The Pennsylvania State University

The Graduate School

College of Information Sciences and Technology

PARALLEL GENETIC ALGORITHM OPTIMIZATION OF A COGNITIVE MODEL: INVESTIGATING GROUP AND INDIVIDUAL PERFORMANCE ON A MATH STRESSOR TASK

A Dissertation in

Information Sciences and Technology

by

Sue E. Kase

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

December 2008

The dissertation of Sue E. Kase was reviewed and approved* by the following:

Frank E. Ritter
Associate Professor of Information Sciences and Technology
Dissertation Advisor
Chair of Committee

Mary Beth Rosson
Professor of Information Sciences and Technology

Xiaolong Zhang
Assistant Professor of Information Sciences and Technology

Suzanne M. Shontz
Assistant Professor of Computer Science and Engineering

Laura Cousino Klein
Associate Professor of Biobehavioral Health

John Yen
Associate Dean for Research and Graduate Programs
College of Information Sciences and Technology

*Signatures are on file in the Graduate School

ABSTRACT

The purpose of the thesis was to enable an individual differences investigation of how stress affects cognitive performance in a mental serial subtraction task. To do this a new optimization approach was developed for computing the fit of a serial subtraction cognitive model to human performance data by varying architectural parameters. The new optimization approach utilizes a modified parallel genetic algorithm (PGA) on a high performance computing (HPC) cluster together with the ACT-R cognitive architecture and a serial subtraction cognitive model. Modifications to the PGA were necessary because of stochasticity embedded in the subsymbolic processes of the model's architecture. The optimization approach in this thesis was highly successful in fitting the serial subtraction model to three different levels of human data: average across subjects, challenge and threat task appraisal groups, and single-subject level performance. The optimization results revealed several interesting patterns in the parametric values found to produce best fits to the human data. Two of the patterns are supported by individual differences theories of stress and anxiety from cognitive performance research. If the field's traditional manual optimization process had been used instead, these patterns would have gone undiscovered. Contributions of the thesis include: development of a software system for individual differences modeling; increased understanding of ACT-R modeling, and the effects of anxiety on cognitive performance; a new application of PGAs; and a new paradigm for model creation and validation as the purposefulness of the thesis optimization approach was shown applicable in both model development and model validation.

TABLE OF CONTENTS

List of Figures

List of Tables

Acknowledgements

This research project was partially supported by the US Office of Naval Research, award number N000140310248. Computational resources were provided by TeraGrid grant DAC TG-IRI070000T. Thanks goes to Laura Klein and her lab at the Department of Biobehavioral Health, Penn State University, for collection of the human performance data; and to Michael Schoelles at the Cognitive Science Department, Rensselaer Polytechnic Institute, for assistance and use of the serial subtraction cognitive model. Thanks also goes to Suzanne Shontz at the Department of Computer Science and Engineering, Penn State University, for her helpful comments concerning the optimization component of the thesis.

Chapter 1

INTRODUCTION

The purpose of the thesis was to enable an individual differences investigation of how stress affects cognitive performance in a mental serial subtraction task. To do this a new optimization approach was developed for computing the fit of cognitive models to human data by varying architectural parameters. Two important background components of the thesis supported this purpose.

The first background component involves performance and appraisal data collected from human subjects performing the serial subtraction task. The serial subtraction task was part of a larger project to study the effects of stress and caffeine on cardiovascular health conducted by Dr. Laura Klein of the Biobehavioral Health Department and Dr. Frank Ritter of the College of Information Sciences and Technology at Penn State University (Kase, Ritter, & Schoelles, 2007, 2008; Klein, Whetzel, Bennett, Ritter, & Granger, 2006; Ritter, Schoelles, Klein, & Kase, 2007; Whetzel, Ritter, & Klein, 2006). The serial subtraction task is the mental arithmetic portion of the Trier Social Stress Task, a task that has been used to induce a reliable stress response in subjects in a laboratory setting since the 1960's (Kirschbaum, Pirke, & Hellhammer, 1993). During the serial subtraction task, subjects repeatedly subtracted either 7 or 13 from a 4-digit starting number. Subjects perform the task mentally, subtracting without visual or paper cues, and verbalizing the answer to each subtraction problem.

The task environment and situational characteristics enhance the stressful nature of the task. There is an anticipation period when the task is explained to the subject, and a pre-task appraisal questionnaire is administered centered on self-reported anxiety associated with performing the task. A female experimenter is seated directly in front of the subject in close

proximity with a timing device and solution checklist. The experimenter tells the subject when he is wrong and at which 4-digit number he should restart. Subjects are told their performance will be voice recorded for the purpose of an across-subject analysis by a panel of psychologists. Emphasis is placed on completing the task as quickly and accurately as possible. The experimenter interjects a comment halfway through the subject's performance about 'running out of time' and 'needing to go faster'.

These situational characteristics trigger stress-related states such as threatening task appraisal, stereotype threat, anxiety, and worry or self-preoccupation about performing poorly. This extra situational burden is thought to interfere with the ability to perform as well on the task at hand as might otherwise be possible. Threat and anxiety effects can degrade the ability to regulate attention during complex tasks where it is necessary to coordinate information processing online and inhibit thoughts and feeling that are counterproductive to one's current goals. Cognitive psychologists describe the mechanism that is responsible for this sort of efficient regulation as the central executive or working memory. In cognitive performance research, stressful states are thought to reduce working memory capacity and effectiveness. In general, working memory provides the resources needed to retrieve and maintain information during a wide variety of cognitive processes. More specifically, working memory resources are required for performing mental arithmetic tasks where an individual must hold the original problem and any intermediate results in memory while performing borrow and place-keeping operations when working toward a final answer. Through a series of studies, Ashcraft and Kirk (2001) found that a major contributor to the performance deficits characterized by math-anxiety subjects stems from the central executive portion of working memory. Anxiety and worrisome thoughts consume the limited attentional resources of working memory which are then less available for task processing.

The task appraisal questionnaire was used to group subjects into a ‘threat’ (little control over performance, less able to cope with the task) or a ‘challenge’ (in control, can more than cope with the task) appraisal group. Subjects’ subtraction answers were scored against a list of correct answers from the starting number. For each subject the number of subtraction problem attempts were recorded and a percentage correct score was calculated by dividing the total number of correct attempts by the total number of attempts for each block of the serial subtraction.

The second background component involves a cognitive model of the serial subtraction task developed to simulate how subjects were performing the task. A cognitive model performs a specific cognitive task and produces behavior in the form of predictions that can be compared to data from human performance of the task. A cognitive model produces both a theory of human behavior on a task and a computational artifact in the form of a computer program that performs the task. Cognitive models are developed and executed in a simulation type of environment called a cognitive architecture. A cognitive architecture is a computational system that specifies a particular way of representing information and a fixed set of cognitive mechanisms for processing that information. Modeling within a cognitive architecture imposes top-down constraints in the theory of the given architecture. Specifically, any model built within a cognitive architecture must use the same representations and mechanisms regardless of the task being modeled, just as the brain presumably employs a common set of information-processing mechanisms across a variety of tasks. What differs across various cognitive models built within a given architecture is the task-specific knowledge.

A cognitive model of the serial subtraction task was developed in the ACT-R 6.0 cognitive architecture by Dr. Michael Schoelles at the Cognitive Science Department at Rensselaer Polytechnic Institute. The ACT-R architecture was chosen to model the serial subtraction task because of its subsymbolic level of processing and parallel execution of its verbal system with control and memory systems. The model performs a block of serial subtractions,

subtracting by 7s or 13s from a 4-digit starting number. The model's declarative knowledge consists of arithmetic facts and goal-related information. The model's procedural knowledge is in the form of production rules that allow for retrieval of subtractions by attempting to recall the appropriate declarative memory fact to produce a subtraction answer. The serial subtraction model uses the ACT-R vocal module to verbalize the subtraction answers as the subjects do.

Declarative and procedure knowledge are symbolic level structures in ACT-R. In most cognitive models much of the quantitative structure of cognition is at the subsymbolic level. ACT-R offers many parameters to manipulate and adjust subsymbolic processes with each parameter having a meaning associated with a specific module and subsymbolic process. Adjusting the values of parameters adjusts the architecture's theory of cognition for the purpose of modeling cognitive aspects of the task. Cognitive models are validated in human behavior where a 'good' model can predict human performance on a specific task. The architectural parameters manipulate the model's performance to more accurately match, or *fit*, the human data. A model's fit is usually assessed by correlating one or several quantitative measures collected in a human subject experiment to the corresponding quantitative measures produced by the model. This is called a parametric approach to cognitive modeling.

Selecting the appropriate parameters to investigate out of the many offered by the architecture is guided by the type of task, hypothesis or theory, and past cognitive modeling research. Once specific architectural parameters are selected, value constraints (minimum and maximum boundaries) for each parameter define a parametric search space. The parameters act as an input to the cognitive model producing performance predictions as output.

In the thesis, three particular ACT-R parameters appeared important in the serial subtraction model. The rate the model speaks is controlled by the seconds-per-syllable parameter (SYL). The other two parameters affect declarative knowledge access: the base level constant

(BLC) and the activation noise parameter (ANS). A minimum and maximum value define the range of interest for each parameter and form a 3-D search space.

Traditionally, cognitive modelers use a manual optimization process to fit the model to the human data. This is a time consuming, iterative, trial-and-error-like process involving selection of a set of parameters, assigning values to those parameters, running the model in the cognitive architecture, and evaluating the resulting model predictions against the human data. If the model's fit is found to be unsatisfactory, this process is repeated a number of times until a 'good enough' fit is found or the model is determined to be an inadequate representation of the task. This optimization process can be complicated by stochastic components in the cognitive architecture and model, or by a wide range of human performance variance on the task, or both—as is the case in the thesis. For example, with a static set of parameters and values, the combination of the model and its execution in the cognitive architecture can yield a distribution of performance not just a single performance statistic.

In the stress and caffeine experiment, the human data from the serial subtraction task showed a wide range of human performance on the task characterized by large standard deviations in most cases when performance was averaged across subjects. This same variability was noted in an earlier experiment utilizing the serial subtraction task where model-fitting attempts to fit an average performance across subjects failed using the manual optimization approach (Ritter, Schoelles, Klein, & Kase, 2007). The human performance variability, and the failed previous attempt to fit the model suggested that the best model fitting approach would be to fit individual subjects or small groups of individual subjects if their behavior could be clustered by common characteristics. The common characteristic in this case was task appraisal which defined challenge and threat appraisal groups of individual subjects.

Modeling the performance of individual subjects is rarely attempted for a number of reasons. A greater amount of noise is associated with individual data, and generally more free

parameters are involved in the modeling process. In cognitive modeling, as the number of parameters manipulated by the model increases, so do the time and computational resources required to search the parameter space for values that produce the best model fits.

The model-to-human data validation is an optimization problem. If the modeler wants to investigate individual-level cognitive performance, the traditional manual optimization process becomes too inefficient and cumbersome. This thesis proposes a systematic and more automated optimization approach that enables the study of individual differences in cognitive behavior.

The optimization approach in this thesis integrates a parallel random search algorithm on a high-performance computing platform with the ACT-R cognitive architecture (Anderson et al., 2004) and the cognitive model of the serial subtraction task (Ritter, Schoelles, Klein, & Kase, 2007). Three algorithmic approaches were considered for the model optimization in a parallel processing environment: a parallel genetic algorithm, a hill-climbing algorithm, and a hill-climbing-genetic hybrid algorithm. The optimization problem of searching the ACT-R parameter space for parameter combinations that adjust the serial subtraction model's predictions to match the human data is a stochastic global optimization problem. The ACT-R parametric landscape is relatively unknown but assumed to be complex, maybe even fragmented, with multiple local optima (Gluck & Harris, 2008a; Young, 2003). The optimization approach in the thesis focuses on finding global minima in the ACT-R landscape.

Three types of algorithms were developed for parallel processing and tested with the ACT-R architecture and the serial subtraction model. Preliminary testing concluded that the parallel genetic algorithm (PGA) was the most appropriate for the optimization. A difficulty in the fitting of the model is finding the global minimum as opposed to a local minimum. In general, genetic algorithms (GAs) have been reported to be successful over a growing range of difficult problems from many different disciplines (Cantu-Paz, 2001; Coley, 1999; Goldberg, 1994; Haupt & Haupt, 2004). Much of this proven utility arises from the robustness of the algorithm in

navigating around complex search spaces and avoiding entrapment by local optima (Goldberg, 1989). A disadvantage of hill-climbing is that it will generally find only local minima or maxima unless the search space is concave or convex. During preliminary tests, the hill-climbing algorithm terminated fairly quickly at the first encountered local minima.

GAs are based on the mechanics of natural selection and genetics (Goldberg, 1989). They combine survival of the fittest among a population of strings, called genotypes, with a structured, yet randomized, search for solutions well-suited for large search spaces. The genotypes making up the population can encode the values of different variables being optimized (Haupt & Haupt, 2004). The PGA algorithm in the thesis encoded the values of the three ACT-R parameters of interest, namely ANS, BLC and SYL. The algorithm proceeds in an iterative manner by applying operators such as selection, crossover, and mutation on the initial random population in order to evolve a new generation of genotypes. A fitness function associates a fitness measure with every genotype indicating its suitability to the problem. For the PGA in the thesis the fitness function was defined by the sum of squares error (discrepancy between the model prediction and the human data) calculated on both the number of attempted subtraction problems and the percentage correct from one block of subtracting by 7s.

Development of the PGA was complicated by stochastic effects in the architecture and in the model. Preliminary testing showed that the stochasticity caused lack of convergence as the PGA could not hold on to best fitting genotypes from one generation to the next. The PGA was redesigned to incorporate genotype collection and testing functions. These additions to the algorithm were shown to ameliorate the lack of convergence problem. This modified version of the PGA was used to fit the serial subtraction model to data from the 15 subjects in the control group of the experiment at the following levels of analysis: fitting to an average performance across subjects; fitting to individuals grouped by challenge and threat appraisal; and fitting to each individual subject that performed the serial subtraction task.

At all three levels of data fitting, the PGA optimization approach found nearly perfect solutions, i.e., sets of ACT-R parameter values that adjusted the serial subtraction model's predictions to match the human data. When fitting to an average across subjects, the PGA found five solutions producing excellent model-to-data fits within one subtraction problem for number of attempts and less than 0.6 difference in percentage correct. This was much improved over the previous attempt of fitting to an average across subjects (Ritter, Schoelles, Klein, & Kase, 2007). When fitting to appraisal group averages, the PGA found 10 solutions for the challenge group average and four solutions for the threat group average. Plots tracking the minimum and average fitness values across the generations showed different degrees of nonmonotonic patterns between the two groups. The optimization for the threat group showed greater variability in minimum fitness forming dynamic peaks and valleys, where as the optimization for the challenge group was less variable and showed signs of flattening out and converging at about 70 generations. The nonmonotonic pattern in the variability of the minimum fitness value appeared to be associated with finding fewer solutions for the threat group average. When fitting to individual subjects, the PGA found good solutions for all subjects with fitness values ranging from 0.0006 to 0.77. The PGA seeks a global minima in the ACT-R parameter space; therefore, model predictions exactly matching human performance data would have a fitness value equivalent to 0.

The optimization results generated by the PGAs revealed several interesting patterns in the ACT-R parameter sets found to produce best fits to the human data. One pattern involved the large portion of ANS values greater than 0.5; this value is considered high by the cognitive modeling community. High ANS values were found in the solutions at all three levels of analysis. This is thought to be related to the fact that the serial subtraction task itself is very stressful in nature. These high ANS values may be indicative of the large amount of variance observed in the subjects' performance and noise usually associated with individual differences that the procedural model structure is not able to explain. In ACT-R high ANS contributes to greater variability in

both the probability of retrieving a chunk (a declarative memory fact) and the retrieval time; this in turn, can be thought to simulate distraction and interference caused by anxiety and worrisome thoughts of poor performance on the attentional resources of working memory (Eysenck & Calvo, 1992).

A small subset of lower range ANS values were contained in solutions for fitting to average over all subjects, average to the challenge group, and several individual subjects, but not when fitting to the average of the threat group. The lower ANS values appeared to be associated with lower BLC values in the solution sets pointing to a pattern of simultaneously increasing/decreasing ANS and BLC parameter value pairs.

The ANS BLC pattern was rather unexpected and difficult to interpret. ANS and BLC are parameters utilized in the declarative module involved in subsymbolic activation processing. The BLC value is a constant value summed into the base-level activation equation resulting in an increase to a chunk's base-level activation. In the retrieval process, the chunk with the highest activation that matches a request is placed into the retrieval buffer thereby becoming available to productions. This would represent a human wanting to access specific information from memory, and being successful in accessing that information without delay. A higher activation in general means greater probability of retrieval in a shorter amount of time. What would be expected is an inverse relationship between ANS and BLC where one parameter increases in value as the other decreases.

For example, when predicting low performers that are anxious about the task or worried about not doing well, performance is generally characterized by fewer number of attempted subtraction problems and increased errors. This type of performance might be simulated with high ANS values reflecting greater noise and more variability, accompanied by low BLC values suggesting little additional activation beyond what is accounted for by recency and frequency of chunk use. Instead the optimization solutions for low performers showed more of a concurrent

relationship, high ANS values associated with high BLC values, and the opposite for high performers, low ANS values associated with low BLC values.

Another pattern involved SYL, the seconds per syllable parameter. The optimization results showed that as performance decreased on serial subtraction the value of SYL increased. This appeared to indicate that high performing subjects were speaking more quickly than their poor performing counterparts. This pattern was especially strong in the individual subject optimization results. In the cognitive psychology literature the concept of worry is sometimes linked to inner verbal activity (e.g., Rapee, 1993). In the ACT-R cognitive architecture inner verbal activity is the responsibility of the vocal module. It might be that a 'threatened' subject's worry over his performance is interfering or competing with his speaking of the subtraction answers thus having the effect of generally decreasing subtraction performance.

If the traditional manual optimization approach had been used to fit the serial subtraction model most of the above described patterns would have gone undiscovered. High ANS values would not have been manually assigned and tested. Manual optimization produces only one solution; therefore, multiple solution sets containing both high and low ANS and BLC value pairs would not have been generated for comparison. Additionally, cognitive modeling almost unanimously is concerned with aggregate performance where manual optimization techniques are adequate for model to data validations. The automated search technique of the optimization approach in the thesis allowed fitting to individual subject performance. The individual subjects' optimizations uncovered fine-grained incremental changes of SYL values as predicted performance gradually decreased.

Despite the success of the PGA optimization approach in fitting the serial subtraction model to three levels of human data flawlessly, some concerns were raised. Two of the three patterns observed in the PGA solution results appeared supported by theories of stress, anxiety, and worry. The third pattern of simultaneously increasing/decreasing values of ANS and BLC

cannot be explained at this point in time. It is likely the case that ANS and BLC are not good parametric choices and that other parameters need to be investigated. In a manual optimization process, most likely, selecting the 'wrong' parameters to manipulate would result in not being able to fit the model to the human data with accuracy. This was not the case with the thesis optimization approach, where every model to human data fitting problem was solved.

Manual optimization generally yields one solution, taken to be the best solution. The PGA optimization technique produced not one solution, but as many as nine different solutions with fitness values close to 0. One characteristic of a genetic algorithm that sets it apart from other search algorithms is its ability to maintain a set of candidate solutions through the optimization process. The problem with the PGA optimizations of the model is that some of the solutions for a given optimization problem lay in different regions of the ACT-R parameter space. It would be preferred, even expected, that the solutions resulting from an optimization would fall approximately in one specific area of the ACT-R parameter space either supporting or not supporting the modeler's hypothesis. This would be the case unless several global minima are found; thereby, the optimization approach may work a little too well for only validation purposes. The purposefulness of the optimization approach in the thesis was shown to be applicable in both the model development process and model validation.

This thesis makes several contributions to cognitive science research. A new optimization system was developed for individual and group differences modeling that utilizes a high performance cluster platform and a parallel genetic algorithm. When analyzed the optimization solutions showed several patterns associated with theories of stress and anxiety on cognitive performance, in general, and specifically arithmetic performance. The solutions, in the form of ACT-R parameter sets, were obtained by sampling the architectural parameter space at 20,000 points during each optimization. The investigations of the many possible parameter combinations contribute to a general understanding of the theory underlying the ACT-R cognitive architecture's

integration of modules and subsymbolic processes. The thesis optimization approach for fitting cognitive models to human data is a new application of PGAs. Although this fitting process is typically associated with model validation, the optimization approach is potentially useful throughout the model development process. The exploratory nature of PGAs allows for hypothesis and theory development to be incorporated with model building and validation.

Structure of Thesis

The thesis is divided into six chapters. The general structure of each chapter is as follows. This chapter, the introduction, served the purpose of summarizing the thesis from start to finish unconfined by chapter divisions and topic headings.

Chapter 2 presents the primary components of the thesis in a literature review format. An abbreviated account of the cognitive science field accompanied by an overview of one of the most commonly used production-system architectures from the symbolic representation perspective of cognitive science is described. The chapter also covers select theories from cognitive performance focused on stress, appraisal, anxiety, and worry. Several working memory models and modeling of individual differences in general and in reference to working memory are discussed. The last section of the chapter overviews two types of search algorithms relevant to the thesis.

Chapter 3 covers the thesis project development. At the beginning of the chapter two background components playing an important role in the thesis are explained: the human subject experiment where data were collected and later used in model optimization; and a cognitive model of a serial subtraction task—the target model for the thesis optimization approach. The primary sections of the chapter describe in detail the development of two types of optimization

algorithms, a genetic algorithm and a hill-climbing algorithm, and also a hybridized version of these algorithms combined.

Chapter 4 presents the results of the thesis optimization approach when applied to three different levels of data analysis: average across subjects, task appraisal group averages, and individual subjects. The chapter also discusses three patterns observed in the solution sets generated by the optimizations.

Chapter 5 is divided into three sections. The first section proposes interpretations of the patterns detected in the previous chapter in reference to working memory and theories of stress. The second and third sections present 3-D visualizations of the ACT-R parameter landscape and prediction distributions generated by multiple model runs.

The final chapter, Chapter 6, summarizes the contributions and presents a variety of implications in reference to the thesis as well as future directions and limitations of the research. One of these implications builds upon theories from organizational science and modeling.

Chapter 2

THEORETICAL COMPONENTS

The thesis is conducted within the field of cognitive science. The hallmark of cognitive science is its interdisciplinary approach and mix of methodologies and perspectives. This interdisciplinary nature is reflected in the contents of this chapter. For the unfamiliar audience, cognitive science is briefly outlined along with the field's unique information processing view of the mind. Research from cognitive psychology contributes substantially to cognitive science. Cognitive scientists package cognitive psychology theories into computational architectures resembling simulation environments. One such cognitive architecture, ACT-R, is described. Cognitive architectures are used to develop and execute process models of human cognition. The modeling of cognitive performance is of primary interest.

Theories and concepts thought to effect specific aspects of cognitive performance related to the thesis are reviewed. The goal of cognitive modeling is to predict and explain cognitive performance. Model validation is accomplished by examining the performance of human subjects. The agreement of model predictions to human data by adjusting model parameters can be formulated as an optimization problem. The types of optimization algorithms utilized in the thesis are discussed. Lastly, the primary components as an integration of the above theoretical and conceptual areas pertaining to the thesis project are summarized.

Cognitive Science and ACT-R

This section briefly overviews the field of cognitive science where the theoretical framework of this thesis is situated. A particular cognitive architecture, ACT-R, is used to

construct a prototype cognitive model of a mental arithmetic task to investigate individual differences in cognitive performance.

Cognitive science can be roughly summed up as the scientific interdisciplinary study of the mind. Its primary methodology is the scientific method, although many other methodologies also contribute. It results from the efforts of researchers working in a wide range of fields. These include philosophy, psychology, linguistics, artificial intelligence, robotics, and neuroscience. Each field brings with it an array of tools and perspectives. The term cognitive science refers not so much to the sum of all these disciplines but to their intersection or converging work on specific problems. In this sense, cognitive science is not a unified field of study like each of the disciplines themselves, but a collaborative effort among researchers working in various fields. The glue that holds cognitive science together is the topic of the mind and, for the most part, the use of scientific methods (Friedenberg & Silverman, 2006).

Cognitive science takes a theoretical perspective of the mind centered on the idea of computation, also called information processing. Cognitive scientists view the mind as an information processor. Information processors must both represent and transform information. That is, a mind, according to this perspective, must incorporate some form of mental representation and processes that act on and manipulate that information.

At a high level of abstraction this perspective of the mind is as analogous to computers as information processors. A computer performs a variety of information processing tasks. Information gets into the computer via input devices, such as a keyboard or network connection. That information can then be stored on the computer on a hard drive or other storage device. The information can then be processed using software applications such as a text editor. The results of this processing may next serve as output, either to a display or printer. People performing tasks can be thought of in a similar fashion. Information is ‘input’ into our minds through perception, i.e., what we see or hear. It is stored in our memories and processed in the form of thought. Our

thoughts can then serve as the bases for 'outputs', such as language or physical behavior. Both systems can be characterized by computation (Friedenberg & Silverman, 2006). One could say that cognitive scientists view the mind as a machine or mechanism whose workings they are trying to understand.

Most cognitive scientists believe that computational modeling is the most promising method for describing, understanding, and predicting how the mechanisms of the mind work. A cognitive model, in the form of a working computer program, is intended to be an explanation of how some aspect of cognition is accomplished by a set of primitive computational processes. A cognitive model performs a specific cognitive task or class of tasks and produces behavior that constitutes a set of predictions that can be compared to data from human performance. It is believed that cognitive models that show behavioral similarity with human subjects in comparable experimental or natural situations, embody internal processes (weakly or strongly) equivalent to those occurring in the subjects (Pylyshyn, 1989). *A cognitive model produces both a theory of human behavior on a task and a computational artifact that performs the task*.

The advantage of using computational models is that models can provide a concise way of formulating how a given mental process might operate. They are more precise than verbal theories and can be tested using experimental data. The implementation of a theory as a model and the subsequent modification of the model on the basis of empirical findings are a fruitful method for uncovering more about a cognitive process. One disadvantage of computational models is that the two distinct model-building paradigms in cognitive science (the traditional information processing view of cognition versus the neural network view of cognition) need to be reconciled (Friedenberg & Silverman, 2006).

There are two views of computation in cognitive science: a symbolic representational view and a connectionist view. The dominant paradigm, the symbolic view, uses symbols to represent information. The symbols are represented, stored, and operated upon by cognitive or

computer processes. Knowledge is represented locally in the form of symbols, and processing occurs in discrete stages. The connectionist view takes a network approach to computation viewing knowledge as a pattern of activation or weights that is distributed throughout a network. Connectionism processing occurs in parallel through the simultaneous activation of nodes. Therefore, in the cognitive modeling process, different functional architectures are used depending on the computational view a cognitive scientist is working under. The cognitive modeling component of the thesis falls under the symbolic representational view of cognitive science.

In the symbolic representational view, generally, cognitive models are embedded in broader computational architectures of cognition (e.g., Anderson et al., 2004; Just & Carpenter, 1992; Meyer & Kieras, 1997; Newell, 1990; Ritter, 2004; Schneider & Chein, 2003) that offer theoretical and/or neurobiological constraints. These cognitive architectures posit that a fixed set of computational mechanisms and resources underlie a wide range of human cognition across tasks and across individuals. A model constructed within a cognitive architecture uses computational mechanisms that are proposed by the architecture's underlying theory.

In the symbolic representational paradigm programming languages designed for cognitive modeling tend to be production systems. Production systems are used as a flexible model of the control structure of human cognition. The flow of processing is not controlled in advance by a fixed program, but instead a set of production rules (condition-action pairs) may execute any time their conditions are satisfied. Therefore, the flow of control is at runtime and is a function of dynamically evolving memory contents triggering the productions. A cognitive model written in a production system makes theoretical commitments at the level of the production rules, and when built within a cognitive architecture defines a computationally complete system. One of the most commonly used production-system type of architectures is the ACT-R (Adaptive Control of Thought - Rational, Anderson et al., 2004) cognitive architecture.

The term “cognitive architecture” was introduced into cognitive science by Allen Newell out of analogy to computer architectures (Bell & Newell, 1971) and introduced into computer science by Fred Brooks (1962) out of analogy to architecture as in the design of buildings. Newell’s (1990) definition of cognitive architecture is “the fixed (or slowly varying) structure that forms the framework for the immediate processes of cognitive performance and learning” (p. 111). Anderson’s (1983) definition of cognitive architecture is “a theory of the basic principles of operation built into the cognitive system” (p. ix); more recently (Anderson, 2007), “a cognitive architecture is a specification of the brain at a level of abstraction that explains how it achieves the function of adaptive cognition” (p.7).

The ACT-R cognitive architecture is the product of a community of researchers led by John Anderson at Carnegie Mellon University. ACT-R is a two-layer, modular, production system architecture based on condition-action rules which execute specified actions when the specified conditions are met. In this architecture cognition emerges through the interaction of a number of independent modules (Anderson, 2005). Each of these modules is associated with specific brain regions and theories about the internal processes of these modules (Anderson & Lebiere, 1998).

Figure 2.1 shows the structure of ACT-R with the eight modules that are standard as part of the ACT-R 6.0 cognitive architecture. There are currently two perceptual modules: a visual module and an aural module. There are two response modules: a manual module and a vocal module. The other four modules are responsible for different aspects of central processing. The imaginal module holds a current mental representation of the problem. For example, if solving an equation $2x – 4 = 10$, the imaginal module might hold a representation of an intermediate equation such as $2x = 14$. The declarative module retrieves critical information from memory such as mathematics facts $10 + 4 = 14$. The goal module keeps track of the model’s current

intentions in solving the problem, for example, intent to factor a quadratic equation. The procedural module embodies various rules for behavior, such as the rules for solving equations.

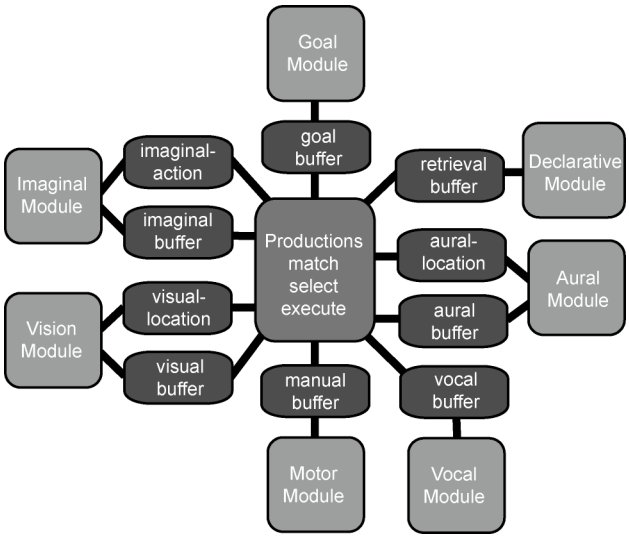

Figure 2.1: The modules implemented in the ACT-R 6.0 cognitive architecture.

Each of the modules is capable of parallel computation, but communication among the modules has some serial bottlenecks (Anderson, 2007). The only way these modules can communicate is through buffers associated with each module. Only a little information can be put into a buffer—a chunk (a structured unit bundling a small amount of information). Communication among the modules is achieved via the procedural module. The procedural module can respond to information in the buffers of other modules and put information into these buffers. The response tendencies of the central procedural module are represented as production rules.

Declarative memory is the module by which one would be able to perceive their past. The declarative module maintains and processes a store of knowledge. It is capable of parallel processing in which a single memory prompt can zoom in on the appropriate memory to retrieve. On the other hand, one may struggle to retrieve a memory that may be there but somehow is not

available. Declarative memory tries to give us, moment by moment, the most appropriate possible window into our past (Anderson, 2007). Chunks in declarative memory have activation values that determine the speed and success of their retrieval.

Building a cognitive model to run in the ACT-R cognitive architecture consists of implementing the declarative and procedural knowledge of the domain and the task. The declarative knowledge is implemented as chunks that contain information about the current perceived state of the world as well as facts related to the domain. Examples of declarative knowledge include “There is an object in front of agent number 5”; “The object in front of agent number 5 is green”; and “Tanks are green.”

The procedural knowledge is implemented as sets of productions where each production is a condition-action pair. The condition specifies what must be known for the production to apply, and the action specifies the things to do if the production applies. The conditions test the knowledge contained in declarative memory, while the actions modify the declarative memory or perform physical actions. Examples of procedural knowledge (production rules) include “IF the goal is to find objects THEN send an agent to search for objects”; “IF the goal is to search for objects and object has been found THEN determine the type of an object”; and “IF the goal is to determine the type of an object and the color of the object is green THEN the object is a tank.” Given this set of productions and chunks shown above, the model would be about to deduce a tank has been found.

With ACT-R models much of the quantitative structure of human cognition is at a subsymbolic level (also referred to as ‘rational’ level), rather than the symbolic level structures described above. The subsymbolic level contains quantities that participate in neural-like processes, which determine the chunk and production access in memory. For example, at the subsymbolic level, every chunk has associated with it a numerical value called its activation. The activation reflects the degree to which past experiences and current context indicate the chunk

will be useful at any particular moment. The chunk with the greatest activation among those that match the specification of a retrieval request is selected. The one constraint is the parameter called the retrieval threshold (RT) that sets the minimum activation a chunk can have and still be retrieved. Additionally, there is a noise value associated with each chunk that represents instantaneous noise (the variance from trial to trial) that can be set with the activation noise parameter (ANS). The activation of a chunk also determines how quickly it can be retrieved. When a retrieval request is made the time it takes until the chunk that is retrieved is available can be controlled with the latency factor (LF) parameter.

ACT-R was chosen for this thesis project because it provides a rigorous framework for cognitive modeling as well as a set of built-in parameters and constraints on cognition to facilitate a priori predictions about the behaviors of interest in this study and psychologically plausible models in general. More specifically related to the mental arithmetic task being modeled, ACT-R's subsymbolic level can implement changes in processing, and permits parallel execution of the verbal system with the control and memory systems. Additionally, ACT-R has been used for other models of addition and subtraction developed by other researchers (e.g., Lebiere, 1998); therefore, representations of integers and math facts can be transferred or reused from other math models.

Relevant Theories in Cognitive Performance

This section overviews selected theories in the field of cognitive performance that are relevant to the thesis project. The first subsection describes four theories focused on anxiety and cognitive performance. The theories are differentiated somewhat by level of analysis (individual versus group) and by degree or type of anxiousness (stress, threatening appraisal, and worry). The second subsection briefly discusses well-known theories of working memory utilized in anxiety

and cognitive performance research. Memory models having influence on the ACT-R cognitive architecture are reviewed as well. The third subsection takes a glimpse at cognitive modeling of individual differences and how these types of investigations might uncover important cognitive processes currently overlooked by the long-standing tradition of modeling average or aggregate performance.

Stress, Threat, Anxiety, and Worry

This thesis project utilized human performance data collected from a mental arithmetic task called serial subtraction. The serial subtraction task is a portion of an experimental protocol used to induce moderate psychological stress in human subjects in a laboratory setting (Kirschbaum, Pirke, & Hellhammer, 1993). In addition to performance data, task appraisal data were collected about self-reported levels of anxiety in anticipation of the serial subtraction task and after performing the task. Based on a task appraisal score, subjects were divided into a 'threat' and 'challenge' appraisal group. One minute passes between the explanation of the task to the subject and actual performance of the task begins. It is emphasized that the task should be performed as quickly and accurately as possible. Subjects were also informed that their performance would be voice recorded, reviewed by a panel of psychologists, and compared to the other subjects participating in the experiment; therefore, it was important to do well on the task.

Of interest in the thesis was the effects of anxiety that may have been brought on by the anticipation and actual performance of the serial subtraction task. Anxiety is defined as an aversive emotional and motivational state occurring in threatening circumstances (Eysenck, 1992). State anxiety (the currently experienced level of anxiety) can be conceptualized as "a state in which an individual is unable to instigate a clear pattern of behavior to remove or alter the event/object/interpretation that is threatening an existing goal" (Power & Dalgleish, 1997).

Individuals in an anxious state frequently worry about the threat to a current goal and try to develop effective strategies to reduce anxiety to achieve the goal (Eysenck, Derakshan, Santos, & Calvo, 2007). Anxiety is of importance within the field of cognition and performance because it is often associated with adverse effects on the performance of cognitive tasks (Eysenck, 1992). This section overviews four theories relevant to the thesis project: Lazarus and Folkman's theory of stress and coping; Steele and Aronson's stereotype threat theory, Ashcraft and Kirk's theory of mathematics anxiety, and Eysenck and Calvo's processing efficiency theory.

At an individual level of investigation, Lazarus and Folkman (1984) presented a theory of stress and coping based on the central tenet that both stressors and coping ability must be understood from the perspective of each individual's appraisal of the environment. In response to an environmental event, primary appraisal occurs when individuals assess whether the event will be impactful or stressful, and secondary appraisal occurs when individuals assess whether or not they have the resources to cope with the event.

In recent stress studies, this appraisal concept has been formulated into a ratio of primary to secondary appraisal to reflect threat and challenge appraisals (Blascovich, Mendes, Solomon, & Hunter, 1999; Ritter, Bennett, Kase, Rodrigues, & Klein, submitted; Tomaka, Blascovich, Kelsey, & Leitten, 1993; Tomaka, Blascovich, Kibler, & Ernst, 1997). In some cases primary and secondary appraisals are referred to as demand and resource appraisals (Blascovich & Mendes, 2000). Demand appraisals are postulated to derive from cognitive assessments of potential harm and required future effort with affective and perceptual cues informing this assessment. Similarly, resource appraisals are cognitive assessments of putative available resources that are influenced by affective and perceptual factors. In the thesis project, post-task appraisal scores were used to group subjects into those who were 'threatened' by the task (i.e., anxious, have no control over the task, appear to have less coping resources than the task appears to demand), and those who

were ‘challenged’ by the task (i.e., interested in the task, have control over their performance on the task, appear to have as many coping resources as the task appears to demand).

At the group level of analysis, another stream of widely studied threat research is stereotype threat. Although the seminal article on this subject (Steele & Aronson, 1995) attempts to provide an explanation for the finding that Blacks tend to underperform on standardized tests as compared to Whites, the effects of stereotype threat have been extended to account for a wide variety of group performance decrements in a wide variety of performance situations (e.g., verbal tests, complex mathematical tasks, tests of memory, and mental rotation).

As Steele (1997) argued, the *threat* of the stereotype threat is in the performance situation itself. Stereotype threat is thought to be triggered by situations that pose a significant threat to self-integrity, the sense of oneself as a coherent and valued entity that is adaptable to the environment (Steele, 1988). This self-integrity threat stems from a state of cognitive imbalance in which one’s concept of self and expectation for success conflict with primed social stereotypes suggesting poor performance (e.g., White men perform more poorly on a math test when they are told their performance will be compared with that of Asian men, Aronson et al., 1999). The state of imbalance acts as an acute stressor that sets in motion physiological manifestations of stress, cognitive monitoring and interpretative processes, affective responses, and efforts to cope with these aversive experiences (Major & O'Brien, 2005; Steele, 1988). The extra situational burden is thought to interfere with the ability to perform as well on the task as might otherwise be possible.

Schmader, Johns and Forbes (2008) assert that increased monitoring, paired with increased physiological arousal and a primed state of cognitive imbalance created by stereotype threat, can lead individuals to appraise their experience in a biased manner that produces negative thoughts and feelings. These stereotype threat effects can degrade the ability to regulate attention during complex tasks where it is necessary to coordinate information processing online and inhibit thoughts, feelings, and behaviors counterproductive to one’s current goals. Cognitive

psychologists describe the mechanism that is responsible for this sort of efficient regulation as executive functioning, a component of working memory (Engle, 2002).

Ashcraft and Kirk (2001) conducted integrative research on the relationships among anxiety, working memory, and math performance. They considered whether math anxiety had any online effect on an individual's math performance, that is, an effect on underlying cognitive processes as the individual performs a math task. Across several of their initial studies (Ashcraft & Faust, 1994; Ashcraft, Kirk, & Hopko, 1998) they found substantial evidence for performance differences as a function of math anxiety. These performance differences were observed in more difficult math problems such as two-column addition problems involving carry operations. High-math-anxiety subjects showed slower, more effortful processing on procedural aspects of performance, such as performing the carry operation. Additionally, the high-math-anxiety subjects exhibited greater error rates on the more complex problems often showing classic speed-accuracy tradeoffs indicating a willingness to sacrifice accuracy to speed the experimental session along.

Ashcraft and Kirk (2001) hypothesized that a major contributor to the performance deficits found for high-math-anxiety participants involves working memory. In particular, such deficits were predicted to stem from that portion of working memory, the central executive, that applies the various procedures of arithmetic during problem solving (Butterworth, Cipolotti, & Warrington, 1996; Darke, 1988; Sorg & Whitney, 1992).

In more generalized theory not specific to only mathematics tasks, Eysenck and Calvo (1992) proposed an overall model of the anxiety-to-performance relationship in cognitive tasks called the *processing efficiency theory*. This theory assumes that worry is the component of state anxiety responsible for effects of anxiety on performance effectiveness and efficiency. Effectiveness refers to the quality of task performance indexed by standard behavioral measures (generally, response accuracy). Efficiency refers to the relationship between the effectiveness of

performance and the effort or resources spent in task performance, with efficiency decreasing as more resources are invested to attain a given performance level (Eysenck, Derakshan, Santos, & Calvo, 2007).

Worry or self-preoccupation is characterized by concerns over evaluation and failure and expectations of aversive consequences (Borkovec, 1994). Worry is activated in stressful situations (e.g., tests, evaluations, competitive conditions) and has two primary effects. One effect involves cognitive interference by preempting the processing and temporary storage capacity of working memory. The worrisome thoughts consume the limited attentional resources of working memory which are therefore less available for concurrent task processing (Eysenck & Calvo, 1992). The other effect involves increased motivation to minimize the aversive anxiety state. This function is accomplished by promoting enhanced effort and use of auxiliary processing resources and strategies, thus, compensating for the potential performance impairments caused by the preemption of working memory resources. If these resources are unavailable, then performance effectiveness will be impaired (Eysenck & Calvo, 1992).

The processing efficiency theory is based on the triparitite working memory model (Baddeley, 1986) that is discussed in more detail in the next section. According to Baddeley's model the limited capacity working memory system consists of a modality-free central executive involved in the processing of information and having self-regulatory functions (e.g., performance monitoring, planning, and strategy selection); a phonological loop for the rehearsal and transient storage of verbal information; and a visuo-spatial sketch pad for the processing and transient storage of visual and spatial information.

Eysenck and Calvo's (1992) theory claims that the main effects of worry (and, more generally, of anxiety) are on the central executive. Accordingly, adverse effects of anxiety on performance and efficiency should be greater on tasks imposing substantial demands on the processing and storage capacity of working memory (especially the central executive).

Worrisome thoughts interfere with this processing and storage function, and there is an additional burden on the self-regulatory mechanism inhibiting such thoughts and producing auxiliary processing activities. Detrimental effects of anxiety are also expected on the phonological loop rather than on the visuo-spatial sketchpad because worry typically involves inner verbal activity rather than imagery representations (Rapee, 1993).

Working Memory

Fluent performance on complex cognitive tasks, such as the mental serial subtraction task used in the thesis project, relies on the ability to coordinate and integrate stored information with ongoing processes. This requires efficient organization and maintenance of intermediate results that can be accessed for use at the appropriate time. Working memory resources would be required during mental arithmetic tasks where one must hold the original problem and any intermediate results in memory while performing carry, borrow, and place-keeping operations in working toward the final answer. As discussed in the previous section, stress or anxiety, or a threatening task appraisal can reduce working memory capacity with intrusive thoughts of poor performance or failure.

For the past 100 years, research on working memory (or short-term memory as it used to be called) has focused on the storage of information for retrieval after a brief interval. A related function attributed to short-term memory is its role as a stepping stone on the path to long-term memory while information is being memorized through rehearsal or elaboration. A somewhat more modern view of working memory takes into account not just the storage of items for later retrieval, but also the storage of partial results in complex sequential computations. The most recent conceptions of working memory extend its function beyond storage to encompass the actual computations themselves. These computations are symbolic manipulations at the heart of

human thinking including comparison, retrieval, and logical and numerical operations (Just & Carpenter, 1992).

Working memory was first considered from a computational perspective by Baddeley and Hitch (1974), who examined tasks comparing the storage and processing aspects of comprehension. Working memory provides the resources needed to retrieve and maintain information during a wide variety of cognitive processes (Baddeley, 1986, 1990; Miyake & Shah, 1999). Given its ubiquitous influence, the study of working memory is critical to understanding how people perform cognitive tasks (Daily, Lovett, & Reder, 2001). Working memory resources are limited, and these limits govern performance on tasks that require those resources. Prior research has demonstrated that as the working memory demands of a task increase, an individual's performance on the task decreases (e.g., Anderson, Reder, & Lebiere, 1996; Just & Carpenter, 1992; Salthouse, 1992; Sweller, 1988). In summary, it appears that working memory resources are drawn upon in the performance of cognitive tasks, are inherently limited, and differ across individuals (Daily, Lovett, & Reder, 2001).

The most prominent working memory model comes from Baddeley (1986, 1992). Baddeley's model details the components and processes of working memory. In the model working memory is composed of three separate units. The primary unit is called the *executive control system*. The job of this system is to initiate and control ongoing processes. Some of its activities are reasoning, language comprehension, and information transfer from long-term memory via rehearsal and chunking, and retrieval of information. The second component is the *articulatory* loop where speech and sound-related information are rehearsed. The third unit, called the *visuo-spatial sketchpad,* is specialized for the processing of visual information. Figure 2.2 shows the components of Baddeley's (1986) working memory model.

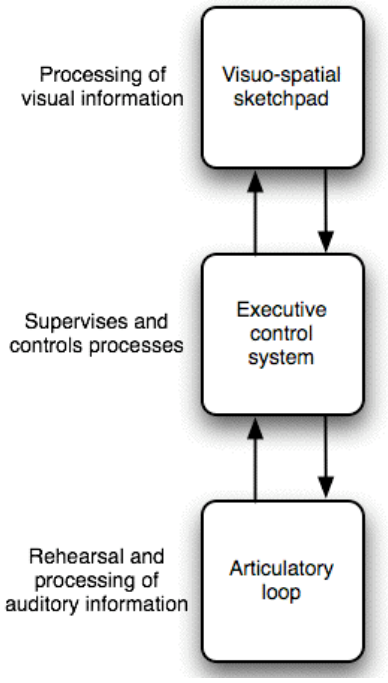

Figure 2.2: The three systems of Baddeley's (1986) working memory model.

Both the loop and the sketchpad are 'slave systems' to the central executive. They carry out processes such as rehearsal and image formation. Other processes, such as reasoning and language, are the responsibility of the executive. The two slave systems are domain-specific: one devoted to the processing of acoustic information only, the other to the processing of spatial information only. Tests of working memory have shown that each slave system relies on its own pool of attentional resources (Baddeley & Hitch, 1974). If the tasks that one of these systems is performing are simple and do not require much in the way of attentional resources, then working memory performance is unchanged. If one of the systems is given a demanding task, it either fails to complete the task or draws on the attentional resources of the executive, which results in impaired working memory performance.

Cognitive architectures generally include a theory of memory, including working memory. The advantage of computational approaches to memory is that they present a theory of working memory in a rigorous enough form to enable quantitative comparisons with human data

(Daily, Lovett, & Reder, 2001). For example, the ACT-R cognitive architecture offers a set of basic mechanisms that constrain how working memory can be modeled. Anderson (1983, 1990) proposed a global model of memory function in an earlier version of ACT-R, called ACT*, that was similar to Atkinson and Shiffrin's (1971) modal model of memory. The modal model of memory was the first model to provide a general overview of how information is processed in each of the different memory types (short-term, long-term, and sensory). Figure 2.3 shows Atkinson and Shiffrin's model compared to Anderson's ACT* memory model. Both models contain components representing different types of memory stores. The arrows indicate different kinds of processing that occur as information is exchanged between the memory systems.

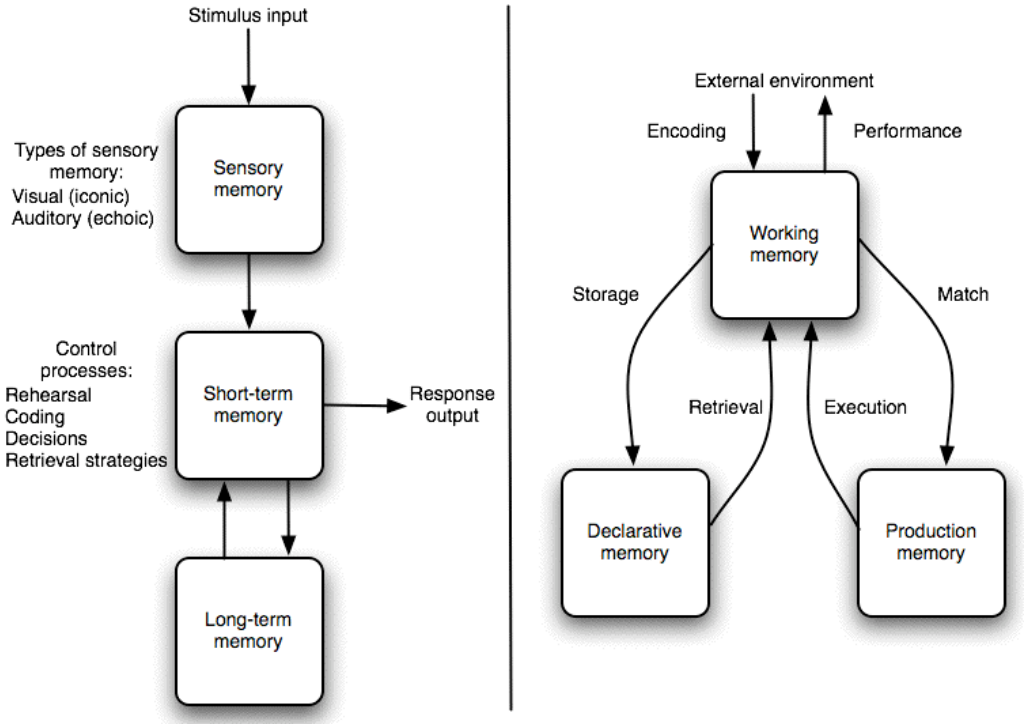

Figure 2.3: Atkinson and Shiffrin (1968) modal memory model (left), and Anderson's (1983) ACT* memory model (right).

The strength of the ACT* memory model is in its description of separate processing loops for declarative and production memories, i.e., there are no connecting arrows. Declarative and production memories are considered two distinct types of long-term memory. Procedural memory holds procedural knowledge that is memory for skill, demonstrated by doing, and arising without conscious recall. Declarative memory contains declarative knowledge. It is memory for facts and events that is demonstrated by speaking and arising with conscious recall. There are two types of declarative memory: semantic memory, the knowledge of facts and general learned knowledge, and episodic memory containing episodes or personally experienced events.

Anderson (1983) proposed an additional component to the ACT* memory model that represented how propositional information in declarative memory was organized. The organization formed a network in which nodes represented concepts and links represented the relationship between them. This allowed for different types of nodes to stand for specific examples of concepts an individual encounters. In this way, the network represented both episodic as well as semantic information, both part of declarative memory.

In recent ACT-R cognitive modeling research, working memory is primarily defined in two ways: as a subset of highly active elements of declarative memory; or as a process of spreading source activation (i.e., attentional resource) from the current goal to declarative elements strongly linked with that goal. ACT-R's process of source activation has been used in several studies to model individual differences in working memory capacity (Daily, Lovett, & Reder, 1999, 2001; Lovett, Daily, & Reder, 2000; Lovett, Reder, & Lebiere, 1997; Rehling, Lovett, Lebiere, Reder, & Demiral, 2004).

Modeling Individual Differences

For many purposes, it is desirable to be able to model or predict individual performance. The thesis project takes a quantitative (or parametric) approach in accounting for individual differences. Quantitative models mimic individual differences in performance only with parameter manipulation, while the model's structure remains unchanged. This is not to be mistaken for the qualitative approach where families of models account for qualitative differences in cognitive mechanisms resulting in differences of processing among individuals.

Research in cognitive modeling, when it seeks validation in the performance of human subjects, almost unanimously is concerned with the average performance of many subjects. The pitfalls associated with averaging over subjects' performance have been known for a long time (Newell & Simon, 1972; Siegler, 1987) but are relatively rarely dealt with satisfactorily in the field. It has been noted that data averaged over individuals may not accurately reflect the behavior of any one person. The same point has been made for data averaged over a task, where subjects may use different strategies (Gobet & Ritter, 2000).

Some researchers have pointed out that models should be able to account for individual data (Daily, Lovett, & Reder, 2001; Gobet & Ritter, 2000; Lewandowsky & Heit, 2006). As most models are fit to aggregate data, they do not reflect individual differences in cognition that could be identified in any cognitive domain. Accounting for different cognitive strategies or for variance in cognitive latent variables (e.g., working memory capacity) prevents losing important data and may provide more information and insight into cognitive mechanisms. It has been argued that the ability of a model to include individual differences constitutes a valuable test of fit (Chuderski, Stettner, & Orzechowski, 2007).

The ACT-R architecture contains a number of parameters that can be used to fix levels of performance to realistic levels. In parametric modeling, the ACT-R community has by custom

sought universal values for these parameters wherever possible, optimizing how well a model fits the data of the average subject. Each of these parameters has a meaning associated with the subsymbolic processes within the architecture, and determines the model's behavior in one specific way. For example, a parameter called *W* determines the sum of the activations of all the pieces of information that may be retrieved at any point in time. Therefore, *W* could be said to control the model's working memory capacity. In ACT-R when the model was meant to predict an average subject, a *W* value of 1.0 produced very good fits to aggregate subject data. Lovett, Reder and Lebiere (1997) found that the *W* parameter could be meaningfully varied to model individual differences in working memory capacity in a digit working memory task.

Work by Daily, Lovett and Reder (2001) is another example of parametric modeling of individual differences using *W*. They constructed a model of working memory functioning in a digit span task (called MODS). The purpose of the research was to model working memory performance on an individual subject level with manipulation of only the *W* parameter influencing activation supply. Values of the *W* parameter for each of 29 subjects were used to control the amount of activation spread from a goal to memory in the model, where each value presented working memory capacity for each individual. Three additional global parameters were held constant across the subjects. The model accurately predicted each subject's performance in four task conditions ($R^2 = 0.92$). Individuals who recalled digits more correctly had higher estimated *W* values. When the *W* parameter was not adjusted individually R^2 dropped to 0.66.

In another study by the same researchers (Lovett, Daily, & Reder, 2000), the *W* value estimates for working memory capacity were tested further. The study examined if the *W* values from individuals in the MODS task could be generalized and used to predict performance in another task (n-back), even though both tasks differed to a great extent. Their model's cross-task predictions were relatively successful ($R^2 = 0.79$). This study also tried to generalize two parameters (*W* value and an additional parameter reflecting both perceptual and motor speed) in a

simplified air traffic control task that required observing simulated aircraft positions on a simulated radar screen. This is one of the few cross-task modeling studies attempting predictions of individual differences in complex tasks.

Modeling individual data has the advantage of producing fine-grained predictions. And as mentioned previously, simulating the behavior of each subject is a powerful test of the model. A disadvantage to modeling individual data is that modeling an individual's performance includes also modeling the noise (Chuderski, Stettner, & Orzechowski, 2007). Therefore, an individually fitted model may provide a weaker explanation of the phenomenon being modeled. Another disadvantage is that modeling individual data usually requires additional parameters. For example, a parametric model of a color classification task by Nosofsky and Palmeri (1997) required setting values of six parameters for each of three participants. It was been pointed out that *overparametrized* models are less capable of being generalized to new situations (Pitt, Myung, & Zhang, 2002).

Another quantitative approach, as an alternative to modeling an average subject, or modeling an individual subject, is modeling individual differences among groups of individuals whose cognitive processing is similar enough to be grouped together. Lee and Webb (2005) proposed that the optimal strategy for parametric modeling is to build families of models that differ in only parameterization, where each model from the family represents aggregated performance of several subjects whose behavior looks similar. As an example, Just and Carpenter (1992) simulated different language performance of low and high working memory span subjects by setting low and high limits on the total amount of activation that is available for productions within the model. This type of approach reduces noise associated with individual data but still provides a way of explaining differences in cognition among individuals.

In cognitive modeling, whether modeling individual differences, group differences, or an average subject, as the number of parameters manipulated by the model increases so does the

time and computational resources required to search the parametric space for values that produce the best model predictions. This optimization process (i.e., agreement between model and human data via optimal parameter values) is at the heart of the thesis project.

Relevant Optimization Methods

A cognitive model, built and run within a cognitive architecture, predicts human performance on a specific task. Part of the validation process in cognitive modeling is comparing the model's predictions of human performance to actual human performance. If the model is structurally correct in simulating how a task is procedurally preformed by humans (from an information processing perspective), then the manipulation of architectural parameters can be used to simulate realistic effects on the cognitive performance of the task. For example, different sets of parameters and values can represent external conditions (e.g., conditions of the task environment, loud noise, human interruptions, visual distractions, or a watching audience), or internal conditions pertaining to psychological or cognitive states of the humans performing the task (e.g., stressed, fatigued, frightened, or anxious).

In the field of cognitive science the process of manipulating architectural parameters to match the model's predictions to the human performance is called *fitting the model*. Finding settings of architectural parameters that produce the desired effect in the model's performance is a combination of theoretical formulation and optimization. Selecting appropriate parameters to investigate out of the many offered by the architecture (e.g., ACT-R has approximately 80 parameters) is guided by the task type, hypothesis or theory, and past cognitive modeling research. Once specific parameters are selected, value constraints (minimum and maximum boundaries) for each parameter can define a parametric search space. The parameters act as input to the cognitive model producing performance predictions as output.

The thesis project implemented two search algorithms, a parallel genetic algorithm (a global search method, Cantu-Paz, 2001) and a hill-climbing algorithm (a local search method, Press, Teukolsky, Vetterling, & Flannery, 1992), for the purpose of fitting the serial subtraction model to the human performance data.

One difficulty in fitting a model to human data is determining if a given minimum fit is the best (global) minimum or a suboptimal (local) minimum. Fitness landscapes of the ACT-R parameter space have been relatively uninvestigated, therefore, their complexity is unknown. Genetic algorithms in general have been reported to be successful over a growing range of difficult problems from many different disciplines (Coley, 1999; Goldberg, 1989; Haupt & Haupt, 2004). Much of this proven utility arises from the robustness of the algorithm and in the way the population (i.e., genotypes, parameter sets) navigates its way around complex search spaces so as to avoid entrapment by local optima.

Additionally, an earlier and very preliminary study (Tor & Ritter, 2004) ran a serial genetic algorithm on a single CPU machine for approximately four generations in an attempt to optimize a cognitive model of a Tower of Nottingham Task. The study, although limited in nature, showed that using a genetic algorithm for model fitting was a viable option if more computational resources were applied.

A hill-climbing algorithm was implemented for the sake of comparison between a local search method and a global search method. It was speculated that the ACT-R parametric landscape would be a varied or fragmented terrain. With this type of landscape the hill-climbing algorithm might get stuck in a local minima and be unable to find a good fit for the model.

Genetic and Parallel Genetic Algorithms

Genetic algorithms (GAs) are search methods based on the mechanics of natural selection and genetics. They combine survival of the fittest among string structures with a structured yet randomized information exchange to form a search algorithm that is well suited for optimizing solutions over large combinatorial spaces. In every generation of a GA's search, a new set of strings is created using bits of the fittest of the old strings combined with occasional new bits. While randomized, GAs are no simple random walk. They efficiently exploit historical information to speculate on new search points with expected improved performance (Goldberg, 1989).

GAs were developed by John Holland and his colleagues at University of Michigan. The primary monograph on the topic is Holland's (1975) *Adaptation in Natural and Artificial Systems*. The goals of their research were to abstract and rigorously explain the adaptive processes of natural systems, and to design artificial systems that retained the important mechanisms of natural systems (Goldberg, 1989). The central theme of research on GAs has been *robustness*. Many papers establish the validity of the technique in function optimization and control applications. Having been established as a valid approach to problems requiring efficient and effective search, GAs are finding more widespread application in business, scientific, and engineering circles (Cantu-Paz, 2001; Coley, 1999; Goldberg, 1994; Haupt & Haupt, 2004; Mitchell, 1997).

Because GAs are rooted in both natural genetics and computer science, the terminology used in GA literature is a mix of the natural and the artificial. Table 2.1 summarizes the most commonly used terms from natural systems used in this area. *Strings* of artificial genetic systems are analogous to *chromosomes* in biological systems. In natural systems, one or more chromosomes combine to form a total genetic prescription called a *genotype*. In artificial genetic

systems the total package of strings is called a *structure*. The structures decode to form a particular *parameter set*, *solution alternative*, or *point* in the solution space. In natural terminology chromosomes are composed of *genes*, which take on some number of values called *alleles*. The position of the gene (its *locus*) is identified separately from the gene's function. In computational terminology, a particular character (a gene) and its position (its locus)—the position of a bit in a string—has a determined meaning (how it decodes) uniformly throughout a population.

Table 2.1: GA nomenclature (taken from "Introduction on Evolutionary Algorithms").

Term	Definition
Genotype	The code devised to represent the parameters of the problem in the form of a string.
Chromosome	One encoded string of parameters (binary, float, etc…).
Individual	One or more chromosomes with an associated fitness value.
Gene	Encoded version of a parameter of the problem being solved.
Allele	Value that a gene can assume (binary, integer, etc…).
Locus	Position that the gene occupies in the chromosome.
Phenotype	Problem version of the genotype (algorithm version) suited for being evaluated.
Fitness	Real value indicating the quality of an individual as a solution to the problem.
Environment	The problem. This is represented as a function indicating the suitability of phenotypes.
Population	Set of individuals with their associated statistics.
Selection	Policy for selecting one individual from the population (selection of the fittest).
Crossover	Operation that merges the genotypes of two selected parents to yield two new children.
Mutation	Operation that spontaneously changes one or more alleles of the genotype.

GAs operate on populations of strings with the string coded to represent an underlying parameter set. Selection, crossover, and mutation operations are applied to successive string populations to create new string populations. These operators perform simple actions such as random number generation, string copying, and partial string exchange.

The *selection* operation attempts to apply pressure upon the population in a manner similar to that of natural selection found in biological systems. Poorer performing genotypes are weeded out more often and better performing, or *fitter*, genotypes have a greater chance of propagating the information they contain within the next generation. An objective or fitness function weeds out poorly performing genotypes. Intuitively, fitness is thought of as some measure of profit, utility or goodness to maximize. In a cognitive modeling optimization problem fitness is in terms of cost or error (i.e., how well model predictions fit the data). Error is something to be minimized; therefore, the optimization becomes a minimization. In this case, genotypes with lower fitness values are considered good performers and remain in the population while genotypes with high fitness values are killed off.

Crossover allows genotypes to exchange information in a way similar to that used by a natural organism reproducing. One method (called single point crossover) chooses pairs of genotypes promoted by the selection operator, randomly chooses a single locus within the binary strings and swaps all the information (bits) to the right of this locus between the two genotypes.

Mutation is used infrequently to randomly change (flip) the value of single bits within individual strings. Traditionally, mutation has been seen as very much a secondary mechanism in comparison to crossover (Coley, 1999). However, there are growing indications among some researchers that mutation may have a more central role to play (Hinterding, Gielewski, & Peachey, 1995; Muhlenbein, 1992).

After selection, crossover, and mutation have been applied to the initial (randomly generated) population, a new population will have been formed and the generational counter is incremented. This process of selection, crossover and mutation is continued until a fixed number of generations have elapsed or a convergence criterion has been met.

There are many implementation approaches to decide on before applying a GA to a particular problem: (1) the method of encoding the parameters (binary or real values); (2) what

variety of selection and crossover operators to use; (3) the population size; (4) how to apply mutation; and (5) the termination condition (Coley, 1999).

The population maintained by the GA consists of tentative solutions (encoded strings); this is unlike other optimization techniques that offer only a single solution. In many practical applications, GAs find good solutions in reasonable amounts of time. However, in some cases, GAs can require hundreds of thousands of expensive function evaluations, and depending on the cost of each evaluation, the GA may take days, months, or even years to find an acceptable solution (Cantu-Paz, 2001). These large time and computational demands emphasize the need for more efficient versions of GAs.

There have been multiple efforts to make GAs more efficient, and one of the most promising alternatives is to use parallel implementations. The parallel nature of genetic algorithms has been recognized for a long time, and many researchers have successfully used parallel GAs to reduce the time required to reach acceptable solutions to complex problems. GAs, working with a population of independent solutions can easily distribute the computational load among several processors. However, despite this operational simplicity, parallel GAs are complex nonlinear algorithms that are controlled by many parameters that affect their efficiency and the quality of their search (Cantu-Paz, 2001). Setting those parameters to the most appropriate values is crucial to obtaining good solutions quickly and reliably.

Parallel GAs (PGAs) have the same advantages as serial GAs: using representations of the problem parameters (not the parameters themselves), robustness, easy customization for new problems, and multiple-solution capabilities. In addition, PGAs are usually faster, less prone to finding only sub-optimal solutions, and capable of cooperating with other parallel search techniques (Cantu-Paz, 2001). Table 2.2 summarized the advantages of using PGAs.

Table 2.2: Advantages of using PGAs (taken from "Introduction on Evolutionary Algorithms").

1. Works on coding of the problem (least restrictive)
2. Basically independent of the problem (robustness)
3. Uses objective function, not derivatives or other auxiliary knowledge
4. Can yield alternative solutions to the problem
5. Parallel search from multiple points in space
6. Easy parallelization as islands or neighborhoods
7. Better search, even if no parallel hardware is used
8. Easy cooperation with other search procedures

Classes of PGAs are distinguished by different types of parallelization (Figure 2.4). The most common type, *master-slave global parallelization* (sometimes called single-population), proceeds in a similar manner to the serial GA, but at a much faster pace. The other types of PGAs fit into two general classes depending on their grain of parallelism, called *coarse-grained* (sometimes called distributed or multiple-population), or *fine-grained* (sometimes called diffusion-model or cellular). Coarse-grained parallelism is typified by long computations consisting of a large number of instructions between communication synchronization points resulting in a high computation-to-communication ratio. In fine-grained parallelism, a small number of instructions are executed between communication cycles resulting in a low computation-to-communication ratio.

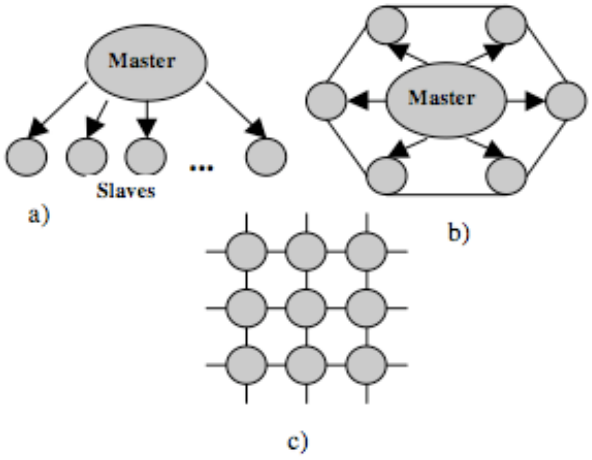

Figure 2.4: Different types of PGAs: (a) master-slave or global parallelization, (b) multiple-population or coarse-grain, and (c) cellular or fine-grain (DeToro, Ortega, Fernandez, & Diaz, 2002).

Master-slave GAs use a single population. One master node executes the GA (selection, crossover, and mutation), and the evaluation of fitness is distributed among numerous slave processors. The slaves evaluate the fitness of the individuals that they receive from the master and return the results. With this type of parallel GA the selection, crossover, and mutation functions consider the entire population, thus the label 'global'.

The second type of PGA, multiple-population, involves several subpopulations that exchange individuals occasionally. This exchange of individuals is called migration and is controlled by several parameters. This type of GA is difficult to control because the effects of migration are not well understood. Researchers using this type of algorithm must determine the number and size of the sub-populations, the frequency of migration, the number and destination of migrants, and the method used to select which individuals migrate (Cantu-Paz, 2001).

The third type of PGA, cellular, consists of a single spatially-structured population. The population structure is usually a two-dimensional rectangular grid, and there is one individual per grid point. Ideally, there is one processor per individual, so the evaluation of fitness is performed simultaneously for all the individuals. Selection and crossover are restricted to a small

neighborhood around each individual. This type of PGA has not been studied as much as the previous two types.

For the thesis project, a master-slave type of PGA was selected to optimize the serial subtraction cognitive model for several reasons. First, it made sense to treat the ACT-R parameter space as a global population and to apply the genetic operators to all genotypes in the population. The PGA should be seeking a global minimum in the ACT-R parameter space as a whole. Secondly, the master-slave involves fewer population variables thus reducing computational complexity and making the algorithm easier to control. Lastly, it was desirable to keep the level of communication at a minimum. Passing sets of parameters to the processors and then collecting up the fitness values could be preformed using the least amount of communication if the whole population could be situated and manipulated by the master processor.

Hill-climbing and Hybrid Algorithms

Hill climbing or gradient ascent/decent algorithms are part of the family of local search techniques. Local search is a widely used, general approach for solving hard combinatorial search problems (Hoos & Stutzle, 2000). The general idea is to examine the search space induced by a given problem instance for solutions, initialize the search, and then from there iteratively move from one search space position to another where the decision on each step is based on information about the local neighborhood only. Local search algorithms make use of an objective function mapping each search space position onto a real or integer number in such a way that an optima of the objective function corresponds to a solution.

The name ‘hill climbing’ comes from visualizing a space of all possible solutions to a given maximization problem as a three-dimensional contour landscape. A given set of coordinates on that landscape represents one particular solution. Those solutions that are better

are higher in altitude, forming hills and peaks; those that are worse are lower in altitude, forming valleys. A hill climber is an algorithm that starts out at a given point on the landscape and moves uphill. Hill climbing is also known as a *greedy algorithm*, meaning it always makes the best choice available at each step in the hope that the overall best result can be achieved that way. In contrast, search methods such as GAs are not greedy; they sometimes make suboptimal choices in the hopes that they will lead to better solutions later on.

Hill climbing is more systematic and less random than GAs. A hill-climbing algorithm begins with one initial solution to the problem (i.e., the result of an objective function applied to a string), usually chosen at random. The string is then systematically mutated (i.e., stepped increment or decrement), and if the mutation results in higher fitness for the new solution than for the previous one, the new solution is kept; otherwise, the current solution is retained. The algorithm is then repeated until no mutation (i.e., step) can be found that causes an increase in the current solution's fitness, and this solution is returned as the result (Koza, Martin, & Streeter, 2003).

A disadvantage of hill climbing is that it will generally find only local optima (maxima or minima). Unless the search space is convex or concave, it will not necessarily find a global optima. Ridges and plateaus in search spaces are also problems for hill climbing. A ridge is a curve in the search space that leads to a maximum, but the orientation of the ridge compared to the available moves that are used to climb is such that each move will lead to a smaller point—each point on the ridge looks like a local maxima to the algorithm. A plateau occurs at a flat part of the search space. This kind of flat surface can cause the algorithm to terminate or wander aimlessly with solutions all being very close together in value.

Every optimization method represents a particular compromise. As far as a standard GA is concerned, its attributes are reported as robustness and the ability to rough out a problem reliably by finding the most promising regions in a large complex search space. Unfortunately,

GAs suffer from inefficiencies, characterized by slow convergence and lack of accuracy when a high-quality solution is required. In contrast, hill-climbing methods focus solely on precision and computational time at the expense of reliability (returning local optima as a solution). Several studies have hybridized genetic algorithms with hill-climbing methods for more efficient and accurate global optimization (Cooper, Corne, & Crabbe, 2003; Renders & Bersini, 1994).

Renders and Bersini (1994) used both the exploration capabilities, parallelism and combination properties of a GA, and the local exploitation power of hill climbing to develop two different types of hybridizations. They used 'global exploitation' for the sampling of areas of the search space that were likely to be of high fitness according to the information present in the population. Then 'local exploitation' extracted the most information possible at a local level and 'climbed the hill' towards the local optimum. Global exploitation finds the most promising basins of fitness, while the local exploitation tries to discover the optimum in the basin.

In the field of computational biology, Cooper, Corne and Crabbe (2003) developed a hill-climbing-genetic algorithm for simulation of protein folding. The hill climbing was found to improve performance of the GA by incorporating an optional search interval allowing local zonal search. The hybrid also executed 20% to 50% faster than other similar protein folding simulations using a variety of test proteins.

In this thesis, once the hill-climbing optimization approach was confirmed to be inferior to the PGA for model fitting, a hybrid was constructed in similar format to the global and local exploitation method described above. The master-slave implementation of the PGA was basically kept intact; an additional focused search around each genotype was conducted at the multiprocessor level by the hill-climbing algorithm.

Components and Concept Integration

The optimization approach presented in the thesis is composed of the following three components. In the first component, observations and appraisal data were collected from human subjects performing a mental serial subtraction task known to elicit a reliable stress response in a laboratory setting. This component involves theories of cognitive performance such as the effects of stress and anxiety on working memory capacity. Working memory is thought to be a required resource for performing mental arithmetic tasks.

The second component is a cognitive model of the serial subtraction task. Cognitive models simulate human performance on a specific task. They are developed and run in a simulation type of environment called a cognitive architecture that incorporates theories of cognitive psychology. In this case the model of the serial subtraction task was developed in the ACT-R cognitive architecture and modified for a parallel processing platform.

In the third component, parallel processing versions of a genetic algorithm and a hill-climbing hybrid algorithm were developed to optimize the serial subtraction model while running in the ACT-R cognitive architecture. The thesis developed this novel optimization approach as a means to efficiently and accurately fit the serial subtraction model to the human performance data. The optimization was focused on fitting the model to individual subject performance and to individual subjects grouped by task appraisal. It is hoped that the improved efficiency, accuracy, and nonbiasness of this optimization approach can assist in understanding how stress influences performance on mental arithmetic tasks.

The next chapter describes the development of the PGA, hill-climbing algorithm, and a hill-climbing-genetic hybrid that incorporates the ACT-R cognitive architecture and the serial subtraction model. Stochasticity is a component of every subsymbolic mechanism in the ACT-R architecture, including activation, utility, and latency computations, which in turn determine

every cognitive step such as production rule firing and memory retrieval. This stochasticity caused difficulties in development of the search algorithms when integrated with the ACT-R architecture.

Chapter 3

PROJECT DEVELOPMENT

This chapter provides a development perspective that sets the stage for the thesis beginning with a description of two background topics: (1) a human subject experiment with collection of cognitive performance and task appraisal data and (2) a cognitive model of a serial subtraction task developed over a series of research studies from observations of human subject performance.

A cognitive model is thought to be *useful* if it can predict human performance on a specific task. The model-to-human data fitting process is a stochastic global optimization problem. The second portion of this chapter describes the development of three optimization approaches to the model-to-human data fitting problem based on a parallel genetic algorithm, a hill-climbing algorithm, and a hill-climbing-genetic hybrid algorithm.

Background

This section describes details of the human subject experiment where performance and task appraisal data were collected and ultimately used to fit the serial subtraction model. Observations from the human subject experiment informed modifications to the structure of the existing serial subtraction model, as well as, guided the interpretation of the resulting model fits.

Human subject data

A cognitive model is validated with human performance. In the thesis, the serial subtraction model was validated with human subject data collected as part of a larger project to study the effects of stress and caffeine on biomarkers of cardiovascular health.

Dr. Laura Klein of the Biobehavioral Health Department at Penn State University performed a mixed experimental design to study the effects of caffeine and stress on cognitive performance and cortisol production in 45 healthy men 18-30 years of age (Klein, Whetzel, Bennett, Ritter, & Granger, 2006; Whetzel, Ritter, & Klein, 2006). The study was approved by the Pennsylvania State University Office for Research Protections Biomedical Institutional Review Board IRB#14948. The subjects were screened as daily caffeine consumers, not using tobacco or nicotine products, not taking over-the-counter or prescription medications, and not having health conditions that would affect the dependent measures.

Subjects were asked to perform a simple reaction time and a working memory task taking 15 minutes to complete. Then subjects were administered one of three doses of caffeine: none (placebo), 200 mg caffeine (equivalent to 1-2, 8 oz cups of coffee), or 400 mg caffeine (equivalent to 3-4, 8 oz cups of coffee). After allowing absorption time post treatment, a 20-minute stress session consisting of the mental arithmetic portion of the Trier Social Stress Task (TSST) was performed. Following completion of this stress session, subjects again were asked to complete the reaction time and working memory tasks. Cognitive performance was determined by calculating accuracy and speed scores.

Several psychosocial questionnaires were used to obtain state and trait measures of the subject's mental health, preference for math and computers, and mood assessments. The questionnaires were used twice during the protocol—before administering the caffeine and

following the stressor arithmetic task. Additionally, several physiological measures were taken throughout the study such as saliva samples, blood pressure, and heart rate.

In the experiment, the stress session included the mental arithmetic portion of the TSST—the serial subtraction task. TSST traditionally consists of an anticipation period, 10 minutes in length, and a test period in which the subjects have to deliver a free speech and perform mental arithmetic in front of an audience, also taking 10 minutes. The TSST protocol has been used for investigating psychobiological stress responses in a laboratory setting since the 1960s (Kirschbaum, Pirke, & Hellhammer, 1993).

This serial subtraction task consisted of four 4-minute blocks of mentally subtracting by 7's and 13's from 4-digit starting numbers. Figure 3.1 illustrates the serial subtraction task. These were the four starting numbers used to begin the four blocks of subtraction during the experiment.

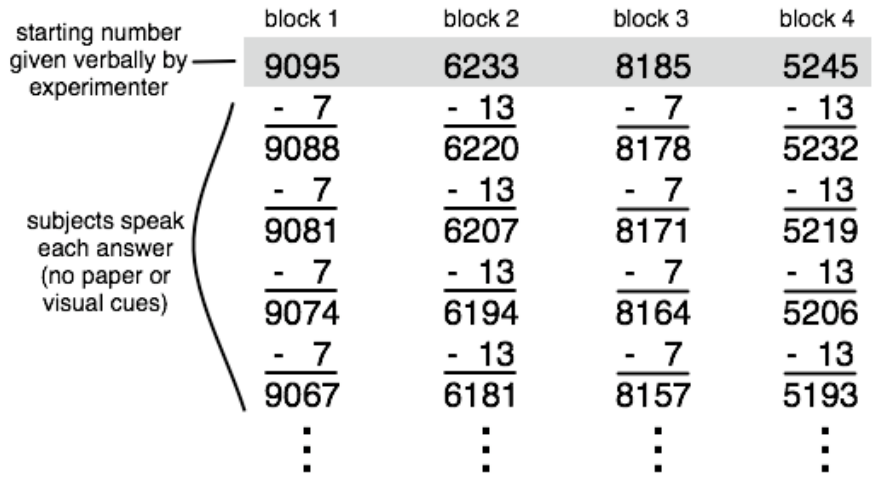

Figure 3.1: An illustration of the four blocks of the serial subtraction task as in the experiment; subjects perform the task mentally without paper or visual cues

The task is performed mentally with no visual or paper clues. An experimenter tells the subject the starting number; from then on, the subject speaks the answer to each subtraction problem. After the task is explained to the subject, a task appraisal questionnaire is completed,

and the subject begins performing the task. It is thought that this anticipation period, for some subjects, increases anxiety and worry about poor performance on the upcoming task.

Before the task begins the experimenter explains that the subject's performance is going to be voice recorded and reviewed by a panel of psychologists for comparison with the other subjects participating in the experiment. Subjects sit in a chair directly in front and near the experimenter who is holding a time keeping device and clipboard of the correct subtraction answers that she checks off as the subject performs the task. The experimenter is female, specifically Dr. Laura Klein; and the subjects are male. For some subjects, the thought of their performance being analyzed and compared to other subjects by a panel of experts, combined with performing in front of a woman scientist in a white lab coat, could trigger a stereotype threat effect. Before the task begins the experimenter emphasizes that the task should be preformed as quickly and as accurately as possible.

The subjects' subtraction answers were scored against a list of correct answers from the starting number. When an incorrect answer was given, the subject was told to "Start over at <the last correct number>". At two minutes into each 4-minute session, subjects were told that "two minutes remain, you need to go faster". This prompt was included to enhance the time-pressure component of the task. For each subject the number of subtraction problem attempts were recorded and a percentage correct score was calculated by dividing the total number of correct attempts by the total number of attempts for each block of the serial subtraction.

Before and after the serial subtraction stress session, subjects completed pre- and post-task appraisals based on Lazarus and Folkman's (1984) theory of stress and coping (discussed in Chapter 2). Each subject was asked five questions orally: two focused on the subject's resources or reserves to deal with the serial subtraction task and three focused on the subject's perception as to how stressful the task would be. After correcting for the imbalance in questions, a ratio of perceived stress to perceived resources was created. For example, if a subject's appraisal score

was 1.0 or less, their perceived stress was less than or equal to their perceived ability to cope, which equated to a *challenge condition*. If a subject's appraisal score was greater than 1.0, their perceived stress was greater than their perceived ability to cope, which equated to a *threaten condition*. The instrument used to measure task appraisal is given as an Appendix.

For the thesis project the serial subtraction performance data from the subjects in the control group (no caffeine) were analyzed at three different levels: average across subjects, average by appraisal group, and individual subjects. The statistics of primary interest were number of attempts and percentage correct for each block of serial subtraction. Additionally, the audio recordings were transcribed to obtain data on subtraction pace and details about error types. Some preliminary findings on error types and frequency by appraisal group are reported in Ritter et al.(submitted).

Serial Subtraction Model

The ACT-R architecture was chosen to model the serial subtraction task for three reasons: it provides a subsymbolic level of processing; it permits the parallel execution of the verbal system with the control and memory systems, and it has been used for other models of addition and subtraction developed by other researchers. The serial subtraction model was originally built in an earlier version of the ACT-R architecture, version 4.0, was later modified to run in ACT-R version 5.0, and then finally rewritten for ACT-R version 6.0.

The model performs a block of the serial subtraction task subtracting by 7s or 13s in a similar manner to that of the human subjects. The model's declarative knowledge consists of arithmetic facts and goal-related information. The model's procedural knowledge is in the form of production rules that allow for retrieval of subtraction and comparison facts necessary to produce an appropriate answer. The model performs subtractions by attempting to recall the appropriate

declarative memory chunk to produce a subtraction answer. The predictions of both the version 4.0 and 5.0 models were compared to human performance data from a study by Tomaka and colleagues (1993).

These models were used to test two theories of stress using architectural overlays in the form of specific parameter settings to simulate the effects of task appraisal and math anxiety (Ritter, Reifers, Klein, Quigley, & Schoelles, 2004). Different types of task appraisal (challenge and threat) were simulated by varying the amount of noise in the procedural knowledge selection process using an expected gain noise (EGN) parameter in the architecture. Adding noise to the decision process negatively influences cognitive performance that is consistent with several theories of stress and anxiety discussed in Chapter 2. The math anxiety overlay was based on Ashcraft and Kirk's (2001) math anxiety theory. An extra production rule was incorporated into the model that simulated active worry as a distracting thought; the central component in Eysenck and Calvo's (1992) processing efficiency theory. The results of the Ritter et.al. (2004) study showed that the default parameter settings for ACT-R produced overly competent performance when compared to published data on the task. Both the task appraisal and math anxiety overlays were found to improve the model's predictions, but a match to the published data was not accomplished. This study suggested that more detailed performance data was needed on the serial subtraction task.

After the previously described series of experiments studying the effects of caffeine and stress on cognitive performance, the serial subtraction model was redesigned by Dr. Michael Schoelles from the Cognitive Science Department at Rensselaer Polytechnic Institute. The model was also updated to run under the new version of the ACT-R architecture, version 6.0. The current version of the model utilizes the new imaginal module in ACT-R. The imaginal buffer implements a problem representation capability. In the serial subtraction model the imaginal buffer holds the current 4-digit number being operated on (the minuend) and the number being

subtracted (the subtrahend). The goal module and buffer implement control of task execution by manipulation of a state slot. ACT-R's vocal module and buffer verbalize the answer to each subtraction problem as the subjects do.

The model starts with the main goal to perform a subtraction and a borrow goal to perform the borrow operation when needed. Both types of goal chunks contain a state slot, the current column indicator, and the current subtrahend. The current problem is maintained in the imaginal buffer. This buffer is updated as the subtraction problem is being solved. The model begins with an integer minuend of 4-digits. All numbers in the model are chunks of type integer with a slot that holds the number. The model also contains subtraction and addition fact chunks whose slots are the integer chunks described above. This representation of the integers and arithmetic facts has been used in other ACT-R arithmetic models.

The model determines if a borrow operation is required by trying to retrieve a comparison fact that has two slots, a greater slot containing the minuend and a lesser slot containing the subtrahend. If the fact is successfully retrieved then no borrow is necessary, otherwise a borrow subgoal is created and executed. Borrowing is performed by retrieving the addition fact that represents adding ten to the minuend. The subtraction fact with the larger minuend is retrieved. The model then moves right one column by retrieving a next-column fact using the current column value as a cue. If this retrieval fails then there are no more columns so the borrow and the subgoal returns back to the main task goal. If there is a next column and its value is not zero than one is subtracted from it by retrieval of a subtraction fact. If the value was 0 then the problem is rewritten in the imaginal buffer with a 9 and the model moves to the next column and repeats the steps discussed above, returning to the main task when there are no more columns. The model outputs the answer by speaking the 4-digit result. If the answer is incorrect, the problem is reset to the last correct answer. If the answer is correct, the main problem task is rewritten in the imaginal buffer.

Three ACT-R parameters appeared important for the model and were selected for use in the thesis optimization project. The rate the model speaks is controlled by the seconds-per-syllable parameter (SYL). The ACT-R default timing for speech is 0.15 seconds per assumed syllable based on the length of the text string to speak. There is a default of three characters per syllable controlled by the characters-per-syllable parameter. The seconds-per-syllable and characters-per-syllable parameters control subsymbolic processes in ACT-R's vocal module. The vocal module gives ACT-R a rudimentary ability to speak. It is not designed to provide a sophisticated simulation of human speech production, but to allow ACT-R to speak words and short phrases for simulating verbal responses in experiments.

The other two parameters used in the thesis project affect declarative knowledge access: the base level constant (BLC), and the activation noise parameter (ANS). The BLC parameter and a decay parameter affect declarative memory retrieval and retrieval time. The ANS value affects variance in retrieving declarative information and error rate for retrievals in the model. This instantaneous noise value can also represent variance from trial to trial. Other parameters, such as base level learning, decay, the characters-per-syllable parameters were built into the model as modifiable but were left fixed at their default values throughout the project. The search space for the optimization problem was defined by the following parameter value boundaries: both ANS and SYL 0.1 to 0.90 and BLC 0.1 to 3.0.

Working with Dr. Schoelles, a new front-end function for the model was developed for execution in a parallel processing environment. The processor id along with three ACT-R parameter values (ANS, BLC, and SYL) had to be passed to the model through a system call from a C program (i.e., the PGA). Normally, the parameter values are set within the model code before runtime. An output file formatting parameter was added to record different levels of model output. Additional descriptive statistics calculations were added to the C program to summarize

model predictions (mean and standard deviation across processors for number of attempts and percentage correct).

While logged into a high-performance cluster a Lisp image was built by loading the ACT-R architecture and model files into Lisp, and then the state of the Lisp environment was saved out as a Lisp core file. The core file can be started up by a system call. Because the same Lisp image is run on all the processors, code to randomize the seed of ACT-R's random number generator had to be added and executed before starting up the model. Table 3.1 compares a function call to start the serial subtraction model on a serial machine, to the function call used in the parallel processing environment. In the function call for a serial machine the number of blocks of subtractions (:blocks 1) and the length of a trial (:trial-time 25) can be passed as arguments. Specific parameter values, the subtrahend, and the starting 4-digit number are set within the model code. The function call for running the model in parallel has arguments for processor id, output file format, subtrahend (7 or 13), the starting 4-digit number, and the parameter values (ANS, BLC, and SYL).

Table 3.1: Function calls to start up the serial subtraction model in serial and parallel processing environments.

Serial Processing
Example function call in Lisp: `(run-exp :blocks 1 :trial-time 25)`
Parallel Processing

Syntax:
`(run-sc-exp procID outputFormat subtrahend start4-digit ANSvalue BLCvalue SYLvalue)`
 	
Example function call in C:
`"lisp -quiet -core SStest.core -eval '(progn(sgp-fct `(:seed (,(random 10000),(random 5))))(run-sc-exp %d %d %d %d %f %f %f)(quit))'", my_rank, fileformat, startval, subth, LPar[0],LPar[1],LPar[2];`

Search Algorithm Development

Three algorithmic approaches were considered to optimize the serial subtraction model to human performance data using a parallel processing environment. The problem is to search the ACT-R parameter space for parameter combinations that would adjust the serial subtraction model's predictions to fit the human performance data gathered from the experiment. This is a stochastic global optimization problem where it is reasonable to assume that multiple local optima exist. In general, the task of global optimization is to find a point for which the objective function obtains its smallest value, the global minimum. The objective function, or fitness function in this case, is the discrepancy between model predictions and the human performance. This is a problem with simple bounds defined by a minimum and maximum for each of three ACT-R parameters being investigated.

For the model-to-human data fitting problem two types of optimization approaches were utilized based on a GA and a hill-climbing algorithm, and then these approaches were combined using a hybrid algorithm. GAs are useful for multidimensional optimization problems, and have the advantage of being less susceptible to getting stuck at local optima than gradient search methods (Cantu-Paz, 2001; Haupt & Haupt, 2004). GAs have the disadvantages of lacking a foundation in mathematical theory, and of being computationally expensive (Cantu-Paz, 2001). With a large parametric search space and a computationally intensive fitness function, a parallel processing implementation of the GA is necessary for optimizing the serial subtraction model to the human data.

The other algorithm utilized for the project was a hill-climbing algorithm. Hill climbing is an optimization technique belonging to the family of local search methods (Press, Teukolsky, Vetterling, & Flannery, 1992). Falling into local optima is a disadvantage of local search methods (Haupt & Haupt, 2004). A parallel processing implementation of a hill-climbing algorithm could

involve a set of hill-climbing algorithms each starting from a random solution and moving toward their own locally found minima with one of the minima possibly being a good solution.

The parallel processing component of these algorithms involves the execution of a program on parallel processing hardware with the objectives of unlimited scalability and reduced execution time. A common technique used in creating applications for parallel processing is *message passing*. Applications can use message-passing mechanisms for communication between processors. The Message-Passing Interface (MPI) is a widely used standard for programming parallel systems (Pacheco, 1997). MPI is a library of subprograms that can be included in C, C++, or Fortran programs. When running with MPI, all processors use the same compiled binary, and therefore, run the exact same code. To utilize the MPI library the search algorithm portion was written in C. The cognitive architecture and serial subtraction model are written in Lisp.

The ideal programming environment for the project would be a parallel computing system with several hundred processors running an operating system that supports an open source Lisp programming language and the MPI library. The version of Lisp would need to comply with the current Common Lisp Standard (CL2) to run the ACT-R 6.0 cognitive architecture. A computing resource was located on the TeraGrid at the National Center for Supercomputing Applications (NCSA). A proposal for supercomputing time of 30000 Service Units (SUs) was written and accepted. The programming platform for the project was setup on a NCSA supercluster called Tungsten. Tungsten is a Xeon cluster with 1750 Dell PowerEdge servers, each with two Intel Xeon 3.2 GHz processors, running Red Hat Linux and Myricom's Myrinet cluster interconnect network.

A simple ACT-R cognitive model of the subitizing task (Cowan, 2001; Trick & Phylyshyn, 1994) was developed to test the parallel processing programming environment for the project. This included the C programming language, MPI library, CMU Lisp, ACT-R 6.0 cognitive architecture, and the subitizing cognitive model. After several successful tests, the more

complex model of the serial subtraction task was substituted for the subitizing model and several parameter sweeps were executed. These two testing scenarios confirmed that the cognitive architecture and model could be executed on any number of processors using a C/MPI program. The next two sections discuss the genetic and hill-climbing optimization approaches for fitting a cognitive model to human data in a parallel processing environment.

Parallel Genetic Algorithm

Two prototypes of GAs using a binary and a continuous representation, were initially developed in Matlab for serial processing. Matlab is an interpreted language that can be used as a rapid prototyping tool. Additionally, a Matlab program under development can be evaluated in components, functions, or individual statements for debugging purposes. Generic Matlab code presented in a textbook by Haupt and Haupt (2004) was used as an initial framework for the prototype GAs and then modified to incorporate ACT-R and the serial subtraction model. Both GAs were based on the same operations for modeling genetic recombination and natural selection. The binary GA represents variables as encoded binary strings while the continuous GA works with the continuous variable values themselves. Serial processing severely limited the size of the population and number of generations that could be executed for testing purposes. Both types appeared to work reasonably well under the serial processing constraints. Because GAs originated with a binary representation of the variables, the binary method was chosen for the project. The binary GA in Matlab was rewritten in C and tested again on a serial machine.

The operation in a GA that is most commonly parallelized is the evaluation of the fitness function because it normally requires only the data of a single genotype, i.e., the genotype being evaluated, thus eliminating the need for communication between processors during evaluation. The parallelization of the fitness function in a GA is usually implemented using the master-slave

approach (Ismail, 2004) and was so for the thesis project. The master process stores the population of genotypes and applies the mutation and selection operations. The slave processors evaluate the fitness of the genotypes in the population. Parallelization of the fitness evaluation can be accomplished by assigning a fraction of the population to each of the available processors or, for this project, by assigning one genotype per processor. Communication between processors occurs only as each slave receives a genotype to evaluate, and when the slaves return the fitness values of the genotypes back to the master processor. Table 3.2 shows the pseudocode for the master-slave PGA using MPI for parallel processing.

Table 3.2: Pseudocode for master-slave PGA using MPI.

```
MPI_Init . . .
if (rank is 0)  // master
    Initialize population
. . . . .
for (each generation)
{
    if (rank is 0)  // master
    {
            Selection
            Crossover
            Mutation
    }

    // find fitness of genotypes in population
    // master and slaves
    MPI_Scatter individuals out to processors
    Run cognitive model
    Calculate fitness of model predictions
    MPI_Gather up resulting fitness values

    if (rank is 0)  // master
            Print out generational statistics
}
Test best solutions found  // master and slaves
MPI_Finalize . . .
```

There are two mechanisms that link a GA to the problem it is solving: a way of encoding solutions to the problem as genotypes, and an evaluation or fitness function that returns a measurement of the worth of a genotype in the context of the problem. In the PGA the initial

population of genotypes is in the form of randomly-generated binary strings. A decoding function transforms a binary string of length *l* first to base-10 integer, *z*, then into a real number, *r*, using:

$$r = (r_{max} - r_{min})/(2^l - 1) * z + r_{min.} \quad (3.1)$$

As an example: parameter *x* has a range of values 2.2 to 3.9, a binary string 10101 is decoded as:

x = 10101; therefore, z = 21, and

$$r = (3.9 - 2.2)/(2^5 - 1) * 21 + 2.2 = 3.3516.$$

The next binary number after 10101 is 10110 = 22, which using the above equation implies r = 3.4065. This shows one problem with floating point systems, i.e., that in this case it is not possible to specify any number between 3.3516 and 3.4065. The only way to improve accuracy is to reduce the size of the search space, or to increase the precision used to represent the parameters. By choosing a suitable string length, the required accuracy can usually be maintained, for example, l = 20 implies an accuracy better than one part in a million. In the PGA each of the three ACT-R parameter values was encoded in a 12-bit string; a 36-bit string was then created by concatenating the 12-bit strings ANS, BLC, and SYL. The variable encoding function allowed for a range to be specified for each parameter rather than having only one range for all the variables. The parameter ranges were 0.1 to 0.90 for both ANS and SYL and 0.1 to 3.0 for BLC.

The population of genotypes (encoded ACT-R parameter values) is in the form of a matrix. The genotypes are 'scattered' row-wise to the processors using the *MPI_scatter* function. Each processor executes the Lisp image file that runs the model within the ACT-R architecture. Each processor then calculates a fitness based on the model's performance predictions and the human performance data collected from the experiment. For the thesis project the fitness function was defined by the sum of squares error (discrepancy between model predictions and human data) calculated on both number of attempts (#attpt) and percentage correct (%corr) from the first 4-minute block of subtracting by 7s:

$$\text{Fitness} = (H\#attpt - M\#attpt)^2 + (H\%corr - M\%corr)^2. \quad (3.2)$$

Where H is human performance and M is the model's prediction of performance. The fitness values calculated by the processors are 'gathered' up by the master process using *MPI_gather*. The master process then applies genetic functions to the population based on the fitness of the genotypes. This process is repeated for a number of generations set by a maximum generation boundary value.

The fitness function plays the same role in the PGA that the environment plays in natural evolution. The interaction of an individual with its environment provides a measure of fitness. Similarity, the interaction of a genotype with an evaluation or fitness function provides a measure of fitness that the PGA uses in the selection and crossover operations.

The communication between the processors is synchronous. *MPI_gather* waits to receive the fitness values from all the genotypes in the population before proceeding with the next generation. The synchronous master-slave GA has exactly the same properties as a serial GA, except for its speed. A significant speedup can be expected if the communication cost does not dominate the computational cost. Communication has the classical bottleneck effect with the processing of each generation waiting on the slowest processor to finish its fitness evaluation.

During the development and later testing phases, the genetic functions (selection, crossover, and mutation) performed by the master process were kept relatively constant. The selection probability (selection of the fittest) was set to 0.5 meaning half the population is replaced each generation by offspring of the fittest genotypes using a roulette wheel rank-weighted parent selection procedure.

The effect of roulette wheel parent selection is to return a randomly selected parent (genotype) from the population. Although this selection procedure is random, each parent's chance of being selected is directly proportional to its fitness. Over a number of generations this type of selection process will drive out the least fit parents by spreading 'genetic material' of the

fittest parents. It is referred to as roulette wheel selection because it can be viewed as allocating pie-shaped slices on a roulette wheel to population genotypes, with each slice proportional to the genotype's fitness. Selection of the genotype to be a parent can then be viewed as a spin of the wheel, with the winning genotype being the one in whose slice the roulette spinner ends up. This parent selection technique has the advantage of directly promoting reproduction of the fittest genotypes in the population by biasing each genotype's chances of selection in accordance with its fitness evaluation.

The crossover operation involves two selected parent genotypes producing two offspring using a single crossover point that is randomly selected between the first and last bits of the parents' genotypes. Parent-1 passes its binary code to the left of the crossover point to Offspring-1. Similarly, Parent-2 passes its binary code to the left of the same crossover point to Offspring-2. Next, the binary code to the right of the crossover point of Parent-1 goes to Offspring-2, and Parent-2 passes its code to Offspring-1. Consequently the offspring contain portions of the binary codes of both parents.

Random mutations alter a certain percentage of the bits in the population of genotypes. This operation introduces traits that are not in the original population and keeps the GA from converging too quickly before sampling the entire search space. For the project, the mutation rate was set at a constant 0.15 (i.e., 15% of the bits in the population were flipped). Mutation was randomly applied throughout the population without discrimination with the exception of the best genotype in each generation, which was not mutated. Also, mutation was not applied during the final generation of the algorithm.

The application of mutation used in the thesis is simplistic. In hindsight, it might have been advantageous to apply mutation in a more directed manner especially when considering the consequences of its application. For example, with a binary representation the magnitude of the disruption caused by mutation (i.e., flipping a bit) depends upon where in the genotype the

mutation occurs. Given a single parameter problem with a length $l = 10$, $r_{min} = 0$ and $r_{max} = 1023$, than a mutation at one end of the genotype changes r by ± 1; whereas, at the other end r would change by ± 512.

This analysis indicates that near the end of the GA's run, when hopefully the majority of the population is in the vicinity of the global optimum, mutation should be confined to lower order bits. Conversely, during earlier generations mutation of the higher order bits would promote full exploration of the search space (i.e., mutation of less significant bits would add little exploration). Another possibility is to bias mutation towards less fit genotypes to increase exploration without degrading the performance of the fitter genotypes. Alternative selection, crossover, and mutation mechanisms have been implemented in GAs to ensure the correct balance between exploration and exploitation (March, 1991), but also as a way to alleviate stochastic sampling errors (Coley, 1999).

Two preliminary tests of the PGA optimizing to the average performance of the challenge appraisal group were run using a small population of 24 genotypes and only two generations, and then 5 generations, both on 24 processors. Table 3.3 summarizes the results that show the minimum fitness in the population of genotypes decreasing with each generation. A decreasing fitness value would be expected as the PGA generates new points (genotypes) in the search space by applying genetic operators to the current points and statistically moving toward more optimal positions in the search space. As discussed previously, the fitness is in terms of error (or cost) and is the least squares discrepancy between the model predictions and the human performance on the cognitive task. In the thesis project, the PGA is seeking a global minima in the ACT-R parameter space. Model predictions exactly matching human performance data would have a fitness value equivalent to 0.

Table 3.3: PGA preliminary test results using a population of 24 genotypes and a small number of generations on 24 processors.

PGA Test 1: Population = 24 Generations = 2				
Generation	Minimum Fitness	ANS	BLC	SYL
1	570.19	0.72	2.86	0.73
2	19.21	0.59	2.31	0.32
Run time: 1 minute 53 seconds on 24 processors				
PGA Test 2: Population = 24 Generations = 5				
Generation	Minimum Fitness	ANS	BLC	SYL
1	709.80	0.62	2.35	0.81
2	122.92	0.49	1.94	0.27
3	96.17	0.57	2.73	0.34
4	35.74	0.57	2.77	0.39
5	3.21	0.95	2.85	0.34
Run time: 4 minutes 44 seconds on 24 processors				

Several more tests of the PGA were run with larger populations and more generations. Figure 3.2 shows the results from the first of the larger scale tests of the PGA with a population of 200 genotypes run for 20 generations on 200 processors. One PGA was set-up to optimize to the average of the challenge appraisal group (top plot), and another to the average of the threat appraisal group (bottom plot). The minimum fitness value is plotted across the generations.

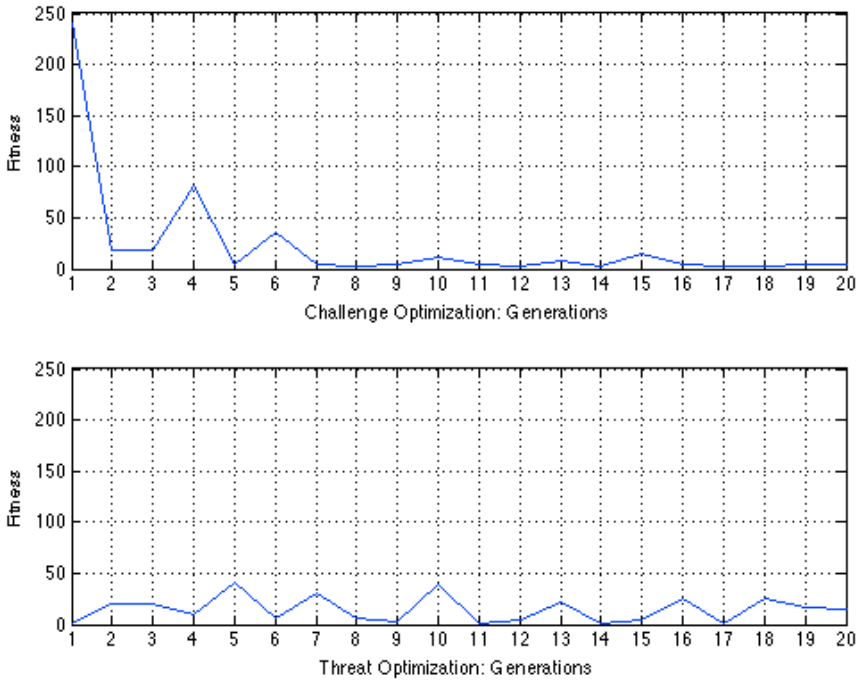

Figure 3.2: PGA test results of minimum fitness value across 20 generations with 200 genotypes run on 200 processors when optimizing to averages of appraisal groups (challenge top plot; threat bottom plot)

The results for the threat optimization were unexpected. Instead of a monotonically decreasing curve as the PGA converges on a best fitting set of parameters, the plot shows the minimum fitness value osculating between nearly 0 and slightly below 50. Minimum fitness values of 0.49 were produced in generations 1 and 11. Additionally, minimum fitness values of 0.45 were produced in generations 14 and 17. These fitness values indicate model predictions that fit the human data to less than half of a subtraction problem with the combined number of attempts and percentage correct statistics. It appears that the PGA could not maintain the genotypes producing these nearly perfect fits through the generations. This illustrates why using an automated search method to optimize cognitive models may be difficult.

The nonmonotonic pattern in the threat optimization plot is not that surprising considering the stochastic effects embedded in the ACT-R architecture and the model. With a static set of parameter values, the combination of the model and architecture yields a distribution

of performance, not a fixed, single statistic. When models themselves also include stochastic components, the model may require 10, 20, or 100s of runs to compute stable performance predictions. Table 3.4 compares five example sets of ACT-R parameters used in the serial subtraction model. For each parameter set, the model was run 10 times and then 100 times. The number of attempts and percentage correct are averaged over the number of model runs.

Table 3.4: Example sets of ACT-R parameters compare model predictions of number of attempts and percentage correct averaged over 10 model runs and 100 model runs.

ACT-R Parameters			Mean Across 10 Model Runs		Mean Across 100 Model Runs	
ANS	BLC	SYL	Number of Attempts	Percent Correct	Number of Attempts	Percent Correct
0.463	0.693	0.526	42.30	82.56	42.88	85.00
0.399	0.839	0.555	40.30	78.05	40.03	89.74
0.251	0.588	0.529	42.50	80.49	41.57	82.93
0.766	0.619	0.531	40.90	90.00	41.11	78.57
0.654	0.078	0.588	39.60	78.95	38.77	89.47

When comparing model performance across 10 and 100 runs, percentage correct shows more variance than number of attempts. This makes for difficult optimization especially if both performance statistics are used simultaneously in the fitting process and the problem itself is multi-dimensional. This could be the reason why previous attempts at fitting the serial subtraction model to data from other human subject experiments using manual optimization techniques have been unsuccessful (Ritter, Schoelles, Klein, & Kase, 2007).

The stochasticity embedded in the architecture complicates the optimization process and contributes to the PGA's lack of convergence on a solution. The PGA was redesigned to compensate for a single genotype returning a distribution of performance predictions instead of a fixed value. To ameliorate lack of convergence, collection and testing features were added into the PGA code. During its generational journey, if the PGA finds a 'good enough' solution, as determined by a boundary fitness value, that particular genotype is remembered for a post-PGA testing phase. In essence, the PGA gathers up good solutions across the generations, as a

substitute for converging on a so-called best set of solutions. Once the PGA terminates, each of the collected genotypes is run on all the allocated processors with a fitness value calculated from the mean number of attempts and percentage correct across all runs.

To test the new genotype collection and post-PGA testing code, two PGAs were set up to run 50 generations of 200 genotypes on 200 processors, each optimizing to an appraisal group average. Table 3.5 shows the results from both optimizations listing the four best fitting genotypes (by post-PGA test) from each appraisal optimization collected by the PGA. The second column shows the genotypes' original fitness values as reported by the PGA compared to their fitness values from post-PGA testing in the third column.

Table 3.5: Four best genotypes collected during optimizations by appraisal group compared to post-PGA test phase results using 200 processors per genotype.

Genotypes	Fitness	
Challenge Optimization	PGA Reported	Post –PGA Testing
0.500, 2.083, 0.365	0.093	0.133
0.271, 1.558, 0.360	0.093	1.244
0.561, 2.279, 0.366	0.093	1.696
0.500, 2.083, 0.365	3.597	3.610
Threat Optimization	PGA Reported	Post-PGA Testing
0.727, 2.538, 0.535	6.191	0.986
0.729, 2.524, 0.593	6.008	7.072
0.693, 2.446, 0.586	3.075	7.896
0.713, 2.446, 0.593	3.614	8.325

In the threat appraisal optimization, the genotype producing the best fitness as reported by the PGA (3.075) does not correspond to the genotype producing the best fitness from the post-PGA testing phase (0.986). In the challenge appraisal optimization, there were three genotypes with a fitness of 0.093. One of the genotypes produced the best post-PGA test fitness value (0.133).

As a validation effort, the best fitting set of ACT-R parameters from each appraisal optimization in Table 3.5 was tested with three additional sets of 200 runs each on 200

processors. Table 3.6 shows the number of attempts and percent correct averaged over each of the 200 runs, and a comparison of the model's mean performance to the subjects' mean performance by appraisal group. Column 1 in Table 3.6 shows a best fitting parameter sets from each appraisal optimization. Test results are show in columns 2 and 3.

Table 3.6: Validation of best fitting parameter sets from appraisal optimization results using the new collection and post-PGA testing code compared to human data. Three tests of each parameter set listed in column 1 were performed with each test consisting of 200 model runs.

Challenge Appraisal Performance		
ACT-R Parameters	Number of Attempts (Average)	Percent Correct (Average)
0.500, 2.083, 0.365	55.0	83.5
	55.1	83.0
	55.0	84.7
Model Performance Means	55.0	83.7
Human Data Means	55.3	85.4
Threat Appraisal Performance		
ACT-R Parameters	Number of Attempts (Average)	Percent Correct (Average)
0.727, 2.538, 0.535	40.8	76.5
	40.9	78.5
	40.8	77.4
Model Performance Means	40.8	77.5
Human Data Means	40.3	78.1

The table results show that for number of attempts the fit is nearly perfect; a difference of 0.3 for the challenge group, and 0.5 for the threat group—half a subtraction problem or less. The fit is slightly less accurate for percentage correct; a difference of 1.7% for the challenge group, and 0.6% for the threat group. The total run time was minimal; 117 minutes on 200 processors for the two PGAs including the post-PGA testing phase, and four minutes for the additional best solution validation runs. The results confirm that the new collection and testing features incorporated into the PGA code appear to compensate for the stochasticity in the architecture and model. This version of the PGA was used throughout the remainder of the thesis project.

Hill-climbing Algorithm

A prototype hill-climbing optimization algorithm for finding the minimum of a fitness function was developed in Matlab for testing on a serial machine. The algorithm begins with one set of ACT-R parameters (ANS, BLC, SYL). The same process of executing a Lisp core file to start the ACT-R architecture and run the serial subtraction model was used. Fitness is again defined as the sum of squares error between the model's predictions and the human data for number of attempts and percentage correct for a block of subtracting by 7s. From the starting set of parameters and three dimensions defined by the number of parameters in the set, the algorithm climbs and descends by a step precision staying within a three-dimensional space constrained by lower and upper boundary values for each parameter. For each iteration of steps in the 3-D space the model is run using the current step's set of parameter values and a fitness value is calculated. The fitness values produced by a single step in all directions are then compared to the fitness of the parameter set that started off the round of steps. If an improved (lower value) fitness is not found the algorithm terminates. Otherwise, the step producing the best fitness is used to define a climb or descent direction. Steps in that direction are taken while fitness continues to improve, and terminates when no additional fitness improvement is found. The parameter set producing the best fitness so far is then used as the starting point for another round of steps in all directions in another iteration of the algorithm's main loop. Table 3.7 shows pseudocode for the hill-climbing algorithm optimization.

Table 3.7: Pseudocode for hill-climbing algorithm.

```
Randomly generate or assign the starting parameter set
Run model, calculate fitness

Loop 1
    Loop 2 "iteration of steps in all directions"
         Climb or descend a step in each parameter dimension
         Run model, calculate fitness
    End Loop 2

    Compare fitness values from Loop 2 to starting fitness

    if (found a better fitness)
         Loop 2  While fitness continues to improve
                 Continue climb or descent in that direction one step
                 Run model, calculate fitness
         End Loop 2

         Save parameter set producing best fitness so far
         Start over with that parameter set, go back to Loop 1

    else (there was no improvement to fitness)
         Break Loop 1, terminate program
End Loop 1
```

The Matlab version of the hill-climbing algorithm was tested by optimizing to the human performance averages of the challenge and threat appraisal groups. The algorithm used randomly generated initial sets of ACT-R parameters. Tables 3.8 and 3.9 show example traces of the algorithm climbing and descending with a step size of 0.01 starting from a random set of parameter values. The resulting minimum fitness values found when the algorithm terminated are shown in bold text. Table 3.8 traces the hill-climbing algorithm optimizing to the challenge group average, and Table 3.9 when optimizing to the threat group average, both run on a serial processor.

When a –1 value appears in the #Attempts and %Correct columns this is a summary row showing the minimum fitness found for that iteration number (last column in table; Loop 1 in the algorithm pseudocode). Both traces show poor optimization results; the algorithms terminated with fitness values 1000s away from a perfect fit of 0 (fitness of 4174.8 for challenge, and 3141.1 for threat). The hill-climbing algorithm appears to get stuck in the nearest local minimum and

terminate. Despite the disappointing results the hill-climbing algorithm was rewritten in C with the expectation of incorporating it with another type of algorithm in the formulation of a hybrid optimization approach.

Table 3.8: Optimization results for fitting to the average of the challenge appraisal group using a randomly generated initial set of ACT-R parameters run on a serial processor.

Optimize to Challenge Average (Human data: #Attempts 55.3, %Correct 85.4)						
Fitness	#Attempts	%Correct	ANS	BLC	SYL	Iteration
7830.3	14	7.14	0.86	0.77	1.86	0
4174.8	14	35.71	0.85	0.77	1.86	1
6818.2	15	13.33	0.87	0.77	1.86	1
6818.2	15	13.33	0.86	0.76	1.86	1
7830.3	14	7.14	0.86	0.78	1.86	1
6762.3	14	14.29	0.86	0.77	1.85	1
6762.3	14	14.29	0.86	0.77	1.87	1
6818.2	15	13.33	0.84	0.77	1.86	1
4174.8	-1	-1	0.85	0.77	1.86	1
7830.3	14	7.14	0.84	0.77	1.86	2
5797.9	14	21.43	0.86	0.77	1.86	2
5797.9	14	21.43	0.85	0.76	1.86	2
4335.4	15	33.33	0.85	0.78	1.86	2
8998.9	14	0	0.85	0.77	1.85	2
6762.3	14	14.29	0.85	0.77	1.87	2
Best fit:						
4174.8	**14**	**35.71**	**0.85**	**0.77**	**1.86**	**1**
Time: 22.834 seconds on serial processor						

Additional tests were performed with the C version of the hill-climbing algorithm. At this point in time, the PGA had been used to optimize to each appraisal group average successfully; therefore, ACT-R parameter sets producing nearly perfect fits for each appraisal group were on hand. The next test involved starting the hill-climbing algorithm with one of these parameter sets known to produce a nearly perfect fit from the appraisal group optimizations. It was hoped that the algorithm could recognize the global optima and terminate appropriately. Tables 3.10 and 3.11 show the results produced by the hill-climbing algorithm when seeded with best fitting

parameter sets found by the PGA for the appraisal groups. Each table compares the PGA-produced parameter set (used as the start-up set for the hill-climbing algorithm) and fitness to the hill-climbing algorithm's ending parameter set and best fitness.

Table 3.9: Optimization results for fitting to the average of the threat appraisal group using a randomly generated initial set of ACT-R parameters run on a serial processor

Optimize to Threat Average (Human data: #Attempts 40.3, %Correct 78.1)						
Fitness	#Attempts	%Correct	ANS	BLC	SYL	Iteration
4839.4	34	8.82	0.58	0.89	0.68	0
4862.5	35	8.57	0.57	0.89	0.68	1
3792.1	36	16.67	0.59	0.89	0.68	1
4182.8	37	13.51	0.58	0.88	0.68	1
4506.2	36	11.11	0.58	0.9	0.68	1
4839.4	34	8.82	0.58	0.89	0.67	1
4141.4	36	13.89	0.58	0.89	0.69	1
3840	37	16.22	0.6	0.89	0.68	1
3792.1	-1	-1	0.59	0.89	0.68	1
4839.4	34	8.82	0.58	0.89	0.68	2
3141.1	36	22.22	0.6	0.89	0.68	2
4222.5	38	13.16	0.59	0.88	0.68	2
3792.1	36	16.67	0.59	0.9	0.68	2
4141.4	36	13.89	0.59	0.89	0.67	2
5255.4	34	5.88	0.59	0.89	0.69	2
4141.4	36	13.89	0.61	0.89	0.68	2
3141.1	-1	-1	0.6	0.89	0.68	2
4141.4	36	13.89	0.59	0.89	0.68	3
4839.4	34	8.82	0.61	0.89	0.68	3
3792.1	36	16.67	0.6	0.88	0.68	3
4506.2	36	11.11	0.6	0.9	0.68	3
3792.1	36	16.67	0.6	0.89	0.67	3
4182.8	37	13.51	0.6	0.89	0.69	3
Best fit:						
3141.1	**36**	**22.22**	**0.6**	**0.89**	**0.68**	**2**
Time: 57.501 seconds on serial processor						

Table 3.10: Optimization results for fitting to the average of the challenge group using ACT-R parameters found to be best fits by the PGA as starting points for the hill-climbing algorithm run on a serial processor. Fitness is the least squares fit for number of attempts and percentage correct.

Optimize to Challenge Average (Human data: #Attempts 55.3, %Correct 85.4)												
PGA results used as starting points for hill-climbing						Hill-climbing results					
Fitness	#Attpt	%Corr	ANS	BLC	SYL	Fitness	#Attpt	%Corr	ANS	BLC	SYL
0.133	54.95	85.30	0.50	2.08	0.37	0.093	55.00	85.45	0.50	2.07	0.37
						4.369	54.00	87.04	0.50	2.09	0.37
						0.588	56.00	85.71	0.49	2.08	0.36
						7.228	53.00	86.79	0.50	2.07	0.38
1.244	54.31	85.91	0.27	1.56	0.36	4.370	54.00	87.04	0.27	1.56	0.36
1.696	55.04	86.68	0.56	2.28	0.37	4.370	54.00	87.04	0.56	2.28	0.37
						0.093	55.00	85.45	0.56	2.28	0.36
						0.093	55.00	85.45	0.57	2.28	0.37

Table 3.11: Optimization results for fitting to the average of the threat group using ACT-R parameters found to be best fits by the PGA as starting points for the hill-climbing algorithm run on a serial processor. Fitness is the least squares fit for number of attempts and percentage correct.

Optimize to Threat Average (Human data: #Attempts 40.3, %Correct 78.1)											
PGA results used as starting points for hill-climbing						Hill-climbing results					
Fitness	#Attpt	%Corr	ANS	BLC	SYL	Fitness	#Attpt	%Corr	ANS	BLC	SYL
0.986	40.87	77.29	0.73	2.54	0.54	3.700	40.00	80.00	0.73	2.44	0.55
7.072	38.08	76.64	0.73	2.52	0.59	3.614	39.00	79.49	0.73	2.52	0.59
7.896	38.37	76.06	0.69	2.45	0.59	3.075	39.00	76.92	0.68	2.44	0.58

More tests were run while optimizing to the challenge group, including several tests per starting set of parameter values, such as the first row of results in Table 3.10. This row indicates that the PGA found the ACT-R parameter set (0.05, 2.08, 0.37) produced a fitness of 0.133. This same parameter set was then used as the starting point for the hill-climbing algorithm for four separate tests. Overall, across both tables, the hill-climbing algorithm returned nearly the same set of parameter values as it was started with, sometimes with a small increment or decrement of 0.01 for one of the parameter values. The fitness values produced by the hill-climbing algorithm were sometimes less than and other times greater than those produced by the PGA. For the threat optimizations, two of the three fitness values produced by the hill-climbing algorithm were lower than those produced by the PGA. For the challenge group, half of the fitness values returned by the hill-climbing algorithm were lower then those produced by the PGA.

It could be concluded from the test results that the hill-climbing algorithm was able to recognize a best fitting set of parameter values. It might be the case that the hill-climbing algorithm could be utilized in combination with another algorithm for 'closing in' on a best solution. This idea was explored in the next section by creating a hybrid combination of the GA and the hill-climbing algorithm.

Hill-climbing-genetic Hybrid Algorithm

A hybrid algorithm was developed by integrating the hill-climbing code as described in the previous section with the PGA. The algorithm integration point was at the fitness evaluation function. In the PGA the population of genotypes is scattered out to the slave processors for executing the model and then calculating a fitness value based on the model predictions compared to the human data. On each processor a genotype and its fitness is used as the starting point for the hill-climbing algorithm. The hill-climbing algorithm would take over and execute a round of

steps at the genotype's point in the parameter space. If the hill-climbing algorithm finds a set of parameters producing a better fit than the original set the processor started with, this new set replaces the original genotype in the population with its fitness value sent to the master process.

The hill-climbing algorithm works with real-value parameters not the binary-encoded genotypes making up the PGA's population. The real values converted from the genotypes using (3.1) are saved in a data structure and used as hill-climbing starting points. If the hill-climbing portion of the hybrid algorithm succeeded in finding a better fit, that set of ACT-R parameter values was encoded back into binary format to replace the original genotype in the PGA's population. The encoding was accomplished by the reverse of (3.1) by translating the real value to a base 10-integer, and then converting that integer to binary format.

Three preliminary tests of the hill-climbing-genetic hybrid were run using small populations of 20, 40, and 80 genotypes for four-generational optimizations to the challenge appraisal group average. The fitness function remained the same; the sum-of-squares error calculated on both number of attempts and percentage correct from the first four-minute block of subtracting by 7s. Table 3.12 summarizes the results from the three tests. In all the tests the minimum fitness in the population of genotypes decreased with each generation, or a genotype producing a very low fitness, once found, was maintained over the remaining generations. In Hybrid Test 1 generations 2, 3, and 4 show a minimum fitness value that fluctuates between 0.09 and 0.59 for the same set of parameter values. As discussed previously in the PGA development section, stochastic components in the cognitive architecture and model cause instability in performance predictions where a static set of parameter values can yield a distribution of performance predictions instead of a single fixed value.

The tests of the hill-climbing-genetic hybrid showed a substantial increase in runtime when compared to the preliminary tests on the PGA without the additional hill-climbing code. With a population of 24 genotypes the PGA processed five generations in 4 minutes and 44

seconds using 24 processors, where as the hybrid took 50 minutes and 24 seconds to process 20 genotypes for four generations using 20 processors. The tradeoff between runtime and quality and quantity of the solutions would need to be examined further.

Table 3.12: Preliminary test results of hill-climbing-genetic hybrid algorithm with four generations of populations 20, 40, and 80.

Hybrid Test 1: Population = 20 Generations = 4				
Generation	Minimum Fitness	ANS	BLC	SYL
1	32.51	0.85	2.44	0.32
2	0.09	0.90	3.00	0.37
3	0.09	0.90	3.00	0.37
4	0.59	0.90	3.00	0.37
Run time: 50 minute 24 seconds on 20 processors				
Hybrid Test 2: Population = 40 Generations = 4				
Generation	Minimum Fitness	ANS	BLC	SYL
1	510.12	0.74	2.18	0.70
2	3.60	0.47	1.63	0.36
3	1.74	0.37	1.53	0.36
4	0.09	0.83	2.79	0.39
Run time: 74 minutes 15 seconds on 40 processors				
Hybrid Test 3: Population = 80 Generations = 4				
Generation	Minimum Fitness	ANS	BLC	SYL
1	33.32	0.56	2.26	0.32
2	0.59	0.90	2.76	0.36
3	0.09	0.90	2.76	0.36
4	0.59	0.79	2.61	0.36
Run time: 72 minutes 44 seconds on 80 processors				

The hybrid was tested with an increased number of generations. Figure 3.3 shows the results from the hybrid executing 10 generations of 40 genotypes on 40 processors. One hybrid was set up to optimize to the average of the challenge group (top plot), and another to the average

of the threat group (bottom plot). The minimum fitness values are plotted across the 10 generations.

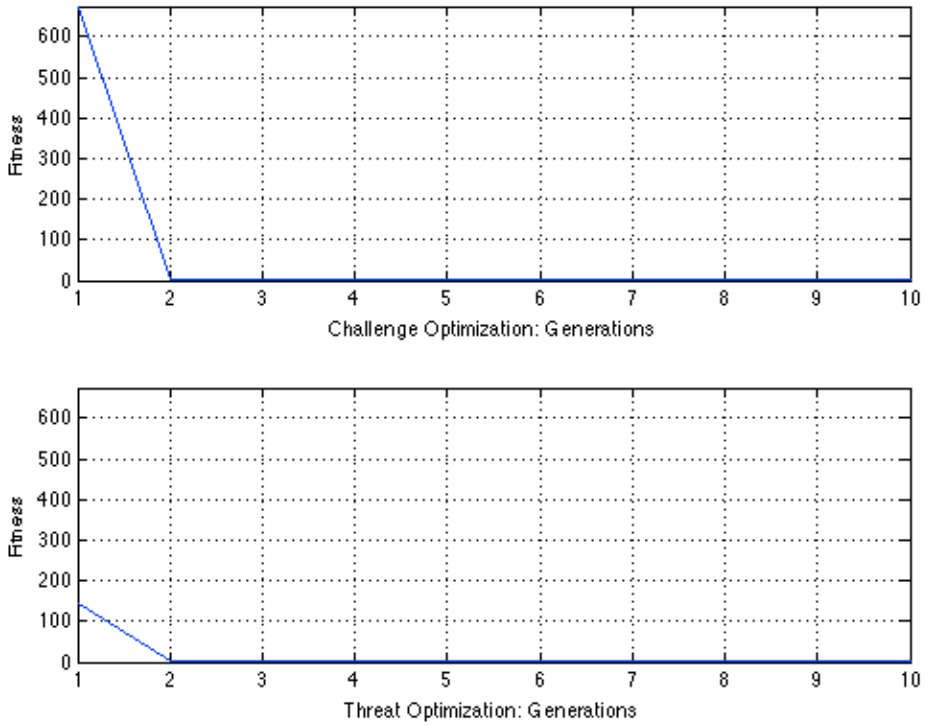

Figure 3.3: Hill-climbing-genetic hybrid test results of minimum fitness value across 10 generations with 40 genotypes run on 40 processors when optimizing to averages of appraisal groups, challenge (top) and threat (bottom).

Both optimizations show the hybrid converging on a good solution within one generation indicated by the steep downward sloping line. At the end of generation two the minimum fitness values for challenge and threat optimization are 1.73 and 0.49, respectively. At first glance, the addition of the hill-climbing code appears to have solved the lack of convergence problem discovered in the initial version of the PGA. The post-PGA testing code used earlier was added to the hybrid approach for validation of the rather amazing optimization results. Table 3.13 compares the minimum fitness values that the hybrid found for each generation to the post-PGA testing phase fitness values. In the post-PGA testing phase the genotypes producing the best fitness values across the generations are tested on all the allocated processors (in this case 40)

after the hybrid completes the last generation. Fitness is then calculated from number of attempts and percentage correct averaged over the model runs across the processors.

Table 3.13: Genotypes from hill-climbing-genetic optimizations producing the minimum fitness during each generation compared to the fitness produced during post-PGA testing phase using 40 processors per genotype

	Genotypes	Fitness	
Generation	Challenge Optimization	Hybrid Reported	Post –PGA Testing
1	0.28, 2.45, 0.71	647.566	Not tested
2	0.61, 2.71, 0.38	1.736	30.927
3	0.90, 2.64, 0.37	0.093	28.183
4	0.80, 2.82, 0.37	0.093	11.057
5	0.90, 2.82, 0.37	0.093	102.576
6	0.90, 2.72, 0.37	0.093	113.663
7	0.66, 2.35, 0.37	0.093	38.746
8	0.66, 2.45, 0.37	0.093	16.369
9	0.66, 2.55, 0.37	0.093	4.139
10	0.66, 2.55, 0.37	0.093	4.064
Generation	Threat Optimization	Hybrid Reported	Post-PGA Testing
1	0.65, 2.36, 0.66	142.414	Not tested
2	0.29, 1.60, 0.53	0.493	133.604
3	0.71, 2.25, 0.56	0.450	53.697
4	0.87, 2.34, 0.56	0.450	163.225
5	0.87, 2.44, 0.56	0.450	400.986
6	0.87, 2.24, 0.56	0.450	216.277
7	0.80, 2.34, 0.54	0.493	85.478
8	0.80, 2.34, 0.54	0.493	105.815
9	0.90, 2.34, 0.54	0.493	399.657
10	0.90, 2.61, 0.56	0.450	147.953

The boundary fitness value that initiates the post-PGA test is 100; therefore, the genotypes in generation one of both appraisal optimizations were not tested. In the challenge optimization, the last two genotypes (generations 9 and 10) producing a fitness of 0.093 by the hybrid, produced the lowest fitness (slightly over 4.0) in the post-PGA testing phase. The hybrid's reported fitness of 0.093 is calculated from the statistics of one model run. The fitness of 4.139 and 4.064 is calculated averaged over 40 model runs. This is the same problem as encountered by the original PGA caused by the stochastic characteristics in the architecture and

the model. In the threat optimization, all but one generation had a minimum fitness value less than 0.50; unfortunately, none of these genotypes produced reasonable fits in the post-PGA testing phase. The lowest fitness from the post-PGA testing phase is 53.697, from a genotype that initially produced a fit of 0.45 during generation 3 of the hybrid execution.

Although the hybrid converged on a best set of solutions as would be expected in a traditional GA, most of these solutions failed the validation process involving multiple model runs per solution. The PGA without the hill-climbing code showed little in the way of convergence on a solution set, but the genotypes producing good fits collected throughout the generations produced better validation results in some cases.

Summary

During the development process, it was found that the PGA could not maintain genotypes with minimum fitness values across generations. Because the ACT-R architecture contains random components embedded in its subsymbolic-level processing a static set of parameter values can yield a distribution of performance when running a model instead of a fixed single performance value. This stochasticity generates instability in genotype fitness values during PGA execution. A genotype representing a set of ACT-R parameter values producing a good model-to-data fit in one generation might not produce the same good fit in the next generation; thereby, losing its fitness ranking within the population of genotypes and disrupting the PGA's convergence.

To compensate for the stochastic effects in the architecture and associated genotype fitness instability and lack of convergence, collection and testing functions were added to the base version of the PGA. The modified version of the PGA successfully optimized the serial subtraction model to the appraisal group averages. Additional testing confirmed that the serial

subtraction model needs to be run multiple times, possibly as many as 100, in order to produce stable performance predictions despite architectural stochastic characteristics.

It is unfortunate that architecture and model stochasticity disrupt the convergence behavior of the PGA. GAs perform a 'intelligent' search by accumulation of historical information across the evolved generations as to the next points in the search space to search. This concept of a guided global search of a parameter space is central to a GA, and its ineffectiveness in the search for ACT-R parameter sets producing good fits raises concern. The PGA in the thesis applies the standard set of genetic operations in the same iterative manner as typical of GAs, but lacks the intelligent aspect of accumulated historical information.

The modified form of the PGA was significantly more effective in its optimization of the model-to-data fit than the hill-climbing algorithm, the hybrid, and especially the traditional manual optimization process. The modified form of the PGA was utilized for the remainder of the thesis project.

Not reported in this thesis, a more formal performance comparison was conducted between the hill-climbing-genetic hybrid, as described above, and the PGA. Both algorithms were compared when optimizing to the challenge and threat appraisal groups. Because the hybrid required a substantially longer runtime then the PGA, population size was limited to 40 genotypes evolving for only 10 generations using 40 processors. Three optimizations were executed comparing the PGA and the hybrid fitting the model to the threat group average. The same execution format was used for fitting the model to the challenge group average.

The optimization results for the hybrid were similar to the test results during the development process. The hybrid converged on a set of solutions by the second or third generation, with most of these solutions not producing good fitness during the post-PGA validation testing. In comparison, the optimization results for the PGA showed the expected nonconvergence, and because the number of generations was restricted to 10, the algorithm did

not have ample opportunity to evolve and collect good fitting genotypes. This resulted in poor fits for the best solution of each generation as well as the post-PGA testing phase. With both algorithms performing poorly during the comparison study, the hill-climbing-genetic hybrid was abandoned for the remainder of the thesis project.

The tests during development and the comparison study described above confirm that a local search algorithm would not be useful in searching the ACT-R parameter space. The hill-climbing algorithm and the hill-climbing portion of the hybrid immediately got stuck at a local minima and then terminated. This suggests that the ACT-R parameter landscape is not a smooth concave or convex surface but is instead characterized by many peaks and valleys. Additionally, the fact that the hill-climbing algorithm was only executing one model run at each step, the same stochasticity causing nonconvergence in the PGA is playing a part in its failed search attempts.

Future research possibilities for the hybrid include restructuring the combination of the two algorithms to take better advantage of the local search capabilities offered by the hill-climbing technique. For example, at each step (point in the search space) taken during hill-climbing the algorithm could run the model several times and calculate a fitness over those model runs. Another alternative is to switch to hill climbing only if the genotype produced a decent fitness in the genetic algorithm code, meaning it found a promising area to search. This would limit fruitless hill climbing at bad starting positions in the search space and may generally increase efficiency.

Chapter 4

PROJECT RESULTS

Traditionally, cognitive modeling researchers use a manual optimization process to fit the model to the human data. This time consuming, iterative process involves selecting a set of parameters, assigning a numeric value to each parameter, running the model in the cognitive architecture, and evaluating the resulting model predictions against the human data. If the model fit is unsatisfactory, the process is repeated. For the most part, parameter values are iteratively selected without awareness that the current model-to-data fit is heading towards either a local optima or global optima.

This optimization process can be complicated by stochastic components in the cognitive architecture and model, or a wide range of human performance variance on the task, or both. With a static set of parameters and values, the combination of the model and its execution in a cognitive architecture yields a distribution of performance in a range of output values, not a single performance statistic. Manual optimization can be a reasonably effective process when using one parameter and fitting to one set of performance data, such as an average across subjects, which typically is the case in most cognitive modeling research.

As an example of the manual optimization process, the same performance statistics (number of attempts and percentage correct) were used in a previous study (Ritter, Schoelles, Klein, & Kase, 2007) that investigated the range of performance on the serial subtraction task in a mixed gender experiment examining hormonal responses to stress. The serial subtraction task was completed by 56 subjects in this experiment. An average performance across subjects was calculated (number of attempts = 47.0; percentage correct = 82%). Fitting to an average across subjects was attempted, even though the subjects exhibited a wide range of performance on the

task. The same ACT-R parameters (ANS, BLC, SYL) were used as in the thesis project. Table 4.1 shows how manual optimization might proceed ordered by rows from the top down. For each set of parameter values in Table 4.1 the model was run 10 times and the results were averaged over the runs.

Table 4.1: Example of the manual optimization process for fitting to the average performance across subjects (number of attempts and percentage correct, rounded)

Human Performance (Avg across 56 subjects)	Model Predictions	ACT-R parameters (ANS, BLC, SYL)
47.0, 82.0	80.2, 92.1	0.38, 1.60, 0.15
	84.1, 90.2	0.38, 1.85, 0.15
	46.7, 69.4	0.38, 1.85, 0.45
	51.1, 80.4	0.38, 1.85, 0.35
	51.0, 73.6	0.38, 1.85, 0.40
	51.2, 87.0	0.38, 2.00, 0.40

The parameter sets in rows 1 and 2 differ by BLC value. Both sets of parameters produced substantially higher number of attempts and percentage correct than the human data. The next attempt, row 3, was to decrease the rate of speech (SYL parameter) from the default value 0.15 to 0.45 seconds per syllable. This resulted in a number of attempts closer to the human data (46.7 model versus to 47.0 human), but the percentage correct is too low (69.4 model versus 82.0 human). Row 4 shows the effect of decreasing SYL by 0.1 to 0.35; both model predictions end up higher than the human data. An SYL value between the previous two attempts is tried next (SYL = 0.40) resulting in lowering only the percentage correct prediction. Row 6 shows the results when BLC is increased to 2.0 while SYL remains at 0.40. The manual optimization process would continue in this trial and error manner building up a history of results that guide the next attempt while closing in on a best-fitting set of parameter values. If the model was fit to individual subjects this same effortful and time consuming manual optimization technique would need to be applied 56 times, once for each subject until a fit was found. Results in the hormonal

study showed that a good model-to-data fit for an average subject was not accomplished because of performance distribution mismatches (Ritter, et al. 2007).

In the stress and caffeine experiment, where the thesis data originated, the serial subtraction task again generated a wide range of human performance. This variability and the failed attempt to fit an average subject in the hormonal study suggested that the best model fitting approach would be to fit individual subjects or small groups of individuals if their behavior could be clustered by a similar characteristic. However, when considering the substantial time and effort involved in fitting individual subjects using manual optimization, individual data fitting did not appear a viable option. The goal of the thesis project was to develop a more efficient and automated optimization approach to enable the investigation of individual differences in serial subtraction performance.

The optimization approach developed by the thesis integrated a search algorithm, a PGA, on a HPC platform with the ACT-R cognitive architecture to optimize the serial subtraction model to individual level performance data. This optimization approach conducts an automated search of the ACT-R parameter space for the best fitting combination of parameter values.

The three sections of this chapter detail the optimization results for three different levels of model data fitting. The PGA, running on the NCSA HPC cluster, was used to fit the serial subtraction model to data from subjects in the control treatment group (15 subjects, no caffeine) at the following levels of analysis: (1) fitting to an average performance across subjects, (2) fitting to individuals grouped by challenge and threat task appraisal, and (3) fitting to each individual subject that performed the serial subtraction task.

Fitting to Average Performance

Table 4.2 shows the average subtraction rates for the subjects' performance on two 4-minute blocks of subtracting by 7s. The large standard deviations indicate that there is a wide range of performance on this task representing a high degree of individual differences. The PGA, as described in the development section of Chapter 3, was run on the NCSA cluster to optimize the serial subtraction model to an average performance across the 15 subjects in the control treatment group of the experiment described in the background section of Chapter 3.

Table 4.2: Human subject (N=15) mean performance and standard deviation for serial subtraction on two 4-minute blocks of subtracting by 7s.

	7s – 1st block	7s – 2nd block
Number of Attempts	47.3 (15.2)	47.8 (19.2)
Percent Correct	82.0 (10.0)	88.8 (7.0)

The PGA was set up to run 100 generations of 200 binary-encoded genotypes (ACT-R parameter sets). The PGA optimized the model to the average human performance means of 47.3 number of attempts and 82.0 percentage correct from the first block of subtractions. The PGA used genotypes encoded as a concatenation of three ACT-R parameter values: activation noise representing variance in retrieving declarative information (ANS), the base level constant affecting activation and declarative memory retrieval (BLC), and syllable rate, seconds per syllable (SYL). One processor was allocated for each genotype (200 processors). The PGA's selection probability was set to 0.5, and the mutation rate was set at 0.15. The terminating condition was a specified number of generations (100). The fitness function compared the sum of squares error for the model's predicted number of attempts and percentage correct to the corresponding human data statistics. The fitness is in terms of error (or cost); therefore, the PGA is seeking a global minima in the ACT-R parameter space as defined by the parameters used.

Figure 4.1 plots the progress of the PGA across the 100 generations. The top plot of Figure 4.1 shows the minimum fitness value and the bottom plot shows the average fitness value when optimizing to average performance across subjects.

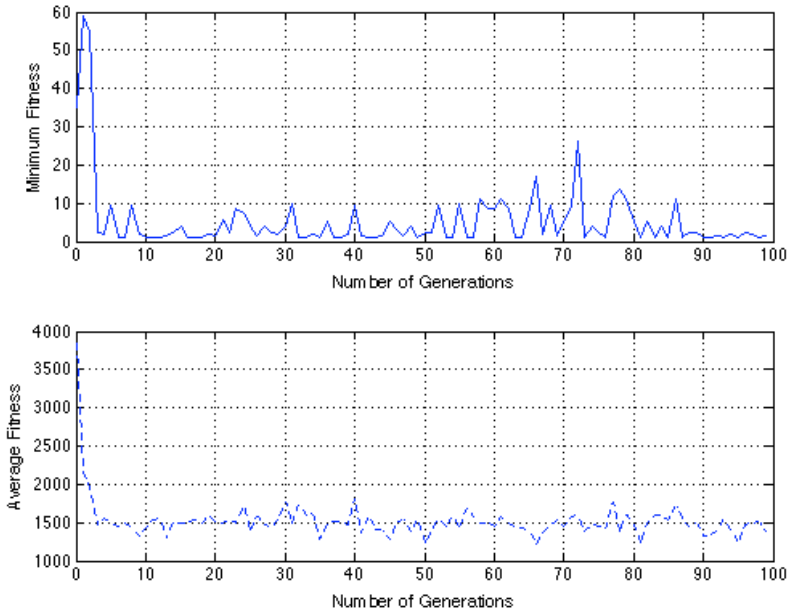

Figure 4.1: Progress of PGA optimization to average performance across subjects in the control group (N=15) showing minimum fitness value (top) and average fitness (bottom) running 100 generations of 200 genotypes using 200 processors

While optimizing to the average performance, the PGA collected 5 genotypes with fitness values less than 1.0 as it evolved the 100 generations. A fitness value equal to 0 means the model predictions match the human performance data exactly. The collected genotypes were tested in the post-PGA testing phase by running each genotype on all the processors and calculating an average fitness from number of attempts and percentage correct across the 200 model runs.

Table 4.3 lists these best fitting genotypes ordered by post-PGA testing fitness value in comparison to the human data averages. In the model prediction column, the first number represents the number of attempts, and the second number is percentage correct. In the genotype column the ACT-R parameter values are ordered ANS, BLC and SYL.

Table 4.3: PGA optimization results for fitting to the average across subjects using 200 genotypes over 100 generations on 200 processors. Human performance (number of attempts and percentage correct, rounded) is compared to model predictions (number of attempts and percentage correct, rounded), and fitness value with the corresponding genotype (ANS, BLC, SYL) producing the fit.

Human Performance (Avg across subjects)	Model Prediction	Fitness Value	Genotype (ACT-R parameters ANS, BLC, SYL)
47.3, 82.0	47.0, 81.8	0.1652	0.76, 2.65, 0.45
	47.0, 81.4	0.4252	0.76, 2.65, 0.45
	46.7, 82.2	0.4418	0.81, 2.82, 0.45
	48.1, 82.4	0.7520	0.64, 2.43, 0.44
	46.5, 81.6	0.8118	0.25, 1.59, 0.45

The results in Table 4.3 show that the PGA optimization was able to produce excellent model to data fits; they are much improved over the previous hormonal study that also attempted to fit the serial subtraction model to the average performance across subjects (Ritter, et al. 2007). Analysis by ACT-R parameter type shows that the value of SYL remained relatively stable at 0.45 in four of the five solutions. The values for ANS could be considered relatively high according to the ACT-R modeling community, greater then 0.5, for all but the last genotype listed in the table, 0.25. Values for ANS between the default (NIL) and less than 0.5 are more typically reported in cognitive modeling research. Notice that the lowest BLC value (1.59) is associated with the lowest ANS value (0.25). The difference between the lowest ANS and BLC pair compared to the next higher up pair is substantial: for ANS, the difference between 0.25 and 0.64 is 0.51; for BLC, the difference between 1.59 and 2.43 is 0.84. Also note that the lowest ANS and BLC pair produced the highest fitness out of this group of genotypes with the model's prediction for the number of attempts which is off by 0.8, compared to the ANS and BLC pair producing the lowest fitness value with the number of attempts which is off only by 0.3. Overall the value of BLC appears to increase as the value of ANS increases. This may be a pattern worth investigating in the next two levels (group and individual) of optimizations. The two genotypes in Table 4.3 with the lowest fitness values (0.1652 and 0.4252, difference of 0.26) are equivalent in ACT-R

parameter value sets. In both these solutions the number of attempts is off by 0.3, and the percentage correct is off by 0.2 for the genotype in row 1 and off by 0.6 for the genotype in row 2.

The computational resources to optimize the model to the average performance across subjects was 200 processors, one for each genotype in the population, and approximately 137 minutes of runtime on the cluster. The runtime includes the post-PGA testing where genotypes collected throughout the generations are run again on all the processors to compute an average fitness from across the model runs.

Fitting to Appraisal Groups

The next level of optimization is to compute the parameters that predict individuals grouped by task appraisal. During the human subject experiment, pre- and post-task appraisals were assessed before and after the four blocks of serial subtraction. Based on these self-reported appraisals, subjects were categorized into one of two appraisal groups: challenge (N=7) or threat (N=8). Subjects in the challenge group felt that they could cope with the stressfulness of the serial subtraction task and perform the task; whereas, subjects in the threat group felt that the task was too stressful and that they lacked the coping ability required to perform the task.

The effects that stress and anxiety might have on cognitive performance, and specifically on mathematics performance, was discussed in detail in Chapter 2. It is thought that mental arithmetic tasks such as the serial subtraction task require working memory resources for problem solving. Stress related states (anxiety, worry) are thought to reduce working memory capacity that is required to perform mental arithmetic, thus, decreasing performance. Table 4.4 shows the subtraction rates for individual subjects grouped by post-task appraisal. Notice that the threat group underperformed the challenge group of subjects in both number of attempts and percentage

correct. The challenge group on average completed 15-25 more subtraction problems per block than did the threat group. The difference is greater for percentage correct where the challenge group completed 7-8% more correct subtraction problems than the threat group. The results in Table 4.4 appear to support the theories presented in Chapter 2; that stressful states negatively affect cognitive performance on mental mathematics tasks.

Table 4.4: Mean performance and standard deviation by post-task appraisal group.

7s – 1st block	Threat (N=8)	Challenge (N=7)
Number of Attempts	40.3 (10.1)	55.3 (16.7)
Percent Correct	78.1 (8.2)	85.4 (10.8)
7s – 2nd block		
Number of Attempts	44.8 (10.2)	70.7 (23.7)
Percent Correct	84.2 (4.6)	92.5 (6.2)

Two independent-samples t-tests were conducted to compare number of attempts and percentage correct scores for challenge and threat appraisal groups. There was a significant difference in number of attempts for appraisal $t(13) = 2.14$, $p=0.05$. There was no significant difference in percentage correct for appraisal $t(13)=1.49$, $p=0.16$. In addition, a between-groups multivariate analysis of variance was performed with number of attempts and percentage correct as dependent variables and appraisal as an independent variable. There was no significant difference between appraisal groups on the combined dependent variables: $F(2,12)=2.17$, $p=0.16$.

The same PGA that optimized average performance across subjects was run on the NCSA cluster to optimize the serial subtraction model to the appraisal group performance statistics of the first block of subtracting by 7s listed in Table 4.4. Two PGAs were set up to run 100 generations of 200 binary-encoded genotypes using 200 processors. Each PGA tried a total of 20,000 ACT-R parameter combinations over the generations. One PGA optimized the model to the challenge appraisal group means (55.3 attempts, 85.4% correct), and the other to the threat appraisal group means (40.3 attempts, 78.1% correct). The PGAs used genotypes composed of

the same three ACT-R parameters (ANS, BLC, SYL). Again one processor was allocated for each genotype. The selection probability and mutation rate were unchanged from the previous optimization of average performance. The terminating condition was also maintained at 100 generations. The same fitness function was used—comparing the sum of squares error for the model's predicted number of attempts and percentage correct to the corresponding human data.

Figures 4.2 and 4.3 plot the progress of the two PGAs. Figure 4.2 shows the minimum and average fitness when optimizing to the challenge group. Figure 4.3 shows the minimum and average fitness when optimizing to the threat group.

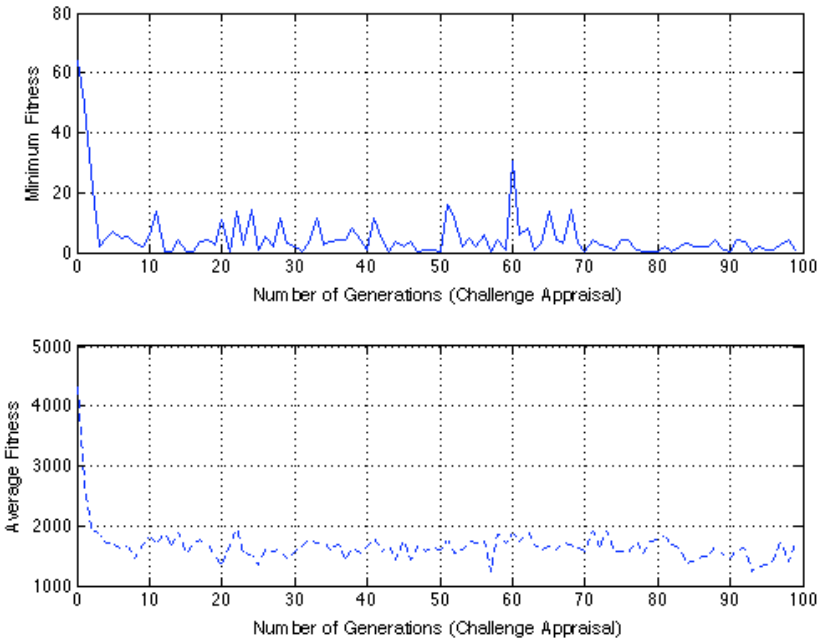

Figure 4.2: Progress of PGA optimization to challenge appraisal group average (N=7) showing minimum fitness value (top) and average fitness (bottom) across the generations running 100 generations of 200 genotypes using 200 processors.

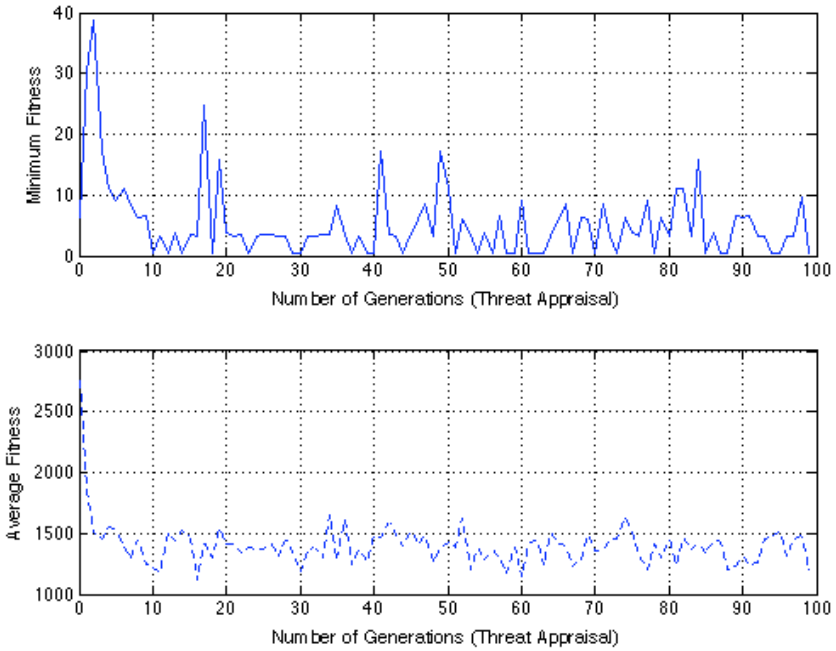

Figure 4.3: Progress of PGA optimization to threat appraisal group average (N=8) showing minimum fitness value (top) and average fitness value (bottom) across the generations running 100 generations of 200 genotypes using 200 processors.

As explained in Chapter 3 in the PGA development section, what would be expected in this type of plot is a curvilinear downward sloping line as the PGA converges on a solution. The nonmonotonic patterns in these plots suggest that the stochastic characteristics of the cognitive architecture and model inhibit convergence. The PGA compromises for lack of convergence by collecting up the best fitting genotypes as it evolves generations and then validating the fitness of these genotypes in a separate testing procedure at the end of the PGA code.

The plots from the threat optimization show greater variability between peaks and valleys than the challenge optimization plots. In the challenge minimum fitness value plot the nonmonotonic pattern starts to flatten out around generation 70. This is not the case in the threat minimum fitness value plot where the pattern does not appear to reach an asymptote. In Table 4.5 the optimization results show that the PGA was able to find multiple good solutions for each optimization: from the challenge optimization 10 solutions and from the threat optimization four

solutions. Possibly the level of variability in the minimum fitness value across the generations is indicative of the difficulty in finding good solutions in the ACT-R parameter space.

Table 4.5 is a summary of the PGA optimization results for the appraisal groups. The top half of the table contains the results from the challenge optimization. The bottom half of the table contains the results from the threat optimization. For each appraisal group the results are ordered by post-PGA testing fitness value.

Table 4.5: PGA optimization results for fitting to the average by appraisal group using 200 genotypes over 100 generations on 200 processors. Human performance (number of attempts and percentage correct, rounded) is compared to model predictions (number of attempts and percentage correct, rounded), and fitness value with corresponding genotype (ANS, BLC, SYL).

Human Performance (Avg by appraisal)	Model Prediction	Fitness Value	Genotype (ACT-R parameters)
Challenge 55.3, 85.4	55.2, 85.4	0.0073	0.60, 2.44, 0.37
	55.4, 85.5	0.0293	0.79, 2.84, 0.37
	55.3, 85.6	0.0481	0.79, 2.84, 0.37
	54.9, 85.3	0.2138	0.39, 1.87, 0.36
	54.8, 85.3	0.2388	0.64, 2.44, 0.37
	55.7, 84.9	0.3800	0.58, 2.35, 0.36
	55.1, 84.7	0.6122	0.61, 2.38, 0.37
	55.5, 86.2	0.6700	0.68, 2.54, 0.36
	54.6, 85.9	0.7594	0.70, 2.61, 0.37
	55.2, 86.4	0.9765	0.38, 1.82, 0.36
Threat 40.3, 78.1	40.7, 78.3	0.1948	0.64, 2.44, 0.54
	40.7, 77.7	0.3671	0.76, 2.60, 0.54
	41.0, 78.1	0.4508	0.59, 2.28, 0.53
	40.0, 77.3	0.7462	0.89, 2.86, 0.55

When considering the effects of stochasticity in the cognitive architecture and model and the resulting instability in performance predictions, genotypes (i.e., ACT-R parameter sets) producing a fitness less than 1.0 are good model-to-data fits. When optimizing to the challenge group performance, the PGA collected 10 genotypes with fitness values less than 1.0. The fitness values ranged from almost 0 (0.0073) to nearly 1 (0.9765) with a difference of 0.9692. When optimizing to the threat group performance, the PGA collected four genotypes with fitness less

than 1.0 ranging from 0.1948 to 0.7462, and less of a difference in fitness value, i.e., 0.5514. The results in Table 4.5 show that the thesis optimization was able to produce good model to data fits for both appraisal groups.

Analysis of the results by ACT-R parameter type shows how cognition is different between these two groups. In the challenge optimization results, SYL shows similar value stability as in the previous optimization across all subjects taking on the value of either 0.36 or 0.37. Six of the ten best fitting genotypes had SYL 0.37, and for the remaining four genotypes SYL was 0.36. There is a concentration of 0.37 SYL values at the top of the challenge section in the table corresponding to the lower fitness values. In the threat optimization results, SYL is slightly less stable than the challenge group with a range of SYL values from 0.53 to 0.55. The genotype in the last row with the highest fitness (0.7462) has the highest SYL (0.55).

The value of SYL represents seconds per syllable in speaking the answer in ACT-R's vocal module. The SYL values for the challenge group (0.36, 0.37) are less than the values for the threat group (0.53, 0.54, 0.55). SYL values for the average across subjects optimization (0.44, 0.45) fall in between the appraisal groups' SYL values, which would be expected. The human performance data shows that the threat group performed fewer attempted subtraction problems and had a lower percentage correct than the challenge group. The SYL values when compared by appraisal group show that the challenge group is speaking a syllable much more quickly than the threat group. The human subjects speak the answer to each subtraction problem; likewise, the model simulates verbalizing the subtraction answers. Within group comparisons of SYL may be too fine-grained with differences of only 0.01 to show patterns. If the model had only been optimized to the average performance across subjects, the relationship of speech rate to performance would not have been detected because SYL values for the average across subject optimization varied by 0.01 in only one out of five best fitting genotypes.

ANS is ACT-R's activation noise parameter effecting subsymbolic processes in the declarative memory module. In the challenge optimization the range of ANS is 0.38 to 0.79 with a difference of 0.41. Again, the majority of the values for ANS are considered high and only 0.02 less than the highest ANS value from the average across subjects optimization. Only two ANS values from the challenge group are less than 0.5 and their placement in Table 4.5 is associated with the genotype with the highest fitness (bottom of challenge group in table, 0.38), and a genotype slightly above midway in the column (0.39). In the threat optimization the range of ANS is less, 0.59 to 0.89, with a difference of 0.31. Here the highest ANS (0.89) exceeds both the highest ANS in the average across subject optimization (0.81), and the challenge group optimization (0.79). In fact, the value 0.89 for ANS is only 0.01 from the maximum boundary value for ANS (9.0) in the PGA's defined search space. The 0.89 ANS is associated with the genotype with the highest fitness in the last row of Table 4.5.

BLC contributes to the base level activation of chunks in declarative memory. The BLC range in the challenge optimization is 1.82 to 2.84 a difference of 1.02; compared to the BLC range in the threat optimization, 2.28 to 2.86, with only a difference of 0.58. BLC values less than 2.0 are found in both average across subjects and the challenge optimizations, but not in the threat optimization. All BLC values are greater then 2.0 in the threat optimization which coincides with this group's high ANS values, all being greater than 0.5. This instantiates the simultaneously increasing/decreasing of ANS and BLC pairs pattern observed in the average over subjects optimization results. This pattern is especially evident if Table 4.5 is ordered by ANS value from low to high within each appraisal group, instead of fitness value. Table 4.6, a reordered Table 4.5, shows that as ANS values increase BLC values also increase with only one exception—rows 5 and 6 in the challenge group. Another interesting observation, slightly lower SYL values appear to be associated with lower ANS values. Possibly less noise enables slightly faster speech.

Table 4.6: A resorting of Table 4.5: ordered within appraisal group by ANS value from low to high showing the relationship between ANS and BLC.

Human Performance (Avg by appraisal)	Model Prediction	Fitness Value	Genotype (ACT-R parameters)
Challenge 55.3, 85.4	55.2, 86.4	0.9765	0.38, 1.82, 0.36
	54.9, 85.3	0.2138	0.39, 1.87, 0.36
	55.7, 84.9	0.3800	0.58, 2.35, 0.36
	55.2, 85.4	0.0073	0.60, 2.44, 0.37
	55.1, 84.7	0.6122	0.61, 2.38, 0.37
	54.8, 85.3	0.2388	0.64, 2.44, 0.37
	55.5, 86.2	0.6700	0.68, 2.54, 0.36
	54.6, 85.9	0.7594	0.70, 2.61, 0.37
	55.3, 85.6	0.0481	0.79, 2.84, 0.37
	55.4, 85.5	0.0293	0.79, 2.84, 0.37
Threat 40.3, 78.1	41.0, 78.1	0.4508	0.59, 2.28, 0.53
	40.7, 78.3	0.1948	0.64, 2.44, 0.54
	40.7, 77.7	0.3671	0.76, 2.60, 0.54
	40.0, 77.3	0.7462	0.89, 2.86, 0.55

Figure 4.4 compares the parameter ranges for ANS, BLC, and SYL for optimization results when fitting to average performance across subjects, and each of the appraisal groups. The y-axis groups the ranges by parameter type, and the x-axis represents the actual parameter values. It would be expected that the parameter ranges for the average across subjects optimization would fall in between the appraisal group parameter ranges. In Figure 4.4, the threat appraisal group parameter ranges extend slightly beyond the average across subject parameter ranges for BLC and more so for ANS. The average across subjects SYL lies in between the threat and challenge SYL, which is not surprising. The challenge appraisal group parameter ranges for BLC and ANS overlap most of the average across subjects parameter ranges with the exception of the lower end of the range for both BLC and ANS.

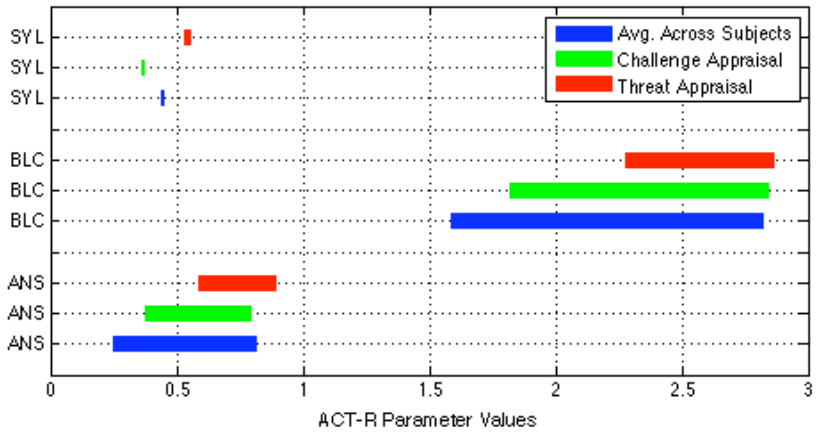

Figure 4.4: Comparison of parameter ranges of good fits between average across subjects optimization and appraisal group optimizations by parameter type on the y-axis.

In summary, the model to data fits for both appraisal group optimizations are all very good with model predictions to within a fraction of a subtraction problem for both number of attempts, and within 1% or less for percentage correct. The fourth row of Table 4.6 in the challenge group optimization results shows the amazing fit of 0.0073 with model predictions short by only 0.1 for number of attempts. Finding a fit at this level of accuracy would not be possible using manual optimization especially considering that two performance statistics are being fitted simultaneously.

The computational resources to optimize the model to the appraisal groups were 200 processors, one processor allocated to each genotype in the population. Runtime for the challenge group optimization was approximately 148 minutes. Runtime for the threat group was slightly less, 133 minutes. There were more good solutions found during the challenge group optimization to test in the post-PGA testing function which may account for the runtime difference.

Fitting to Individual Subjects

After successfully optimizing to groups of subjects defined by the challenge and threat task appraisal, the next phase of the project was to compute the model fits for each individual subject. Developing an optimization approach that automates the model fitting process and produces accurate fits enables the investigation of individual differences in cognitive performance. In theory the thesis optimization technique could be used to fit any cognitive model written in the ACT-R architecture to any level of human performance data with only minor modifications. Reformatting a model's startup function to accept parameter values coming in as arguments would be one such modification. The primary purpose of this section is fitting the model to each of the 15 subjects in the control treatment group. This means one optimization would be run on the cluster for each subject the model should fit.

Basically the same format was used here as in the previous two levels of optimization. Fifteen PGAs were set-up to run 100 generations of 200 binary-encoded genotypes on 200 processors. Each PGA optimized the serial subtraction model to an individual subject's performance data gathered from the experiment. The PGA used genotypes composed of the same three ACT-R parameters (ANS, BLC, SYL). Again one processor was allocated for each genotype. The selection probability and mutation rate were unchanged from the previous two optimizations. The terminating condition was 100 generations. The same fitness function was used, i.e., comparing the sum of squares error for the model's predicted number of attempts and percentage correct to the corresponding human data. Once again the PGAs were seeking a global minima within the ACT-R parameter space during optimization.

In this section the optimization results of three of the 15 subjects is examined visually by minimum and average fitness plots. Then the results for all the subjects are presented in table format and summarized. Three particular subjects, represented by subject ID# assigned during the

experiment, were selected as analysis examples because they represent the wide range of performance on the serial subtraction task. The subjects selected include: subject 21, performed second to best in both number of attempts and percentage correct; subject 47, performed the worst in percentage correct; and subject 27, with performance falling in between subjects 21 and 47 and close to the average performance across all the subjects. Table 4.7 lists the human performance data (number of attempts and percentage correct) for each of these subjects.

Table 4.7: Data from three subjects performing the first 4-minute block of serial subtracting by 7s.

	Subject 21	Subject 27	Subject 47
Number of Attempts	65	46	29
Percent Correct	90.77	86.96	62.07

Figures 4.5a, b, and c show the accompanying fitness value plots for the PGA optimizations of subjects 21, 27, and 47, respectively. Each subject is represented by a pair of plots with the top plot showing the minimum fitness value across generations and the bottom plot showing the average fitness value across generations. In the final generation, the population is not mutated; this causes the sudden drop in each of the average fitness plots.

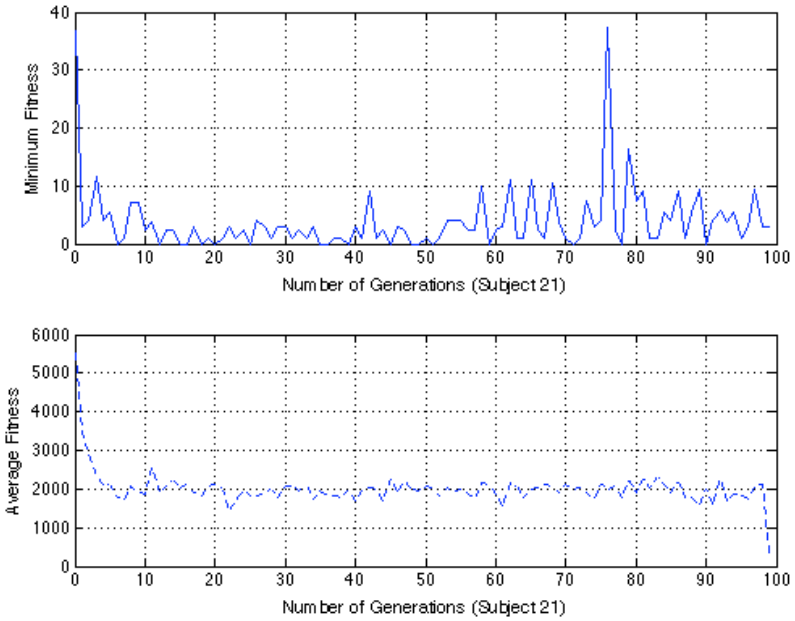

Figure 4.5a: Progress of PGA when searching for a global minima in the ACT-R parameter space when optimizing to subject 21 running 100 generations of 200 genotypes using 200 processors. The top plot shows minimum fitness value, and the bottom plot shows average fitness value.

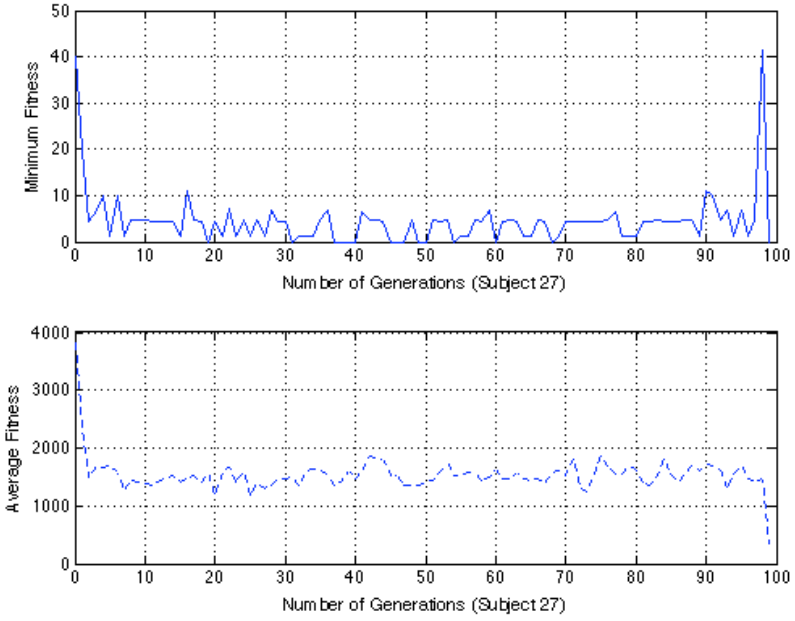

Figure 4.5b: Progress of PGA when searching for a global minima in the ACT-R parameter space when optimizing to subject 27 running 100 generations of 200 genotypes using 200 processors. The top plot is minimum fitness value, and the bottom plot shows average fitness value.

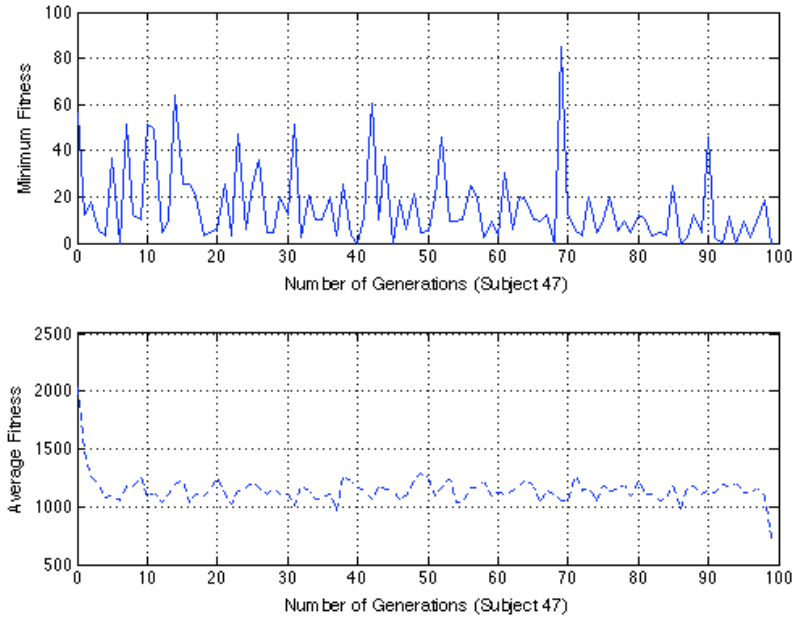

Figure 4.5c: Progress of PGA when searching for a global minima in the ACT-R parameter space when optimizing to subject 47 running 100 generations of 200 genotypes using 200 processors. The top plot shows minimum fitness value, and the bottom plot shows average fitness value.

In comparing the minimum fitness value plots (top plot in Figures 4.5a, b, and c), the nonmonotonic pattern is least variant for subject 27's optimization (close to average performance), and most variant for subject 47's optimization (the worest performing subject). The nonmonotonic pattern for subject 21 appears to fall in between the other two patterns. The pattern for subject 47 appears similar to the minimum fitness value plot for the threat optimization (Figure 4.3) but more erratic in fluctuations between peaks and valleys. In subject 47's plot a difference between a peak and valley value can range as much as 80 within a span of three generations. In the lower plot of Figures 4.5a, b, and c, the average fitness decreases by nearly 3000 in the first several generations and then levels off in a more shallow nonmonotonic pattern between 1000 and 2000 in the subjects' optimizations.

In general, the PGAs appear to be able to find genotypes resulting in fitness values close to 0 at some point or several points during the 100 generations. As discussed previously the stochasticity in the architecture contributes to the nonmonotonic behavior of the minimum fitness

value and results in the lack of convergence on a solution set. The PGA testing phase was added to ameliorate this lack of convergence.

Table 4.8 is a summary of the PGA optimization results for all 15 subjects. The table only shows one solution (genotype), the one with the lowest fitness that was found by each PGA optimization. The optimization results are ordered by human performance on number of attempts from worst performance to best performance by corresponding subject ID#.

Table 4.8: Best solutions found for each subject using PGA optimization with 200 genotypes over 100 generations on 200 processors. Human performance (number of attempts and percentage correct, rounded) is compared to model predictions (number of attempts and percentage correct, rounded), and fitness value with corresponding genotype (ANS, BLC, SYL).

Subject	Human Performance	Model Prediction	Fitness Value	Genotype (ACT-R parameters)
1	28, 67.9	28.0, 67.8	0.0006	0.83, 2.76, 0.87
47	29, 62.1	29.3, 62.0	0.0866	0.66, 2.25, 0.83
25	31, 80.7	30.8, 80.8	0.0487	0.48, 2.25, 0.76
11	35, 65.7	34.5, 65.1	0.6836	0.82, 2.49, 0.69
14	37, 75.7	36.3, 75.8	0.5523	0.83, 2.75, 0.62
2	37, 78.4	36.2, 78.6	0.7682	0.81, 2.80, 0.63
46	45, 80.0	44.7, 80.4	0.2510	0.43, 1.90, 0.47
27	46, 87.0	46.1, 87.7	0.4917	0.76, 2.96, 0.46
16	50, 92.0	50.4, 92.3	0.2233	0.50, 2.46, 0.41
43	54, 89.0	53.9, 89.0	0.0214	0.72, 2.88, 0.38
41	55, 87.3	55.2, 86.8	0.2261	0.54, 2.32, 0.36
23	57, 84.2	56.8, 84.4	0.0744	0.79, 2.71, 0.35
9	57, 87.7	57.2, 87.1	0.4089	0.78, 2.92, 0.35
21	65, 90.8	64.8, 91.2	0.1997	0.53, 2.24, 0.29
26	83, 94.0	83.3, 94.2	0.1463	0.47, 2.14, 0.16

The PGA optimizations for individual subjects found good fits for all the subjects with fitness values in the range 0.0006 to 0.7682 with a difference of 0.7676. When comparing the human performance column to the model prediction column, both number of attempts and percentage correct were fit to the human data to within a fractional part of a subtraction problem for all subjects. It is surprising that the lowest fitness value (0.0006) in Table 4.8 corresponds to the subject with the worst performance by number of attempts (subject 1). In the minimum fitness

value plots (Figures 4.5a, b, and c) the greatest variations in the nonmonotonic pattern occurred during the optimization of subject 47, the lowest performer by percentage correct. The nonmonotonic patterns characterizing prediction instability did not appear to inhibit the optimization approach from finding nearly perfect fits in the ACT-R parameter space for the worst performing subjects. In fact, there is a cluster of excellent fits, the first three rows in the Table 4.8, for the poor performers (0.0006, 0.0866, 0.0487). In the next three rows, there is a cluster of high fits (0.6836, 0.5523, 0.7682), with row 6, subject 2's optimization, producing the worst fit out of all the subjects.

Analysis by parameter type across the table is as follows. The SYL parameter value shows a nearly perfect increase as performance decreases pattern, with the exception of subjects 14 and 2 that are misordered by 0.01. The range for SYL in the individual optimizations is a speedy 0.16 to a slow-speaking 0.87, a substantial difference of 0.71. SYL values from all the other optimizations (average across subjects, and appraisal groups) fall within the middle of this range. Any information deduced from the large range of SYL values uncovered by the individual optimizations would have been lost if only the upper two levels of optimizations would have been performed (average across subjects, and appraisal groups). This emphasizes the importance of investigating individual differences in cognition.

The range for ANS in the individual optimizations is 0.43 to 0.83 with a difference of 0.4. This is less than the range for ANS in the average across subjects optimization (0.25 to 0.81; difference 0.56). In the individual optimizations the value of ANS did not reach the lower values. It might be the case that the lowest overall ANS value (0.25) found in the average across subjects optimization is somewhat of an anomaly. In the individual optimizations with the exception of the lowest ANS (0.43) all other ANS values were greater than or equal to 0.5. The highest ANS in the individual optimizations (0.83) was not the highest overall ANS value. The highest overall ANS value occurred during the threat group optimization (0.89).

BLC values across the table of individual optimizations show only one value under 2.0; subject 46, one of the average performers, has a BLC of 1.90. This subject also has the lowest ANS value (0.43). This is the simultaneously increasing/decreasing pattern of the ANS and BLC pair of parameters. The range for BLC across the individual optimizations is 1.90 to 2.96, which corresponds to a difference of 1.06. This difference in BLC is similar to the challenge optimization difference in BLC of 1.02. The individual optimizations' BLC range is shifted up slightly in comparison to the BLC range of the challenge optimization (1.82 to 2.84). The lowest BLC value was produced when optimizing to subject 46, row 7 in the table. In row 8, the optimization for subject 27 produced the highest BLC value, 2.96. This value is very near the parameter space boundary limit of 3.0 for BLC.

Subjects 46, 27, and 16 could be considered median performers when compared to average human performance on the task across all the subjects. These subjects are positioned in the middle of Table 4.8 in rows 7 to 9. The average number of attempts and percentage correct calculated across all subjects is 47.3 and 82%, respectively. The average 47.3 in number of attempts falls between subject 27 and subject 16 in number of attempts (46, 50). The average 82% falls between subject 46 and subject 27 in percentage correct (80%, 87%). Except for SYL values within the 0.40s, the optimization solutions for these subjects do not appear to exhibit any other similarities.

Computational runtime on the cluster to optimize the model to individual subjects using 200 processors ranged from 112 minutes to 176 minutes. The two worst performing subjects from Table 4.8, subject 1 and 47 had the shortest runtimes, 113 and 112 minutes, respectively. The best performing subject, 26, had the longest runtime (176 minutes). From the patterns of minimum fitness values observed in the previous optimization plots, it is most likely that during shallow nonmonotonic behavior, the PGA is able to find and collect up more good genotypes (solutions) while searching the parameter space. This was confirmed in the challenge versus threat

optimizations where 10 solutions were found in the challenge optimization and only four were found in the threat optimization. Each saved genotype is validated in the post-PGA testing phase of the code. The more good solutions collected up by the PGA the more runtime needed to validate them; therefore, increasing the amount of runtime on the cluster.

Summary

The goal of this thesis was to develop a more efficient and accurate optimization approach enabling the investigation of individual differences in serial subtraction performance due to stress. The serial subtraction task, characterized as a stressful task, produced a wide range of human performance necessitating an individual subject level of investigation. The optimization results in this chapter confirm that the thesis optimization approach can produce accurate and efficient model-to-human data fits not only at the individual level of analysis, but also at multiple levels of analysis. This automated (modeler-free) optimization approach has the added advantage of eliminating modeler bias that most certainly exists in the predominant manual optimization process.

The thesis optimization approach was tested by optimizing to three different levels of human data: average across subjects, challenge and threat appraisal groups, and individual subjects. At all three levels of data fitting the thesis optimization approach found nearly perfect solutions in what is thought to be a complex parametric search space. The resulting solutions in the form of ACT-R parameter sets adjusted the serial subtraction model's predictions to match the human performance data.

The optimization solutions generated by the PGA revealed three interesting patterns in the ACT-R parameter sets found to produce best fits to the human data. The optimization results showed that a large portion of the ANS (activation noise parameter) values were greater than 0.5

which is considered ‘high’ by the cognitive modeling community. For example, 60% of the values for ANS in Table 4.8 are substantially above 0.5. ANS values below 0.5 were very few and scattered among the different levels of optimization solutions. The average across subjects optimization found one low ANS value (0.25). The challenge optimization found two solutions with lower ANS values (0.38, and 0.39). In the individual optimizations’ best fitting genotypes, three instances of low ANS values in the 0.40s were found: 0.48, for subject 25, a poor performer; 0.43, for subject 46, an average performer; and 0.47, for subject 26, the best performer. These lower range ANS values were accompanied with low BLC values that positioned these solutions in a much different region of the ACT-R parameter space then the majority of the solutions.

One of the low ANS values appeared as part of a solution when optimizing to subject 26, the best performing subject in the control treatment group. During a brief retrospective interview, subject 26 reported that he used a ‘pattern-like’ strategy in performing the serial subtraction task instead of actually subtracting by 7 or 13. The model performs the serial subtraction task by subtracting. No other subjects were interviewed as to their task strategy. It might be the case that the small set of solutions characterized by lower ANS and BLC values were the result of strategy discrepancies between how the model was procedurally performing the task and how the individual subjects were actually performing the task.

Another consideration, the proportionally large number of high ANS values could be related to the fact that the serial subtraction task itself is very stressful. In psychology and physiology studies this task is specifically used to produce a stressful response in human subjects. The variability in the human performance and the high-ANS solutions may be indicative of noise that the model is not attempting to explain and that is usually associated with individual differences. In this case, if the manual optimization was applied in fitting the model to data averaged across subjects, large ANS values would not have been used in model fitting attempts

and the distinction between high and lower sets of ANS values would have gone unnoticed. This is an example of how the PGA's non-bias randomized search of the ACT-R parameter space could inform theory building in the cognitive modeling field.

A second pattern, the simultaneously increasing/decrease of ANS and BLC parameter value pairs, was observed at all three levels of optimization. The BLC parameter is a component in the base level activation of chunks in declarative memory. Base level activation is a component in the source activation equation part of the declarative module's subsymbolic processes. Source activation was utilized by a group of researchers (Daily, Lovett, & Reder, 1999, 2001; Lovett, Daily, & Reder, 2000; Lovett, Reder, & Lebiere, 1997; Rehling, Lovett, Lebiere, Reder, & Demiral, 2004) to represent working memory capacity (discussed in Chapter 2). Source activation represents a type of attentional activation that is divided equally among the items in the current focus of attention. It spreads from these items to related chunks that are necessary for task performance and in this way maintains those task-relevant chunks in an available state relative to the rest of declarative memory. Because the amount of source activation is limited in quantity and because this quantity is divided among the various items in the current focus of attention, the more items in the focus, the less source activation each can spread to its related chunks. For example, a complex task that requires a greater number of items in the focus of attention implies that each item in the focus of attention has a smaller share of source activation to spread to task-relevant chunks.

These factors are important for the serial subtraction model as all numbers and subtraction facts are represented as chunks in declarative memory, and the goal is to perform a specific subtraction problem. ANS and BLC are obviously related by way of source activation and its effect on chunk retrieval in declarative memory. An interpretation of how ANS and BLC values might be reflective of stressful or anxious states of specific subjects and their working memory capacity is discussed in the next chapter.

The third pattern involves SYL, the seconds per syllable parameter. SYL controls the rate of speech in ACT-R's vocal module. The human subjects speak the answer to each subtraction problem; likewise, the model simulates verbalizing the subtraction answers through the vocal module. The optimization results showed that as performance decreased on serial subtraction, the value of SYL increased. This means that high performing subjects speak a syllable more quickly than their poor performing counterparts. This was an especially visible pattern in the individual subject optimizations (Table 4.8).

One interpretation associated with cognitive psychology literature is the worry concept and its link to inner verbal activity (Rapee, 1993). In the ACT-R cognitive architecture inner verbal activity would be the responsibility of the vocal module. It might be that a subject's worry (inner verbal activity) over his performance is interfering or competing with his speaking of the subtraction answers with the effect of generally decreasing subtraction performance. Another interpretation is related to working memory capacity. From Chapter 2, Baddeley's (1986, 1992) working memory model is composed of three units or systems. One of the slave-systems to the executive control system is the articulatory loop where speech and sound-related information is rehearsed. Each slave system relies on its own pool of attentional resources (Baddeley & Hitch, 1974). If tasks being performed do not overextend attentional resources of the systems then working memory performance is unaffected. If one of the systems becomes overloaded, it either fails to complete the task or draws resources from the executive control system resulting in impaired working memory performance. During the audio transcriptions of the subjects' performance, the subtraction answers appeared to be spoken as the subject was in the process of calculating the remainder of the answer. In this case subjects could only speak the answer as quickly as it was being solved in their working memory; meaning that slow speech is reflective of slow subtraction calculations.

A key component in the success of the optimizations reported in this chapter is the use of a cluster computing resource. This type of resource is commonly available in medium to large sized universities. The population size and number generations evolved could be modified to suit the resources available (number of processors). As the optimization results reported in this chapter indicate, the PGA is an effective global search technique. With 200 processors and approximately two hours of runtime on the cluster, the PGA optimized the serial subtraction model to an individual subject's performance data. Totaling approximately a day and a half of time on the cluster, the PGA optimized the model to all 15 subjects in the control treatment group of the experiment.

Chapter 5

INTERPRETATIONS, VISUALIZATIONS, AND DISTRIBUTIONS

The optimization results and implications of the results presented in Chapter 4 confirm that the thesis optimization approach using a HPC platform and a PGA is a success. If this new approach to fitting cognitive models to human data is adopted by cognitive modelers replacing the long-standing manual optimization process, theory building within the field would undoubtedly progress at an accelerated pace. Although this is only an optimization approach, its potential as an architectural parameter and prediction exploration tool nearly exceeds it primary purpose of optimization for the sake of validation. There are over 80 available parameters to manipulate model behavior within the ACT-R architecture. Only a very small subset would be used in any one cognitive model; nevertheless, the combinatory complexity and size of the search space appears daunting without the assistance of an automated search method. Doubling as an optimization technique and an exploratory development tool, the thesis optimization approach can efficiently search ACT-R parameter spaces seeking promising prediction areas within the bounds of any proposed hypothesis.

The topics covered in this chapter include: additional interpretations of patterns in the optimization solutions presented in Chapter 4 in reference to working memory and theories of stress; preliminary 3-D visualizations of the ANS, BLC, and ANS parameter space in reference to the worst and best performing subject; and distributions of model predictions created from 200 model runs and then overlaid to enable a comparison between optimizing to individual subjects and optimizing to an average across subjects.

Working Memory and Stress Interpretations

Key cognitive modeling goals are prediction and understanding. Experimental data can reveal individual differences in cognitive processes if they are not averaged across subjects. Simulating these cognitive processes by way of modeling and parametric manipulation can provide insight and explanation of the underlying psychological phenomenon. The thesis project proposes a more efficient and accurate optimization approach freeing modelers of the time spent in trial-and-error parameter fitting efforts. Automating the model fitting process allows more emphasis to be placed on understanding the cognitive differences observed in the data and building new theory to reflect this understanding.

Several patterns in ACT-R parameter values were observed in the optimization solutions presented in Chapter 4. In ACT-R a particular parameter has a specific meaning in application of subsymbolic processes within a particular module in the architecture. The parameters and their values affect how the model runs in the architecture and likewise its predictions of human performance. When manipulating only one parameter the resulting effects on model behavior are generally interpretable. However, a weakness in multi-parametric manipulation is uninterpretability of the parameters' combined effect on the model's cognitive processes.

For this project three different parameters were manipulated in the model in the hopes of understanding how stress influences performance on the serial subtraction task. Several of the studies described in Chapter 2 yielded a common conclusion concerning suboptimal performance arising in mathematics problem solving. These studies involved working memory, a short-term system that maintains, in an active state, a limited amount of information with immediate relevance to the task at hand while preventing distractions from the environment and irrelevant thoughts (Hasher, Lustig, & Zacks, 2008; Kane & Engle, 2000, 2002). Ashcraft and Kirk (2001) suggested that anxiety generates intrusive worries about the situation that occupy the part of

working memory capacity normally devoted to skill execution. Schmader and Johns (2003) and Steele (1997) argued that stereotype threat interferes with performance by consuming or reducing working memory capacity that individuals need to perform successfully. Beilock and colleagues (Beilock & Carr, 2001, 2005; Beilock, Kulp, Holt, & Carr, 2004) found support for distraction theories of 'choking under pressure', according to which, like anxiety, pressure creates mental distractions that compete for reduced working memory capacity that would otherwise be allocated to skill execution.

Working memory in ACT-R has been defined in two ways: as a subset of highly active elements of declarative memory or as a process of spreading source activation (i.e., attentional resources) from the current goal to declarative elements strongly linked with that goal (Chuderski, Stettner, & Orzechowski, 2006). The majority of working memory modeling in ACT-R has used the concept of spreading source activation with some of this modeling focused on individual differences. For example, Lovett, Reder and Lebiere (1997) used the *W* quantity in ACT-R version 4.0 as a parameter to control the amount of spreading activation that was associated with working memory capacity. Individual differences in working memory capacity were modeled by varying *W*. They assumed that the limit on *W* is not the same for each individual and that different individuals have different *W* values. The higher the value of *W*, the higher task-relevant chunks' total activations will be relative to the rest of declarative memory. These activations in turn impact the probability and speed of successful retrieval of the task-relevant chunks; a higher *W* facilitates the retrieval process by increasing spreading activation. Lovett, Reder and Lebiere's (1997) working memory model predicted that an individual with a larger *W* value would be able to retrieve task-relevant information more accurately and more quickly than an individual with a smaller value of *W*.

ACT-R uses a common formula in activation theories, the activation of a chunk is a sum of a base-level activation, reflecting its general usefulness in the past, and an associative

activation, reflecting its relevance to the current context (Anderson et al., 2004). The activation of a chunk controls both its probability of being retrieved and its speed of retrieval. The activation of a chunk *i* (A_i) is defined as

$$A_i = B_i + S_i + P_i + \varepsilon \qquad \text{(Activation equation, 5.1)}$$

with B_i as the base-level activation of chunk *i,*. S_i the spreading activation value computed for chunk *i* representing the impact that the contents of the buffers have on the retrieval process, P_i as the partial matching value computed for the chunk that reflects the degree to which the chunk matches the specification requested, and ε as a noise value consisting of a fixed noise and a transient noise.

The base-level activation is calculated by one of three equations depending on the setting of parameters BLL and OL. In the serial subtraction model BLL was set to 0.5 and OL was left at the default value. Therefore, the approximation form of the equation was used for base-level activation

$$B_i = \ln(n/(1-d)) - d * \ln(L) + \beta_i \qquad \text{(Base-level activation, 5.2)}$$

The number of presentations of chunk *i* is *n*. The lifetime of chunk *i* is *L* (time since its creation). The decay parameter is *d*. A constant offset β_i is determined by parameter BLC.

When enabled, spreading activation to a chunk is computed using

$$S_i = \sum_k \sum_j W_{kj} S_{ji} \qquad \text{(Spreading activation equation, 5.3)}$$

The elements *k* are summed over all of the buffers in the model. The elements *j* are summed over the chunks that are in the slots of the chunk in buffer *k* (referred to as the sources of activation). The amount of activation from source *j* in buffer *k* is W_{kj}. It is the source activation of buffer *k*

divided by the number of sources *j* in that buffer. The strength of association from source *j* to chunk *i* is S_{ji}.

The third component in the activation equation is for partial matching of chunks. In the serial subtraction model the partial matching process was not enabled; therefore, the value of P_i is defaulted to 0.

The noise component of the activation equation contains two sources of noise: a permanent noise that can be associated with a chunk and an instantaneous noise value that is recomputed at each retrieval attempt. Both noise values are generated according to a logistic distribution characterized by a parameter *s*. The mean of the logistic distribution is 0 and the variance, σ^2 , is related to the *s* value by this equation

$$\sigma^2 = \frac{\pi^2}{3} s^2 \qquad \text{(Noise component of activation, 5.4)}$$

Typically, modelers are only concerned with the instantaneous noise and leave the permanent noise parameter turned off. The parameter ANS sets the instantaneous noise value that is added to the activation equation.

In the thesis optimization two of the parameters (ANS, BLC) are part of the declarative module and components in the activation equation explained above. ANS, is *s* in the noise component of activation (5.4), and BLC is the constant value, β, in the base-level activation (5.2) of a chunk. The other parameter investigated in the thesis, SYL, is in the vocal module. The vocal module gives ACT-R a rudimentary ability to speak, such as speaking words or short phases for simulating verbal responses in experiments and for subvocalizing text internally. SYL is the seconds-per-syllable parameter, or syllable rate. This parameter controls the time it takes the model to articulate a text string measured in seconds. Its default value is 0.15 s/syllable with three characters defining a syllable by default. For example, the time it takes to articulate a string of text is calculated as

$$\text{Articulation time} = SYL * L/C \qquad (5.5)$$

where SYL is the seconds-per-syllable parameter, L is the length of the text string in characters, and C is the characters-per-syllable parameter with the default of three characters. If the string of text is "seven" and the default values are used for both parameters (SYL and C) then the articulation time is

$$\text{Articulation time} = 0.15 * 5/3 = 0.25 \text{ seconds.}$$

The strongest pattern observed amongst the three parameters making up the solution sets found by the PGA optimization involved SYL. The serial subtraction model simulates verbalizing the subtraction answers as the subjects do. Nearly all subjects spoke the subtraction answers out in full and likewise so does the model. For example, the answer 8185 was spoken as "eight thousand one hundred and eighty five" (about 8 to 9 syllables), instead of "eight one eight five" (4 syllables).

The optimization results showed that as performance decreased on the serial subtraction task the value of SYL increased (poor performers take longer to speak a syllable). This correlates with working memory and stress theories in two ways. Stress, anxiety, and worry create mental distractions that compete for the limited working memory capacity required in online mental arithmetic. The competition results in reduced performance speed and accuracy on the task. In addition to affecting the central executive component of working memory, worry may also impact phonological aspects of working memory where speech and sound-related information are stored and rehearsed. Worry has been viewed to be principally a verbal behavior (Borkovec & Inz, 1990; Rapee, 1993). From the separate systems of working memory perspective (Baddeley, 1986), each system has its own store of resources. When demands exceed a system's resources, overflow into other systems' resources may occur resulting in reduced performance and increased errors on the task in progress.

Another important pattern was the proportionally large number of high ANS values in the solution sets. High ANS values increase the noise component in the activation equation adding more variability in the retrieval of a chunk. This variability in both the probability of retrieval and the retrieval time can simulate distraction and interference caused by anxiety and worrisome thoughts on the attentional resources of working memory (Eysenck & Calvo, 1992). Loss of attentional focus can result in missed, erroneous, or delayed retrieval of arithmetic facts. There was also a small subset of low ANS values accompanied by low BLC values. This relationship takes the form of simultaneously increasing/decrease values of ANS and BLC parameter pairs. Unfortunately, this pattern is not as interpretable as the first two patterns in reference to theories of stress, anxiety, and worry.

The BLC value is a constant value summed into the base-level activation equation (5.2) resulting in an increase to the chunk's base-level activation. Base-level activation represents the recency and frequency of the use of the chunk so any BLC value over 0 acts as constant increase of activation value. In the retrieval process, the chunk with the highest activation that matches a request and is greater than the retrieval threshold is selected and placed into the retrieval buffer, thereby becoming available to productions. This would be equivalent to a human wanting to access specific information from memory, and being successful in accessing that information without delay.

A higher activation in general means greater probability of retrieval in a shorter amount of time. What would be expected is more of an inverse relation between ANS and BLC. For example, when predicting low performers that are anxious about the task or worried about not doing well, one would expect performance characterized by fewer number of attempted subtraction problems and increased errors. This type of performance might be simulated with high ANS values reflecting high noise and more variability accompanied by low BLC values suggesting little additional activation beyond what is accounted for by recency and frequency of

chunk use. Instead, the prediction solutions for low performers show more of a concurrent relationship between ANS and BLC, high ANS values associated with high BLC values, and likewise, the opposite for high performers, low ANS associated with low BLC values.

The plots in Figure 5.1 assist in visualizing the relationship between ANS and BLC, in addition to SYL. The three parameters are positioned on the x-axis in all three plots. The solutions (i.e., the parameters values producing best fits with human data) for the three worst performing subjects are shown in the top plot; solutions for the three median subjects in the middle plot; and the solutions for the three best performing subjects in the bottom plot. Each blue line represents an ACT-R parameter set solution with a fit of less than 1.0. There is more than one solution for each subject with a total of 10 best-fitting parameter sets across the three low performers, and 14 solutions for the median and high performers. The bold red line in each plot is the average (i.e., mean solution) over the parameter set values shown in blue.

In comparing the average parameter values across the groups of three subjects (red line forming an inverted 'V'), the low performers' inverted V is shifted highest on the y-axis, higher in ANS than the other two groups, only slightly higher in BLC than the middle group, and substantially higher in SYL than the other two groups. The high performers' inverted V is shifted lowest on the y-axis, but only slightly so for ANS when compared to the middle group, and fairly significantly lower for BLC and SYL when compared to the other groups. The average parameter values for the middle group fall in between the other two groups.

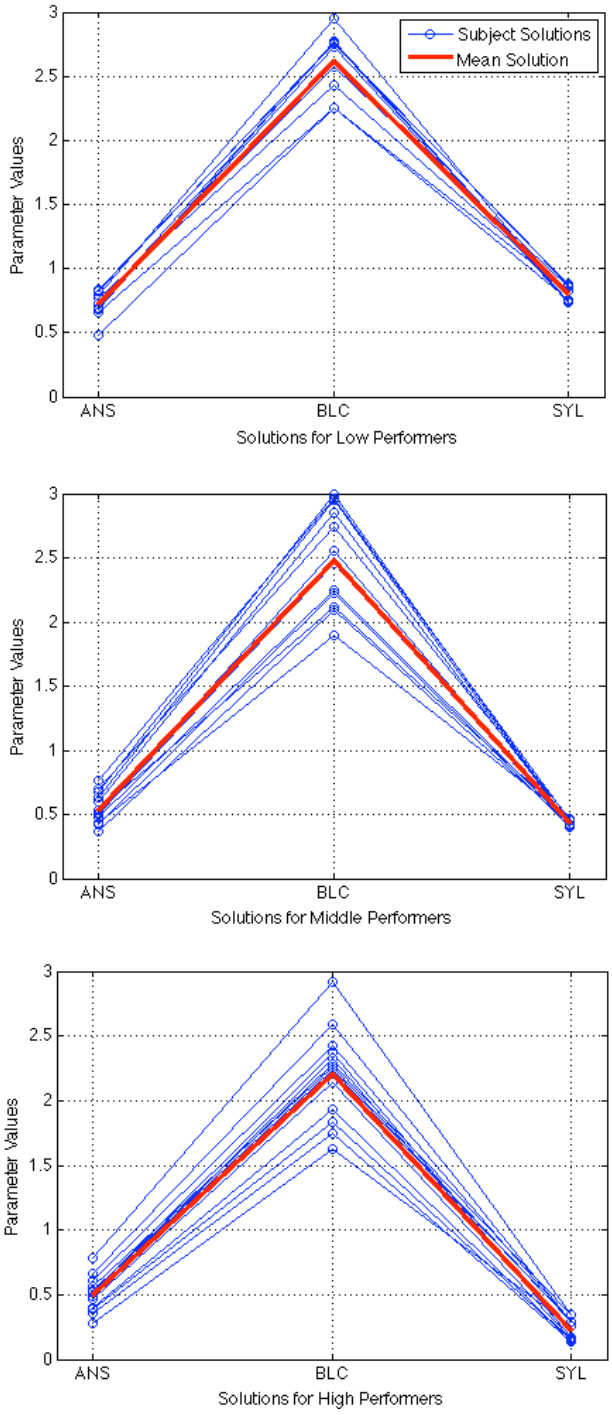

Figure 5.1: Comparison of parameter values (ANS, BLC, SYL) from optimization solutions for the three worst performing subjects (top), three average performing subjects (middle), and three best performing subjects (bottom). Each blue line is a solution from a subject optimization; the red line is the average across the solutions.

Taking into consideration that the low group represents four less solutions than the other two groups, the value ranges for ANS and BLC appear more compact. The middle group has the smallest range of SYL values. Value ranges for all three parameters is greatest for the high performers, and this is especially true for the value of BLC. The solution represented by the top line in the plot could be considered an outlier.

Using the same format, Figure 5.2 compares prediction solutions for only the best performer (subject 26) and the worst performer (subject 1). This plot clearly shows the differences in parameter values in reference to performance. Solutions for subject 1 have higher ANS, BLC and SYL values than solutions for subject 26 without overlap. Only three solutions were found during subject 1's optimization, whereas seven were found during the optimization of subject 26, inferring that subject 26's data was easier for the PGA to optimize.

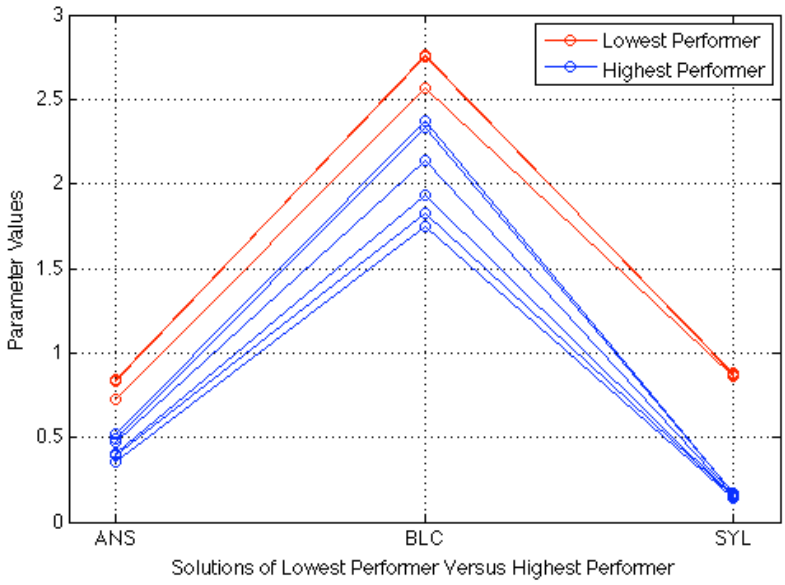

Figure 5.2: Comparison of parameter values (ANS, BLC, SYL) for optimization solutions of the worst performing subject (subject 1, in red), and the best performing subject (subject 26, in blue).

Two of the three patterns observed in the solution sets found by the PGA optimizations appear supported by theories of stress and anxiety and their influence on aspects of working

memory capacity. The third pattern, simultaneously increasing/decreasing values of ANS and BLC parameter pairs, needs to be examined further before an interpretation can be proposed. When considering the parameter sets overall (Figures 5.1 and 5.2), an inversion relationship of high parameter values associated with low performance, and low parameter values associated with high performance is visible. The plot (Figure 5.2) comparing parameter sets from solutions of the worst performer versus and the best performer illustrates this point. The fairly clear distinction of performance and solution set values between subject 1 and subject 26 shown in Figure 5.2 prompted construction of prediction landscape visualizations.

Parameter Space Visualizations

The PGA optimizations to performance data from subject 26 (best performer) and subject 1 (worst performer) produced several fits less than 1.0: three solutions for subject 1, and seven solutions for subject 26. The solutions for these two subjects when plotted by parameter value (Figure 5.2) clearly clustered inversely by level of performance. High parameter values characterize the worst performer and low parameter values characterized the best performer. A three-dimension visualization technique, called slice planes, was used to sample specific areas in the parameter spaces of these two subjects.

Two preliminary data volumes were constructed from coarse-grained grids of 4913 data points per volume. A data point represents a parameter set and associated average fitness value calculated across 20 model runs. To create one data volume, 98260 model runs were required that consuming approximately 30 CPU hours on 20 processors. The data for the volumes is scalar with *XYZ* representing ANS, BLC, and SYL, respectively. Fitness values (sum of squares error between model predictions and human data on number of attempts and percentage correct) calculated from the model predictions are the 4^{th} dimension, or *V*, represented by a color bar scale

from black (fitness = 0, perfect fit) to white (fitness > 200). In the visualizations the landscape boundaries were set approximately the same as the parameter space searched by the PGAs: for ANS from 0.1 to 0.9, with an increment of 0.05; for BLC from 0.1 to 3.3, with an increment of 0.1; and for SYL from 0.1 to 0.9, with an increment of 0.05.

Figure 5.3 shows two views of the parametric landscape for each of the two subjects. The top two plots visualize subject 1, the worst performer. The bottom two plots visualize subject 26, the best performer. The same color bar scale is used for all the plots; it appears in the top left-hand plot. The slice planes are positioned and perspectives rotated to illuminate locations of good fits within each parameter space.

In subject 1's plots, the ANS and BLC slice planes remain at a constant position while the slice plane for SYL is initially positioned at 0.7 in the top left-hand plot and then increased by 0.1 in the top right-hand plot. For subject 26's plots, the slice planes remain in a constant position for both the bottom left- and right-hand plots with the ANS plane set to 0.35, BLC to 2.9, and SYL to 0.2. The view is rotated clockwise between the left-hand and right-hand plots.

The color scale represents best fitting predictions as black with the fit becoming less accurate as the color moves towards white. Best solutions for subject 1 appear more concentrated in a small area and then extend into a band-like pattern perforated by some black squares of perfect fitness. In contrast, the best solutions for subject 26 are more massed in an area near the upper constraint of BLC and lower constraint of SYL.

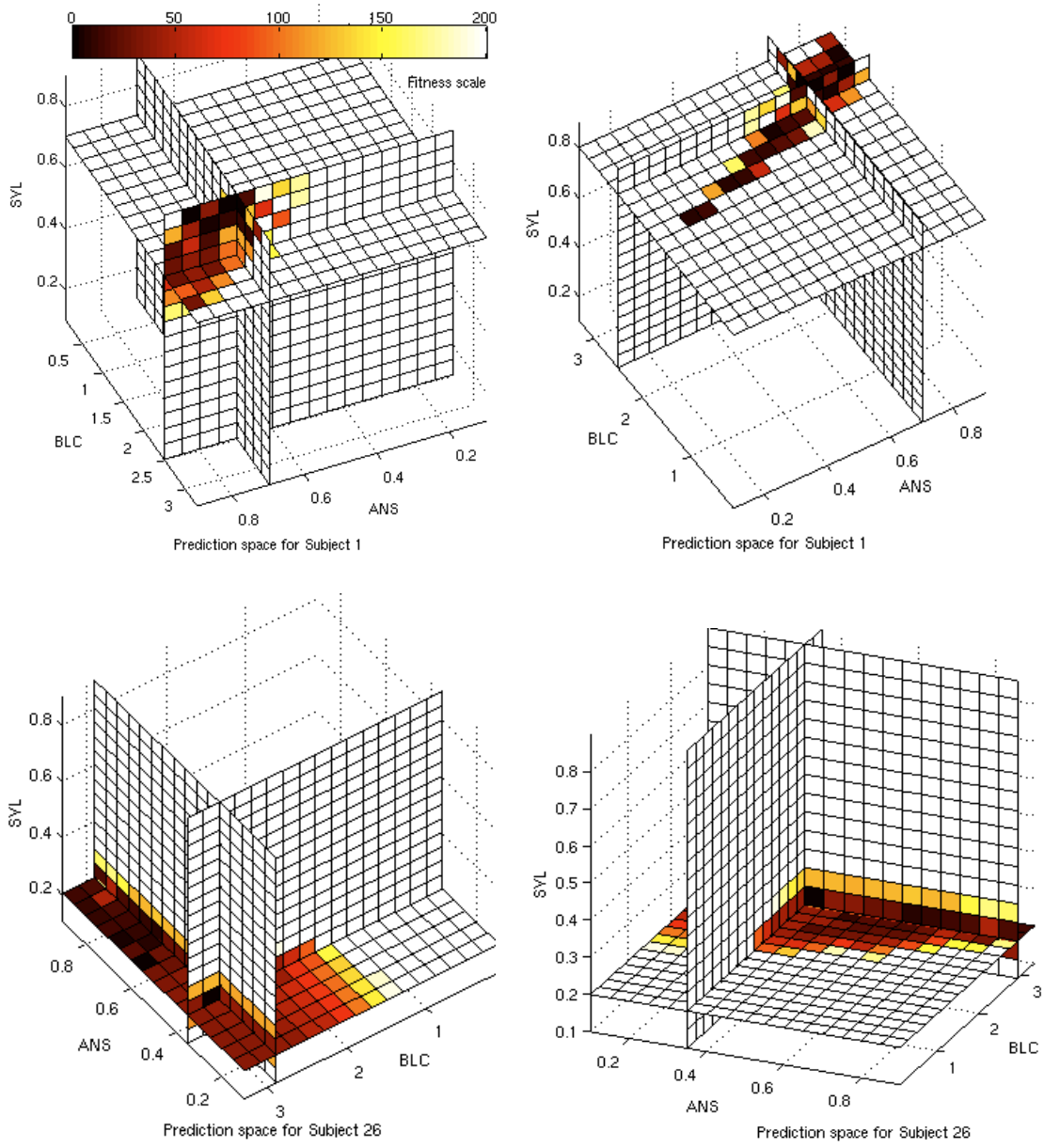

Figure 5.3: Parameter landscapes for subject 1 (top two plots) and subject 26 (bottom two plots). Color bar scale (top left-hand plot) 0 to 200 represents fitness values for all plots.

As noted above, the PGA found more than twice as many nearly perfect fits for subject 26 then for subject 1. Subject 1 is in the threat appraisal group, and performed the worst overall. Subject 26 is in the challenge appraisal group, and performed the best overall. Interestingly, subject 26 reported using a different strategy than the model for performing the serial subtraction

task. Limitations of the visualizations in Figure 5.3 (i.e., one data set, one model, only two subjects, course-gained grid) restrict interpretative insights as to the differently-shaped patterns of best solutions. It could be the case that the different patterns indicate levels of stress, strategy type, performance, or any such combination. More finer-grained data volumes are needed of each individual subject's optimization solutions before proposing interpretations. If plots such as these were refined and incorporated in the model fitting process, modelers might be able to choose the best set of parameters with confidence.

In this section it was noted that the PGA found three solutions for subject 1's optimization data, and seven solutions for subject 26's optimization. In fact, in nearly all cases, the individual subject optimizations produced several solutions with a fitness of less than 1.0 per subject. These sets of solutions could be considered as distributions of ACT-R parameter values producing good fits. Each one of these solutions was tested over 200 model runs on 200 processors in the post-PGA testing portion of the optimization. The model predictions resulting from these model runs are prediction distributions much like subject performance distributions. The next section briefly discusses several types of distributions produced as byproducts in the PGA optimizations of individual subjects.

Prediction Distributions

A summary of the PGA optimization results for each of the 15 subjects is shown in Table 4.8. Only one solution for each subject is listed in Table 4.8, i.e., the one with the lowest fitness that was found by the optimization. With the exception of subject 14, the individual subject optimizations returned multiple solutions with fitness less than 1.0. Table 5.1, similar in format to Table 4.8, includes a count of how many total solutions were found with a fitness less than 1.0, and the parameter ranges across those solutions for each individual subject optimization. Again,

the subjects are ordered by performance in number of attempts from low to high. Column 3 contains the solution count, and columns 4, 5, and 6 are parameter ranges for ANS, BLC and SYL by individual subject optimization.

When the model was optimized to subject 43's data the PGA found six solutions with fitness less than 1.0. These solutions showed ANS values ranging from 0.32 to 0.72 (a difference 0.4), BLC values ranging from 1.74 to 2.88 (a difference 1.14), and SYL varying by only 0.01. This subject's solutions showed the greatest variability in ANS and BLC values overall. It is interesting that this subject is positioned in the top half of performance (bottom half of table) but belongs to the threat appraisal group. In the BLC range column there are 5 subject optimizations with a minimum value less than 2.00 (subjects 46, 43, 41, 21, and 26). These subjects are all positioned in the average to above average performance portion of the table. Note also, that the corresponding ANS range minimum value is below 0.5 for these five optimizations. This is another instance of the pattern: low ANS and BLC value pairs and high ANS and BLC pairs.

Table 5.1: Number of solutions found (column 3) and ranges of parameter values (columns 4, 5, 6) resulting from individual subject optimizations using 200 genotypes over 100 generations on 200 processors. Subjects ordered from low to high performance by number of attempts.

Subject	Human Data	Fittest	ANS Range	BLC Range	SYL Range
1	28, 67.9	3	0.73–0.84	2.57–2.76	0.86–0.88
47	29, 62.1	2	0.66–0.77	2.25–2.43	0.82–0.83
25	31, 80.7	5	0.48–0.79	2.25–2.95	0.74–0.76
11	35, 65.7	2	0.80–0.82	2.35–2.49	0.68–0.69
14	37, 75.7	1	0.83	2.75	0.62
2	37, 78.4	3	0.73–0.81	2.59–2.80	0.60–0.63
46	45, 80.0	2	0.43–0.53	1.90–2.12	0.47
27	46, 87.0	2	0.70–0.76	2.74–2.96	0.46–0.47
16	50, 92.0	9	0.37–0.68	2.09–2.99	0.40–0.42
43	54, 89.0	6	0.32–0.72	1.74–2.88	0.37–0.38
41	55, 87.3	3	0.36–0.60	1.86–2.44	0.36–0.37
23	57, 84.2	6	0.52–0.79	2.15–2.71	0.34–0.35
9	57, 87.7	2	0.53–0.78	2.28–2.92	0.34–0.35
21	65, 90.8	5	0.28–0.66	1.63–2.59	0.26–0.29
26	83, 94.0	7	0.36–0.52	1.75–2.37	0.14–0.17

Figure 5.4 plots the number of solutions from column 3 in Table 5.1 in a histogram format with subject numbers on the x-axis ordered from low to high performance and colored according to task appraisal. For example, the greatest number of solutions, nine, was found during subject 16's optimization. Subject 16 was one of the three subjects characterized as a median performer (subjects 46, 27 and 16) plotted earlier in Figure 5.1. In Figure 5.4, subject 16 is denoted as S16 located near the middle of the x-axis with nine solutions indicated on the y-axis.

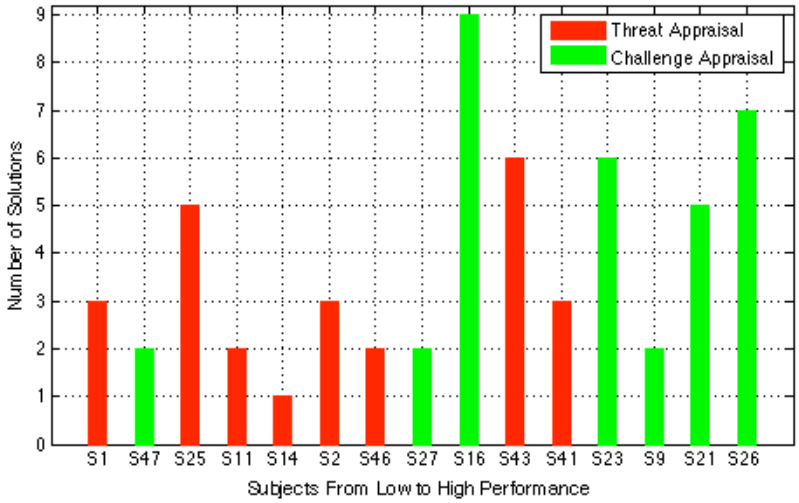

Figure 5.4: Number of solutions found during individual subject optimizations. Subjects ordered from low to high performance (x-axis) with line color by appraisal group.

A line color of green indicates that the subject was in the challenge task appraisal group; a red line indicates the threat appraisal group. The two optimizations that found the most solutions correspond to challenge subjects (subjects 16 and 26). The third and fourth most solutions were tied between a challenge subject's optimization and a threat subject's optimization. The third highest number of solutions is six, observed for subject 43 (threat appraisal) and subject 23 (challenge appraisal), both are positioned in the top half of performance. The fourth highest number of solutions is five for subject 25 (threat appraisal, in low half of

performance) and subject 21 (challenge appraisal, in top half of performance). Overall the four optimizations returning the greatest number of solutions are in the top half of performers.

Returning back to Table 4.8 that listed the solutions with the lowest fitness value produced by each individual subject optimization. Each of these solutions was tested with 200 model runs in the post-PGA testing portion of the optimization. Because of human performance variance in performing the serial subtraction task and stochastic components in the architecture and model, the 200 model runs using one static set of ACT-R parameter values as input produced a distribution of performance predictions as output. For each of these best solutions (one per subject optimization) an absolute frequency histogram of the model's predictions was created. Therefore, in total, there were 15 different prediction distributions, one for each subject's optimization using the best solution of ACT-R parameter values as input. These 15 prediction distributions were overlaid in two plots, one plot for number of attempts predicted by the model, and a second plot for the percentage correct predicted by the model. Both of these plots are shown in Figure 5.5.

The distributions of predicted number of attempts in the upper plot are colored green, and distributions of predicted percentage correct in the lower plot are colored red. The black distribution on each plot represents the model prediction distribution from optimizing to an average across all subjects. The distributions for predicted number of attempts were distinct enough to allow labeling of the subject number associated with each optimization solution that created the distribution. The predicted number of attempts distributions tended to more compact and unimodal. The 15 distributions of percentage correct overlap too much to allow labeling of individual subject numbers. These distributions tended to be bimodal with a detached smaller, lower mode preceding a larger, higher primary mode.

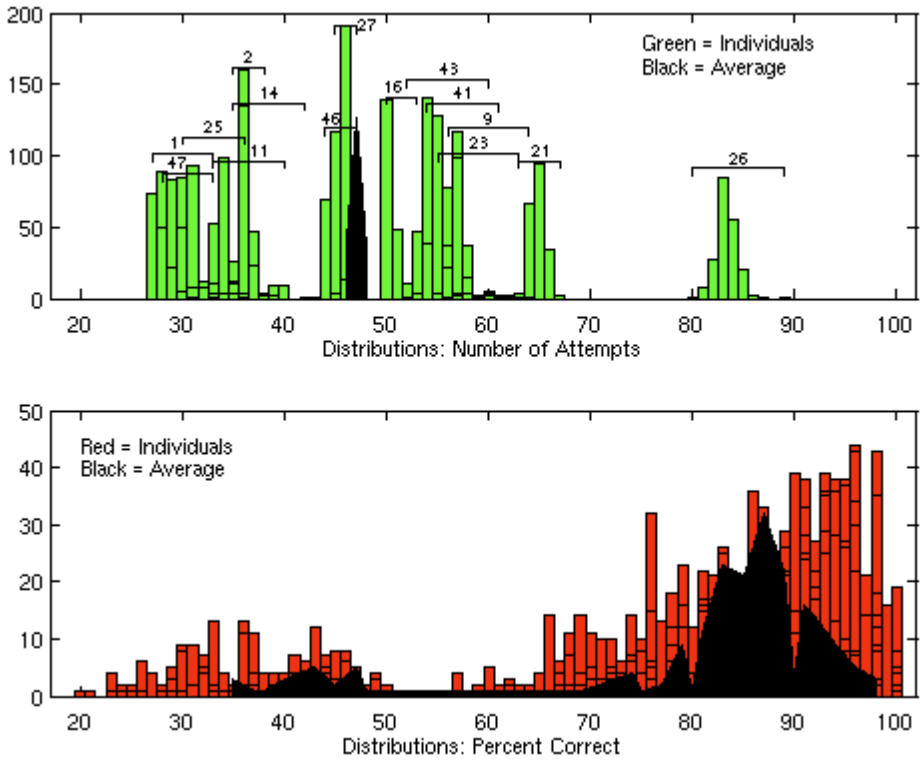

Figure 5.5: Model prediction distributions (each 200 runs of the individual subject optimizations in Table 4.8). The 15 prediction distributions are overlaid and grouped by number of attempts (top plot, green) and percentage correct (bottom plot, red). The black distributions on plots represent the model prediction distributions from optimizing to an average across all subjects.

In Chapter 4, Table 4.2 showed the average number of attempts and percentage correct across all 15 subjects. The large standard deviations listed in Table 4.2 indicated that there was substantial variance in human performance on this task. The black distributions were added to the plots in Figure 5.5 to compare optimizing to averaged data and optimizing to individual data. The black distribution in the top plot produced by 200 model runs using the best solution from optimizing to an average performance for number of attempts covers only one-fourteenth of the combined distributions from optimizing to individual subject data (green distributions). In addition, the shape and position of the black distribution in the upper plot does not appear to resemble any individual subject distribution for number of attempts.

In contrast, the black distribution in the bottom plot produced by using the best solution from optimizing to the average percentage correct resembles the shape (bimodal, small mode preceding larger mode) of the combined distributions for individual subject optimizations. Although the black distribution of average percentage correct is similar in shape, it is shorter in range in both dimensions. For example, the combined red distributions have the range approximately 20 to 100, while the black distribution ranges from 35 to about 97, with a difference as much as 18% correct.

The green distributions in the top plot of Figure 5.5 appear to be clustered into a poor performance group with the range of 25 to about 40 number of attempts. The individual distributions for subjects 1 and 47 overlap at the low end of this range. There is a second cluster of distributions in the middle of the plot in the range 45 to around 67 number of attempts. From the subject numbers, and the contents of previous tables, these are recognized as average to above average performers. The distribution from subject 21's solution stands apart from the other distributions, as subject 21 is the second best performing subject. Likewise, the distribution from subject 26's solution stands well apart from all the other distributions in the range 80 to nearly 90 number of attempts. Also, subject 26's distribution appears wider than most of the other individual distributions.

The red distributions in the bottom plot from individual optimization solutions overlap and intertwine in reflection of the shape of each individual distribution (bimodal with smaller mode preceding larger mode). And the black distribution of average percentage correct echoes this same shape. This black average percentage correct distribution is a better approximation of the range of performance produced by all the individual subjects than the black average number of attempts distribution in the top plot.

During the experiment subjects performed two blocks of subtracting by 7s and two blocks of subtracting by 13s. If subjects had performed more trials of the serial subtraction task,

the distributions of performance statistics produced by the model for the individual subject optimizations as shown in Figure 5.5, could be compared to individual-level human performance distributions. Modeling the variance of the human performance for both number of attempts and percentage correct would make a more powerful fitting criteria which in the end might improve the results when using a parallel processing global search optimization approach.

Summary

Two of the three patterns observed in the PGA's solution results appeared to have support from theories of stress, anxiety, and worry. The third pattern of simultaneously increasing/decreasing values of ANS and BLC cannot be currently explained. It might be the case that the subsymbolic processes utilizing these two parameters is not fully understood in reference to the architecture and the data being fit. The thesis project investigated only three ACT-R parameters of many involved in declarative memory processes. Possibly, BLC was not a good choice and other parameters or combinations of parameters need to be investigated. In a manual optimization process selecting the 'wrong' parameters to manipulate would most likely not result in being able to fit the model to the human data with accuracy. This was not the case with the thesis optimization approach where every model to human data fitting problem was solved.

Table 5.1 showed that several solutions were found for each optimization problem. One advantage of a GA that sets it apart from other search algorithms is its ability to maintain a set of candidate solutions through the optimization process. The problem here is in the fact that the solutions lay in different areas of the parameter space. For example, the solutions when optimizing to subject 21 go the extremes of ANS 0.28, BLC 1.63, SYL 0.26 to ANS 0.61, BLC 2.43, and SYL 0.29. One solution has quite a low ANS and BLC while the other has a high ANS and BLC. An ANS value over 0.5 would not normally be assigned in manual optimization. This

type of solution situation occurred in several individual subject optimizations, as well as the optimization to averaged data across subjects and the optimization to the challenge group average. In manual optimization this might characterize the initial steps in the optimization process. For example, the modeler tries a set of low values, then a set of high values, in attempting to zero in on the best solution. The thesis optimization in its extensive search of the parameter space is able to find both a low set and high set of values that fit the human data nearly perfectly.

The contents of this chapter raises a concern that thesis optimization approach may be able to fit a model that is incorrect, i.e., does not represent how the human subjects are performing the task. For example, subject 26, when interviewed about the strategy he used to perform the serial subtraction task, reported that he was in fact not subtracting, and yet, the optimization approach found a fit. The tradition in cognitive modeling is model validation in the performance of human subjects. The thesis project proposes a superior optimization approach in replacement of the effortful trial-and-error manual optimization approach that has been utilized in the cognitive modeling field since its inception. There is no doubt that the thesis optimization approach can replace manual optimization in finding accurate model-to-human data fits. But, whether this approach is actually validating the model is in question. It will take more models and many more optimized parameters sets for an answer.

Chapter 6

DISCUSSION AND CONCLUSIONS

Discussion

This section first provides a discussion of the main contributions of the thesis project: the development of a software system for individual differences modeling; an increased understanding of the effects of anxiety on cognitive performance, and ACT-R modeling in general; a new application of PGAs; and a new paradigm for model creation and validation. The second and third subsections discuss several future directions for the thesis project, as well as limitations of the research.

Contributions

Contribution 1: Software system for individual differences modeling. The primary purpose of the thesis project was to better enable individual subject investigations by proposing an automated optimization approach or 'system' for fitting a cognitive model to single-subject performance data. The optimization system runs on a high performance cluster platform and is composed of a PGA, a computational cognitive architecture, and a target cognitive model of a serial subtraction task producing predictions of individual performance. The thesis optimization system successfully fit the serial subtraction model to two groups of individual subjects and 15 individual subjects.

Cognitive modeling researchers rarely attempt to capture particular individual subjects' behavior. Most often human subjects are modeled as invariants, not as individuals, with data that

have been averaged or aggregated across all subjects. This type of modeling assumes that there are no individual differences between subjects. However, when the performance of subjects has genuine differences it is well known that averaging produces data that do not accurately represent the behavior of individuals and provides a misleading basis for modeling, theory, and data summarization.

Investigations into individual differences in cognition is inhibited by the field's widespread practice of manual optimization—a time consuming process of repeatedly tweaking the model in terms of real-valued parameters in an attempt to improve the model's fit to the data. Fitting a model to individual data using manual optimization requires substantial amounts of modeler's time and computational resources depending on the number of subjects and type of data. The thesis optimization system offers an automated and more accurate model-fitting alternative to manual optimization.

When fitting to individual subjects, the thesis optimization system found good fits for all 15 subjects defined by fitness values calculated from the objective function ranging from 0.0006 to 0.77 (0 is equivalent to model predictions exactly matching human performance data).

There are two disadvantages associated with modeling individual differences: models are fit to all of the noise in the data; and from a theoretical perspective, fitting each subject requires extra free parameters that lead to progressively more complicated accounts of the data as a whole. As has been pointed out in the psychology literature, it is important to maximize goodness of fit and to minimize model complexity to achieve the basic goals of modeling.

An alternative to single-subject level analysis is to partition subjects according to their individual differences and model the aggregated data from each group. Subjects can be grouped in any number of ways from performance similarities, to psychosocial characteristics, to predispositions or attitudes about the experimental task (e.g., task appraisal). Under this approach,

a set of parameterizations can be applied to each group of subjects to model differences in cognitive processes of each group.

When fitting to subjects divided into two different task appraisal groups (challenge and threat), the thesis optimization system found 10 different fits for the challenge group ranging from 0.0073 to 0.9765, and four fits for the threat group average ranging from 0.1948 to 0.7462.

The thesis optimization has proved its usefulness in both fitting to individual subject data as well as individual grouped data. Based on theoretical hypotheses, the task under investigation, and the nature of the data itself, the modeler must decide if fitting to individual data or group data is preferred, and which provides a better account of the data. The automated aspect of the thesis optimization allowed for efficient fitting of the model practically devoid of modeler intervention or timely effort to both levels of performance data. If the modeler is unsure which level of analysis will produce the best account of the data and support of theory, both levels of data fitting could be executed and then the results compared by level of analysis in determining the best model fits and corresponding account of the data.

Contribution 2: Increased understanding of anxiety and cognitive performance. The thesis optimization solutions that adjusted the serial subtraction model's predictions to fit the individual subjects' performance data are in the form of ACT-R parameter sets. When analyzed, these solution sets exhibited several patterns within the parameter values with one of these patterns supported by several theories of stress and cognitive performance.

The SYL pattern found in the optimization solutions suggests interaction or competition for working memory resources during calculation of the subtraction problems and verbalization of the problem answers. SYL, the seconds per syllable parameter, controls the rate of speech in ACT-R's vocal module. The human subjects speak the answer to each subtraction problem. Likewise, the model simulates verbalizing the subtraction answers through ACT-R's vocal module.

As discussed in Chapter 2, Eysenck and Calvo's (1992) processing efficiency theory claims that the main effects of worry (and, more generally anxiety) affect the central executive system of working memory. According to Baddely (1986), the working memory system has limited capacity and is composed of three distinct systems, one of which is the phonological loop used for rehearsal and transient storage of verbal information. The phonological loop and the visuo-spatial sketchpad are slave systems to the central executive. Each slave system relies on its own pool of attentional resources. If tasks that one of these slave systems is performing are simple and do not require much in the way of attentional resources, then working memory performance is unchanged. However, if one of the slave systems is given a demanding task, it either fails to complete the task or draws on the attentional resources of the central executive, which results in impaired working memory performance.

Ashcraft and Kirk (2001) found that high math-anxiety subjects showed slower, more effortful processing on procedural aspects of performance, such as performing carry or borrow operations. Both Ashcraft et. al. (2001) and Eysenck et. al. (1992) believe that worrisome thoughts of failure can consume the limited attentional resources of working memory therefore leaving less available for concurrent task processing. Concurrent task processing in serial subtraction involves the mental arithmetic processes and the verbalization of the subtraction answer both underway at the same time. Borkovec and Inz (1990) viewed worry as principally a verbal behavior. This is in agreement with Rapee (1993) who suggested that worry involves inner verbal activity.

These intertwining theories of worry and anxiety suggest that the central executive component and the verbal component of working memory are competing for working memory resources and possibility drawing on one another's resources during high demand situations resulting in failures (i.e., subtraction errors), or impaired working memory performance, or both.

The optimization results showed a slower rate of verbalization of the subtraction answers associated with longer response times and more errors. This behavior was most evident for subjects with a self-reported threatening task appraisal and/or poorer performance in number of attempts and percentage correct. The SYL pattern observed in the optimization results showed that as performance decreased on serial subtraction the value of SYL increased. This means that poor performing subjects speak more slowly than their high performing counterparts.

The detailed human performance data collected from subjects performing the serial subtraction task combined with the solutions produced by the thesis optimization in fitting the model to the individual subjects assisted in understanding how these two components of working memory are competing for limited resources with the result of impaired cognitive performance on the task.

Contribution 3: Increased understanding of ACT-R modeling. Another pattern uncovered while analyzing the solution sets produced by the thesis optimization involved two parameters, ANS and BLC. This pattern of simultaneously increasing/decreasing values of ANS and BLC parameter pairs was not really supported by theories of stress or anxiety. It was thought that the parameter BLC, in combination with ANS, might not be the best choice in the simulation of working memory capacity for the serial subtraction task. Which architectural parameter or combination of parameters to subsymbolically manipulate in simulating working memory individual differences can be a complicated decision involving many variables.

This is an example of how combinations of parameters, even just two different parameters, are difficult to interpret in reference to the subsymbolic processes in the ACT-R architecture and consequently the model's behavior. Fortunately, the thesis optimization technique is efficient enough to allow investigations of many possible architectural parameter combinations and values, thereby, contributing to a general understanding of the theory underlying the ACT-R cognitive architectural and its integration of separate modules.

By fitting the model to groups of individuals by task appraisal and individual subjects, the solution sets of ACT-R parameters showed a relationship between the pair of parameters (ANS and BLC) and the quality of performance on the task. Even though this relationship was unexpected in respect to theories of stress, the model-to-data fits showed a concurrent relationship between ANS and BLC values and an inverse relationship with performance. High ANS and BLC value pairs were associated with poorer performance, and low ANS and BLC value pairs with higher performance. At this point theories of stress and anxiety will need to be reevaluated in reference to ACT-R's activation equation and base-level activation of declarative memory chunks in order to interpret the meaning of this parametric pattern.

As mentioned in the thesis, the ACT-R architecture has approximately 80 different parameters available for manipulating a model's behavior while running in the architecture. Often only a very small number of parameters are investigated in reference to a particular model leaving much of the architectural parameter space unexplored. The thesis optimization and its utilization of the PGA are ideally suited for exploring the architectural parameter space. Up until now a search method has not been available to enable an exhaustive investigation of the architectural parameter space including confirmation of the architecture's default parametric values and a nonbiased comparison with cognitive psychology theories.

In reference to ACT-R's default parameter values, the optimization results of the serial subtraction model to individual subjects strongly suggests that the default parameter value for SYL, the seconds-per-syllable parameter, requires a substantial increase in value. The best performing subject was the only subject speaking a syllable close to ACT-R's default rate of speech. It is comprehensive investigations of model-to-data fits such as this thesis that need to be performed to validate architectural defaults for parameters. Many ACT-R modelers claim a parameter-free fit of a model to human data as noteworthy, nearly the ultimate in model

validation. The presence of incorrect default parameter values that supposedly represent typical or average human behavior negate parameter-free claims of modeling success.

Does ACT-R accurately codify accumulated cognitive psychology theory as to what is known to be true so far? This is a long-standing and probably the most important question posed of any cognitive architecture. The process of manual optimization where a modeler has formulated a hypothesis and then conducts a hit or miss search for a set of parameter values that validates his claim encourages biasness in the scientific process. For example, the modeler may find a solution that is 'close enough' for validating his hypothesis concluding his search prematurely when better solutions remain undetected. The modeler may find several solutions proceeding to select the one most supportive of his hypothesis while ignoring the others. Or, the modeler may labor under the inefficiencies of manual optimization not finding any solutions and consequently terminating his research before the entirety of the parameter space has been explored.

Biasness in problem-solving activities such as model fitting via manual optimization can result in missed opportunities for learning and acquiring new knowledge. The automatedness of the thesis optimization approach reduces this biasness, promotes knowledge exploration and theory development, and progresses cognitive architecture research in general.

Contribution 4: New application of PGA. The thesis project utilized a master-slave PGA to search the ACT-R parameter space for model-to-data fits. With the exception of a preliminary study using a serial GA (Tor & Ritter, 2004) and the building of a software infrastructure consisting of military, volunteer, and high-performance resources for cognitive modeling research by the Air Force Research Laboratory (Gluck & Harris, 2008a, 2008b; Gunzelmann, 2008), there have been no other attempts employing a global search technique for the purpose of model fitting within the ACT-R cognitive modeling community, and possibly any of the other symbolic representation-based cognitive architectures.

In the thesis PGA optimization approach, the master node executes the GA operations (selection, crossover, mutation), and the evaluation of fitness is distributed among slave processors. The slave processors evaluate the fitness of the individual genotypes (encoded ACT-R parameter sets) that they receive from the master and return the results. Evaluating the fitness of a genotype involves running the serial subtraction model with the ACT-R parameters (encoded genotypes) as input to the ACT-R architecture and then comparing the model's predictions to the human data.

Randomness is incorporated in every component of ACT-R's subsymbolic level of processing. The stochastic components in the architecture, and therefore the model, result in a distribution of performance instead of a single performance prediction. The PGA's search of the parameter space was complicated by this stochasticity that disrupted convergence on a solution set. The PGA was modified with the addition of collection and testing functions to compensate for the lack of convergence caused by stochastic effects in the architecture and model. Through the evolved generations, if the PGA finds a 'good enough' solution, as determined by a boundary fitness value, that particular genotype is 'collected' and remembered for a post-PGA testing phase.

The modified version of the PGA worked well in finding fits to both appraisal groups and the 15 individual subjects. However, in this version of the PGA some of the intelligent characteristics of a typical GA have been circumvented such as accumulation of historical information across the generations guiding selection of the next points to search. Additionally, because stochastic effects inhibited convergence, a termination condition based on fitness value could not be used when cognitive models and their predictions are part of the fitness function. In the modified version of the PGA, the termination condition was a specified number of generations. Table 6.1 summarizes the modifications to the PGA and extensions to the model that were required in the thesis optimization approach.

Table 6.1: Summary of PGA modifications and model extensions enabling optimization of ACT-R cognitive models

1. Specialized front-end function written in Lisp to start up the model from a system call within the PGA code of the fitness function.
2. Genotypes encoded as ACT-R parameter values passed as arguments in the front-end function from the PGA through the system call to the model.
3. Performance predictions of the ACT-R model are written out to a file for the PGA to read and utilize in the genotype fitness calculations.
4. Compensate for lack of PGA's convergence caused by architecture and model stochasticity by saving best-fitting genotypes from each generation to data structures within the PGA code.
5. Upon completion of last generation, test each best-fitting genotype saved across the generations by running the model on all allocated processors with resulting performance predictions averaged across model runs.

This redesign of the PGA was necessary for fitting the serial subtraction model and most likely any cognitive model written in ACT-R that uses the architecture's subsymbolic level processing. The successfulness of the PGA modifications confirms that global search algorithms are a viable option for cognitive model fitting, and in general can be used to search the parametric landscapes of cognitive architectures.

Contribution 5: New paradigm for model creation and validation. Research on modeling and simulation of organizational processes focusing on the determinants of organizational performance often refer to the concepts of exploitation and exploration (Axtell, Axelrod, Epstein, & Cohen, 1996; Burton, 2003; Carley, Prietula, & Lin, 1998). Organizational theorists utilize formal models developed using mathematics, simulation, expert systems, and formal logic to describe organizations and predict behavior. The complex adaptive nature of organizations makes formal models a valuable tool for theory development.

The seminal article on exploration and exploitation (March, 1991) is a culmination of three decades of modeling and is widely accepted for its depiction of the adaptive tension between exploration and exploitation. The concepts of exploration and exploitation have been

extended beyond application in organizational science and immigrated into research areas such as artificial systems, biotechnology, product and software development, economic development, and technological innovation (e.g., Bray & Prietula, 2007; Dyba, 2000; Rothaermel & Deeds, 2004; Woolcock, 1998; Yang & Ye, 2002).

For March (1991) a central concern of studies of adaptive processes is the relationship between the exploration of new possibilities and the exploitation of old certainties. March references Holland's (1975) research on the application of adaptive processes present in natural systems to the design of artificial systems. As noted in Chapter 2, Holland is the developer of the genetic algorithm.

Exploration includes things captured by terms such as search, variation, risk taking, experimentation, flexibility, discovery, and innovation. Exploitation includes things such as refinement, choice, production, implementation, and execution (March, 1991). Tension exists between exploration and exploitation where exploitation represents refinement and extension of existing competencies producing returns that are positive, proximate, and predictable; while exploration represents experimentation with new alternatives and diversity producing uncertain, distant, and sometimes negative returns.

March (1991) argued that a balance between exploration and exploitation must be maintained to ensure the survival of the system. Adaptive systems that engage in exploration to the exclusion of exploitation are likely to find that they suffer the costs of experimentation without gaining many of its benefits. They exhibit too many underdeveloped new ideas and too little distinctive competence. Conversely, systems that engage in exploitation to the exclusion of exploration are likely to find themselves trapped in suboptimal stable equilibria (March, 1991).

When the concepts of exploration and exploitation are considered in reference to the PGA-based thesis optimization approach and the established manual optimization process similarities emerge. The PGA optimization is representative of exploration, i.e., biologically

motivated searching in a space for desired solutions. In the context of population genetics, a fitness landscape is a representation of the space of all possible genotypes along with their fitness. Evolution causes the population to move along the landscape in particular ways, with 'adaptation' seen as the movement toward local peaks or optimum. In GAs, the initial population is randomly generated, and the genetic operators are composed of random elements. This embedded randomness sometimes allows suboptimal choices to be made in the hopes that they will lead to better solutions down the road. GAs maintain a set of candidate solutions until termination of the algorithm occurs. Termination can be determined by how long the algorithm has been running or if one of the solutions is close enough to the desired fitness.

In contrast, the manual optimization process most currently used by the cognitive modeling community is representative of exploitation. A specific starting point is selected (i.e., initial parameter values are assigned) based on the modeler's hypothesis and what he expects to conclude. Results from the initial attempt are analyzed, and then a small step is made in the direction of the modeler's goal. This process is repeated until the modeler decides to terminate it. This type of manual optimization resembles a manual hill-climbing effort. The manual optimization of cognitive models is planned, incremental, proximate, iterative, and predictable.

The importance of this comparison lays in the nature of the results produced by these two different types of optimization. In the PGA optimization results discussed in Chapter 5, the resulting solutions sets were many in number and from different regions in the parameter space thus casting uncertainty as to their theoretical interpretation and validity. In the manual optimization process there is only one result with the modeler deciding upon its worth, namely a solution, or a step in the right or wrong direction of a solution.

The thesis optimization approach was developed for the purpose of fitting the serial subtraction model to single-subject level performance data. The PGA optimization approach was successful by finding several model-to-data fits for nearly all individual subject optimizations.

Surprisingly, these fits were found in different regions of the ACT-R parameter space. Because of its exploratory nature, the strength of the PGA optimization approach might lay more in model development instead of limited to the model validation process.

For example, a model development process incorporating the PGA optimization approach could have the following stages. In Stage 1, given that a preliminary model has been built based on human subject observation, identify core architectural parameters for investigation. The parameters selected are hypothesized to be the most relevant in simulating the cognitive behavior observed in the human subjects. A fitness function should be developed based on human performance statistics. The fitness function should not be so complex as to inhibit the PGA's exploration but not so simple as be unsupportive of the hypothesis.

In Stage 2, once the parameters have been identified, define the range of values for each parameter that will be explored. The choice should reflect concerns with these parameters, and expectations as to how different values of the parameters will affect the architecture's subsymbolic processes and the model's behavior. In the beginning, the range of values should be defined rather broadly allowing for exploration of multiple theoretical assumptions. In essence, this defines a search space for the PGA to explore possible solutions.

In Stage 3 run the PGA and analyze the solutions found. If the solutions appear to point in the direction of the hypothesis, then begin to narrow down the search space that the PGA will explore next. This stage might be an iterative process of continuing to constrain the search space as long as the hypothesis appears supported by the results. If at some point the solutions appear unrepresentative of the hypothesis, then returning to Stage 1 is necessary for reevaluation of the parameters selected in support of hypothesis, the hypothesis itself, the model, the fitness function, or any such combination. The 'go back to Stage 1' can be thought of as the 'main loop' in the development process and as a valuable technique for generating hypotheses.

In Stage 4, once a constrained search space yielding solutions in the range of what would be expected is found, the space should be explored multiple times by the PGA varying the number of model runs. If the architecture and model contain stochastic elements it is important to find the number of model runs that give the most stable model predictions. The number of model runs will most likely be much greater than the number of observations collected from the human subject experiment. This should be considered in the next stage if distribution comparisons are planned for use in the validation criteria.

In Stage 5 the PGA optimization serves a validation function. The hypothesis, the computational model of the hypothesis, and the predictions of the model should reflect the cognitive behavior of the human subjects' performance on the task. The fitness function should be strengthened to ensure the best model-to-data fits. For example, additional criteria could be added to create a multiple-objective fitness function, or variance in performance (i.e., performance distributions) could be considered along with the more typical performance statistics.

Ideally, the thesis PGA optimization approach should be tested in the above framework of development stages. This would confirm that the thesis optimization approach could contribute to cognitive modeling research both as a model development approach and as a validation process.

Future Directions

This subsection enumerates several possible future directions for the thesis project. (1) Simulating the effects of anxiety on performance using knowledge representations instead of parameterization. (2) Implementing other more complex types of PGAs utilizing multiple populations and different encoding schemes. (3) Accumulating parametric patterns and their

effects on cognitive processes in pattern libraries. (4) Enabling across-task modeling of individual subjects. (5) Making the optimization approaches available to modelers through open source applications and common academic computational resources

Knowledge representation simulations of anxiety. A study by Miwa and Simon (1993) suggested an alternative to parametric modeling of individual differences. They believe that knowledge representations in production systems (i.e., declarative and procedure knowledge) can accommodate explanations of individual differences. Miwa and Simon (1993) presented a model representation that could make a distinction between common skills and individual differences. They used the example that 70% of the subject's behavior could be explained by the common part of the model, and the remaining 30% by the individual part of the model. Additionally, they defined two levels of individual differences: slight differences that can be represented as small modifications of common rules of the basic model preserving the model's essential properties; and gross differences that are beyond the range of the basic model such as entirely new methods that require new rules.

Similar to Miwa and Simon's model representation, the serial subtraction model could be envisioned as a set of primary production rules that execute the task. Additional production rules could be added to implement a particular theory of stress or anxiety. In the thesis, simulating a stressed subject was implemented with architectural parameters, not by modifying the productions in the model.

For example, the current serial subtraction model does not possess self-evaluative mechanisms. In other words, the model does not know whether it is performing poorly or not. Audio file transcriptions of subjects' performance confirmed that when a subject made an error, especially consecutively errors, the subject would become discouraged with his performance and appear to intentionally slow down his mental step-wise calculations of the subtraction problem in order to get the answer correct. A production or productions representing appraisal feedback

could be added to the serial subtraction model executing when the model detects it has made an error. These extra productions could 'waste time' in simulating the subject slowing down his calculations, or execute internal verbalizations of worry about doing poorly and feeling embarrassed thus interfering with the vocal module's speaking of the subtraction answer. This would be supportive of what was observed in the pattern of SYL values.

Exploring other types of PGAs. Another future direction reconsiders the type of PGA in reference to genotype population and genotype encoding scheme selected in the thesis project. Table 6.2 notes variations in PGA populations and genotypes that could be investigated in future research, and that would affect the efficiency and accuracy of fitting cognitive models to human data in general.

Table 6.2: Possible variations in PGA population type and genotypes affecting cognitive model fitting in future research investigations

1. Method of encoding parameter values as genotypes (binary or real values).
2. Definition of a genotype in number of parameters, length of substrings, and associated accuracy and conversion error of the representation.
3. Size (number of genotypes) and type of genotype population used in a parallel processing environment (global, multiple, or cellular).

The thesis used a binary representation of the variables (i.e., the ACT-R parameters) because that type of representation is more typically associated with traditional GAs (Haupt & Haupt, 2004). In multi-parameter problems, the parameters are each represented as binary substrings of a specific length. The substrings are then concatenated to form an individual genotype in the population. The thesis used three binary substrings each 12-bits in length; however, substrings of equal length are not a requirement. Substrings of variable length would allow varying degrees of accuracy (i.e., longer length increases accuracy) to be assigned to different parameters, this in turn could speed up the search process (Coley, 1999).

Besides increasing the substring length, different numerical notations of genotype representations can also improve accuracy and solution quality depending on the problem (Michalewicz, 1994). If it is important in the model optimization problem that the values of the parameters be continuous and to full machine precision, than it may be more logical to represent the parameters by floating-point numbers. With a floating-point representation considering the number of bits necessary to accurately represent a value is no longer an issue. Instead, the parameters have continuous values that fall between bounds that are appropriate for the problem. GAs that use floating-point number representations for genotypes are called continuous or real-valued GAs (Haupt & Haupt, 2004). Continuous GAs have the advantage of requiring less storage than binary GAs because a single floating-point number represents the variable instead of N_{bits} integers. Continuous GAs are also inherently faster than binary GAs, because the genotypes do not have to be decoded prior to the evaluation of the fitness function.

The thesis implemented a master-slave GA with a single global population. The global population consisted of one large search space defined by the three ACT-R parameters under investigation (ANS, BLC, SYL) with minimum and maximum values of each parameter constraining the space.

The next algorithm having greater complexity is the multiple-population GA, briefly described in Chapter 2. This type of GA consists of several subpopulations that exchange individuals occasionally. The exchange of individuals is called migration and is controlled by several parameters. Each sub-population can find a different locally-optimal solution. The populations send emigrants who have the effect of attracting the other subpopulations to their solutions possibly crossing valleys of low fitness that would have remained unexplored otherwise. This additional exploration may discover even better solutions.

In the thesis PGA results, the patterns of ANS and BLC value pairs were not interpretable in reference to theories of stress and anxiety. Most of the solution sets had an intermingling of

low value ANS and BLC pairs among high value ANS and BLC pairs. Subpopulations of a multiple-population PGA could be set up to divide the ACT-R parameter space for searching. The divisions could be based on different hypotheses about cognitive performance on the task. A processor would be allocated for each genotype as in the global population PGA but the genetic operators would be applied to each subpopulation separately. After the PGA terminates, the solutions from each subpopulation could be compared and evaluated in reference to theory. The search space defined by the best subpopulation could then be used in a global population PGA for a more fine-grained search.

The multiple-population PGA would make for more efficient theory development as one run of the PGA could focus search on several different regions thought to be theoretically feasible. Other alternatives for use of a multiple-population PGA include subpopulations based on the use of different fitness functions, or subpopulations each running a different version of the model. As discussed above, a base model with a primary set of productions, and then several other versions of the base model with different additional productions. The 'family' of models could be optimized in one run of the PGA and then compared.

Parametric pattern libraries. The PGA optimization technique is efficient enough to be employed as an educational tool in learning about a cognitive architecture and how different combinations of parameters influence specific cognitive processes. As mentioned above, the ACT-R architecture has approximately 80 different parameters available for manipulating a model's behavior while running in the architecture. With the PGA optimization, any sized search space could be explored quickly using many different sets of parameter combinations. The results of each search could be overlaid to interpret the effects of each combination of parameters.

With the search space and parameter combinations held constant, different models could be used during each search. If tasks being modeled were broadly classified into categories, libraries or repositories could be set up to collect patterns of parameter combinations with their

resulting effects on specific cognitive processes, possibly ordered by architectural module. Patterns of predictions could be submitted by researchers in the cognitive modeling community to a central library located on the architecture's website forming a collective pattern language of architectural parameter combinations when applied to different classes of tasks.

Fitting individual subjects across tasks. In the thesis project only one model, of the serial subtraction task, was optimized using the PGA. In the case of the human subject experiment explained in Chapter 3, a series of cognitive tasks were performed by each subject (MODS, signal detection, and serial subtraction). The thesis optimization approach could be easily extended to handle the optimization of individual subjects performing a variety of tasks that exercise multiple cognitive abilities (e.g., perception, memory, and problem solving).

With an exception of modeling two different working memory tasks (Lovett, Daily, & Reder, 2000), fitting models of individual subjects across different types of tasks has not been studied by the cognitive modeling community. The best theoretical mechanisms for understanding individual-level cognitive performance across tasks are basically unknown. The PGA optimization approach could uncover strategies for parametric across-task modeling by running different tasks in parallel and evaluating the results as evolved over the generations. Or as an alternative, the PGA could simulate the sequence of tasks in the experimental protocol by running one task model for a specific number of generations with the resulting genotypes passed on for running the next task model for another set of generations and so forth.

With the choice of several execution formats, the thesis optimization could enable fitting individual subjects across tasks thus advancing the state of the art in cognitive modeling. Across task modeling would also allow modelers to extend and test their cognitive architectures in new ways.

Available and extendable by modelers. A large cluster computing resource was used for the thesis project. Most medium- to large-sized universities have some type of computing

cluster resource. With the exception of the serial subtraction model, all other applications used in the thesis project are open source (CMUCL, ACT-R) or reside on the cluster itself (C, MPI).

As mentioned above the thesis optimization technique could be used to fit other models besides the serial subtraction model. In theory, any parameterized cognitive model could be modified in the same way as the serial subtraction model to run in a parallel processing environment. The primary modifications required include reformatting the model's startup function to accept parameter values coming in as arguments and rewriting the backend function to support the fitness criteria of the PGA.

As demonstrated in Chapter 3, different types of search algorithms could be applied, even in combination, to find the best model-to-data fits. In the thesis project, the majority of the search algorithm code was written separately from the cognitive model, interfacing with the model only in the fitness function. For example, the basic code for the PGA was taken from a textbook and then modified to incorporate the cognitive architecture and model.

With the availability of open source applications and academic resources, and easy integration of a cognitive architecture and model, the thesis project can be adopted for use by other cognitive modelers using different architectures.

Limitations

As present in any research study, limitations exist, sometimes known in advance of the study and other times realized after the fact. The thesis project has several limitations related to each of its components. This subsection briefly discusses limitations associated with the human performance data set, the cognitive modeling component, and the HPC platform resource where the thesis project resides.

Data set limitations. Only men were included in the human subject experiment where data were collected from the serial subtraction task. Women were not included because of the hormonal variation associated with the menstrual cycle which can impact caffeine metabolism and stress hormone levels. Much research has already been done on gender differences in response to stress (Taylor et al., 2000; Turton & Campbell, 2005) with some of this research specifically focused on math performance (Miller & Bichsel, 2004; O'Brien & Crandall, 2003). There are no known studies that utilize a cognitive model of gender differences in stress responses. For the most part, the cognitive modeling community has shown no interest in modeling gender difference in any aspect of cognitive performance.

In the experiment each subject performed four blocks of serial subtraction, two blocks of subtracting by 7s and two blocks of subtracting by 13s. Chapter 5 discusses distributions of model predictions produced in the optimization to each subject's data. The fitness criteria of the PGA could be strengthened by comparing distributions of performance between the model and the human subjects. Figure 5.5 showed that distribution comparisons could be modeling using the ACT-R architecture.

Unfortunately, in the experimental protocol, each subject only performed the serial subtraction task twice for each type of subtraction. Obviously, two trials are not enough data to create a distribution. Time and expense can be considerable when subjects perform multiple trials of the same task, but multiple trials would benefit the test of model fit especially when using an automated search approach.

Modeling limitations. Several limitations are related to the modeling component of the thesis: only three architectural parameters were manipulated during fitting, and only one target model was optimized, the serial subtraction model. Roberts and Pashler (2000) suggested that the behavior of cognitive models should be studied over their entire range of possible parameters to determine not only what data models account for, but also what data they cannot explain. It is an

open question what ‘model parameters’ are. Real-valued architectural and knowledge parameters seem to qualify. Baker, Corbett, and Koedinger (2003) suggested that every knowledge structure, such as a chunk and production rule, should be counted as a free parameter. This seems a bit extreme because the architecture has over 80 parameters, and the serial subtraction model has over 200 declarative knowledge chunks and 25 productions.

The thesis investigated the impact of variations of only three parameters directly involved in the declarative memory retrieval process and speech production thought to be central to individual differences in performance of the serial subtraction task. As mentioned in Chapter 5, there are several other parameters involved in the subsymbolic processes of declarative memory. BLC and ANS were selected as first choices to investigate; there may be other better choices or combinations.

Development of the PGA optimization approach was specifically designed to optimize the serial subtraction model. The model had been recently rewritten to run under the new version of the ACT-R architecture therefore making it an ideal candidate as a target model for the thesis optimization project. Fitting of other cognitive models in addition to the serial subtraction model would be critical in evaluating the usefulness of this new optimization approach for cognitive modeling.

HPC platform limitations. The thesis project was set up and run on a large cluster at NCSA with a total of 2500 available processors. For a single run of the PGA, usually 200 processors were requested for several hours CPU time. Because the cluster itself was so large, the optimization jobs when submitted to the batch queue usually ran within two days time—not a long wait considering many clusters are oversubscribed. The requested number of processors (200) allowed each genotype in the population to be assigned to a single processor with the population evolved for 100 generations. In this case each run of the PGA sampled 20,000 parameter combinations in a rather tightly constrained ACT-R parameter space. These conditions

were ideal for development of the thesis project. As mentioned in Chapter 3, it took several months of HPC resource investigations and an NSF development allocation to find and access an appropriately sized cluster with a Unix-based operating system needed for the project.

As of July 21, 2008, the cluster that the thesis project resides is going out of commission. This means that for the project to continue in any capacity another cluster will need to be located and the project set up again from scratch.

Conclusions

The purpose of the thesis was to better enable an individual differences investigation of how stress affects cognitive performance in a mental serial subtraction task. To do this, a new optimization approach was developed for computing the fit of a serial subtraction cognitive model to human performance data by varying architectural parameters.

Fitting a cognitive model to data is a stochastic global optimization problem. The thesis optimization approach utilizes a modified PGA on a HPC platform together with the ACT-R cognitive architecture and the serial subtraction cognitive model. The modification of the PGA was necessitated by stochasticity embedded in the cognitive architecture and model causing lack of convergence on a solution set. The PGA was redesigned to compensate for a single parameter set returning a distribution of performance predictions instead of a static performance statistic. To ameliorate the lack of convergence, best fitting genotype collection and testing features were added to the PGA.

The PGA optimization approach was highly successful in fitting the serial subtraction model to three different levels of human data: average across subjects, challenge and threat task appraisal groups, and single-subject level performance. The optimization results revealed several patterns in the parameter values found to produce best fits to the human data. Two of the patterns

are supported by individual differences theories of stress and anxiety from cognitive performance research.

One pattern involved the large proportion of high ANS (activation noise parameter) values. High ANS values are thought to be related to the fact that the serial subtraction task itself is stressful. These high ANS values may be indicative of the large amount of variance observed in the subjects' performance and noise usually associated with individual differences. In ACT-R high ANS contributes to greater variability in both the probability of retrieving a declarative memory fact and the retrieval time. This, in turn, can be thought to simulate distraction and interference caused by anxiety and worrisome thoughts of poor performance on the attentional resources of working memory.

The strongest pattern involved SYL, the seconds per syllable parameter. The optimization results showed that as performance on serial subtraction decreased, the value of SYL increased. This indicates that high performing subjects were speaking more quickly than their poor performing counterparts. In cognitive psychology literature, the concept of worry has been linked to inner verbal activity. The optimization solutions simulated a subject's worry over his performance as interfering or competing with his speaking of the subtraction answers with the effect of impairing subtraction performance.

In the cognitive science field, the traditional manual optimization process in many cases is only a step above trial-and-error (Ritter, 1990). If the manual optimization process had been used to fit the serial subtraction model to the human performance data, the above described patterns would have gone undiscovered. The results reported in the thesis establish that the PGA-based optimization approach could supersede the field's manual optimization process. Additional contributions of the thesis optimization approach to cognitive science research include: development of a software system for individual differences modeling and analysis; increased understanding of the ACT-R cognitive architecture and its parameterization; increased

understanding of how stress and anxiety effect cognitive performance on serial mathematics tasks; a new application of PGAs in the domain of cognitive science; and a new paradigm for cognitive model creation and validation.

Appendix A

Post-task Appraisal

Directions: For each of the following questions, please circle the answer on the scale below the question which best describes how you feel about the task you just completed.

1. How stressful did you find the task to be?

1	2	3	4	5
Not at all Stressful		Moderately Stressful		Very Stressful

2. How well do you think you were able to cope with the task?

1	2	3	4	5
I did not cope well with the task.				I coped very well with the task.

3. How demanding did you find the task to be?

1	2	3	4	5
Not at all Demanding		Moderately Demanding		Very Demanding

4. How threatening did you find the task to be?

1	2	3	4	5
Not at all Threatening		Moderately Threatening		Very Threatening

5. How well did you perform the task?

1	2	3	4	5
I did not perform well at all.				I performed very well.

Appendix B

Glossary

ACT-R. Adaptive Control of Thought – Rational (Anderson et al., 2004) is a production-system type of cognitive architecture developed by a community of researchers led by John Anderson at Carnegie Mellon University. ACT-R is a two-layer, modular architecture where cognition emerges through the interaction of a number of independent modules.

ANS. The activation noise parameter is a subsymbolic component of ACT-R's declarative module and the activation equation. The value of ANS affects the probability of retrieving declarative memory chunks and the time it takes for retrieval. This parameter value represents instantaneous noise and can be used to simulate trial-to-trial variance.

BLC. The base level constant parameter is a subsymbolic component of ACT-R's declarative module and the base level activation equation. The BLC value is a constant value summed into the base-level activation equation resulting in an increase to a declarative memory chunk's base-level activation. Base-level activation represents the recency and frequency of the use of the chunk, therefore, a BLC value over 0 acts as constant increase of activation value.

GA. Genetic algorithms are a numerical optimization procedure based on evolutionary principles. They are useful in maximizing or minimizing an objective function within a set of constraints. A population of fixed-length strings is evolved by application of selection, crossover, and mutation operators along with a fitness function that determines a string's suitability to the problem. The process is continued through a number of generations with the aim that the population should evolve to contain an acceptable solution.

HPC. High performance computing is a branch of computer science that concentrates on developing supercomputers and applications to run on supercomputers. A main area of this discipline is developing parallel processing algorithms and software programs that can be divided into pieces so that each piece can be executed simultaneously by separate processors.

MPI. The Message Passing Interface is a widely used standard for creating applications for parallel processing. Applications can use message-passing mechanisms for communication between processors. MPI is a library of subprograms that can be included in C, C++, or Fortran programs for communication between processors.

NCSA. The National Center for Supercomputing Applications is part of the Teragrid. The Teragrid is considered to be the world's most comprehensive distributed cyberinfrastructure for open scientific research. The thesis project was setup on a cluster located at NCSA.

PGA. Parallel genetic algorithms are parallel implementations of GAs that can provide considerable gains in terms of performance and scalability. PGAs are usually implemented on networks or parallel computers. An important design consideration of PGAs involves the choice of using a single or multiple populations.

SYL. The seconds-per-syllable parameter controls the rate of rudimentary speech of a cognitive model when running in the ACT-R cognitive architecture. The seconds-per-syllable parameter is a subsymbolic component in ACT-R's vocal module. The default timing for speech is 0.15 seconds per assumed syllable based on a three-character syllable.

Appendix C

Previous Publications from Thesis

Kase, S. E., Ritter, F. E., & Schoelles, M. (2008). HPC + PGA + ACT-R = perfect fits (lacking validation?). Presented at the *Fifteeth Annual ACT-R Workshop*, Pittsburgh, PA, July 2008.

Kase, S. E., Ritter, F. E., & Schoelles, M. (2008). From modeler-free individual data fitting to 3-D parametric prediction landscapes: A research expedition. *Proceedings of the 30th Annual Meeting of the Cognitive Science Society*.

Kase, S. E., Ritter, F. E., & Schoelles, M. (2007). Using HPC and PGAs to optimize noisy computational models of cognition. *International Joint Conferences on Computer, Information, and System Sciences and Engineering.*

Bibliography

Anderson, J. R. (1983). *The architecture of cognition*. Cambridge, MA: Harvard University Press.

Anderson, J. R. (1990). *The adaptive character of thought*. Hillsdale, NJ: Erlbaum.

Anderson, J. R. (2005). Human symbol manipulation within an integrated cognitive architecture. *Cognitive Science, 29*, 313-342.

Anderson, J. R. (2007). *How can the human mind occur in the physical universe?* Oxford, NY: Oxford University Press.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Quin, Y. (2004). An integrated theory of the mind. *Psychological Review, 111*, 1036-1060.

Anderson, J. R., & Lebiere, C. (1998). *The atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum.

Anderson, J. R., Reder, L. M., & Lebiere, C. (1996). Working memory: Activation limits on retrieval. *Cognitive Psychology, 30*, 221-256.

Aronson, J., Lustina, M. J., Good, C., Keough, K., Steele, C. M., & Brown, J. (1999). When White men can't do math: Necessary and sufficient factors in stereotype threat. *Journal of Experimental Social Psychology, 35*, 29-46.

Ashcraft, M. H., & Faust, M. W. (1994). Mathematics anxiety and mental arithmetic performance: An exploratory investigation. *Cognition and Emotion, 8*, 97-125.

Ashcraft, M. H., & Kirk, E. P. (2001). The relationships among working memory, math anxiety, and performance. *Journal of Experimental Psychology: General, 130*(2), 224-237.

Ashcraft, M. H., Kirk, E. P., & Hopko, D. (1998). On the cognitive consequences of mathematics anxiety. In C. Donlan (Ed.), *The development of mathematical skills* (pp. 175-196). East Sussex, Great Britain: Psychology Press.

Atkinson, R. C., & Shiffrin, R. M. (1971). The control of short-term memory. *Scientific American, 225*, 82-90.

Axtell, R., Axelrod, R., Epstein, M. J., & Cohen, M. D. (1996). Aligning simulation models: A case study and results. *Computational and Mathematical Organization Theory, 1*, 123-141.

Baddeley, A. D. (1986). *Working memory*. Oxford, UK: Oxford University Press.

Baddeley, A. D. (1990). *Human memory: Theory and practice*. Boston, MA: Allyn and Bacon.

Baddeley, A. D. (1992). Working memory. *Science, 255*, 556-559.

Baddeley, A. D., & Hitch, G. (1974). Working memory. In G. H. Bower (Ed.), *The psychology of learning and motivation* (pp. 47-89). New York, NY: Academic Press.

Baker, R. S., Corbett, A. T., & Koedinger, K. R. (2003). *Statistical techniques for comparing ACT-R models of cognitive performance.* Paper presented at the Proceedings of the 2003 ACT-R Workshop, Pittsburgh, PA.

Beilock, S. L., & Carr, T. H. (2001). On the fragility of skilled performance: What governs choking under pressure? *Journal of Experimental Psychology: General, 130*, 701-725.

Beilock, S. L., & Carr, T. H. (2005). When high-powered people fail: Working memory and "choking under pressure" in math. *Psychological Science, 16*(2), 101-105.

Beilock, S. L., Kulp, C. A., Holt, L. E., & Carr, T. H. (2004). More on the fragility of performance: Choking under pressure in mathematical problem solving. *Journal of Experimental Psychology: General, 133*, 584-600.

Bell, C. G., & Newell, A. (1971). *Computer structures: Readings and examples*. New York: McGraw-Hill.

Blascovich, J., & Mendes, W. B. (2000). Challenge and threat appraisals: The role of affective cues. In J. P. Forgas (Ed.), *Feelings and thinking: The role of affect in social cognition* (pp. 59-82). New York: Cambridge University Press.

Blascovich, J., Mendes, W. B., Solomon, K., & Hunter, S. B. (1999). Social facilitation as challenge and threat. *Journal of Personality and Social Psychology, 77*, 68-77.

Borkovec, T. (1994). The nature, functions and origins of worry. In G. Davey & F. Tallis (Eds.), *Worrying: Perspectives on theory, assessment and treatment* (pp. 5-34). Chichester, England: Wiley.

Borkovec, T., & Inz, J. (1990). the nature fo worry in generalized anxiety disorder: A predominance of thought activity. *Behavior Research and Therapy, 28*, 153-158.

Bray, D. A., & Prietula, M. (2007, December). *Extending March's exploration and exploitation: Managing knowledge in turbulent environments.* Paper presented at the 28th International Conference on Information Systems.

Brooks, F. P. (1962). *Advanced computer organization: Addressing.* Paper presented at the Proceedings of IFIP Congress, Munich, Germany.

Burton, R. M. (2003). Computational laboratories for organizational science: Questions, validity and docking. *Computational and Mathematical Organization Theory, 9*, 91-108.

Butterworth, B., Cipolotti, L., & Warrington, E. K. (1996). Short-term memory impairment and arithmetical ability. *Quarterly Journal of Experimental Psychology, 49A*, 251-262.

Cantu-Paz, E. (2001). *Efficient and accurate parallel genetic algorithms*. Norwell. MA: Kluwer Academic Publishers.

Carley, K. M., Prietula, M. J., & Lin, Z. (1998). Design versus cognition: The interaction of agent cognition and organizational design on organizational performance. *Journal of Artificial Societies and Social Simulation, 1*(3).

Chuderski, A., Stettner, Z., & Orzechowski, J. (2006). *Modeling individual differences in working memory search task.* Paper presented at the Proceedings of the Seventh International Conference on Cognitive Modeling, Trieste, Italy.

Chuderski, A., Stettner, Z., & Orzechowski, J. (2007). Computational modeling of individual differences in short term memory search. *Cognitive Systems Research, 8*, 161-173.

Coley, D. A. (1999). *An introduction to genetic algorithms for scientists and engineers.* Hackensack, NJ: World Scientific Publishing.

Cooper, L. R., Corne, D. W., & Crabbe, M. J. (2003). Use of a novel hill-climbing genetic algorithm in protein folding simulations. *Computational Biology and Chemistry, 27*, 575-580.

Cowan, N. (2001). The magical number 4 in short-term memory: A reconsideration of mental storage capacity. *Behavioral and Brain Sciences, 26*, 87-116.

Daily, L. Z., Lovett, M. C., & Reder, L. M. (1999). *Cross-task prediction of working memory performance: Working memory capacity as source activation.* Paper presented at the Sixth Annual ACT-R Workshop, Fairfax, VA.

Daily, L. Z., Lovett, M. C., & Reder, L. M. (2001). Modeling individual differences in working memory performance: A source activation account. *Cognitive Science, 25*, 315-353.

Darke, S. (1988). Effects of anxiety on inferential reasoning task performance. *Personality and Social Psychology, 55*, 499-505.

DeToro, F., Ortega, J., Fernandez, J., & Diaz, A. (2002). *PSFGA: A parallel genetic algorithm for multiobjective optimization.* Paper presented at the Proceedings of the Tenth Euromicro Workshop on Parallel, Distributed and Network-based Processing, Spain, Canary Islands.

Dyba, T. (2000). Improvisation in small software organizations. *IEEE Software, 17*(5), 82-87.
Engle, R. W. (2002). Working memory capacity as executive attention. *Current Directions in Psychological Science, 11*, 19-23.
Eysenck, M. W. (1992). *Anxiety: The cognitive perspective*. Hove, England: Erlbaum.
Eysenck, M. W., & Calvo, M. G. (1992). Anxiety and performance: The processing efficiency theory. *Cognition and Emotion, 6*, 409-434.
Eysenck, M. W., Derakshan, N., Santos, R., & Calvo, M. G. (2007). Anxiety and cognitive performance: Attentional control theory. *Emotion, 7*(2), 336-353.
Friedenberg, J., & Silverman, G. (2006). *Cognitive science: An introduction to the study of mind*. Thousand Oaks, CA: Sage Publications.
Gluck, K. A., & Harris, J. (2008a). *Mindmodeling@Home.* Paper presented at the 30th Annual Meeting of the Cognitive Science Society, Washington, DC.
Gluck, K. A., & Harris, J. (2008b). *The MindModeling meta-computing infrastructure.* Paper presented at the Fifteenth Annual ACT-R Workshop, Carnegie Mellon University, Pittsburgh, PA.
Gobet, F., & Ritter, F. E. (2000). *Individual data analysis and unified theories of cognition: A methodological proposal.* Paper presented at the Proceedings of the 3rd International Conference on Cognitive Modeling.
Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine Learning*. Reading, MA: Addison Wesley.
Goldberg, D. E. (1994). Genetic and evolutionary algorithms come of age. *Communications of the ACM, 37*(3), 113-119.
Gunzelmann, G. (2008). *Enabling scientific progress through the use of large scale computing resources.* Paper presented at the Fifteeth Annual ACT-R Workshop, Carnegie Mellon University, Pittsburgh, PA.
Hasher, L., Lustig, C., & Zacks, R. (2008). Inhibitory mechanisms and the control of attention. In *Variation in working memory* (pp. 227-249). New York, NY: Oxford University Press.
Haupt, R. L., & Haupt, S. E. (2004). *Practical genetic algorithms* (2 ed.). Hoboken, NJ: Wiley & Sons, Inc.
Hinterding, R., Gielewski, H., & Peachey, T. C. (1995). *The nature of mutation in genetic algorithms.* Paper presented at the Proceedings of the 6th International Conference on Genetic Algorithms.
Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor, MI: University of Michigan Press.
Hoos, H. H., & Stutzle, T. (2000). Local search algorithms for SAT: An empirical evaluation. *Journal of Automated Reasoning, 24*, 421-481.
Introduction on Evolutionary Algorithms. Retrieved May 15, 2007.
Ismail, M. A. (2004). Parallel genetic algorithms (PGAs): Master slave paradigm approach using MPI. *IEEE e-Tech, 31*, 83-87.
Just, M. A., & Carpenter, P. N. (1992). A capacity theory of comprehension: Individual differences in working memory. *Psychological Review, 99*, 122-149.
Kane, M. J., & Engle, R. W. (2000). Working-memory capacity, proactive interference, and divided attention: Limits on long-term memory retrieval. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 26*, 336-358.
Kane, M. J., & Engle, R. W. (2002). The role of prefrontal cortex in working-memory capacity, executive attention, and general fluid intelligence: An individual-differences perspective. *Psychonomic Bulletin and Review, 9*, 637-671.

Kase, S. E., Ritter, F. E., & Schoelles, M. (2007). *Using HPC and PGAs to optimize noisy computational models of cognition.* Paper presented at the International Joint Conference on Computer, Information, and System Sciences and Engineering.

Kase, S. E., Ritter, F. E., & Schoelles, M. (2008). *From modeler-free individual data fitting to 3-D parametric landscapes: A research expedition.* Paper presented at the 30th Annual Meeting of the Cognitive Science Society, Washington, DC.

Kirschbaum, C., Pirke, K. M., & Hellhammer, D. H. (1993). The Trier Social Stressor Test—A tool for investigating psychobiological stress responses in a laboratory setting. *Neuropsychobiology, 28*, 76-81.

Klein, L. C., Whetzel, C. A., Bennett, J. M., Ritter, F. E., & Granger, D. A. (2006). Effects of caffeine and stress on salivary alpha-amylase in young men: A salivary biomarker of sympathetic activity. *Psychosomatic Medicine, 68*(1), A-4.

Koza, J., Martin, K., & Streeter, M. (2003). Evolving inventions. *Scientific American, 228*(2), 52-59.

Lazarus, R. S., & Folkman, S. (1984). *Stress, appraisal and coping*. New York: Springer Publishing.

Lebiere, C. (1998). *The dynamics of cognition: An ACT-R model of cognitive arithmetic.* Ph. D. Dissertation. Computer Science Department Technical Report CMU-CS-98-186. Carnegie Mellon University, Pittsburgh.

Lee, M. D., & Webb, M. R. (2005). Modeling individual differences in cognition. *Psychonomic Bulletin and Review, 12*, 605-621.

Lewandowsky, S., & Heit, E. (2006). Some target for memory models. *Journal of Memory and Language, 55*, 441-446.

Lovett, M. C., Daily, L. Z., & Reder, L. M. (2000). A source activation theory of working memory: Cross-task prediction of performance in ACT-R. *Cognitive Systems Research, 1*, 99-118.

Lovett, M. C., Reder, L. M., & Lebiere, C. (1997). *Modeling individual differences in a digit working memory task.* Paper presented at the Proceedings of the Nineteenth Annual Conference of the Cognitive Science Society.

Major, B., & O'Brien, L. T. (2005). The social psychology of stigma. *Annual Review of Psychology, 56*, 393-421.

March, J. G. (1991). Exploration and exploitation in organizational learning. *Organizational Science, 2*, 71-87.

Meyer, D. E., & Kieras, D. E. (1997). A computational theory of executive cognitive processes and multiple-task performance: Part 1: Basic mechanisms. *Psychological Review, 104*(1), 3-65.

Michalewicz, Z. (1994). *Genetic algorithms + data structures = evolution programs* (2nd ed.). Heidelberg: Springer-Verlag.

Miller, H., & Bichsel, J. (2004). Anxiety, working memory, gender, and math performance. *Personality and Individual Differences, 37*(3), 591-606.

Mitchell, M. (1997). *An introduction to genetic algorithms*. Cambridge, MA: MIT Press.

Miwa, K., & Simon, H. A. (1993). Production system modeling to represent individual differences: tradeoff between simplicity and accuracy in simulation of human behavior. In A. Sloman, D. Hogg, G. Humphreys, A. Ramsay & D. Partidge (Eds.), *Prospects for artificial intelligence* (pp. 158-167). Amsterdam, Netherlands: Kaigai Publications.

Miyake, A., & Shah, P. (Eds.). (1999). *Models of working memory: Mechanisms of active maintenance and executive control*. New York: Cambridge University Press.

Muhlenbein, H. (1992). How do genetic algorithms really work? 1. Mutation and hill-climbing. In R. Manner & B. Manderick (Eds.), *Parallel problem solving from nature 2*. Amsterdam: Elsevier.

Newell, A. (1990). *Unified theories of cognition*. Cambridge, MA: Cambridge University Press.

Newell, A., & Simon, H. A. (1972). *Human problem solving*. Englewood Cliffs, NJ: Prentice-Hall.

Nosofsky, R. M., & Palmeri, T. J. (1997). An exemplar-based random walk model of speeded classification. *Psychological Review, 104*, 266-300.

O'Brien, L. T., & Crandall, C. S. (2003). Stereotype threat and arousal: Effects on women's math performance. *Personality and Social Psychology Bulletin, 29*(6), 782-789.

Pacheco, P. S. (1997). *Parallel programming with MPI*. San Francisco, CA: Morgan Kaufmann Publishers.

Pitt, M., Myung, I., & Zhang, S. (2002). Toward a method of selecting among computation models of cognition. *Psychological Review, 109*, 472-491.

Power, M. J., & Dalgleish, T. (1997). *Cognition and emotion: From order to disorder*. Hove, England: Psychology Press.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (1992). *Numerical recipes in C* (Second ed.). Cambridge: Cambridge University Press.

Pylyshyn, Z. W. (1989). Computing in cognitive science. In M. Posner (Ed.), *Foundations of cognitive science* (pp. 49-92). Cambridge, MA: MIT Press.

Rapee, R. M. (1993). The utilization of working memory by worry. *Behaviour Research and Therapy, 31*, 617-620.

Rehling, J., Lovett, M. C., Lebiere, C., Reder, L. M., & Demiral, B. (2004). *Modeling complex tasks: An individual differences approach.* Paper presented at the Proceedings of the 26th Annual Conference of the Cognitive Science Society.

Renders, J. M., & Bersini, H. (1994). *Hybridizing genetic algorithms with hill-climbing methods for global optimization: Two possible ways.* Paper presented at the International Conference on Evolutionary Computation.

Ritter, F. E. (1990). *Mendel-DP: Optimizing PDP learning rates.* Paper presented at the Eighth Annual Pitt-CMU Conference and CMU PDP Seminar, Pittsburgh, PA.

Ritter, F. E. (2004). Choosing and getting started with a cognitive architecture to test and use human-machine interfaces. *MMI-Interaktiv-Journal's special issue on Modeling and Simulation in Human-Machine Systems, 7*, 17-37.

Ritter, F. E., Bennett, J. M., Kase, S. E., Rodrigues, I., & Klein, L. C. (submitted). Cognitive performance on a serial subtraction task. *Memory and Cognition*.

Ritter, F. E., Reifers, A., Klein, L. C., Quigley, K. S., & Schoelles, M. (2004). *Using cognitive modeling to study behavior moderators pre-task appraisal and anxiety.* Paper presented at the Proceedings of the Human Factors and Ergonomics Society, Santa Monica, CA.

Ritter, F. E., Schoelles, M., Klein, L. C., & Kase, S. (2007). *Modeling the range of performance on the serial subtraction task.* Paper presented at the Proceedings of the 8th International Conference on Cognitive Modeling.

Roberts, S., & Pashler, H. (2000). How persuasive is a good fit? A comment on theory testing. *Psychological Review, 107*, 358-367.

Rothaermel, F. T., & Deeds, D. L. (2004). Exploration and exploitation alliances in biotechnology: A system of new product development. *Strategic Management Journal, 25*(3), 201-221.

Salthouse, T. A. (1992). Why do adult age differences increase with task complexity? *Developmental Psychology, 28*, 905-918.

Schmader, T., & Johns, M. (2003). Converging evidence that stereotype threat reduces working memory capacity. *Journal of Personality and Social Psychology, 85*, 440-452.

Schmader, T., Johns, M., & Forbes, C. (2008). An integrated process model of stereotype threat effects on performance. *Psychological Review, 115*(2), 336-356.

Schneider, W., & Chein, J. M. (2003). Controlled and automatic processing: Behavior, theory, and biological mechanisms. *Cognitive Science, 27*, 525-559.

Siegler, R. S. (1987). The perils of averaging data over strategies: An example from children's addition. *Journal of Experimental Psychology: General, 116*, 250-264.

Sorg, B., & Whitney, P. (1992). The effect of trait anxiety and situational stress on working memory capacity. *Journal of Research in Personality, 26*, 235-241.

Steele, C. M. (1988). The psychology of self-affirmation: Sustaining the integrity of the self. In L. Berkowitz (Ed.), *Advances in experimental social psychology* (Vol. 21, pp. 261-302). San Diego, CA: Academic Press.

Steele, C. M. (1997). A threat is in the air: How stereotypes shape intellectual identity and performance. *American Psychologist, 52*, 613-629.

Steele, C. M., & Aronson, J. (1995). Stereotype threat and the intellectual test performance of African Americans. *Journal of Personality and Social Psychology, 69*, 797-811.

Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. *Cognitive Science, 12*(1), 257-285.

Taylor, S. E., Klein, L. C., Lewis, B. P., Gruenewald, T. L., Gurung, R., & Updegraff, J. A. (2000). Biobehavioral responses to stress in females: Tend-and-befriend, not fight-or-flight. *Psychological Review, 107*(3), 411-429.

Tomaka, J., Blascovich, J., Kelsey, R. M., & Leitten, C. L. (1993). Subjective, physiological, and behavioral effects of threat and challenge appraisal. *Journal of Personality and Social Psychology, 65*, 248-260.

Tomaka, J., Blascovich, J., Kibler, J., & Ernst, J. M. (1997). Cognitive and physiological antecedents of threat and challenge appraisal. *Journal of Personality and Social Psychology, 73*, 63-72.

Tor, K., & Ritter, F. E. (2004). *Using a genetic algorithm to optimize the fit of cognitive models.* Paper presented at the Proceedings of the International Conference on Cognitive Modeling.

Trick, L. M., & Phylyshyn, Z. W. (1994). Why are small and large numbers enumerated differently? A limited-capacity preattentive stage in vision. *Psychological Review*(101), 80-102.

Turton, S., & Campbell, C. (2005). Tend and befriend versus fight or flight: gender differences in behavioral response to stress among university students. *Journal of Applied Biobehavioral Research, 10*(4), 209-232.

Whetzel, C. A., Ritter, F. E., & Klein, L. C. (2006). DHEA-S and cortisol responses to stress and caffeine in healthy young men: Is DHEA-S a reliable marker for stress? *Psychosomatic Medicine, 68*(1), A-77.

Woolcock, M. (1998). Social capital and economic development: Toward a theoretical synthesis and policy framework. *Theory and Society, 27*(2), 151-208.

Yang, C., & Ye, H. (2002). *A distributed algorithm for function optimization based on artificial life.* Paper presented at the International Conference on Control and Automation.

Young, R. (2003). *A new view of parameter search.* Paper presented at the Tenth Annual ACT-R Workshop, Carnegie Mellon University, Pittsburgh, PA.

VITA

Sue Ellen Kase

Sue Ellen Kase received a Bachelor of Arts (B.A.) in Art from West Chester University (West Chester, PA USA) and a Bachelor of Fine Arts (B.F.A) from Philadelphia University of the Arts (Philadelphia, PA USA) in May 1984. She was employed as a graphic designer until beginning graduate studies in computer science. Kase received a Master of Science (M.S.) in Computer Science from State University of New York (New Paltz, NY USA) in May 1992. She held several computer programming positions centered on recording and imaging applications, training simulations, and educational games for children. In August of 1999, Kase became an instructor for the Computer Science and Engineering Department at the Pennsylvania State University (University Park, PA USA). She taught a variety of undergraduate courses in computer science while completing graduate course work in the College of Information Sciences and Technology. Kase completed the requirements for a Ph.D. in Information Sciences and Technology from the Pennsylvania State University in December 2008.

The focus of Kase's interdisciplinary research is to understand how stress influences human cognition in the performance of mental arithmetic tasks. Her research utilizes cognitive architectures, parametric modeling, and high-performance optimization techniques. As a co-principal investigator, Kase was recipient of two NSF TeraGrid DAC (Development Allocation Committee) awards supplying the high-performance computing resources needed to complete her thesis (TG-IRI070000T, 2006 – 2009; 60,000 Service Units). In addition, Kase participated in research projects conducted in the Center for Human-Computer Interaction and Computer-Supported Collaboration and Learning Laboratory headed by Dr. John Carroll and Dr. Mary Beth Rosson at the Pennsylvania State University. Kase has published in the areas of cognitive science, parallel genetic algorithm optimization, agent simulation, end-user programming, community informatics, and gender differences in mathematics and computing.