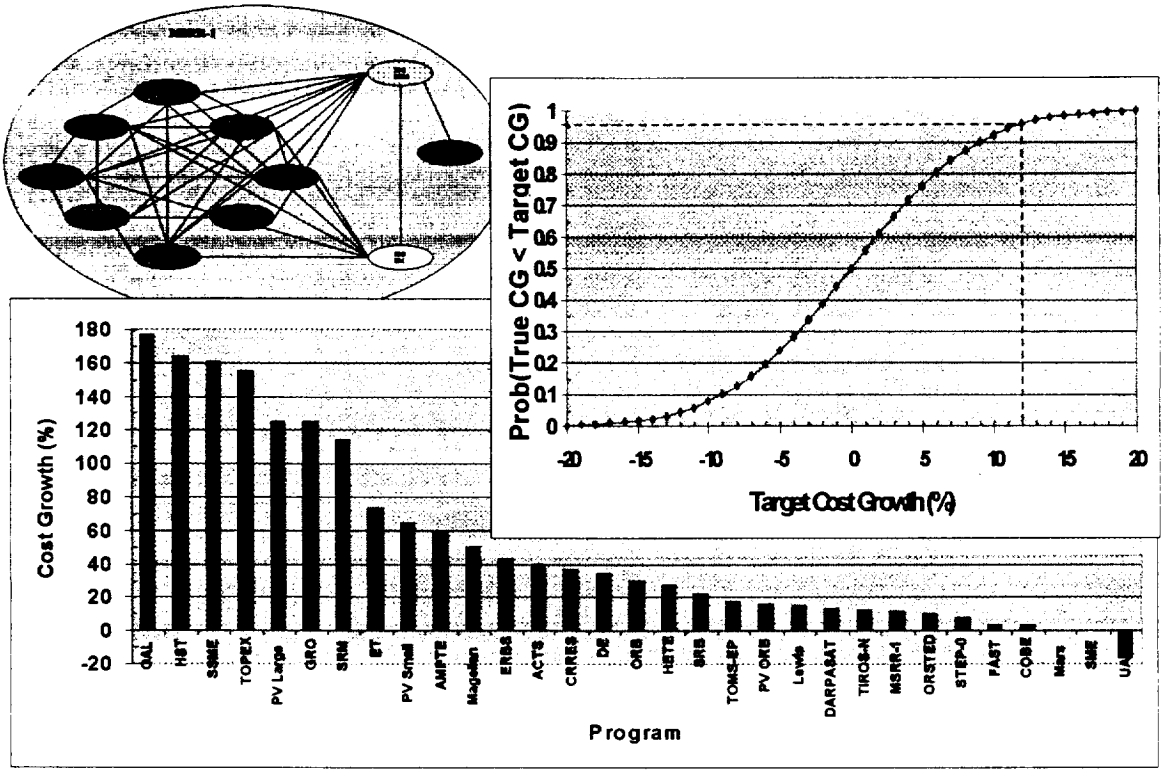



**Figure 2.5-3. MSRR-1 Cost Growth Analysis Using COMPRÉ**

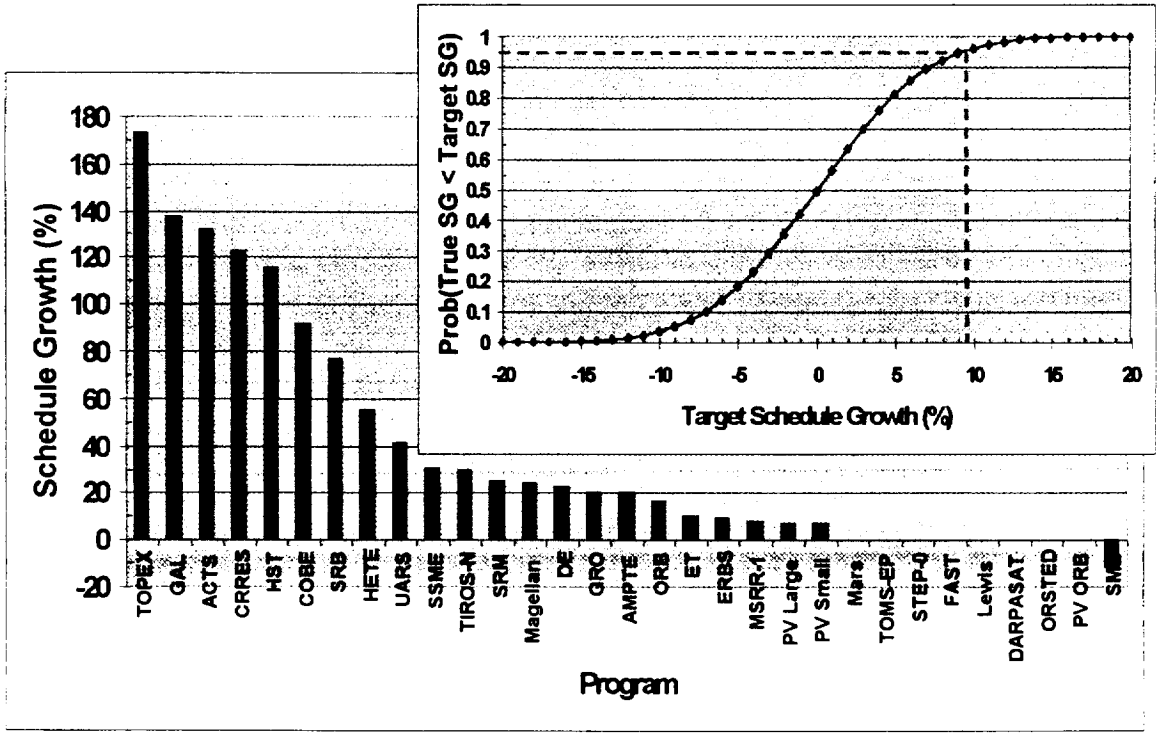



**Figure 2.5-4. MSRR-1 Schedule Growth Analysis Using COMPRÉ**

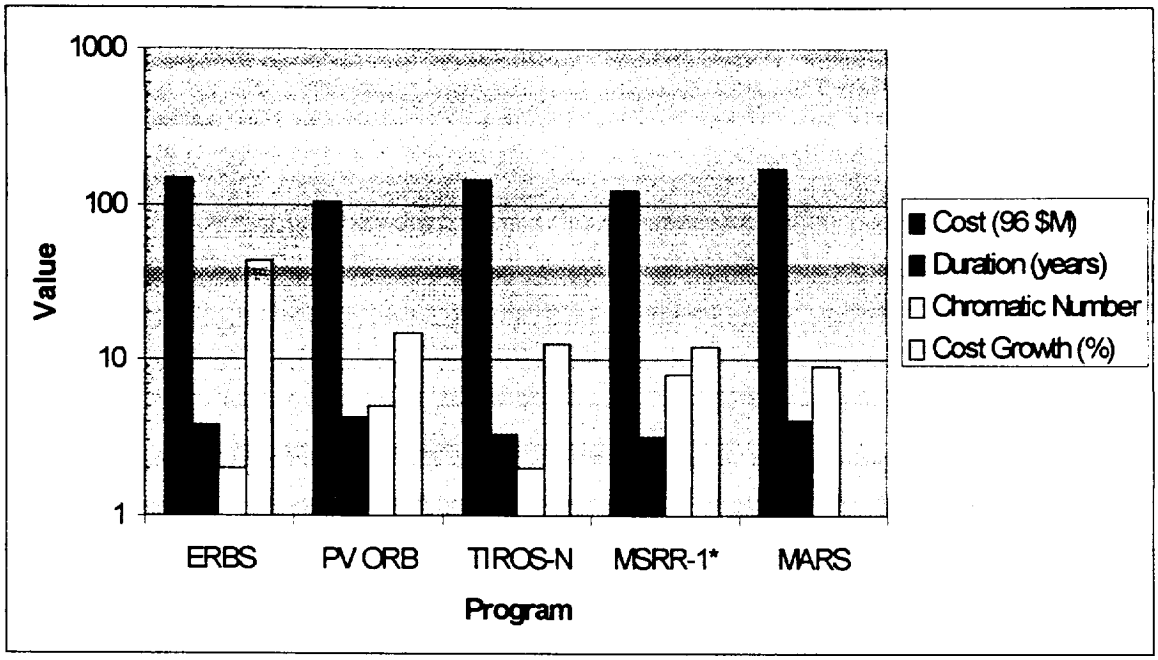



**Figure 2.5-5. MSRR-1 Comparative Feature Analysis**

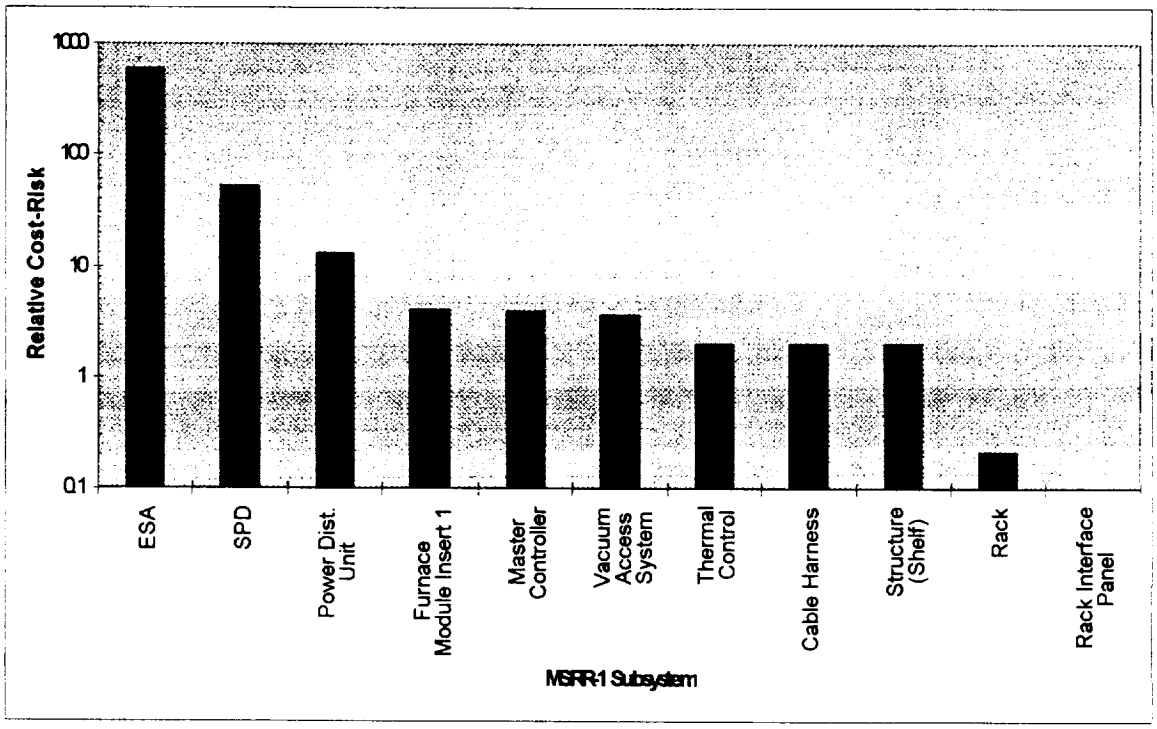



**Figure 2.5-6. Relative Cost-Risk Values for MSRR-1 Subsystems**

Figure 2.5-7 shows the significance of chromatic number in the COMPRÉ model. Recall that chromatic number is inherent in any programmatic architecture. Thus, this "connectivity" measure is important in the covariance calculation, which is highly

dependent on interfaces among subsystems. Furthermore, the chromatic number is used as a global input to the neural network. In this figure, it is clear that the effect of chromatic number is highly sensitive to the programmatic cost. For instance, up to a cost of about 4 times the estimated cost of MSRR-1, chromatic number has a negligible effect on predicted cost growth. However, above 7 times the estimated cost the effect is striking. Moreover, this effect varies with the chromatic number itself. For chromatic numbers in the ranges of 1-2 or 7-8, little effect is seen. For numbers in the range of 3-6, it is very powerful. Since a chromatic number of 1-2 indicates extremely low connectivity, and is, thus, very impractical for functioning systems, this chart would indicate that for MSRR-1, more connectivity is better. This result is program dependent, however, and should be carefully examined for each individual circumstance. There is no denying, however, that chromatic number can, in certain circumstances, have a striking effect on cost growth prediction. Finally, this graph shows the subtle interaction of the local covariance analysis with the global neural network synthesis.

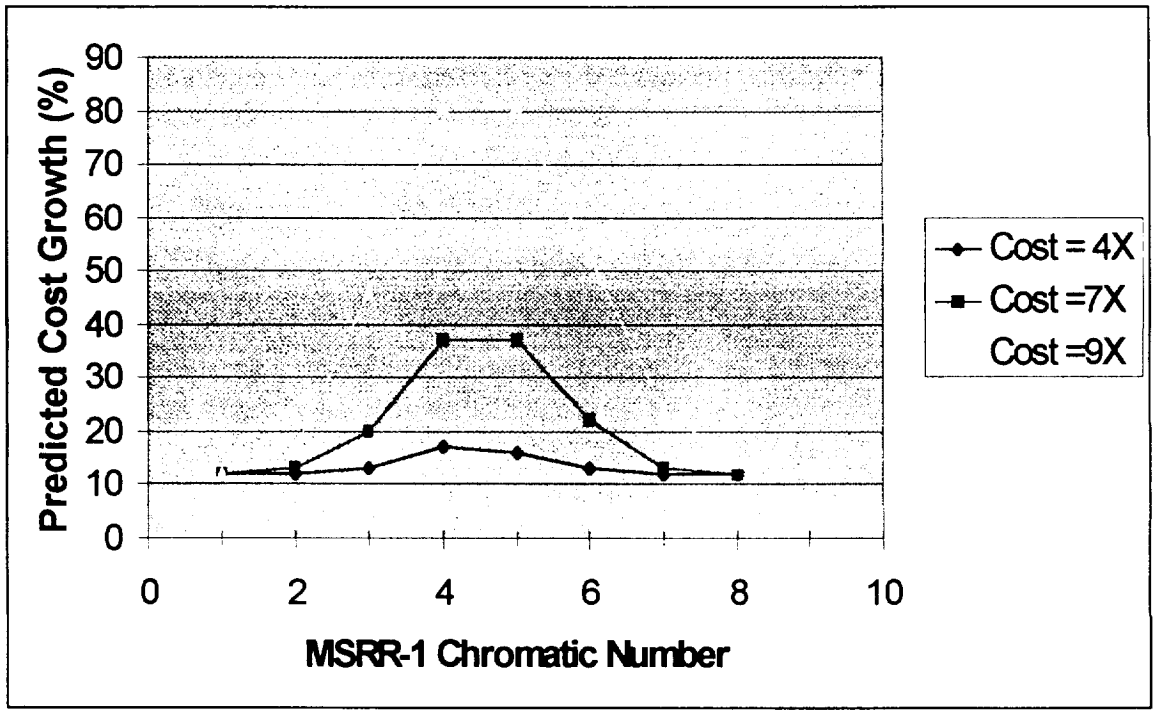



**Figure 2.5-7. Significance of Chromatic Number**

## 3.0 Conclusions

1. Chromatic number (a graph metric) is a statistically significant measure of programmatic complexity, and, therefore, is included as a variable in the current COMPRÉ model. Chromatic number also appears to work well in capturing organizational complexity through covariance.

2. There are tradeoffs among user-friendliness, model realism, and input reliability for various levels of architectural decomposition. Level III COMPRÉ decompositions appear to be a good balance of these factors.

3. Artificial neural networks significantly outperform sophisticated nonlinear regressions in modeling system complexity for NASA programs. An artificial neural network performs a macro-assessment on global variable inputs in COMPRÉ.

4. Component-level cost growth forecasting adequately addresses program evolution, and is employed in COMPRÉ as a generator for cost growth risk.

5. COMPRÉ includes a micro-assessment of programmatic complexity through interaction and a macro-assessment of global parameters through synthesis. These two types of assessments simultaneously complement and compete with each other to determine the metric of system complexity.

## 4.0 Bibliography


Bunde, A., and Havlin, S., Fractals in Science, Springer-Verlag, 1994.

Cameron, K.S., and Whetten, D.A., Organizational Effectiveness: A Comparison of Multiple Models, Academic Press, 1983.

Chartrand, G., Introductory Graph Theory, Dover Publications, Inc., 1977.

Crouch, E.A.C., and Wilson, R., Risk/Benefit Analysis, Ballinger Publishing Co., 1982.

Draper, N., and Smith, H., Applied Regression Analysis, John Wiley & Sons, 1981.

Drouin, M., Abou-Kandil, H., and Mariton, M., Control of Complex Systems: Methodology and Technology, Plenum Press, 1991.

Efron, B., "Bootstrap Methods: Another Look at the Jackknife," *The Annals of Statistics*, Vol. 7, No. 1, 1979.

Freudenberg, W.R., and Rursch, J.A., "The Risks of 'Putting the Numbers in Context': A Cautionary Tale," *Risk Analysis*, Vol. 14, No. 6, 1994.

Flood, R.L., and Carson, E.R., Dealing With Complexity: An Introduction to the Theory and Application of Systems Science, Plenum Press, 1988.

Gramb, T., et al., Non-Standard Computation, Wiley-VCH, 1998.

Hastings, H.M, and Sugihara, G., Fractals: A User's Guide for the Natural Sciences, Oxford University Press, 1995.

Harvard School of Public Health, "Analyzing Risk: Science, Assessment, and Management," September, 1994.

Hertz, J., Krogh, A., and Palmer, R.G., Introduction to the Theory of Neural Computation, Addison-Wesley, 1991.

Hudak, D.G., "Adjusting Triangular Distributions for Judgmental Bias," *Risk Analysis*, Vol. 14, No. 6, 1994.

Juran, J.M., Juran's Quality Control Handbook, McGraw-Hill, Inc., 1988.

Klir, G.J., Architecture of Systems Problem Solving, Plenum Press, 1985.

Kronsjo, L., Computational Complexity of Sequential and Parallel Algorithms, John Wiley & Sons, 1985.

Lee, S.M.S., and Young, G.A., "Sequential Iterated Bootstrap Confidence Intervals," *Journal of the Royal Statistical Society*, Vol. 58, No. 1, 1996.

Lloyd, S., "Complexity Simplified," *Scientific American*, May 1996.

Mog, R.A., "Systems Engineering Metrics: Organizational Complexity and Product Quality Modeling," H-27289D, Final Report, OR Applications, September 1997.

Mog, R.A., "Risk Assessment Methodologies for Spacecraft in Particulate Environments: Enhancements Through Operations Research," OR Applications Final Report, September 1996.

Mog, R. A., "A Leveraged Frontier Model for Real Estate Investment Decisions," OR Applications, September 1996.

Mog, R.A., "Complexity and Meta$^2$modeling Innovations for Electronic Manufacturing Systems Decision and Modeling Support," OR Applications, September, 1996.

Mog, R. A., "Portfolio Diversification in a Real Estate Investment Environment," OR Applications, March 1996.

Mog, R.A., "The Use of the Poisson Process in Environmental Risk Assessment," Internal Memo, July 1994.

Nemhauser, G.L., and Kan, A.H.G., Handbooks in Operations Research and Management Science: Computing, Vol. 3, North-Holland, 1992.

Nemhauser, G.L., Kan, A.H.G., and Todd, M.J., Handbooks in Operations Research and Management Science: Optimization, Vol. 1, North-Holland, 1989.

Neuts, M.F., Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach, Dover Publications, Inc., 1981.

Nicolis, J.S., Chaos and Information Processing: A Heuristic Outline, World Scientific, 1991.

Papadimitriou, C.H., Computational Complexity, Addison-Wesley, 1994.

Peliti, L., and Vulpiani, A., Measures of Complexity, Springer-Verlag, 1988.

Sedgewick, R., Algorithms, Addison-Wesley, 1988.

Society for Risk Analysis, *Risk Analysis: Uncertainty Analysis*, Vol. 14, No. 4, August 1994.

Streufert, S., and Swezey, R.W., Complexity, Managers, and Organizations, Academic Press, Inc., 1986.

Thomas, D., and Mog, R., "A Quantitative Metric of System Development Complexity," 7th Annual INCOSE 1997 Symposium, August, 1997.

Thomas, D., Private Communication, August 12, 1996.

Tsoukalas, L. H., and Uhrig, R. E., Fuzzy and Neural Approaches in Engineering, John Wiley & Sons, Inc., 1997.

West, D.B., Introduction to Graph Theory, Prentice-Hall, Inc., 1996.

# APPENDIX A: Programmatic Database

| Program | Total Cost (1996 $M) | Cost Growth (%) | Duration (years) | Chromatic # |
|---|---|---|---|---|
| ORB | 11838 | 30 | 7.7 | 11 |
| SSME | 2617 | 161 | 6.9 | 5 |
| HST | 1893.3 | 163.8 | 9 | 4 |
| GAL | 1119 | 177 | 10.2 | 6 |
| ET | 811 | 73 | 7.5 | 6 |
| SRB | 664 | 22 | 5.3 | 5 |
| UARS | 567.1 | -15.8 | 6.5 | 2 |
| SRM | 567 | 114 | 6 | 4 |
| Magellan | 543.2 | 50.1 | 5.583 | 2 |
| ACTS | 399.8 | 39.4 | 9.083 | 3 |
| GRO | 394 | 125 | 7 | 6 |
| COBE | 193 | 3 | 7.33 | 4 |
| Mars | 170 | 0 | 4.1 | 9 |
| TOPEX | 156.3 | 155.4 | 9.083 | 3 |
| ERBS | 146.1 | 42.8 | 3.83 | 2 |
| TIROS-N | 145.3 | 12.6 | 3.25 | 2 |
| DE | 114.7 | 34.6 | 4.0833 | 2 |
| PV ORB | 105.3 | 15.3 | 4.25 | 5 |
| PV Large | 99.2 | 125.1 | 4.833 | 4 |
| CRRES | 72.6 | 36.4 | 7.23 | 6 |
| TOMS-EP | 51 | 17 | 4 | 6 |
| AMPTE | 44.2 | 59 | 2.5 | 2 |
| PV Small | 38.5 | 64.4 | 4.833 | 4 |
| STEP-0 | 31 | 8 | 3.5 | 6 |
| FAST | 28 | 3 | 2.8 | 5 |
| Lewis | 25 | 15 | 3.2 | 6 |
| DARPASAT | 16 | 13 | 4 | 5 |
| SME | 11.4 | 0 | 2.3 | 2 |
| ORSTED | 9 | 10 | 4.1 | 5 |
| HETE | 4.9 | 27 | 3.1 | 5 |

# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No. 0704-0188*



| 1. AGENCY USE ONLY *(Leave blank)* | 2. REPORT DATE | 3. REPORT TYPE AND DATES COVERED |
|---|---|---|
| | April 1999 | Final Report |

**4. TITLE AND SUBTITLE**
Systems Engineering Design Via Experimental Operations Research

**5. FUNDING NUMBERS**
C - H27473D

**6. AUTHORS**
Robert A. Mog, Ph.D.

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**

OR Applications
7811 Lent Drive
Huntsville, AL 35802

**8. PERFORMING ORGANIZATION REPORT NUMBER**

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**

NASA-MSFC
George C. Marshall Space Flight Center
MSFC, AL 35812

**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

**11. SUPPLEMENTARY NOTES**

**12a. DISTRIBUTION/AVAILABILITY STATEMENT**

Unlimited Distribution

**12b. DISTRIBUTION CODE**

**13. *ABSTRACT (Maximum 200 words)***

*Unique and innovative graph theory, neural network, organizational modeling, and genetic algorithms are applied to the design and evolution of programmatic and organizational architectures. Graph theory representations of programs and organizations increase modeling capabilities and flexibility, while illuminating preferable programmatic/organizational design features. Treating programs and organizations as neural networks results in better system synthesis, and more robust data modeling. Organizational modeling using covariance structures enhances the determination of organizational risk factors. Genetic algorithms improve programmatic evolution characteristics, while shedding light on rulebase requirements for achieving specified technological readiness levels, given budget and schedule resources. This program of research improves the robustness and verifiability of systems synthesis tools, including the Complex Organizational Metric for Programmatic Risk Environments (COMPRE).*

**14. SUBJECT TERMS**
Complexity, Systems Engineering, Metrics, Architecture, Hierarchy, Programmatic, Risk Assessment,

Artificial Neural Networks, Graph Theory, Genetic Algorithms.

**15. NUMBER OF PAGES**
31

**16. PRICE CODE**

| 17. SECURITY CLASSIFICATION OF REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
|---|---|---|---|
| Unclassified | Unclassified | Unclassified | UL |