Data & Knowledge Engineering 86 (2013) 1–18

Contents lists available at SciVerse ScienceDirect

Data & Knowledge Engineering

journal homepage: www.elsevier.com/locate/datak

Diamond dicing

Hazel Webb [a,*], Daniel Lemire [b], Owen Kaser [a]

[a] University of New Brunswick Saint John, Canada
[b] TELUQ, Université du Québec, Canada

ARTICLE INFO

Article history:
Received 16 June 2010
Received in revised form 31 December 2012
Accepted 7 January 2013
Available online 18 January 2013

Keywords:
OLAP
Information retrieval
Multidimensional queries

ABSTRACT

In OLAP, analysts often select an interesting sample of the data. For example, an analyst might focus on products bringing revenues of at least $100,000, or on shops having sales greater than $400,000. However, current systems do not allow the application of both of these thresholds simultaneously, selecting products and shops satisfying both thresholds. For such purposes, we introduce the diamond cube operator, filling a gap among existing data warehouse operations. Because of the interaction between dimensions the computation of diamond cubes is challenging. We compare and test various algorithms on large data sets of more than 100 million facts. We find that while it is possible to implement diamonds in SQL, it is inefficient. Indeed, our custom implementation can be a hundred times faster than popular database engines (including a row-store and a column-store).

1. Introduction

An analyst often wants to focus on an *interesting* part of her data set. Sometimes this means she wants to focus on only some attribute values. For example, she might select only the data related to the cities of Montreal and Toronto between the months of July and October. This operation is a *dice* (Section 3.1). Unfortunately, dicing requires that the analyst know exactly which attribute values she needs. Instead of specifying the attribute values, the analyst might prefer to specify a threshold. For example, she can make an *iceberg query* (Section 6.3): e.g., the cities responsible for at least $10 million in sales.

Unfortunately, it is difficult to apply thresholds over several dimensions. The analyst might have selected cities generating at least a certain volume of sales ($10 million), and then select products responsible for a certain sales volume (say $5 million) in these cities. Unfortunately, after selecting the popular products ($5 million), the constraint on cities ($10 million) may no longer be satisfied. Moreover, the analyst could equally start from a product selection that generates a sales volume of at least $5 million, and then ask which cities have sales of at least $10 million when considering only these products. This could produce a different result.

Instead, we propose diamond dicing. It applies constraints simultaneously on several dimensions in a consistent manner. For example, we may seek the cities with a sales volume of at least $10 million dollars, and products with a sales volume of at least $5 million. We require *both* constraints to be *simultaneously* satisfied. Intuitively, diamond dicing is a multidimensional generalisation of icebergs. It is also an instance of dicing, but one where the analyst need not manually specify the interesting attribute values: instead, as with an iceberg query, the analyst might only specify interesting thresholds (on sales, quantities and so on).

Unlike regular dicing or iceberg queries, the computation of a diamond dice (henceforth called a diamond) is a challenge because of the interaction between the dimensions. Indeed, consider Fig. 1. Applying a threshold of $10 million on sales for the

* Corresponding author. Tel.: +1 5067578570.
E-mail addresses: hazel.webb@unb.ca (H. Webb), lemire@gmail.com (D. Lemire), owen@computer.org (O. Kaser).
URL: http://hazel-webb.com (H. Webb).

0169-023X/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.datak.2013.01.001

	Chicago	Montreal	Miami	Paris	Berlin	Totals
TV	3.4	0.9	0.1	0.9	2.0	7.3
Camcorder	0.1	1.4	3.1	2.3	2.1	9.0
Phone	0.2	6.4	2.1	3.5	0.1	12.3
Camera	0.4	2.7	5.3	4.6	3.5	16.5
GameConsole	3.2	0.3	0.3	2.1	1.5	7.4
DVDPlayer	0.2	0.5	0.5	2.2	2.3	5.7
Totals	7.5	12.2	11.4	15.6	11.5	58.2

Fig. 1. Sales (in million dollars): the shaded region is a *diamond* where stores in selected cities need to have sales above $10 million whereas products need sales above $5 million.

cities would eliminate Chicago, whereas applying the $5-million threshold on products would not terminate any product. However, once the shops in Chicago are closed, the products TV and Game Console fall below the threshold of $5-million.[1] We cannot stop now, after processing each dimension once: removal of these products causes the removal of the Berlin store and, finally, the termination of the DVD Player product-line. Thus, simultaneously satisfying constraints on several dimensions may require several iterations.

We must also provide guidance regarding the selection of the thresholds. In our example based on Fig. 1, we used two thresholds ($10 million for stores and $5 million for products) — but what if the analyst does not have specific thresholds in mind? As a sensible default, we might put the same threshold k on both stores and products. If k is too high, the diamond is empty. So we might seek κ, which is the largest value of k so that the diamond is not empty. This value κ could be an interesting default threshold for the analyst. In our example, $\kappa = 7.4$ and the corresponding diamond comprises the attribute values Phone, Camera, Montreal, Miami, and Paris. Within this dice, all cities and products have at least $7.4 million in sales. We present and test efficient algorithms for finding κ (starting in Section 3.3).

Our next section presents several motivating examples. Then we present formal definitions in Section 3. In particular, we show that our definition of a diamond is sound by proving that there is a unique solution to the diamond query. In Section 4, we present efficient algorithms to compute diamonds. We review experimentally the efficiency of our algorithms in Section 5. Finally, we review related work.[2]

2. Motivating examples

We consider example applications to further motivate diamond dicing. We show how diamonds allowed us to find facts that surprised us in different applications.

2.1. Bibliometrics example

Consider a bibliographic table with columns for author and venue. Perhaps we want to analyse the publication habits of professors, but much work would be required to identify precisely which authors are professors. However, perhaps we can assume that most authors without at least 5 publications, in venues where professors publish, are not professors. The diamond with a threshold of 5 publications per author and a threshold of one publication (from these authors) per venue will exclude them. This diamond is the largest author-venue subcube where authors have 5 publications each in selected venues, and where selected venues each have at least one publication from selected authors.

For illustration, we processed conference publication data available from DBLP [2]; the data and details of its preparation are given elsewhere [3]. See Table 1 for some characteristics of diamonds in this data. We find that the diamond corresponding to "professors" prunes about 82% of all authors (115,341 out of 640,674). Maybe surprisingly, it only prunes 4 venues.[3] A similar result remains true if we compute the diamond corresponding to prolific "professors" having published at least 50 papers: out of 5065, only 249 venues are pruned. Yet this diamond contains only 4790 authors out of 640,674 possible authors, which is a selective group (less than 1%). Setting high thresholds is particularly useful in obtaining smaller, more easily analysed, sets of data. For these purposes, we built an interactive tool that finds the highest thresholds generating non-empty diamonds. For example, we may query for the largest value of κ such that the following diamond is not empty: authors with at least κ publications each in retained venues, and retained venues each with at least κ publications from retained authors. In this case, the answer is $\kappa = 119$. We found this occurrence surprising. This diamond contains 11 prolific authors in the area of digital hardware and computer-aided design, who publish in 7 venues.

In a modified form of the bibliometrics cube, we associated each publication with a main keyword, obtaining a 3-dimensional cube [4]. Putting a threshold on the keyword dimension can restrict analysis to popular or mainstream topics.

Consider constraining the authors to have at least 108 publications (on mainstream topics, in popular venues), the topics to have at least 6 occurrences (by prolific authors, in popular venues), and the venues to have at least 20 publications (by prolific

[1] The sum of TV sales is now 3.9, and the sum of Game Console sales is 4.2.
[2] Our work extends a conference paper [1] where a single algorithm was tested over small data sets.
[3] If we require that each author published in at least 5 *different* venues, then we prune about 86% of authors, and only 5 venues.