Towards Efficient Dominant Relationship Exploration of the Product Items on the Web

Zhenglu Yang, Lin Li, Botao Wang, and Masaru Kitsuregawa
Dept. of Info. and Comm. Engineering, University of Tokyo
4-6-1 Komaba, Meguro-ku, Tokyo, Japan
{yangzl, lilin, botaow, kitsure}@tkl.iis.u-tokyo.ac.jp

ABSTRACT

In recent years, there has been a prevalence of search engines being employed to find useful information in the Web as they efficiently explore hyperlinks between web pages which define a natural graph structure that yields a good ranking. Unfortunately, current search engines cannot effectively rank those relational data, which exists on dynamic websites supported by online databases. In this study, to rank such structured data (i.e., find the "best" items), we propose an integrated online system consisting of compressed data structure to encode the dominant relationship of the relational data. Efficient querying strategies and updating scheme are devised to facilitate the ranking process. Extensive experiments illustrate the effectiveness and efficiency of our methods. As such, we believe the work in this poster can be complementary to traditional search engines.

Categories and Subject Descriptors: H.3.3[Information search and retrieval]

General Terms: Algorithms, Experimentation.

Keywords: Query optimization, information extraction.

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. 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 exploration of the problem to search for such "best" items has a long history and 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 [2]. However, these main-memory algorithms are inefficient for online query processing.

Recently, the $skyline$ operator [3] was proposed to tackle the maximal vector problem in database context. However, the existing work in skyline mining context concerned only the pure binary relationship among a dataset, i.e., a product item p is weither or not worse than (dominated by) others. Interestingly, Li et al. [5] proposed to analyze a more general dominant relationship in a business model, that users preferred the details of the dominant relationship more, i.e., an item p dominates (and vice versa, is dominated by) how many other items. Unfortunately, the existing methodologies [5] cannot tackle the dominant relationship analysis in a dynamic environment, i.e., the Web.

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

Figure 1: Digital camera example

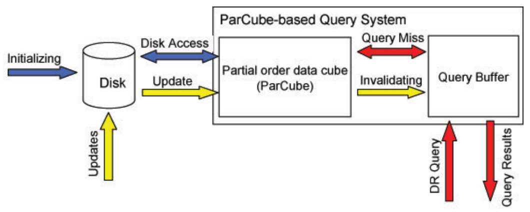


Figure 2: The proposed system based on the partial order data cube (ParCube) (this figure is best viewed in color)

We find the interrelated connection between the dominant relationship and the partial order. We propose effective methods to construct a data cube, $ParCube$, which concisely represents the complete information of the general dominant relationship as partial order representation (DAGs). Based on $ParCube$, we efficiently construct an online query system, which employs an object-aware update scheme to maintain $ParCube$ in a dynamic environment. We conduct comprehensive experiments to illustrate the effectiveness and efficiency of our strategies.

2. INTEGRATED SYSTEM BUILDING, MAINTAINING AND QUERYING

The framework of our system is illustrated in Fig. 2. There are three major parts involved with this system: 1) system construction; 2) query processing; and 3) system updating.

2.1 System Construction

We formally justify the statement that the dominant relationship can be encoded in partial order representation (DAGs), while constructing the partial order data cube ($ParCube$), which is the core data structure of our system. We explain how to efficiently construct $ParCube$ with a spatial dataset input[1]. To our best knowledge, there has been no work has tackled this problem. In this poster, we propose to apply strategies from another research context, sequential pattern mining [1], to get the partial order representation from a spatial dataset. There are three processes employed in

[1]The product items dataset can be seen as a spatial dataset (i.e., item v.s. point and attribute v.s. dimension).

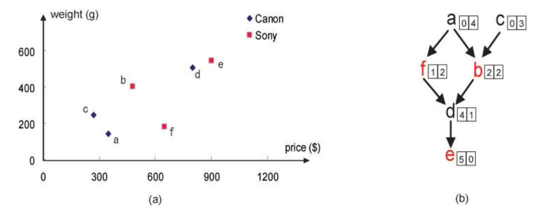
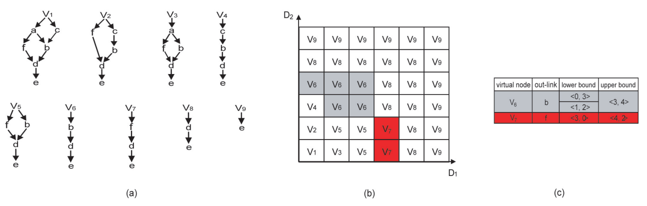


Figure 3: Product attributes in 2-dimensional space and the corresponding partial order



virtual node	out-link	lower bound	upper bound
V_6	b	<0, 3> <1, 2>	<3, 4>
V_7	f	<3, 0>	<4, 2>

Figure 4: All virtual nodes representation in 2-dimensional space $\{\mathbf{D}_1, \mathbf{D}_2\}$ (this figure is best viewed in color)

the system construction. We propose a simple method of converting the spatial dataset to the corresponding sequence dataset in the first process and then, apply existing strategies [4] with little modification in the second and third processes to generate DAGs from the transformed sequence dataset.

2.2 Query Processing

We differentiate two cases based on whether the point P_{query} is in the dataset D. Case 1 ($P_{query} \in D$): All the general dominant relationship related to P_{query} can be easily discovered by traversing the DAG in a specific subspace. Case 2 ($P_{query} \notin D$): We propose to solve this problem by traversing the partial order representation (DAGs) with semantic cutting. To illustrate this, we parse the DAG in Fig. 3 (b) by adding some $Virtual\ Node$s which act as real nodes that dominate different sets of points in the original dataset. Fig. 4 (a) shows all the virtual nodes (denoted as V$_i$). We can see for the nodes which dominate three points, there are two virtual nodes (i.e., V$_6$ and V$_7$). Fig. 4 (b) shows the virtual nodes effect in grid model. It is deemed as a semantic representation of that proposed in [5]. We propose an efficient algorithm to find such kind of virtual nodes and store them into compressed format.

2.3 System Maintenance

We propose our efficient update scheme to maintain the system. We differentiate three cases when update happens: node deleting, inserting and modifying. Case 1 (Deleting a node θ): To tackle this case is straightforward, that we only need to link the parents of θ and the children of θ, respectively. Case 2 (Inserting a node θ): We need to further explore the topology of the DAG to locate the exact position of θ, because of the multiple possibilities that θ can be embedded. We use the same strategies as those used in the last Section to find the corresponding internal virtual nodes of θ, by range search. Case 3 (Modifying a node θ): It is tackled as a sequel execution of node deleting and node inserting.

3. PERFORMANCE ANALYSIS

We conducted experiments on both real and synthetic datasets by comparing two algorithms, $Naive$[2] and $ParOrder\_Web$[3].

[2]*Naive* was tested with the extension of DADA [5], by storing the dominated points in the corresponding class.

[3]*ParOrder_Web* was implemented as described in this poster.

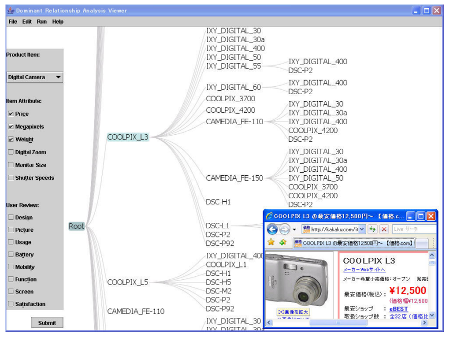


Figure 5: Visualization of our system

3.1 Effectiveness of Our System

To convenient users to find the “best” items with regard to their preferences, we built a system with a graphical user interface, as shown in Fig. 5. The interface of our system is interactive and expandable. For example, it is easy to check each item’s information by clicking on its represented node and meanwhile, browser its dominating nodes.

3.2 Query Performance and System Maintenance

From the experimental results of query performance, we know the $ParOrder\_Web$ algorithm outperforms the $Naive$ in both cases, $P_{query} \in D$ and $P_{query} \notin D$. This is because the compact representation of partial orders leads to faster routing. We also know that the performance difference between the $Naive$ method and the $ParOrder\_Web$ in the first case is much larger than that in the seconde case. The reason is that we can directly traverse the DAGs when $P_{query} \in D$, instead of judging the virtual nodes when $P_{query} \notin D$.

From the experimental results of system maintenance, we know that the $ParOrder\_Web$ algorithm outperforms the $Naive$ in both cases, $P_{query} \in D$ and $P_{query} \notin D$. The reason is similar to that for query performance improvement. Due to the curse of dimensionality, both of these two methods become inefficient when the dimensionality increases.

4. CONCLUSIONS

We have introduced an integrated system for analyzing Dominant Relationship. We found the connection between the partial order and the dominant relationship, and proposed efficient strategies to answer the dominant relationship queries. The performance study confirmed the efficiency of our strategies.

5. REFERENCES

[1] R. Agrawal and R. Srikant. Mining sequential patterns. In *ICDE*, pp. 3-14, Taipei, Taiwan, 1995.

[2] 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. *Journal of ACM*, 25(4), pp. 536-543, 1978.

[3] S. Borzsonyi, D. Kossmann and K. Stocker. The skyline operator. In *ICDE*, pp. 421-430, Heidelberg, Germany, 2001.

[4] G. Casas-Garriga. Summarizing sequential data with closed partial orders. In *SDM*, pp. 380-391, Newport Beach, CA, USA, 2005.

[5] C. Li, B. C. Ooi, A. K. H. Tung, and S. Wang. DADA: A Data Cube for Dominant Relationship Analysis. In *SIGMOD*, pp. 659-670, Chicago, IL, USA, 2006.