OSGS - A Personalized Online Store for E-Commerce Environments*

Raz Lin[1], Sarit Kraus[1], Jeffrey Tew[2]

[1] Department of Computer Science, Bar-Ilan University
Ramat-Gan, 52900, Israel
{linraz, sarit}@cs.biu.ac.il
[2] MSRL Lab, GM R&D Center,
Warren, MI USA
jeffrey.tew@gm.com

Abstract. Current online stores suffer from a cardinal problem. There are too many products to offer, and customers find themselves lost due to the vast selection. As opposed to traditional stores, there is little or no guidance that helps the customers as they search. In this paper, we propose a new approach for designing a successful personalized online store enabling the successful searching of customers in the store. This approach is based on algorithms commonly used in recommendation systems, but which are rarely used for searches in online stores. We employ this approach for both keyword and browse searches, and present an implementation of this approach. We compared several search guide algorithms experimentally, and the experiments' results show that the suggested algorithms are applicable to the domain of online stores.

1 Introduction

As on-line stores offer more and more products, it becomes more difficult for customers to find what they need with a reasonable amount of time. As the time the customers spend on searching for the desired product increases, the time they spend in the store focusing on a single search increases, and thus the chances they would wish to visit the store again in the future decreases. While in a retail store there is a salesperson, who guides the customer during the search process and helps him find the best product that suits his needs, such sales personnel are not available in online stores. Thus, online stores need to use other mechanisms that will replace the traditional role of the salesperson.

Thus it is crucial to design an online store in a way that will compensate for this disadvantage. A recent published survey by Andersen (2001) states that nearly 60% of consumers who buy from online stores have purchased from the same retailer both online and in person. A survey published by CARAVAN (2001) states that convenience is the top reason consumers like to shop on the Web (56%). Thus, a good

* This work is supported by a grant from GM.

online store design should make the purchasing process as easy and convenient as possible, and allow customers to find the desired product with the least possible time effort.

We have developed a general personalized system for guided search in an online store (OSGS) that assists customers in completing their purchases without wasting time. In this way the system will adapt itself to best fit the customer in question, fulfilling our aim to provide an efficient and fast mechanism. We believe that a general system for guided search also requires algorithms to identify the customers' profiles. This will lead to long-term relationships with customers and will increase the store's long-term benefits. This system should work in any domain, with minor adjustments to the algorithms and the database it uses. We have implemented and tested our system in the domain of GM's auto-spare parts.

OSGS supports both keyword search and browse search and applies three different personalization algorithms: (a) customer's history, (b) neighbors by history profile, and (c) neighbors by demographics and preference-profile, as explained in Section 3.

The experiments show that OSGS improves the customers' satisfaction from the search process, and reduces customer effort during the search itself. In the next section we discuss previous research in the area of recommendation systems and search engines. Section 3 describes our proposed OSGS and its implementation. Section 4 describes the experiments and our results. Finally, Section 5 states our conclusions.

2 Background

Recommendation systems recommend the customer products they think the customer would be interested in buying. They differ from our proposed system since they do not support the customer in the search process itself; they merely provide suggestions. The main technologies in recommendation systems that are commonly used to address the challenge of finding the right product for the customer are information retrieval and collaborative filtering (Aggrawal et al. 1999, Chen et al. 1998, Goffinet et al. 1995, Good et al. 1999, Herlocker et al. 2000, Sarwar et al. 2000, Shardanand et al. 1995). Our system is not a recommendation system. We help the customer *while* he engages in the actual searching for products.

Kitts et al. (2000) developed a recommendation system which uses probabilities derived from the interests of the customer's "soul mates" (neighbors), where the neighbors are formed using the customer's history-profile alone. When computing the probabilities, they operate under the assumption of conditional independence of past behavior of the customer. Like Kitts et al. (2000), we also use the interest of the customer's neighbors. However, in our system we do not assume conditional independence, and also we maintain two different neighbors for the customer, and not just one: one is based on the customer's history, and the other is based on his demographics- and preference-profile.

Shardanand and Maes (1995) developed the Ringo system in which they used history based customer profile and history based neighborhoods. However, their system is a recommendation system, and not guided search system, and thus they differ from our system.

Pazzani and Billsus (2002) suggest using adaptive web site agents in order to capture the intention of the user. Essentially, they use the user's history, patterns of access by other visitors, and similarity of documents to assist the user with navigating a web site. In our system we also follow the same intuition. While Pazzani and Billsus (2002) personalize and adapt the web site to the user, we personalize and adapt the search to the current user. Pazzani and Billsus (2002) take into consideration the history of the user, the history of other users, and similarity of documents. We, however, use the history of the user, the history of other users that are similar to the current user (and not the history of all users), and other factors, such as the preferences of the user, as well. Since we use a search engine, we also take into consideration the keywords the user supplies, and use all these elements in the personalization process.

McGinty and Smyth (2002) also try to capture the intention of the user. They do so by using a comparison-based recommendation. In this technique the user makes a query, then reviews the recommendation and selects a preference case as feedback. The information about the difference between the selected case and the remaining alternatives is used to revise the query for the next cycle. Although the system tries to help the user while he engages in the search it still differs from our system. As opposed to our system, this system does not keep any profile of the customer. Thus, the customer must always start the search from the same start point. His history and preference profiles are not saved and thus the system only relies on the initial query and the feedback given at each iteration.

Moreover, all the systems, which use personalization techniques and that were described above, typically do not combine different personalization techniques. We integrate information retrieval and collaborative filtering techniques and apply a user profile that consists of a demographic profile, a preferences profile and a history profile.

Another way to help customers is to provide search engines to find necessary products. Most search engines provide a simple interface for either keyword search or browse search. Our system enables both.

In a keyword search, the customer provides the system with keywords and the system, in turn, provides the customer with a list of the products which best fit his keywords. The choice of the appropriate products is done using a similarity measure. The most effective one for keyword search has been established as being the inverse document frequency multiplied by term frequency (IDF*TF) (Goffinet et al. 1995), i.e., the inverse of the number of occurrences of the keyword in the whole collection of documents multiplied by the number of occurrences of the keyword in the document. In online stores, most stores retrieve all the documents or use some table lookup for popular keywords. Another option for assisting a user in his search is to try to improve his query by adding, deleting, and replacing the user's keywords with new ones (Brin et al. 1998, Chen et al. 1998, Goffinet et al. 1995). This approach is not usable in a browse search, since there are no queries in browse searches. Moreover, in the domain of online stores, in general, and in auto spare parts in particular, there is little maneuvering space for identification of additional keywords other than those provided by the customer. Thus, we do not believe that applying sophisticated algorithms to the query would be of much assistance.

Another technique suggested by Limthanmaphon et al. (2002) is Case-Based Reasoning (CBR). They state that CBR techniques are used in order to reuse previously successful negotiation experiences and to retrieve the relevant product from the database. They suggest taking the user's query, which consists of several attributes and preferences about the desired product, and using CBR to find the desired product, perhaps after modifying the query. Once found, the query, the modifications to the query and the result are saved and used in future searches for the user. Basically, our system also uses some manipulation on the purchase's history of the customer. However, we do not modify the customer's query. Also, the customer's query in our case differs from the customer's query in (Limthanmaphon et al. 2002). In Limthanmaphon et al. (2002) the customer's query contains also preferences of the customer regarding the desired product. In our system, the query only contains keyword related to the item. The preferences regarding the products are retained in the customer's demographics- and preference-profile. We also use those preferences while searching for products. Also, while our algorithms are also applicable to browse search, it seems very difficult to use CBR techniques in browse search.

Browsing enables the customer to search for the product by using categories. The customer starts with the top-most level category and continues to explore new sub-categories until he finds the desired product. One advantage of this kind of search is that it allows the customer to obtain information on possible products during the search. Another advantage is *scope reduction*, i.e. reducing the range of products of interest to the customer.

In the browsing task, the customer is assumed to have incomplete, imperfect knowledge of the contents and its organization. Because of this, the browsing process is fundamentally uncertain and iterative (Holte et al. 1994). The browsing allows the customer to search for a product even without knowing any keywords to aid in his search.

Most of the current browsing systems, such as Yahoo! (http://www.yahoo.com) and Amazon (http://www.amazon.com), do not personalize the browsing process. Holte and Drummond (1994) describe an approach for helping users while they browse a library of software. They try to infer the customer’s intention as he browses in order to limit the time involved and help perform a successful search by using the past history of the user and then suggesting other pages he might be interested in. Our system tries to infer the customer’s intention by using a combination of algorithms, rather than just one algorithm.

Our system uses a mechanism similar to Henry Lieberman’s Letizia (1995). The Letizia system tracks the user’s browsing behavior, processes it and employs some heuristics. Then it displays the recommendations. As in the case of Letizia (1995), we also use the history profile of the users. However, browsing in an online store is quite different than browsing on the Web. In an online store there is a tight connection between the links the user follows, while in Web browsing the connection between the visited pages is much less tight. Browsing on the Web can only provide a limited view of the user’s intentions. This is because the user usually searches numerous sites, and most of them do not have a specific relationship among them. As opposed to this, browsing in a specific online store can be used to estimate the user’s intention more quickly and accurately, since the focus is only on a certain limited domain**.** Thus, our algorithms can be more easily adjusted and fine-tuned to the specific domain.

3 The Online Store Guided Search System (OSGS)

We propose a general system for searching in an online store (OSGS). Our system helps a customer attain a fast and successful search within the vast array of products available. OSGS assists the customer in the two most popular search techniques today: keyword and browse search. We tested OSGS using the domain of GM's auto-spare parts and we call this system GMSIM. An overview of the system is illustrated in Figure 1.

A good online store should enable the customer to search in the best way he is accustomed to. Thus, the design of the online store should include both keyword search and browse search. By combining both searches we allow full search coverage and more maneuver space for the customer to search for an item in the best way he thinks or accustomed to.

As shown in Figure 1, our system supports both keyword search and browse search. Also, our system enables to incorporate keyword search with the browse search: the customer can start searching using browse search, getting preliminary results, and continue by applying keyword search on the items found in the preliminary results. After getting the initial results, either by using the keyword search or the browse search, or the combination of them, OSGS applies some or all of the personalized algorithms. Finally, OSGS displays the final results to the customer, as shown in Figure 1.

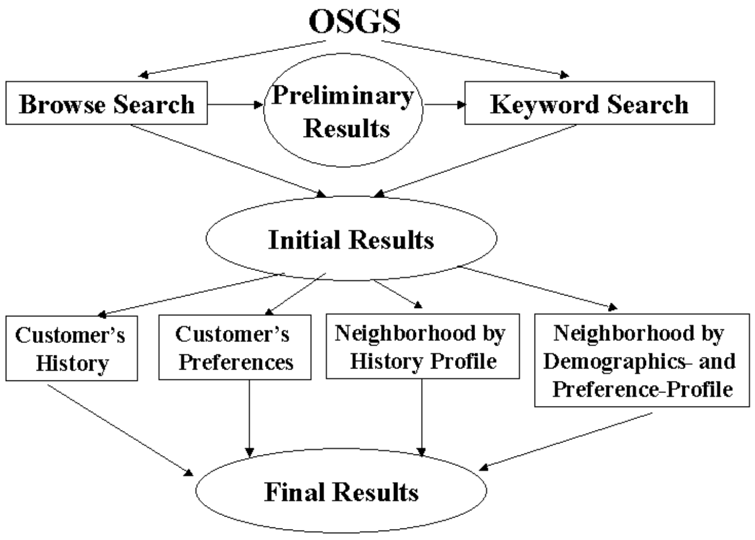


Figure 1. An overview of OSGS

In the remainder of this section we explain how we enable the keyword search and browse search and how we apply the personalization algorithms in OSGS.

A product search can be implemented using a search tree. Each internal node of the tree is labeled by a category name, and the leaves are labeled by names of products. Each successor node is a sub-category of its predecessor node. The root node represents the initial problem situation. For example, in GM's "Auto Spare Parts" domain, the root node is labeled by "Auto Spare Parts," and a leaf can be labeled by "Relay Fan." When a customer searches for a product in OSGS a search tree is constructed in both the cases of keyword search and browse search. A product path is denoted <v_0, v_1, ..., v_k>, where v_0 is the root node, v_k is the leaf that is labeled by the product's name, there is an edge between v_i and v_{i+1}, $0 \leq i \leq k-1$, and v_{i+1} is a successor node of v_i. For example, in GM's "Auto Spare Parts" domain, one of the paths from the root node to the leaf node labeled by "Relay Fan" contains the nodes labeled: "Auto Spare Parts" → "2001" → "Cadillac" → "Catera" → "Relay Fan."

In a keyword search, the search tree is hidden from the customer, while in a browse search the search tree is presented incrementally to him. In addition, in a browse search the customer interactively influences how the search tree is developed by choosing which category to further explore. Efficiency of the search and the satisfaction of the customer depend on the size of the developed search tree and on the decision of which nodes to explore. It also depends on the order in which the nodes in a given level of the tree are presented to him.

In a regular keyword search, every customer who uses the same keywords will get the same results back. In a regular browse search, also due to lack of personalization, a different problem arises: each customer sees the same categories in the *same* order.

We suggest personalizing the search for each customer using a combination of common techniques that are used today in recommendation systems. OSGS prunes the search tree according to the customer's profiles and the profiles of customers that are similar to him. Thus, for each category only a subset of its sub-categories is included in the search tree. In addition, in a browse search the order in which categories and the products are presented to the customer is personalized.

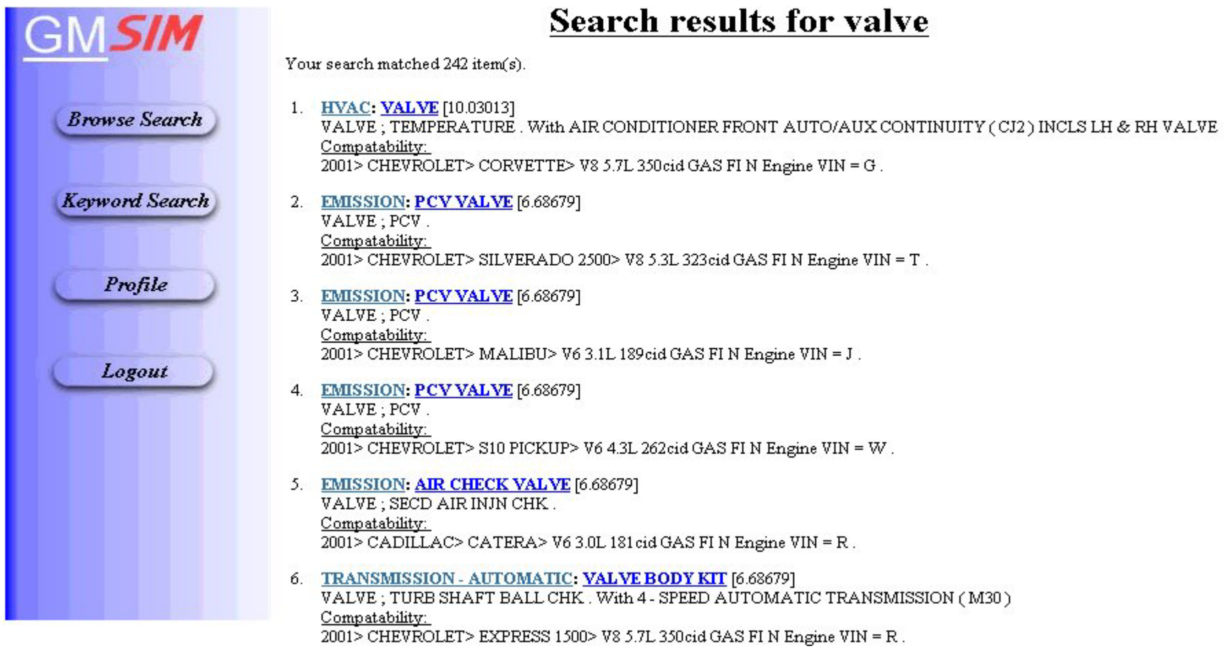
For the personalizing of the search, OSGS maintains two different profiles for each customer: (a) the customer's history profile and (b) the customer's demographics and preference profile. OSGS can choose which combination of the personalization techniques described above to use during each search, as summarized in Table 2 of Section 3.5. Figure 1 displays an overview of our proposed OSGS.

By applying the proposed personalization algorithms we try to overcome the problem arisen: in the keyword search each customer will get, perhaps, different items, since different weights are assigned to the different items, based on the algorithms and the personality of the customer in question. In browse search, the order in which categories and products are presented to the customer is also personalized now. This is due to the fact that we use the weights that are generated by the algorithms stated above. Those weights are used for changing the *order of display* for each customer. Categories and items with higher weights will appear higher in the search tree for the customer. Thus, the order of display of the search tree differs from customer to customer and is personalized for each customer.

Figures 2 and 3 illustrate this idea by a comparison between two different examples of keyword searches. The first, depict in Figure 2 is of a customer searching for "valve" without using any personalization algorithm. The second search, in Figure 3, is of another customer searching also for "valve", however, this time he is using the personalization algorithms. We can see that using the personalization algorithms gives the customer results that differ from other customers, and perhaps more suited for his needs. This is due to the different weights each algorithm gives to different items of different customers. In this example the customer has a history of purchases of Cadillac parts, thus the algorithms took it into account.

Figures 4 and 5 present the same comparison, only this time using the system's browse search. We can see how the order of the categories differs in the different searches, and thus the view of the tree is different and customized to the customer. Again, this is due to the personalization algorithms and the different weights assigned for each item of each customer, where the customer in question had purchased "interior parts" in the past. Also, it is shown how at each point of the browse search the customer is able to continue his search using keyword search. In such a case, more weight is given to items that are located at the current category the customer browsed.

To allow the most flexibility of the online store to the customer, we allow OSGS to choose which combination of the personalization techniques to use during each search. Detailed description of the algorithms is presented in the following subsections.



Search results for valve

Your search matched 242 item(s).

1. HVAC: VALVE [10.03013]
VALVE ; TEMPERATURE . With AIR CONDITIONER FRONT AUTO/AUX CONTINUITY (CJ2) INCLS LH & RH VALVE
Compatability:
2001> CHEVROLET> CORVETTE> V8 5.7L 350cid GAS FI N Engine VIN = G .

2. EMISSION: PCV VALVE [6.68679]
VALVE ; PCV .
Compatability:
2001> CHEVROLET> SILVERADO 2500> V8 5.3L 323cid GAS FI N Engine VIN = T .

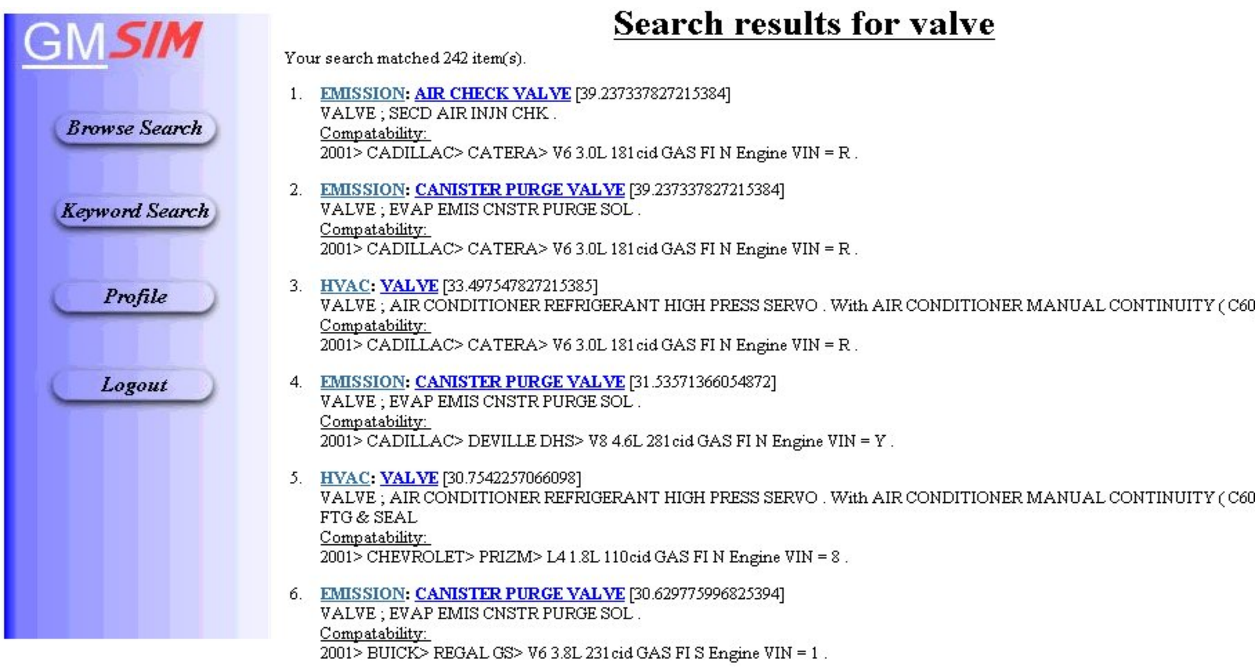
3. EMISSION: PCV VALVE [6.68679]
VALVE ; PCV .
Compatability:
2001> CHEVROLET> MALIBU> V6 3.1L 189cid GAS FI N Engine VIN = J .

4. EMISSION: PCV VALVE [6.68679]
VALVE ; PCV .
Compatability:
2001> CHEVROLET> S10 PICKUP> V6 4.3L 262cid GAS FI N Engine VIN = W .

5. EMISSION: AIR CHECK VALVE [6.68679]
VALVE ; SECD AIR INJN CHK .
Compatability:
2001> CADILLAC> CATERA> V6 3.0L 181cid GAS FI N Engine VIN = R .

6. TRANSMISSION - AUTOMATIC: VALVE BODY KIT [6.68679]
VALVE ; TURB SHAFT BALL CHK . With 4 - SPEED AUTOMATIC TRANSMISSION (M30)
Compatability:
2001> CHEVROLET> EXPRESS 1500> V8 5.7L 350cid GAS FI N Engine VIN = R .

Figure 2. Keyword example without personalization

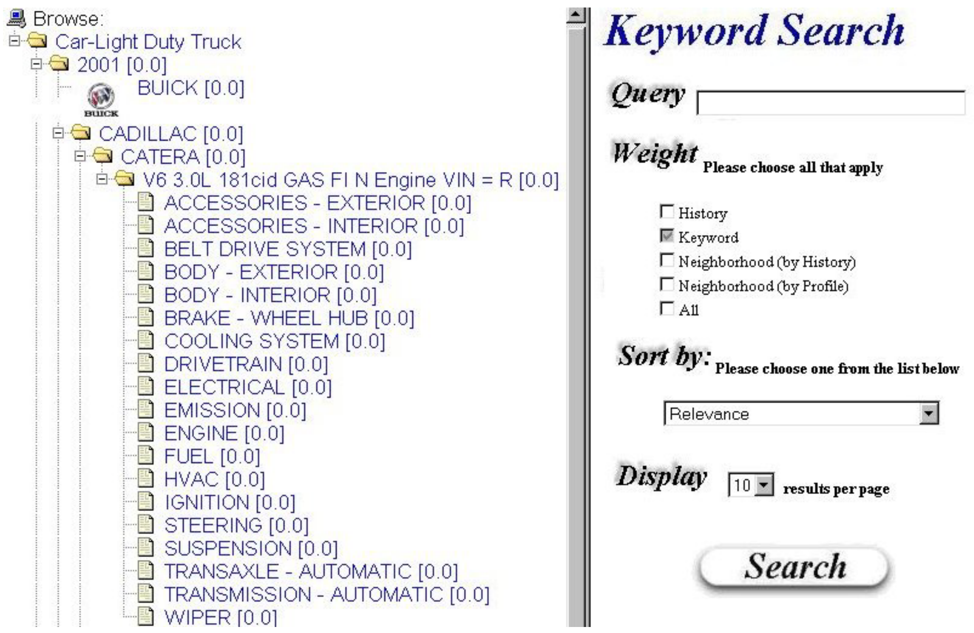


Search results for valve

Your search matched 242 item(s).

1. EMISSION: AIR CHECK VALVE [39.237337827215384]
   VALVE ; SECD AIR INJN CHK .
   Compatability:
   2001> CADILLAC> CATERA> V6 3.0L 181cid GAS FI N Engine VIN = R .
2. EMISSION: CANISTER PURGE VALVE [39.237337827215384]
   VALVE ; EVAP EMIS CNSTR PURGE SOL .
   Compatability:
   2001> CADILLAC> CATERA> V6 3.0L 181cid GAS FI N Engine VIN = R .
3. HVAC: VALVE [33.497547827215385]
   VALVE ; AIR CONDITIONER REFRIGERANT HIGH PRESS SERVO . With AIR CONDITIONER MANUAL CONTINUITY (C60
   Compatability:
   2001> CADILLAC> CATERA> V6 3.0L 181cid GAS FI N Engine VIN = R .
4. EMISSION: CANISTER PURGE VALVE [31.53571366054872]
   VALVE ; EVAP EMIS CNSTR PURGE SOL .
   Compatability:
   2001> CADILLAC> DEVILLE DHS> V8 4.6L 281cid GAS FI N Engine VIN = Y .
5. HVAC: VALVE [30.7542257066098]
   VALVE ; AIR CONDITIONER REFRIGERANT HIGH PRESS SERVO . With AIR CONDITIONER MANUAL CONTINUITY (C60
   FTG & SEAL
   Compatability:
   2001> CHEVROLET> PRIZM> L4 1.8L 110cid GAS FI N Engine VIN = 8 .
6. EMISSION: CANISTER PURGE VALVE [30.629775996825394]
   VALVE ; EVAP EMIS CNSTR PURGE SOL .
   Compatability:
   2001> BUICK> REGAL GS> V6 3.8L 231cid GAS FI S Engine VIN = 1 .

Figure 3. Keyword example with personalization



Browse:
- Car-Light Duty Truck
  - 2001 [0.0]
    - BUICK [0.0]
    - CADILLAC [0.0]
      - CATERA [0.0]
        - V6 3.0L 181cid GAS FI N Engine VIN = R [0.0]
          - ACCESSORIES - EXTERIOR [0.0]
          - ACCESSORIES - INTERIOR [0.0]
          - BELT DRIVE SYSTEM [0.0]
          - BODY - EXTERIOR [0.0]
          - BODY - INTERIOR [0.0]
          - BRAKE - WHEEL HUB [0.0]
          - COOLING SYSTEM [0.0]
          - DRIVETRAIN [0.0]
          - ELECTRICAL [0.0]
          - EMISSION [0.0]
          - ENGINE [0.0]
          - FUEL [0.0]
          - HVAC [0.0]
          - IGNITION [0.0]
          - STEERING [0.0]
          - SUSPENSION [0.0]
          - TRANSAXLE - AUTOMATIC [0.0]
          - TRANSMISSION - AUTOMATIC [0.0]
          - WIPER [0.0]

Keyword Search

Query

Weight Please choose all that apply

☐ History
☑ Keyword
☐ Neighborhood (by History)
☐ Neighborhood (by Profile)
☐ All

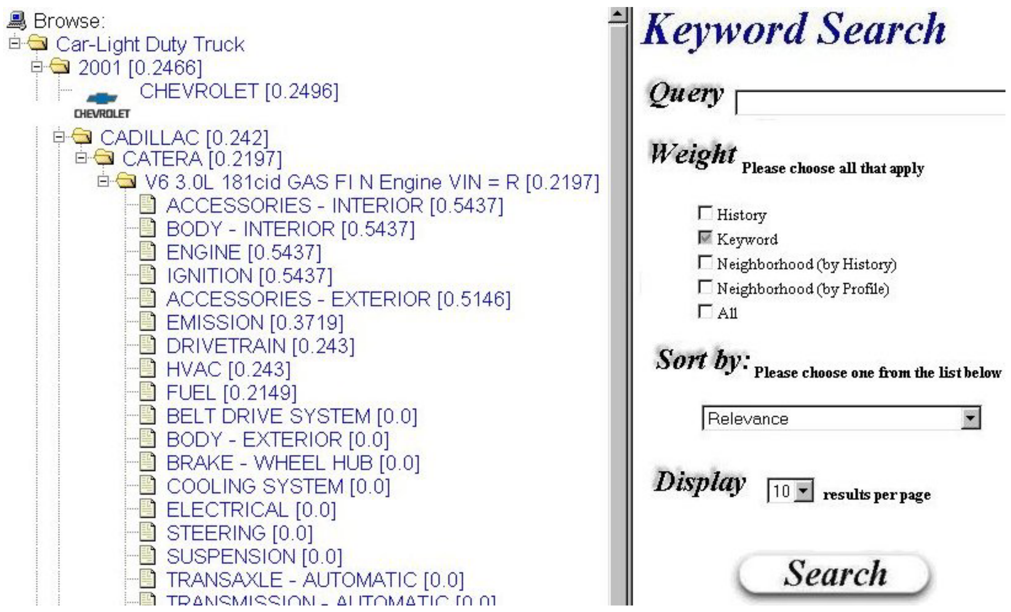
Sort by: Please choose one from the list below

Relevance

Display 10 results per page

Search

Figure 4. Browse example without personalization



Browse:
- Car-Light Duty Truck
  - 2001 [0.2466]
    - CHEVROLET [0.2496]
    - CADILLAC [0.242]
      - CATERA [0.2197]
        - V6 3.0L 181cid GAS FI N Engine VIN = R [0.2197]
          - ACCESSORIES - INTERIOR [0.5437]
          - BODY - INTERIOR [0.5437]
          - ENGINE [0.5437]
          - IGNITION [0.5437]
          - ACCESSORIES - EXTERIOR [0.5146]
          - EMISSION [0.3719]
          - DRIVETRAIN [0.243]
          - HVAC [0.243]
          - FUEL [0.2149]
          - BELT DRIVE SYSTEM [0.0]
          - BODY - EXTERIOR [0.0]
          - BRAKE - WHEEL HUB [0.0]
          - COOLING SYSTEM [0.0]
          - ELECTRICAL [0.0]
          - STEERING [0.0]
          - SUSPENSION [0.0]
          - TRANSAXLE - AUTOMATIC [0.0]
          - TRANSMISSION - AUTOMATIC [0.0]

Keyword Search

Query

Weight Please choose all that apply

- [ ] History
- [x] Keyword
- [ ] Neighborhood (by History)
- [ ] Neighborhood (by Profile)
- [ ] All

Sort by: Please choose one from the list below

Relevance

Display 10 results per page

Search

Figure 5. Browse example with personalization

3.1 Customer History

The history is used for predicting the customer's needs, under the assumption that customers often buy similar products. A customer's search or browsing is included in his history by saving the path from the root of the search tree to the destination product. (Note that using either searching or browsing eventually gives us a path.)

A *historical event h* consists of a triplet ($<v_1, v_2, \ldots, v_k>, t, l$), indicating that at time t the customer performed a search or browsed for a product whose path on the search tree consists of the nodes $<v_1, v_2, \ldots, v_k>$. v_k is labeled by a product the customer is interested in, v_i, $0 < i < k$, denotes a category in the product's path and l is the *likeability* parameter, which helps to distinguish between a successful search or browsing (a search or browsing which led to a purchase by the customer), and an unsuccessful search or browsing. Every time a customer is interested in a product, we add an historical event, which consists of that product, to the customer's history. A customer *is interested* in product *j* if one of the following two conditions holds: (a) the customer purchased the product, or (b) the customer visited the product path at least *K* times, where *K* is a predefined bound. The motivation behind entering into the history products the customer did not buy but only visited a certain number of times is our assumption that if the customer visited the product many times, even without buying the product online, he may be interested in some details about the product. Maybe he has not bought the product yet because he wants to compare it with other products or products from stores, maybe he is interested in a similar product, or maybe he purchased the product offline in a regular store or by phone. We define the *history H* of a customer to be the set of his most recent historical events.

3.2 Customer Demographics and Preference Profile

The demographics and preference profile captures general information and preferences about the customer and his domain-dependent information. For example, in GMSIM the demographics- and preference-profile consists of the following fields: name, gender, age range, occupation, and location. The fields of the domain dependent information in GMSIM are quality preference, price preference, warranty preference, expertise knowledge and car details (make, model, and year). The values of the demographics and preference profile are given explicitly by the customer himself, and can be modified at any time. This as opposed to the history profile which is an implicit one, very dynamic and changes quite often.

3.3 Neighborhood Formation

The main goal of the neighborhood formation is to find for each customer *x* a set of *n* customers that are the most similar to him. The similarity is given by the similarity function *sim*. The neighbors are formed by applying proximity measures, such as the *Pearson correlation* (Sarwar et al. 2000, Shardanand et al. 1995), or *cosine similarity* (Good et al. 1999, Sarwar et al. 2000) or *mean squared differences* (Shardanand et al. 1995), between two opinions or profiles of the customers. Given experiments done in other systems (Herlocker et al. 2000, Shardanand et al. 1995), OSGS uses the mean squared differences proximity measure. We apply it to (a) the history of the customers and to (b) their demographics and preference profile in order to decide if *customerx* is a neighbor of *customery*. This is done by applying a similarity function for each field of the profiles (see (Lin 2002) for a detailed description). Different similarity functions are used for each field in order to normalize the fields' values to the range [0, 1]. As stated above, the similarities of all the fields are combined using the mean squared differences proximity measure in order to obtain one weight for the profile (Shardanand et al. 1995).

For example, in GMSIM the similarity function for the gender, age, quality, price, domain expertise, and warranty fields is given by the following formula:

$$sim(P_{x_i}, P_{y_i}) = \frac{W(x_i)}{1 + \left|P_{x_i} - P_{y_i}\right|}, \tag{1}$$

where x_i and y_i represent field *i* in the profile for customers *x* and *y*, respectively, P_{x_i} and P_{y_i} represent the values of the fields, and $W(x_i)$ represents the importance of field *i* for the customer for which we search neighbors. Then a threshold is applied so only the most similar customers are chosen as neighbors for a given customer (for example, *Top-N* neighbors (Sarwar et al. 2001)). OSGS constructs two neighborhoods for each customer, as described above. The idea behind this is that when a customer is searching for a specific product, it might be useful to assist him in finding the desired product by using information derived from customers who have similar tastes, or similar characteristics.

3.4 Weight Functions

In order to prune the search tree effectively, each node in the search tree is assigned a weight that estimates the customer's interest in the category or product associated with the node. In this section we present a description of the four weight functions that are used in our algorithms. The weight functions are based on the customer's preferences, the keywords provided by the customer (in case of a keyword search), the customer's history and his neighbors (neighbors-by-history-profile and neighbors-by-demographics-and-preference-profile). Table 1 summarizes all the weight functions and their notations.

Table 1. Summary of weights and notation used in OSGS

Customer's Weights	Notation
Preference weight	W_p
Keyword weight	W_k
Customer's *x* history weight	W_h^x
Neighbor weight according to history profile	W_{n_h}
Neighbor weight according to demographics and preference profile	W_{n_p}
Customer's *x* category weight	$W^x(category)$

3.4.1 Customer's Preference Weight

In addition to constructing a neighborhood for a customer from his profile, we use the customer demographics and preference profile to give weight to products and categories in the search tree. Since each product is constituted of different attribute fields (such as price, quality, and so on), we give higher weights to products that correspond to the preferences indicated in the customer's profile. The demographics and preference profile also includes, for example, the car information of the customer, and this information is also used to give weight to different categories, since each category is associated with a different car. This weight is calculated using a proximity measure between the fields of the products that correspond to the profile's fields. We denote that weight as W_p. The customer's preference weight is static and only changes if the customer changes his profile or if a product or a category is updated or added.

3.4.2 Keyword Weight

When the customer searches using keywords, he supplies the keywords, which are then used in order to select the relevant products from the entire collection of products. The weight that captures the relevance of the product with respect to the keywords is denoted W_k and is calculated using the *IDF*TF* approach (Chen et al. 1998, Goffinet et al. 1995), which is the one most commonly used in search engines. However, since this method can be used only with respect to a corpus of documents, in order to use this approach we needed to define what constitutes a document in OSGS. For each $product_i$ we define $document_i$ to be the collection of the following fields of the product: (a) description, (b) name, (c) category, (d) notes, and (e) keywords.

3.4.3 History Weight

We define the weight of each historical event (see Section 3.1) to be equal to the likeability value of the product to which the historical event belongs multiplied by $exp(t/_{current\_time})$. The history weight of a product or category is defined to be the sum of total weight of the historical events in which it appears, divided by the number of historical events in which it appears. The history weight for customer *x* is denoted W_h^x.

3.4.4 Neighborhood Weight

The neighborhood weight is based on the proximity measure. For each neighbor N_i we compute the mathematical product of the category's history weight for that neighbor with the value of the neighbor's similarity to the customer:

$$W_h^{N_i}(category) \cdot sim(x, N_i) . \tag{2}$$

The total neighbors' weight is an average of all the mathematical products above. We denote the neighbors' weight according to the customer's history profile as W_{n_h} and the one according to customer's demographics and preference profile by W_{n_p}.

3.5 Searching Using Keyword(s)

In the following subsections we present the keyword search algorithm applied in our system. We describe two options for keyword search: top-down search, and bottom-up search, we explain each one and provide theoretical proof concerning these two techniques that help determine which technique to use in different cases.

3.5.1 Keyword Search in OSGS

There are two techniques that are applicable when searching using keywords: (1) *bottom-up* search and (2) *top-down* search. The most common technique among online shops is bottom-up search (for example, as in the keyword search in Amazon: http://www.amazon.com). In top-down search the search begins with the categories themselves and ends with the products. This can be achieved if the domain's structure is hierarchical and if there is some sort of a link or relation between each category and its products.

As opposed to a top-down search, in a bottom-up search the search begins with the possible products and continues to the top-most category. In this approach the keywords and their weights are used to find products that are most likely wanted, and then OSGS tries to reconstruct the path back to the root category. Though the products have already been found, when deciding on which products to present to the customer OSGS uses the weight functions that take the paths into consideration. The reason behind this approach is to consider more than just the products found according to the keyword weight. Thus, in case of a failure in the search, the customer can see the categories that were related by OSGS to each product and choose from among them in order to find more appropriate products. This approach is also good when the customer does not really know what he is looking for, but has some idea as to the direction of the search. In particular, in bottom-up search OSGS first computes the keyword weight for all products, according to the customer's query. Afterwards, it computes the neighbors' weight, history weight, and preference weight for those products, in order to choose the products with the highest normalized weight.

Then, for *each* product with positive weight OSGS chooses at least one ancestor, using the normalized weight of the neighbors (neighbors by history profile and neighbors by demographics and preference profile), history and customer's preferences, as described below. OSGS continues this process until for each chosen product it gets to the root, i.e., to the top-most category.

In top-down search OSGS does approximately the same thing, but in reverse. At each iteration it chooses sub nodes with the highest weights.

The total normalized weight for a category and customer x is given by:

$$W^x(category) = \alpha_1 W^x_{n_h}(category) + \alpha_2 W^x_{n_p}(category) + \beta W^x_h(category) + \gamma W_k(category) + \eta W^x_p(category). \quad \textbf{(3)}$$

The parameters $\alpha_1, \alpha_2, \beta, \gamma, \eta$ are determined, in the implementation of OSGS, using trial and error. Table 2 lists the different algorithm versions available in OSGS.

Table 2. Versions of algorithms in OSGS

Algorithm Version	Option	Formula
History-based algorithm	$\alpha_1 = \alpha_2 = 0$	$W^x(category) = \beta W^x_h(category) + \gamma W_k(category) + \eta W^x_p(category).$
Neighbors-by-history algorithm	$\alpha_2 = \beta = 0$	$W^x(category) = \alpha_1 W^x_{n_h}(category) + \gamma W_k(category) + \eta W^x_p(category).$
Neighbors-by-demographics-and-preference algorithm	$\alpha_1 = \beta = 0$	$W^x(category) = \alpha_2 W^x_{n_p}(category) + \gamma W_k(category) + \eta W^x_p(category).$
Search without algorithms	$\alpha_1 = \alpha_2 = \beta = \eta = 0$	$W^x(category) = \gamma W_k(category).$

When considering the history and the historical events, using a top-down search gives more consideration to the path of the products, meaning that we will get to the final product using the most common path the customer uses. This is because we begin selecting items in the search tree from the root node. Thus, our selection will choose the nodes with the highest weight that are descendant of the root node. These nodes are the ones of the most common path the customer uses. Using a bottom-up search, however, gives more consideration to the weight of the product, with less consideration to the path of that product. This is because our selection of products begin from the items themselves and not from the root node.

Due to the above considerations, using a top-down search enables us to consider the number of keywords searched per category, i.e., giving more weight to categories in which the likelihood of finding a product that fits the keywords is higher. In a bottom-up search only the number of keywords searched per product is taken into consideration, thus possibly avoiding finding other solutions that might appeal more to the customer.

Thus, the keywords impose constraints on the search domain. We would like to avoid conditions in which too many keywords cause over-constraint (not enough results) and too few keywords cause under-constraint (too many results). To do so we use the number of keywords in order to decide whether to use a bottom-up search or a top-down search. The formal justification of this intuition is presented below.

3.5.2 Top-Down Search vs. Bottom-Up Search

In this section we provide theoretical proof concerning the results generated using both of our proposed keyword search algorithms: the top-down search and the bottom-up search. The following proof assumes that for each node in the search tree we assign only *one* immediate ancestor and the nodes that will appear in the result list are all the nodes whose weight is higher than a certain threshold.

We denote $K = \{k_1, k_2, \ldots, k_n\}$ the set of keywords used in the search, $P_{BU} = \{item_1, item_2, \ldots, item_l\}$ the set of items returned using the bottom-up search, and $P_{TD} = \{item'_1, item'_2, \ldots, item'_m\}$ the set of items returned using the top-down search. The following proposition will help us decide when it is best to use top-down search over bottom-up search, or the opposite.

In Proposition 1 we state that given the algorithms used in OSGS, using bottom-up search returns less results than using top-down search.

Proposition 1: $P_{TD} \supseteq P_{BU}$.

Proof: We will prove the proposition using some claims stated below.

Claim 1: $W_h(father(node)) \geq W_h(node)$.

In Sections 3.1 and 3.4.3 we defined the history weight. We stated that if a node v_k is labeled by a product that the customer was interested in, and if the path of that product on the search tree consists of the nodes $<v_1, v_2, \ldots, v_k>$ then all the nodes in the path $v_1, v_2, \ldots, v_k$ are assigned the same weight, i.e., we assign the same weight to all the nodes in the path of the item and not only to the final item. Since each non-leaf node u in the search tree may have several children nodes, the history weight of u is at least the same as all of its children nodes. This is due to the fact that u must be included in the path from the root of the search tree to any of its children nodes.

Claim 2: $W_k(father(node)) \geq W_k(node)$.

In Section 3.4.2 we stated that we use the *IDF*TF* technique in order to assign the keyword weight to the different items. The weight assignment of the keyword weight to each non-leaf node is done by using *all* the keywords of its children nodes. Thus, we get that the keyword weight of each non-leaf node is larger than or equal to the keyword weight of its children nodes.

Claim 3: $W_p(\,father(\,node\,))\geq W_p(\,node\,)$.

In Section 3.4.1 we described how we calculate the preference weight for each item. The weight assignment of the preference weight for each non-leaf node is done by combining *all* the preference weights of its children nodes. Thus, we get that the preference weight for each non-leaf node is higher than or equal to the preference weight of its children nodes.

Claim 4: $W(\,father(\,node\,))\geq W(\,node\,)$.

The proof is trivial following claims 1-3 and using equation (3) above.

In the bottom-up search if we chose l items, then their weights are higher than a certain threshold. Thus, *all* their ancestors' nodes also have weights that are higher than the threshold. Since in the top-down search we choose the nodes whose weights are higher than the threshold, we will choose those l items. However, we might also choose other non-leaf categories whose weights are higher than the threshold, but they were not chosen in the bottom-up search since their children nodes do not have weights higher than the threshold (this is possible due to Claim 4). In this case, the top-down search will return categories in the result list, in addition to other items that were returned in the bottom-up search. Thus we conclude that $P_{TD}\supseteq P_{BU}$.■

In the next propositions we use the term of a *good* result list and a *better* result list, as defined below. We also use thresholds on the size of the result list and on the number of high ranking items in the result list. We denote T^- as the threshold of the minimum size of the result list, and T^+ as the threshold of the maximum size of the result list. We also denote L as a predefined threshold on the number of high ranking items in the result list.

We assert that the result list is *good* if the following two conditions hold:

a. The size of the result list is between T^- and T^+. That is, it consists of not too many and not too few items.

b. The number of results in the result list which have a high relevance (or ranking) is at least L.

We say that the result list RL_A is *better* than the result list RL_B if the size of RL_A is between T^- and T^+, and RL_A contains more items with high relevance than RL_B.

We denote $n_{keyword}$ to be the number of keywords included in the query.

In the next two propositions we show the differences in the results gathered by the two search techniques: bottom-up search and top-down search.

Proposition 2: For large $n_{keyword}$'s, top-down search with "*AND*" constraints presents better results than bottom-up search using the same keywords and constraints.

Proof: From Proposition 1 we see that $|\,P_{BU}\,|\leq|\,P_{TD}\,|$. For large $n_{keyword}$'s with "*AND*" constraints between the keywords the probability to obtain a very small result list using bottom-up search is quite high, since there is a smaller probability to find items that contain all the keywords (i.e., an over-constraint situation). Thus, using top-down search might result in a larger result list and supply better results for the customer.■

Proposition 3: For small $n_{keyword}$'s, bottom-up search with "*AND*" or "*OR*" constraints yields better results than top-down search using the same keywords and constraints.

Proof: From Proposition 1 we know that $|P_{BU}| \leq |P_{TD}|$. For small $n_{keyword}$'s with "*AND*" or "*OR*" constraints between the keywords the probability to attain a very large result list using bottom-up search is quite high, since there are fewer keywords to match and the probability to find items that contain less keywords is quite high (i.e., an under constraint situation). Thus, we get a larger result list and we would not want to increase its size. Consequently, using bottom-up search would yield better results in this situation. ■

Proposition 4: For large $n_{keyword}$'s, bottom-up search with "*OR*" constraints yields better results than top-down search using the same keywords and constraints.

Proof: The proof follows from Propositions 2 and 3. For large $n_{keyword}$'s there is a higher probability to find many items that match the constraints using bottom-up search and thus we will find that the result list's size is quite large. Using top-down search will only increase the size of the result list. ■

Some observations can be made from the above propositions and proofs:

(a) Consider limiting the number of returned results at each level of the tree (i.e., every iteration (cf. Section 3.5.1) only a specific number of items it chosen). If the limit in the higher levels, i.e., close to the root of the search tree, is less than the limit in the lower levels, we might return fewer results using top-down search than when using bottom-up search. The reason follows from Proposition 1: assume that we choose the same number of nodes at each level, denoted *X*. Bottom-up search will return the *X* items with the highest weight. Since each node has no more than one immediate ancestor, then there are *at most X* immediate ancestors for those items. Thus, at each level there are no more than *X* different nodes that will lead to those items. Using top-down search will eventually lead to at least *X* items, which will include the *X* items found by the bottom-up search.

However, if we limit the number of nodes at *each* level, using top-down search may cause us not to explore all the ancestors' categories of the chosen items from the bottom-up search. Instead we would choose only some of them, which have the highest weight. Then, we could only continue the search using those nodes and return only few of the items and maybe return other items, which have a lower weight and were not chosen in the bottom-up search.

(b) If the database is maintained such that there are more than one immediate ancestor for an item then we cannot know anything about the relation between the weight of a node and its child's weight. This is because Claim 1 no longer holds. For example, suppose a node, denoted as *node*, has a weight of *w* and has two immediate ancestors, $father_a$ and $father_b$. Assume that most of the searches go through the path that consists of $father_a$ and only a few searches go through the path that consists of $father_b$. Also assume that $father_b$ has no other children nodes and that $father_a$ has other children node(s). Obviously, in this situation $W_h(father_b) \leq W_h(node)$, but we know nothing about the relation between the weights of $father_a$ and *node* (it might be higher, equivalent or lower).

3.6 Searching Using Browsing

In browsing, the idea is to start with the categories and get to the final product. Unlike searching with keywords, there is little logic in allowing browsing to use the bottom-up method, so we focus only on the top-down method.

We use a similar normalized weight for a product or category as used in the keyword search (formula 3), but without the keyword weight. At each level the customer is presented with a list of categories, ordered according to the weights. The customer takes an active part in the search and decides on how the developed search tree is to be constructed. At any point the customer can also choose to backtrack and change or retry his selection. This can be easily done since the entire category tree is presented to him. Since the weights of the categories differ from one customer to another, the order in which the categories are displayed to each customer also differs. This is as opposed to a regular browse search, in which all customers see the same categories, in the same order.

4. Experiments

To test the applicability and efficiency of our approach, we conducted experiments on GMSIM (the implementation of OSGS using the domain of GM's auto-spare parts). In those experiments, the subjects used (a) GMSIM with our suggested algorithms, (b) GMSIM without our algorithms, and (c) ACDelco system (http://www.acdelco.com). The ACDelco system provides services similar to those of GMSIM and allows the customer to use either a keyword or browse search, but not, to our knowledge, any special algorithm.

When analyzing the experiments' results we say that system *A* is better than system *B* if (a) the customer's satisfaction with the search process is higher when using system *A* than when using system *B*, and if (b) the customer's effort (the effort the customers exert to find what they are looking for, measured by search time and number of clicks) is lower when using system *A* than when using system *B*.

Our main hypothesis was that GMSIM is always better than ACDelco. In addition, we had the following three hypotheses:

(a) If customers, who have a similar history of purchases, tend to buy products similar to products their “soul-mates" (customers with a similar profile) have already purchased, then GMSIM with a neighborhood by history algorithm will be better than GMSIM without algorithms.

(b) If customers, who have a similar demographics and preference profile, tend to buy products similar to products their “soul-mates” have already purchased, then GMSIM with neighborhood by demographics and preference profile algorithm will be better than GMSIM without algorithms.

(c) If customers, who buy products and thus have a history of purchases, tend to buy products similar to products they have already purchased, then GMSIM with a customer's history-based algorithm will be better than GMSIM without algorithms.

A plain search with no algorithm can be best, or can at least generate the same results as any of the stated algorithms in some cases, for example cases in which the customer is looking for a "one-time product". However, a plain search with no algorithm gives no advantage and benefit when dealing with repeated and veteran customers. Thus, those customers might feel unsatisfied with the system and choose another system instead.

The reason we tested all three hypotheses, and not just one, and designed a system which uses algorithms that support these hypotheses, is that there are domains in which one algorithm might perform better than the other. For example, in the domain of drugstores, if the customer has a history of disease, an algorithm that supports the customer's history (hypothesis c) might be the best solution. On the other hand, in the domain of books, if the customer is a science-fiction fan, we might find the algorithm that supports neighborhood according to the customer's preferences (hypothesis b) more useful. In yet another option, in the domain of accessories, one might find the algorithm which supports neighbors according to the customer's history (hypothesis a) more useful, since it might be useful to use items other customers, with taste similar to that of the current customer, have bought. Using the algorithm that supports only the customer's history might even turn out to be futile.

Thus, if we have one system which supports all of those hypotheses, we can adjust and adapt it to many domains. We can also use trial and error in order to empirically test which algorithm performs better and even combine different algorithms.

4.1 Experiments Methodology

We divided our experiments into three sub-experiments. In each sub-experiment, when the subjects used the "GMSIM with our proposed algorithms" option they were provided with a different algorithm. The subjects themselves did not know which algorithm the system was using. In the first sub-experiment, the neighborhood-by-history-algorithm was used; in the second, the neighborhood-by-demographics-and-preference-profile based algorithm was used; and in the third sub-experiment the customer's history-based algorithm was used. Twenty people participated in each sub-experiment. Thus, a total of 60 people experimented with our system.

The algorithms were tested using the domain of GM's auto spare parts, which contains about 45,000 products (about 5,000 different catalog numbers).

Each subject participated in only one sub-experiment in which he or she was given a list of 25 short scenarios. Each scenario describes in a short paragraph the reason for the need to buy a car part (for example, due to a problem with the part). The subjects needed to use the scenarios in order to find each part, either using the keyword search or the browse search. When the subject had to use the keyword search, he had to extract the keywords from the scenario. When the subject had to search using the browse search, he had to associate the item in the scenario with its categories.

In order to enable the subjects to know whether or not they found the right part, a picture of the part or some other identification was attached to each scenario. The subjects were instructed to search for at least several minutes, in case they did not find the right part, before skipping to the next part.

Note that either using GMSIM with our proposed algorithms or using GMSIM without it or using ACDelco, eventually returns results to the subject. The question is how much time and effort it will take the subject until he finds the desired product from the results list, and thus how satisfied the subject will be with the search process.

For example, one of the scenarios was as follows: "Last night an officer stopped you since one of your taillight lamps wasn't working. You were instructed to have the problem fixed. You noticed that the problem was caused by a blown bulb." An image was attached to the scenario so the subject would know if he got to the right part. A keyword search for this item could consist of these keywords: “lamp,” “taillight,” “bulb.” After obtaining the results the subject had to find the right item from the results list. If the subject had to use a browse search, he could have seen all categories from which he had to select the correct ones associated with the item. In this case, the subject had to choose the “Electrical” category and then “Lamp – Taillight” in order to get the items in the last category, which contains the item in the scenario.

Among the 25 different products of the scenario, 10 products were searched using a GMSIM browse search and a keyword search with our algorithms (the exact one depends on the part of the experiment), 10 products were searched without our algorithms, and 5 products were searched using ACDelco. A search using only ACDelco and our system without our algorithms allowed us to compare our system to ACDelco, in general, and to compare the efficiency of our algorithms in particular.

Note that in order not to bias results due to, perhaps, scenarios that are more “difficult” than others, the order of the scenarios for each subject was drawn from the uniform distribution (with limits [0-1]). Thus, some subjects had to search for certain scenarios, for example, by using a keyword search with our proposed algorithm, while other subjects had to search for the same scenarios by using a keyword search without an algorithm, or by doing a browse search. Also, the order in which the subjects searched, e.g., searching with an algorithm before searching without an algorithm, or vice versa, was also chosen using the uniform distribution. This was done in order not to bias the results as a result of the subjects becoming more familiar with the searches and the system, thereby yielding better results in their second search.

During the experiment we measured (a) the satisfaction of the subjects with the search process; (b) the overall satisfaction with GMSIM; (c) the time taken for each purchase; and (d) the number of times the customer clicked until he found the desired part. The time and overall satisfaction were acquired using questionnaires presented to each subject. An example of such questionnaire is presented in the Appendix.

After concluding each sub-experiment we performed *t-tests* for the time duration and the average number of clicks and *non-parametric test - Wilcoxon matched-pairs signed-ranks test* (Siegel 1956) for the answers in the questionnaires, comparing the rankings (1-7) and the general preferences.

In order to perform our experiments we had to create and maintain virtual customers, create profiles for our subjects, assign them a history, and compute their neighbors. To do this we created 3 different clusters of customers, each cluster containing 30 customers. For each cluster we randomly chose, using the uniform distribution, 100 products. In each cluster, the products were taken from the same car year and make. Then, for each customer in the cluster we randomly chose about 40 products out of the products associated with his cluster. Each customer derived from a certain cluster had values in the demographics and preference profile fields similar to those of other virtual customers in the same cluster. After creating all of the virtual customers, we calculated (a) the history weight, (b) the neighbors by history profile, and (c) the neighbors by demographics and preference profile for each customer. Each subject was associated with one of those profiles and the products of the scenarios were chosen randomly from the larger set of products associated with his cluster.

4.2 Experiments Summary

In the following subsections we describe the results of our three sub-experiments we have conducted. In the first sub-experiment we used the neighborhood by history algorithm. In the second sub-experiment we used the neighborhood by demographics- and preference-profile algorithm, and in the last sub-experiment we used the customer's history algorithm. The results of all the experiments are also described below and are summarized in Section 4.2.4.

4.2.1 Sub-experiment I: Neighborhood by History Algorithm

Table 3 presents the average ratings the subjects gave to the general satisfaction, time satisfaction, and the ease and flexibility in both the keyword search and browse search with the neighborhood by history algorithm and without any algorithm, and to the keyword search and browse search in general. The table also presents the average time duration and the corresponding standard deviation in all the searches, and the average number of clicks and its standard deviation. The third and sixth rows in the table present the average ratings the subjects gave to the keyword search and browse search in general, accordingly. The results of the experiment supported our hypothesis (a) above. The general satisfaction of the keyword search with the neighborhood by history algorithm was significantly higher than with the keyword search without the algorithm ($z = -1.43$, $p < 0.08$). The ease and flexibility were also somewhat better in the keyword search with the algorithm, in comparison to the keyword search without the algorithm ($z = -1.19$, $p < 0.12$).

Table 3. Average results (*avg*) and standard deviation (*stdev*) of GMSIM's keyword and browse search, GMSIM, and ACDelco in general in the first sub-experiment. Satisfaction and flexibility varied between 1-7

	General Satisfaction		**Time Satisfaction**		**Ease and Flexibility**		**Average Time Duration**		**Average No. of Clicks**	
	avg	**stdev**	**avg**	**stdev**	**avg**	**stdev**	**avg**	**stdev**	**avg**	**stdev**
Keyword Search with alg.	5.80	0.89	5.65	1.27	5.70	1.42	16.40	4.60	4.83	2.75
Keyword Search w/o algorithm	5.40	0.94	4.80	1.20	5.30	1.26	14.55	5.22	5.58	3.53
Keyword Search – general	5.50	0.95	5.20	1.24	5.40	1.19				
Browse Search with alg.	4.50	1.70	4.45	1.67	4.40	1.47	17.55	6.59	7.99	3.46
Browse Search w/o algorithm	4.35	1.57	4.05	1.88	4.45	1.43	16.30	6.01	7.98	2.20
Browse Search – general	4.65	1.42	4.20	1.61	4.35	1.31				
GMSIM – general	5.45	1.05	5.45	1.00	5.50	1.00				
ACDelco	4.95	1.39	5.05	1.73	4.50	1.85				

The time-frame satisfaction using a keyword search with the neighborhood by history algorithm was significantly higher than the level of satisfaction when using the keyword search without the algorithm ($z = -2.23$, $p < 0.01$). It can be explained by the fact that the average number of clicks was somewhat lower in the keyword search with the algorithm than in the keyword search without the algorithm ($t(19) = 1.2$, $p < 0.2$), despite the fact that the average time it took for finding a product using keyword search with the neighborhood by history algorithm was higher than the time it took using the keyword search without the algorithm. At the end of the experiment, the subjects were explicitly asked which search technique they would prefer to use in the future: a keyword search with the neighborhood by history algorithm or a keyword search without the algorithm. Significantly more subjects (60% of the subjects) preferred the keyword search with the algorithm, as compared to only 25%, who preferred the keyword search without the algorithm (z = -1.49, p < 0.07). The other 15% had no preference.

Concerning browse search, we can see from the table that the ratings in the browse search with the algorithm and without it are quite similar. Although we thought that the browse search with the algorithm would yield higher satisfaction than the browse search without it, and even though the average satisfaction of the browse search with the algorithm was higher and the time satisfaction was significantly higher (z = -1.37, p < 0.09), both the general satisfaction and the ease and flexibility result were inconclusive. We believe that one of the reasons for that is the lack of proficiency on the part of the subjects in the browse search. This is since most of the searches on the web are done using keyword search. This is not a factor in the keyword search, since the subject needs only to input the keywords he searches for, without needing to relate the keywords to the appropriate category. Another reason might be the fact that the display of the categories in the search tree, in the browsing with the algorithm, are not in an alphabetical order, but rather according to relevance, i.e., by the weight of the category. Thus, subjects, who lack knowledge about auto spare-parts admitted getting lost in the categories and not really knowing where to look for the parts. Still, 55% of the subjects preferred the browse search with the algorithm, as compared to 35%, who preferred the browse search without the algorithm.

Table 3 also presents the results of the subjects' ratings for the GMSIM and ACDelco system in general after finishing the sub-experiment. The average general satisfaction is significantly higher in GMSIM than in ACDelco (z = -1.24, p < 0.1). However, we could not conclude anything about the difference in the time-frame satisfaction in both systems. The reason might be due to the fact that ACDelco enabled the subjects to utilize their search by using a browse search in some cases and a keyword search in other cases, while in GMSIM we instructed them on when to use keyword search and when to use browse search. Despite that, the ease and flexibility were significantly higher in GMSIM, in comparison to ACDelco (z = -2.17, p < 0.02).

4.2.2 Sub-experiment II: Neighborhood by Demographics and Preference Profile Algorithm

In this sub-experiment the subjects experimented with GMSIM using the algorithm with neighborhood by demographics- and preference-profile. Like Table 3, Table 4 presents the results of the average ratings the subjects gave, only this time to a different algorithm. The table also presents the average time duration and the corresponding standard deviation in all the searches, and the average number of clicks and its standard deviation. The results of the experiment supported our hypothesis (b) above. The general satisfaction of the keyword search with the algorithm was significantly higher than with the keyword search without the algorithm (z = -3.3, p < 0.001). The ease and flexibility were also significantly better in the keyword search with the algorithm, in comparison to the keyword search without the algorithm (z = -3.2, p < 0.001).

Table 4. Average results (*avg*) and standard deviation (*stdev*) of GMSIM's keyword and browse search, GMSIM, and ACDelco in general in the second sub-experiment. Satisfaction and flexibility varied between 1-7

	General Satisfaction		**Time Satisfaction**		**Ease and Flexibility**		**Average Time Duration**		**Average No. of Clicks**	
	avg	**stdev**	**avg**	**stdev**	**avg**	**stdev**	**avg**	**stdev**	**avg**	**stdev**
Keyword Search with alg.	6.50	0.61	5.55	1.32	6.45	0.89	15.50	7.51	2.74	1.61
Keyword Search w/o algorithm	4.80	1.47	4.30	1.42	4.75	1.62	19.00	7.53	7.36	5.14
Keyword Search – general	6.05	0.69	5.40	1.10	6.00	0.79				
Browse Search with alg.	5.40	1.10	5.25	1.62	5.15	1.66	19.15	8.13	6.78	2.16
Browse Search w/o algorithm	4.75	1.25	4.65	1.66	4.50	1.57	15.40	4.77	6.92	2.63
Browse Search – general	5.00	1.26	4.90	1.41	4.70	1.63				
GMSIM – general	5.75	1.02	5.50	1.00	5.90	0.72				
ACDelco	5.40	0.88	5.45	1.15	4.90	1.55				

The time-frame satisfaction using a keyword search with the neighborhood by demographics and preference profile algorithm was significantly higher than the level of satisfaction when using the keyword search without the algorithm (z = -2.43, $p <$ 0.007). This is supported by the fact that the average time duration using the keyword search with the algorithm was significantly lower than when using the keyword search without the algorithm ($t(19)$ = 1.65, $p < 0.1$) and that the average number of clicks was significantly lower in the keyword search with the algorithm than in the keyword search without the algorithm ($t(19)$ = 3.53, $p < 0.001$). At the end of the experiment, the subjects were explicitly asked which search technique they would prefer to use in the future: a keyword search with the neighborhood by demographics and preference algorithm or a keyword search without the algorithm. Significantly more subjects (85% of the subjects) preferred the keyword search with the algorithm, as compared to only 5%, who preferred the keyword search without the algorithm (z = -3.31, $p <$ 0.001). The other 10% had no preference.

The table also presents the results for the browse search. The general satisfaction using browse search with the neighborhood by demographics and preference profile algorithm was significantly higher that using browse search without the algorithm (z = -1.75, $p < 0.04$). The ease and flexibility using the browse search with the algorithm were significantly higher than without the algorithm (z = -2.0, $p < 0.02$). Also, the time-frame satisfaction was significantly higher in the browse search with the algorithm (z = -2.4, $p < 0.008$). However, no conclusion could be reached regarding the average number of clicks and the time duration in both searches.

Table 4 also presents the results of the subjects' ratings for the GMSIM and ACDelco system in general after finishing the sub-experiment. The average general satisfaction is significantly higher in GMSIM than in ACDelco (z = -1.13, $p < 0.13$). However, as in the first sub-experiment, we could not conclude anything about the difference in the time-frame satisfaction in both systems. Despite that, the ease and flexibility were significantly higher in GMSIM, in comparison to ACDelco (z = -2.67, $p < 0.003$).

4.2.3 Sub-experiment III: Using the Customer's History Algorithm

In this sub-experiment we used the customer's history algorithm. As the previous tables, Table 5 presents the average ratings the subjects gave to the general satisfaction, time satisfaction, and the ease and flexibility in both the keyword search with the history algorithm and without any algorithm, and to the keyword search in general. The table also presents the average time duration and the corresponding standard deviation in both searches, and the average number of clicks and its standard deviation. The third row in the table presents the average ratings the subjects gave to the keyword search in general. The results of the experiment supported our hypothesis (c) above. The general satisfaction of the keyword search with the history algorithm was significantly higher than with the keyword search without the algorithm (z = -2.48, $p < 0.006$). The ease and flexibility were also significantly better in the keyword search with the history algorithm, in comparison to the keyword search without the algorithm (z = -2.91, $p < 0.002$).

Table 5. Average results (*avg*) and standard deviation (*stdev*) of GMSIM's keyword search and browse search, GMSIM, and ACDelco in general in the third sub-experiment. Satisfaction and flexibility varied between 1-7

	General Satisfaction		**Time Satisfaction**		**Ease and Flexibility**		**Average Time Duration**		**Average No. of Clicks**	
	avg	**stdev**	**avg**	**stdev**	**avg**	**stdev**	**avg**	**stdev**	**avg**	**stdev**
Keyword Search with alg.	6.00	0.92	5.65	0.99	5.95	1.00	12.70	5.25	4.49	2.23
Keyword Search w/o algorithm	4.75	1.59	4.40	1.98	4.65	1.90	14.85	5.71	6.92	5.06
Keyword Search – general	5.60	0.75	5.45	1.00	5.45	1.00				
Browse Search with alg.	5.20	0.95	5.30	1.38	4.75	1.52	13.15	4.12	6.79	2.25
Browse Search w/o algorithm	4.05	4.05	4.25	1.68	3.95	1.70	16.85	7.96	8.63	4.10
Browse Search – general	4.45	0.89	4.60	0.99	4.25	1.07				
GMSIM – general	5.95	0.69	5.65	0.88	5.85	0.88				
ACDelco	4.80	1.70	4.65	1.79	4.45	1.88	14.60	6.80		

The time it took to perform a keyword search with the history algorithm was significantly lower than the time it took for the keyword search without the algorithm (*t(19)* = 1.39, $p < 0.1$). This also explains the fact that the time-frame satisfaction using a keyword search with the history algorithm was significantly lower than the level of satisfaction when using the keyword search without the algorithm (z = -2.23, $p < 0.01$). This is also supported by the fact that the average number of clicks was significantly lower in the keyword search with the history algorithm than in the keyword search without the algorithm (*t(19)* = 1.8, $p < 0.05$). At the end of the experiment, the subjects were explicitly asked which search technique they would prefer to use in the future: a keyword search with the history algorithm or a keyword search without the algorithm. Significantly more subjects (80% of the subjects) preferred the keyword search with the history algorithm, as compared to only 15%, who preferred the keyword search without the algorithm (z = -2.62, $p < 0.004$). The other 5% had no preference.

Results of the browse search are also shown in Table 5. The general satisfaction (z = -2.77, $p < 0.002$), the ease and flexibility (z = -2.1, $p < 0.02$), and the time-frame satisfaction (z = -2.1, $p < 0.02$) were significantly higher in the browse search with the customer's history algorithm than without the algorithm. This is supported by the fact that the average number of clicks ($t(19) = 1.81$, $p < 0.05$) and the average time duration ($t(19) = 1.84$, $p < 0.05$) were significantly lower in the browse search with the customer's history algorithm than without it. The results of the browse search with the customer's history algorithm were better than the previous algorithms used in the browse search. This can be explained by the fact that the customer's history algorithm also gives weight to the path of each item the customer had purchased before. Under the hypothesis that customers tend to buy items similar to items they had purchased in the past, assigning weight to the path of the items better helps the customers in their future searches.

Table 5 also presents the results of the subjects' ratings for the GMSIM and ACDelco system in general after finishing the experiment with the history algorithm. The average general satisfaction is significantly higher in GMSIM than in ACDelco (z = -2.78, $p < 0.002$). The time-frame satisfaction was also significantly higher in GMSIM, in comparison to ACDelco (z = -2.29, $p < 0.01$). Finally, the ease and flexibility were also significantly higher in GMSIM, in comparison to ACDelco (z = -2.81, $p < 0.002$). We can conclude that these results support our hypothesis that, all in all, in the third sub-experiment the subjects prefer using GMSIM to ACDelco.

4.2.4 Overall Results

We now discuss the combined results of all the sub-experiments. We compared our keyword search and browse search with any one of our algorithms to ACDelco. We asked the subjects which system they would prefer to use in the future, ACDelco or GMSIM? The results presented in Figure 6 show that significantly more subjects prefer using keyword search with algorithm in future searches (90% of the subjects) than using ACDelco in future searches (only 8% of the subjects) (z = -4.6, $p < 0.001$).

The results presented in Figure 6 also show that significantly more subjects in all of the experiments prefer to use a browse search with an algorithm in the future (55% of the subjects) and not ACDelco (only 38% of the subjects) (z = -1, $p < 0.16$). Although the results are significant, the difference in the percentages is not as high as in the keyword search. We stated some of the reasons for that in Section 4.2.1.

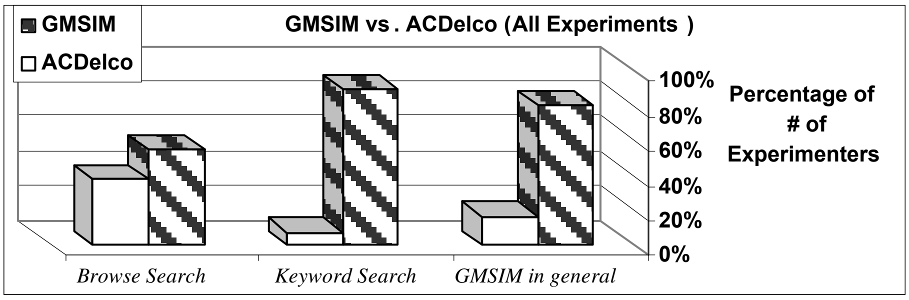


Figure 6. Preference of all the subjects concerning GMSIM vs. ACDelco

The results presented in Figure 6 also show that significantly more subjects (80% of the subjects) would prefer to use GMSIM in the future (regardless of whether they use a keyword search or a browse search), as compared to only 17%, who prefer to use ACDelco (z = -4.34, p < 0.001). These results are also supported by the data in Table 6 that presents the average results and standard deviation of all the subject ratings for the GMSIM system in general and for ACDelco. The general satisfaction using GMSIM is significantly higher than using ACDelco (z = -3.07, p < 0.001). Also, the time-frame satisfaction in GMSIM is significantly higher than in ACDelco (z = -2.03, p < 0.02). Finally, we can see that the ease and flexibility of GMSIM are significantly better than in ACDelco (z = -4.38, p < 0.001). We can conclude that all of the ratings for GMSIM are markedly higher than the ratings for ACDelco.

Together, the three sub-experiments supported all of our hypotheses and the main hypothesis that combining the algorithms (see formula 3 and Table 2 in Section 3.5) would yield better satisfaction and less effort than using a simple search system.

Table 6. Average results (*avg*) and standard deviation (*stdev*) of GMSIM in general and ACDelco, for all the experiments. Satisfaction and flexibility varied between 1-7

	General Satisfaction		**Time Satisfaction**		**Ease and Flexibility**	
	avg	**stdev**	**avg**	**stdev**	**avg**	**stdev**
GMSIM	5.72	0.94	5.53	0.95	5.75	0.88
ACDelco	5.05	1.37	5.05	1.59	4.62	1.75

5. Conclusions

In this paper we have proposed adding recommendation system logic and functionality to customer searches in online store systems. By this we suggested broadening the scope where information retrieval tasks are encountered, i.e., not just traditional document searches or other types of web page searches, but also online stores. We proposed that the online store system and the customer cooperate to identify the appropriate item, rather than independent functioning of the system. Thus, OSGS tries to bridge the gap between finding the right product the customer is looking for and the search time involved. We used algorithms from collaborative filtering and information retrieval to help customers in search for products, and have thus created a personalized online store, adapting itself to each customer in the system.

Providing such service and customer convenience seems to increase sales and benefits (Lohse et al. 1998, Sarwar et al. 2000), and time saving is a factor that is as significant as cost for on-line customers (Bellman et al. 1999). Thus, as long as the personalization is good enough and generates satisfactory results for the customer, the better the chances will be that the customer will visit the online store again and develop a long-term relationship with the retailer.

Our experiments demonstrate that OSGS outperforms other non-personalized search systems.

Although our system was adapted to the domain of auto spare-parts, adapting it to any domain is not difficult, but it does require some thinking with the stakeholders concerning the target audience of the system, adapting the algorithms and testing the system.

References

Aggrawal C C, Wolf J L, Wu K and Yu P S (1999) Horting Hatches an Egg: A New Graph-Theoretic Approach to Collaborative Filtering. In: Proceedings of ACM Knowledge Discovery and Data Mining (KDD-99), pp. 201-212.

Andersen's Internet Survey conducted between May 1, 2001 and May 6, 2001, and based on responses from 1,317 online users. Online User Shopping Habits Survey 9 (2001). http://www.andersen.com/website.nsf/content/MarketOfferingseBusinessResourcesOnlineUserPanelShoppingExperience (visited November, 2002).

Bellman S, Lohse G L and Johnson E J (1999) Predictors of Online Buying. Findings from the Wharton Virtual Test Market. Communications of the ACM, 42(12):32-38.

Brin S and Page L (1998) The Anatomy of a Large-Scale Hypertextual Web Search Engine. Computer Networks and ISDN Systems, 30(1-7):107-117.

CARAVAN Survey conducted in August 2001, and based on responses from 1,003 adults (2001) http://www.nclnet.org/shoppingonline/shoppingsurvey.htm (visited November, 2002).

Chen L and Sycara K (1998) WebMate: A Personal Agent for Browsing and Searching. In: Proceedings of the Second International Conference on Autonomous Agents (Agents'98), pp. 132-139.

Goffinet L and Noirhomme-Fraiture M (1995) Automatic Hypertext Link Generation based on Similarity Measures between Documents. Research Paper, RP-96-034, Institut d'Informatique, FUNDP. Available at http://www.fundp.ac.be/~lgoffine/Hypertext/semantic_links.html (visited November, 2002).

Good N, Schafer J B, Konstan J A, Borchers A, Sarwar B, Herlocker J and Riedl J (1999) Combining Collaborative Filtering with Personal Agents for Better Recommendations. In: Proceedings of AAAI-99, pp. 439-446.

Herlocker JL, Konstan J A, Riedl J (2000) Explaining Collaborative Filtering Recommendations. In: Proceedings of the ACM 2000 Conference on Computer Supported Cooperative Work, pp. 241-250.

Holte R C and Drummond C (1994) A learning apprentice for browsing. In: Working Notes of the AAAI Spring Symposium on Software Agents, pp. 37-42.

Kitts B, Freed D and Vrieze M (2000) Cross-Sell: A Fast Promotion-Tunable Customer-item Recommendation Method Based on Conditionally Independent Probabilities. In: Proceeding of the ACM SIGKDD International Conference, pp. 437-446.

Lieberman H (1995) Letizia: An Agent that Assists Web Browsing. In: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95), pp. 924-929.

Limthanmaphon B, Zhang Z and Zhang Y (2002) Querying and Negotiating Structured E-Commerce Activities. Fourth Agent Mediated Electronic Commerce Workshop.

Lin R (2002) Attaining Fast and Successful Search in E-Commerce Environments. A Master's Thesis, Bar-Ilan University.

Lohse G L and Spiller P (1998) Electronic Shopping: The Effect of Customer Interfaces on Traffic and Sales. Communications of the ACM, 41(7): 81-87.

McGinty L and Smyth B (2002) Deep Dialogue vs. Casual Conversation in Recommender Systems. In: Proceedings of the Second International Conference on Adaptive Hypermedia and Web-Based Systems (AH-2002), pp. 80-89.

Pazzani M J and Billsus D (2002) Adaptive Web Site Agents. Autonomous Agents and Multi-Agent Systems 5(2):205-218.

Sarwar B, Karypis G, Konstan J and Riedl. J (2000) Analysis of Recommendation Algorithms for E-Commerce. In: Proceedings of ACM E-Commerce, pp. 158-167.

Sarwar B, Karypis G, Konstan J and Riedl J (2001) Item-Based Collaborative Filtering Recommendation Algorithms. In: Proceedings of the Tenth International World Wide Web Conference on World Wide Web, pp. 285-295.

Shardanand U and Maes P (1995) Social Information Filtering: Algorithms for Automating "Word of Mouth". In: Proceedings of CHI-95 Conference on Human Factors in Computing Systems, pp. 210-217.

Siegel S (1956), ed. Nonparametric Statistics for the Behavioral Sciences. McGraw-Hill, New York.

Appendix: Performance Evaluation

After each search group (5 items) a questionnaire was given to the subjects. In it the subjects needed to rate three statements in the range of 1 to 7, where 1 indicates "I completely disagree with the statement", and 7 indicates "I completely agree with the statement". A sample questionnaire is presented below:

General Satisfaction with GMSIM							
General satisfaction with GMSIM.	1	2	3	4	5	6	7
Satisfaction with the time-frame for finding items.	1	2	3	4	5	6	7
Ease and flexibility of GMSIM.	1	2	3	4	5	6	7

Remarks: ____

____