Smart Drill-Down

Manas Joglekar
Stanford University
manasrj@stanford.edu

Hector Garcia-Molina
Stanford University
hector@cs.stanford.edu

Aditya Parameswaran
University of Illinois (UIUC)
adityagp@illinois.edu

ABSTRACT

We present *smart drill-down*, an operator for interactively exploring a relational table to discover and summarize "interesting" groups of tuples. Each group of tuples is described by a *rule*. For instance, the rule $(a, b, \star, 1000)$ tells us that there are a thousand tuples with value a in the first column and b in the second column (and any value in the third column). Smart drill-down presents an analyst with a list of rules that together describe interesting aspects of the table. The analyst can tailor the definition of interesting, and can interactively apply smart drill-down on an existing rule to explore that part of the table. We demonstrate that the underlying optimization problems are NP-HARD, and describe an algorithm for finding the approximately optimal list of rules to display when the user uses a smart drill-down, and a dynamic sampling scheme for efficiently interacting with large tables. Finally, we perform experiments on real datasets on our experimental prototype to demonstrate the usefulness of smart drill-down and study the performance of our algorithms.

1. INTRODUCTION

Analysts often use OLAP (Online Analytical Processing) operations such as drill down (and roll up) [1] to explore relational databases. These operations are very useful for analytics and data exploration and have stood the test of time; all commercial OLAP systems in existence support these operations. (Recent reports estimate the size of the OLAP market to be $10+ Billion [3].)

However, there are cases where drill down is ineffective; for example, when the number of distinct values in a column is large, vanilla drill down could easily overwhelm analysts by presenting them with too many results (i.e., aggregates). Further, drill down only allows us to instantiate values one column at a time, instead of allowing simultaneous drill downs on multiple columns—this simultaneous drill down on multiple columns could once again suffer from the problem of having too many results, stemming from many distinct combinations of column values.

In this paper, we present a new interaction operator that is an extension to a traditional drill down operator, aimed at providing *complementary* functionality to drill down in cases where drill down is ineffective. We call our operator *smart drill down*. At a high level, smart drill down lets analysts zoom into the more "interesting" parts of a table or a database, with fewer operations, and without having to examine as much data as traditional drill down. Note that our goal is *not* to replace traditional drill down functionality, which we believe is fundamental; instead, our goal is to provide auxiliary functionality which analysts are free to use whenever they find traditional drill downs ineffective.

In addition to presenting this new operator called smart drill down, we also present novel sampling techniques to compute the results for this operator *in an interactive fashion* on increasingly larger databases. Unlike the traditional OLAP setting, these computations require no pre-materialization, and can be implemented within or on top of any relational database system.

Store	Product	Region	Count	Weight
⋆	⋆	⋆	6000	0

Table 1: Initial summary

Store	Product	Region	Count	Weight
⋆	⋆	⋆	6000	0
▷ Target	bicycles	⋆	200	2
▷ ⋆	comforters	MA-3	600	2
▷ Walmart	⋆	⋆	1000	1

Table 2: Result after first smart drill down

The best way to explain smart drill down is through a simple example.

Example 1. *Consider a table with columns 'Department Store', 'Product', 'Region' and 'Sales'. Suppose an analyst queries for tuples where Sales were higher than some threshold, in order to find the best selling products. If the resulting table has many tuples, the analyst can use traditional drill down to explore it. For instance, the system may initially tell the analyst there are 6000 tuples in the answer, represented by the tuple (⋆, ⋆, ⋆,* 6000, 0*), as shown in Table 1. The ⋆ character is a wildcard that matches any value in the database. The Count attribute can be replaced by a Sum aggregate over some measure column, e.g., the total sales. The right-most Weight attribute is the number of non-⋆ attributes; its significance will be discussed shortly. If the analyst drills down on the Store attribute (first ⋆), then the operator displays all tuples of the form (X, ⋆, ⋆, C,* 1*), where X is a Store in the answer table, and C is the number of tuples for X (or the aggregate sales for X).*

Instead, when the analyst uses smart drill down on Table 1, he obtains Table 2. The (⋆, ⋆, ⋆, 6000*) tuple is expanded into* 3 *tuples that display noteworthy or interesting drill downs. The number* 3 *is a user specified parameter, which we call* k*.*

For example, the tuple (Target, bicycles, ⋆, 200, 2*) says that there are* 200 *tuples (out of the 6000) with Target as the first column value and bicycle as the second. This fact tells the analyst that Target is selling a lot of bicycles. The next tuple tells the analyst that comforters are selling well in the MA-3 region, across multiple stores. The last tuple states that Walmart is doing well in general over multiple products and regions. We call each tuple in Table 2 a* rule *to distinguish it from the tuples in the original table that is being explored. Each rule summarizes the set of tuples that are described by it. Again, instead of Count, the operator can display a Sum aggregate, such as the total Sales.*

Say that after seeing the results of Table 2, the analyst wishes to dig deeper into the Walmart tuples represented by the last rule. For instance, the analyst may want to know which states Walmart

Store	Product	Region	Count	Weight
⋆	⋆	⋆	6000	0
▷ Target	bicycles	⋆	200	2
▷ ⋆	comforters	MA-3	600	2
▷ Walmart	⋆	⋆	1000	1
▷ ▷ Walmart	cookies	⋆	200	2
▷ ▷ Walmart	⋆	CA-1	150	2
▷ ▷ Walmart	⋆	WA-5	130	2

Table 3: Result after second smart drill down

has more sales in, or which products they sell the most. In this case, the analyst clicks on the Walmart rule, obtaining the expanded summary in Table 3. The three new rules in this table provide additional information about the 1000 *Walmart tuples. In particular, one of the new rules shows that Walmart sells a lot of cookies; the others show it sells a lot of products in the regions CA-1 and WA-5.*

When the analyst clicks on a rule r, smart drill down expands r into k sub-rules that as a set are deemed to be "interesting." (We discuss other smart drill down operations in Section **??***.) There are three factors that make a rule set interesting. One is if it contains rules with high Count (or total sales) fields, since the larger the count, the more tuples are summarized. A second factor is if the rules have high weight (number of non-⋆ attributes). For instance, the rule (Walmart, cookies, AK-1,* 200*,* 3*) seems more interesting than (Walmart, cookies, ⋆,* 200*,* 2*) since the former tells us the high sales are concentrated in a single region. A third desirability factor is diversity: For example, if we already have the rule (Walmart, ⋆, ⋆,* 1000*,* 1*) in our set, we would rather have the rule (Target, bicycles, ⋆,* 200*,* 2*) than (Walmart, bicycles, ⋆,* 200*,* 2*) since the former rule describes tuples that are not described by the first rule.*

In this paper we describe how to combine or blend these three factors in order to obtain a single desirability score for a set of rules. Our score function can actually be tuned by the analyst (by specifying how weights are computed), providing significant flexibility in what is considered a good set of rules. We also present an efficient optimization procedure to maximize score, invoked by smart drill down to select the set of k rules to display.

Compared to traditional drill down, our smart drill down has two important advantages:

- Smart drill down limits the information displayed to the most interesting k facts (rules). With traditional drill down, a column is expanded and *all* attribute values are displayed in arbitrary order. In our example, if we drill down on say the store attribute, we would see all stores listed, which may be a very large number.
- Smart drill down explores several attributes to open up together, and automatically selects combinations that are interesting. For example, in Table 2, the rule (Target, bicycles, ⋆, 200, 2) is obtained after a single drill down; with a traditional approach, the analyst would first have to drill down on Store, examine the results, drill down on Product, look through all the displayed rules and then find the interesting rule (Target, bicycles, ⋆, 200, 2).

Incidentally, note that in the example we only described one type of smart drill down, where the analyst selects a *rule* to drill down on (e.g., the Walmart rule going from Table 2 to Table 3. In Section **??** we describe another option where the analyst clicks on a ⋆ in a column to obtain rules that have non-⋆ values in that column.

Our work on smart drill down is related to table summarization and anomaly detection [5, 4, 6, 2]. These papers mostly focus on giving the most "surprising" information to the user, i.e., information that would minimize the Kullback-Liebler(KL) divergence between the resulting maximum entropy distribution and the actual value distribution. For instance, if a certain set of values occur together in an unexpectedly small number of tuples, that set of values may be displayed to the user. In contrast, our algorithm focuses on rules with high counts, covering as much of the table as possible. Furthermore, our summarization is couched in an interactive environment, where the analyst directs the drill down and can tailor the optimization criteria. Nevertheless, one can envision extending traditional and smart drill down to provide an additional option of anomaly detection. We discuss related work in detail in Section **??**.

To reiterate, our chief contribution in this paper is the *smart drill down* interaction operator, an extension of traditional drill down, aimed at allowing analysts to zoom into the more "interesting" parts of a dataset. In addition to this operator, we develop techniques to support this operator on increasingly larger datasets:

- *Basic Interaction:* We demonstrate that finding the optimal list of rules is NP-HARD, and we develop an algorithm to find the approximately optimal list of rules to display when the user performs a smart drill down operation.
- *Dynamic Sample Maintenance:* To improve response time on large tables, we formalize the problem of dynamically maintaining samples in memory to support smart drill down. We show that optimal identification of samples is once again NP-HARD, and we develop an approximate scheme for dynamically maintaining and using multiple samples of the table in memory.

We have developed a *fully functional and usable prototype tool* that supports the smart drill-down operator. From this point on, when we provide result snippets, these will be screenshots from our prototype tool. Our prototype tool also supports traditional drill-down: smart drill-down can be viewed as a generalization of traditional drill-down (with the weighting function set appropriately). In Section **??**, we compare smart drill-down with traditional drill-down and show that smart drill-down returns considerably better results.

Overview of paper:

- In Section **??**, we formally define smart drill down. After that, we describe different schemes for weighting rules, and our interactive user interface.
- In Section **??**, we present our algorithms for finding optimal sets of rules, as well as our dynamic sampling schemes for dealing with large tables.
- Based on our implemented smart drill down, in Section **??** we experimentally evaluate performance on real datasets, and show additional examples of smart drill down in action.
- We describe related work in Section **??**, and conclude in Section **??**.

2. SUMMARY OF CONTRIBUTIONS

3. DEMO OVERVIEW

4. REFERENCES

[1] A. Bosworth, J. Gray, A. Layman, and H. Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals. Technical report, Microsoft Research, 1995.

[2] K. E. Gebaly, P. Agrawal, L. Golab, F. Korn, and D. Srivastava. Interpretable and informative explanations of outcomes. *PVLDB*, pages 61–72, 2014.

[3] R. Kalakota. Gartner: Bi and analytics a $12.2 billion market, july 2013 (retrieved october 30, 2014).

[4] S. Sarawagi. User-adaptive exploration of multidimensional data. In *VLDB*, pages 307–316, 2000.

[5] S. Sarawagi. User-cognizant multidimensional analysis. *The VLDB Journal*, pages 224–239, 2001.

[6] S. Sarawagi, R. Agrawal, and N. Megiddo. Discovery-driven exploration of olap data cubes. In *EDBT*, pages 168–182, 1998.