Interactive Product Catalogue with User Preference Tracking

Sheng-Uei Guan[1] and Yuan Sherng Tay[2]
[1]School of Engineering and Design
Brunel University, Uxbridge, Middlesex, UB8 3PH, UK

[2]Department of Electrical and Computer Engineering
National University of Singapore
10 Kent Ridge Crescent, Singapore 119260

ABSTRACT

In the context of m-commerce, small screen size poses serious difficulty for users to browse effectively through a product catalogue, given the limited number of products that may be presented on-screen. Despite the availability of search engines, filters and recommender systems to aid users, these techniques focus on a narrow segment of product offering. The users are thus denied the opportunity to do a more expansive exploration of the products available. This paper describes a novel approach to overcome the constraints of small screen size. Through integration of a product catalogue with a recommender system, an adaptive system has been created that guides users through the process of product browsing. An original technique has been developed to cluster similar positive examples together to identify areas of interest of a user. The performance of this technique has been evaluated and the results proved to be promising.

Keywords:

Interactive Catalogue, Clustering, Genetic Algorithm, m-Commerce

---

1. INTRODUCTION

M-commerce possesses two distinctive characteristics that distinguish it from traditional e-commerce: the mobile setting and the small form factor of mobile devices. While the processing capabilities of mobile devices and the bandwidth of wireless networks may increase significantly, the size of a mobile device will remain largely unchanged due to the tradeoff between size and portability.

These two traits of m-commerce have serious implications on its usage. Small screen size and limited input capabilities pose a great challenge for developers to conceptualize user interfaces that have good usability while working within the size constraints of the device. In addition, the mobile setting of m-commerce is characterized by a disconcerting environment, where a multitude of external stimuli simultaneously vie for the attention of a user. The design of m-commerce interfaces also have to accommodate the limited concentration of a user [1].

In response to the limited screen size of mobile devices, there has been unspoken consensus that certain tools must be made available to aid users in coping with the relatively large volume of information. Recommender systems have been proposed for such a purpose [2]. Instead of having the user browse painstakingly through the product catalogue of a retailer, a recommender system could be used to narrow down the choices before presenting them to the user.

1.1. Catalogue Browsing

Online stores of today are generally equipped with search engines, filters and recommender systems to help user locate relevant products. These traditional information retrieval techniques are useful in finding items when a user has a clear picture of what is being sought. This ideal scenario however is not an accurate depiction of reality [3].

In a study conducted by Doug *et al.*, a new user behavior termed opportunistic exploration has been identified [4]. Opportunistic exploration is characterized by users having multiple ill-defined overlapping interests. In this state, users tend to view diverse items but examine few in detail. Throughout the course of browsing, exposure to items affect interests, and interest may evolve due to exposure or whim.

A similar line of thought was shared by Tateson *et al.* in their design of an interactive online catalogue named ShoppingGarden [5]. Again, the emphasis was that the paradigm of online shopping is fundamentally different from that of information retrieval. Tateson believed that the browsing process currently lacking in online shopping experiences should be reintroduced for the benefit of shoppers.

These studies establish the importance of having well-designed online catalogue that supports the shopping behavior of users. Yet the challenge of including such browsing capabilities in m-commerce is great, given that small screen size of mobile devices severely limits the number of products that may be presented on-screen.

Currently, the predominant strategy to overcome this problem is through careful organization of products into narrow categories. The downside of such an attempt has been well documented [6]. Users who do not understand or concur with the organization of a catalogue often find navigation through these artificially imposed categorizations counter-intuitive and frustrating.

The alternative solution is to have interactive catalogues such as the ShoppingGarden. Through user feedback mechanisms, interactive catalogues allow users to customize the browsing process in the way they want. This allows for fluid navigation of the product space, whereby users are given the freedom to redirect the browsing process as and when their interests change.

For the purpose of m-commerce application, the interface of such a catalogue must be kept simple to reduce the cognitive effort of the user. Feedback input should be straightforward and kept to a minimal so as not to disrupt the browsing process. Instead of having a user explicitly control browsing parameters, viable improvement would be to have a recommender system infer the preference of the user and adjust the browsing process accordingly.

1.2. Recommender System

Recommender systems are intelligent software that performs the role of sales agents by first understanding a user's preferences through querying and profiling, and subsequently presenting information or products of relevance to the user [7]. Recommender systems have long been regarded as a highly desirable feature of e-commerce. With the inception of m-commerce, the need for an efficient and reliable recommender has become more pressing.

Currently there are numerous ongoing studies to improve recommender technology in the context of e-commerce [8][9]. Yet the approaches of such studies are seldom directly applicable to the domain of m-commerce. One important reason for such lack of transferability is the tradeoff between user preference elicitation and recommendation quality. The quality of recommendation is directly correlated to how well a recommender system understands a user. In order to understand a user well however, the system has to demand feedback from the user, which obliges substantial user involvement.

In the case of e-commerce, the balance has traditionally favored higher recommendation quality. This is due to the belief that customers who heed the advice of inaccurate recommendations end up being frustrated [10]. This mentality stems from the architecture of existing recommender systems, whereby the recommender is often expected to behave as a proxy to the user in the choice making process. With such heavy responsibility placed on the recommender, extensive information must be made available before a recommender can make a sound judgment.

In reality however, the luxury of having the user provide comprehensive feedback is seldom feasible in the mobile setting of m-commerce. Owing to such a constraint, a “best effort” recommender system that make do with whatever information available will serve as an interesting alternative to the “best quality” emphasis of current recommendation technology. This is especially relevant when the need for an m-commerce recommender is to narrow down the choices instead of giving precise suggestions.

1.3. Research Objectives

We thus propose to have a product catalogue where browsing is directed by an integrated recommender system. The recommender system is to take incremental feedback in return for browsing assistance. Product appearance in the catalogue will be dynamically determined at runtime based on user preference detected by the recommender system.

The design of our hybrid m-commerce catalogue-recommender system was separated into two stages. To begin with, we investigated the typical constraints of m-commerce applications to conceptualize a suitable catalogue interface. This included modelization of the browsing process of our system. For the purpose of this study, the scope was restricted to the case of having Personal Digital Assistant (PDA) as the mobile device. This is due to the consideration that the larger screen size of PDAs as compared to mobile phones makes it a more suitable instrument for m-commerce.

Thereafter, a preference detection technique was developed to serve as the recommender layer of the system.

2. INTERFACE DESIGN

The interface of a catalogue is divided into three components: visual presentation, browsing process and feedback mechanism. The main objective of our design is to have a clean and simple interface that is intuitive to use and corresponds well to the browsing needs of a user.

2.1. Presentation

In designing a retail catalogue, proper product presentation is crucial to the browsing satisfaction of a user. Given the constraint of a PDA screen, the main concern of our design is to maximize emphasis on product presentation while simplifying the control elements. Visual elements are an important component of our screen design. Owing to the mobile setting, it is often difficult for users to concentrate on an m-commerce activity. Human cognition is more adapted to the processing of visual images as compared to textual information [11]. Visual elements are thus useful mechanisms to improve the usability of a catalogue. For example, the display of a product photo greatly reduces the cognitive effort for an unfamiliar user to associate with a product. In addition, the presence of product photos also contributes towards an appealing visual presentation, which is an essential ingredient to trigger impulse purchase [12]. Figure 1 shows a screenshot of the implemented user interface.

Figure 1: Screenshot

2.2. Browsing Process

The process of browsing through a catalogue is essentially an act of inspecting sets of items in a sequential manner. Browsing naturally induces a sense of flow, which may be imagined as a navigation process through the product space.

The main challenge in the design of such a navigation system is to define the relation of products with respect to one another. Differing standpoints of people dictate that each individual sees the product relations from a different perspective. It is thus imperative to custom define product relations in a way that is meaningful to each user. One method of doing so is through interactive critiquing of products [13]. Interactive critiquing involves allowing a user to express the goals that are not satisfied by current items.

Another method to understand the preference of a user is through clustering. Clustering is more commonly used in data mining operations to detect trends. In our case, clustering may be used to group items that receive similar feedback from a user in an attempt to identify the underlying pattern that matches the preference of the user.

Both interactive critiquing and clustering are means to help user navigate towards items that are of interest to the user. While interactive critiquing is an excellent technique for making fine adjustment in the browsing direction, its sharp focus on a single point in the product space makes it unsuitable for expansive browsing.

In our catalogue, one desirable feature is to have an adaptable focus that allows user to glance at the entire product range as well as zoom in on a few products of interest. We define two parameters in our browsing: breadth and preference. Breadth is a measure of diversity in the product presentation whereas preference is the inferred interest of the user.

Breadth needs to be changed according to the state of browsing. Initially, a large breadth should be used to expose a user to a wide range of products. As the user increasingly grasps some understanding of the available choices, breath should be narrowed down to focus on recommended products based on the user's preference. This allows the user to discover products of increasing interest, and at the same time facilitate a comparison of close alternatives to aid in the purchase decision. At any time, should a shift be detected in the user interest, breadth has to be relaxed accordingly to allow the user the possibility to explore again products of differing nature.

To implement such a mechanism, we divided each page of the catalogue into two portions; the first containing products recommended based on the detected preference of the user and the second containing randomly sampled products. Breadth is defined as the size of the latter portion.

In practice, it is not necessary to have completely random products make up the breadth portion. Instead, it is possible to define a browsing policy that dictates the default order of product presentation. Doing so gives the advantage of a possibly meaningful browsing even when user feedback is not forthcoming. In addition, retailers will be given the choice to bias the browsing process towards certain type of products. Alternatively, the choice of policy may be presented to the user for selection before the browsing process. Care must be taken however to ensure that a browsing policy offers sufficient diversity for the purpose of expansive browsing.

2.3. Feedback mechanism

A simple feedback mechanism serves to save space, encourage users to volunteer information as well as minimize disruption to the browsing process. In our case, we note that the most intuitive and compact feedback method is for a user to comment directly on the products on display, as proposed by Burke *et al.* in his case-based critiquing approach [14]. To further simplify the approach, we adopt a bipolar rating system where the user is allowed to click on any product to indicate interest.

Using the bipolar rating system, we obtain a set of selected products and its complement. Though both sets potentially contain useful information on a user's interest, our approach focuses solely on analyzing the selected products.

The principal motivation for this bias is negligence on the part of the user. This is especially relevant in the context of m-commerce where users are prone to being distracted by external stimuli. In addition, a user may not be familiar with the products being presented and is unlikely to meticulously examine every product before making an informed choice. Assuming non-selected products to be examples of negative interest will thus be unreliable.

A similar reasoning can be made based on the relativistic nature of product selection. Being unfamiliar to the product offering, a user tends to select the best available option. With greater exposure to relevant products, it is natural for a user to become more discerning in making a choice. Again, products that are not selected do not necessarily indicate disinterest.

In addition, emphasis on the selected products has the added benefit of improving responsiveness of the system to user input. In our design, we seek to have an engaging browsing process where user perceives a shift in the direction of browsing with every input provided. This rewards user for providing feedback and at the same time helps to instill confidence in the functionality of the system.

2.4. Prototype

A Java-based prototype of the catalogue was developed for testing purposes. The interface was designed to be easily presentable in HTML format. In an actual implementation, the catalogue software is to reside on a web server and remotely accessed via PDA.

3. PREFERENCE DETECTION

Clustering is the conceptual grouping of similar products. In the recommendation context, it is commonly used to identify trends in the data set to use as preference generalization [15]. For our case, we seek to identify a few dominant areas of interest associated with a user so as to find relevant products for recommendation. To do so, we perform clustering on the set of positive examples volunteered by the user. This process is synonymous to the identification of areas of interests associated with a user.

3.1. Product Ontology

Before any clustering could be done, it is essential to define the representation of each product in our system. This is done through the specification of an encoding scheme that maps products from the same category into a conceptual product space. The encoding scheme is responsible for the enumeration of product attributes and in so doing, determines the relationship between products.

While dynamic encoding schemes are possible [16], such an approach requires substantial user feedback in the attribute detection process. This requirement clearly contradicts our design objectives. For this reason, we adopted a static encoding scheme in the form of product ontology [17].

In our context, product ontology is simply a descriptive tree that defines the key attributes of each product category as well as their relevant enumeration schemes. Since each product category has its unique characteristics, such an implementation allows flexibility to have varied encoding scheme for different product categories. Figure 2 shows an example of a product ontology.

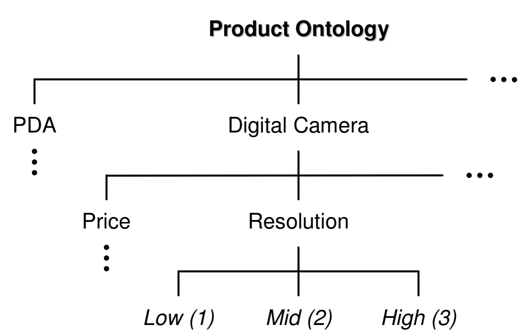

Figure 2: Product Ontology

3.2. Product Definition

Let p denote a product and P the product space such that $p \in P$.

A product is characterized by a set of attributes as well as their associated value. We define an attribute as a particular aspect of a product's characteristics (e.g. weight, color) while an attribute instance is a value taken by a product attribute (e.g. 100g, red).

Let α denote an attribute instance and A the domain of α such that $\alpha \in A$.

A product space P is defined as a vector space of η dimensions where η is the total number of unique attributes possessed by products in P.

$$\mathrm{P} : A_1 \times A_2 \times \cdots \times A_\eta$$

Products are mapped into the product space through a predefined product ontology. Products may then be represented by ordered η-tuples with the i^{th} value representing the attribute instance for the i^{th} attribute of the product. We shall refer to this η-tuple as the product characteristic.

$$p : \{\alpha_1, \alpha_2, \ldots, \alpha_\eta\} \quad , \alpha_i \in A_i$$

A product is assumed to be entirely characterized by the set of ordered attribute instances it is associated with.

3.3. Example

Given S a sequence of n user selections $S = \{ p_1 , p_2 , \ldots , p_n \}$.

Each product in S is a positive example of preferred products by the user. A cluster C is defined as a subset of S such that all products within the cluster have certain similarity to one another.

The objective of clustering is to group similar products together to form generalization of an area of interest. Generalization in this case is the identification of common traits of products within a cluster such that the set of common traits becomes a distinctive signature used to distinguish members of the cluster.

For example given a cluster C_1 of 3 products:

p_1 : [1 , 2 , 3 , 3 , 3 , 4]

p_2 : [2 , 3 , 3 , 3 , 4 , 1]

p_3 : [3 , 4 , 3 , 3 , 1 , 2]

An evident signature that may be derived is $\alpha_3 = \alpha_4 = 3$.

Thus given another 2 products:

p_4 : [4 , 1 , 3 , 3 , 2 , 3]

p_5 : [1 , 2 , 3 , 4 , 5 , 6]

We see that p_4 is a member of the cluster while p_5 is not.

By clustering p_1, p_2 and p_3 together, we generalize that the user may have an interest for products with $\alpha_3 = \alpha_4 = 3$. In considering whether to recommend p_4 or p_5 to the user, we observe that p_4 is a member of C_1 while p_5 does not belong to any cluster. We thus conclude that p_4 would be a better choice than p_5 since p_4 falls within a possible area of interest we have identified.

3.4. Cluster Definition

To facilitate the clustering of products, we adopt the concept of a schema proposed by John Holland in his Schema Theorem [17]. In our context, a schema is a template that specifies partially a set of product characteristic. This is possible with the introduction of wildcards that match with any value. A schema effectively defines a subset of the product space for all products that match with the schema.

Let χ denote a schema and X the schematic domain such that $\chi \in X$.

$X : G_1 \times G_2 \times \cdots \times G_\eta$ where $G_j = A_j \cup *$

$\chi : \{ \gamma_1 , \gamma_2 , \ldots , \gamma_\eta \}$ where $\gamma_j \in G_j$

To determine if a product p matches with a schema χ, we define the following functions:

$$\delta(\alpha,\gamma)=\begin{cases}1 & \alpha=\gamma \quad or \quad \gamma=* \\ 0 & else\end{cases} \tag{1}$$

$$\delta_{match}(p,\chi)=\prod_{j=1}^{\eta}\delta(\alpha_j,\gamma_j) \tag{2}$$

To illustrate the relevance of schema, let χ_1 = [* , * , 3 , 3 , * , *].

Carrying on from the previous example, χ_1 would be a schema that is equivalent to the signature $\alpha_3 = \alpha_4 = 3$. We thus observe that a schema serves as a useful means to define a cluster, providing both a signature to determine membership to the cluster as well as a definition of product similarity. Products within a cluster are similar in the sense that they match with the schema representative of the cluster.

In this paper, we shall adopt the schema as the sole definition of a cluster, $\chi \equiv C$. We term such an approach schematic clustering.

$$p \in C \Leftrightarrow \delta_{match}(p,\chi)=1$$
$$p \notin C \Leftrightarrow \delta_{match}(p,\chi)=0$$

3.5. Scoring

With the definition of cluster in place, the objective of preference detection may then be carried out through finding the best cluster that generalizes a sequence of user selection S. For this purpose, we need to be able to evaluate the relative quality of each possible cluster as a generalization of S.

3.6. Span

Let S be mapped into an $n\times\eta$ matrix { α_{ij} }, such that α_{ij} denotes the j^{th} attribute instance of the i^{th} product.

Adapting the match function (2) for use on a matrix,

$$\delta_{match}(p_i, \chi) = \prod_{j=1}^{\eta} \delta(\alpha_{ij}, \gamma_j) \tag{3}$$

We define span as the number of matches a schema has on a set of products,

Span

$$\sigma(S, \chi) = \sum_{i=1}^{n} \delta_{match}(p_i, \chi) \tag{4}$$

Span is a measure of the number of selected products present within a cluster. Given two clusters with different span, we derive greater confidence in the cluster with a larger span as an area of interest with greater significance. For example if a user selected 6 products, of which 5 belongs to cluster A while only 1 belongs to cluster B, we naturally conclude that cluster A serves as a better representation of the user's area of interest. Span thus serves as an important measure of quality.

3.7. Order

Given a schema, we define order as the number of non-wildcard values present in the schema.

$$\overline{\delta}_{wildcard}(\gamma) = \begin{cases} 1 & \gamma \neq * \\ 0 & \gamma = * \end{cases} \tag{5}$$

Order

$$d(\chi) = \sum_{j=1}^{\eta} \overline{\delta}_{wildcard}(\gamma_j) \tag{6}$$

Considering the definition of span, it is clear that the number of wildcards present in a schema is proportionate to the chances of the schema having a large span. However having too many wildcards may not be a desirable because it dilutes the interpretation of the area of interest.

For example, the null schema [*,*,…,*] is undoubtedly the schema with the largest span in any situation for it encompasses the entire product space. However the null schema does not gives any inference as to where the actual area of interest may lie. Assuming that a product fits the cluster [1,*,*,*,*] as well as the cluster [1,2,3,*,*], we see that the latter is a more precise interpretation of the area of interest because it has a more exclusive membership. Order thus serves as an equally important measure of quality as compared to span.

3.8. Coverage

Having established that span and order are two competing objectives, it is not possible to maximize both measures simultaneously. To discourage schemas with extreme span or order and favor those with a balance of the two, we define coverage as our main score matrix.

Coverage

$$\kappa(S,\chi) = \sigma(S,\chi) \cdot d(\chi) \qquad (7)$$

Score$_1$

$$\Gamma_1(S,\chi) = \kappa(S,\chi) \qquad (8)$$

The definition of coverage gives rise to a common situation where the multiple schemas share the same score. This is usually through the sacrifice of order for an equivalent increase in span or vice versa. Though each of these schemas represent a reasonable interpretation in reality, it is useful to further distinguish the quality of these schemas in the event that a single recommendation is required.

To do so, we have to decide whether to give greater priority to span or order. Since span represents a measure of the level of confidence in an area of interest, we adopt a prudent approach by giving it a higher priority.

Score$_2$

$$\Gamma_2(S,\chi) = \kappa(S,\chi) + \mu \cdot \sigma(S,\chi) \qquad (9)$$

where $0 < \mu < 1$

3.9. Noise Correction

In the context of data processing, data is often distorted by a certain level of noise due to uncontrollable factors. In our case, noise may be introduced either due to ignorance on the part of the user, or the lack of appropriate choices for the user to express freely a preference.

Suppose that the user has a strong preference for a product p:[1,2,3,4,5]. Due to noise, the user selections differ slightly from the actual preference.

p_1 : [2 , 2 , 3 , 4 , 5]
p_2 : [1 , 3 , 3 , 4 , 5]
p_3 : [1 , 2 , 4 , 4 , 5]
p_4 : [1 , 2 , 3 , 5 , 5]
p_5 : [1 , 2 , 3 , 4 , 6]

Using the existing measure, the best possible schema can only achieve a score of 7.5. The noise that has been introduced prohibits a perfect match between the actual preference and the selected products.

Schema	**Span**	**Order**	**Coverage**	**Score$_3$***
[* , * , 3 , 4 , 5]	2	3	6	7
[* , * , * , 4 , 5]	3	2	6	7.5
[1 , 2 , * , * , *]	3	2	6	7.5
[1 , 2 , 3 , * , *]	2	3	6	7

* with constant $\mu = 0.5$

To overcome this limitation, we introduce a noise threshold *K* to relax the condition for a match between a schema and a product.

$$\gamma(p_i,\chi)=\sum_{j=1}^{\eta}(1-\delta(\alpha_{ij},\gamma_j)) \tag{10}$$

$$\delta'_{match}(p_i,\chi)=\begin{cases}1 & \gamma(p_i,\chi)\le K\\ 0 & \gamma(p_i,\chi)>K\end{cases} \tag{11}$$

where $0 \le K < \eta$

Span'

$$\sigma'(S,\chi)=\sum_{i=1}^{n}\delta'_{match}(p_i,\chi) \tag{12}$$

With such an allowance given for noise, the scoring system will be able to pick up the optimum schema that matches the user preference. This is because the noise threshold allows schemas to be credited for partial matches with the selected products.

Schema	**Span'**	**Order**	**Coverage**	**$Score_3$***
[1 , 2 , 3 , 4 , 5]	5	5	25	27.5

* with constant $\mu = 0.5$, $K = 1$

With the introduction of the noise threshold, it becomes necessary to revise the previous definition of coverage.

Consider the following example,
p_1 : [1 , 2 , 3 , 4 , 1]
p_2 : [1 , 2 , 3 , 4 , 2]
p_3 : [1 , 2 , 3 , 4 , 3]

Schema	**$Span_2$**	**Order**	**Coverage**	**$Score_3$***
[1 , 2 , 3 , 4 , 1]	3	5	15	16.5
[1 , 2 , 3 , 4 , 2]	3	5	15	16.5
[1 , 2 , 3 , 4 , 3]	3	5	15	16.5
[1 , 2 , 3 , 4 , *]	3	4	12	13.5

* with constant $\mu = 0.5$, $K = 1$

Owing to the noise threshold, ambiguity appears in the assessment of schemas. A schema that takes advantage of the threshold term in an unwarranted context stands to gain a higher coverage. One main reason is the simple definition of coverage as a product of span and order, which gives unnecessary credit to schema values that do not match the actual attribute instance value.

$$\delta_{pt}(\alpha,\gamma)=\begin{cases}1 & \alpha=\gamma \;\; and \;\; \gamma\neq * \\ 0 & else\end{cases} \tag{13}$$

Coverage

$$\kappa'(S,\chi)=\sum_{i=1}^{n}\sum_{j=1}^{\eta}\delta'_{match}(\alpha_{ij},\gamma_j)\delta_{pt}(\alpha_{ij},\gamma_j) \tag{14}$$

Schema	**Span'**	**Coverage'**	**Score$_3$***
[1 , 2 , 3 , 4 , 1]	3	13	14.5
[1 , 2 , 3 , 4 , 2]	3	13	14.5
[1 , 2 , 3 , 4 , 3]	3	13	14.5
[1 , 2 , 3 , 4 , *]	3	12	13.5

* with constant μ = 0.5, K = 1

With the redefinition of coverage, there is an improvement in the score to give less emphasis to matches that makes use of the noise threshold. This is illustrated below for the case of χ_1:[1,2,3,4,1], where only values in bold are considered in the coverage.

p_1 : [**1** , **2** , **3** , **4** , **1**]		p_1 : [**1** , **2** , **3** , **4** , **1**]
P_2 : [**1** , **2** , **3** , **4** , **2**]	→	P_2 : [**1** , **2** , **3** , **4** , 2]
P_3 : [**1** , **2** , **3** , **4** , **3**]		p_3 : [**1** , **2** , **3** , **4** , 3]

Despite having a more equitable score, the redefined coverage is still incapable of differentiating between the sensible use of the noise threshold to accommodate noise or the abuse of it to increase coverage.

Observing the example on the coverage of χ_1:[1,2,3,4,1], what caused the problem is the misuse of the noise threshold to cover the additional α_{14} = 1. This coverage is unreasonable because α_{14} is the sole instance of a_4 with a value of 1. In this case, no generalization is possible because it is not justifiable to infer an area of interest with only one example.

To correct this error, the approach adopted is the inclusion of a penalty term to penalize the usage of the noise threshold.

Penalty

$$\pi(S,\chi)=\sum_{i=1}^{n}\delta'_{match}(p_i,\chi)\gamma(p_i,\chi) \qquad (15)$$

Score

$$\Gamma(S,\chi)=\kappa'(S,\chi)+\mu\cdot\sigma'(S,\chi)-\lambda\cdot\pi(S,\chi) \qquad (16)$$

where $0 < \mu < 1$, $\lambda > 1$

Schema	Span'	Cover'	Penalty	Score*
[1 , 2 , 3 , 4 , 1]	3	13	2	11.5
[1 , 2 , 3 , 4 , 2]	3	13	2	11.5
[1 , 2 , 3 , 4 , 3]	3	13	2	11.5
[1 , 2 , 3 , 4 , *]	3	12	0	13.5

* with constant $\mu = 0.5$, $\lambda = 1.5$, K = 1

The choice of a penalty coefficient λ greater than 1 is an important criterion to penalize the coverage of a sole attribute instance.

3.10. Emphasis

Finally, we recognize that a user's preference may evolve in the course of browsing. Products that were selected more recently are thus likely to be more in line with the current preference of the user. To take this factor into account, we allow a progressive emphasis to be set on more recent selection.

We define *E(i)* the emphasis factor on a product p_i as a function of the product index in the sequence of user selection S. The function may follow either a linear or a geometric progression depending on the desired degree of emphasis. The emphasis factor is then applied to all application of the match function (11). The optimal emphasis varies in different context. Though a high degree of emphasis improves the responsiveness of the system, the tradeoff is poorer overall generalization. It is thus advisable to use moderate values of E(i) in most circumstances. Empirical trial tests must be carried out to investigate the effect of a chosen emphasis.

3.11. Non-Linearity

The nature of clustering techniques necessitates the definition of similarity metric. Traditionally, this is the Euclidean distance between items mapped into a common vector space. Weights are then assigned to each item attribute to reflect their varying importance in the decision making process of a user. This approach assumes the additive independence of attributes, whereby the value of an item is broken down to the sum of individual attributes. Such a simplification however, is incorrect in many cases [19].

For example, a user may like the color red but only prefer red shirts and not red pants. Assigning a high importance to the color red without considering it in the context of other attributes will thus lead to inaccurate predictions.

In our case, the design of our clustering algorithm allows for this traditional limitation to be overcome. Through the detection of repetitive patterns instead of using a distance function, the algorithm is able to detect non-linear preferences in the user selection.

4. GLOBAL OPTIMIZATION

Having defined a scoring function to evaluate the relative superiority of each schema, we seek to design an algorithm to search for the best schema given a sequence of user selection.

Global optimization is the task of finding the best set of parameters to optimize an objective function. Traditional optimization techniques rely on mathematical properties of an objective function to carry out either gradual improvement of a given solution or a divide and conquer search. These approaches generally do not work on objective functions that are discontinuous and non-differentiable. To overcome such limitations, more adaptive algorithms have thus been proposed that use heuristics to search for a good solution [20].

In view of the combinatorial nature of our scoring function, it is of interest to make use of heuristic optimization algorithms for global optimization. At present, the options consist of three main algorithms: Tabu Search, Simulated Annealing and Evolutionary Algorithms.

4.1. Choice of Technique

Tabu Search (TS) works by exploring progressively the solution space for better solution while avoiding solutions that have already been visited. Simulated Annealing (SA) modelizes the manner in which metals recrystalize in the process of annealing. Evolutionary Algorithm (EA) is a generational population based approach inspired from natural selection.

Out of the three algorithms, EA was found to be a more appropriate choice in our context. In particular, we chose Genetic Algorithm (GA) which is a form of EA for the optimization of our scoring function. To justify this, we compare the nature of the problem of schematic clustering to Schema Theorem [18], which is the underlying principal of GA.

The main heuristic nature of GA lies in the association of schemas with good solutions. Instead of evaluating schemas directly, their qualities are inferred through the value of the solutions they are associated with. This is the main source of complexity in the algorithm, as plenty of trials and errors are required to identify a good schema. In our case however, the solution we are seeking is a schema itself. The scoring function is thus able to assess each schema according to its own merit. The heuristic nature of the GA is thus extensively lessened. We thus have reasons to believe that GA will perform exceptionally well in our context.

In addition, GA is the only population based approach out of the above mentioned searches. This gives it the capability to exhibit parallelism in its search by simultaneously localizing its search in multiple regions of the solution space. This is a useful property not only for avoiding local minima, but also to produce a range of optimal points whenever a unique global optimum does not exist. In our context, it is possible that there is more than one interpretation for a given set of user input. GA thus has the capability to identify all the interpretations in a single run.

4.2. Genetic Algorithm

By assigning a value of zero to the wildcard, the η-tuple of positive integer values of a schema is encoded directly into a chromosome as an array of integers. In this case, there is no difference between a phenotype and a genotype. Figure 3 illustrates the encoding process.

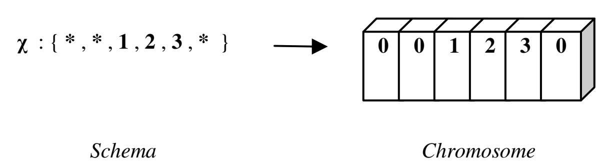

Figure 3: Genetic Encoding

Having defined the chromosomes, we apply the typical genetic algorithm as summarized in Figure 4. Evaluation is done using the scoring function defined in the previous section. The more unique features in our implementation included an initialization with random schemas from the product selection. Kin competition was introduced to maintain diversity in the genetic pool. This is crucial to offset the excessive selective pressure exerted by a strictly fitness based survivor selection strategy. In addition, we use mutation strategies that restrict mutation to only values meaningful in product selection or to the wildcard value.

```
INITIALIZE population with random candidate solutions
repeat until TERMINATION CONDITION
    1. EVALUATE chromosomes
    2. SELECT parents
    3. RECOMBINE pairs of parents
    4. MUTATE offspring
    5. EVALUATE offspring
    6. SELECT survivors to next generation
```

Figure 4: GA Pseudocode

4.3. Performance

To determine the performance of the algorithm, we define accuracy and efficiency as the performance measures. Accuracy is the frequency that results produced by the genetic algorithm matches the actual global optimum. We calculate accuracy as the average percentage of such matches.

On the other hand, efficiency is the amount of computational effort required to execute the algorithm. This is proportional to the number of iterations taken by the genetic algorithm to produce an output. We note that evaluation of a schema is the elementary task being iterated in the algorithm. Since the number of chromosomes is fixed, efficiency of the algorithm is thus solely determined by the number of evolutionary generations. We thus calculate efficiency as the average number of generations.

4.4. Parameter Choice

In the design of a GA, there is no established method for finding suitable parameters to the algorithm. Appropriate choices have to be made based on experience and their working validated empirically through trial and error. After analyzing the context of our application, we started with a series of trial tests to determine suitable parameters for an initial working configuration. Table 1 shows the configuration produced by the trail tests.

Genetic Domain:	{ 7, 7, 7, 7, 7, 7, 7 }
No. of Chromosomes:	50
Parent Selection:	Rank Selection
Crossover:	Single Point
Mutation Strategy:	Meaningful Values
Mutation Rate:	Linear [0.15, 0.5] (step:0.1)
Elitism Strategy:	Strict
Termination Condition:	Steady Fitness (5 gen) (±5 pt)
Kin Competition:	Yes: - 10 %/kin

Table 1: Initial Configuration

With the initial configuration, a long test was run to measure its performance. This accurate measurement serves as a benchmark for comparison with future modifications to the configuration. Thereafter, a series of short tests were conducted by varying each parameter of the initial configuration individually. The results were then analyzed in the context of one another to come up with a final configuration shown in Table 2.

Genetic Domain:	{ 7, 7, 7, 7, 7, 7, 7 }
No. of Chromosomes:	50
Parent Selection:	Tournament Selection (3)
Crossover:	Uniform
Mutation Strategy:	Wildcard
Mutation Rate:	Linear [0.15, 0.5] (step:0.1)
Elitism Strategy:	Strict
Termination Condition:	Steady Fitness (5 gen) (±5 pt)
Kin Competition:	Yes: - 25 %/kin

Table 2: Final Configuration

Finally a long test was conducted on the final configuration to compare its performance against the benchmark. Marked increase in performance was seen in the final configuration, with average efficiency improving from 16.0 generations to 13.3 generation and average accuracy rising from 87.0% to 96.8%. Figure 5 and 6 illustrate the performance gain.

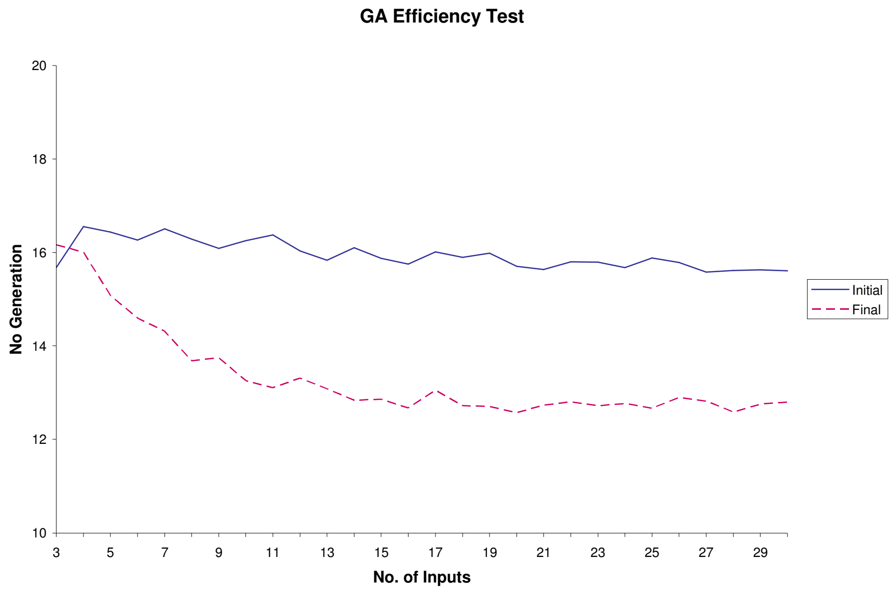

Figure 5: Efficiency Gain

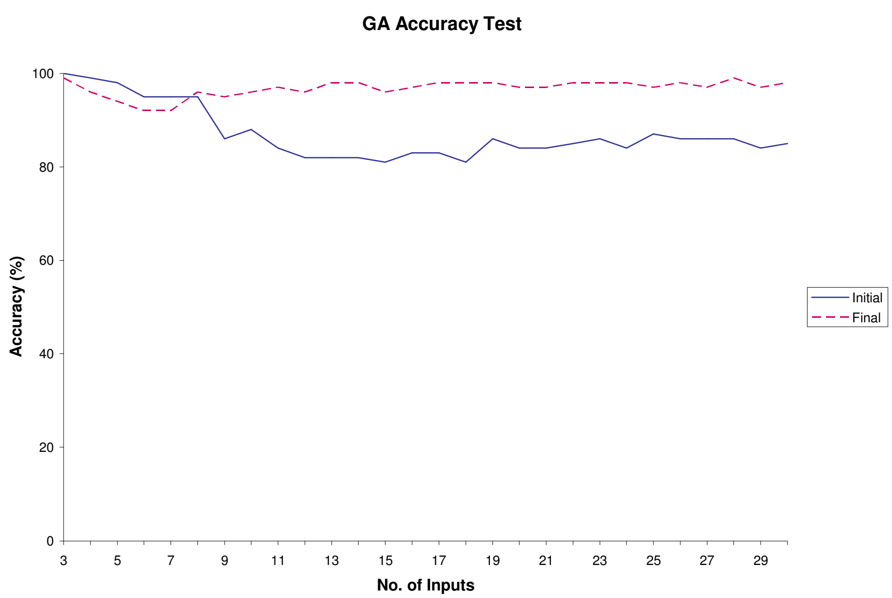

Figure 6: Accuracy Gain

5. SYSTEM EVALUATION

To evaluate the effectiveness of our catalogue, a test module was developed that simulates the response of users. For each simulated user, a preference is generated based on a random subset of attributes from a random product in the database. The simulated user then browses through the catalogue, selecting products that bear resemblance to the preference.

Two error parameters were defined for the simulated user. Match tolerance *m* determines the probability of a product being selected depending on the number of feature mismatch ε it has with the preference. P(selection of product) = m^{ε}. Negligence *n* determines the probability of a non-matching product being selected and conversely for a matching product being ignored.

For the purpose of testing, a product database was created as a replica of 3 categories of products available from an online shopping search engine – BizRate [21]. The categories are: Digital Cameras *(249 entries)*, MP3 Players *(209 entries)* and PDAs *(95 entries)*.

To assess the performance of the recommender, we measure the likelihood of the recommender in arriving at a reasonable interpretation of a user's preference. This likelihood depends on the number of pages that a user has browsed through. The longer a user interacts with the catalogue, the more information available to the catalogue for the understanding of the user. We thus define accuracy as the probability of a successful preference interpretation at a given browsing depth. Browsing depth is defined as the number of catalogue pages a user has browsed through.

5.1. Test Strategy

Through a series of experiments, we aimed to investigate the performance of the catalogue under different circumstances and analyze the factors that affect performance. Such factors can be classified into three categories: user related, database related and algorithm related.

User related factors are those associated with the erratic behavior of a user. This is modelized by the two error parameters defined in a simulated user. On the other hand, database related factors are those such as the diversity of products and the number of attributes of each product. These define the complexity of navigation through a product space. Finally, algorithm related factors are caused by inherent limitation of the schematic clustering algorithm as well as the performance of the genetic algorithm.

5.2. Category Profiling

The strongest factors that limit the performance of the catalogue are database related factors. Certain categories of products are intrinsically more difficult to browse through using the browsing process of our catalogue. It is thus important to measure the suitability of each category to ensure that the catalogue is not used where it is not appropriate.

We carried out a set of profiling test to determine the characteristics of the three product categories we used. Since this is a test of the theoretical performance limit for each category, both error parameters set to zero. Figure 7 shows the result of the test.

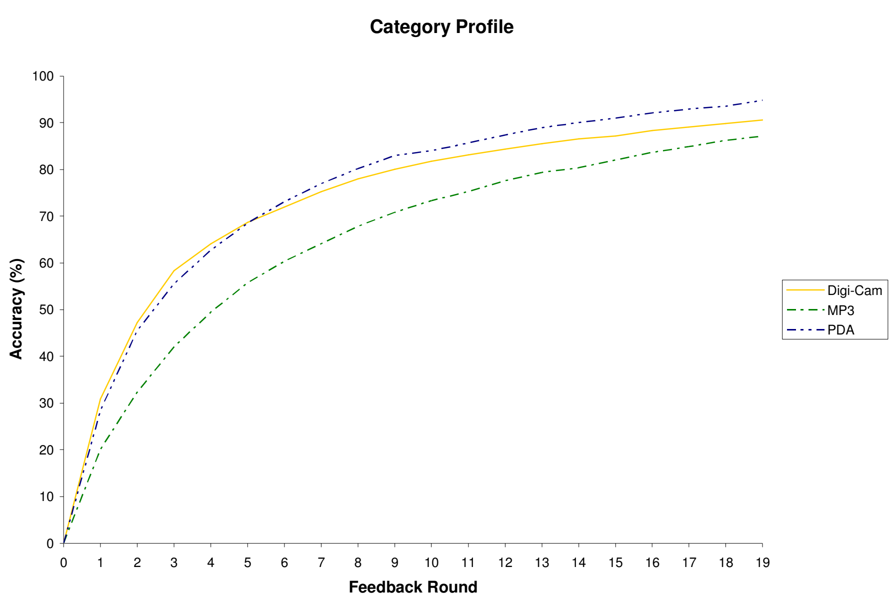

Figure 7: Category Profile

From the graph, the PDA category gives the best overall performance while the MP3 category is definitively more difficult to browse compared to the other categories. The Digi-Cam category however displayed good performance at low browsing depth but lagged behind in later part of the test. The initial lead may be attributed to a large pool of products with similar counterparts while the eventual lag implies the presence of unique products that bears no resemblance to others.

5.3. Error Tolerance Test

With the category profiles in place, simulations were carried out with different parameter settings to determine the effect of match tolerance and negligence. For each setting, 1000 simulations were carried out with the preference changing every 10 repetitions.

From the simulations, it was found that negligence has little impact except for extending the depth of browsing required for the detection of the preference. On the other hand, match tolerance represents inaccurate input that confuses the catalogue as to the real intent of the user. As such, the maximum performance of the catalogue is heavily influenced by the level of match tolerance. Figure 8 shows the effect of match tolerance on the accuracy of preference interpretation.

To understand better the causes of error brought about by high match tolerance, we logged the simulated browsing process in the cases of failed browsing attempt where the preference of the user is not detected even after 20 rounds of feedback. From the log, we observed a correlation between user preference and the number of failed browsing attempts. The majority of failed attempts were concentrated on a few preferences that are harder to detect than others. We thus repeated the experiment to confirm this observation. By keeping track of the number of failed attempts for each preference, we tabulated the error distribution as shown in Figure 9.

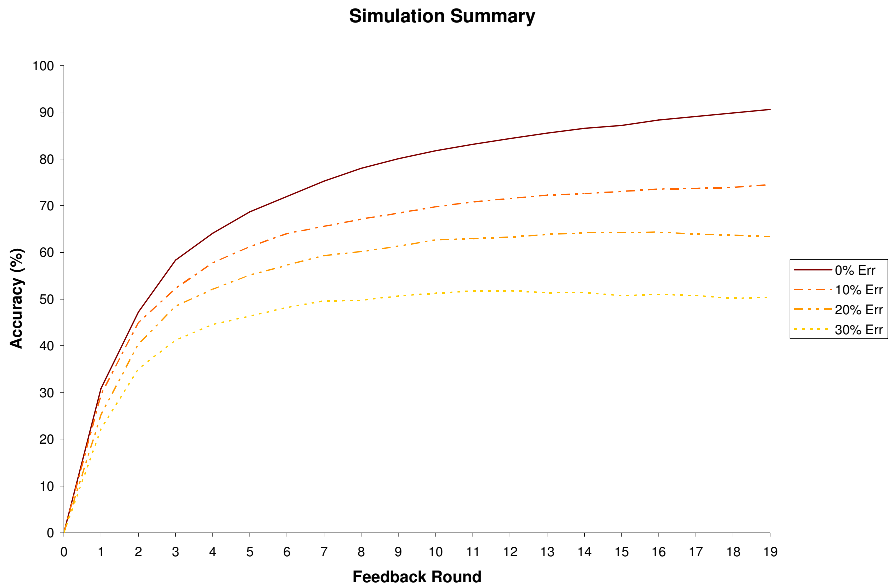

Figure 8: Match Tolerance Test

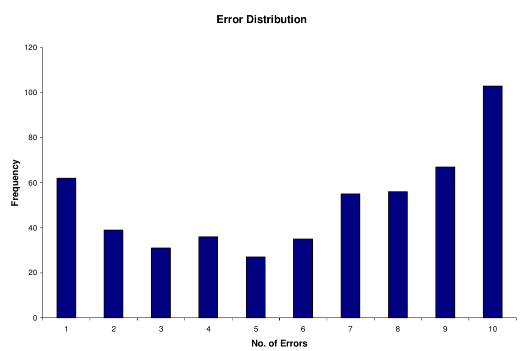

Figure 9: Error Distribution

From the error distribution, we see that preferences with 7 failures and above clearly form the majority of failed browsing attempts. By referring back to the browsing log, we concluded that such failures are attributed to users having unique preferences that can hardly be found in the database. Such failures account for 55% of failed browsing attempts.

5.4. Preference Change Test

Finally, the adaptability of the catalogue to accommodate changing preferences was also put under test. In this experiment, a set of two different preferences was used for each simulation. The first preference was used for the initial 5 rounds of feedback, after which the test module switches to the second preference. Error parameters were neglected for such a test. Figure 10 shows the result of this preference change simulation.

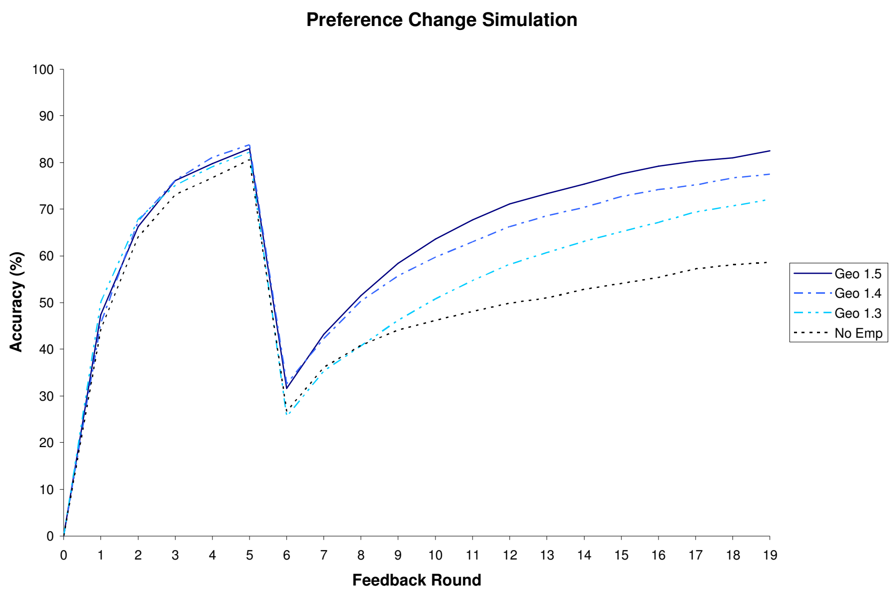

Figure 10: Preference Change Simulation

From the graph, it is clear that the rate at which the catalogue adapts to changing preference depends on the emphasis factor of the clustering algorithm. A higher emphasis serves to improve the responsiveness of the recommendation at the expense of better preference generalization in the long run.

6. RELATED WORK

The approach in this study focused on realizing the possibility for more complete m-commerce environment. This outlook is shared by other researchers who attempt to tackle the same problem with different strategies. For example a study by Guan *et al.* highlighted the advantages of using agent technology to bring about better user experience in m-commerce [22]. To the best of our knowledge, a customized catalogue for m-commerce has not been conceived. In the domain of traditional e-commerce however, the attempt to overcome the limitation of the traditional online catalogue has led to the conception of many interesting designs.

Among them, Bryan *et al.* came up with the Aquarium metaphor, which is a novel visual metaphor for opportunistic exploration [3]. The concept works by having a collection of products in a catalogue behave like fishes in an aquarium. Products enter and leave the aquarium automatically, and customers may choose either to watch passively the aquarium changing or interact with it. The result is a relaxed exploration of the product space that requires low cognitive overhead and removes the need to understand complex information structure.

In a similar approach by Tateson *et al.*, a metaphor called ShoppingGarden was used [4]. A garden is an area within which a shopper will cultivate a collection of pleasing items. The shopper opens a garden with a more or less specific descriptor. Items that fit with the descriptor are initially selected to fill the garden. A shopper may observe passively gradual change in the garden, or at any time reward an item to give it the advantage to breed and fill the garden with similar items. This approach creates a profile of the shopper over the course of a single session without any explicit questionnaire or form filling. Tateson believes that this metaphor serves the needs of a naïve customer wishing for a more expansive browsing experience.

Compared to the two approaches above, this study shares the same intent to make shopping a more pleasant experience for users. Owing to the initial motivation for m-commerce application, we targeted the development of such ideas in a different direction.

Our approach differs in the absence of a passive viewing mode, as the context of m-commerce makes it unfeasible for users to concentrate on the screen for an extended period of time. Interaction control was greatly simplified in our catalogue. Through the usage of recommender technology, we streamlined the browsing process by using a reduced form of feedback. People are good at identifying what they want when they see it. We thus believe that a good strategy lies in facilitating rapid catalogue browsing instead of having more elaborate controls.

7. CONCLUSION

In summary, this paper highlighted the need for specialized applications in the domain of m-commerce. In particular, the need for expansive browsing as a complement to existing search and filter functions has been emphasized.

As a possible solution, a novel method of product catalogue navigation with the aid of a recommender system has been proposed. This approach emphasizes a minimal-attention user interface that allows user to browse through a catalogue quickly with as little cognitive effort as possible. The associated recommender system that has been conceived adopts a best effort strategy that accommodates any level of user participation. It has been shown to be capable of detecting non-linear preferences in a set of incremental feedback, as well as tolerate noisy input produced by a user.

One drawback of this design is the danger of using predefined product ontology in the enumeration of attribute instances. This leads to stereotypic preference interpretation whose relevance depends largely on how the product ontology is defined. Such a criticism is indeed valid. However given the limited feedback that is obtainable from a user, it is crucial for the recommender to make use of some sort of domain knowledge to make it possible for any reasonable interpretation to be done at all. Since each domain has its unique features, such information will have to be provided from an informed source.

For future improvement, it may be worth investigating the possibility of having the recommender generate the ontology from the collective feedback of an ensemble of users. However it is unlikely that such inferred ontology will prove to be superior to a well crafted ontology by experienced sales personnel.

A more feasible enhancement to the existing system would be to incorporate Fuzzy Logic into the enumeration process. Doing so eliminates the problem around segment boundaries where similar attribute values may be arbitrarily classified into different clusters.

REFERENCES

[1] P. Tarasewich, "Designing Mobile Commerce Applications", *Communications of the ACM*, Vol. 46, No. 12, 57-60, 2003.

[2] S. Feldman, "Mobile Commerce for the Masses", *IEEE Internet Computing 4*, 75-76, 2000.

[3] C. Olston, E. H. Chi, "ScentTrails: Integrating Browsing and Searching on the Web", *ACM Transactions on Computer-Human Interaction*, Vol. 10, No. 3, 177-197, 2003.

[4] D. Bryan, A. Gershman, "Opportunistic Exploration of Large Consumer Product Spaces", *Proceedings of the 1st ACM conference on Electronic commerce*, 41-47, 1999.

[5] R. Tateson, E. Bonsma, "ShoppingGarden - Improving the Customer Experience with Online Catalogues", *BT Technology Journal*, Vol. 21, No. 4, 84-91, 2003.

[6] J. Y. Lee, H. S. Lee, P Wang, "An Interactive Visual Interface for Online Product Catalogs", *Electronic Commerce Research*, Vol. 4, 335-358, 2004.

[7] J. B. Schafer, J. Konstan, J. Riedl, "E-Commerce Recommendation Applications", *Data Mining and Knowledge Discovery*, Vol. 5, 115-153, 2001.

[8] J. A. Konstan, “Introduction To Recommender Systems: Algorithms and Evaluation”, *ACM Transactions on Information Systems*, Vol. 22, No. 1, 1-4, 2004.

[9] M. Montaner, B. Lopez, J. Lluis, “A Taxonomy of Recommender Agents on the Internet”, *Artificial Intelligence Review*, Vol. 19, 285-330, 2003.

[10] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, “Analysis of Recommendation Algorithms for E-Commerce”, *Proceedings of the 2nd ACM Conference on Electronic Commerce*, 2000.

[11] J. Y. Lee, H. S. Lee, P. Wang, “An Interactive Visual Interface for Online Product Catalogs”, *Electronic Commerce Research*, Vol. 4, 335-358, 2004.

[12] P. K. Kannan, A. M. Chang, A. B. Whinston, “Wireless Commerce: Marketing Issues and Possibilities”, Proceedings of the 35rd Hawaii International Conferenceon System Science, 2001.

[13] R. D. Burke, “Interactive Critiquing for Catalog Navigation in E-Commerce”, *Artificial Intelligence Review*, Vol. 18, 245-267, 2002.

[14] R. D. Burke, K. J. Hammond, B. C. Young, “The FindME Approach to Assisted Browsing”, *IEEE Expert: Intelligent Systems and Their Applications*, Vol. 12, No. 4, 32-40, 1997.

[15] G. D. Ramkumar, A. Swami, “Clustering Data Without Distance Functions”, *IEEE Data Engineering Bulletin*, Vol. 21, No. 1, 9-14, 1998.

[16] S. U. Guan, P. C. Tan, T. K. Chan, “Intelligent Product Brokering for E-Commerce: An Incremental Approach to Unaccounted Attribute Detection”, *Electronic Commerce and Research Applications*, Vol. 3, No. 3, 232-252, 2004.

[17] B. Smith, “Ontology”, *Blackwell Guide to the Philosophy of Computing and Informatio*n, Oxford, Blackwell, 155-166, 2003.

[18] J. H. Holland, “Adaptation in Natural and Artificial Systems”, *The University of Michigan Press*, Ann Arbor, 1975.

[19] L. Chen, P. Pu, “Survey of Preference Elicitation Methods”, *Technical Report No. IC/200467*, Swiss Federal Institute of Technology in Lausanne, 2004.

[20] R. L. Rardin, R. Uzsoy, “Experimental Evaluation of Heuristic Optimization Algorithms: A Tutorial”, *Journal of Heuristics*, Vol. 7, No. 3, 261-304, 2001.

[21] www.bizrate.com, 2004.

[22] S. U. Guan, C. S. Ngoo, F. M. Zhu, “Handy Broker: an Intelligent Product-Brokering Agent for m-Commerce Applications with User Preference Tracking”, *Electronic Commerce Research and Applications*, Vol. 1, 314-330, 2000.