# Column Heterogeneity as a Measure of Data Quality

Bing Tian Dai
National Univ. of Singapore
daibingt@comp.nus.edu.sg

Nick Koudas
University of Toronto
koudas@cs.toronto.edu

Beng Chin Ooi
National Univ. of Singapore
ooibc@comp.nus.edu.sg

Divesh Srivastava
AT&T Labs–Research
divesh@research.att.com

Suresh Venkatasubramanian
AT&T Labs–Research
suresh@research.att.com

## ABSTRACT

Data quality is a serious concern in every data management application, and a variety of quality measures have been proposed, including accuracy, freshness and completeness, to capture the common sources of data quality degradation. We identify and focus attention on a novel measure, *column heterogeneity*, that seeks to quantify the data quality problems that can arise when merging data from different sources. We identify desiderata that a column heterogeneity measure should intuitively satisfy, and discuss a promising direction of research to quantify database column heterogeneity based on using a novel combination of *cluster entropy* and *soft clustering*. Finally, we present a few preliminary experimental results, using diverse data sets of semantically different types, to demonstrate that this approach appears to provide a robust mechanism for identifying and quantifying database column heterogeneity.

## 1. MOTIVATION

Data quality is a serious concern in every data management application, severely degrading common business practices, and industry consultants often quantify the adverse impact of poor data quality in the billions of dollars annually. Data quality issues have been studied quite extensively in the literature (e.g., [3, 5, 1]). In particular, a variety of quality measures have been proposed, including accuracy, freshness and completeness, to capture the common sources of data quality degradation [6, 9]. Data profiling tools like Bellman [4] compute concise summaries of the values in database columns, to identify various errors introduced by poor database design; these include approximate keys (the presence of null values and defaults in a column may result in the approximation) and approximate functional dependencies in a table (possibly due to inconsistent values). This vision paper identifies and focuses attention on a novel measure, *column heterogeneity*, that seeks to quantify the data quality problems that can arise when merging data from different sources.

Textbook database design teaches that it is desirable for a database column to be homogeneous, i.e., all values in a column should be of the same "semantic type". For example, if a database contains email addresses, social security numbers, phone numbers, machine names and IP addresses, these semantically different types of values should be represented in separate columns. For example, the column in Figure 1(a) contains only email addresses and is quite homogeneous, even though there appears to be a wide diversity in the actual set of values present. Such homogeneity of database column values has obvious advantages, including simplicity of application-level code that accesses and modifies the database.

In practice, operational databases evolve over time to contain a great deal of "heterogeneity" in database column values. Often, this is a consequence of large scale data integration efforts that seek to preserve the "structure" of the original databases in the integrated database, to avoid having to make extensive changes to the application level code. For example, one application might use email addresses as a unique customer identifier, while another might use phone numbers for the same purpose; when their databases are integrated into a common database, it is feasible that the CUSTOMER_ID column contains both email addresses and phone numbers, both represented as strings, as illustrated in Figure 1(b). A third independently developed application that used, say, social security numbers as a customer identifier might then add such values to the CUSTOMER_ID column, when its database is integrated into the common database. As another example, two different inventory applications might maintain machine domain names (e.g., abc.def.com) and IP addresses (e.g., 105.205.105.205) in the same MACHINE_ID column for the equivalent task of identifying machines connected to the network. While these examples may appear "natural" since all of these semantically different types of values have the same function, namely, to serve as a customer identifier or a machine identifier, potential data quality problems can arise in databases accessed and modified by legacy applications that are unaware of the heterogeneity of values in the column.

For example, an application that assumes that the CUSTOMER_ID column contains only phone numbers might choose to "normalize" column values by removing all special characters (e.g., '-', '.') from the value, and writing it back into the database. While such a transformation is appropriate for phone numbers, it would clearly mangle the email addresses represented in the column and can severely degrade common business practices. For instance, in our previous example, the unanticipated transformation of email addresses in the CUSTOMER_ID column (e.g., "john.smith@noname.org" to "johnsmith@nonameorg") may mean that a large number of customers are no longer reachable.

Locating poor quality data in large operational databases is a non-trivial task, especially since the problems may not be due to the data alone, but also due to the interactions between the data and the multitude of applications that access this data (as the previous


*CleanDB* Seoul, Korea, 2006

| CUSTOMER_ID |
|---|
| lkjkjjk@321.zzz.info |
| h8742@yyy.com |
| kkjj+@haha.org |
| qwerty@keyboard.us |
| 555-1212@fax.in |
| alpha@beta.ga |
| john.smith@noname.org |
| jane.doe@1973law.us |
| gwb.dc@universe.gov |
| jamesbond.007@action.com |

(a)

| CUSTOMER_ID |
|---|
| lkjkjjk@321.zzz.info |
| h8742@yyy.com |
| kkjj+@haha.org |
| qwerty@keyboard.us |
| 555-1212@fax.in |
| (908)-555.1234 |
| 973-360-0000 |
| 360-0007 |
| 8005551212 |
| (877)-807-4596 |

(b)

| CUSTOMER_ID |
|---|
| lkjkjjk@321.zzz.info |
| h8742@yui.com |
| kkjj+@haha.org |
| qwerty@keyboard.us |
| 555-1212@fax.in |
| alpha@beta.ga |
| john.smith@noname.org |
| jane.doe@1973law.us |
| gwb.dc@universe.gov |
| (877)-807-4596 |

(c)

| CUSTOMER_ID |
|---|
| 123-45-6789 |
| 135-79-2468 |
| 159-24-6837 |
| 789-12-3456 |
| 987-65-4321 |
| (908)-555.1234 |
| 973-360-0000 |
| 360-0007 |
| 8005551212 |
| (877)-807-4596 |

(d)

**Figure 1: Example homogeneous and heterogeneous columns.**

example illustrates). Identifying heterogeneous database columns becomes important in such a scenario, permitting data quality analysts to then focus on understanding the interactions of applications with data in such columns, rather than having to simultaneously deal with the tens of thousands of columns in today's complex operational databases. If an analyst determines that a problem exists, remedial actions can include:

- modification of the applications to explicitly check for the semantic type of data (phone numbers, email addresses, etc.) assumed to exist in the table, or
- a horizontal splitting of the table to force homogeneity, along with a simpler modification of the applications accessing this table to access and update the newly created tables instead.

We next identify desiderata that a column heterogeneity measure should intuitively satisfy, and discuss a promising direction of research to quantify database column heterogeneity.

## 2. HETEROGENEITY: DESIDERATA

Consider the example shown in Figure 1. This illustrates many of the issues that need to be considered when coming up with a suitable measure for column heterogeneity.

**Number of Semantic Types:** Many semantically different types of values (email addresses, phone numbers, social security numbers, circuit identifiers, IP addresses, machine domain names, customer names, etc.) may be represented as strings in a column, with no *a priori* characterization of the set of possible semantic types present.

Intuitively, the more semantically different types of values there are in a database column, the greater should be its heterogeneity; thus, heterogeneity is better modeled as a numerical value rather than a boolean (yes/no). For example, a column with both email addresses and phone numbers (e.g., Figure 1(b)) can be said to be more heterogeneous than a column with only email addresses (e.g., Figure 1(a)) or only phone numbers.

**Distribution of Semantic Types:** The semantically different types of values in a database column may occur with different frequencies.

Intuitively, the relative distribution of the semantically different types of values in a column should impact its heterogeneity. For example, a column with many email addresses and phone numbers (e.g., Figure 1(b)) can be said to be more heterogeneous than a column that has mainly email addresses with just a few outlier phone numbers (e.g., Figure 1(c)), or vice versa.

**Distinguishability of Semantic Types:** Semantically different types of values may overlap (e.g., social security numbers and phone numbers) or be easily distinguished (e.g., email addresses and phone numbers).

Intuitively, with no a priori characterization of the set of possible semantic types present in a column, we cannot always be sure that a column is heterogeneous, and our heterogeneity measure should conservatively reflect this possibility.

The more easily distinguished are the semantically different types of values in a column, the greater should be its heterogeneity. For example, a column with roughly equal numbers of email addresses and phone numbers (e.g., Figure 1(b)) can be said to be more heterogeneous than a column with roughly equal numbers of phone numbers and social security numbers (e.g., Figure 1(d)), due to the greater similarity between the values (and hence the possibility of being of the same unknown semantic type) in the latter case.

## 3. QUANTIFYING HETEROGENEITY

We now discuss approaches to quantify database column heterogeneity that meet the desiderata outlined above.

**Number of Semantic Types:** A first approach to obtaining a heterogeneity measure is to use a *hard clustering*. By partitioning values in a database column into clusters, we can get a sense of the number of semantically different types of values in the data. However, merely counting the number of clusters does not suffice to quantify heterogeneity. Two additional issues, as outlined above, make the problem challenging: the relative sizes and the distinguishability of the clusters. A few phone numbers in a large collection of email addresses (e.g., Figure 1(c)) may look like a distinct cluster, but should not impact the heterogeneity of the column as much as having a significant number of phone numbers with the same collection of email addresses (e.g., Figure 1(b)). Again, a social security number (see the first few values in Figure 1(d)) may look similar to a phone number, and we would like the heterogeneity measure to reflect this overlap of sets of values, as well as be able to capture the idea that certain data might yield clusters that are close to each other, and other data might yield clusters that are far apart.

**Distribution of Semantic Types:** To take into account the relative sizes of the (possibly multiple) clusters, *cluster entropy* is a better measure for quantifying heterogeneity of data in a database column than merely counting the number of clusters. Cluster en-

tropy is computed by assigning a "probability" to each cluster equal to the fraction of the data values it contains, and computing the entropy of the resulting distribution [2]. Consider a hard clustering $T = \{t_1, t_2, \ldots t_k\}$ of a set of $n$ values $X$, where cluster $t_i$ has $n_i$ values, and denote $p_i = n_i/n$. Then the *cluster entropy* of the hard clustering $T$ is the entropy of the cluster size distribution, defined as $\sum p_i \ln(1/p_i)$. By using cluster entropy, the mixture of email addresses and phone numbers in column Figure 1(b) would have a higher value of heterogeneity than the data in Figure 1(c), which consists of a few phone numbers in a collection of mainly email addresses.

**Distinguishability of Semantic Types:** The cluster entropy of a hard clustering does not effectively take into account distinguishability of semantic types in a column. For example, given a column with an equal number of phone numbers and social security numbers (e.g., Figure 1(d)), hard clustering could either determine the column to have one cluster (in which case its cluster entropy would be 0, which is the same as that of a column with just phone numbers) or have two equal sized clusters (in which case its cluster entropy would be $\ln(2)$, which is the same as that of a column with equal numbers of phone numbers and email addresses). Intuitively, however, the heterogeneity of such a column should be somewhere in between these two extremes to capture the uncertainty in assigning values to clusters due to the syntactic similarity of values. *Soft clustering* has the potential to address this problem; each data value in soft clustering has the flexibility of assigning a probability distribution for its cluster membership, instead of belonging to a single cluster (equivalently, assigning its entire probability distribution to a single cluster), as in hard clustering. Heterogeneity can now be computed as the cluster entropy of the soft clustering.

To summarize, the desiderata that a column heterogeneity measure should depend on the number, the distribution and the distinguishability of the semantic types of string values in a column have the potential of being satisfied by using a novel combination of *cluster entropy* and *soft clustering*. We next discuss some promising results that we have obtained by following this research direction.

# 4. PRELIMINARY RESULTS

As a concrete realization of our vision, we present a few experimental results using diverse data sets of semantically different types, mixed together in various ways, to provide different levels of heterogeneity.

**Data Sets:** We consider mixtures of four different data sets. `email` is a set of 509 email addresses collected from attendees at the 2001 SIGMOD/PODS conference, `ID` is a set of 609 employee identifiers, `phone` is a diverse collection of 3064 telephone numbers, and `circuit` is a set of 1778 network circuit identifiers. Strings in `ID` and `phone` are numeric (`phone` data contains the period as well). Strings in `email` and `circuit` are alphanumeric, and may contain special characters like '@' and '-'.

**Soft Clustering:** We will use the *Information Bottleneck Method*, developed by Tishby et al. [8], and implemented by Slonim in his thesis as the algorithm `iIB` [7], to compute a soft clustering of the data sets. Intuitively, `iIB` takes as input a joint distribution $(X, Y)$, where $x \in X$ represents a string value in the data set, $y \in Y$ is chosen to represent tokens ($q$-grams) extracted from the string values, and the joint distribution reflects an entropy weighting of the tokens. The output of `iIB` is a cluster membership distribution $p(T|x)$ for each $x$, representing the conditional probability

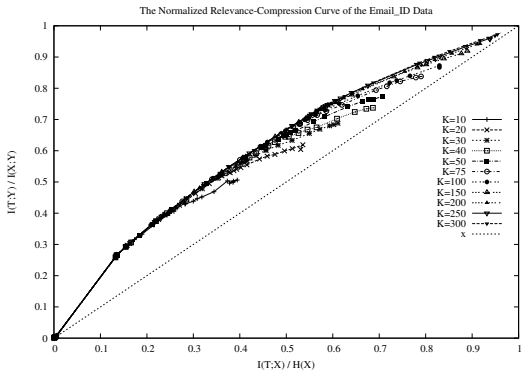


**Figure 2: Rate-Distortion curve for example data.**

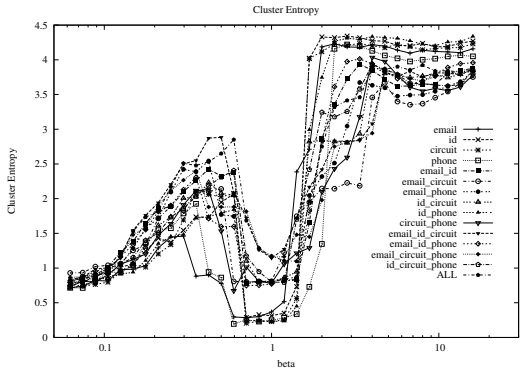


**Figure 3: Cluster entropy as a function of $\beta$. The x-axis plots a normalized version of $\beta$ on a logscale.**

that string value $x$ is placed in cluster $t \in T$.

**Canonical Rule:** `iIB` uses a parameter $\beta$ that trades off cluster quality against cluster compression; increasing $\beta$ increases cluster quality while decreasing cluster compression. Interestingly, *for all the data sets*, there is a unique value of $\beta$ given by $\beta^* = H(X)/I(X;Y)$ (where $H(X)$ is the entropy of the data values $X$ and $I(X;Y)$ is the mutual information between the data values $X$ and the tokens $Y$), which marks the "point of diminishing returns"; after this $\beta$ value, the loss we suffer from reducing the (normalized) cluster compression is not paid for by a commensurate increase in (normalized) cluster quality. This behavior can be observed in the rate distortion curve for our example data, shown in Figure 2; this curve is always concave, and the point on the curve with a slope of 1 identifies $\beta^*$. This is also the point that is the closest to the $(0, 1)$ point, which is the point representing perfect quality with no space penalty.

**Cluster Entropy:** Using the soft clustering output of `iIB` for different values of $\beta$ in the vicinity of $\beta^*$, and computing heterogeneity by combining estimates of the cluster entropies of the various hard clusterings derived from the soft clustering via the soft clustering distribution, we empirically observed that the cluster entropy is minimized at $\beta^*$. This behavior can be observed in Figure 3. Further, the relative ordering of cluster entropy values obtained at $\beta = \beta^*$ is consistent with the expected relative heterogeneities of these data sets, as shown in Figure 4. Specifically, all the individual data sets have very small cluster entropies, and are distinguishable from the mixtures. Further, mixtures of two data

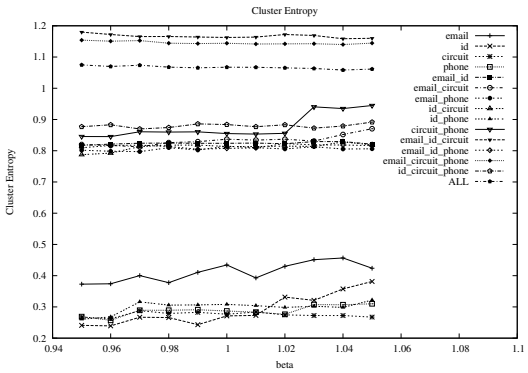


**Figure 4: Cluster entropy as a measure of heterogeneity. The x-axis plots a normalized version of $\beta$ on a logscale.**

sets in general have lower cluster entropy than mixtures of three and four data sets. We observe that as the number of elements in the mixture increases, the heterogeneity gap decreases, and that the separations are not strict for the more heterogeneous sets; this is natural, as individual data sets may have characteristics that are somewhat similar (for example, `ID` and `phone`).

**Validating the Soft Clustering:** Cluster entropy appears to capture our intuitive notion of heterogeneity. However, it is derived from a soft clustering returned by the `iIB` algorithm. Does that soft clustering actually reflect natural groupings in the data? It turns out that this is indeed the case. In Figure 5, we display bitmaps that visualize the clusterings obtained for different mixtures. In this representation, columns are clusters, rows are data values, and darker probabilities are larger. For clarity, we have reordered the rows so that all data elements coming from the same source are together, and we reordered the columns based on their distributional similarities. To interpret the figure, recall that each row of a bitmap represents the cluster membership distribution of a data point. A collection of data points having the same cluster membership distributions represent the same cluster. Thus, notice how the clusters separate out quite cleanly, clearly displaying the different data mixtures. Also observe how, without having to specify $k$, the number of clusters, `iIB` is able to separate out the groups. Further, if we look at Figure 5(d), we notice how the clusters corresponding to `ID` and `phone` overlap and have similar cluster membership distributions, reinforcing our observation that they form two very close (not well-separated) clusters.

To summarize the experimental results, our novel combination of *cluster entropy* and *soft clustering* appears to provide a robust mechanism for identifying and quantifying database column heterogeneity.

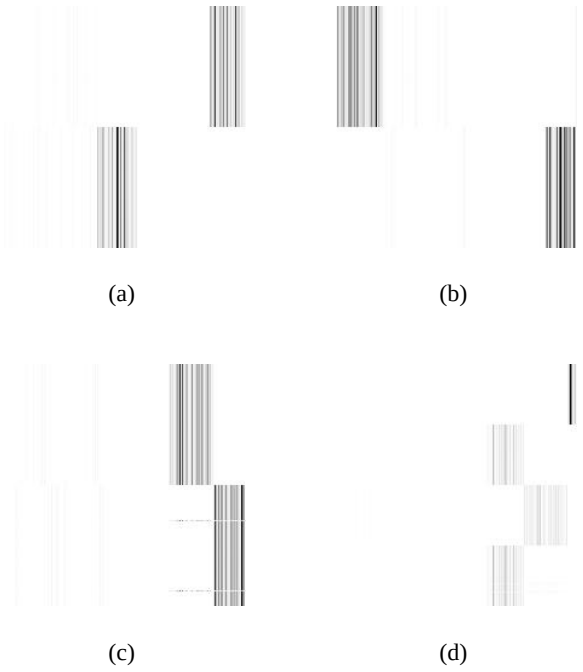


**Figure 5: Soft Clustering of `email`/`ID`, `email`/`circuit`, `circuit`/`phone`, `email`/`ID`/`circuit`/`phone` mixtures.**

## 5. CONCLUSION

In this vision paper, we identified a new data quality measure, column heterogeneity, and outlined a general approach to quantify this measure in database columns. The rapid identification of heterogeneous columns in a database with tens of thousands of columns provides a unique opportunity to understand and characterize the quality of data in today's complex operational databases, using the tools of information theory.

## 6. REFERENCES

[1] C. Batini, T. Catarci, and M. Scannapieco. A survey of data quality issues in cooperative information systems. In *ER*, 2004. Pre-conference tutorial.

[2] T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley-Interscience, New York, NY, USA, 1991.

[3] T. Dasu and T. Johnson. *Exploratory data mining and data cleaning*. John Wiley, 2003.

[4] T. Dasu, T. Johnson, S. Muthukrishnan, and V. Shkapenyuk. Mining database structure or how to build a data quality browser. In *SIGMOD*, 2002.

[5] T. Johnson and T. Dasu. Data quality and data cleaning: An overview. In *SIGMOD*, 2003. Tutorial.

[6] G. Mihaila, L. Raschid, and M.-E. Vidal. Querying "quality of data" metadata. In *Proc. of IEEE META-DATA Conference*, 1999.

[7] N. Slonim. *The Information Bottleneck: Theory and Applications*. PhD thesis, The Hebrew University, 2003.

[8] N. Tishby, F. Pereira, and W. Bialek. The information bottleneck method. In *Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.

[9] J. Widom. Trio: A system for integrated management of data, accuracy, and lineage. In *CIDR*, 2005.