A Computational Geometry Approach to Web Personalization

Maria Rigou[1,3]
[1]*Research Academic Computer Technology Institute, 61 Riga Feraiou Str. 26110 Patras, Greece*

Spiros Sirmakessis[1,2]
[2]*Technological Educational Institution of Messolongi Nea Ktiria, Messolongi, Greece*
{rigou, syrma, tsak}@cti.gr

Athanasios Tsakalidis[1, 3]
[3]*Department of Computer Engineering and Informatics University of Patras, 26500 Patras, Greece*

Abstract

In this paper we present an algorithm for efficient personalized clustering. The algorithm combines the orthogonal range search with the k-windows algorithm. It offers a real-time solution for the delivery of personalized services in online shopping environments, since it allows on-line consumers to model their preferences along multiple dimensions, search for product information, and then use the clustered list of products and services retrieved for making their purchase decisions.

1. Introduction

Nowadays, Internet has become one of the largest data repositories and the increasing number of people that use it as information source face the problem of information overload. In this difficult to manage volume of data, web users are trying to identify explicit information that satisfies their needs and suits their preferences. When the results matching a user's query, are of a size that does not allow for easy manipulation and hinder –instead of favoring- decision-making, further processing must be applied, so that results are presented in a way that can help users evaluate all available alternatives and perform relative comparisons before proceeding with the online purchase. Applications that operate in such an assistive manner are already available for online shopping (usually mentioned as shopping aids) and appear to have strong favorable effects on both the quality and the efficiency of purchase decisions [10].

Clustering is a data mining technique [16] that has been extensively studied and used in numerous cases where big volumes of information must be handled in a way that allows for knowledge extraction. Cluster analysis aims to discover objects that have representative behavior in the collection. The basic idea is that if a rule is valid for one object, it is very possible that the rule also applies to all the objects that are very similar to it. Algorithms for clustering data have been widely studied in various fields including Machine Learning, Neural Networks, Databases and Statistics. Based on the characteristics or attributes used [1], the available clustering algorithms can be categorized into *text-based* [9], [11], *link-based* [4], [5], and hybrid like [15], [12].

The utility of clustering in the web personalization domain and more especially in e-Commerce lies in the so-called *cluster hypothesis*; given a 'suitable' clustering of a product collection, if the user is interested in product *p*, he is also likely to be interested in other members of the cluster to which *p* belongs. As with products, we can set up a bipartite relation for people liking or being interested in products, and use this to cluster both people and products, with the premise that similar people like similar products, and vice versa. This important ramification of clustering is called *collaborative filtering* [8] and is the basis for the majority of recommender systems we meet in today's e-stores.

Personalization on the other hand, is an approach that has already spread widely in the web and is used -in some form- by all well-established web shopping environments. Tracing back its roots one ends up at the introduction of *adaptive hypermedia applications* in Brusilovsky's work of 1996 [6] and its updated version of 2001[7]. Adaptive hypermedia was introduced as an alternative to the traditional "one-size-fits-all" approach with the purpose of addressing the specific needs of individual users. Personalization on the web today covers a broad area, ranging from check-box customisation to recommender systems, and adaptive web sites. In this paper we are focusing on personalization for electronic commerce. With the enormous and ongoing growth of products and services available from different sources for online transactions, many online customers face the problem of putting together the appropriate list of products based on their needs. This is where personalization may come into the picture and transparently 'deliver' to users tailored products and services.

In our context, personalization takes up the form of providing the online purchaser the ability to specify the range of individual preferences (in terms of product features such as price, functionalities, appearance, size, etc.). Using this range as input, e-stores can process online user requests and return products that reside inside the preference spectrum of the individual customer. The idea behind this is to accomplish effective (in terms of

Proceedings of the IEEE International Conference on E-Commerce Technology
0-7695-2098-7/04 $20.00 © 2004 **IEEE**

time and space) *personalized clustering* of online products. The next section describes the idea of personalized clustering in more detail. In section 3 we present the proposed algorithm and the final section discusses potential applications and implications of the idea, as well as conclusions.

2. Personalized clustering

Typically, when we want to deliver personalized results, we deploy the personalization process as the final phase in order to filter the initial set of data and compose a subset that satisfies individual constraints. In the case of online clustering though, and for applications that are time sensitive and should execute in real time, response times must be kept to a minimum and thus it is much wiser to apply the personalization filtering before entering the clustering phase in order to reduce the data set to be clustered (since execution times increase dramatically with the size of the data set to be clustered). The *k-means range* algorithm [14] performs personalized clustering and bases the clustering step on the well documented and widely used *k-means* algorithm, while for the personalization step preceding the clustering a range tree is constructed and parsed in order to limit the initial set of products to those that are of interest for the current user.

In the following, we propose a new algorithm, which we argue that outperforms the latter since it takes advantage of the range tree already constructed during the personalization step and then uses this tree structure in the clustering step that follows by applying an algorithm that improves the k-means; the *k-windows* algorithm [18]. K-windows reduces the number of patterns that need to be examined for similarity using a windowing technique that by exploiting the range tree structure achieves lower time complexity compared to other well-known clustering algorithms and high quality clustering results.

3. The proposed algorithm

The *range tree* [19] was introduced to solve the range-searching problem; *preprocess a set S of points in R_d so that the points in S lying inside a query range can be reported or counted quickly*. A *d*-dimensional range tree is defined recursively from the corresponding tree for the (*d*-1) dimensional case. A 1-dimensional tree can be considered as a leaf-oriented balanced binary search tree. The range tree uses $O(n\log^{d-1}n)$ space and answers the range-searching problem in $O(\log^{d}n+m)$ time, where n is the total number of points, and m is the number of points reported. The query time is further reduced to $O(\log^{d-1}n+m)$ using a portion on the fractional cascading technique [17].

The *k*-windows algorithm [18] clusters an initial set of n patterns in k groups (or clusters) according to the following methodology. At first, k means are selected (possibly in a random manner) and initial d-ranges (windows) have as centers these initial means. Then, the patterns that lie within each d-range are identified using the Orthogonal Range Search technique which is based on a preprocess phase where a range tree is constructed. Patterns that lie within a d-range can be found traversing the range tree, in polylogarithmic time. Next, the mean of patterns that lie within each range, is calculated with each mean defining a new d-range that is considered a movement of the previous d-range. The last two steps are executed repeatedly, until there is no *d*-range that includes a significant increment of patterns after a movement. After finalizing the initial partition the *d*-ranges are enlarged in order to include as many patterns as possible from the cluster. This can be achieved by forcing *d*-ranges to preserve their mean during enlargement. Then, *k*-windows calculates the relative frequency of patterns assigned to a *d*-range in the whole set of patterns. If the relative frequency is small the whole process must be repeated.

The time complexity of *k*-means is $O(ndkt)$ while in the case of *k*-windows, time is reduced to:

$$O(dkqr(\frac{\log^{d-2} n}{d}+s))$$

, where n is the total number of patterns (representing products in our case), k is the number of clusters to be constructed, t is the number of iterations (during the execution of the *k*-means), q is the number of movements and r of iterations (for *k*-windows), and s is the number of patterns within a *d*-range. It is worth mentioning that $s << n$ and that qr is proportional to t.

The idea of basing personalization on the individual preference ranges (in terms of product parameters) can be easily transferred to the Euclidean space using vectors for representing available e-store products. More specifically, products are coded in the form of vectors with the value in each dimension representing the measurement of the respective parameter (a product parameter is in essence a descriptive feature such as size, price, etc.). The total number of parameters used to model products dictates the number of dimensions our space will be made of (i.e. products modeled using n parameters are represented as vectors in the *n-dimensional* space). An individual preference range is made up of parameter intervals in the form of starting and ending values. For clarity reasons we constrain our example in the 2-dimensional space and thus assume that products and preferences are expressed in terms of 2 parameters, X and Y (Figure 1).

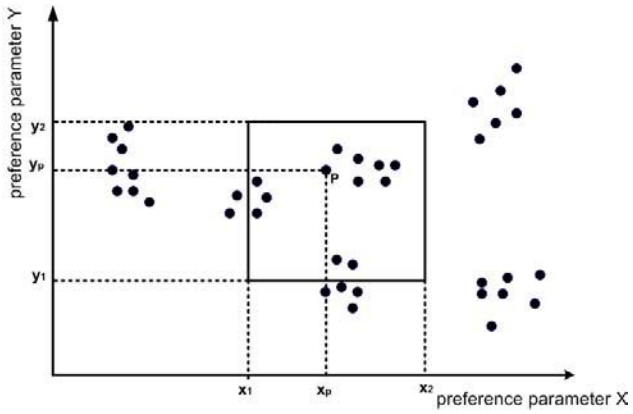

Figure 1. The preference range represented as a rectangle in the 2-dimentional space of preference parameters X and Y

A product P is represented by value x_p for parameter X and y_p for parameter Y and therefore is point (x_p,y_p) in our space. Assuming that the preference range of user U is $\{(x_1,x_2), (y_1,y_2)\}$ an orthogonal rectangle R_u is defined. In order to meet personal preferences the e-store must answer product requests received from user U only with products that satisfy the preferences stated by U, or in other words, products that lay within R_u. The question of whether P lies within R_u or not, is typically addressed in bibliography using the range tree approach. This ends the 1st step of the algorithm. For the points reported from the 1st step, the k-windows algorithm is applied. The k-windows algorithm is perform to a limited set of points m<<n reported from the first step increasing significantly the speed of the clustering process. The input and the output of the algorithm follow:

Input:
a) A set V = $\{(p_1, p_2, \ldots, p_n)|$ where $p_i \in \Re_d$, and $i, d \in I\}$ of n points.
b) A *d-range* query $Q = [\alpha_1,b_1] \times [\alpha_2,b_2] \times \ldots \times [\alpha_d,b_d]$ specified by points $(\alpha_1, \alpha_2, \ldots \alpha_d)$ and $(b_1, b_2, \ldots, b_n)$, with $a_j \le b_j$.
c) The number $k \in I$ of clusters to be outputted.
Output: All points m lying within the *d-range* Q, clustered in k groups.

The time complexity of the algorithm is calculated by adding the respective complexities of its two successive steps. The multidimensional range search can be solved in $O(\log^{d-1} n+m)$ time, where m is the size of the answer, as described in the previous section. The time complexity of k-means is $O(ndkt)$ while in the case of k-windows, time is reduced to:

$$O\left(dkqr\left(\frac{\log^{d-2} n}{d} + s \right) \right)$$

where n is the total number of patterns (representing products in our case), k is the number of clusters to be constructed, t is the number of iterations (during the execution of the k-means), q is the number of movements and r of iterations (for k-windows), and s is the number of patterns within a d-range. It is worth mentioning that $s<<n$ and that qr is proportional to t.

The overall complexity of the algorithm is:

$$O\left(\log^{d-1} n + dkqr\left(\frac{\log^{d-2} m}{d} + s \right) \right)$$

This is an improved time complexity compared to $O(\log^{d-1} n+m)$ as derived in [14]. Notice that the basic operation is the arithmetic comparison between two numbers without any distance computation. Therefore, the k-windows algorithm has a significantly superior performance than the direct k-means algorithm used in [14].

The complexity of the algorithm can be further reduced by using different structures for the range searching. For $d \ge 3$ dimensions a simple solution is given in [2], in $O(n\log^{d-1} n)$ preprocessing time and space and $O(\log^{d-2} n+m)$ query time. Using this approach the complexity is reduced to:

$$O\left(\log^{d-2} n + dkqr\left(\frac{\log^{d-2} m}{d} + s \right) \right)$$

5. Implications and conclusion

In this paper we presented an algorithm for personalized clustering that improves the k-means range algorithm by using instead of the k-means, the k-windows algorithm for the clustering step. The improvement is not only due to the fact that k-windows outperforms k-means but also because the range tree that is constructed in the first phase of the k-means range algorithm (for restricting the data set to the preference range) is also used during the second phase by the k-windows.

One practical implication may be caused by the superlinear space requirements for the construction and maintenance of the range tree, a problem that is enlarged when the number of dimensions increases (i.e. products are represented using more and more parameters). Alevizos et al. in [3] introduce an improved version of the k-windows algorithm that uses (instead of a range tree) a multi-dimensional binary tree for performing the range search. In the case of the proposed algorithm though, the potential of adopting this approach will trigger changes in the initial phase of filtering the data set before the clustering. The authors are planning to study these implications.

Summarizing, we must point out that when we cluster products and not users (or user preferences, which is the case for instance in collaborative filtering), we have the advantage of maintaining product related information in a structure that rarely ever changes. This is opposed to the case where user behavior is used for the clustering, and thus we have to restrict our solution to structures that in addition to fast retrieval, also allow for easy and fast updating. In practice, the advantage in the product clustering scenario is that the process of constructing and keeping updated the internal structure, which stores product data (i.e. the quite expensive step of constructing the range tree in our case) may execute off-line.

6. References

[1] Aldenderfer M., and Blashfield R., *Cluster Analysis*, in series: Quantitative Applications in the Social Sciences, Sage Publications Inc., 1984.

[2] P. Alevizos, "An algorithm for orthogonal range search in d≥3 dimensions", *Proceedings of 14th European Workshop on Computational Geometry*, Barcelona, 1998.

[3] P. Alevizos, B. Boutsinas, D.K. Tasoulis, and M.N. Vrahatis, "Improving the orthogonal range search k-windows algorithm", *Proceedings of the "14th IEEE International Conference on Tools with Artificial Intelligence, (ICTAI 2002)"*, Washington D.C., U.S.A., November 4-6, 2002, pp. 239-245.

[4] R.A. Botafogo, and B. Shneiderman, "Identifying aggregates in hypertext structures", *Proceedings of the 3 rd ACM Conference on Hypertext*, 1991, pp. 63-74.

[5] R.A. Botafogo, "Cluster analysis for hypertext systems", *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, 1993, pp. 116-125.

[6] P. Brusilovsky, "Methods and Techniques of Adaptive Hypermedia", *Journal of User Modelling and User-Adaptive Interaction* 6, n 2-3, 1996, pp. 87-129.

[7] P. Brusilovsky, "Adaptive hypermedia", *User Modeling and User-Adapted Interaction*, Ten Year Anniversary Issue 11 (Alfred Kobsa, ed.), 2001, pp. 87-110.

[8] Chakrabarti S., *Mining the Web: Discovering Knowledge from Hypertext Data*, L. Homet ed., Morgan Kaufmann Pub., USA, 2003.

[9] R.C. Dubes, and A.K. Jain, "Clustering methodologies in exploratory data analysis", *Adv. Comput*., 19, 1980, pp.113–228.

[10] G. Haübl, and V. Trifts, "Consumer Decision Making in Online Shopping Environments: The Effects of Interactive Shopping Aids", *Marketing Science* 19 (1), 2000, pp. 4-21.

[11] G. Karypis, E.H Han, and V. Kumar, "CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modelling", *IEEE Computer*, Vol. 32, No. 8, 1999, pp. 68-75.

[12] D. Modha, and W.S. Spangler, "Clustering hypertext with applications to web searching", *Proceedings of the ACM Conference on Hypertext and Hypermedia*, 2000, pp 143-152

[13] G. Papamichail, and D. Papamichail, "Towards using computational methods for real-time negotiations in electronic commerce", *European Journal of Operational Research*, Vol 145 (2), March 2003, pp. 3-9.

[14] G. Papamichail, "The K-means Range Algorithm for Personalized Data Clustering in E-Commerce", *14th Mini Euro Conference, Human Centered Processes, Distributed Decision Amking and Man-Machine Cooperation*, 5-7 May 2003, Luxemburg, pp 239-245.

[15] P. Pirolli, J. Pitkow, and R. Rao, "Silk from a sow's ear: Extracting usable structures from the Web", *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing*, 1996, pp 118–125.

[16] Rasmussen E., *Clustering Algorithms*, in Information Retrieval, W.B. Frakes & R. Baeza-Yates (eds.), Prentice Hall PTR, New Jersey, 1992.

[17] Y.K. Vaishnavi, "Computing Point Enclosures", *IEEE Transactions on Computing* C-31 (1) 1982, pp. 22-29.

[18] M.N. Vrahatis, B. Boutsinas, P. Alevizos, and G. Pavlides, "The k-windows Algorithm for Improving the k-means Clustering Algorithm", *Journal of Complexity*, vol. 18, 2002, pp. 375-391.

[19] D. Willard, "New data structures for orthogonal range queries", *SIAM Journal on Computing* (14) 1985, pp. 232-253.