Providing a Better User-Interface to Explore

Large Product Spaces

by

Ankur Chandra

Submitted to the Department of Electrical Engineering and Computer Science in Partial Fulfillment of the Requirements for the Degrees of Bachelor of Science in Computer Science and Engineering and Masters of Engineering in Electrical Engineering and Computer Science at the Massachusetts Institute of Technology

May 22, 2000

June 2000

Author ______________________________

Department of Electrical Engineering and Computer Science
May 22, 2000

Certified by ______________________________

Henry Lieberman
Thesis Supervisor

Accepted by ______________________________

Arthur C. Smith
Chairman, Department Committee on Graduate Theses

Providing a Better User-Interface to Explore Large
Product Spaces
by
Ankur Chandra

Submitted to the
Department of Electrical Engineering and Computer Science

May 22, 2000

In Partial Fulfillment of the Requirements for the Degree of
Bachelor of Science in Computer Science and Engineering
and Master of Engineering in Electrical Engineering and Computer Science

ABSTRACT

Purchasing a product on the Web is difficult as the product space is gigantic, users often have difficulty specifying what they want, and current systems do not help users learn as they explore the space. The idea in this thesis is to utilize a genetic algorithm to efficiently navigate this hyperspace. From examples, product specifications can be inferred and used to explain the functioning of the genetic algorithm. To test these ideas, we built the Auto-ficial Life system, a generic UI akin to Karl Sims' Genetic Images or Richard Dawkins *Blind Watchmaker*[23][10]. Explanation and control features augment the system's operation. As a result, users efficiently traverse the hyperspace because of the ease in expressing imprecise preferences. This system was implemented in two product domains, that of cars and men's shoes, and provided efficient navigation in both domains.

Thesis Supervisor: Henry Lieberman
Title: Research Scientist, MIT Media Lab

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

I wish to thank my family for their unconditional support. I would like to thank Pattie Maes, for her suggestions and direction, without which, I would not have embarked upon this research project. I extend tremendous gratitude for my advisor, Henry Lieberman, who focused me on the issues and areas of my research, even when this was difficult. His patience and support, particularly on those late drafts of this paper are appreciated. Henry's guidance allowed me to accomplish much more than I had originally thought possible.

Chapter 1

INTRODUCTION

As Mr. Rogers would say, "won't you be my broker?"

Electronic commerce has ushered in a new age of progress. Consumers purchase gifts in their pajamas and businesses streamline their supply-chains. However, the enormous valuations underlying technology companies reflect the expectation of a return by selling products and services. Assisting the customer in choosing a product or service is critical to sustaining the growth of the Internet.

This thesis handles product brokering, the process of assisting a user choose a product. The scope is a specific domain, for example, a consumer looking for a new car. The consumer may be unable to precisely specify what she desires.

Product brokering is the electronic version of shopping. As such, it must be efficient, enjoyable, and user-driven. It combines logical needs and emotional tastes.

Chapter 2

SCENARIO

That's right, it's car-shopping time!

Beth is in the market for a new car. As an accountant, she makes $80,000 a year. She has two pre-teenage children and expects to purchase a minivan to transport her kids from school to soccer practice. However, ever since she was a child, she dreamed of owning a car like her father's Ford Mustang.

Beth wants to find her dream car over the Internet. However, purchasing products over the Internet can be an ordeal. Current methods of finding a product are not conducive to users like Beth. Her criteria are fuzzy and she has preferences, not hard constraints, on her desires. She would not know what to type into a search engine nor does she want to browse through every car in a catalog. Beth specifies what she wants with fuzzy terms that a salesman understands, but a currently-equipped website cannot.

Using Auto-ficial Life, Beth solves the previous problems. Initially she sees a random sampling of cars.

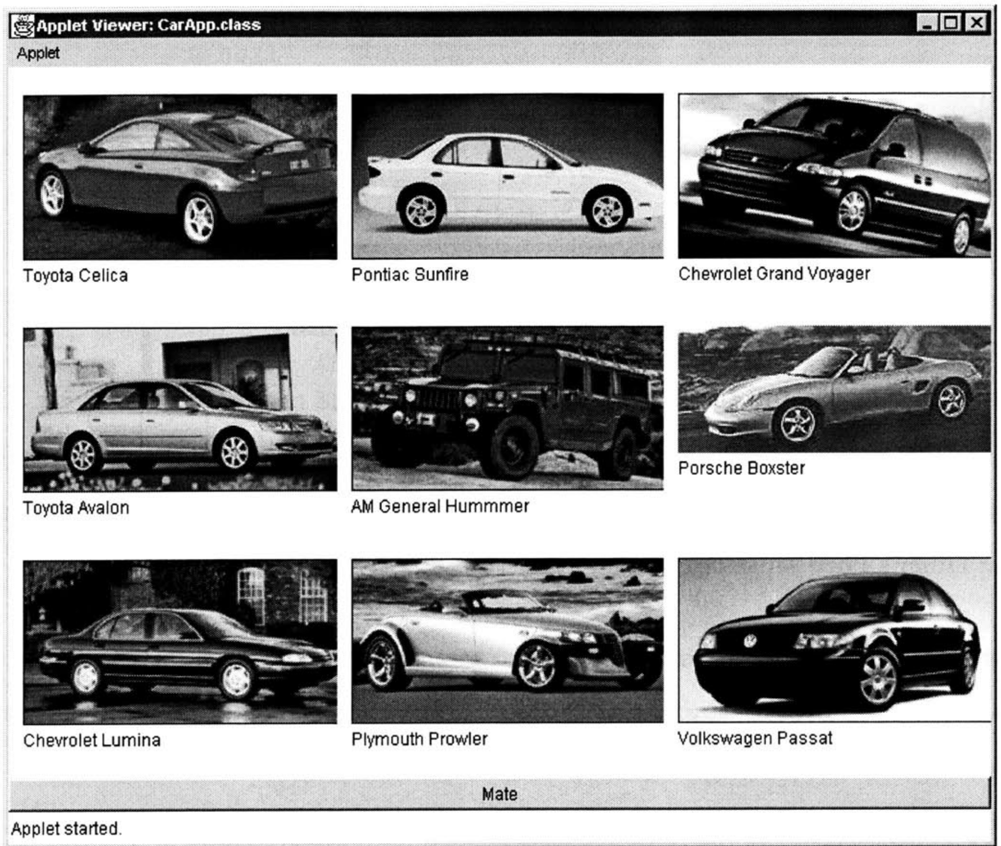

Figure 1: Initial Screen

Beth eyes the minivan, and sees herself driving Timmy's soccer team. She selects it. Her attention is then drawn to the upper-left corner, to the Toyota Celica. She is enamored and selects it as well. Beth then clicks on the mate button for additional cars.

A new generation of cars with traits of both the Celica and the Grand Voyager appears. Beth is interested in the Honda Passport. It is a sports-utility vehicle (SUV), a class of car she had not considered. By right-clicking on the SUV, a dialog pops up depicting the Passport's underlying attributes.

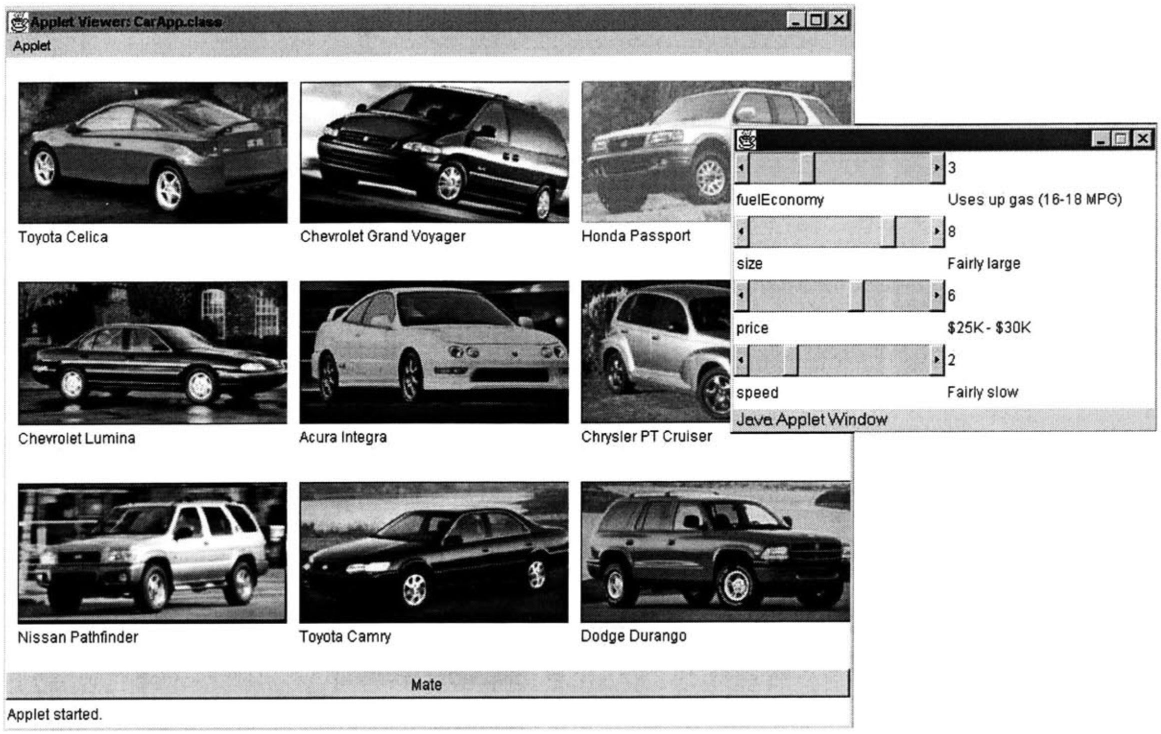

Figure 2: Product's Underlying Attributes

Beth likes the look of the Passport as well as its qualities. Beth also eyes another SUV, the Dodge Durango. After examining its attributes, she decides that she can afford a more expensive car. As such, she moves the scrollbar representing the Durango's price from $25,000 to about $60,000.

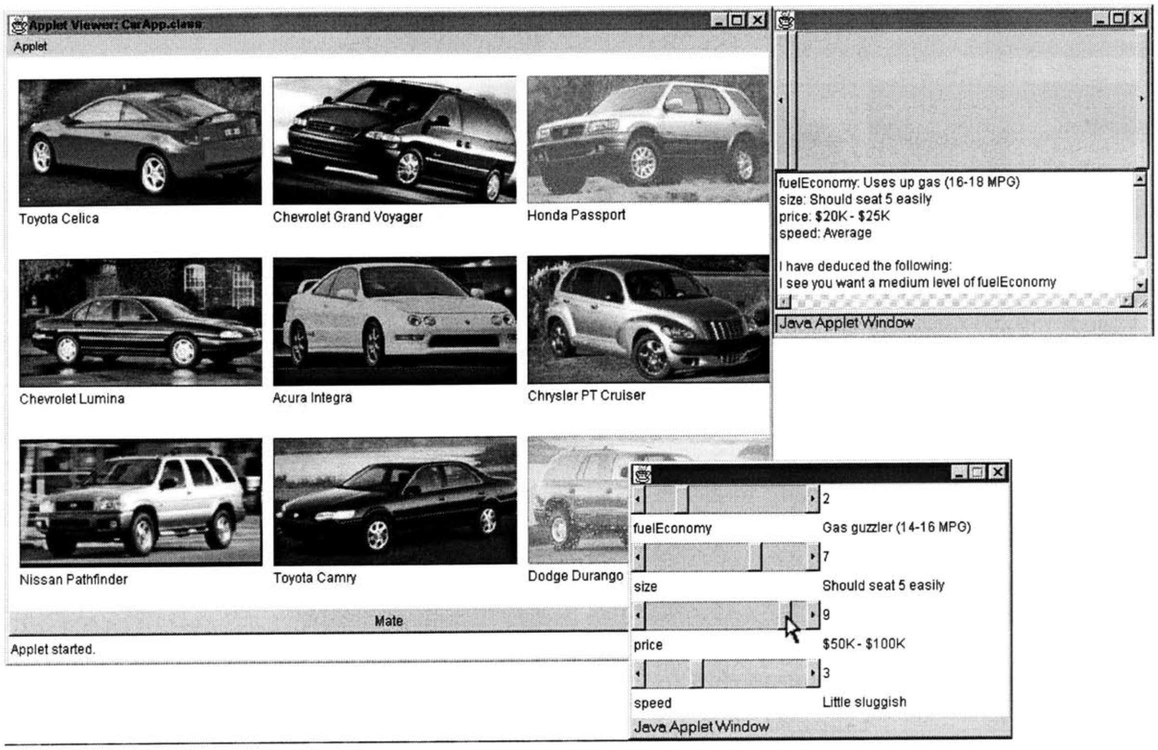

Figure 3: Genetically engineered product

Beth receives a third generation of cars, consisting primarily of SUVs with a few minivans and large family cars. Beth is impressed with the rather accurate profile suggesting she is less concerned with fuel economy as she is with a large car in her price range. One of the choices, an extra-large van is distasteful to Beth, so she crosses it out by selecting it, then pressing delete.

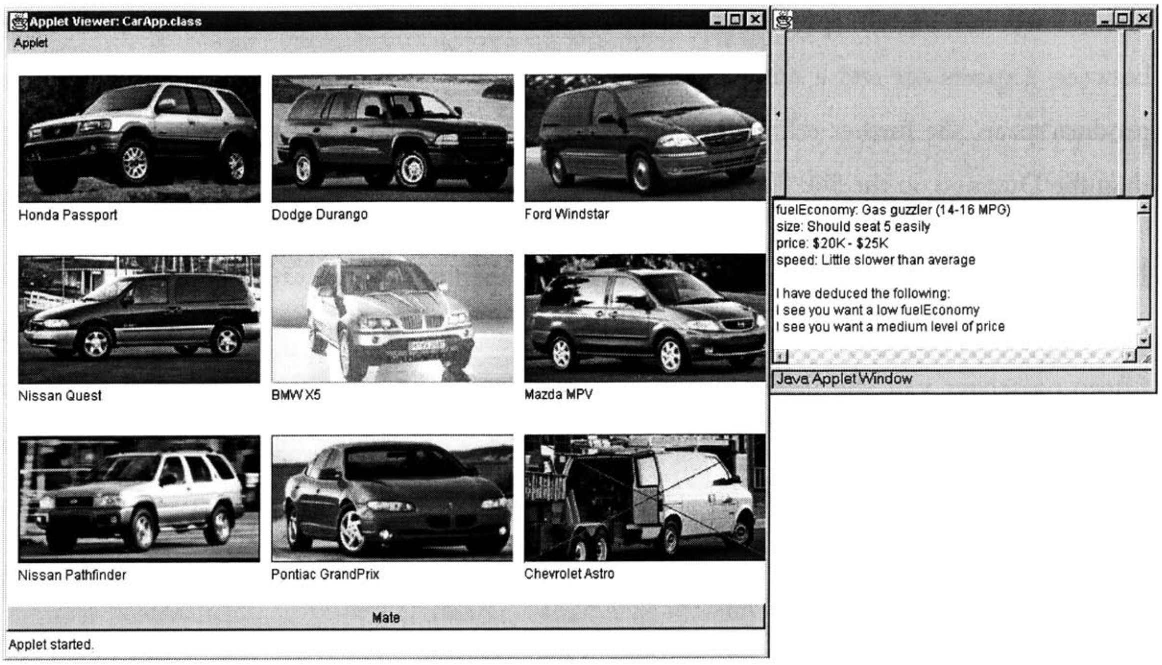

Figure 4: Negative Preference

After deleting the van, Beth examines the remaining cars. She falls in love with the BMW X5 and has finished her product search with her ideal car.

From this scenario, several salient points are observed. Beth did not initially know for what she was searching. She could, however, point to paragons epitomizing the traits she desired. The Auto-ficial Life system inferred her preferences for a sporty family car. Using examples, Beth was empowered to express imprecise, relative preferences.

Beth initially resigned herself to purchasing a minivan, yet she was excited when shown examples of Sports-Utility Vehicles, or SUVs. She did not need to previously know about them to use the Auto-ficial Life interface, yet she was able to learn about them. Serendipitous exploration of the product space is an important feature of Auto-ficial Life.

By examining the underlying characteristics of the Honda Passport and the Dodge Durango, Beth learned about the attributes that characterized cars and how these cars rated in each attribute.

She refined her search specification by using Auto-ficial Life. She started with "a cross between a sports car and a minivan" and then "an SUV". Using her new knowledge of the product space, she further refined her specification, to "I want an SUV a little more expensive than the Durango, in the $60,000 range."

By expressing her desire for a more expensive SUV than the Dodge Durango, Beth expressed control over the application. A second example of expressing control is her crossing out the fifteen-passenger van she disliked. This ability to control is similar to a physical shopping scenario where Beth maintains complete control. She controls the navigation from one set of products to another and can direct a salesperson.

Chapter 3

CURRENT STATE OF THE ART

Browsing, Searching, and Recommendation Systems, oh my!

Current web-based product-brokering systems fall into one of three categories, searching, browsing, or recommendation system, each is limited by its underlying assumption's regarding the users' specification of what they are looking for.

Searching

Virtually every product-brokering web page has a search button labeled with a magnifying glass. The user enters a specific description of the product they are looking for and the search engine returns a rank-ordered list. The search engine compares the text description with its indexed list and the closest matches are returned. Search systems are very efficient and return the exact match provided that users exactly specify their desires through the text field. Search engines have an underlying requirement that the consumer precisely specify his desired product in a textual format.

The user must use the same language as that used to index the product. If the user wants a car with manual transmission, but the product is indexed with either standard transmission or stick shift (both synonyms for manual transmission), the search engine will not find it. Modern search engines use synonyms, but this approach still has semantic limitations. For example, white vinegar has excellent cleaning properties and is the recommended antidote for stained clothing. However, it is always indexed as a food, never a cleaning supply.

Example: Buy.com

Buy.com uses a search engine

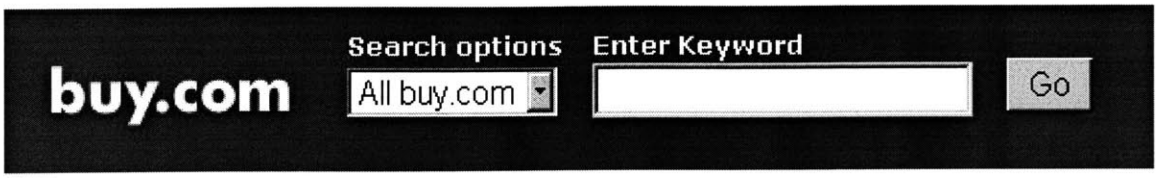

Figure 5: Buy.com Search Engine

Browsing

Browsing-based product brokering systems are organized like a catalog. The user explores all products within a product subspace. These systems lack intelligence to direct users to products they would like.

Ontologies facilitate browsing by reducing the examined subspace, but provide further limitations on the characterization of products searched. The ontology is a categorization of all the products in the space. Since the entire product space is large, the user first navigates through a hierarchical ontology to ascertain the desired type of product. If the product space has n products and the user navigates the ontology to m levels deep (typically two to five levels in an e-commerce setting), each subspace is roughly of size $n/2^m$. Thus, the ontology greatly reduces the size of the examined subspace. The ontology is constructed so that products in the same subspace will have similar characteristics. However, by following the ontology, the user has to have some knowledge of the space a priori. Additionally, they are limited to browsing by the way the ontology is set up and have less control over the navigation of the product space.

Example: CarOrder.com

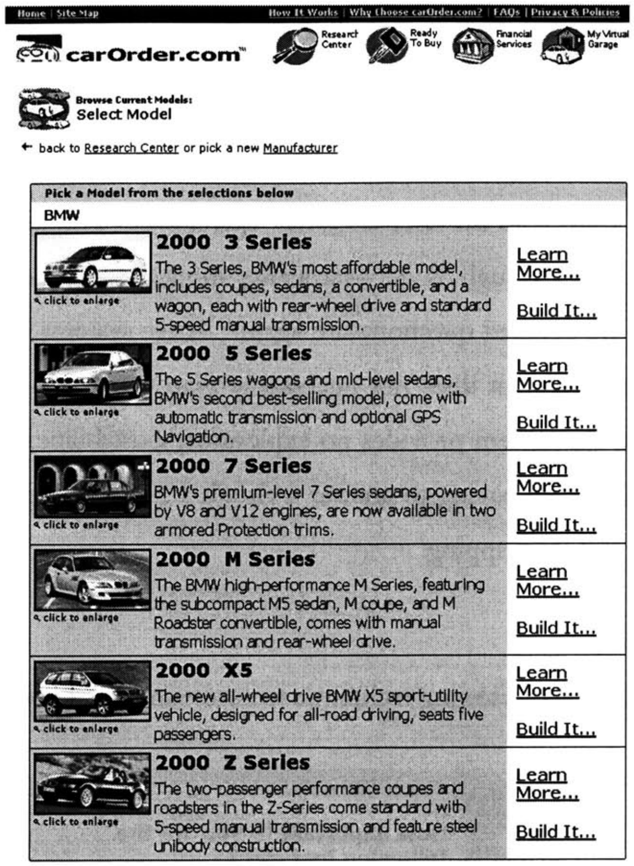

Figure 6: CarOrder.com's Browsing system

Recommendation systems

Recommendation systems take an initial set of examples and produce a set of other products in which the user may be interested. Recommendation systems currently fall into one of two categories, constraint filtering or collaborative filtering systems [12].

Constraint-filtering recommendation systems

Constraint-filtering recommendation systems ask the user several questions about the product she would like to purchase. The constraint satisfaction engine returns a set of products matching these criteria. Three examples of product-brokering systems that user a constraint-satisfaction engine are Tete-a-Tete, mySimon, and PersonaLogic. Constraint-filtering recommendation systems provide good exploitation properties. Once they ascertain the specific traits desired, they recommend further products with similar attributes.

Additionally, the user may be unable to answer many of the questions and become frustrated. In fact, many of these attributes are irrelevant to the user, and a better system would not burden the user with questions the user does not care about. Additionally, constraint-based systems are hard to iterate through as the user learns about the product space. To iterate, the user needs to fill out the detailed questionnaire again. These systems invoke hard constraints, not negotiable preferences, so it does not necessarily map to a user's approach to shopping. By forcing constraints, the system provides no exploratory possibilities, locking the user into a subspace. Last but not least, having to answer a detailed questionnaire before seeing a single product takes the fun out of shopping.

Example: PersonaLogic

PersonaLogic uses a deep-interviewing technique to ascertain the customer's criteria [11].

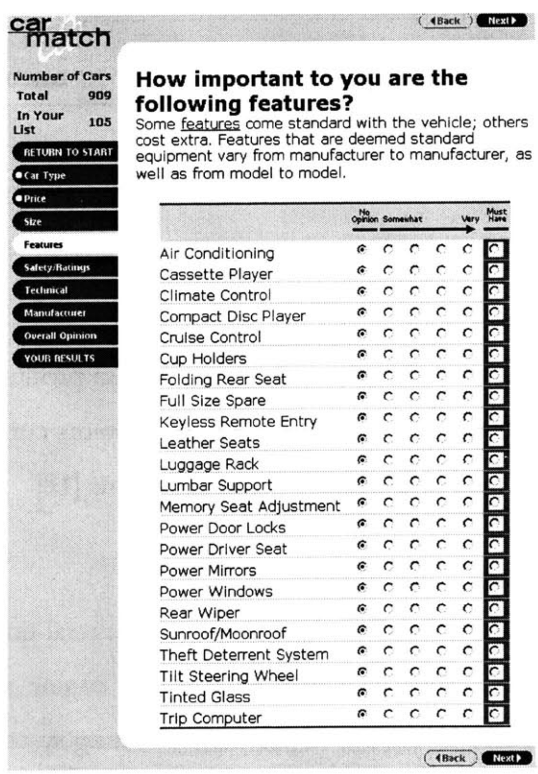

Figure 7: PersonaLogic Consraint-Based Filtering System

Collaborative-filtering recommendation systems

A "people like you liked" button on product-brokering websites epitomizes the second type of recommendation system, automatic collaborative filtering. By collecting information from many users regarding their likes, collaborative-filtering systems create groups of users with similar tastes. When a new user gives examples of his interests, the recommendation system matches him to a group of with similar preferences. Then, it recommends products that these others liked.

Automatic collaborative filtering systems have good exploitation properties by suggesting products similar to those in which a user has issued interest. This results as people's interests tend to cluster. These systems also provide good exploration characteristics. Recommendation systems occasionally suggest products that are very different from the user-specified examples, yet are likely to be interesting to the consumer as they were to other similar customers. However, the balance between exploration and exploitation depends entirely on the existing data in the system.

Example: Amazon

Amazon uses an automatic collaborative filtering engine for recommendations.

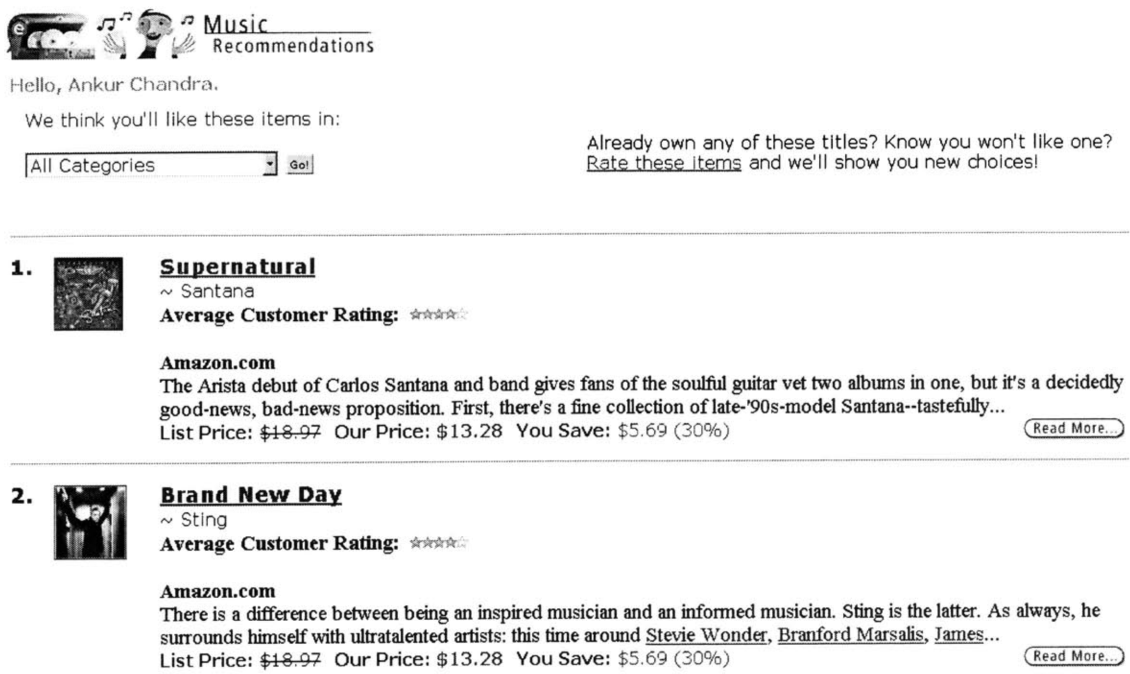

Figure 8: Amazon's Automatic Collaborative Filtering Recommendation System

Chapter 4

PRODUCT BROKERING AS MULTIDIMENSIONAL SEARCH

Looking for a broker in all the wrong places

Product Brokering Rephrased as Search

The user is performing a search, the end-result of which is the product to be purchased. This search eventuates in the consumer specifying the exact product to be purchased.

A search through the product space requires two properties. Exploitation involves hill-climbing, and suggesting a product similar to products already determined to be favorable. However, this behavior may result in a good product, but may miss an optimal one because it may get stuck at a local-maximum. To find a global maximum, the search algorithm will need to examine different subspaces as well, thus providing exploration.

Specification of Information Spectrum

Initially, users are unlikely to provide a detailed description of their desires. After exposure to the product space and the attributes that differentiate products, users ascertain their needs. Through iterative searching, users better refine the definition of their goal. This is inline with Shneiderman's four phases of information search, "formulation, initiation of action, review of results, and refinement" [21].

How well a problem can be specified is a spectrum. I refer to this line as the browse-search spectrum.

No specification / Browsing ◄────────────────► Full specification / Searching

Figure 9: Specification of Information Spectrum

Users tend to be located anywhere in this spectrum, not just at the endpoints. When a user is just browsing, she has little specification of what they like. She is looking around to get a better feel for the product space. Searching users, on the other hand, know exactly what they want and fully specify it with exact language. However, most users are not browsers or searchers, but fall in-between on the spectrum. Newer product-brokering applications will need to accommodate these users by addressing this entire specification spectrum.

The Andersen paper has an idea in common with this browse-search spectrum [6]. They separate their product space into a hierarchical ontology. They represent how specifically a user can express his desires by his level on the ontology. A browsing user is at the highest, or least-granular, level of the ontology. As a user more specifically categorizes his desires, he moves down to a more specific level of the product space ontology. In this system, the user's movement is limited to moving up and down a pre-determined ontology tree.

Current web-based product-brokering systems only handle part of the browse-search spectrum. As such, they are one dimension "too small". However current systems lack a time dimension as well. As a user learns about a product space, she moves from a position of less specification to one of more specification. Thus, the same user is located at different points at different times. New product brokering systems will need to address the user's specification as a function of two variables, user's initial specification and time.

Chapter 5

SHOPPING REQUIREMENTS

All shopping cart operators - please put on your helmets

Certain elements of shopping behavior influence the design of this system, primarily in the need for trust, control and fun.

Trust and control are central to product-brokering applications. As shopping involves parting a user with his money, a consumer must trust that the system works well and operates in his interests.

Like trust, user control is critical to a successful product-brokering system. Shopping is a user-driven process. "Evidence from empirical studies shows that users perform better and have higher subjective satisfaction when they can view and control the search [21]. Thus, control engenders a better shopping experience, and is a cornerstone of the Auto-ficial Life design.

Shopping requires an interesting combination of qualitative and quantitative aspects. While a user has requirements such as the number of people he needs to fit in his car, he also desires a sleek, aesthetically-pleasing automobile. Shopping needs to provide what the user needs in a desirable package. Current product-brokering websites cater towards either qualitative or quantitative factors. A qualitative-focused product-brokering website would request a consumer pick a car type, or body styling. A quantitative product-brokering application would first solicit the user's quantitative requirements. However, shopping is neither solely a quantitative nor just a qualitative experience. Shopping applications must mix these two aspects of shopping.

Chapter 6

USER INTERFACE

WYSIWYW (What you see is what you want)

The primary focus of this research project was to create a simple user-interface that allows users to easily navigate a multidimensional product space. The interface should be so simple that it does not require an instruction manual. The interface must be visual, allow for direct manipulation, jump straight into the product space, and provide useful feedback.

A visual, directly manipulated interface allows for effective navigation and search of the product space. A visual interface simplifies problem specification and comparison. "By pointing at visual representations of objects and actions, users can carry out tasks rapidly and can observe the results immediately" [21]. A simple, visual interface allows users to navigate the hyperspace quickly, much like they would walk through a store. Another advantage of using a visual, point-and-click interface is its elimination of language and ontology issues. Finally, keeping the interface visual retains many pleasurable aspects of browsing a mall.

Examining products upon entering a store is a selling point of brick-and-mortar commerce, and should be for E-Commerce. As such, a focus of the user-interface was to examine products initially. Users interact with products from the get-go with no instructions or questionnaires. As consumers need not fill out a questionnaire or read instructions regarding how to browse a store, this should be equally unnecessary in an online store.

Another design principle is Shneiderman's visual-information-seeking-mantra "Overview first, zoom and filter, then details on demand" [21]. The overview requires that the system begin at the left-end of the browse-search spectrum and place no initial assumptions on the user. Zooming and filtering requires that the user control the navigation of the hyperspace. To provide details on demand, the system only gives information if requested. Much like a chapter of an etiquette book, these systems avoid rudeness by not providing unsolicited information.

Feedback is necessary for fast visualization and effective user response. "Each action produces a comprehensible result in the task domain that is visible in the interface immediately" [22]. Whenever the user interacts with the system, it needs to let him know that this interaction has been noted. Similarly, the user should be able to ascertain the system's state with a quick glance.

Expressing preferences for and against products

Auto-ficial Life uses a comparative interface to assist user navigation. The ability to compare different products to one another is necessary as people are poor at analyzing in isolation. "People tend to perceive the world using both local detail and global context; Yet we rely on global context for orientation and to understand local detail." [5]. By examining the local detail of a product with the global context, a consumer gains a better understanding of the product.

A user selects a product by clicking on it, a natural behavior to illustrate interest. For feedback, Auto-ficial Life lightens the product's image, as is traditionally used to identify a selected image.

Ford Mustang Ford Mustang

Figure 10: Unselected and Selected Product

A strong negative product response should be a simple, one-step process. As such the user can either alt or ctrl-click the product. To provide consistency, users alternatively select the product and press delete.

Figure 11: Crossed-out product

Underlying product attributes

Consumers learn about a product by explicitly requesting additional information. This is patterned after a windowing environment, right-clicking the product brings up its attributes.

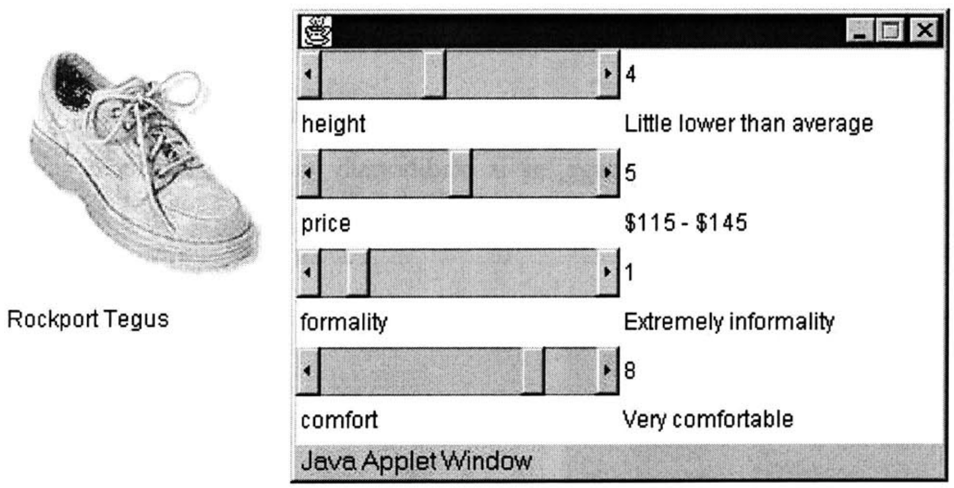

Figure 12: Underlying Product Attribute Values

By examining this information, the user learns about the product space. She discovers which attributes make up the space, and how this particular product fits.

Auto-ficial Life visualizes the attributes by expressing each attribute's domain by a line [21]. The scrollbar represents this line, and its position expresses the attribute value. A secondary effect of the scrollbar is the user's ability to edit the trait in a process referred to as genetic engineering.

Genetic Engineering

Genetic engineering, or the modifying of product attributes, allows users to direct navigation. A user may like a convertible like a Mazda Miata, but want something a little larger. When she pops up the Miata's underlying attributes, she can edit the size to ensure that future generations will be alike the Miata but larger.

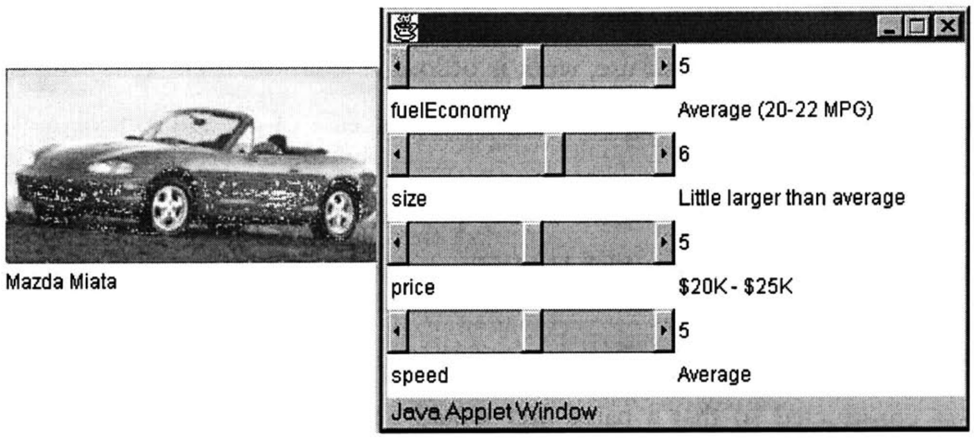

Figure 13: Genetically Engineered Product

Visual feedback needs to handle that a genetically-engineered product is similar to the original, yet its attributes have been modified. A label listing the edited attributes is placed above the picture. Thus, the product image remains the same, with an identifiable tag that it has been modified.

History

The history component is another method of user-controlled navigation. One desire of Auto-ficial Life users was to return to a previous generation they liked. The history component allows them to return to this "golden" generation and observe what would have happened if they had made a different decision. Thus, unlike life, they are given a second chance to repair previous mistakes. The history component is patterned after a web-browser. Users navigate using "back" and "forward" buttons to represent moving forward and backward a generation.

Design Tradeoffs

With navigability requirements such as ease-of-use, low cognitive overhead, and direct manipulation, and shopping requirements involving trust and control, the two types of requirements conflicted requiring design decisions.

Automation versus Control

The fundamental tradeoff, automation versus user control, provided the central conflict in the application design. To facilitate ease-of-use, work is offloaded from the user. However, this also takes away the user's control, creating the tradeoff between ease-of-use and control.

This automation-control tradeoff is inherent in any product brokering shopping agent, an oxymoron of sorts. Shopping requires a high level of control while the agent requires ease-of-use and task delegation.

The system was constructed so that a naïve user would not be distracted by complicated features, yet an advanced user can easily get to the tools he needs. This concept is also called

> *level-structuring. Novices can be taught a minimal subset of objects and actions with which to get started. They are most likely to make correct choices when they have only a few options and are protected from making mistakes – when they are given a training-wheels interface. After gaining confidence from hands-on experience, these users can progress to ever-greater levels of task concepts and the accompanying interface concepts* [21]

Using level-structuring, ease-of-use is a default. Advanced features are optional and explicitly requested.

A user always has the final say. "The user's inability to bypass the agent can cause her to feel out of control" [25]. Again, control is always possible.

An example of the automation versus control tradeoff occurred with a suggestion to have Auto-ficial Life automatically mate after a second product is selected. However, automatic mating would distract if not annoy the user as it would usurp authority to navigate. Although the lack of automated mating forced additional work on the user, this design decision ultimately gave the user more control over the shopping process.

Simple GUI versus functionality

To best assist a user's hyperspace navigation, a question arises whether it is more efficient to have more tools or a less cluttered interface. This question also arises in the user-interface of handheld devices, as screen real-estate and user's time is at a premium. The issue was how much product information to display. One idea was to place all product traits visually underneath the product's image. This gives the user more information than just the visual stimulus. However, it may be distracting, particularly for users who do not care or do not understand certain attributes. When browsing though a store, people walk down an aisle visually inspecting the merchandise. Only when they find a product they wish to examine further do they pick it up and ascertain its attributes carefully. The behavior of Auto-ficial Life acts in a similar manner. Namely, a user interested in learning about the underlying product space does so by explicitly requesting more information via a right-click. Thus, they quickly scan through a space visually, and when they want more information, they "pick up the product and examine it" with a right-click. This behavior also satisfies Shneiderman's information-seeking mantra regarding details on demand.

Hiding the attribute information augmented ease-of-use and navigation speed as potentially distracting visual clutter was removed. Secondly, users who do not care about the underlying attributes are not burdened with excess information. Additionally, cognitive overhead is reduced. Product attribute information is quantitative while the image is qualitative. Having to handle two separate modes of comparison simultaneously would incur a significant cognitive cost.

Not showing the attributes on the primary screen also solved a scalability issue. As products are described by more attributes, it becomes increasingly difficult to visually depict them in such a limited space.

Consistency versus functionality

Another tradeoff was consistency versus functionality. Consistency of interface is the first of Shneiderman's eight golden rules of interface design [21]. While abusing notation or combining two modes of operation has functional advantages, doing so often breaks the interface's consistency.

"Genetic engineering" a product by editing its attribute values adds a second mode of operation under the mating interface. Should the user just be allowed to select one genetically-engineered product and continue? While this provides good functionality, the user is used to selecting two products. To ensure a consistent interface, the user must select a second product with which to mate. While this may not be as functional nor easy-to-use, it maintains a consistent interface.

A second consistency issue occurs if the user likes more or less than two products. While being forced to choose two products may not be as functional as a more generic interface, it provides a consistent interface and a clearer definition of mating.

Chapter 7

BACK-END GENETIC ALGORITHM

Point-and-click Reproduction

Background

Genetic algorithms were created when John Holland realized that biological systems were more advanced than any made by man. As such, he patterned the theory of genetic programming after the way that he feels biological creatures evolve.

A genetic algorithm works as follows: it starts with an initial population of individuals, each with an underlying genome representing its characteristics. A fitness function takes an individual's genotype and returns a value representing its goodness. This function is applied to all individuals in the population, and the fittest are selected to reproduce. Mating combines the genetic traits of the selected individuals to produce offspring. The less-fit individuals are replaced with offspring by a replacement policy [13].

Genetic algorithms have good multidimensional search characteristics. As they mate the more fit individuals, the algorithm proceeds towards a solution by exploitation, or hill-climbing. One problem with just hill-climbing is the possibility to get stuck at a local maximum. It is possible that the hill being climbed is not the largest on a global perspective. As such, good search algorithms allow for exploration, the ability to visit other possible hills. Genetic algorithms provide exploration through mutation of the offspring's genome. By providing good exploitation and exploration characteristics, genetic algorithms quickly converge on an optimal solution.

Implementation

Genome

The underlying genome consists of a product's attributes. For example, a car's genome includes fuel economy and safety rating. These ratings are converted into an integral representation.

Initial Population

The initial population is a random selection of products from the multidimensional product space.

Fitness Function

Instead of a pre-specified fitness function, this genetic algorithm allows the user to specify which individuals are desirable. In this respect, it is similar to the user-specified fitness function in the Sims and Dawkins work [23][10]. Unlike those works, users of Auto-ficial Life are also empowered to de-select individuals from the population by crossing them out.

Selection Function

The selection function is the two individuals the user selects.

Mating

The selected product's traits are combined using a modified crossover technique. For every product attribute, an offspring's value either comes from one of the parents, each with a fifty percent chance. Additionally, mutations will cause occasional genes to be set to a random value in the attribute domain. Mutations occur at random with a fixed probability for all genes.

Replacement Strategy

All products are replaced by a new generation of offspring except for the two parents, who remain.

GA for selection, not evolution

According to traditional genetic algorithm literature, the Auto-ficial Life back-end genetic algorithm is complete as it results in an evolved population of children. However, since the purpose of this algorithm is not to create new products, but to navigate existing ones, an added step is necessary.

The Auto-ficial Life algorithm maps newly created product offspring back to existing products. It uses a linear distance metric to map the new offspring to the one nearest it in the product hyperspace. This can alternatively be seen as a mapping of the product's genotype, or genetic material, to a phenotype, or physical appearance. Thus, instead of evolving new products, this algorithm selects from existing products. I refer to this behavior as a genetic algorithm for selection rather than one for evolution.

Chapter 8

MACHINE LEARNING

Dues ex Machina

To adapt to the user's changing search specification, the program needs to learn about the user. By observing the user's interactions and decisions, the program develops a model of the user and her preferences. An implicit profile produces much information with no explicit user instruction.

Auto-ficial Life endeavors to ascertain the user's ideal combination of product traits. In the economic literature, the user is defined to have a utility function, a measure of happiness. This function is dependent on the underlying attributes of the products. By observing the user decide between several possible allocations of these attributes, the program maps out the user's multidimensional utility function in a process akin to Multi-Attribute Utility Theory.

This learning behavior is also similar to conjoint analysis, where preferences are detected after the user rates several products. Both conjoint analysis and the Auto-ficial Life machine learning algorithm use a decompositional technique. Users respond to the products, not the underlying attributes, and the importance of the attributes is inferred [20]. Since the Auto-ficial Life's dialog is limited to selecting or crossing products out, this learning algorithm differs from conjoint analysis.

Preference Detection

By examining the user's history, Auto-ficial Life uses a statistical pattern recognition metric to identify important attributes and develop a user model. Auto-ficial Life detects two types of user preferences. The first is an exact preference

Exact Preference

An example of an exact preference is a user needing a car with exactly enough room for five people, nothing bigger or nothing smaller.

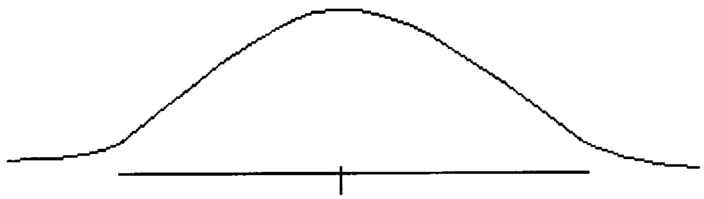

Figure 14: Exact Preference

This preference is modeled by a Gaussian random variable. Auto-ficial Life identifies the mean and variance of this variable's distribution. If the attribute's values follow this pattern, the system deduces an exact preference on this attribute's value. The determining criterion is if the variable's sample variance falls below a threshold.

One-Sided Preference

A second type of preference is a one-sided preference.

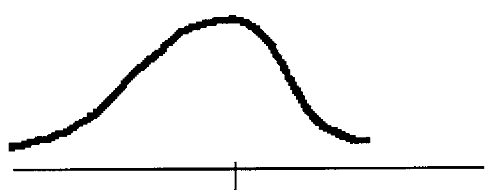

Figure 15: One-sided Preference

This preference is an upper or lower bound on an attribute value. For example, the user may have an upper price point on how much he can spend. As long as the product is less expensive than this value, he will consider other attributes. An example of a lower bound may be a minimum comfort level on a shoe required by the consumer. One-sided preferences are more difficult to detect, as they do not follow the symmetric Gaussian bell-shaped curve.

Auto-ficial Life examines the minimum and maximum values for this attribute. If either is significantly different than the minimum and maximum of the attribute value domain, the system concludes that this preference is one-sided.

Revealed Preferences

Revealed preferences also assist the machine-learning algorithm in developing a user model. If a user has a choice between two products, the chosen product is revealed to be preferred to the other [17]. Though a simple concept, chosen products likely have certain characteristics in common that the other members of the population do not possess. Thus, the user implies proclivity for these differences.

In practice, revealed preferences are not as useful as initially surmised. The population is not a random sampling, but chosen to have similar traits by the genetic algorithm. As a result there is less heterogeneity and difference between products. However, if a significant difference is found, it is useful to the learning algorithm.

Explicit negative semantics

A third avenue of learning is an explicit negative preference. Crossing-out a product has two semantically plausible meanings. The user may not like the specific product, even if she likes its underlying attributes. Another meaning could be that the user does not like the product because of its underlying attributes.

Auto-ficial Life detects which of these meanings is intended through revealed preference. If the crossed-out product has similar attributes to the selected products, the user likes the attributes, just not the individual product. For example, the user may be given a choice between a Chevrolet Corvette, a Chevrolet Camaro, and a Pontiac Firebird.

Chevrolet Camaro   Chevrolet Corvette   Pontiac Firebird

Figure 16: Revealed Preference Similar Products

These three cars have the same engine and similar attributes. However, if the user kills the Corvette, the user just dislikes the Corvette, not its attributes. As their attributes are all similar, it is unlikely that the user quantitatively dislikes the Corvette, rather she qualitatively dislikes the individual product.

A second case compares a product that is different than the others.

Chevrolet Camaro Nissan Quest Pontiac Firebird

Figure 17: Revealed Prefrences Dissimilar Products

However, if the user had to choose between the Camaro, Firebird, and a minivan, and the minivan was crossed-out, it is likely that the user does not like the minivan's underlying attributes. The minivan's quantitative attributes significantly differ from its brethren, so it is both qualitatively and quantitatively selected against.

Browse-Search Spectrum

Knowing the user's current position on the browse-search spectrum enables the system to tailor its behavior accordingly. To detect if a user is on the more-specific side of the spectrum, Auto-ficial Life verifies that the user's product choices are similar in the important attributes. A searching user will want fewer choices, each with more detailed information. To provide more information, the values of the most important traits could be depicted below the product. Although this breaks consistency and involves quantitative and qualitative information being handled simultaneously, searching users likely spend more time with their decision. Here, ease-of-use means offering information the searching user is likely to request. If the user has little specification, in other words is browsing, the user will want many choices, with little information on each choice.

I had difficulty developing a good metric for detecting the user's position on the browse/search spectrum. Initially, I had figured that it would be easy to tell if a user was browsing or searching. I planned to use a weighted average of the attribute differences to ascertain where the user is on the browse-search spectrum. The weights measure the attributes' importance. However, as I used a variance metric for ascertaining attribute importance and also for the attribute's variance, I was short a degree of freedom. Also, I had difficulty creating a metric that would be independent of number of products in the user's history, and the number of attributes. Additionally, the metric should more highly weigh recent product choices as they are of greater importance than choices in a user's past.

Domain-Specific Questions

Domain-specific questions provide personalization, further engendering an environment of trust and user model accuracy. By possessing specific domain and user information, the system can ask the user specific questions to further refine its user model. For example, the system may not be sure of the user's maximum price point. By explicitly asking the user, the system resolves this ambiguity. Since these questions are directly related to the user and her search, she will be more likely to answer. Also, just by observing that the system asks specific questions, the consumer sees that the system is personalizing itself to the user. This engenders further trust.

Chapter 9

INTERACTIONS OF SUBSYSTEMS

The pinky bone's connected to the foot bone, and the …

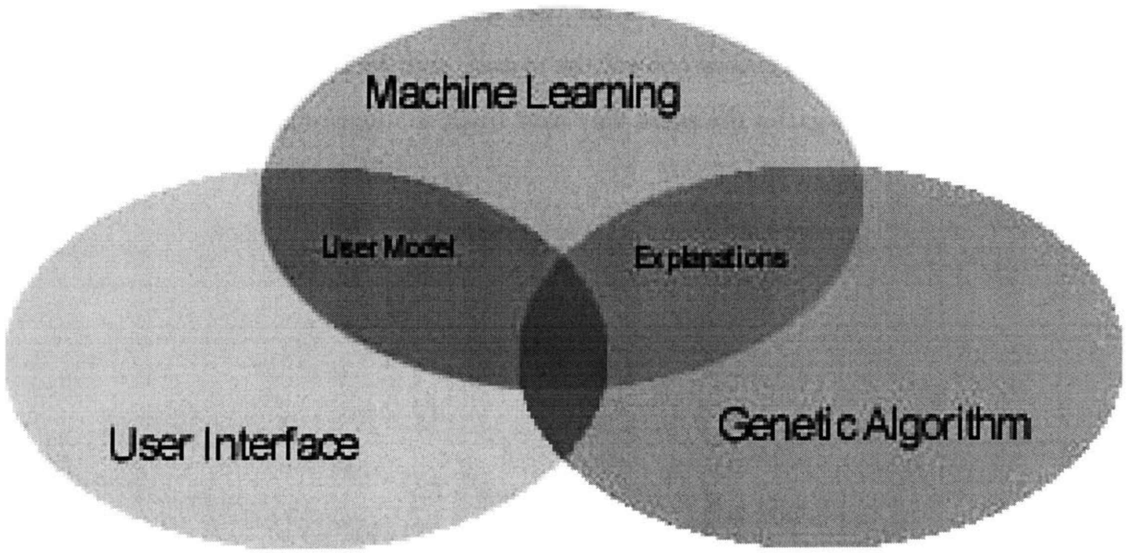

Figure 18: Subsystem Interaction

Machine Learning + Genetic Algorithms = Explanations

Soft computing, which consists of genetic algorithms, fuzzy programming and neural nets cannot explain their behavior. They work by manipulating numbers and weights to arrive at an optimal solution. However, by pairing such a mechanism with a machine-learning algorithm, it is possible to provide explanations as to the system's functioning.

Explanations allow the user to ascertain why she is given a set of products. Perhaps the program has erroneously deduced that the user wants a big car, when in fact the user wants a smaller sedan. By accessing this profile, she can detect why the program is offering her strange and undesired product choices. By editing this user model, she repairs this machine-learning mistake.

User Interface + Machine Learning = User Model

Exposing the user model to the user allows the user to better understand the system, resulting in more trust and control. "An agent which can show and possibly explain its model of the user is more likely to be trusted than one which hides this model." [25]. Exposing the user model also puts the user in control. As the user makes further selections and her profile is updated, she sees how her choices control the system and the resulting agent behavior. The user can correct possible gaffes the agent may have made in interpreting user behavior.

Chapter 10

RELATED WORK

"Everything you can do, I can do better"–Nike ad

Genetic Algorithms systems

The impetus for this paper came from two sources, both based in genetic algorithm literature. Karl Sims created a system of images with various visual qualities. By selecting two desirable pictures from a population of twenty, users receive a new population of drawings with similar characteristics [23]. Likewise, Richard Dawkin's *Blind Watchmaker* involved selecting two simple line drawings, and having new drawings generated from the two [10]. Both systems involve a user-specified fitness function and a directly-manipulated interface.

Product Brokering Systems

The most direct comparison would be the MDS-INTERACTIVE paper released at CHI 2000. MDS-I is a two-dimensional visualization of the multidimensional product space. The user clicks on a product for more information or he clicks between products to examine the subspace. The MDS-I framework has been implemented in the rollerblade, whiskey, and color- choosing product domains [14] [15].

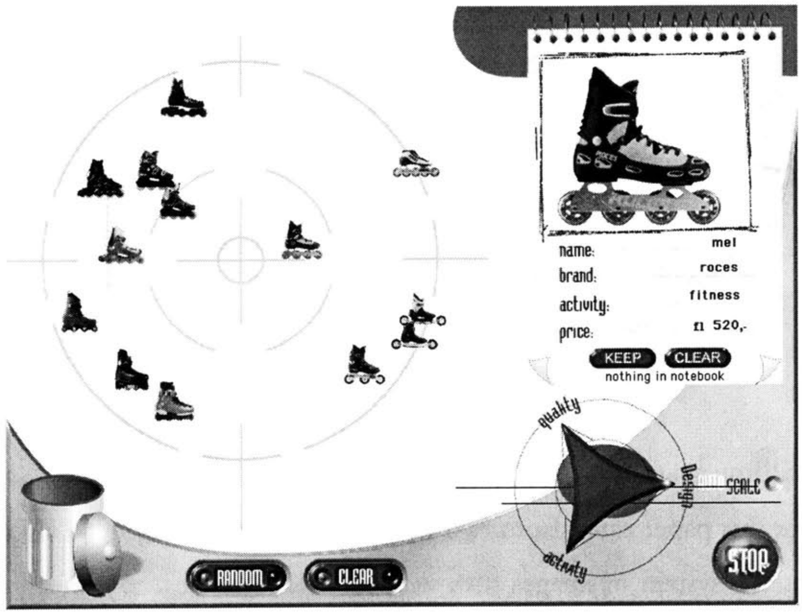

Figure 19: MDS-I for Rollerblades

The further a product is from another, the more dissimilar are the two items. The space of n dimensions is compressed to two using a technique called dimensional scaling. The user weights attributes so traits differing in an important attribute are further away than those that differ in a less important attribute.

As most of the screen is white space, or space between products, there is less focus on the products and more on where they fit in hyperspace.

One advantage to MDS-I over Auto-ficial Life is the ability to navigate by clicking in the space between two objects. The only way to get similar behavior in Auto-ficial Life is to genetically engineer a product, and mate it with another product. However, MDS-I's operation has usability issues. Since multiple dimensions are compressed into two, a two-dimensional position represents several multidimensional positions. Depending on the interpreted meaning, MDS-I may or may not explore the subspace the user desires. Further comparison of the two systems will ascertain each system's navigation semantics.

The user is given more control by personally handling factor weightings. Noisy data and inconsistencies that plague conjoint analysis systems are eliminated.

MDS-I could be placed as a user-interface atop Auto-ficial Life to leverage the exploration capabilities of its genetic algorithm and the machine learning to modify the attribute weights, as well as handling one-sided preferences.

The second related product-brokering system is the virtual fishbowl intending to cross-sell, or sell products for which the user has latent desire. While this system emulates a mall, Auto-ficial Life emulates a single store. Their product space is partitioned by an ontology. Although users move up and down the ontological hierarchy, they cannot explore easily, nor can they control the navigation [6].

PersonaLogic is a constraint-filtering, product-brokering system and while it works well to handle users with hard constraints browse, it suffers from the same difficulties that plague constraint-filtering recommendation systems.

Chapter 11

USER EXPERIENCE

I've never done that before

I have made several claims in this thesis, namely that Auto-ficial Life provides some of the best features of current web mechanisms as well as some they do not have. This system assists users who can poorly formulate their requirements. I suggest that Auto-ficial Life helps the user learn about a space as they move through it. I assume that the system creates an accurate user model. Plus, I suggest that the explorative versus exploitative nature of my genetic-algorithm backend provides the user with utility maximization. Another claim is that the clean GUI and easy interface provide a lightning-quick method to examine a large product space. Auto-ficial Life provides enjoyment, a key ingredient of a successful shopping experience. Finally, Auto-ficial Life is a more effective product-brokering system than others on the Web.

User testing is necessary to substantiate these claims in the areas of enjoyment, effectiveness, handling different requirements, learning about a product space and usefulness.

Unfortunately, there was not enough time to conduct a formal user study properly. However, informal user testing produced a number of interesting results.

User experience was overall positive. Users were able to quickly navigate the product space and they found the system intuitive. The time between generations was on the order of seconds, and users could analyze hundreds of products within minutes. Most users were surprised that no similar technology existed on the Web. However, they were occasionally confused as to how many products to select, and whether or not to press the mate button. Better feedback such as disabling the mate button until two products were selected and beeping when the user attempts to select a third product helped guide users.

Users selected a few products in which they were not interested. This "irrational" behavior violated the basic assumption that users express desires through selection. However, since they are doing so for fun, this behavior satisfies the need for enjoyment. A by-product of this enjoyment is that the machine-learning algorithm becomes confused and cannot make guarantees about the user model.

Another issue was that users did not know that they could cross-out a product by alt or ctrl-clicking it. One user ascertained this behavior by selecting a product and pressing delete. A more intuitive interface is necessary for users to know how to cross-out a product.

One unexpected result is that a couple users questioned how the system works. In this day and age, the lines between shopping and advertising are blurred. Users wondered if the recommended products were suggested were in the buyer's or seller's interest. They were somewhat reassured upon observing the user model and underlying product attributes. Formal user testing will need to assure users that the recommendations are objective and in the buyer's interest.

In conducting a formal user study, I would compare Auto-ficial life to other systems on the Web. Users would be given a scenario with imprecise preferences and asked to find their desired car. Their observations and statistics regarding the products analyzed would be recorded. Finally, they would be given a survey to compare Auto-ficial Life with the other system. From this survey, it would be possible to ascertain whether or not Auto-ficial Life is more useful and enjoyable of a product-brokering system.

The scenarios are listed in the appendix.

Chapter 12

CONCLUSIONS

Everything I needed to know, I learned at the Media Lab

This research project is split up into three different areas, the user-interface, the genetic algorithm back-end, and machine-learning. The areas are independent. In fact, each can be replaced wholesale without hardly affecting the other two components. This system does not require a genetic algorithm-based back end to provide the user-interface features such as quick comparison and learning about a product's attributes. In fact, the modular system makes it easy to replace the back-end with another. For example, the back-end could be replaced with an automatic collaborative filtering engine.

User Interface

Building the user interface revealed the importance of level-structuring of an application. An application's use should be simple so novices are not confused with advanced features. However, these tools should be readily available for an advanced user.

Also, I learned about the importance of feedback. The user needs to easily identify that his actions were recognized. Subtle visual and aural cues were necessary to direct the user. By disabling the mate button until the user selects two products, the user has no choice but to select products. Additionally, the user needs to be instructed that he cannot select more than two products. To provide this feedback, the system beeps if the user attempts to select more than two products. From a cue in Shneiderman's book regarding interface consistency, I allowed the user to select a product, and type delete as this is common behavior in windowing systems to cross-out the item [21]. This feedback also allows the user to quickly ascertain the system's state with a quick glance.

The underlying assumption that users select a product to express interest is not entirely accurate. During user tests, it was painfully obvious that users occasionally click on a product even if they are not at all interested.

Genetic Algorithm Back-End

The genetic algorithm back-end worked well in accelerating the user's multidimensional product search. Its combination of good exploration and exploitation properties allowed the user to reach a global maximum efficiently.

Using a genetic algorithm is intuitive and goes back to the standard, "what do you get if you cross a" joke. People are good at combining characteristics in their head, and understand this process easily as it is an inherent concept.

I learned that genetic algorithm metrics are hard. Determining the proper mutation probability and the mutant gene's value was difficult. Although I used trial and error to establish these values, they will need to be more robust to changes in the attribute domain and different users.

There were several different genetic algorithm-mating algorithms that could have been used. I discuss each as described in the Sims paper as follows [23].

The simplest, and most common mating strategy, is the one employed in Auto-ficial Life. The modified crossover algorithm randomly takes a gene from either of the parents to construct the child's gene. This mating strategy provides good mixing of the parent's attributes.

A second mating strategy involves weighting the parents' genes. A random percentage of one parent's gene is combined with one minus the percentage of the other parent's gene. Linear interpolation is when this probability is set to ½ and was the initial mating strategy employed. While this is a valid mating strategy, it provided poor results because it does not semantically map to mating products. If a user selects a sports car and a minivan, she is likely interested in a car with the speed of the sports car and the size of the minivan. However, linear interpolation produces cars with mediocre sports and size characteristics. Additionally, all offspring occur on a line equidistant between the parents. Another problem with this strategy is that upon iteration, the user ends up in the middle of the hyperspace. Although the user is not picking at random, in practice the user typically ends in the product space's median. This strategy was replaced with the modified crossover technique to provide better mating semantics.

The third technique is similar to the weighting strategy. However, the weight is established separately for each gene. This has the similar failings of the weighting strategy, though the offspring have more genetic diversity.

The fourth technique is crossover, which starts copying the genome from one parent, but randomly switches to the other parent, at which point the process repeats. Crossover works well for keeping related genes together. For example, if the genes for a hand and those to control the hand neighbor on the genome, they will likely stay together in the offspring. Auto-ficial Life makes no restrictions on the order or relation of the attributes to one another. As such, the crossover mating strategy has little practical value to this system.

One unexpected result is that users expressed concern about the system's objectivity and asked how the items were recommended. For example, Amazon's recommendation system was brought into question when it was revealed that publishers paid to have their books recommended [5].

> *"Is your agent neutral, biased, or merely weighted? And can you really tell the difference? Who really controls your agent, you, the designer, or the person feeding it information? How can you be sure that your agent has not been 'turned', being now a double agent."* [5]

Since Auto-ficial Life exposes the user model and provides explanations, the user can verify that recommended products have attributes in common with those selected. Thus, the system proves the objectivity of its exploitation behavior. However, since the explorative nature of the genetic algorithms is random, it is impossible to justify that the seller is not occasionally showing a product they want to sell. However, modifying the mutation algorithm may be used to place guarantees.

Machine Learning

The machine-learning algorithm worked well in detecting the user's preferences as seen in various scenario and user testing. However, when a user breaks from a consistent pattern, the system was confused. If the user is just fooling around, the system needs to be robust to this. However, if the user has made a serious context switch, the system needs to handle it.

Another issue is that the program occasionally ascertains preferences that are not intended. For example, sports cars have high speed at the expense of low fuel economy. If a user selects several sports cars, Auto-ficial Life deduces that the user wants high speed and low fuel economy. The low fuel economy is an undesired by-product of the negative relationship between the two variables. The application designer could express the desired direction of each attribute. However, certain users may want different values for an attribute. A second option is to have the application designer write domain-specific rules representing these relationships. The program would then deduce which real preferences from unintended by-products. This solution requires the application designer to imagine any possible user behavior. Additionally, the designer will invariably eliminate certain valid possibilities by over constraining an attribute. The best solution involves exposing an editable user model so the user can correct erroneous or by-product inferences. The designer focuses on delineating the product's attributes and the user gains control. This approach also decouples the machine-learning algorithm from the partitioning of the product space.

It is difficult to compare one-sided and two-sided preferences since they are measured on different metrics. Perhaps by using a better model of a one-sided preference's distribution, I can use a metric of how well the sample data matches the expected distribution to compare the two types of preferences.

Shopping

Shopping needs to be fun. Users are frustrated when they are consistently shown products they do not like. Additionally, users are annoyed when a system takes too much control or makes incontrovertible assumptions. One user had purchased a self-help book and a how-to-dress-for-business book from a major web retailer, and was chagrined to be recommended

"Play Like a Man, Win Like a Woman". However, he was even angrier that he could not correct the system's egregious mistake. At least in Auto-ficial Life, the user could express his distaste at having his gender misjudged by crossing out the book and avoiding future mishaps.

Initially, this program exposed the internal, integral representation of the product's genome. However, users do not respond well to a one-to-ten scale of product attribute values. As a result, I incorporated a mapping from these integers to real-world values. This was a huge improvement to the users' understanding of the product space, and how products fit.

As shopping is a mix of emotional wants and logical needs, it involves both sides of the brain and requires qualitative and quantitative characteristics. This combination turned out to be very important to shopping.

Shopping's qualitative and the quantitative aspects each provide different comparison and merging characteristics. Ironically, it is possible to handle both better by treating each as the other.

The qualitative aspects are represented by the product's visual image. Qualitative factors allow for easy visual comparison, yet they provide poor mating behavior. It is difficult to mate a product's qualitative aspects as these are often impossible to delineate. It is possible to account for some of these properties by enumerating related aspects on a subjective scale. For example, users can rate the product's trendiness or the average age of the product's typical owner. These integral characteristics would allow for mating of a product's qualitative aspects by treating them as quantifiable.

The quantitative aspects provide opposite properties. It is easy to merge quantitative factors yet it is difficult to compare them. However, quantitative comparison is easier with a better visualization interface. The MDS-I interface, for example, allows easy comparison of quantitative attributes.

Domain-independence

Auto-ficial Life worked well in both product domains, cars and men's shoes. Users easily navigated and learned about both product spaces. The system effectively ascertained important attributes in both domains, whether it was desired car size or shoe height. Auto-ficial Life is proven to extend to future domains.

Other mechanisms as heuristics

Although Auto-ficial Life provides many advantages over existing technology, other technologies can optimize Auto-ficial Life's operation. For example, automatic collaborative filtering provides better exploration characteristics as it detects exploratory behavior in similar people. Thus, there is no reason why Auto-ficial Life cannot be augmented with a collaborative filtering engine. This way, when it suggests exploratory choices, it can make a smart decision using an automatic collaborative filtering engine as a heuristic. Also, ontologies or domain-specific information could augment the system's functionality.

Final Commentary

Being able to partly or imprecisely specify desires is critical to new applications. Many concepts developed in this thesis extend beyond product brokering. In agent literature, "Users of an agent system should be able to describe their desired end result without needing to specify precise methods for achieving these results" [25]. Future "smart" applications will use context and implicit profiling along with a mechanism for users to imprecisely state what they want. The personalization elements are important too. Currently, an application looks the same to all users. In the future, applications will customize themselves to their users.

While Auto-ficial Life satisfies the needs of product brokers, it extends to looking for anything that cannot be specified precisely. As the prototype of Auto-ficial Life was completed around Valentine's Day, a suggestion was to create a dating service. It would be populated with pictures of eligible bachelors/bachelorettes, each with traits representing their education, and physical characteristics such as the color of their eyes, shape of their nose, and so on. Here, learning about the product space means that users ascertain their own preferences for what they find attractive in others. Thus, in addition to handling services as well as products, this dating service shows how Auto-ficial Life assists users in learning about themselves.

A more general search problem that also requires enjoyment is looking around the Web. There is a prototype of Auto-ficial Life that works like a web browser/ search engine. It shows thumbnails of news sites, each rated in story length, amount of editorial commentary, and number of pictures. Thus, users can efficiently navigate the web while the system provides good recommended sites while it learns about the user's interests.

These are new times that require new interfaces. 3D virtual shopping, and shopping agent personification are doomed to failure in the near term. They use a familiar physical-world scenario and bring it onto the Web. Three-dimensional shopping is similar to browsing a store and talking to an agent mimics conversing with a salesperson. As such, users are lulled into thinking that these applications have more functionality than they do. Users are annoyed and less trusting of the system when they learn that these applications are incapable of acting as instructed. As the web is a new medium, instead of wholesale porting of existing concepts, new systems should leverage the best of the web with the best of the physical world to provide next-generation product brokering.

Chapter 13

FUTURE WORK

"If only I could turn back time", Aqua

There is much more work that needs to be done in this field, certainly more that can be done furthering the work of this thesis.

User-Interface

There should be future work in extending the user's dialog with the program without breaking the simple user-interface. The user-interface currently supports selecting, selecting against, further information, and mating commands by the user. As the system gains functionality and handles new commands (purchase, for example), there will be further challenges to add functionality without sacrificing ease-of-use. The goal is to keep the system from requiring an instruction manual.

Another area of further study is the consistency-versus-functionality tradeoff in requiring two products to be mated. However, the user may only have one product they like, or more than two. If Auto-ficial Life were to support this behavior, there are questions as to its usability and consistency. However, because the back-end genetic algorithm uses a crossover technique, any change to the number selected can be easily handled. Further testing will ascertain the merits of this tradeoff.

The passive nature of a negative response should be better modeled. A user's disinterest in a product more likely results in inaction rather special attention. As such, the system needs to better model a negative reaction to a product. Users would not likely alt-click an object to express dislike. Perhaps a system more in tune with human behavior, such a biometric one with eye-tracking would be more useful in identifying disliked products. If the user is examining a product and the biometric sensors detect a negative response, it indicates that the user does not like the product. This appears to be a more efficient and effective method of negative feedback, though it would need to be tested.

Further research should consider better visualization tools to make the quantitative characteristics more easily comparable. One thing that the MDS-I system does well is its visualization of the space based on dimensional scaling. Perhaps additional visualization in Auto-ficial Life could better explain where products fit into the hyerspace. Also, knowledge of how populated a subspace is could be useful. For example, if the consumer is in a sparsely-populated region of the product space, perhaps his criteria is at odds. For example, there are not too many cars with high fuel economy and fast acceleration. Another possible visualization tool is Richard Chimera's Value Bars. Value Bars graphically depict the value of a product's attribute along a line. The longer the line segment, the larger the attribute's value. This allows the user to quickly and with low cognitive load, navigate through the subspace, and visualize how important attributes of products compare. Although this behavior is implicit in genetically engineering two products, it entails a larger cognitive load [9].

Another research area is providing additional tools for the user to control the hyperspace navigation. One such tool is sorting all products by a characteristic.

Currently, all product images are the same size, and thus judged to be equal possibilities. However, a suggestion is to make the more likely products bigger [5]. While this tradeoff will break the consistency of the interface, it utilizes the user's massive visual subsystem to suggest more information. Additionally, this technique could temper exploratory options that might otherwise be distracting.

When a user makes a strong movement from one side of the spectrum to the other, perhaps this has some significant semantic meaning. Further research will ascertain if this is an event that should be submitted to the user's approval. It is likely that the machine-learning algorithm will require user-assistance to ascertain the meaning of this jump.

To better handle level-structuring, the application can modify its user-interface depending on the sophistication level of the user. A user judged to be advanced would have easier access to the more advanced tools. The results from this idea behind the menus of Microsoft Office 2000 will tell whether breaking the interface's consistency is worth the functionality.

Shopping Aspects

Qualitative product attributes effects on this system need to be further analyzed. These include color, orientation as well as other uncharacterizable criteria. The orientation problem can be solved as in Boo.com, which uses several pictures of the product that can be manipulated by the user. This is similar to the technology underlying holograms. I have not researched the effects of these qualitative factors in shopping, but perhaps further research will elucidate this relationship.

Back-end Genetic Algorithm

Mutation

The mutation factor underlying the genetic algorithm will need to be made more powerful. Currently, the mutation probability is a fixed number for all attributes and all users. Ideally, this factor should be tailored to the individual based on the amount of exploration and exploitation she would like. For example, a user who enjoys more serendipity should have a higher mutation factor, while a more focused user should have a lower mutation probability. If the user has explored serendipitous (seemingly unrelated) products in the past, she is likely to enjoy more exploration in the future. .

Another change to the mutation factor should have it differ for each attribute. More important attributes should vary less, as the user cares more that they fall within a certain range. However, less important traits should have a higher mutation probability as this considers a more accurate subspace.

Attributes

Partitioning the attribute value domain is another open research question. With Auto-ficial Life, data was sampled and attribute value ranges created to separate products maximally. The effects of attribute partitioning and the domain's range on other the genetic algorithm and preference detection systems needs further study.

Converting the data into an internal representation was also an issue. For products that are not as well defined as cars, deducing attribute values is not trivial. For example, with shoes, personal judgment determined their comfort level. These new attribute-based product mechanisms will be dependent on an impartial third party such as Consumer Reports or surveys to report these subjective attribute values.

Non-integral attribute values, such as Booleans or multiple values need to be added to the system. Booleans are useful, as they ascertain whether or not a product has a certain property. A second useful data type is multi-valued. For example, an attribute can have multiple values, and the system would determine how likely is each value. This would need to be handled similarly to Open Sesame. The effects of adding new data types particularly in the genetic algorithm distance metric, and the learning and preference detection areas require further study.

Distance metrics mapping the offspring genotype to existing products should be further researched. A linear distance metric was used, but I had experimented with a quadratic difference metric. This did not seem to generate desired results as an unimportant attribute often kept the user from getting the products he desired. However, by weighting the difference by the attribute's importance, this can be alleviated. More work needs to be done to determine a good metric for mapping a evolved offspring to an existing product.

While Auto-ficial Life selects from existing products, perhaps a future version will actually generate new products to allow the ultimate in product customization.

Machine Learning

Further research will need to deal with typical human, irrational behavior affecting the machine learning algorithms. One issue seen in the conclusions is that users may click on a product even if they are not seriously interested. Perhaps, machine learning can detect the user's seriousness. Another possibility is to have the user rate his interest in a product, much like in conjoint analysis.

The machine-learning algorithm needs a better metric for representing one-sided preferences. A low variance is not the only indication of an important attribute. While I also examine the maximum and minimum values chosen by the user, I do not have a good metric to use this as well as the variance to identify one-sided preferences that are important to the user.

Auto-ficial Life is not as helpful to users who do not know what an attribute means. Although they learn where the product fits on this dimension, they may not realize what the dimension represents or its importance. Auto-ficial Life should be augmented to provide meta-information on the attributes. Perhaps the user can click on an attribute to bring up a detailed description.

Another area that can be researched is data-mining the attributes of products and habits of multiple users, and inferring further information. This would provide domain-specific rules or attribute rules such as automobile fuel economy and speed are inversely related. This information could be fed back into the program to enhance future behavior.

APPENDIX

Vestigality in action

User survey

Scenario #1: You are a single mother for four active children, aged 5 to 14. You have $22,000 to spend on a new car.

Scenario #2: You are a 25-year old ad executive in the city making about $55,000 a year. You are looking for a car that can be maneuvered around the city, parked easily, and one that will keep you safe.

Scenario #3: You are a rich 21 year old entrepreneur who had dropped out of college to start your multi-billion dollar technology company, Macrohard. The sky's the limit.

BIBLIOGRAPHY

[1] Ahlberg, Christopher, and Ben Shneiderman. "Visual Information Seeking using the FilmFinder". *Conference Companion CHI '94.* Boston, MA: MIT, 1998. 433.

[2] Amazon.com URL. http://www.amazon.com.

[3] Bartram, Lyn, et al. "The Continuous Zoom: A Constrained Fisheye Technique for Viewing and Navigating Large Information Spaces". *UIST '95.* 207-215.

[4] Boo.com URL. http://www.boo.com.

[5] Brown, John Seely, and Paul Daguid. "The Social Life of Information". http://www.firstmonday.dk/issues/issue5_4/brown_chapter2.html.

[6] Bryan, Doug, and Anatole Gershman. "Opportunistic Exploration of Large Consumer Product Spaces". *Proceedings of the ACM Conference on E-Commerce EC '99.* ACM Press, 1999. 41-47.

[7] Buy.com URL. http://www.buy.com.

[8] CarOrder.com URL. http://www.carorder.com.

[9] Chimera, Richard. "Value Bars: An Information Visualization and Navigation Tool for Multi-Attribute Listings". [*CHI '92.* ACM Press, 1992 – proper citation]. 293-4.

[10] Dawkins, Richard. *The Blind Watchmaker.* WW Norton & Co, 1986.

[11] Guttman, Robert H. *Merchant Differentiation through Integrative Negotiation in Agent-mediated Electronic Commerce.* SM Thesis. MIT Media Lab, MA: MIT, 1998.

[12] Guttman, Robert H, et al. "Agent-mediated Electronic-Commerce: A Survey" *Knowledge Engineering Review* June 1998. http://ecommerce.media.mit.edu/papers/ker98.pdf.

[13] Heitkotter, J, and D. Beasley. *The Hitch-Hiker's Guide to Evolutionary Computation.* http://www.wi.leidenuniv.nl/~gusz/Flying_Circus

[14] Keller, Janus. "MDS-I for 1-1 E-commerce: A position paper". *CHI 2000 Workshop on 1-1 E-commerce.* http://www.zurich.ibm.com/~mrs/chi2000/contributions/keller.pdf.

[15] MDS-Interactive URL. http://www.io.tudelft.nl/research/IDEATE/mdsi/mdsi.html.

[16] mySimon URL. http://www.mySimon.com.

[17]Nicholson, Walter. *Microeconomic Theory: Basic Principles and Extensions*, 6th ed. New York, NY: Dryden Press, 1995.

[18]PersonaLogic URL. http://www.personalogic.com.

[19]Schafer, J. Ben, Josheph Konstan, and John Reidl. "Recommender Systems in E-Commerce". *Proceedings of the ACM Conference on E-Commerce EC '99*. ACM Press, 1999. 158-166.

[20]Sclove, Stanley. "Notes to accompany HAIR et al 4th ed: Chapter 10: Conjoint Analysis". http://www.uic.edu/classes/idsc/ids470/hair5ed/nts07b.htm.

[21]Shneiderman, Ben. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. 3rd ed. Addison Wesley, 1998. Reading, MA

[22]Shneiderman, Ben. "Direct Manipulation for Comprehensible, Predictable, and ControllableUser Interfaces". *[cite this]*. Addison Wesley, 1998. Reading, MA

[23]Sims, Karl. "Artificial Evolution for Computer Graphics". *Computer Graphics* **25**(4), July 1991. 319-328 http://www.genarts.com/karl/papers/siggraph91.html.

[24]Tete a tete URL. http://ecommerce.media.mit.edu/tete-a-tete.

[25]Wexelblat, Alan, and Pattie Maes. *Issues for Software Agent UI*. http://mevard.www.media.mit.edu/people/wex/agent-ui-paper/agent-ui.html. cited with permission of author

GLOSSARY

Automatic Collaborative Filtering. A recommendation system that groups a new user with an existing group based on similar likes and recommends items that they liked to the user.

Automatic Constraint Filtering. A recommendation system that asks users questions regarding desired criteria in the item in question. It then filters its database on this criteria.

Conjoint Analysis. A marketing tool that asks consumers to rate examples. It analyzes their ratings to ascertain which attributes are important to the consumers.

Exact Preference. A preference that a variable have a desired value. An exact preference is modeled as a Gaussian random variable with a mean of the desired value.

Exploitation. Using existing knowledge of a good solution to try incrementally neighboring solutions.

Exploration. Exploring other areas of the multidimensional space to ensure that a search is complete.

Hyperspace. A multi-dimensional space.

Implicit Profiling. Ascertaining attributes of the user without explicit user instruction. This often occurs by observing user behavior.

One-Sided Preference. A preference that represents an upper or lower limit on the variable's values.

Ontology. A descriptive language. Often used to create a hierarchy.

Product Brokering. The act of assisting a user purchase a product or service.

Qualitative Aspects of Shopping. Product traits that cannot be measured. These include styling and color.

Quantitative Aspects of Shopping. These represent the values of a product that can be measured.

Serendipity. See exploration

SUV. Sports Utility Vehicle. A truck-like car.