Ranking the Product Items on the Web with Multi-dimensional Preferences

ZHENGLU YANG,[†1] LIN LI[†1] and MASARU KITSUREGAWA [†1]

Current search engines cannot effectively rank those relational data, which exists on dynamic websites supported by online databases. In this work, to rank such structured data, we introduce an integrated system consists of three parts such as web crawler and information extractor, index constructor, and query processor on (relaxed) dominant relationship analysis of the product items on the Web. Extensive experiments are conducted and the results illustrate the effectiveness and efficiency of our methods.

1. Introduction

Suppose you are buying a digital camera from a website (i.e., www.bestbuy.com) and you are looking for one that is cheap and with high sensor resolution to take beautiful pictures. Unfortunately, these two goals are complementary to one another as cameras with more megapixels tend to be more expensive. Moreover, you would appreciate the search if only a few "best" goods are recommended by the website's system with regard to your preference, instead of checking all the items manually. Intuitively, these "best" goods are all cameras that are not worse than any other camera in both attributes, i.e., price and sensor resolution. For instance, Fig. 1 shows some sample cameras and among them, only the items *c* (PowerShot A630) and *d* (EOS Digital Rebel XTi) should be the candidates recommended to the user.

This problem of searching for such "best" items, can be traced back to the 1960s in the theory field. The set of these "best" items is called the Pareto set and the objects are called maximal vectors[1)]. However, these main memory algorithms are inefficient for online query processing.

Recently, the *skyline* operator[2)] was proposed to tackle the maximal vector problem in database context. Efficient skyline querying methodologies have been studied extensively (i.e.,[3)]). All of these works, however, concerned only the pure binary relationship, i.e., a product item *p* is whether or not worse than (dominated by) others. In this paper, we propose to analyze a more general dominant relationship in a business model, that users preferred the details of the dominant relationship, i.e., an item *p* dominates (and vice versa, is dominated by) how many other items. Here we show an example.

Item ID	Product name	Company	Price ($)	Weight (g)	Sensor resolution (M)
a	PowerShot SD630	Canon	349.99	142	6
b	Cyber-shot DSC-H5/B	Sony	479.99	404	7.2
c	PowerShot A630	Canon	269.99	245	8
d	EOS Digital Rebel XTi	Canon	799.99	509	10.1
e	DSLR-A100K	Sony	899.99	545	10
f	Cyber-shot DSC-M2	Sony	649.99	180	5.1

Fig. 1 Digital camera example

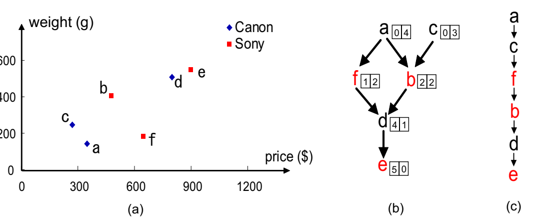


Fig. 2 Product attributes in 2-dimensional space and the corresponding partial order

Example 1: Consider you are a manager of Sony corporation. You want to know the business position of a digital camera *f* (Cyber-shot DSC-M2) in the current market with regard to your preference, i.e., price and weight, by checking how many other items are better/worse than *f* and what they are. For the sample cameras shown in Fig. 1, you can deduct the conclusion that item *f* is better than items *d* and *e* but worse than item *a* with regard to your preference[⋆1].

We found the interrelated connection between the dominant relationship and the partial order. To illustrate the idea, here we show a simple example. Fig. 2 (a) represents the product items in two attributes (dimensions)

†1 Institute of Industrial Science, The University of Tokyo
{yangzl, lilin, kitsure}@tkl.iis.u-tokyo.ac.jp

⋆1 Note that the analysis here can be further used to determine the price of a new product, which should be competitive in the current market while reserving the most profit.

space, i.e., price and weight. Fig. 2 (b) is the corresponding partial order (encoded as DAG format). We can know that item f dominates items d and e and is dominated by item a, by checking the *out-link* and *in-link* of f, respectively. In this paper we aim to explore how to efficiently find and query such succinct representative partial orders.

Moreover, although the general dominant relationship analysis plays an important role in business decisions, it may fails due to the market adjustment mechanism, that product items are always complementary to one another, i.e., lightweight cameras tend to be more expensive. Therefore, it may be that no products are worse than (dominated by) a given query item, especially when querying in high dimension (attribute) space. As such, users would like to get the items with highest comprehensive scores by fusing the values on different attributes. For example, Fig. 2 (c) shows one fused rank list of the sample dataset in Fig. 2 (a)⋆1. For the same example 1, by checking Fig. 2 (c), you can know that f dominates b, d and e by considering on the comprehensive value of items. Note that the result here is different from that discovered by general dominant relationship analysis (i.e., f only dominates d and e). In this paper, we relax the strict meaning of "dominate" in general dominant relationship analysis by considering on the comprehensive value of items, which incorporates rank aggregation methods.

Different users may have different preferences. To cope with this problem, we propose a compressed data cube to encode all the possibilities in a concise format and devise efficient strategies to extract the information.

2. Integrated system building, maintaining and querying

In this section, we introduce our online query system, which is illustrated in Figure 3. There are three core parts involved with this system: 1) Web crawler and information extractor; 2) Data index constructor; and 3) Query processor.

2.1 Web Crawler and Information Extraction

We implement a module that first crawls the product pages and then convert the unstructured text into structured tuple data. The idea follows that in[5]. While it is time consuming and prohibitive to crawl every page in real time in a resource limited environment, we, fortunately, only need to monitor large online shopping websites (i.e., www.bestbuy.com). The reasons are twofold: first, these websites make a good job on integrating and summarizing all the products; second and most important, many customers purchase the goods through these websites because they always provide desirable services (i.e., discount price).

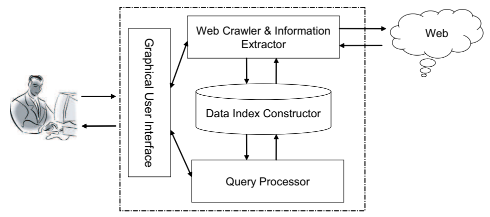


Fig. 3 The proposed system based on the partial order data cube (ParCube) (this figure is best viewed in color)

Upon the product html pages are crawled, we parse them based on Document Object Model (DOM) by employing predefined templates. The values of desirable items are extracted and stored as tuples. The information extraction process is executed repeatedly (i.e., every one hour).

2.2 Data Index Construction

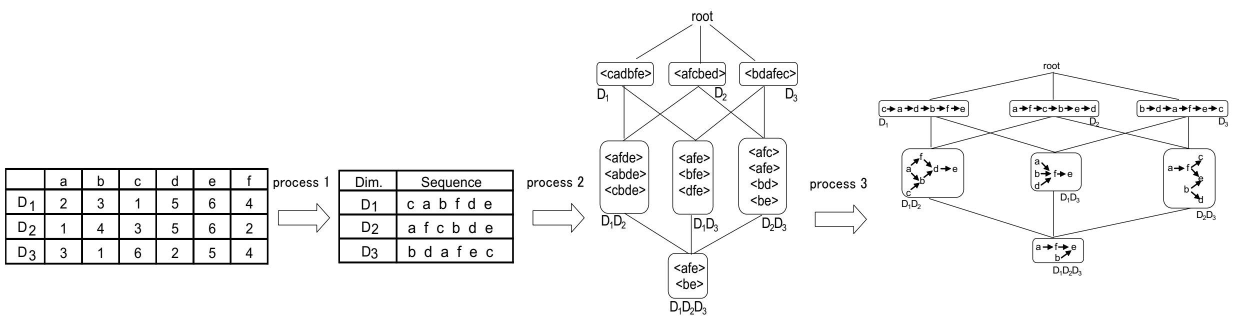
To illustrate the processes of building the data index, an example data set is shown in Fig. 4 (a) ⋆2. We propose to apply strategies from another research context, sequential pattern mining[6], to get the partial order representation cube (*ParCube*) from a spatial dataset. There are three processes for *ParCube* construction.

The first process is to convert the spatial dataset to the sequence dataset. With a k-dimensional dataset, we simply get a k-customer sequence dataset, by sorting the objects in each customer (dimension) according to their value in ascending order. For example, Fig. 4 (b) shows the converted sequence dataset of the spatial dataset in Fig. 4 (a).

The second and the third processes aim to determine a partial order that describes the point set in the subspace S' of data space S in D'. In the second process, we discover the sequential patterns from the transformed se-

⋆1 *Borda Count* rank aggregation method[4] is employed here.

⋆2 For simplicity, we use small integer to simulate items' values on the attributes for convenience of description in this paper.



	a	b	c	d	e	f
D1	2	3	1	5	6	4
D2	1	4	3	5	6	2
D3	3	1	6	2	5	4

Dim.	Sequence
D1	c a b f d e
D2	a f c b d e
D3	b d a f e c

(a) Example spatial dataset (b) Transformed sequence dataset (c) The data cube (lattice) whose cuboid consists of the local maximal common sequential patterns (d) DAG representation of partial order in data cube ParCube

Fig. 4 The result representation of each process for the example spatial dataset

quence dataset by applying some state-of-the-art algorithm , PrefixSpan[7)][⋆1]. To save space, we merge these sequential patterns as *local maximal sequential sequences*, which are not the subsequence of other sequential sequences. For example, in subspace {D$_1$, D$_2$}, although $\langle afd\rangle$, $\langle afe\rangle$, $\langle ade\rangle$, and $\langle fde\rangle$ are length-3 patterns, we merge them as length-4 *local maximal sequential sequences*, as $\langle afde\rangle$. The resultant data cube (*SeqCube*) from the second process is shown in Fig. 4 (c).

In the third process, the combinations of the *local maximal sequential sequences* are enumerated to generate partial orders with DAG representation, by applying the method proposed in[8)]. The resultant data cube (*ParCube*) for the example dataset is shown in Fig. 4 (d).

2.3 Query Processing

- General Dominant Relationship Query

All the general dominant relationship related to the query point, P_{query}, can be easily discovered by traversing the DAG in a specific subspace. The example DAG shown in Fig. 2 (b) illustrates this scenario. To facilitate the counting, the numbers of points dominating/dominated by the current node are inserted into each node.

- General Relaxed-Dominant Relationship Query

The intuition idea is that the general dominant relationship has a stronger rank meaning compared with relaxed dominant relationship. Therefore the semantic meaning compressed in the partial order representation, DAG, can be utilized to deduce relaxed dominant relationship with the help of rank aggregation methods.

We have implemented the algorithm of rank aggregation in the query processing based on weighted Borda Fuse[9)]. To present the rank normalization and fusion methods, some basic notations are introduced[⋆2]. Given a d-dimension (attribute) space S=$\{s_1, s_2, \ldots, s_d\}$, a set of product items D=$\{p_1, p_2, \ldots, p_n\}$ is said to be a dataset on S if every $p_i \in D$ is a d-dimensional item on S. The formula what we use to compute the integrated score of the items are shown as follows.

$$\omega^{\tau}(p_i) = \begin{cases} 1 - \frac{\tau(p_i)-1}{|D|} & p_i \in \tau \\ \frac{1}{2} + \frac{|\tau|-1}{2\cdot|D|} & \text{otherwise} \end{cases} \qquad (1)$$

Weighted Borda Fuse ([9)]):

$$s^{\hat{\tau}(p_i)} = \sum_{\tau\in S} \alpha_\tau \cdot \omega^{\tau}(p_i) \qquad (2)$$

where $\sum_{\tau\in S} \alpha_\tau = 1$ and $\alpha_\tau \geq 0$. In this paper, we assume every dimension has the same value of weight.

3. Performance Analysis

We performed the experiments using an Intel Core 2 Duo 2.33GHz PC with 2G memory. The data index construction and query processing were written in C++, and the other parts (i.e., crawler) were written in Java. We conducted experiments on both real[⋆3] and synthetic datasets. The synthetic datasets have *independent* distribution, with dimensionality d in the range [3, 7] and data size in the range [10k, 50k]. The default values of dimensionality were 5. The default value of cardinality for each dimension was 50k. For the comparison on query performance, we compare *RDRA*, which was implemented as described in this paper, and naive Borda Count method[9)].

⋆1 Any other sequential pattern mining algorithm can be used as well.

⋆2 Due to space limited, refer[10)] for more detail.

⋆3 Products on the website of www.kakaku.com.

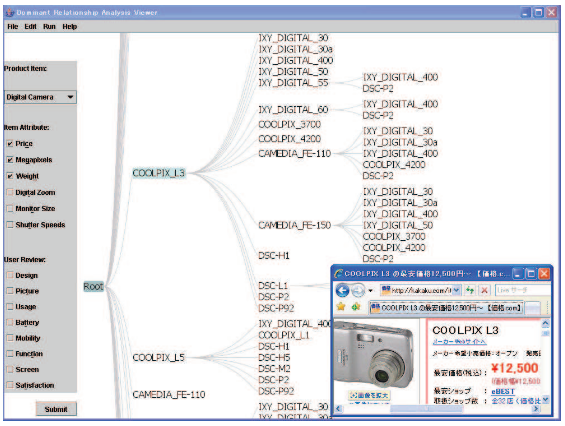
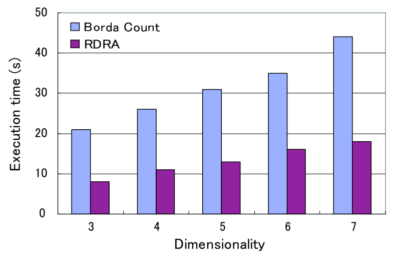
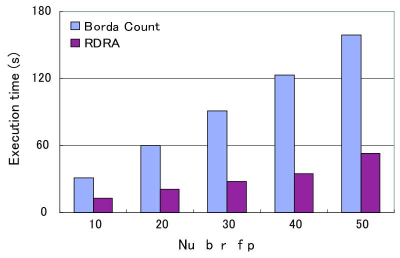


Fig. 5 Visualization of our system



(a) Execution time comparison when varying dimensionality (number of points=10K)



(b) Execution time comparison when varying number of points in the dataset (dimensionality=5)

Fig. 6 Execution time comparison of querying between *RDRA* and *Borda Count* against dimensionality and number of points in the dataset (number of query points=500)

3.1 Effectiveness of Our System

To convenient users to find the dominant relationship between items with regard to their preferences, we built a system with a graphical user interface based on *Prefuse* toolkit[⋆1]. Fig. 5 illustrates the snapshot for one user query (Product = "Digital Camera" ∧ skyline attributes: Price ∨ Sensor Resolution ∨ Weight). The dominant relationship tree shown in the middle of the page clearly illustrates the business status of each product item. The node on the left part of the tree dominates (is better than) the node on the right. Furthermore, it is easy to check each item's information by clicking on its represented node and meanwhile, browse its dominating nodes.

3.2 Query Performance

The comparison of the execution time on querying relaxed dominant relationship between *RDRA* and traditional rank aggregation (*Borda Count*) is shown in Fig. 6. The reason why we compared with *Borda Count* is that we want to demonstrate the efficiency of partial orders on querying relaxed dominant relationship, rather than comparing two rank aggregation methods themselves. The latter issue is beyond the scope of this paper. We can know that *RDRA* is much efficient than its competitor for the two cases (varying dimensionality and number of points) because of the effect of partial orders we used.

We have also conducted other experiments (i.e., data index construction), refer[10] for more detail.

4. Conclusions

We have proposed effective methods to construct a data cube, *ParCube*, which concisely represents the dominant relationship as partial orders. We introduced efficient query processing strategies to address the (relaxed) dominant relationship problem. The performance study confirmed the efficiency of our strategies.

⋆1 http://www.prefuse.org

References

1) J. L. Bentley, H.T. Kung, M. Schkolnick, and C.D. Thompson, "On the Average Number of Maxima in a Set of Vectors and Applications," JACM, 1978.
2) S. Borzsonyi, D. Kossmann, and K. Stocker, "The skyline operator," ICDE, 2001.
3) D. Kossmann, F.Ramsak, and S. Rost, "Shooting stars in the sky: An online algorithm for skyline queries," VLDB, 2002.
4) J. C. Borda, "Mémoire sur les élections au scrutin," Histoire del'Académie Royal des Sciences, 1781.
5) N. Kushmerick, D. S. Weld, and R. Doorenbos, "Wrapper induction for information extraction," IJCAI, 1997.
6) R. Agrawal and R. Srikant, "Mining sequential patterns," ICDE, 1995.
7) J. Pei, J. Han, B. Mortazavi-Asl, and H. Pinto, "PrefixSpan:Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth," ICDE, 2001.
8) G. Casas-Garriga, "Summarizing sequential data with closed partial orders," SDM, 2005.
9) J. A. Aslam and M. Montague, "Models for metasearch," SIGIR, 2001.
10) Z. Yang, L. Li, and M. Kitsuregawa, "Efficient Querying Relaxed Dominant Relationship between Product Items based on Rank Aggregation," AAAI, 2008.