PrefShop - a web shop with user preference search capabilities

Branislav Vaclav, Alan Eckhardt, Peter Vojtáš

Department of Software Engineering, Charles University
brano.vaclav@gmail.com, (eckhardt,vojtas)@ksi.mff.cuni.cz

Abstract

Abstract. The main topic of this paper is description of a proposal of a web shop with user preference searching - PrefShop. A typical web shop was implemented, but new capabilities were added to help the user with finding desired object. Besides preference search, visual hints that clarify the object relevance or the lack of relevance are proposed.

1. Introduction

E-commerce is nowadays ubiquitous on the web. Buying things online is becoming more and more common. The arising problem is the increasing number of goods available and the traditional approaches to product search. The traditional approach is based on restricting the attributes of objects, thus decreasing the number of displayed objects.

In this paper, we propose a more flexible approach to product search, which is more convenient and user-friendly. The proposed PrefShop enhances user experience in several ways, making easier to find the most desired object. The main focus is on displaying the information to the user and giving the user the possibility to specify his preferences more exactly.

2. Related work

The main area of this paper is to support the user searching for a desired object. This topic was addressed mainly in web information retrieval (WIR). WIR differs mainly in the environment – searching for a web page has some particularities. The main difference is that we are dealing with nicely structured objects. Notebooks have processors, RAM, hard disks, displays etc. Web pages have only text content. The paradigm of keyword search is prevalent in current web search, but it is quite different to searching by specification of desired attribute values.

The other difference is that WIR deals with millions of pages, uncertain information, approximate results and other problems related to web environment. We deal with finding one (or a small set of) particular object, with very well defined attributes, with no uncertainty (of objects properties) and with a demand for exact answer. Beside the differences, we can inspire ourselves in WIR area. E.g. in [2] is proposed a nice user interface to help the user understand the quality of results. This is related to the visualization of results proposed in Section 5.

An example of preference based searching is Proactive [3]. Although the system does not provide very sophisticated user interface for user preference specification, the whole system for job search is based on the preferences specified by the user. An interface for preference specification is proposed in [5], but it still lacks the user-friendliness one would expect in a user-oriented system.

We base the work on fuzzy preference searching, particularly on the model of fuzzy logic programming (FLP) from [6], which is equivalent to general annotated programs (GAP) [7]. The preference of an object is determined by the preferences of the attribute values (denoted as \( f_i(a_i) \)), which are aggregated with an aggregation function \( @ \). An example of a general annotated program is \( A:@(b_1, ..., b_N) \Rightarrow B_1:b_1, ..., B_N:b_N \), where \( A \) is the preference of the object and \( b_i = f_i(a_i) \).

3. Searching with preferences

PrefShop is a system for searching with preferences, which is independent on the underlying data. It can be adapted to any area of interest, with a well defined attributes and their values.

The motivation for preference search is to allow flexible answers to a user query. When a user restricts the price to less than $1000, there can be an object with price $1001 that would satisfy his needs perfectly. In traditional search, the user would never see this object.
4. Implementation

An e-shop represents an exemplary web system where searchable data are formed by objects with exact predefined internal structure, thus leaving no uncertainty in the process of evaluating specified search criteria. This allows searching that can be precisely aimed at very specific areas of user interest.

Each searchable object in our application represents a product in the e-shop's offering. An object is defined by its attributes, which can be administrator-specified.

An example of a typical product would be "Notebooks", while its products' attributes would be "Processor", "Screen size" and "Disk size. As for attribute values, four different data types are being supported, each having its own method of preference calculation, based on the input search criteria.

4.1. Numerical attributes

Numerical attribute can be either integer or real, strictly positive or also of a negative value. A typical example of a numerical attribute is price. User preference is expressed as an interval value (with one border potentially set to infinity). When a user specifies that he wants the price between $100 and $200, all prices matching this interval get the

preference 1. Let us note this interval as $C = [c_{min},c_{max}]$. The price $99$ gets preference $0$ in the traditional search, but in our proposition, it gets preference e.g. $0.9$ We propose function $f(a): D_i \rightarrow [0,1]$ for normalization of attribute $A$:

$$f(a) = \exp\left( -\frac{(a-c)^2}{(\delta)^2} \right),$$

where $a$ is the value to normalize and $c$ is either $c_{min}$ or $c_{max}$, whichever is closer to $a$. $\delta$ is a parameter to tune the behavior of the function. In Figure 1 is an example of such function with various $\delta$.

This way a number slightly out of the required interval can also gain a high preference value, while even for very undesirable values their relative distance to the required interval can be determined. Each numerical attribute within a specific product category includes a definition of the $\delta$ value, which is used to adjust the shape of the preference curve to reflect the expected precision requirements for the provided search criteria. As, for example, "Year" attribute might require a different precision in one product category than it might in the other, $\delta$ values are defined per product category and are not assigned to the individual attributes themselves.

Finally, each numerical attribute also contains a definition of a format string, which is used to present the value together with a corresponding unit or other text information. An example of a numerical attribute would be "RAM memory size", which supports only positive integers, and is being displayed in the format "1024 MB".

4.2. Boolean attributes

A boolean attribute can be either true or false, search criteria being expressed strictly by one of these values. Resulting preference ends up with value 0 or 1, depending on whether the actual value has met the specified criteria. An example of a boolean attribute would be "Ethernet port", which may contain only value Yes or No.

4.3. String attributes

String attributes may contain arbitrary text values. Distinct values for a specific attribute are collected and presented by the user interface, which allows for exact
specification of the search term without the need to manually compose the search expression. This is typically the desired behavior, since string attributes are usually used for low cardinality domains, such as "Brand" or "Graphic card" attributes.

The process of evaluating the resulting preference is then equivalent to matching of the specified search term to the actual value, resulting in 1 in case the specified value was found and 0 otherwise.

4.4. String-list attributes

String lists are an extension to the simple string attributes, where multiple text values can be assigned to a single attribute value. Searching is performed with multiple expressions, and the resulting preference is based on the ratio of found expressions to the number of all search terms specified by the search criteria.

String lists are very useful in practice; an example of such attribute would be "Video outputs", with a single possible value of "HDMI, VGA, Composite".

5. Searching

Since the popularity of a web shop is highly dependent on the overall user experience, our implementation pays strong focus to the user interface aspect of the application design. The basic idea is to preserve any control schemes users might already know (based on a typical experience with existing applications), while utilization of several new interface elements designed specifically for the extended functionality was required.

Any user working with our implementation should be immediately able to pick product categories, browse product pages, switch view modes with different levels of detail, or change ordering of the presented objects. User registration, shopping and shopping cart modification also obey the traditional use case principles.

As for the search functionality, user is expected to specify the attributes of interest first. For these attributes, search criteria must be specified. Each type of attribute has its own controls, using which numeric intervals, boolean or text values can be set. Examples for boolean, resp. string attributes are in Figure 2 in the middle, resp. on the right.

We use a special slider element to specify numeric interval criteria (in Figure 2 on the left), where the desired border values can be set visually by dragging the appropriate slider handles. This method is more convenient, since there's no need to enter the desired values manually (while this possibility has also been preserved).

For each of the selected attributes, priority can be set, as represented in Figure 3. Correct priority values greatly contribute to the accuracy of the result, since the decisive preference value is calculated as an aggregated weighted sum of all the partial attribute preferences, as described in Section 3.

Figure 2. Numerical, Boolean and String attributes search criteria specification interface

On top of priorities, the possibility of cropping the object set is also provided. Any objects, that do not satisfy the specified condition completely, are excluded from the result set. This functionality is similar to the traditional search methods, and was found useful in case the user requires eliminating any undesired objects from search completely, such as products of an unpopular manufacturer.

Figure 3. Search criteria summarization and priority selection interface

The resulting preference is visualized as a background-colored percentage value next to each product's name, and serves as a default measure by which the objects in the result set are being organized.

Ordering of products can however be changed anytime, while the preference percentage information remains visible throughout the whole existence of the actual search. This way the user can keep an overview about how individual objects satisfy the search criteria, even in case a custom ordering is being used.

In the "Detailed" view however, not only the overall resulting preference is being displayed. Partial preferences by which individual attribute values satisfy the specified search conditions are being visualized as well. For this purpose, colored circular bullets placed in front of the actual values are used - with red, orange and green colors indicating low, medium and high preference values respectively, and circle sizes determining priority. An example is in Figure 4.

Furthermore, the display order of attributes is also dependent on the provided priority values. This behavior increases a chance of being able to display all the important attributes in the limited area of a single product box.
As a result, the described functionality of the "Detailed" view permits the user to keep track of all important attribute values and understand how they contribute to the resulting object preference value.

Finally, as bonus functionality, we allow all registered users to save complex search criteria under a specific name, which is a convenient way of storing search conditions without the need of reentering them at a later time.

6. Collaborative filtering

Apart from the possibility of searching objects by directly specified search criteria, the application also implements a collaborative approach (proposed in [5], see also [4]) to recommending potentially desirable products to the user. This functionality is available whenever the actual user profile is known, and is activated automatically in the browse mode, when no actual search is being performed.

The idea behind the collaboration lies in collecting of each user’s taste information, which acts as a key factor for mutual resemblance evaluation. For the active user, a set of similar users is determined, and their object’s popularity is used to contribute to the resulting preference value.

User taste information is formed either by explicit object rating (Figure 5 on the left), for which we use a custom star based interface component, or by more implicit methods, such as purchasing of an item.

Resulting preference value is visualized as a percentage next to each product’s name (this time without the appropriate background color), and a user is provided with a possibility to display the individual preference components by hovering the mouse cursor over the actual preference value (Figure 5 on the right).

7. Conclusion

We have proposed an e-commerce web site that uses more advanced preference handling than the current representants such as Amazon.com. The web shop is designed to be easily configured to other domains with different attributes. In future, we plan to identify more ways of using preferences in e-commerce environment and study the approaches to design of particular components for preference specification. We would like also to implement content-based recommendation which would combine with collaborative filtering.

Acknowledgement. This research was supported by projects SVV-2010-261312, GACR 201/09/H057, GACR P202/10/0761.

The web shop is online at http://195.113.17.17:8080/eshop

8. References


