An Agent-Based Distributed Decision Support System Framework for Mediated Negotiation

Frank A. LoPinto

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in
Business

Dr. Cliff T. Ragsdale, Chairman

Dr. Loren Paul Rees

Dr. Roberta Russell Tillar

Dr. Lance A. Matheson

Dr. Christopher W. Zobel

April 26, 2004
Blacksburg, Virginia

Keywords: Decision Support Systems, Preferences, Conjoint Analysis, Agents, Web Services, Linear Programming

An Agent-Based Distributed Decision Support System Framework for Mediated Negotiation

Frank A. LoPinto

(ABSTRACT)

Implementing an e-market for limited supply perishable asset (LiSPA) products is a problem at the intersection of online purchasing and distributed decision support systems (DistDSS). In this dissertation, we introduce and define LiSPA products, provide real-world examples, develop a framework for a distributed system to implement an e-market for LiSPA products, and provide proof-of-concept for the two major components of the framework.

The DistDSS framework requires customers to instantiate agents that learn their preferences and evaluate products on their behalf. Accurately eliciting and modeling customer preferences in a quick and easy manner is a major hurdle for implementing this agent-based system. A methodology is developed for this problem using conjoint analysis and neural networks.

The framework also contains a model component that is addressed in this work. The model component is presented as a mediator of customer negotiation that uses the agent-based preference models mentioned above and employs a linear programming model to maximize overall satisfaction of the total market.

DEDICATION

I dedicate this work to my wife, Stacie, and to our sons, Frankie, Dominic, Vincent, and Joseph. Stacie, endeavoring upon this journey would have been impossible without your support, understanding, and adventurous nature. Your patience and belief in my abilities are the reasons for our success.

Boys, your existence alone provides the daily motivation I need to be successful, and your individual and collective success continually inspires me to do the best work possible.

ACKNOWLEDGEMENTS

I would first like to thank my major professor, Dr. Cliff T. Ragsdale, for all your patience and attention. Any success I achieve in my career will be rooted in the firm foundation that was constructed during my time under your guidance. In your work and professionalism I have seen the top of the mountain. You are truly a master at your craft.

Thank you to my committee members: Dr. Loren Rees, Dr. Roberta Russell Tillar, Dr. Lance Matheson, and Dr. Christopher Zobel. Thank you all for your encouragement, guidance, and friendship.

Thank you to the entire BIT faculty and staff for your support in research, teaching and administrative hoop-jumping as well as for your friendship. This is truly a special department.

Finally, thank you to my family and friends for all your support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction and Literature Review

Introduction

Decision Support Systems (DSS) research has been a fertile field during the past twenty years since the landmark text by Sprague and Carlson (1982). A DSS couples the intellectual resources of humans with the capabilities of the computer to improve the quality of decisions by people dealing with semi-structured problems (Gorry and Scott Morton, 1989). More recently, Distributed Decision Support Systems (DistDSS) has provided the basis for a consistent flow of research due in part to the growth of the World Wide Web (Web). DistDSS is best defined as:

> "… a DSS where the various DSS components- decision models, data and documents relevant to a decision, visualization technologies, data collection sensors, rules engines, etc. – are located on computers distributed over a global network, in a way that they can be seamlessly integrated and used by a decision maker." (Power 2003).

DSS researchers have explored theories, developed frameworks, and addressed many types of problems (Cubert & Fishwick, 1997; Pinson, Louca, & Moraitis, 1997; Bui & Lee, 1999; Hess, 1999; Bhargava & Power, 2001; Cohen, Kelly, & Medaglia, 2001; Liu, Ma, Zhou, Zhang, 2001). One problem that has not received much research attention to date is that of online purchasing and brokering of *limited supply perishable asset* products (LiSPA). We refer to this as the LiSPA problem. Examples of these types of products include seating for venues, spring break vacation packages, cruise vacation and hotel room assignment, teacher and/or classroom allocation, and timeshare vacation property management. Some of these problems will be used throughout this research for the purpose of demonstrating different aspects of addressing this problem. This dissertation addresses this general problem and suggests a framework and supporting methodologies for its implementation and solution via software agents.

The LiSPA problem involves a brokering aspect where there exists a many-to-one relationship between customers and a broker. Thus, many customers (likely geographically distributed) vie for products, either cooperatively or competitively. One of many product vendors can be represented by one broker. For example, a venue seating market (for a single venue) would have many customers and one vendor.

Although many of the techniques in this research can be individually applied to different products and be generalized to fit a wider range of e-commerce applications, this study is wholly limited to products that can be viewed as perishable assets in limited supply, meaning that once the useful period of the product passes the product is no longer available. For example, seats for a particular sports event (season or single game) will not generate revenue after the event is finished.

In addition, LiSPA products are similar but differentiable. For example, consider spring break vacation packages. Spring break vacation packages all include some location, at a hotel with some level of comfort rating, some level of transportation, and a level of inclusion of amenities. Likewise, seats for a venue are similar in that all seats are for the same venue, yet they differ in location: row, aisle, and section.

Finally, we are dealing with online purchasing. There is currently a lack of sophistication in online markets for purchasing and/or brokering these types of products. The extent of these online environments is typically limited to marketing, searching, and product purchase fulfillment (payment and delivery). An improvement to these systems would involve customers accessing the market through a Web interface, instantiating agents to act on their behalf, and allowing the agents to interact with the broker and perhaps each other to negotiate for products.

Objective of the Study

A broadly defined objective of this study is to work toward the implementation of a DistDSS and e-market for LiSPA products. Doing so requires extraction of the steps commonly involved in the current process, as well as identification of enhancements to the process that can be achieved as a result of this research. Chapters 2-4 of this study will incrementally advance the process.

Specifically, the objectives of this dissertation are to develop a DistDSS framework for a LiSPA e-market, provide proof-of-concept theory and application for the two major components of the system, and to contribute to the research literature in a number of areas. First, this work proposes a framework for an agent-based DistDSS for negotiations using an intermediary and identifies and formalizes the venue seating problem (a specific LiSPA case). Second, it employs conjoint analysis (CA) and a neural network (NN) to elicit and model individual customer preferences. Third, a methodology using NNs, linear programming, and a software agent paradigm is developed to handle negotiation and trading of products based on customer preferences using a mediator. This will result in increased service levels to customers in terms of automation and increased satisfaction.

Research Methodology

Current literature from the areas of Web-based simulation, Web Services, and Web-based DSS are integrated in order to develop a unique theoretical framework for a DistDSS for negotiations using an intermediary. This study is intended to extend theoretical DistDSS research through a LiSPA market. Design methodologies are employed to provide proof-of-concept for the theoretical claims made as part of this research. Management science and artificial intelligence techniques are used along with CA and a software agent-based paradigm to develop an elegant approach to representing customer preferences and solving the problem.

Scope and Limitations

This research draws from the areas of software agents, Web Services, DSS, DistDSS, NNs, linear programming, CA, preference modeling, and the problem of venue seating. While each of these elements provides contributions to the problem solution and the implementation of the framework, an endless amount of research can be pursued regarding individual issues associated with each area. This study is limited to finding a unique way to address this specific class of problems and does not include such issues as security of distributed systems and efficacy. Furthermore, the specific LiSPA scenarios described in this study are used for the purpose of discussion, and are not intended to be an exhaustive generalization of any of the example problems. Other LiSPA problems exist with different seating rules and goals. It is not the intent of this study to account for all methods.

Contributions of the Research

- This study provides a framework for a DistDSS that addresses the venue-seating problem, specifically, and in general, negotiation using an intermediary.
- In this research, we formalize the venue seating problem and provide a real-world example of the problem.

- This research proposes a methodology using CA to elicit and model individual online customer preferences for spring break vacation packages and compares the new modeling methodology to current, well-accepted techniques.
- In this study, we develop a market for timeshare property management where agents can find products for customers, and perhaps trade with other customers though a broker, while accounting for their preferences using the preference models mentioned above.

Plan of Presentation

This chapter provides background regarding LiSPA products, identification of the need for improvements in the area, and discussion of a number of potential components of a DistDSS to address these issues. Chapters 2, 3 and 4 are three separate but related papers written with the intent of publishing each as individual journal articles in the general stream of DistDSS, agent preference modeling, and mediated negotiation. Chapter 2 presents a framework for an agent-based DistDSS for negotiation using an intermediary and demonstrates its use via venue seating. Chapter 3 provides a methodology for eliciting and modeling customer preferences using CA and a NN. Chapter 4 develops an agent-based online mediated market where consumer agents evaluate and trade products through a mediator using the preference models constructed in Chapter 3 along with a linear programming model to increase overall customer satisfaction. Chapter 5 discusses our conclusions and future work, including potential improvements to our preference modeling methodology and alternative ways of developing an online market for the LiSPA products for variable pricing.

Chapter 2

A Framework for an Agent-Based

Distributed Decision Support System using Web Services

ABSTRACT

This paper describes an agent-based DistDSS framework for decisions involving negotiation through an intermediary that utilizes Web services for communication among components as well as instantiation and enablement of agents. Our DistDSS design uses an agent-based paradigm for modeling and representing customer preferences for agent negotiation. The framework is first presented theoretically with some real-world applications in mind, then using this framework, we describe a DistDSS system for the recently publicized problem of seating sports venues.

1. INTRODUCTION

The Semantic Web envisioned by Berners-Lee, Hendler, and Lassila (2001) is a ubiquitous connected virtual environment where people use various devices to assist them in dealing with the nuts and bolts of life's daily tasks. The literature is replete with research in the areas of ontologies, intelligent agents, and knowledge representation: the main drivers of the Semantic Web as outlined by Berners-Lee *et al*. (2001). Concurrent to the inevitable establishment of the Semantic Web is the evolution of Web-based Decision Support Systems (DSS). Some researchers tout the Web as one vast DSS tool, where search engines aid decision makers by providing countless sources of information for virtually any product, service, topic, or decision; a place where product prices are compared, airline tickets can be instantly ordered, or hotel reservations booked. Moreover, websites containing agent-assembled databases of these products and services can be searched and sorted in order to find the best deals. These tools do help users with the data-driven decision process, however, they do little for the model-driven Web-based DSS, hereafter referred to as Distributed DSS (DistDSS). Research in these types of DistDSS is also burgeoning (Bui & Lee, 1999; Cohen, Kelley, & Medaglia, 2001; Lang & Whinston, 1999; Pinson, Louca, & Moraitis, 1997).

DistDSS as discussed here assumes that (1) stakeholders of the system (users, developers, service providers, etc.) are geographically dispersed, and (2) the traditional decision support system (DSS) subsystems of dialog (interface), model, and data are also geographically distributed and most likely owned by separate entities (Sprague & Carlson, 1982). It may be obvious that the Web will be the enabler of DistDSS, however, the most effective theoretical frameworks and implementations are yet uncertain.

The Web has evolved from initially having only people as end-users to a more robust environment where people remain the end-users, but either knowingly or unknowingly, rely on the help of applications known as software agents when performing many Web-based tasks. The use of agents for everyday Web operation continues to grow and agents will have an enormous impact on DSS (Whinston, 1997).

Another technology that has gained much publicity lately is Web services, a new breed of software component that can be registered, described, and invoked over a network while providing the benefits of language, platform, and location independence (Glass, 2002; Nghiem, 2003). Web services are alternatively touted as either the next evolutionary step for the Web or the most recent falsely-hyped fad. Either way, Web services are being heavily explored in both technical magazines and academic literature. In addition, Web services ensure availability of the most current application versions and data privacy for service users. The individual strengths of agents and Web services make these technologies a natural fit for use in a DistDSS framework for decisions that are asynchronous and "pull" in nature.

This paper describes an agent-based DistDSS framework for decisions involving negotiation through an intermediary that utilizes Web services for communication among components as well as instantiation and enablement of agents. Web services are used as the basic communication and data exchange technology. This is important because of the modularity and asynchronous nature of this DistDSS framework. The framework is first presented theoretically with some real-world applications in mind, and then specifically demonstrated using the recently publicized problem of seating season ticket requestors for a sports venue (Fatsis, 2002).

Section 2 provides background on the areas of DistDSS, Web services, and software agents. Section 3 suggests a framework for a DistDSS using Web services and software agents. Section 4 provides a DistDSS design for the real-world venue seating problem using an agent-based paradigm that accounts for customer preferences. Section 5 presents our conclusions and opportunities for future work.

2. BACKGROUND

2.1 Decision Support Systems and the Web

To date, the primary DSS approach to incorporating the Web into the DSS field has been through the application of frameworks and implementations to specific problems as well as generalizations such as DSS generators. Bhargava and Power (2001) discuss a number of interesting insights about the Web and DSS. Of particular relevance are the issues regarding the under-utilization of the Web for Model-driven DSS, the paradigm of "Web as Computer," and the identification of future challenges for Web-based DSS. In terms of Web-based DSS, the vast majority of applications involve the Data-driven paradigm, with few real-world model-driven implementations. Viewing the "Web as Computer," firms can offer execution of application-specific Model-based DSS tools via the Web, saving potential users the cost and trouble of installing and maintaining complex DSS tools on their own. The challenges facing Web-based DSS fall into three categories: technological, economic, and social and behavioral. The primary technological issue is that traditional DSS applications are persistent whereas the Web is not based on persistent connections. Economically, the challenge is determining a viable payment model and the development of a market for Web-based DSS. Socially and behaviorally, Web-based DSS systems must be developed for both the frequent and the casual user. Additional insight can be observed in simulation research.

Page and Opper (2001) offer a vision of the future for the practice of Web-based simulation that is very consistent with that described by Bhargava and Power (2001) for Web-based DSS. Other simulation researchers have contemplated the benefits and detriments of the Web on simulation concluding that in order for Web-based simulation to truly be a successful cost-saving option in the private sector, it is critical to capitalize on the concept of component *reuse*. Corporations must cooperate by sharing components and standards. One way to help businesses embrace this "sharing" paradigm is by providing technology that assures their privacy. Experts agree object-oriented design (OOD) may provide the best solution to these challenges (Buss & Stork, 1996; Kilgore, Healy, & Kleindorfer, 1998). OOD allows for access to "black-box" objects through an object's interface as well as inheritance, but encapsulates private information, barring unauthorized access. Component-based simulation (Buss, 2000) is another option, where inheritance is not used, but interfaces provide a tighter coupling between components. MOOSE (multimodeling object-oriented simulation environment) is an environment that has been developed as a model repository that employs several of these component-based concepts (Cubert & Fishwick, 1997).

2.2 Software agents and DSS

Although researchers have not yet agreed upon a single definition for software agents, they are generally viewed as software applications that perform tasks on behalf of users, independently or with little guidance (Bui & Lee, 1999). The spectrum of definitions vary over agent characteristics such as level of autonomy, persistence, mobility, reactivity, and the ability to communicate with its environment (Hess, Rees, & Rakes, 2000; Tecuci, 1998). Furthermore, there are few agreed upon theories and architectures for the use of agents in DSS (Hess, 1999). Despite these facts, there is a steady stream of research dealing with the use of agents in DSS (Luo, Liu, & Davis, 2002; Oliver, 1996; Pinson, Louca, & Moraitis, 1997; Tewari, Toull, & Maes, 2002).

The existing literature contains implementations as well as a number of frameworks. Bui and Lee (Bui & Lee, 1999) provide an excellent description of software agents from a DSS perspective, including a taxonomy of agent characteristics and a development lifecycle for building agent-based DSS.

2.3 Web Services

The Web was originally built for the purpose of sharing information among researchers. As the number of users increased there was a need for better navigation tools like browsers (e.g., Mosaic, Netscape, and Internet Explorer) to render Web pages for people to view. Just as these Web pages and browsers are built for use by people, Web services are applications that are built for use by other applications on the Web (i.e., they are developed with application-to-application communication in mind).

Although Web services have been embraced by Microsoft, and thus can be developed fairly easily using Microsoft .Net technology (Kilgore, 2002), Web services now have the backing of most major information technology, software, and hardware organizations (Huhns, 2002; Vaughan-Nichols, 2002). At the heart of Web services is its main communication language, XML. XML is used because it is an open standard language that is readable by a computer as well as a human. This provides the platform for Web services communication as well as intuitive code deciphering for humans.

Logistically, a Web service is an application that is located on a Web server that is "called" by another application (usually located on a remote machine) over a network. The Web service performs its task, then sends a reply back to the requestor. This may be compared to performing a function call over the Web.

The uniqueness of Web services can be attributed to the combination of its three components: SOAP, WSDL, and UDDI. SOAP (Simple Object Access Protocol) enables communication among Web services in different programming languages and across platforms. Many languages currently contain SOAP implementations that automatically generate and process SOAP messages.

WSDL (Web Services Description Language) provides a description of a Web service that is readable by a computer. Essentially, WSDL describes a Web Service's interface. WSDL is much like the English language in that it provides the rules for conversation among applications and Web services.

UDDI (Universal Description, Discovery, and Integration) is a directory, or registry of Web services descriptions. A centralized repository for Web services was developed in 2001 as an experiment by Microsoft Corporation (http://uddi.microsoft.com/). Since then, other Web Service registries have been established using the UDDI standard. The UDDI standard is built on XML and provides definitional and access information for each registered service, much like those of a phone book (Cubera *et al.*, 2002).

Together, SOAP, WSDL, and UDDI comprise Web services. A user can now search a number of UDDI-based directories for a usable service of interest. In the future, applications will be able to do the same without human intervention. Once the service is located, the user/application can then obtain the service's WSDL, which will tell the user/application how to interact with it (using SOAP). Additional reading on Web services can be found in Curbera *et al.* (2002).

Combining Web services and software agents will enable powerful DistDSS implementations for a number of classes of problems. The remainder of this paper describes some of these problems and the benefits achieved by using Web services and software agents. The following sections outline a DistDSS framework for improving the solutions to these types of problems, using the venue seating problem as an example.

3. A FRAMEWORK FOR DistDSS USING WEB SERVICES AS ENABLERS OF AGENTS

3.1 Applicability

We now introduce a DistDSS framework for any decision requiring support that can be accomplished through an intermediary, or broker. (The terms intermediary and broker are used interchangeably in this paper.) Figure 1 depicts an environment with multiple customers and multiple service (or product) providers where transactions utilize an agent (representative) that acts to some extent on behalf of the customer.

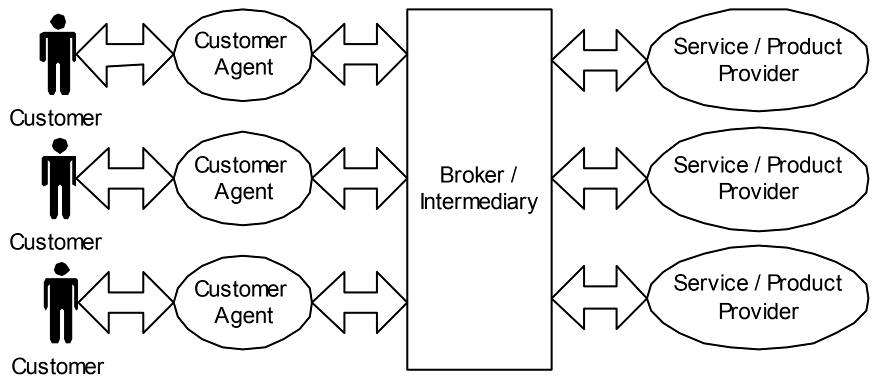


Figure 2-1
An Intermediary-Based Market

We divide brokered deal decisions into two categories: aggregate and non-aggregate. Examples of non-aggregate brokered deal decisions include purchasing a house or purchasing a car. These are one-to-one purchases, where a deal constitutes consumption of an entire product. Examples of aggregate brokered deal decisions include dividing timeshares (as in real estate or aircraft), delivery of cargo, some financial investments, and as detailed later, seating sports and entertainment venues. These are many-to-one purchases, where many customers are aggregated to buy one product or service. Each individual deal only consumes a portion of the product or service, and these products or services are differentiable somehow based on user preferences.

For each deal category there is (1) a customer (and a customer agent unless the customer represents him/herself), (2) a service (or product) provider from whom the customer is purchasing, and (3) an intermediary deal coordinator. Table 1 shows a classification of these relationships for the examples mentioned above.

The examples in Table 1 are for the purpose of adding context to the abstract description of the framework, and are not intended to be an exhaustive list of applications. For each deal type in Table 1 several different transaction models exist. For example, for a house sale transaction, one can either buy or sell by owner, or have a realtor represent them in the transaction. Neither alternative is explicitly stated in Table 1, but each scenario can be accommodated. Furthermore, given the model-driven nature of this framework, we will focus on aggregate transactions.

Deal Category	Transaction Type	Agent / Customer	Broker	Product / Service Provider
Non-aggregate	House Purchase	House Buyer / Realtor	Real Estate Broker	House Seller
	Car Purchase	Car Buyer / Salesperson	Dealer	Car Manufacturer
Aggregate	Timeshare purchase	Unit Buyer	Timeshare agent	Builder / Owner
	Sports Venue Seating (discussed in section 4)	Spectator	Ticketron	Sport Team Venue Management

Table 2-1
Aggregate / Non-Aggregate Deal Types

3.2 Basic architecture and DSS Components

Figure 2 illustrates our DistDSS framework. It contains the three components of a traditional DSS (i.e., interface, model base, and database). The interface is represented by customer agents (or more precisely, a customer agent-building Web service). The model base can be a generally usable component supplied by a third-party vendor as shown in Figure 2, or can be specific to each implementation. The service/product provider maintains the database of its products/services and customers. The DistDSS Manager serves as an intermediary and provides, at a minimum, all communication protocols, coordination among agents and components, instantiation and execution of the agent-building shell, and execution of the model base. These are all provided over the Web as Web services with each component being housed potentially on its own Web server. The agent-building shell and model base can be located on different servers at different locations, owned and provided by different entities. Data are owned by separate entities (service/product providers), located on private servers behind Web services, and distributed throughout the Web. Figure 2 shows the simplified case of many customers, one service/product provider (a.k.a. provider), and one model. However, a real system would potentially have many customers, several providers, and potentially multiple models – all geographically distributed, hence the need for a distributed system. This is ideal for aggregating transactions in deal types such as seating sports venues (See Table 1). For example, consider the problem of dealing with football season ticket requests for several teams. Many customers would request tickets, many providers (stadiums) would require seating services, and several third-party companies may provide the model base to perform the task.

Finally, the DistDSS is comprised of several distributed components that are asynchronously accessed and invoked. These components are divided into the four classes of functionality discussed above: DistDSS Manager, provider, customer, and model.

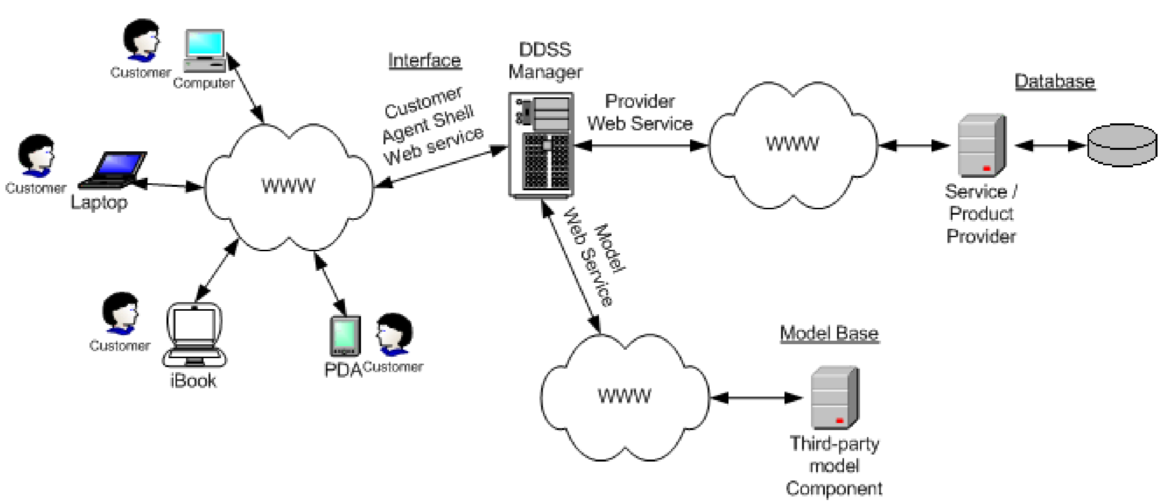


Figure 2-2
DistDSS Architecture

3.2.1 Customer

Customers are the end consumers/users of the products or services offered through the DistDSS. Customers benefit from the DistDSS by receiving better service (e.g., more personal and efficient) from providers. Customers will have agents in the system representing them, making decisions, and soliciting feedback for further negotiation and/or better future decisions. In addition, the framework suggests that all customer agents will work either cooperatively or competitively in the model base component in order to achieve their goals.

3.2.2 Service / Product Provider

The provider is the business entity that is actually providing a service or product to the (paying) customer. However, the provider uses, benefits from, and pays for the services of the DistDSS Manager to broker deals. Benefits include the potential for higher revenue and better service to the customer to its customers such as faster, more efficient, and more customized transactions. The provider's aggregating and revenue decisions are one aspect of the decisions being supported in this framework. In addition, private, provider-specific data that is used in the DistDSS is housed, maintained and owned by the provider.

3.2.3 DistDSS Manager

The DistDSS Manager, the coordinator of all activities and communication, is the heart of the DistDSS. It maintains the repository of services, including the Web services for use by the provider and the customers. It also provides a common warehouse for private customer information that will be maintained and updated for agent development. It stores encoding protocols for data and communication, executes the DistDSS model base at the appropriate time, and provides any distributed reporting services to customers and Providers.

The DistDSS Manager is developed as a Web service-enabled application. It is developed with the intent of having minimal human interaction. Since the DistDSS Manager provides coordination and notification tasks, it requires three main Web services (Figure 2). The *provider Web service* can be accessed by the providers and must be developed so that providers and the DistDSS Manager can pass each other requests, data, and solutions. *The customer Web service* must be developed to interact with customers in order to build agents that will represent them in the system as well as provide notification to them. The *model Web service* communicates with and invokes the model component of the DistDSS.

3.2.4 The Use of Web Services and Agents

Web services play a critical role in the development of agents for the system. In particular, the *customer Web service* acts as an agent shell, developing the customer agents for later use during implementation of the model portion of the system. Each customer is represented by an agent in the model portion of the system. The agent is used to act on behalf of the customer by negotiating with other customers' agents during the execution of the model. This is possible in part due to the homogeneous nature of aggregate problems. A customer begins the process of using the DistDSS by accessing a prescribed Web service that creates an agent. The Web service interacts with the customer in order to decipher his/her personal preferences. This information is used by the Web service to develop an agent that represents the customer in the system. This is of particular interest in multi-attribute aggregate problems, as customers preferences must be adequately modeled so that an intermediary can optimize factors such as aggregate satisfaction or revenue.

3.2.5 Model Base and Model Execution

The model may entail solving an optimization problem (e.g., an assignment problem) or the management of an e-market (e.g., online auction). Therefore, the model can be run either in batch or continuously. For either case, iterations of the process can continue if necessary. Execution of the DistDSS occurs in "pull" mode. The DistDSS Manager and the model base component must both be developed and "up and running" prior to any DistDSS activity. Once the DistDSS Manager service is published, the providers' private data must be transformed into a form that is consistent with what is needed for the DistDSS Manager and the model, based on the communication protocols of the Web services. With the system waiting, a request can be made by a provider to the DistDSS Manager via a call to the *provider Web service*. The DistDSS Manager will then perform the requested tasks such as preparation of customer agent interfaces and invocation of the model.

During model execution the intermediary attempts to improve the aggregate position of all customers by improving individual deals based on the personal preferences of each customer. When the model portion is completed, the agent reports back to the customer, possibly receiving additional input from the customer. This feedback allows the agent to refine its representation of the customer's preferences. This is one aspect of decision support for the customer.

Following model completion and acceptance from customers and providers, generation of customer and/or provider reports will occur. This final step in this process will most likely involve notification in the form of email or instant messages to customers and providers.

4. A VENUE SEATING DistDSS EXAMPLE

We now use an example to illustrate the potential operation and implementation of this framework. Seating season-ticket holders for entertainment venues such as sporting events is a laborious process. While software is currently available to manage customer accounts and seating assignments, there are no applications that claim to automate the seating process. This leaves experts familiar with the venue characteristics, the customer requests, and the venue management software with the task of "manually" assigning seats. The benefits to automating this process include reducing the amount of time required to perform the venue seating operation (resulting in reduced cost), faster delivery of seating information to customers, more desirable seating arrangements, and (possibly) more value/profit/income to the venue.

4.1 The current process

This scenario involves a season ticket request for seats for an entire college football season. (Note: The process described in Section 4 closely resembles the process used by the Athletic Department at the authors' institution.) Each spring the Athletic Department at a major land grant university solicits requests for season tickets for the upcoming football season. The process begins with mailings and notification via the Athletic Department website. Season ticket requestors can either mail a hard copy form of their requests or fill out an online form that is then sent electronically to the Athletic Department, and automatically transferred into the ticket management software. One's options for requesting seats at this point are very limited, specifically:

1. Returning ticket holders who wish to keep their current (previous season) seat(s) can simply check the appropriate box on the form. These seats (or seat) are then taken off the available seats list.
2. Returning ticket purchasers may request a change of seat(s). A “change of seat(s)” request does not guarantee a “better” seat, nor does it guarantee that the purchaser keep their current seat(s). A “change of seat(s)” request basically puts the purchaser in the common pool with new purchasers, without regard to their previous status.
3. New ticket purchasers are treated as returning ticket purchasers who request a change of seat(s), category 2 above.

Currently, official “special requests” are not solicited, however, purchasers are given the opportunity to write in comments that are later interpreted as “special handling.” These special handling cases range from group seating requests to specific seat requests, but can include specific yardage, distance to the field, proximity to aisles, etc.

The seat request process is closed in June and all requests are printed out in priority order (priority at this particular venue is determined by level of donation to the athletic program). The ticket seating manager then proceeds sequentially through the list, manually seating each purchaser, entering the assignment into the management software.

4.2 Problems with the current process

There are several problems with the current process:

- Because seating priority is determined by donation level, a first-time customer can request seats based on their current one-year donation level, obtain a premium seat; then reduce their donation level, but continue to renew their seats. This can result in lost revenue for the venue from year-to-year. The reason for this procedure is the amount of time consumed by the seating process. To alleviate this problem, seats could be reallocated every year, but this would be very time-consuming. Using the current procedure approximately 10-20 percent of seats are typically reallocated, all others are renewals.
- Seating and special handling is performed “manually”, and requires approximately 2 weeks to complete for a sizable football venue (60,000 seats) with only 10-20 percent reallocation.
- Seating is performed solely on a priority basis, without serious consideration to improving the satisfaction of each individual and the overall satisfaction of the entire venue.
- Once a seat is assigned, that seat is taken out of consideration and never looked at for consideration by others, even though there may be an equitable trade with another seat requestor that would increase the satisfaction of each spectator.

4.3 The DistDSS process

A potential first-time user of the system (customer or season ticket requestor) begins the process by initiating a request for a season ticket. The customer request is made through the Web, where a Web service (unbeknownst to the customer) is automatically accessed. The Web service provides a form for input of information. Since the Web service is an agent enabler as well as the DSS customer interface component, it will interact with the customer. The customer does not know, at this point, and may never know that the Web service is developing an agent that will represent him/her in the system. The customer only knows that the system is asking questions and attempting to "understand" what and how much the customer values particular attributes of the product. For sports venue seating this may include yard line, field side, row, aisle preference, or price. This interaction is a feedback system that will create an agent that is attempting to decipher the personal preferences of the customer as well as the trade-offs he/she is willing to make. Once the customer agent is specified, it will represent the customer in negotiations with the intermediary/broker during the execution of the "seating" portion of the process.

To make this example clearer and to help visualize the process, consider the following live, in-person seating system scenario. Suppose you are interested in purchasing season tickets to a sporting event. The venue owners want to spend only one day assigning seats for the entire venue. They invite all interested season ticket requestors to the actual stadium to claim their seats. First they allow current seat holders into the stadium to claim their renewed seats. Next, in priority order, potential spectators are given entry to the stadium and are told to claim the seats they desire. Once everyone is seated, the "seating manager" asks individuals if they are satisfied with their seats. If they are not, they have the option to explore the possibility of improving them by trading with others who are also interested in a change. This continues with different seat requestors until: (1) everyone is satisfied, (2) no further improvements in satisfaction is possible, or (3) the seating manager decides it is time to end the negotiations.

The seating manager's role as intermediary is important because of aggregation. Tickets for seats are not typically purchased individually, but are usually ordered in groups. The complexity of seating increases when grouping of seats is considered. For example, suppose an order consists of a group of six tickets, and it is important to this customer that the seats remain grouped together. In order for this customer to participate in the negotiation process they have two options: find another group of six to trade with; or find a group of desirable seats that consist of a few smaller groups (totaling six). The intermediary would play a critical role in coordinating this effort.

Now suppose you cannot attend the seat claiming event. Instead, you must send another person as your proxy, but first you must convey your preferences to this person. Your preferences might include some non-negotiable items such as "I need 6 seats." They might also include some negotiable desires such as "I want to be close to the 50 yard line, but I'll take the 20 yard line if I can get into the first 20 rows. I will take the end zone if nothing else is available. I will not pay more than $30 for tickets, unless I can get in the west stands on the 50 yard line, in the bottom 15 rows, in which case I will pay $60." This scenario is analogous to our system model in the sense that you have employed a proxy to act as your agent in the seating process.

Following the closing of the request period in our DistDSS system, the Provider (here the athletic department venue ticket manager who actually performs the manual seating process) sends a request to the DistDSS Manager to perform the seating service. The DistDSS Manager then makes a request to the Model base to perform the seating service. This is accomplished by calling the model-base Web service and passing it the customer and provider information. At a minimum, for the customer this includes preference information, and for the Provider this includes stadium and seat attributes.

Upon completion of the seating operation, the assignment information is sent back to the customer and feedback is requested. This is a decision support tool, and by definition, there must be user interaction with the system by the decision maker. The customer is a decision maker, but is not necessarily the primary decision maker in the system- a role typically assumed by the athletic department seating manager. In this case, the primary decision maker may or may not have an agent in the system. The seating manager decides whether the assignment is satisfactory, and based on this analysis, can manually change assignments, change and/or request a re-assignment, or accept the assignment and request that customers be notified.

5. CONCLUSIONS AND FUTURE WORK

In this paper we described a framework for an agent-based DistDSS using Web services as enablers of agents. Web services are used because of their platform independence, transparency, updatability, and their loose coupling making them highly available to customers and providers in a model-driven DistDSS. Agents are used as a means to capture and use customer preferences in negotiations with other customers' agents in an optimization-type model.

Customer preferences are mentioned briefly since preference elicitation and modeling are left for future work. Preferences will be elicited through questioning, feedback, and graphical visualization of the venue. One would expect that most spectators would want the most desirable seat at the venue, but elicitation of preferences goes deeper than that. As mentioned, seating is accomplished on a priority basis, and the agents need to discover what customers value in a seat and what trade-offs they are willing to make, as well as what price they are willing to pay.

Solving the problem of seating a venue using agent negotiations requires a system that allows customer agents to be developed and transported into the model base where they can negotiate with other agents. Via negotiation, a solution can evolve to a point where individual, and resulting overall venue, satisfaction is improved. This is also left for future research.

Finally, the popular press has recently publicized that a number of professional sports teams are adopting variable-price models for their venues. Prices are being determined not only by the desirability and location of the seat, but are also based on the quality of the opposing team and the celebrity of the opposing team's players. This framework can be extended with the establishment of an e-market for professional sports venues, including an initial offering auction house and a secondary market for resale of individual games.

REFERENCES

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web. *Scientific American,* 284 (5), 34-43.

Bhargava, H. & Power, D. (2001). Decision support systems and web technologies: a status report. *Seventh Americas Conference on Information Systems (AMCIS '01)*. Association for Information Systems. Boston, MA.

Bui, T. & Lee, J. (1999). An agent-based framework for building decision support systems. *Decision Support Systems*, 25(3), 225-237.

Buss, A. (2000). Component-based simulation modeling. *Winter Simulation Conference (WSC '00)*. IEEE. Orlando, FL.

Buss, A. & Stork, K. (1996). Discrete event simulation on the world wide web using java. *Winter Simulation Conference (WSC '96)*. IEEE. Piscataway, NJ.

Cohen, M., Kelly, C., & Medaglia, A. (2001). Decision support with web-enabled software. *Interfaces,* 31(2), 109-129.

Cubert, R. & Fishwick, P. (1997). A framework for distributed object-oriented multimodeling and simulation, *1997 Winter Simulation Conference (WSC '97)*. IEEE. Atlanta, GA.

Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., & Weerawarana, S. (2002). Unraveling the web services web: an introduction to SOAP, WSDL, and UDDI. *IEEE Internet Computing*, 6(2), 86-93.

Fatsis, S., (2002). The Barry Bonds tax: teams raise prices for good games, *The Wall Street Journal*. New York, 3 December, D1.

MOOSE Homepage. (n.d.). Retrieved April 21, 2002, from http://www.cise.ufl.edu/~fishwick/moose.html

Glass, G. (2002). *Web services: building blocks for distributed systems*. Upper Saddle River: Prentice Hall.

Hess, T. (1999). A study of autonomous agents in decision support systems. *PhD. Dissertation*. Department of Management Science and Information Technology, Virginia Polytechnic Institute and State University, Blacksburg, VA, 135.

Hess, T., Rees, L., & Rakes, T. (2000). Using autonomous software agents to create the next generation of decision support systems. *Decision Sciences,* 31(1), 1-31.

Huhns, M. (2002). Agents as web services. *IEEE Internet* Computing, 6(4), 93-95.

Kilgore, R., Healy, K., & Kleindorfer, G. (1998). The future of java-based simulation. *1998 Winter Simulation Conference (WSC '98)*. IEEE. Washington, D.C.

Kilgore, R. (2002). Simulation web services with .NET technologies. *2002 Winter Simulation Conference (WSC '02)*. IEEE. San Diego, CA.

Lang, K. & Whinston, A. (1999). A design of DSS intermediary for electronic markets. *Decision Support* Systems, 25(3), 181-197.

Luo, Y., Liu, K., & Davis, D. (2002). A multi-agent decision support system for stock trading. *IEEE Networking*, 16(1), 20-27.

Nghiem, A. (2003). *IT web services: a roadmap for the enterprise*. Upper Saddle River: Prentice Hall.

Oliver, J. (1996) A machine-learning approach to automated negotiation and prospects for electronic commerce. *Journal of Management Information Systems*, 13(3), 83-112.

Page, E. & Opper, J. (2001). Investigating the application of web-based simulation principles within the architecture for a next-generation computer generated forces model. *Future Generation Computer Systems*, 17, 159-169.

Pinson, S., Louca, J., & Moraitis, P. (1997). A distributed decision support system for strategic planning. *Decision Support Systems*, 20(1), 35-51.

Sprague, R. & Carlson, E. (1982). *Building effective decision support systems*. Englewood Cliffs, NJ: Prentice-Hall.

Tecuci, G. (1998). *Building intelligent agents*. Academic Press.

Tewari, G., Youll, J., & Maes, P. (2002). Personalized location-based brokering using an agent-based intermediary architecture. *Decision Support Systems*, 34(2), 127-137.

Vaughan-Nichols, S. (2002). Web services: beyond the hype. *IEEE Computer*, 35(2), 18-21.

Whinston, A. (1997). Intelligent agents as a basis for decision support systems. *Decision Support Systems*, 20(1), 1.

Chapter 3

Modeling user preferences using conjoint analysis and neural networks for e-commerce

1. INTRODUCTION

The ever expanding number of merchants and products available on the World Wide Web (Web) continues to increase the effort, time, and uncertainty involved with business-to-consumer (B2C) online purchasing. As B2C e-commerce matures, the need for better product searching and purchasing tools increases. Many websites currently implement user-driven sorting and filtering to aid consumers with the difficult task of finding the right product for their needs. A recent novel attempt to improve this task is Yahoo Shopping's Smartsort (Yahoo Smartsort), a tool for self-explicating preferences while shopping online. Using Smartsort, a consumer can specify levels of importance for different attributes of a product, enabling the search engine to return more relevant results. In essence, Smartsort develops a searching agent. Many researchers look toward software agents to perform these searching, negotiation, and purchasing tasks in the future. Moreover, some envision the future of the Web as a giant distributed virtual assistant, taking care of daily tasks, setting appointments, and negotiating on one's behalf (Berners-Lee, Hendler, & Lassila, 2001). Self-explicating preferences may be a next-generation approach to enabling more intelligent agent-based product searching. However, it has been shown that self-explication does not necessarily outperform another technique, conjoint analysis (CA), for eliciting and modeling customer preferences (Aggarwal & Vaidyanathan, 2003).

Much research has been pursued on using agents in a distributed environment, yet little has been done to realistically elicit and model the preferences of consumers so that an online agent can truly act on a consumer's behalf. Traditionally, agent developers acquire knowledge from domain experts, then program the agent to make autonomous and/or intelligent decisions. This process requires significant time and effort on the part of the user, developer, and domain experts, and is impractical to perform individually for every consumer in the fast-paced global environment of the Web. It is evident that a fast, effective, and easy method for eliciting and modeling individual consumer preferences is needed for future e-commerce markets.

Along with the recent increase in online commercial activity, there has been a corresponding increase in the amount of research addressing online data-driven decision problems (Fan, Gordon, & Pathak, 2003). Web-based decision support researchers have focused primarily on addressing these data-driven problems, and have largely ignored model-driven decision support problems (Bhargava & Power, 2001). However, a class of model-driven problems exists that would benefit from improved online systems; specifically, the ability to quickly elicit and model individual customer preferences for product attributes.

Virtually all B2C online transactions involve customers searching for products that match their personal preference criteria, choosing among a number of potential products, and making a purchase that maximizes some measure of their personal satisfaction. However, some types of transactions are more complicated in that for a given event, or time period, there is a limited supply of the product. Online purchasing of season tickets for sporting events, vacation purchasing (especially cruises), and timeshare trading are such examples. Consumers interested in purchasing season tickets to their favorite sports venue must consider not only that there is limited seating for the venue, but also that there are often tens of thousands of other consumers attempting to purchase tickets for the same venue with limited seating capacity. The most effective way to implement an online system to improve this process is to be able to elicit and model each individual's preferences for different seats, using such factors as distance to the field (or court), location within the facility (yard line, direction), and propensity to pay. By modeling the preferences of each customer, satisfaction can be explicitly accounted for in a model-based decision support system (DSS) that creates seating plans for the entire venue.

In this study, we construct and test the accuracy of preference models for use in a model-driven Web-based DSS. CA, a technique primarily used in marketing to elicit and model target market preferences, was used to elicit the preferences of potential customers. However, traditional CA alone did not provide suitable preference models to adequately predict individual customer preferences. Therefore, we developed a methodology for modeling customer preferences using neural networks, and demonstrated the superiority of this new approach versus a traditional CA method in the context of spring break vacation package purchasing. We also consider how this technique can be used in future work to develop a Web-based model-driven DSS.

Section 2 of this paper provides background on software agents, CA, and neural networks. Section 3 outlines our new methodology for constructing user preference models using neural networks. Section 4 presents our analysis and results, and in Section 5 we discuss our conclusions and future work.

2. BACKGROUND

2.1 Software Agents and Preference Modeling

Software agents are viewed by many researchers as enablers of the next generation of e-commerce and Web-based decision support. West *et al.* (1999) describe several roles agents can play in electronic markets. Of particular interest related to this work is the role of tutor. In this role, an agent discovers user preferences and then educates the user about options and features in a particular domain. This is similar to a realtor interacting with a client in order to learn their price range and importance of certain home features, then suggesting neighborhoods and finding homes that meet the client's criteria. Another example of an agent-tutor is Amazon's (Amazon Homepage) recommendation system. It uses other customers' historical purchases to recommend additional items for one's consideration. When a customer shopping on Amazon.com adds a product to their virtual shopping cart, the system identifies complementary products for consideration. These complementary products are presented as products purchased by other customers who also purchased the product currently in the shopper's cart.

Lee, Liu, & Lu (2002) develop a multi-agent-based recommender system for DVD films in an effort to reduce the number of items a consumer must search when making decisions. They employ a genetic algorithm whose chromosome contains numeric values representing movie keywords and their associated relative importance to the customer (determined from feedback such as browsing history and past purchases and rentals). The algorithm reportedly fared well in experiments, predicting with approximately 70-77% accuracy, and outperforming a k-Nearest Neighbor method. The authors of the study identify two major shortcoming of their system: lack of robustness of the models with sparse training data, and difficulty capturing product knowledge from domain experts.

Gershoff & West (1998) incorporate others' opinions in a multiple regression to predict individual preferences, extending a traditional CA approach and thus creating an improved preference model. For further reading on software agents in DSS and e-commerce refer to (Hess, Rees, & Rakes, 2000; Guttman, Moukas, & Maes, 1998; Weiss, 1999).

2.2 Conjoint Analysis

CA is the method most used by marketing researchers and practitioners for analyzing consumer group trade-offs and eliciting and modeling product attribute preferences (Green, Krieger, & Wind, 2001). CA grew out of the work of Luce and Tukey (1964) and was made popular a decade later by Green and Wind (1975). Although CA has been used extensively in the marketing community during the past thirty years, the operations research and management science (OR/MS) research community is just beginning to embrace its power (Green, Krieger, & Wind, 2001). CA has recently been applied to such problems as multi-period sales promotion design (Nair & Tarasewich, 2003) and hospital advertising (Tscheulin & Helmig, 1998). CA uses a variety of techniques to elicit and then express customer utility functions for different product designs.

In short, CA presents individuals in target consumer groups different (but similar) product descriptions / profiles with various attributes and asks them to either rank or rate the items. CA then uses quantitative techniques to estimate the structure of the target group's preferences (Green & Srinivasan, 1978). Desirable product attributes can be determined from this information. CA has also been used to estimate individual preferences (West *et. al.*, 1999). While CA can be very effective, it also has its limitations. For instance, it has been shown that an individual can only accurately evaluate up to 30 product profiles before becoming overwhelmed with data, potentially confounding the analysis (Green & Srinivasan, 1978). This restriction on data collection (sample size) limits the tools one can use to construct individual consumers' preference models.

2.3 Neural Networks and Conjoint Analysis

Artificial Neural Networks, or neural networks (NNs), are biologically-inspired computer-based systems that are modeled after the brain and nervous system (Burke, 1991). The strength and allure of neural networks is that they rely on pattern recognition for determining a response to input stimuli rather than domain specific logic and rules required by expert systems. Thus, NNs are an especially attractive alternative for modeling consumer choice as consumers often do not know or cannot communicate why they prefer one product over another, therefore making it difficult to induce rules or account for confounding variables. West *et. al.* (1997) show that when attempting to predict consumer group choice decisions, NNs can often outperform statistical methods. To date, NNs have not been used successfully on individual consumers in conjunction with CA because NNs require large data sets for training purposes. While CA produces large data sets for aggregate studies, it only produces small data sets (25-30 records) for each individual within the study. Therefore, in order to apply neural networks to the construction of an individual's preference model, we must develop a methodology for increasing the number of data records available from the data collected via CA.

3. METHODOLOGY

3.1 Overview

In this study we collected data using a CA technique, constructed several preference models for each individual participant, and tested the models for prediction accuracy. This section details the data collection and experiment methodology.

3.2 Data Collection

Participants in this study completed a CA instrument designed to assess individual preferences for spring break vacation packages. This topic was chosen for its familiarity and relevance to its participants - fall semester senior level undergraduate students. We collected our data using a *full-profile* CA. Several CA data collection methods exist such as adaptive, choice-based, partial-profile, and full-profile (Orme, 2003). A full-profile CA was chosen because of its straightforward manner and the size of the experiment (fewer than 6 attributes per package). It has been shown that the full-profile method is appropriate for experiments of this size (Green & Srinivasan 1978; Orme, 2003), while other methods are better suited for experiments with many more attributes. Using the full-profile method, participants are shown “cards” that describe complete vacation packages (showing all attributes), and are asked to rate each card (as opposed to other CA methods that attempt to dissect each attribute and individually and collectively look at different levels of each attribute).

Attributes and levels were determined in advance of our experiment by polling students and through investigation of popular Web-based spring break vacation advertising. Attributes and their levels are detailed in Table 3-1.

Attributes	Attribute Levels
Location	Mexico
	Florida- Atlantic Coast
	USA Gulf Coast (Not Florida)
	Florida - Gulf coast
	Southern California Coast
Hotel Rating	1 star
	2 star
	3 star
	4 star
	5 star
Transportation	None
	Bus
	Train
	Car
	Fly
Inclusion	None
	Bronze
	Silver
	Gold
	Platinum
Price	$500
	$1,000
	$1,500
	$2,000
	$2,500
Inclusion	
Platinum: Transportation, Activities, Food, Drinks, Tips & Cover charges	
Gold: Transportation, Activities, Food	
Silver: Transportation, Activities	
Bronze: Transportation	
None: Nothing included	

Table 3-1
Attributes and Attribute Levels

Therefore, a card might be represented as follows:

Location: Mexico
Hotel Rating: 3 star
Transportation to destination: Fly
Inclusion: None (inclusion was detailed in the CA)
Price: $2,500

Mathematically, a card is defined as $c_i = (a_{1_i}, a_{2_i}, \ldots, a_{n_i})$ where a_{k_i} represents the level of attribute a_k contained in card c_i.

Each participant was provided a paper survey containing the same unique 25 spring break vacation packages (cards) and asked to perform two tasks. First, each participant was asked to assign the most preferable package a rating of 100, and the least preferable a rating of 0. Next, they were instructed to assign a unique rating from 0 to100 for each of the remaining packages. Therefore, for each card, c_i, each participant provided a unique individual rating r_i such that: $r_i \neq r_j \ \forall i, j \in [0,100]$.

It has been shown that the number of stimuli should be limited to no more than thirty cards for full-profile CA data collection (Green & Srinivasan 1978; Orme, 2003). As a result, we employed a 5 x 5 orthogonal fractional factorial design requiring that 25 stimuli be presented to each participant. Surveys were collected from ninety-nine participants. Of the original ninety-nine, seven improperly completed the study, therefore leaving ninety-two completed questionnaires. An incentive of $100 was provided to the participant whose preferences were most accurately modeled.

4. ANALYSIS AND RESULTS

4.1 Overview

Once collected, the data were used to build six different preference models for each participant; three using regression and three using a neural network. We refer to these six models using the names listed in Table 3-2.

Regression Models	Neural Network Models
REGR23	NN23
REGR253	NN253
REGR506	NN506

Table 3-2
Preference Model Names

The letter portion of the model name designates whether regression (REGR) or a neural network (NN) was used to construct the model. The number portion of the model name represents the number of records used to construct the model. The regression models (detailed in section 4.3) were constructed using Microsoft Excel's Solver to calculate the least squares parameter estimates for each model. The NNs were constructed using Neuralworks Predict version 3.12 by NeuralWare. Default settings were used for NN model construction and selection of the training, validation, and testing sets.

4.2 Evaluation

For each participant, 300 models were constructed for each model type in Table 3-2. We constructed these models using a pairwise "hold out" procedure in order to assess the effectiveness of each modeling technique in an unbiased manner. For each model, two unique cards, c_i and c_j, where $i \neq j$, were removed from the original data set prior to model construction. The model was then constructed using the remaining 23 cards. Recall that the original data contained 25 records per participant. Repeating this for every unique combination of i and j resulted in $\binom{25}{2} = 300$ unique combinations of hold out cards, and subsequently 300 models and tests.

Each model was then used to predict a relative preference ordering between hold out cards c_i and c_j. In general, a given participant prefers c_i over c_j if $r_i > r_j$, and prefers c_j over c_i if $r_j > r_i$. The model's prediction, $\hat{r}_i$ and $\hat{r}_j$, was then compared to the participant's actual ratings of r_i and r_j to determine accuracy. Predicting relative preference in this way is consistent with the philosophy of ordinal optimization. Fu (2002) states that:

> "It is generally easier to compare solutions and find relative ordering among them rather than it is to estimate them precisely."

That is, by predicting $\hat{r}_i$ and $\hat{r}_j$ individually and then comparing values, a relative preference ordering between cards c_i and c_j can be determined and compared to the participant's specified preference indicated by r_i vs. r_j.

4.3 Models

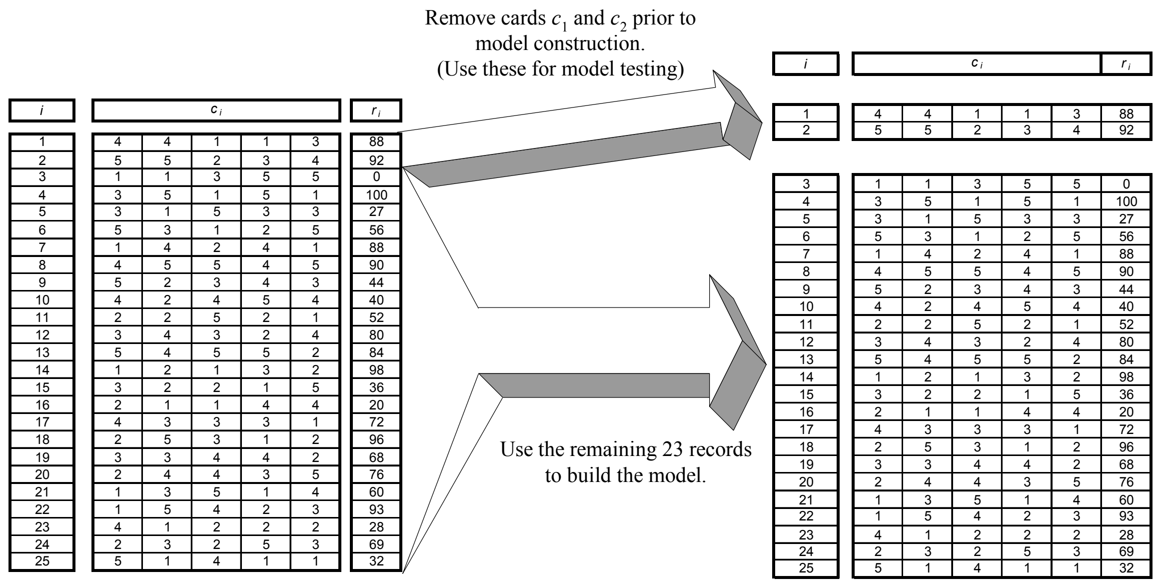
REGR23 vs. NN 23

The REGR23 and NN23 models were initially intended to provide a prediction, $\hat{r}_i$, of the actual precise rating value, r_i, for each "hold out" card for each participant. Preliminary results showed that the models performed poorly for predicting ratings, but did a fair job at ordering, or determining the relative preferences of pair-wise comparisons.

For the REGR23 and NN23 models, each record used for model construction is in the form:

$$[\ c_i, r_i\ ]$$

where i is the original data record card number ($1 \leq i \leq 25$), c_i represents the card as described in section 2.2, and r_i is the unique numeric rating provided by each participant. Figure 3-1 shows the model and hold out procedure for cards c_1 and c_2.



i	c_i					r_i
1	4	4	1	1	3	88
2	5	5	2	3	4	92
3	1	1	3	5	5	0
4	3	5	1	5	1	100
5	3	1	5	3	3	27
6	5	3	1	2	5	56
7	1	4	2	4	1	88
8	4	5	5	4	5	90
9	5	2	3	4	3	44
10	4	2	4	5	4	40
11	2	2	5	2	1	52
12	3	4	3	2	4	80
13	5	4	5	5	2	84
14	1	2	1	3	2	98
15	3	2	2	1	5	36
16	2	1	1	4	4	20
17	4	3	3	3	1	72
18	2	5	3	1	2	96
19	3	3	4	4	2	68
20	2	4	4	3	5	76
21	1	3	5	1	4	60
22	1	5	4	2	3	93
23	4	1	2	2	2	28
24	2	3	2	5	3	69
25	5	1	4	1	1	32

i	c_i					r_i
1	4	4	1	1	3	88
2	5	5	2	3	4	92

3	1	1	3	5	5	0
4	3	5	1	5	1	100
5	3	1	5	3	3	27
6	5	3	1	2	5	56
7	1	4	2	4	1	88
8	4	5	5	4	5	90
9	5	2	3	4	3	44
10	4	2	4	5	4	40
11	2	2	5	2	1	52
12	3	4	3	2	4	80
13	5	4	5	5	2	84
14	1	2	1	3	2	98
15	3	2	2	1	5	36
16	2	1	1	4	4	20
17	4	3	3	3	1	72
18	2	5	3	1	2	96
19	3	3	4	4	2	68
20	2	4	4	3	5	76
21	1	3	5	1	4	60
22	1	5	4	2	3	93
23	4	1	2	2	2	28
24	2	3	2	5	3	69
25	5	1	4	1	1	32

Figure 3-1
REGR / NN 23 Model Construction

Figure 3-2 provides further clarification of the entire hold out procedure.

Model Number	Hold Out	Construct Model Using
1	1 & 2	3-25
2	1 & 3	2, 4-25
3	1 & 4	2-3, 5-25
4	1 & 5	2-4, 6-25
…	…	…
300	24 & 25	1-23

Figure 3-2
Hold Out Procedure

Using the predicted numeric ratings $\hat{r}_i$ and $\hat{r}_j$ for cards c_i and c_j, the preference determination rules were as follows:

If $\hat{r}_i > \hat{r}_j$ and $r_i > r_j$, then the model accurately predicted the preference.

If $\hat{r}_i \leq \hat{r}_j$ and $r_i < r_j$, then the model accurately predicted the preference.

For each subject, we computed the percentage of trials in which each of the 300 models made an accurate prediction. Table 3-3 summarizes the overall prediction accuracy for these two models for all 92 subjects.

	REGR23	NN23
n	92	92
Mean	0.6513	0.7111
Minimum	0.2100	0.4200
Maximum	0.8833	0.9867
Standard Error	0.0115	0.0118
Standard Deviation	0.1099	0.1128

Table 3-3
REGR23 and NN23 Results

The mean, minimum, and maximum values reported in Table 3-3 represent the percentage of times the models correctly predicted the participants' preference between cards c_i and c_j. For the 92 observations, the NN models performed better than the regression models in general. A standard t-test for two-sample means indicated the difference in means is significant ($p \ll 0.0001$).

REGR253 vs. NN253

Although the NN23 models outperformed the REGR23 models above, recall, that only 23 records were used to train the NN23 model. Because NNs typically perform better when trained with larger data sets we developed a method for modeling preference orders that modifies the original data and increases the number of training records. In REGR253 and NN253, the original data was expanded by creating a single record for every pairwise comparison in the model-building card set. As in REGR23 and NN23, the "hold out" procedure was used such that 300 different models were constructed per participant for every card pair c_i and c_j, where $i \neq j$ such that c_i and c_j were excluded from construction of the model.

Therefore, each record used in the model took the form:

$$[\ c_i, c_j, y_{ij}\ ]$$

where i and j are the original data record card numbers ($1 \leq i \leq 24$, $2 \leq j \leq 25$), and $i < j$ for every record, c_i and c_j represent the respective cards, and y_{ij} is a binary value representing the participant's preference between c_i and c_j as follows:

If $r_i > r_j$, participant prefers c_i, $y_{ij} = 0$

If $r_i < r_j$, participant prefers c_j, $y_{ij} = 1$

Note that $r_i \neq r_j$, since participants were instructed to provide unique ratings for each card. The resulting models each contained 253 records each as depicted in Figure 3-3.

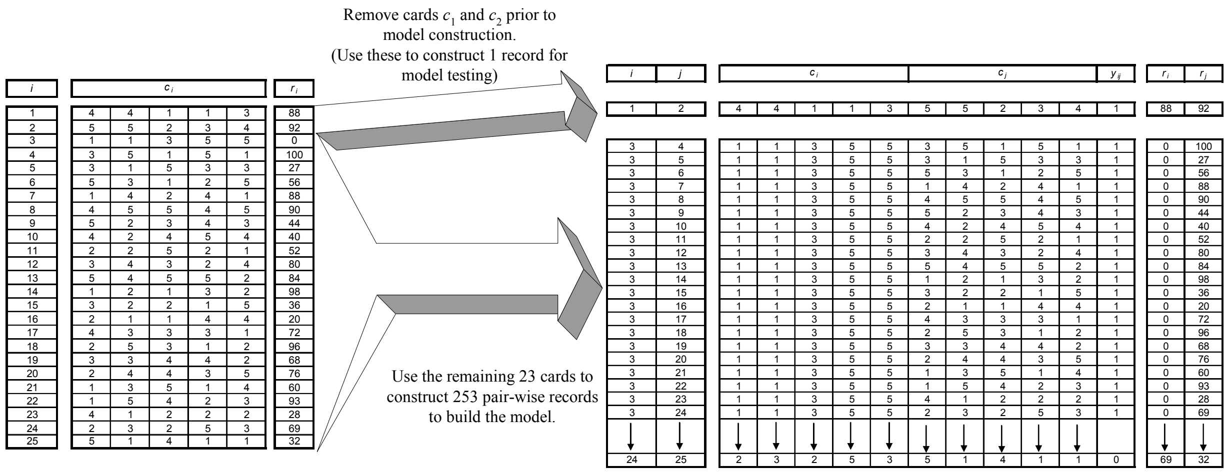


Figure 3-3
REGR / NN 253 Model Construction

The model output, $\hat{y}'_{ij}$, took on the form of a real number from 0 to 1. The value of 0.5 was used as the cutoff such that:

If $\hat{y}'_{ij} < 0.5$, c_i is predicted as preferred over c_j, and $\hat{y}_{ij} = 0$

If $\hat{y}'_{ij} \geq 0.5$, c_j is predicted as preferred over c_i, and $\hat{y}_{ij} = 1$.

Once again, 300 unique models were constructed using the hold out procedure. For each model, $\hat{y}_{ij}$ and y_{ij} were calculated for the comparison between the held out cards c_i and c_j. Table 3-4 summarizes the prediction accuracy observed in this testing.

	REGR253	NN253
n	92	92
Mean	0.5703	0.7344
Minimum	0.3833	0.4500
Maximum	0.7100	0.9700
Standard Error	0.0071	0.0119
Standard Deviation	0.0678	0.1137

Table 3-4
REGR253 and NN253 Results

The mean, minimum, and maximum values reported in Table 3-4 represent the percentage of times the models correctly predicted the participants' preference between cards c_i and c_j. For the 92 observations, the NN models again outperformed the regression models. A standard t-test for two-sample means indicated a significant difference in the means ($p << 0.0001$).

REGR506 vs. NN506

The REGR253 and NN253 showed promising results for our methodology. However, in these models, the $i < j$ constraint is bothersome inasmuch as products in real-world pairwise comparisons are not ordered, or in some sense, are randomly ordered. Therefore, we altered the REGR253 and NN253 methodologies to be consistent with real-world comparisons by removing the $i < j$ constraint, thereby removing any effects associated with the presentation order of the pairwise comparisons. The resulting models are REGR506 and NN506. They are similar to the REGR253 and NN253 models in that each record represents two cards, c_i and c_j, and the preference between them, y_{ij}. However, by relaxing the $i < j$ constraint for each record, the number of records used to construct the models is doubled. Therefore, each record from the REGR253 and NN253 models are exactly reproduced in these models as:

$$[\ c_i,\ c_j\ ,y_{ij}\ ]$$

For each of these records a complementary record is added having the form:

$$[\ c_j, c_i, y_{ji}\ ].$$

This procedure is illustrated graphically in Figure 4.

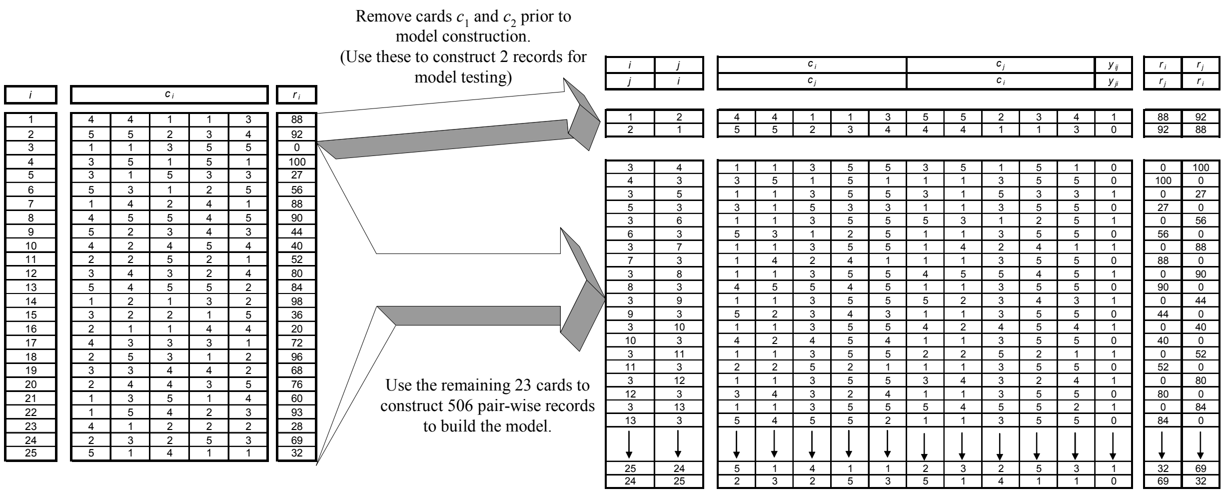


Figure 3-4
REGR / NN 506 Model Construction

The evaluation of the model output was more complex for this methodology as compared to the previous methodologies in that for each pairwise comparison, two calculations were actually performed. For each card pair, c_i and c_j, model outputs $\hat{y}'_{ij}$ and $\hat{y}'_{ji}$, and predictions $\hat{y}_{ij}$, and $\hat{y}_{ji}$ were calculated as follows:

If $\hat{y}'_{ij}$ < 0.5 and $\hat{y}'_{ji}$ ≥ 0.5, c_i is predicted as preferred over c_j, and $\hat{y}_{ij}$ = 0 and $\hat{y}_{ji}$ = 1

If $\hat{y}'_{ij}$ ≥ 0.5 and $\hat{y}'_{ji}$ < 0.5, c_j is predicted as preferred over c_i, and $\hat{y}_{ji}$ = 0 and $\hat{y}_{ij}$ = 1 for the cases where $\hat{y}'_{ij}$ < 0.5 and $\hat{y}'_{ji}$ < 0.5 or $\hat{y}'_{ij}$ ≥ 0.5 and $\hat{y}'_{ji}$ ≥ 0.5, the absolute difference $\hat{y}'_{ij}$ -0.5 and $\hat{y}'_{ji}$ -0.5 were calculated, and the greater difference was given priority such that:

If $\hat{y}'_{ij}$ -0.5 > $\hat{y}'_{ji}$ -0.5, c_i is predicted as preferred over c_j, and $\hat{y}_{ij}$ = 0 and $\hat{y}_{ji}$ = 1

If $\hat{y}'_{ij}$ -0.5 < $\hat{y}'_{ji}$ -0.5, c_j is predicted as preferred over c_i, and $\hat{y}_{ji}$ = 0 and $\hat{y}_{ij}$ = 1

This was done as a consistency check since the evaluation of two cards: [c_i, c_j, y_{ij}] and [c_j, c_i, y_{ji}] could potentially yield inconsistent predictions such that card c_i could be predicted as preferred in one evaluation and c_j could be predicted as preferred in the second.

Again, three hundred unique models were constructed for this method using the same hold out procedure described earlier. For each model, $\hat{y}_{ij}$, $\hat{y}_{ji}$, y_{ij}, and y_{ji} were calculated for the "held out" cards, c_i and c_j. Table 3 reports our prediction accuracy.

	REGR506	NN506
n	92	92
Mean	0.7171	0.7400
Minimum	0.2433	0.4100
Maximum	0.9767	0.9933
Standard Error	0.0126	0.0124
Standard Deviation	0.1206	0.1190

Table 3-5
REGR506 and NN506 Results

The mean, minimum, and maximum values reported in Table 3-5 represent the percentage of times the models correctly predicted the participants' preference between cards c_i and c_j. For the 92 observations, the NN models again outperformed the regression models. A standard t-test for two-sample means indicated a significant difference in the means (p << 0.0001).

	REGR23	REGR253	REGR506	NN23	NN253	NN506
n	92	92	92	92	92	92
Mean	0.6513	0.5703	0.7171	0.7111	0.7344	0.7400
Minimum	0.2100	0.3833	0.2433	0.4200	0.4500	0.4100
Maximum	0.8833	0.7100	0.9767	0.9867	0.9700	0.9933
Standard Error	0.0115	0.0071	0.0126	0.0118	0.0119	0.0124
Standard Deviation	0.1099	0.0678	0.1206	0.1128	0.1137	0.1190

Table 3-6
All Results

Table 3-6 provides a statistical summary of results for the entire study. Reported are the percent of times each model correctly predicted the preferences of each individual. In general, the neural network models outperform the regression models, and NN506 outperforms all other methodologies. Recall that REGR23 represents the standard CA methodology. When comparing the results of the NN506 and REGR23 techniques, average prediction accuracy is increased by 13.6% by using NN506 vs. REGR23 without a sacrifice in standard error. The most accurate NN506 model provided correct predictions 99.33% (298 correct out of 300) of the time versus 88.33% for REGR23, a 12.5% improvement. Additionally, the least accurate prediction increased from 21% with REGR23 to 41% with NN506. Figure 3-5 graphically shows the distribution of the accuracy for both methods.

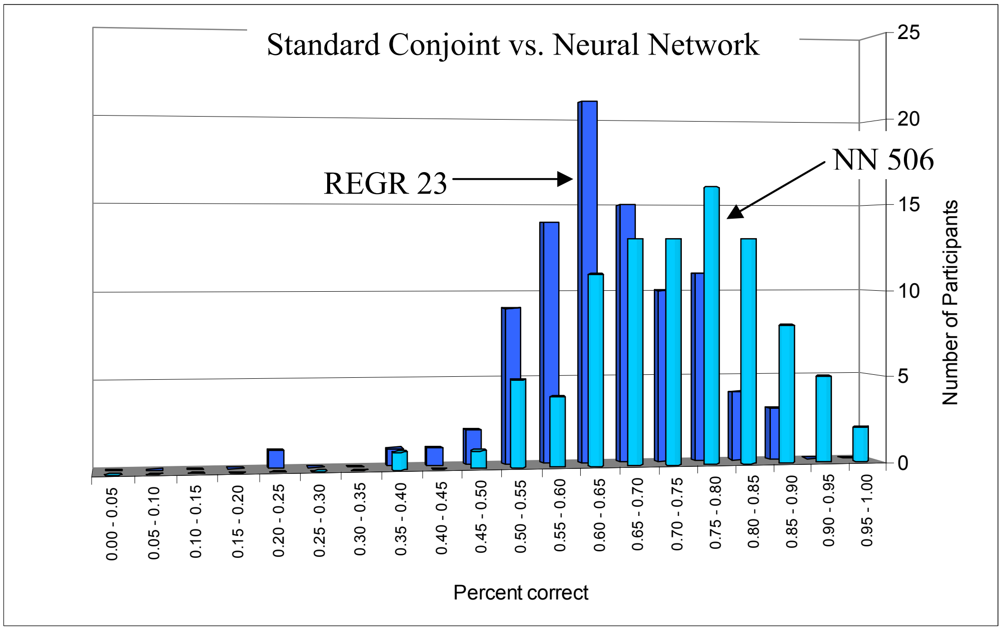


Figure 3-5
Distribution of REGR23 and NN506 Model Results

5. CONCLUSIONS AND FUTURE WORK

Efficiently and effectively obtaining and modeling consumer preferences is a major roadblock in enabling agent-based negotiation on the Web. This paper suggests a promising approach to removing this roadblock and allowing the information superhighway to realize its full potential for B2C e-commerce. In this paper we examine the problem of modeling consumer preferences for agent-based B2C online purchase transactions. We introduce a class of problems that would benefit greatly from the addition of a system that could easily and accurately elicit and model individual customer preferences. Addressing solutions to these problems, we propose CA as the elicitation tool and conduct a study using 92 participants. While conducting a standard CA regression analysis of the data, we propose using a neural network as an alternative method for modeling individual customer preferences and propose a number of ways to expand the original CA data set for use in a NN.

In comparing the proposed methodologies, the NN506 method proved to be best. This methodology resulted in a significant improvement over its regression counterpart, REGR506 while providing a more robust methodology than NN23 and NN253. NN23 as a general methodology is subject to potential criticism due to the few number of data records used to train the NN. NN253 falls short in that it requires that pairwise comparisons must be ordered for both training and evaluation data. This is an unrealistic requirement, as order does not matter in true pairwise comparisons. NN506 overcomes both these shortcomings. Using the NN506 method, we take an original data set of 25 records and expand the data to 506 records of unique pairwise comparison combinations of the original data. In addition to outperforming a standard CA methodology our method also overcomes problems in currently proposed systems such as the requirement of domain experts to build a knowledge base, and the previously mentioned problem of sample size.

The positive results of our work present several opportunities to address additional questions as future work. One such question deals with the ability to instantiate an agent such that it will use the models developed here to act on behalf of a customer by either searching or negotiating for products. Another asks if we can improve model prediction accuracy by incorporating additional tools into our methodology such as using AHP for evaluating the consistency of the NN predictions. Finally, the ability to incorporate this methodology into a current or new innovative distributed system architecture must be explored.

REFERENCES

Aggarwal, P. & Vaidyanathan, R. (2003). Eliciting online customers' preferences: conjoint vs. self-explicated attribute-level measurements. *Journal of Marketing Management*, 19, 157-177.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web. *Scientific American,* 284 (5), 34-43.

Bhargava, H. & Power, D. (2001). Decision support systems and web technologies: a status report. *Seventh Americas Conference on Information Systems (AMCIS '01)*. Association for Information Systems. Boston, MA.

Burke, L. (1991). Introduction to artificial neural systems for pattern regognition. *Computers and Operations* Research 18(2), 211-220.

Fan, W., Gordon, M., & Pathak, P. (2003). A generic ranking function discovery framework by genetic programming for information retrieval. *Information Processing & Management*. In press.

Fu, M. (2002). Optimization for simulation: theory vs. practice. *INFORMS Journal on Computing*. 14(3), 192-215.

Gershoff, A. & West, P. (1998). Using a community of knowledge to build intelligent agents. Marketing Letters, 9(1), 79-91.

Green, P., Krieger, A., & Wind, Y. (2001). Thirty years of conjoint analysis: reflections and prospects. *Interfaces*, 31(3), Part 2 of 2, S56-S73.

Green, P. & Srinivasan,V. (1978). Conjoint analysis in consumer research: issues and outlook, *The Journal of Consumer Research*, 5(2), 103-123.

Green, P. & Wind, Y. (1975). New ways to measure consumers' judgements. *Harvard Business Review*, 53(4), 107-117.

Guttman, R., Moukas, G, & Maes, P. (1998). Agent-mediated electronic commerce: a survey. *The Knowledge Engineering Review*, 13(2), 143-152.

Hess, T., Rees, L., & Rakes, T. (2000). Using autonomous software agents to create the next generation of decision support systems. *Decision Sciences,* 31(1), 1-31.

Lee W.-P., Liu C.-H., Lu C.-C., 2002. Intelligent Agent-based Systems for Personalized Recommendations in Internet Commerce. *Expert Systems with Applications* 22(4): 275-284.

Luce, D. & Tukey, J. (1964). Simultaneous conjoint measurement: a new type of fundamental measurement, *Journal of Mathematical Psychology*, 1, 1-27.

Nair, S. & Tarasewich, P. (2003). A model and solution method for multi-period sales promotion design. *European Journal of Operational Research*, 150, 672-687.

Orme, B. (2003). Which conjoint method should I use? *Sawtooth Software Research Paper Series*.

Tscheulin, D. & Helmig, B. (1998). The optimal design of hospital advertising by means of conjoint measurement. *Journal of Advertising Research*, (38)3, 35-46.

Weiss, G. (1999). *Multiagent systems: a modern approach to distributed artificial intelligence*, Cambridge, MA: The MIT Press.

West, P., Ariely, D., Bellman, S., Bradlow, E., Huber, J., Johnson, E., *et. al*. (1999). Agents to the rescue? *Marketing Letters* 10(3), 285-300.

West, P., Brocket, P., & Golden, L. (1997). A comparative analysis of neural networks and statistical methods for predicting consumer choice, *Marketing Science*, 6(4), 370-391.

Yahooo Smartsort. (n.d.). Retrieved February 1, 2004, from http://shopping.yahoo.com/smartsort/

Chapter 4

An Agent-Based Mediated Market for Online Purchases

1. INTRODUCTION

Successful online retail businesses accept and embrace the business paradigm shift from a supply-driven to demand-driven process where consumers expect and demand customization of the purchasing experience. This is evident from the recent academic research into and retail adoption of several new personalized online services such as recommender systems, self-explicating product search, and custom information retrieval (Amazon Homepage; Lee, Liu, & Lu, 2002; Yahoo Smartsort; Fan, Gordon, & Pathak, 2003). As technology continues to advance, e-businesses will employ these technologies in more intelligent ways to a wide range of products, eventually improving and customizing all aspects of the purchasing experience.

Many practitioners and researchers look toward software agents to automate the daily and mundane activities that can be handled through the Web. In order for this to occur, agents must be able to easily learn the preferences of the person (or employer) they are representing for whatever task they are trying to accomplish. This represents a significant barrier to widespread general agent use as, currently, a significant time investment is required of the employer, domain experts, and developer(s) to create an agent capable of accurately modeling behavior/decisions on complex tasks. An efficient and effective methodology is needed to elicit and model an employer's preferences and implement this model in an agent. In addition, a market must be created where these agents can perform their tasks of negotiating and trading on behalf of their employers.

A virtual market of this type would allow homogeneous agents to perform these tasks while working together under the supervision of a mediator to accomplish individual or shared goals. Such a market could be developed for online purchasing of a variety of products. In this paper, we restrict our attention to a particular class of *limited supply perishable asset* (LiSPA) products. LiSPA products share the distinction of requiring the assignment of a collected group of customers to products based on their personal preferences. Some examples of LiSPA products include stadium and venue seating, classroom and/or teacher allocation, cruise ship room assignment, airline flight seat assignment, and vacation property management.

Timeshares provide a specific example of vacation property management. A timeshare is a joint ownership of a vacation property by several people who take turns occupying the property at different periods of time. Timeshare owners have the option of trading their prescribed time and place with other owners through a mediator (timeshare property manager). An owner who desires to trade must contact a timeshare property management service and request a new time and/or place. In turn, the property manager would have to search for an appropriate vacancy or, failing that, look for another owner who wants to trade and is willing to give up his or her timeshare. Online portals currently exist where this type of information can be exchanged. However, a time share property manager must ultimately decide how to allocate the inventory of available accommodations to the various individuals wanting to stay at a particular property during a given time period. This process could be automated and improved by: (1) accurately assessing and modeling the preferences of customers regarding different features/attributes of time share products; (2) creating an agent to express a customer's preference for different decision alternatives that arise, and (3) developing a mediator to oversee the process of allocating accommodations to different customers in an optimal manner.

In this paper we introduce and test a methodology for improving customer satisfaction in mediated LiSPA purchases. We do this by modeling individual customer preferences and developing agents to express preferences on their behalf. We use data collected by conjoint analysis (CA) and modeled using a neural network (NN) to develop customer preference models. We show that using a linear programming modeling technique as a virtual mediator in conjunction with our preference modeling agents results in an effective system for improving product allocation. The background section of this paper briefly introduces CA and software agents. In section 3 we illustrate our methodology with an example application involving timeshare property management. Section 4 presents our analysis and results, and conclusions and possible future work are given in section 5.

2. BACKGROUND

2.1 Conjoint Analysis

To develop a market where agents express preferences for various products, a method is needed for eliciting and modeling individual customer preferences. Consumer group preference elicitation is a mature research area that has been used by marketers for decades. Conjoint analysis (CA) is a well-accepted marketing technique for eliciting and modeling consumer group preferences. CA's origin is credited to work by Luce and Tukey (1964), then called trade-off analysis, and was made popular by Green, *et. al.* (Green & Srinivasan, 1978). Since 1978, CA has primarily been used for marketing purposes but has largely been ignored by investigators in operations research and management science (ORMS) (Green, Krieger, & Wind, 2001). However, ORMS researchers recently have employed CA in optimization type problems such as multi-period promotion (Nair & Tarasewich, 2003) and hospital advertising (Tscheulin & Helm, 1998).

Aggarwal and Vaidyanathan (2003) compare CA versus self-explication for eliciting online peferences. It is apparent that the Web and e-commerce can provide a vehicle for exploiting the strengths of CA. From a modeling perspective, CA provides a fairly easy mechanism for modeling individual customer preferences. Such a methodology is suggested in the previous chapter of this work. For further reading on CA see Green & Srinivasan (1978) and Green, Krieger, & Wind (2001).

2.2. Software Agents

Researchers continue to debate the meaning of the term *software agent*. Hess, Rees & Rakes (2000) define a software agent as having a homeostatic goal, persistence, and reactivity, among other aspects, while Maes (1995) defines an electronic agent as "a software program that 'knows' users' preferences and can act autonomously on their behalf." Both definitions provide flexibility in agent designation, and the agent paradigm used in this research is consistent with these definitions.

One of the major hurdles that must be cleared in developing good online shopping environments is the need for assistance in preference construction and discovery (West *et. al.*, 1999). Software agents appear to be the most encouraging alternative for enabling preference knowledge, but knowledge acquisition and representation remains one of the great challenges of AI (and agent) systems (Feigenbaum, 2003). This research investigates the use of CA for knowledge acquisition and NNs for knowledge representation to create agents capable of accurately expressing preferences for decision alternatives on behalf of the individual consumers.

3. METHODOLOGY

3.1 Rationale

In order to effectively maximize the aggregate satisfaction of customers for LisPA products (e.g., venue seating, timeshares, cruise vacations), a virtual market must be created where customers can easily instantiate agents to act on the their behalf. Traditionally, vendors of LiSPA products use a business rule such as donation level or first-in, first-out (FIFO) to determine a priority order for their customers. Using the priority order, customers are given the option to choose their product from the remaining available inventory. This often results in the high-priority customers obtaining a highly preferred product (for them), while the low-priority customers "get stuck" with the remainder and can only choose from what the customers ahead of them do not choose. By eliciting and modeling customer preferences and using a virtual mediator, the negative effects of a prioritization system can be overcome so that everyone has a better opportunity to obtain a product that is closer to their ideal than their initial selection, thus improving the aggregate satisfaction level of the customer group.

3.2 Preference Elicitation and Modeling Procedure

In this study, preferences were elicited via an experiment where the participants completed a full-profile conjoint analysis (CA) and were modeled using a special neural network (NN) procedure (described in the previous chapter). For the CA, participants were shown the same 25 product profiles (a.k.a. "cards") in the form of a paper CA instrument, and were asked to uniquely rate each product from 0 (least preferable) to 100 (most preferable). These participant-specified ratings are used to develop preference models using a NN methodology that employs pairwise comparisons of the CA data. Using this methodology, we can expand the original 25 data points to 600 NN training records.

A card can be described as $c_i = (a_{1_i}, a_{2_i}, \ldots, a_{n_i})$ where element a_{k_i} represents the level of attribute a_k contained in card c_i. For each card, c_i, each participant provides a unique rating, r_i. For each card pair, c_i and c_j, there are associated ratings, r_i and r_j, such that a binary ordinal preference between cards c_i and c_j can be defined as:

$$y_{ij} = \begin{cases} 0, & \text{if } c_i \text{ is preferred} \\ 1, & \text{if } c_j \text{ is preferred} \end{cases}$$

By taking every possible combination pair of cards, c_i and c_j, where $i \neq j$, $1 \leq i \leq 25$, and $1 \leq j \leq 25$ each c_i, c_j combination appears as two records in the training set represented as:

$$[\ c_i, c_j, y_{ij}\ ]$$

$$[\ c_j, c_i, y_{ji}\ ]$$

This is illustrated in Figure 1

		TRAINING DATA												
i	j	c_i					c_j					y_{ij}	r_i	r_j
j	i	c_j					c_i					y_{ji}	r_j	r_i
1	2	4	4	1	1	3	5	5	2	3	4	1	88	92
2	1	5	5	2	3	4	4	4	1	1	3	0	92	88
1	3	4	4	1	1	3	3	5	1	5	1	1	88	100
3	1	3	5	1	5	1	4	4	1	1	3	0	100	88
1	4	4	4	1	1	3	3	1	5	3	3	0	88	27
4	1	3	1	5	3	3	4	4	1	1	3	1	27	88
1	5	4	4	1	1	3	5	3	1	2	5	0	88	56
5	1	5	3	1	2	5	4	4	1	1	3	1	56	88
1	6	4	4	1	1	3	1	4	2	4	1	0	88	75
6	1	1	4	2	4	1	4	4	1	1	3	1	75	88
1	7	4	4	1	1	3	4	5	5	4	5	1	88	90
7	1	4	5	5	4	5	4	4	1	1	3	0	90	88
1	8	4	4	1	1	3	5	2	3	4	3	0	88	44
8	1	5	2	3	4	3	4	4	1	1	3	1	44	88
1	9	4	4	1	1	3	4	2	4	5	4	0	88	40
9	1	4	2	4	5	4	4	4	1	1	3	1	40	88
1	10	4	4	1	1	3	2	2	5	2	1	0	88	52
10	1	2	2	5	2	1	4	4	1	1	3	1	52	88
1	11	4	4	1	1	3	3	4	3	2	4	0	88	80
11	1	3	4	3	2	4	4	4	1	1	3	1	80	88
1	12	4	4	1	1	3	5	4	5	5	2	0	88	84
12	1	5	4	5	5	2	4	4	1	1	3	1	84	88
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
25	24	5	1	4	1	1	2	3	2	5	3	1	32	69
24	25	2	3	2	5	3	5	1	4	1	1	0	69	32

Figure 4-1

NN Training Data

3.3 The virtual marketplace

For this study, we create a virtual marketplace for a generic LiSPA product. Thirty customer agents are created to express preferences for and be assigned to *n* products. Several experiments are performed as outlined in section 4 of this paper to examine the effect of different parameters on the assignment process. By applying each agent's NN preference model to the entire inventory of products, ordinal preferences can be determined for all products. Using each agent's preferences, a virtual mediator assigns products to agents to optimize a prescribed goal.

3.3.1 Determining Preferences

Preferences are determined using a multi-step process. First, each agent evaluates each product in the market against all other products in the market. Agents do this by using their previously described NNs to construct an n x n preference matrix of pairwise product comparisons. For example, each agent compares product c_1 to all products ($c_1, c_2, \ldots, c_n$) one-at-a-time, then c_2 to all products ($c_1, c_2, \ldots, c_n$) one-at-a-time, continuing until c_n is compared to c_n. Note that when identical products are compared to each other the agent will be indifferent to either package. The agent's preference measures p_{ij} are integer values defined as:

$$p_{ij} = \begin{cases} -1, & \text{if } c_i \text{ is preferred over } c_j \\ 0, & \text{if indifferent between } c_i \text{ and } c_j \\ 1, & \text{if } c_j \text{ is preferred over } c_i \end{cases}$$

Figure 2 further illustrates the preference matrix.

c_i	c_j				
	c_1	c_2	c_3	$\cdots$	c_n
c_1	$p_{1,1}$	$p_{1,2}$	$p_{1,3}$	$\cdots$	$p_{1,n}$
c_2	$p_{2,1}$	$p_{2,2}$	$p_{2,3}$	$\cdots$	$p_{2,n}$
c_3	$p_{3,1}$	$p_{3,2}$	$p_{3,3}$	$\cdots$	$p_{3,n}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
c_n	$p_{n,1}$	$p_{n,2}$	$p_{n,3}$	$\cdots$	$p_{n,n}$

Figure 4-2
Preference Matrix

Figure 4-3 illustrates a sample matrix with values.

c_i	c_j				
	c_1	c_2	c_3	…	c_n
c_1	0	1	-1	…	1
c_2	-1	0	-1	…	0
c_3	1	1	0	…	-1
⋮	⋮	⋮	⋮	⋮	⋮
c_n	-1	0	1	…	0
Number of times c_j is preferred	2	7	3	…	10

Figure 4-3
Preference Matrix with Sample Values

A count of the number of 1's that appear in each column in Figure 4-3 provides a measure of preference for each package in the market. That is, each "1" represents a comparison in which card c_j is preferred over card c_i. Therefore, the more times the number 1 appears in each column, the more times the associated product is preferred in the pairwise comparisons, and hence, the more preferable the product is relative to the rest of the inventory. A count of n-1 means card c_j is preferred over all the others, a count of 0 means card c_j is not preferred over any of the others. All other counts can be interpreted as a relative preference of each product in the market such that the higher the count, the more preferred the product. However, it is inappropriate to use these values to explicitly quantify numeric ratio-based preference values of each product as the agent uses the NN only to predict ordinal preference -- not a ratio-based preference rating. Thus, the count for agent k is used to assign an ordinal rank to each product (denoted R_{k_j}) where the highest count is assigned a rank of 1, second highest, 2, and so on.

It is not uncommon to have two or more products with equal counts for a given agent. This could occur for one of three reasons: 1) the agent is truly indifferent to the products in question; 2) the method of counting does not unveil the true difference in preferences; or 3) the agent is inconsistent in estimating preference for the products in question. If the agent is truly indifferent to the products with equal counts (case 1 above), the preferences, p_{ij} for any "tied" products will equal zero. In this case all "tied" products will receive the same rank value.

In the case of "tied" products where the method of counting does not unveil the true difference (case 2 above), we assess all the possible pairwise comparisons among the "tied" products in question, and count the total number of times each product is preferred to the others in question (vote style). If a clearly preferred product is unveiled by the vote, rank is appointed accordingly (the one with the highest number of votes gets the first available rank, the one with the next highest vote gets the next available rank, and so on), otherwise equal rank is given to any remaining "tied" products.

Each time a tie occurs that is not a result of indifference or cannot be resolved with a vote, the agent's preference for the products in question are deemed inconsistent (case 3 above). While inconsistency is generally an undesirable occurrence, we note that an agent can be no more accurate than its human employer and humans are sometimes inconsistent in their stated preferences for multiple decision alternatives. We later provide a measure for assessing the amount of inconsistency associated with in our proposed technique. Computational testing indicates inconsistency is rare in the testing scenarios presented in this paper.

The resulting preference measures R_{kj} are ranks from 1 to a maximum of n with R_{kj} =1 being the most preferred. Measuring preferences by ranking in this manner enables us to determine optimal product assignments.

3.3.2 Improving Product Assignments

It can be shown that the FIFO heuristic assignment as described in section 3.1 may be sub-optimal in terms of the maximum satisfaction of the market since there is a priority order to the initial assignments. That is, it is likely the high priority agents will obtain a highly preferred product (for them), whereas the low priority agents are likely to be assigned a “leftover” one. Given the ability to instantiate an agent that can effectively assess one’s preference between two products, it may be possible to arrange more desirable product assignments. Depending on the mediator’s goal, new product assignments can result in agents obtaining more preferred, less preferred, or indifferent products over their originals;ing however, the aggregate satisfaction of the entire market should not decrease. This is accomplished by using a linear program model as the market mediator.

3.3.3 The LP Model

The LP model used in this study is that associated with the well-known assignment problem (Garfinkel & Nemhauser, 1972). For a problem involving n agents and m products, x_{ij} represents a binary variable indicating whether agent i is assigned to product j, and R_{ij} represents agent i’s rank (preference) for product j in the market (recall that a rank of 1 is most preferred). Since the preference is measured by rank (where a lower number is better), we would solve the LP by minimizing z, our measure of overall market preference. The formulation of this problem is as follows:

$$\min z = \sum_{i=1}^{n}\sum_{j=1}^{n} R_{ij} x_{ij}$$

subject to:

$$\sum_{j=1}^{m} x_{ij} = 1, \qquad i = 1, \ldots, n$$

$$\sum_{i=1}^{n} x_{ij} \leq 1, \qquad j = 1, \ldots, m$$

$$x_{ij} = 0, 1$$

3.4 Market Implementation

Overall, the market is implemented by creating customer agents and products. The agents evaluate each product in the market and provide the mediator with their evaluations, and the mediator in turn assigns products to agents with the objective of maximizing overall market preference. A summary of the proposed methodology follows:

0. <u>CA & NN are used to develop an agent for each customer</u>
1. <u>A priority number is assigned to each agent (optional)</u>. The need to assign priority orders to the customers and their agents will vary based on the business rules of the vendor and/or broker.

2. Each agent ranks each inventory product. The initial inventory of products is determined by vendors. The methodology in section 3.2 is used to rank each product.
3. Each agent chooses its FIFO product (optional). The FIFO products are chosen in priority order. This simulates a "schoolyard" style of choosing, where the highest priority agent chooses its most preferable product, and that product is removed from the market. Then the second highest priority agent picks its most preferable product of the remaining, and it is removed from the market. This continues until each agent has picked a product.
4. Mediator assigns products to agents. This is accomplished using a linear programming model (section 3.3.4) where each agent is assigned to exactly one product. The preference value (rank) for each product by each agent is used in the objective function in the linear program. Therefore, by summing the ranks of the assigned products for all agents, we obtain a measure for total market satisfaction. Minimizing this value provides us with a product assignment that represents the minimum (most desirable) total aggregate market rank.

4. ANALYSIS AND RESULTS

4.1 Data Collection and Experiments

In order to examine the effectiveness of our proposed methodology we simulate a virtual market of agents and products and answer three questions through experimentation. First, does our methodology produce assignments that improve the overall satisfaction of the market?

Second, can we make overall improvements to the market without any individual sacrificing their assignment for a less preferred one? Third, do different market assumptions alter the ability of our methodology to make improvements?

Twelve experiments are performed as illustrated in Table 4-2 using the preference modeling and virtual marketplace procedure described in Section 3. Our preference models are developed using data collected in a study of 92 participants. The 30 most accurate models are included in our marketplace. Thirty (30) trials are performed for each experiment. The experimental factors are: (1) product ownership, (2) market objective, and (3) size of the initial product inventory. Product ownership assumption “Owns” represents a scenario where the customer has a claim to their FIFO product and is not willing to trade his product for a less desirable product. Product ownership assumption “Does not own” means that the customer does not have a claim to his FIFO product and everyone has an equal chance of obtaining any product.

Finally, the product inventory size is varied, changing the amount of flexibility each agent has in choosing products. Holding the number of participants constant at 30, the pool of products from which agents choose their FIFO (and final) product is changed to the different levels of 30, 45, and 60.

Experiment Number	Product Ownership Assumption	Objective	Product Inventory Size
1	Does not own	Maximize overall market satisfaction	30
2			45
3			60
4		Maximize overall least preferred product	30
5			45
6			60
7	Owns	Maximize overall market satisfaction	30
8			45
9			60
10		Maximize overall least preferred product	30
11			45
12			60

Table 4-1
Experiments

In general, these experiments entail creating customer agents and market products, and placing both in a virtual market. The experiments and results are detailed in this section.

For the first set of experiments (1, 2, & 3) we assume the agents do not already own a product (*does not own*) and the objective of the mediator is to maximize overall market preference. To test this we begin by building each agent's preference model, having each agent evaluate all products in the market, and allowing each agent to choose its FIFO product. The FIFO product ranks are recorded and used to measure market satisfaction. Summing the ranks of the FIFO products for all agents provides us with a cumulative measure of market satisfaction. Table 4-4 reports the minimum, average (30 trials), and maximum values at each product inventory level. In general, the inventory size of products has no impact on the relative performance of our methodology, although as the inventory increases, the average rank of all assigned (or chosen) products improves (e.g., average FIFO rank is 9.2 for 30 products and 5.5 for 60). Overall, using our methodology improves aggregate market preference by 32% (from 212.82 to 143.89). By dividing these values by 30 (the number of agents) we get an average rank improvement from 7.1 to 4.8. In addition, the average worst rank (this represents the lowest ranking product that any agent chose) observed for each case on average is 20.42 for FIFO and 13.11 for *does not own* (35.8% improvement).

Product Inventory Size (n)		Minimize sum of ranks (maximize preference)				Minimize worst rank		
		Total of Ranks		Worst Rank		Total of Ranks	Worst Rank	Best Rank
		FIFO	Does not own	FIFO	Does not own	Does not own	Does not own	Does not own
30	Min	215	150	20	12	215	12	1
	Avg	276.27	189	26.13	18.53	274.17	16.1	1.07
	Max	318	227	30	26	315	21	2
45	Min	152	90	13	7	150	7	1
	Avg	197.73	132.17	19	11.4	193.53	10.8	1
	Max	245	173	25	15	245	15	1
60	Min	125	81	11	6	120	6	1
	Avg	164.47	110.5	16.13	9.4	165.27	9.2	1
	Max	194	139	20	14	205	12	1
All	Min	125	81	11	7	120	6	1
	Avg	212.82	143.89	20.42	13.11	210.99	12.03	1.02
	Max	318	227	30	26	315	21	2

Table 4-2
FIFO vs. Does Not Own Results

Following completion of experiments 1-3, we executed experiments 4-6 by changing the objective function (using the agent-determined product ranks from experiments 1-3) and employing a genetic algorithm to minimize the worst rank in the market. This resulted in much better "worst rank" results (12.03) compared to the FIFO technique (20.42), and slightly better results than the *does not own* scenario (12.03 vs. 13.11). Note, however, the total of ranks increase from 143.89 using the *does not own scenario* to 210.99 when minimizing the worst individual rank.

In experiments 7-9, the *owns* scenario, requires that each agent modify its preferences by "blacklisting" products that are less preferred (ranked lower) than its FIFO product. Blacklisting is accomplished by assigning a rank of 999 to any less preferred product(s). Results of this testing are reported in Table 4-4.

Product Inventory Size (n)		Minimize sum of ranks (maximize preference)				Minimize worst rank		
		Total of Ranks		Worst Rank		Total of Ranks	Worst Rank	Best Rank
		FIFO	Owns	FIFO	Owns	Owns	Owns	Owns
30	Min	215	215	20	20	215	20	1
	Avg	276.27	274.17	26.13	26.13	274.17	26.13	1
	Max	318	315	30	30	315	30	1
45	Min	152	150	13	13	150	13	1
	Avg	197.73	193.53	19	19	193.53	19	1
	Max	245	245	25	25	245	25	1
60	Min	125	120	11	11	120	10	1
	Avg	164.47	158.37	16.13	15.9	165.27	13.73	1
	Max	194	194	20	20	205	20	1
All	Min	152	120	11	11	120	10	1
	Avg	212.82	208.69	20.42	20.34	210.99	19.61	1
	Max	318	315	30	30	315	30	1

Table 4-3
FIFO vs. Owns Results

As before, as product inventory increases, average rank improves. The *owns* results, however, are only slightly better than FIFO, from an improvement of 1% (inventory of 30 products) to 4% (inventory of 60 products). This is due to the fact that the mediator's ability to improve aggregate market satisfaction is severely restricted when it is prohibited from assigning a customer a product that is less desirable than the one the customer owns (even if such an assignment would allow large gains in satisfaction for other customers).

Experiments 10-12 followed by again blacklisting any product less desirable than FIFO and minimizing the overall worst rank in the market (results are shown in Table 4-4). The greatest improvement for this experiment is at inventory 60 (15%). Note that improvements in these experiments occur only when customers are indifferent between the product they own and other available products. Such indifference becomes more likely as the inventory of available products increases thereby creating greater opportunity for the mediator to improve the product allocations without assigning any customers to less desirable products.

4.2 Overall Results

Summarizing the results, Figure 4-4 shows the average ranks for each method at each inventory level. From Figure 4-4 we see that increasing the inventory level improves the average rank in all cases. It is also clear from Figure 4-4 that the *does not own* method of product allocation does much better than FIFO while the *owns* method only slightly outperforms FIFO. Figures 4-5 through 4-7 illustrate the mean product rank for each participant (horizontal axis) for inventories of 30, 45, and 60, respectively. Note that although *does not own* in these figures show a slight improvement over the FIFO assignments, their points may be indistinguishable on the graph. Figures 4-5 through 4-7 graphically illustrate the source of the market improvements. In all cases (inventories 30, 45, and 60) the graphs show that the FIFO method benefits the high priority agents at the expense of the low priority agents, whereas the *does not own* method creates an equal benefit to all participants.

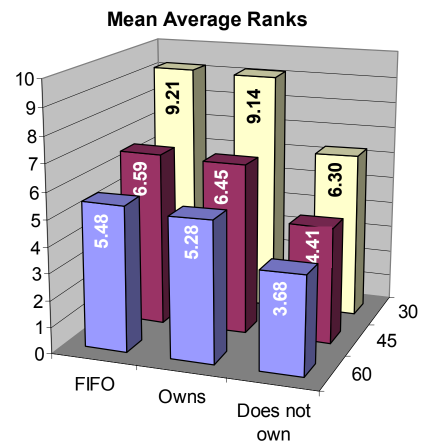


Figure 4-4
Summary of Results

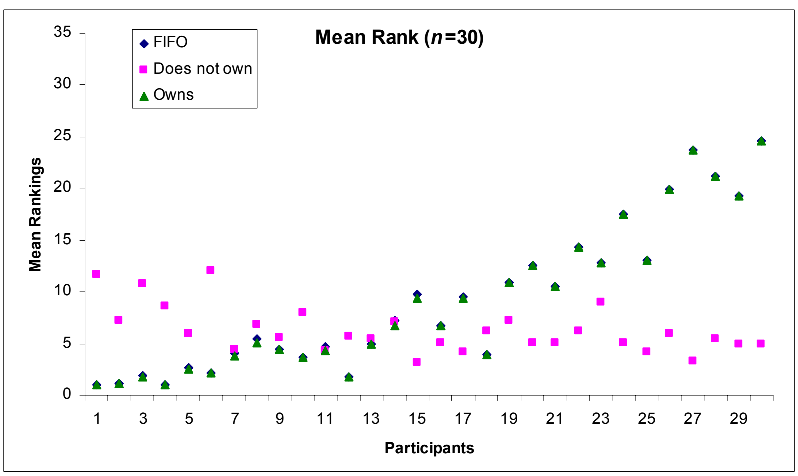


Figure 4-5
Average Rankings, n=30

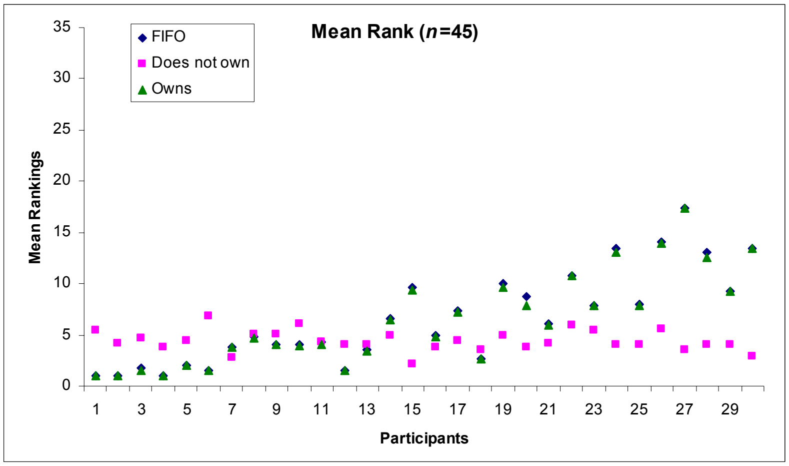


Figure 4-6
Average Rankings, n=45

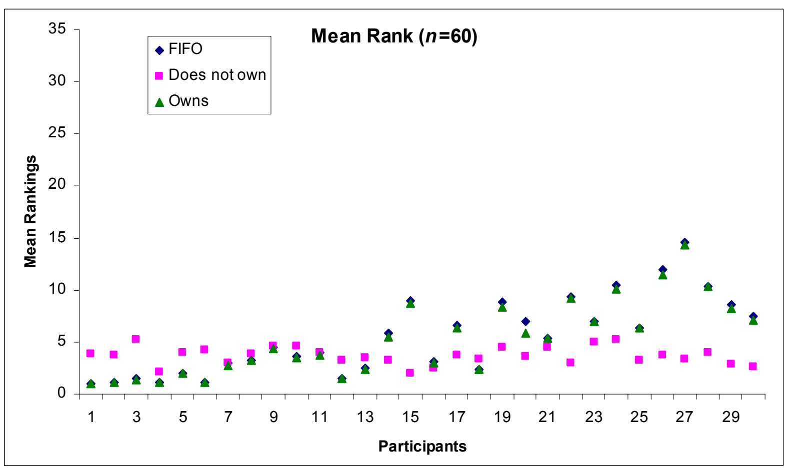


Figure 4-7
Average Rankings, n=60

4.3 Consistency

During the execution of these experiments, it was observed that agents were assigning duplicate ranks to many packages. Many of these duplicate ranks were a result of actual indifference among products, but occasionally, a circular inconsistency arose (as described in section 3.3.2). For example, suppose products A, B and C are all initially equally ranked. Initiating a pairwise comparison vote among A, B, and C, the agent determines that A is preferred over B, B is preferred over C, but then C is preferred over A. This cannot be resolved, and we assume for this case that the agent, not unlike a real person, cannot determine a preference for any of the three products. Every occurrence of this example was recorded during experimentation, and the results are presented in Table 4-5. The first main row of this table reports the number of times inconsistency was encountered in the entire market (30 participants) for each inventory level (*m*). The mean value represents the number of packages out of 900, 1,350, 1,800 for inventory levels of 30, 45, and 60, respectively. Thus, inconsistencies occurred in less than 1.5% of the preference matrices created in these experiments. The second main row in the table represents the total number of products involved when these inconsistencies occurred.

		Inventory (*m*)		
		30	45	60
Total number of times an inconsistency was encountered	Min	6	14	11
	$\bar{x}$	14.5	20.67	24.37
	Max	22	30	36
Total number of inconsistent packages	Min	18	42	33
	$\bar{x}$	43.77	62.33	73.17
	Max	66	92	108

Table 4-4
Consistency Check

5. CONCLUSIONS AND FUTURE WORK

In this paper we introduce a class of products, termed LiSPA, and provide a methodology for creating a market that accounts for customer preferences through the use of agents and improves overall customer satisfaction. We develop a virtual market that allows agents to evaluate market products and uses a mediator to assign products to agents. We compare our assignments to a commonly used FIFO method and show significant improvements in overall satisfaction.

Although we show promising results for our methodology, a number of improvements can be made and further issues investigated. One such investigation involves further understanding and better measurement of inconsistency. Our measurement shows a very small portion of the total market to be inconsistent, and perhaps it is that the agents are acting as their employers and truly cannot distinguish from the products. Nevertheless, further exploration and understanding is warranted.

Another improvement is to extend the market to a variable pricing scheme, where agents can truly negotiate with each other over price and the mediator role is possibly changed to finding good candidates for negotiation.

REFERENCES

Aggarwal, P. & Vaidyanathan, R. (2003). Eliciting online customers' preferences: conjoint vs. self-explicated attribute-level measurements. *Journal of Marketing Management*, 19, 157-177.

Amazon Homepage. (n.d.). Last accessed February 1, 2004, from www.amazon.com

Fan, W., Gordon, M., & Pathak, P. (2003). A generic ranking function discovery framework by genetic programming for information retrieval. *Information Processing & Management*. In press.

Feigenbaum, E. (2003). Some challenges and grand challenges for computational intelligence. *Journal of the ACM,* 50(1), 32-40.

Garfinkel, R. & Nemhauser, G. (1972). *Integer Programming*. New York, NY: John Wiley. 61-62.

Green, P., Krieger, A., & Wind, Y. (2001). Thirty years of conjoint analysis: reflections and prospects. *Interfaces*, 31(3), Part 2 of 2, S56-S73.

Green, P. & Srinivasan,V. (1978). Conjoint analysis in consumer research: issues and outlook, The Journal of Consumer Research, 5(2), 103-123.

Hess, T., Rees, L., & Rakes, T. (2000). Using autonomous software agents to create the next generation of decision support systems. *Decision Sciences,* 31(1), 1-31.

Lee W., Liu C., & Lu C. (2002). Intelligent agent-based systems for personalized recommendations in internet commerce. *Expert Systems with Applications,* 22(4), 275-284.

Luce, D. & Tukey, J. (1964). Simultaneous conjoint measurement: a new type of fundamental measurement, *Journal of Mathematical Psychology*, 1, 1-27.

Maes, P. (1995). Intelligent software. *Scientific American, 150 Anniversary Issue: Key Technologies for the 21st Century.*

Nair, S. & Tarasewich, P. (2003). A model and solution method for multi-period sales promotion design. *European Journal of Operational Research*, 150, 672-687.

Tscheulin, D. & Helmig, B. (1998). The optimal design of hospital advertising by means of conjoint measurement. *Journal of Advertising Research*, (38)3, 35-46.

West, P., Ariely, D., Bellman, S., Bradlow, E., Huber, J., Johnson, E., *et. al*. (1999). Agents to the rescue? *Marketing Letters* 10(3), 285-300.

Yahooo Smartsort. (n.d.). Retrieved February 1, 2004, from http://shopping.yahoo.com/smartsort/

Chapter 5

Summary and Directions of Future Research

SUMMARY

Implementing an e-market for LiSPA products is a problem at the intersection of online purchasing and DistDSS. It involves the purchase of similar but differentiable products by customers where the products are in limited supply. Three examples of LiSPA products used in this dissertation are vacation packages, venue seats, and timeshares, but the online LiSPA techniques discussed in this research can be extended to the other products. In this dissertation, we introduce LiSPA and provide real-world examples, develop a framework for a distributed system to implement an e-market for LiSPA products, and provide proof-of-concept for the two major components of the framework.

Our distributed system framework uses Web Services for communication. This provides many advantages such as a guarantee of the most recent software and algorithms and privacy of data for all parties involved. The main system components are the customer agent enabler and the model base.

A key hurdle for the agent enabler is to be able to elicit and model customer preferences in a quick and easy manner. A methodology was developed for this problem using conjoint analysis and neural networks. A study was conducted to test our methodology. Using the data from the study, several regression and neural network models were constructed, tested, and compared for each participant.

The model component of this framework is implemented in this work. The model uses the preference models mentioned above and employs a linear programming model to maximize overall satisfaction of the total market.

FUTURE RESEARCH

Predicting Explicit Preference Values

In this research we use a full-profile conjoint analysis to elicit individual customer preferences and a neural network to build the preference models. Although the methodology performed exceptionally well for predicting pairwise comparisons, it did a poor job at predicting actual rating values. Development of a methodology to improve actual rating predictions by employing other quantitative techniques will open up several more doors for this work.

Exploring Model Inconsistency

In the market model portion of this study, we quantified inconsistency for each preference model. Inconsistency occurs when the model cannot determine a preference among more than 2 products. For example, suppose product A is preferred over product B in a pairwise comparison, then product B is preferred over product C in a second pairwise comparison, and finally, C is preferred over product A in a third pairwise comparison. A circular inconsistency exists in this example that should be explored.

A Variable Priced Market for Online Negotiation

The next step for improving this process is the development of a market where agents can truly negotiate with other agents for products using a variable pricing model. By improving the ability to accurately predict a precise value (instead of a pairwise preference), price can be used in a conjoint analysis as a response instead of a rating. Sensitivity to price can then be modeled and negotiation can occur.

Implementation of the LiSPA Framework

Now that we have provided proof-of-concept for the preference and model portions of our framework, a prototype system can be built in order to explore the difficulties with implementing such a system.

CONCLUSIONS

This research introduces an interesting and overlooked online problem in LiSPA. In this dissertation we describe a realistic framework for a DistDSS LiSPA product market. We provide proof of concept for a realistic and theoretically sound methodology for eliciting and modeling individual customer preferences that outperforms currently accepted methods. Furthermore, we develop a methodology for enabling agents to use these preference models to trade products on customers' behalf.

REFERENCES

Aggarwal, P. & Vaidyanathan, R. (2003). Eliciting online customers' preferences: conjoint vs. self-explicated attribute-level measurements. *Journal of Marketing Management*, 19, 157-177.

Amazon Homepage. (n.d.). Last accessed February 1, 2004, from www.amazon.com

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web. *Scientific American,* 284 (5), 34-43.

Bhargava, H. & Power, D. (2001). Decision support systems and web technologies: a status report. *Seventh Americas Conference on Information Systems (AMCIS '01)*. Association for Information Systems. Boston, MA.

Bradshaw, J. M. (1997). *Software Agents.* Menlo Park: American Association for Artificial Intelligence.

Bui, T. & Lee, J. (1999). An agent-based framework for building decision support systems. *Decision Support Systems*, 25(3), 225-237.

Burke, L. (1991). Introduction to artificial neural systems for pattern regognition. *Computers and Operations* Research 18(2), 211-220.

Buss, A. (2000). Component-based simulation modeling. *Winter Simulation Conference (WSC '00)*. IEEE. Orlando, FL.

Buss, A. & Stork, K. (1996). Discrete event simulation on the world wide web using java. *Winter Simulation Conference (WSC '96)*. IEEE. Piscataway, NJ.

Chu, P.C. & Beasley, J.E. (1997). A genetic algorithm for the generalized assignment problem. *Computers and Operations Research* 24(1), 17-23.

Cohen, M., Kelly, C., & Medaglia, A. (2001). Decision support with web-enabled software. *Interfaces,* 31(2), 109-129.

Cubert, R. & Fishwick, P. (1997). A framework for distributed object-oriented multimodeling and simulation, *1997 Winter Simulation Conference (WSC '97)*. IEEE. Atlanta, GA.

Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., & Weerawarana, S. (2002). Unraveling the web services web: an introduction to SOAP, WSDL, and UDDI. *IEEE Internet Computing*, 6(2), 86-93.

Fan, W., Gordon, M., & Pathak, P. (2003). A generic ranking function discovery framework by genetic programming for information retrieval. *Information Processing & Management*. In press.

Fatsis, S., (2002). The Barry Bonds tax: teams raise prices for good games, *The Wall Street Journal*. New York, 3 December, D1.

Feigenbaum, E. (2003). Some challenges and grand challenges for computational intelligence. *Journal of the ACM,* 50(1), 32-40.

Fu, M. (2002). Optimization for simulation: theory vs. practice. *INFORMS Journal on Computing*. 14(3), 192-215.

Garfinkel, R. & Nemhauser, G. (1972). *Integer Programming*. New York, NY: John Wiley. 61-62.

Gershoff, A. & West, P. (1998). Using a community of knowledge to build intelligent agents. Marketing Letters, 9(1), 79-91.

Glass, G. (2002). *Web services: building blocks for distributed systems*. Upper Saddle River: Prentice Hall.

Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimisation, and Machine Learning*. Reading, MA: Addison-Wesley.

Gorry, G.A. & Scott Morton, M.S. (1989). A framework for management information systems. *Sloan Management Review,* 13(1), 49-62.

Green, P., Krieger, A., & Wind, Y. (2001). Thirty years of conjoint analysis: reflections and prospects. *Interfaces*, 31(3), Part 2 of 2, S56-S73.

Green, P. & Srinivasan,V. (1978). Conjoint analysis in consumer research: issues and outlook, *The Journal of Consumer Research*, 5(2), 103-123.

Green, P. & Wind, Y. (1975). New ways to measure consumers' judgements. *Harvard Business Review*, 53(4), 107-117.

Guttman, R., Moukas, G, & Maes, P. (1998). Agent-mediated electronic commerce: a survey. *The Knowledge Engineering Review*, 13(2), 143-152.

Hess, T. (1999). A study of autonomous agents in decision support systems. *PhD. Dissertation*. Department of Management Science and Information Technology, Virginia Polytechnic Institute and State University, Blacksburg, VA, 135.

Hess, T., Rees, L., & Rakes, T. (2000). Using autonomous software agents to create the next generation of decision support systems. *Decision Sciences,* 31(1), 1-31.

Huhns, M. (2002). Agents as web services. *IEEE Internet* Computing, 6(4), 93-95.

Kilgore, R., Healy, K., & Kleindorfer, G. (1998). The future of java-based simulation. *1998 Winter Simulation Conference (WSC '98)*. IEEE. Washington, D.C.

Kilgore, R. (2002). Simulation web services with .NET technologies. *2002 Winter Simulation Conference (WSC '02)*. IEEE. San Diego, CA.

Lang, K. & Whinston, A. (1999). A design of DSS intermediary for electronic markets. *Decision Support* Systems, 25(3), 181-197.

Lee W., Liu C., & Lu C. (2002). Intelligent agent-based systems for personalized recommendations in internet commerce. *Expert Systems with Applications,* 22(4). 275-284.

Liu, O., Ma, J., Zhou, D., & Zhang, Q. (2001). A distributed DSS framework based on multi-agent paradigm and its implementation schema. *Seventh Americas Conference on Information Systems (AMCIS '01)*. Association for Information Systems. Boston, MA.

Luce, D. & Tukey, J. (1964). Simultaneous conjoint measurement: a new type of fundamental measurement, *Journal of Mathematical Psychology*, 1, 1-27.

Luo, Y., Liu, K., & Davis, D. (2002). A multi-agent decision support system for stock trading. *IEEE Networking*, 16(1), 20-27.

Maes, P. (1995). Intelligent software. *Scientific American, 150 Anniversary Issue: Key Technologies for the 21st Century.*

MOOSE Homepage. (n.d.). Retrieved April 21, 2002, from http://www.cise.ufl.edu/~fishwick/moose.html

Nair, S. & Tarasewich, P. (2003). A model and solution method for multi-period sales promotion design. *European Journal of Operational Research*, 150, 672-687.

Nghiem, A. (2003). *IT web services: a roadmap for the enterprise*. Upper Saddle River: Prentice Hall.

Oliver, J. (1996) A machine-learning approach to automated negotiation and prospects for electronic commerce. *Journal of Management Information Systems*, 13(3), 83-112.

Orme, B. (2003). Which conjoint method should I use? *Sawtooth Software Research Paper Series*.

Page, E. & Opper, J. (2001). Investigating the application of web-based simulation principles within the architecture for a next-generation computer generated forces model. *Future Generation Computer Systems*, 17, 159-169.

Pinson, S., Louca, J., & Moraitis, P. (1997). A distributed decision support system for strategic planning. *Decision Support Systems*, 20(1), 35-51.

Sprague, R. & Carlson, E. (1982). *Building effective decision support systems*. Englewood Cliffs, NJ: Prentice-Hall.

Tecuci, G. (1998). *Building intelligent agents*. Academic Press.

Tewari, G., Youll, J., & Maes, P. (2002). Personalized location-based brokering using an agent-based intermediary architecture. *Decision Support Systems*, 34(2), 127-137.

Tscheulin, D. & Helmig, B. (1998). The optimal design of hospital advertising by means of conjoint measurement. *Journal of Advertising Research*, (38)3, 35-46.

Vaughan-Nichols, S. (2002). Web services: beyond the hype. *IEEE Computer*, 35(2), 18-21.

Weiss, G. (1999). *Multiagent systems: a modern approach to distributed artificial intelligence*, Cambridge, MA: The MIT Press.

West, P., Ariely, D., Bellman, S., Bradlow, E., Huber, J., Johnson, E., *et. al*. (1999). Agents to the rescue? *Marketing Letters* 10(3), 285-300.

West, P., Brocket, P., & Golden, L. (1997). A comparative analysis of neural networks and statistical methods for predicting consumer choice, *Marketing Science*, 6(4), 370-391.

Whinston, A. (1997). Intelligent agents as a basis for decision support systems. *Decision Support Systems*, 20(1), 1.

Yahooo Smartsort. (n.d.). Retrieved February 1, 2004, from http://shopping.yahoo.com/smartsort/

VITA

FRANK ANTHONY LoPINTO

EDUCATION

- Doctor of Philosophy (Business Information Technology), Pamplin College of Business, Virginia Polytechnic Institute and State University, expected Spring 2004

  Dissertation Title: An Agent-Based Distributed Decision Support System Framework for Mediated Negotiation

  Major Professor: Cliff T. Ragsdale

- Master of Business Administration (ISM), University of North Carolina at Greensboro, 2000.
- Bachelor of Civil Engineering, University of Delaware, 1988.

REFEREED CONFERENCE PROCEEDINGS

1. LoPinto, F.A. (2002). Extending the concepts of Web-Based Simulation to a Web-Based Decision Support System," In the Proceedings of the Thirty-Eighth Annual Meeting of Southeastern INFORMS, Myrtle Beach, SC.

2. Dong, X., Liu, Y., LoPinto, F.A., Scheibe, K.P., Sheetz, S.D. (2002). "Information Model for Power Equipment Diagnosis and Maintenance," In the Proceedings of Power Engineering Society Winter Meeting, New York.

3. LoPinto, F.A. (2001). "Refinements to a Genetic Algorithm for the Generalized Assignment Problem, "In the Proceedings of the Thirty-Seventh Annual Meeting of Southeastern INFORMS, Myrtle Beach, SC.

CONFERENCE PRESENTATIONS

1. "Problem solving, team building, time management, and communication through First Lego League Robotics: developing a championship team in two years," Annual VSTE (Virginia Society for Technology in Education) State Conference, Washington, D.C., March 9-11, 2003.

WORKING PAPERS

1. LoPinto, F.A. and Ragsdale, C.T., "A framework for an agent-based distributed decision support system using Web Services."

2. LoPinto, F.A. and Ragsdale, C.T., "Modeling user preferences using conjoint analysis and neural networks for e-commerce."

3. LoPinto, F.A. and Ragsdale, C.T., "An agent-based mediated market for online purchases."

WORK EXPERIENCE

- 2000-Present, Graduate Assistant, Department of Business Information Technology, Virginia Tech, Blacksburg, VA
- 2003, Instructor, Department of Business Information Technology, Virginia Tech, Blacksburg, VA
- 2000-2002, Network Administrator /IT Specialist, ES&T, Blacksburg, VA
- 1998-2000, Department Manager, DSAtlantic Corporation, Winston-Salem, NC
- 1997-1998, Project Manager, Samet Corporation, Greensboro, NC
- 1995-1997, Project Manager, Foster Wheeler Environmental Corporation, Greensboro, NC
- 1993-1995, Project Manager, Coastal Engineering Consultants, Naples, FL
- 1988-1993, Project Engineer, Lord, Anderson, Worrell and Barnett, Burlington, NJ

TEACHING EXPERIENCE

Virginia Tech

- Instructor, BIT 4594: Electronic Commerce Systems (Fall 2003)
- Instructor, BIT 2406: Quantitative Methods II (3 semesters - 5 sections total)
- Instructor, BIT 2405: Quantitative Methods I (1 semester)
- Graduate Assistant, BIT 5474: Computer Based Decision Support Systems (TV MBA Class, 1 semester)
- Graduate Assistant, BIT 4584: Client / Server Applications Development for Business (2 semesters)
- Graduate Assistant, BIT 4554: Networks and Telecommunications for Business (4 semesters)
- Graduate Assistant, BIT 5984: (special study): Decision Technologies for Managers (TV MBA Class, 1 semester)

SPECIAL AWARDS and HONORS

- R. B. Pamplin Doctoral Fellowship