An Entropy-Based Approach to Visualizing Database Structure

Dennis P. Groth Edward L. Robertson [*]
Computer Science, Indiana University,
Bloomington, IN 47405, USA.
Email: {dgroth, edrbtsn}@cs.indiana.edu

Abstract

This paper explores the use of entropy for visualizing database structure. In particular, we show how visualizing the entropy of a relation provides a global perspective on the distribution of values and helps to identify areas within the relation where interesting relationships may be discovered. The type of structure we are interested in discovering is related to functional dependencies. Our approach is not dependent on the underlying domain of the data, providing a view of the dependency landscape within a relation. Using these techniques we described comparative results for a wide variety of synthetic and real data.

1 Introduction

Developing visualizations of database content is of extreme interest to the research community. There are numerous examples of applications developed around a graphical display of database content, including relationship discovery, outlier detection, and trend analysis.

Our approach to visualizing the structural, information content (which is distinct from the data content) of databases is motivated by the discipline of information theory. In particular, we utilize entropy as the basis for our visualizations. Within the field of information theory, entropy is the central concept, related to the encoding of messages. As such, entropy is a statistic that provides a global description of the information content of data. We defer to Section 2 for the definition of entropy, as well as other formal notions. This research utilizes entropy to visualize the structure of a database relation, independent of the underlying domain datatype.

When developing effective visualizations of database content there are three significant challenges that must be addressed. First, we are often faced with high-dimensional data. Second, databases are awash with categorical data, which are often totally lack any meaningful order or scale. Thirdly, the sheer mass of data in even a moderately sized database may overwhelm the user when trying to simply display a two-dimensional scatter plot.

There has, of course, been a great deal of research effort aimed at addressing these challenges. For example, a number of techniques have been applied to high dimensional data, such as parallel coordinate displays, worlds within worlds, dimensional stacking, and grand tour methods.[10, 11, 16, 8] Likewise, the field of information visualization has many techniques for dealing with abstract, categorical data.[3] For techniques dealing with visualizing the contents of large databases see [14, 12, 13].

[*] All authors were supported by NSF Grant IIS-0082407

Whereas the goal of almost all other visualization techniques is to facilitate understanding of particular values, our approach specifically remains aloof from those values in two distinct ways. First, we are interested in large-scale properties of instances – properties that are related to attributes rather than values. For example, a relation with 10 attributes requires 45 different 2D scatter plots to provide the same amount of information our technique provides in a *single* 2D scatter plot. Indeed, the axes for many of the visualizations we present are the collection of attributes in the relation, or characteristics of those attributes, rather than the values in a particular attribute. Second, even within a particular attribute, it is the distribution of values rather than the actual presented values that matter. Indeed, we typically code values as integers to simplify processing.

This second item is a consequence of the notion that the structure of data is independent of data values. Formally, structure is *generic*, in that it is invariant under 1-1 substitution of data values (hence the encoding with integers). For example, functional dependencies describe a particular type of structure - independent of the actual values. The definition of the functional dependency $X \rightarrow Y$, namely "$\forall t_1 \in r, \forall t_2 \in r(t_1[X] = t_2[X] \Rightarrow t_1[Y] = t_2[Y])$", exhibits this formal genericity. Information theory, which uses only probabilities associated with values and not the values themselves, therefore provides techniques that allow us to view structure. As a result, we are able to develop visualizations of database structure that are applicable to any context.

Because entropy-based visualizations show global properties, they are more in tune with natural uses of visualization, where global structure and detail through drill-down are most effective. Figure 1 illustrates how our natural perception - a glance suggests that the left pane is more "function-like", while in fact the right pane exhibits the functional dependency $A \rightarrow B$ while the left does not. While the presence or absence of these functional dependencies is easy to evaluate in Figure 1, with only eight data points, this task becomes increasingly difficult as the number of data points increases. In addition, determining when the data contains an approximate dependency [15], in which a functional dependency holds except for a small number of violations, is equally difficult.

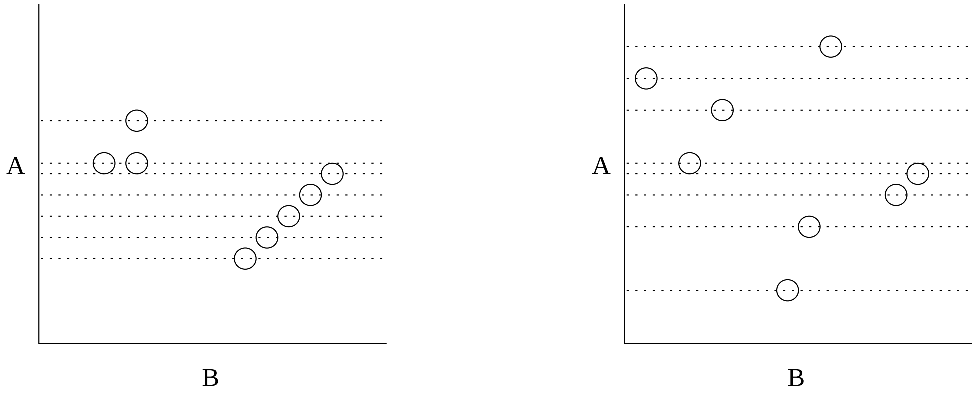

Figure 1: On the left, a scatter plot of data where the functional dependency $A \rightarrow B$ does not hold. On the right, a scatter plot where the functional dependency does hold. The dashed lines have been added to show the alignment of the points along the vertical dimension.

This paper is structured as follows. Section 2 provides the formal notation and definitions used throughout the paper. Section 3 explores the use of entropy to visualize frequency distributions of attribute values in database relations. Section 4 demonstrates the use of entropy for visualizing dependencies, or relationships between attributes. Section 5 provides examples of these these techniques for making broad comparisons of different datasets. Lastly, Section 6 provides future directions and concluding remarks.

1.1 User Interaction

In order to visualize database structure we have developed a system to support interactive displays of database visualizations. Based on an architecture we proposed in [9], the system supports the specification of database queries and the mapping of the query results to a graphical representation. These mappings are entirely under the control of the user.

The system is implemented in Java, using JDBC for query processing. The visualizations are implemented using Java 3D, allowing the user to manipulate the display for standard actions such as rotating, scaling and translating. In addition, the user is able to drill down at any data point to see the raw data values, as well as identifying datapoints that are particularly interesting to them. These selected points can be utilized in subsequent visualizations for comparative purposes.

2 Definitions

In this section we provide the formal notation and definitions used in this paper. Our focus is on visualizing database information, so we begin with the basics of the relational model. Let $R = \{A, B, C, \dots\}$ be a relation schema for instance $\mathbf{r}$. For attribute $A \in R$, A denotes $\{A\}$. Sets of attributes are denoted by $X, Y, Z \subseteq R$. For X and $Y \subseteq R$, XY denotes $X \cup Y$. The notation we use for tuples is $t \in \mathbf{r}$, with $t.A$ representing the value for attribute A in tuple t.

With its genesis in message theory, entropy is defined over a set of messages $M = \{m_1, \dots, m_n\}$, with associated probabilities $P_M = \{p_1, \dots, p_n\}$. The entropy of M is $\mathcal{H}_M = \sum_{i=1}^{n} p_i \log \frac{1}{p_i}$. Entropy provides us with an average cost (in bits) for each message. The upper bound on the entropy of M is log n, which occurs when each message has equal probability. Additional details on entropy, as well as other information theory topics is covered in [4].

For databases, the message set we are interested in is taken from the relation instance $\mathbf{r}$. When projecting attributes from R, we do not eliminate duplicate values, which allows us to compute the probabilities using the counts of each value in the active domain. For example, the probability $P(A = a) = \frac{count(\sigma_{A=a}(\mathbf{r}))}{count(\mathbf{r})}$.

For any set of attributes $X \subseteq R$, we can compute $\mathcal{H}_X$ using an SQL aggregate query. The query is shown in Figure 2 for the case of $A \in R$. For the purposes of the visualizations generated for this research, we pre-compute $\mathcal{H}_A$ for each $A \in R$, as well as $\mathcal{H}_{AB}$ for each $A, B \in R$. Note that $\mathcal{H}_A \leq$ log $|\text{adom}(A)|$, where adom(A) is the active domain of A. When $\mathcal{H}_A =$ log $|\mathbf{r}|$, A is a key.

```
Select   SUM((R1.frequency/R2.rowcount) *
              LOG(2,1/(R1.frequency/R2.rowcount)))
From    (Select    A, COUNT(*) as frequency
         From      R
         Group By  A) as R1,
        (Select    COUNT(*) as rowcount
         From      R) as R2
```

Figure 2: SQL query to calculate the entropy of A

2.1 Information Dependencies

Within the database research field, the concept of functional dependencies is well understood. The functional dependency $A \rightarrow B$ holds in instance $\mathbf{r}$, when for any two tuples $t_1, t_2 \in \mathbf{r}, t_1.A = t_2.A \implies t_1.B = t_2.B$. Functional dependencies always hold for an instance, a particular functional dependency may be specified as a constraint in a database management system.

As is often the case, however, large, complex data rarely exhibits many functional dependencies beyond those specified as constraints. As shown in [6], an *Information Dependency Measure* is defined using entropy. The information dependency measure $\mathcal{H}_{X \rightarrow Y}$ provides a measure indicating the average number of bits we need to use to determine Y if we know a value for X. Another way to look at this measure is in terms of surprise. In other words, how surprising is a particular value for Y when we know X.

The information dependency $\mathcal{H}_{X \rightarrow Y}$ can be calculated by $\mathcal{H}_{XY} - \mathcal{H}_X$. For more details on information dependencies, as well as the proof for this calculation, see [5]. When $\mathcal{H}_{X \rightarrow Y} = 0$, the functional dependency $X \rightarrow Y$ holds. The upper bound on $\mathcal{H}_{X \rightarrow Y}$ is $\mathcal{H}_X + \mathcal{H}_Y$, which is the case of independence of X and Y. Another weakness of using a traditional approach for identifying dependencies is shown in the right pane. We can verify by checking across the display that there are no violations of the dependency $A \rightarrow B$. However, as the number of datapoints increase the task becomes increasingly difficult. In addition, determining when the data contains an approximate dependency [15], in which a functional dependency holds except for a small number of violations, is equally difficult. Figure 1 may be used to illustrate the applicability if the information dependency measure: $\mathcal{H}_{A \rightarrow B}$ is 0.25 in the left pane and 0 in the right. Whereas a visual estimation of approximate functional dependencies does not scale, estimation via $\mathcal{H}_{A \rightarrow B}$ does.

3 Visualizing Distributions

Using the measures defined in Section 2, we turn to the visualization problem addressed in this research. The data we use in our visualization is drawn from a variety of sources, including the U.S. Census [1], the U.C.I. Machine Learning Repository [2], and the Wisconsin Benchmark [7]. The specific dataset we used for the Census was the 1990 Indiana Public Use Microdata Sample (PUMS), which has 125 attributes.

Our first application is the visualization of frequency distributions. An obvious technique for visualizing frequency distributions is to use histograms, with the height of each bar representing the frequency. Figure 3 shows the log of the size of the active domain for each attribute in the U.S. Census (Left) compared to the calculated entropy value for each attribute value (Right). The leftmost bar in each display corresponds to a key in the relation; otherwise, the attributes are arbitrarily ordered.

Note that the height of the bars varies according to the probabilities associated with each value in the active domain, resulting in differences in the heights for the same attribute in each display. To highlight these differences, consider Figure 4, which shows the same information for a subset of the attribute space. The attributes displayed include: Hours Worked Per Week, Immigration Year, Income, Non-farm Income, Farm Income, Interest and Dividend Income, Social Security Income, Public Assistance Income, and Retirement Income. In this case, we can see that certain attributes that have dominant values have their corresponding entropy values change in a dramatic way.

In order to gain an overall view of the attribute space, we can compare $\mathcal{H}_A$ to $\log |\text{adom}(A)|$ using a two-dimensional scatterplot. This visualization is shown in Figure 5, in which the attribute that is a key has been omitted. In the visualization, points that lie on the diagonal have an

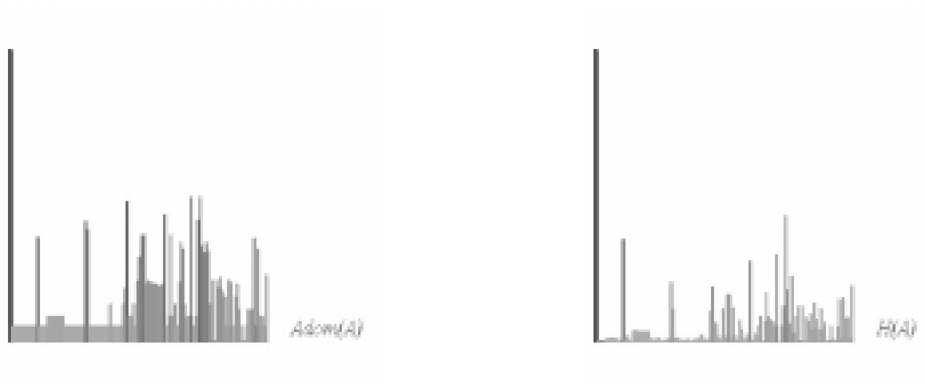

Figure 3: Comparing the size of the active domain for each attribute (Left) to the entropy of each attribute (Right) in the U.S. Census dataset.

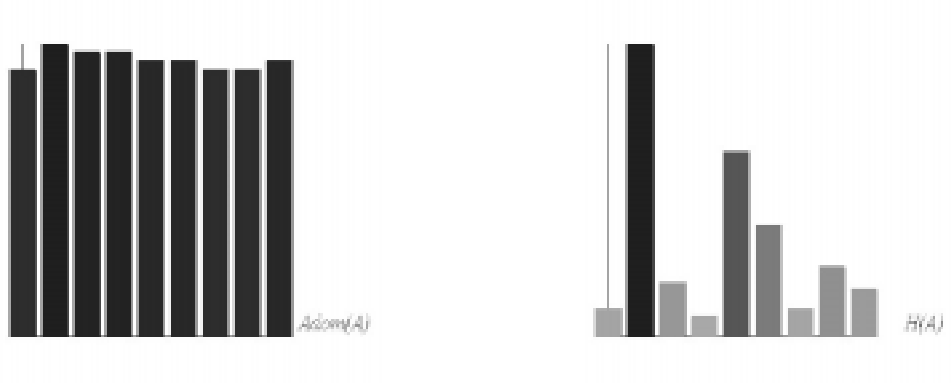

Figure 4: A view of the differences between the size of the active domain for (Left) compared to the entropy values for the same attributes (Right).

(approximately) uniform distribution. The further a point is from the diagonal, the less uniform is the associated distribution.

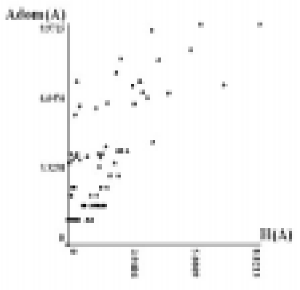

Figure 5: Comparing the entropy of each attribute in the census data to the log of the size of the corresponding active domain.

4 Visualizing Relationships

While the previous section demonstrated the use of entropy to gain insight into frequency distributions within database relations. In this section we extend the technique in order to explore relationships between attributes. In particular we utilize the information dependency measure described earlier to visualize these relationships.

While we have formally described the concept of an information dependency, we have not yet discussed visualizing them. Figure 6 characterizes the space of $\mathcal{H}(AB)\times\mathcal{H}(A)$, which is encountered when visualizing the values in a 2D scatter plot. This type of visualization allows us to get an overall view of all possible attribute pairs in a compact space. A critical advantage of this approach is that the visualizations do not depend on the actual values or types of data.

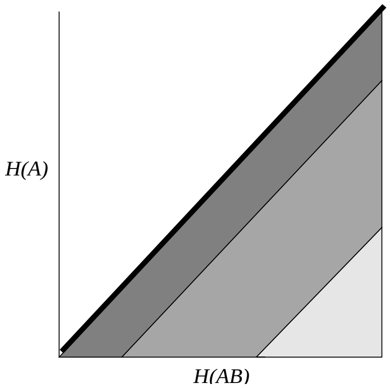

Figure 6: Characterizing the space $\mathcal{H}(AB) \times \mathcal{H}(A)$.

The dark, diagonal line in the figure represents functional dependencies in the relation. Above the diagonal the space is empty, since the lower bound of $\mathcal{H}(AB)$ is $\mathcal{H}(A)$. As you move away and below the diagonal, the structure becomes less like a functional dependency. There is an area of potential interest close to the diagonal, in which the space represents approximate functional dependencies that are *almost* a pure functional dependency. The space furthest from the diagonal contains attribute pairs that are independent of a dependency.

Figure 7 shows a scatter plot comparing $\mathcal{H}_{AB}$ to $\mathcal{H}_A$ for the census data. Recall that the information dependency $\mathcal{H}_{A\to B} = \mathcal{H}_{AB} - \mathcal{H}_A$. We can easily see that there are certain attributes that behave relatively consistently as ranges of functional dependencies (the "B" position); these are the origin of the diagonal bands. Similarly, many attributes behave consistently as domains ("A" position), corresponding to horizontal bands.

This suggests a more detailed examination using three dimensions, comparing $\mathcal{H}_{AB}$, $\mathcal{H}_A$ and $\mathcal{H}_B$. In this case, the shape of the space is constrained in the Z-dimension by $\mathcal{H}_B$. Points of particular interest are those that have $\mathcal{H}_{A\to B} = 0$ (or, very near 0), and $\mathcal{H}_B$ is interestingly large. The determination of what is interesting in this context is dependent on the application. However, when $\mathcal{H}_B$ is large and $\mathcal{H}_{A\to B} = 0$, the relation may be decompose losslessly into smaller subrelations, which saves space and may dramatically improve query performance. As seen in Figure 8 (Left), we can see the space does contain points that are very near the diagonal (approximate dependencies) and that have somewhat large $\mathcal{H}_B$ values. The same data with the key attribute filter out is shown on the right.

In this case, we find some points along the line bisecting the space. These points indicate attributes that are functionally co-dependent. In addition, In the upper right hand corner are

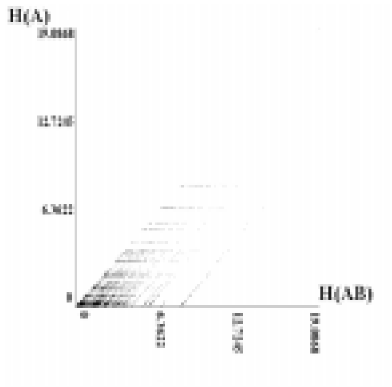

Figure 7: Scatter plot comparing $\mathcal{H}(AB)$ to $\mathcal{H}(A)$.

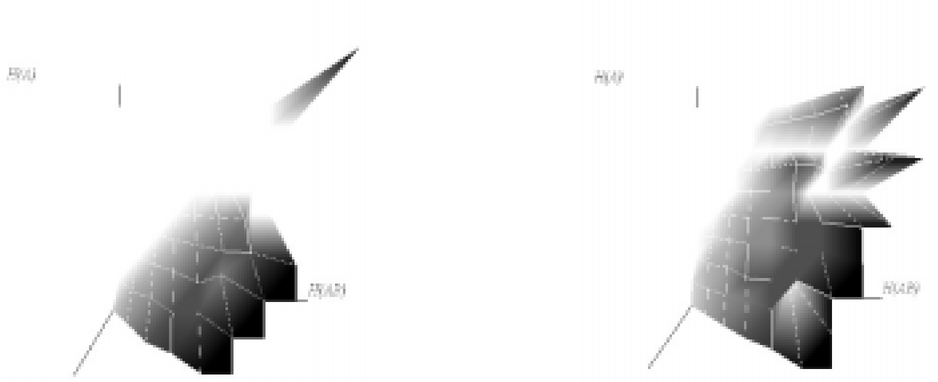

Figure 8: 3D surface plot comparing $\mathcal{H}_{AB}$, $\mathcal{H}_A$ and $\mathcal{H}_B$. The right plot factors out the attribute that is a key.

points related to a key within the relation. Notice that $\mathcal{H}_A$ is fixed for these points, with $\mathcal{H}_B$ determining the position.

In addition to identifying potentially interesting relationships between attributes, the visualizations also highlight additional information. For example, when $\mathcal{H}_A$ is low and $\mathcal{H}_{AB} - \mathcal{H}_A = 0$, it is possible to decompose the original relation into smaller sub-relations, taking advantage of space savings. When the difference is very near to zero, you may decide to ignore the noise entirely and clean the data by removing the noisy data.

4.1 Drilling Down

The discussion thus far has involved global characterizations of attributes, but information-based visualization can also drill-down to reveal local structures. This makes use of the fact that the functional dependency $A \to B$ holds iff $\mathcal{H}_{A \to B} = 0$ and thus the quantity $\mathcal{H}_{A \to B}$ is a measure of how close $A \to B$ is to holding in an instance. The characterization of $\mathcal{H}_{A \to B}$ as $\sum_{a \in A} p(a) \times \mathcal{H}_B(\sigma_{A=a}(r))$ suggests that the "landscape" of $p(a)$ and $\mathcal{H}_B(\sigma_{A=a}(r))$ might reveal something about local structure related to $A \to B$. Indeed this is the case, as we see in examples from the census data.

The first example examines `AGE` → `DEPART` (with `AGE` as A and `DEPART` as B). The plot of $p(a)$ versus $\mathcal{H}_B(\sigma_{A=a}(r))$, shown in the first panel of Figure 9, has several interesting features:

1. `AGE` values with low probability have low diversity of associated `DEPART`, and this holds uniformly

2. the relationship of $\mathcal{H}_B(\sigma_{A=a}(r))$ versus $p(a)$ is essentially a smooth function for low $p(a)$ values

3. when $p(a)$ exceeds a certain value, the corresponding $\mathcal{H}_B(\sigma_{A=a}(r))$ is typically close to the maximum; this cutoff is surprisingly sharp

4. there are a few higher probability `AGE`s which differ from the typical by having $\mathcal{H}_B(\sigma_{A=a}(r))$ values that are lower or 0; these `AGE`s are interesting in themselves. Indeed, further investigation of these values seems to indicate anomalies in the way the census data was collected.

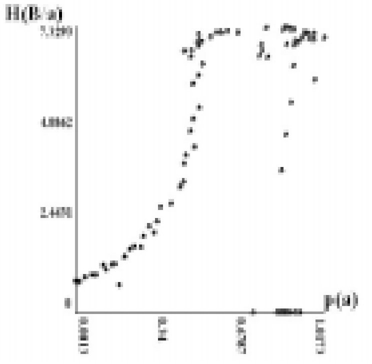

Figure 9: Comparing $p(a)$ to $\mathcal{H}_B(\sigma_{A=a}(r))$ for census data. In this example, A is `AGE` and B is `DEPART`.

5 Visual Comparisons Of Datasets

In previous sections we have demonstrated the use of entropy to visualize the information content of database relations. In this section we show how multiple, diverse datasets can be compared within the same display in order to understand the degree to which the datasets might be similar in terms of their structure.

We have used this particular technique to compare various benchmark datasets in order to evaluate their structure. Although benchmark datasets are used for a variety of applications, a primary use is the performance evaluation of new algorithms. For example, the Wisconsin benchmark [7] has been used to test various join algorithms. Within the machine learning community a large number of benchmark datasets are available.[2] Many of these datasets have been used for evaluating various data mining techniques.

Figure 10 (Left) shows $\mathcal{H}_A$ compared to log $|\text{adom}(A)|$ for the Wisconsin benchmark data. The Wisconsin data can be seen to have a nearly perfect uniform distribution within each attribute. When compared on the census data, seen in Figure 10 (Right), it is clear that this synthetically generated data demonstrates significant differences from real data, which has much more complexity to its structure.

As another example, Figure 11 shows a number of datasets from the machine learning repository displayed for comparison. We can see in this visualization that these datasets have different struc-

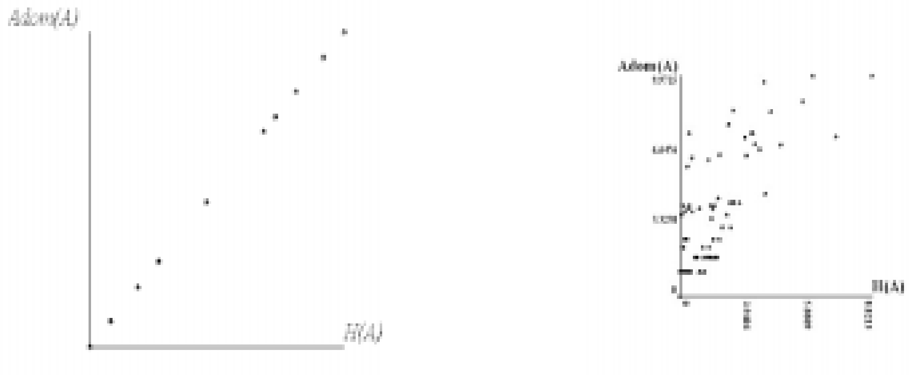

Figure 10: $\mathcal{H}_A$ compared to log $|\text{adom}(A)|$ for the Wisconsin benchmark data (Left). The same comparison from the census data (Right).

ture as well, although the sparseness of the data does have an effect. In addition, these datasets tend to have a large number of boolean valued attributes.

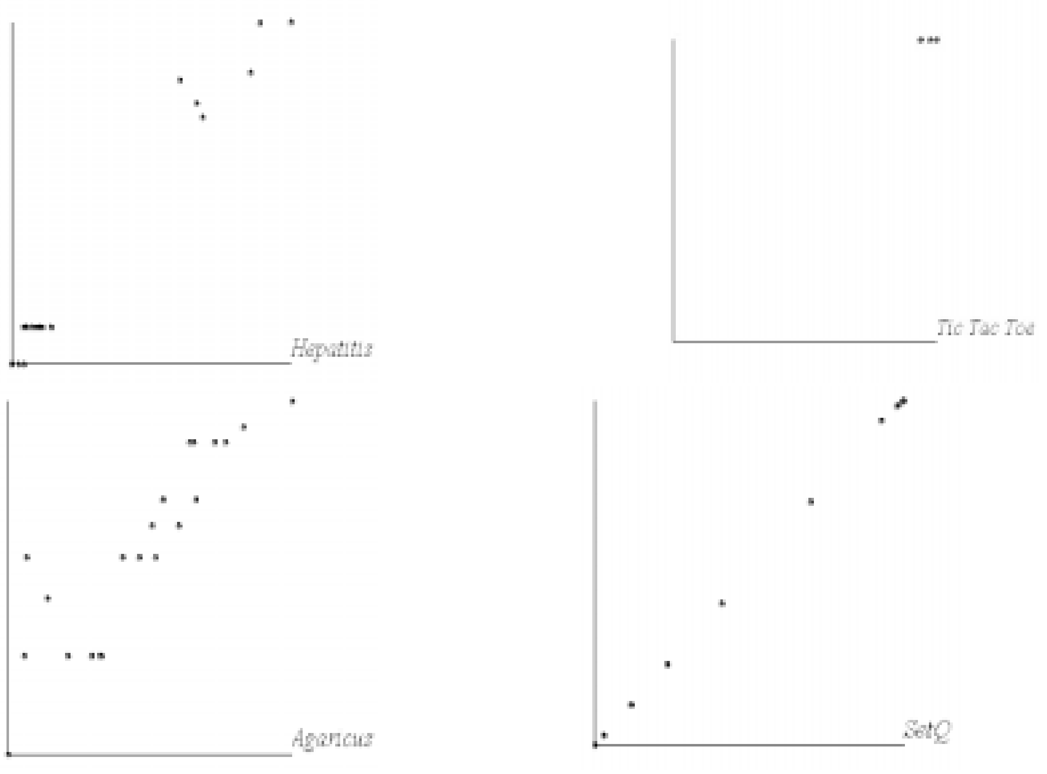

Figure 11: $\mathcal{H}_A$ compared to log $|\text{adom}(A)|$ for datasets taken from the machine learning repository. Clockwise from top left - Hepatitis, Tic Tac Toe, Agaricus, SetQ.

6 Conclusion

In this paper we have shown how entropy, a central concept in information theory, can be used for visualizing the structure of information within database relations. The technique simplifies the display of complex relationships, allowing for dependencies to be spotted. Our use of entropy is independent of the underlying datatypes, handling all in a consistent fashion. Furthermore, we have demonstrated the technique on a wide variety of data, some of which are quite large. The census dataset, for instance, contains 125 attributes and approximately 300,000 rows of data.

While this particular research is reported in terms of database visualization problems, the techniques we have employed are applicable to several areas. Within data mining we envision that these techniques can be used to assist an expert in exploring their particular problem space. In addition, database designers can use the visualization to assist in the construction of decompositions, either for OLTP systems, or for OLAP data warehouses.

References

[1] www.census.gov. On The Web.

[2] Blake, C., and Merz, C. UCI repository of machine learning databases, 1998.

[3] Card, S. K., Mackinlay, J. D., and Shneiderman, B., Eds. *Readings in Information Visualization:Using Vision to Think.* Morgan Kaufmann Publishers, Inc., 1999.

[4] Cover, T. M., and Thomas, J. A. *Elements of Information Theory.* John Wiley & Sons, New York, NY, USA, 1991.

[5] Dalkilic, M. M. *Foundations of Data Mining.* PhD thesis, Indiana University, Computer Science, 2000.

[6] Dalkilic, M. M., and Robertson, E. L. Information dependencies. In *Proceedings of the Nineteenth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, May 15-17, 2000, Dallas, Texas, USA* (2000), ACM, pp. 245–253.

[7] DeWitt, D. J. The wisconsin benchmark: Past, present, and future. In *The Benchmark Handbook for Database and Transaction Systems (2nd Edition)*, J. Gray, Ed. Morgan Kaufmann, 1993.

[8] Feiner, S. Virtual worlds for visualizing information. In *Advanced Visual Interfaces* (1992), pp. 3–11.

[9] Groth, D. P., and Robertson, E. L. Architectural support for database visualization. In *Proceedings of the Workshop on New Paradigms in Information Visualization and Manipulation* (1998).

[10] Inselberg, A., and Dimsdale, B. Parallel coordinates for visualizing multi-dimensional geometry. In *Proceedings of Computer Graphics International '87* (Tokyo, 1987), Springer-Verlag.

[11] Inselberg, A., and Dimsdale, B. Parallel coordinates: A tool for visualizing multi-dimensional geometry. In *Proceedings of IEEE Visualization '90* (Los Alamitos, CA, October 1990), IEEE Computer Society Press, pp. 361–375.

[12] Keim, D. A. Databases and visualization. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, Quebec, Canada, June 4-6, 1996* (1996), H. V. Jagadish and I. S. Mumick, Eds., ACM Press, p. 543.

[13] Keim, D. A. Pixel-oriented database visualizations. *SIGMOD Record 25*, 4 (1996), 35–39.

[14] Keim, D. A., Kriegel, H.-P., and Seidl, T. Supporting data mining of large databases by visual feedback queries. In *Proceedings of the Tenth International Conference on Data Engineering, February 14-18, 1994, Houston, Texas, USA* (1994), IEEE Computer Society, pp. 302–313.

[15] Kivinen, J., and Mannila, H. Approximate dependency inference from relations. In *Database Theory - ICDT'92, 4th International Conference, Berlin, Germany, October 14-16, 1992, Proceedings* (1992), J. Biskup and R. Hull, Eds., vol. 646 of *Lecture Notes in Computer Science*, Springer, pp. 86–98.

[16] LeBlanc, J., Ward, M. O., and Wittels, N. Exploring n-dimensional databases. In *Proceedings of IEEE Visualization '90* (Los Alamitos, CA, October 1990), IEEE Computer Society Press, pp. 230–237.