Adaptive Organization of Tabular Data for Display

Robert M. Losee[*]
Manning Hall, CB #3360
U. of North Carolina
Chapel Hill, NC, 27599-3360, U.S.A.
losee@ils.unc.edu
http://ils.unc.edu/losee
FAX: 919-962-8071

April 9, 2003

Abstract

Tabular representations of information can be organized so that the subject distance between adjacent columns is low, bringing related materials together. In cases where data is available on all topics, the subject distance between table columns and rows can be formally shown to be minimized. A variety of Gray codes may be used for ordering tabular rows and columns. Subject features in the Gray code may be ordered so that the coding system used is one that has a lower inter-column subject distance than with many other codes. Methods by which user preferences may be incorporated are described. The system optionally may display unrequested columns of data that are related to requested data.

Keywords: browsing, tables, Gray code, visualizing data, adaptive interfaces, relational databases, information retrieval

[*]The author wishes to thank Gary Marchionini, Lewis Church, and anonymous referees for helpful comments on an earlier draft of this manuscript.

1 Introduction

Given a query and a database containing structured data, how should the retrieved data be organized into rows and columns in a tabular display? Common sense suggests that when there are large numbers of columns in a table, those columns on similar topics should be grouped together. What might be the formal rationale for this arrangement? Below we present criteria and methods through which these questions may be addressed.

We develop an explicit process by which features in table columns may be arranged or weighted so as to increase the similarity between adjacent table columns. This arrangement can be made adaptive, incorporating user expressions of preferences. This model can be seen as consistent with some user heuristics, such as ordering tables alphabetically, from largest to smallest values, etc. We will also suggest that database and retrieval systems may retrieve and display columns of data that were not requested, when there is a close topical proximity to data that were requested.

The nature of these features has a large impact on the quality of the organization of the items. The best features may be chosen from the set of existing features so that there is very little subject overlap between the chosen features, as in the case of *facets* used in indexing systems (Foskett, 1996). The features instead may be artificial codes, such as one finds with methods that produce artificial, statistically independent features (Borko, 1985; Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990).

The spatial organization of information is improved by placing those informational entities with similar features near each other. The most common example of such an organizing principle is found in libraries, where books with similar features are placed near each other on the library's shelves. We refer to the enumeration system used to assign numbers (such as those placed on the spine of a book) and then to arrange these books as a *classification system*. Popular classification systems include the Dewey Decimal System and the Library of Congress Classification System. Similar arrangements of columns based on column features may produce more browsable tables.

One often finds the rows of a table are placed in an order based on non-topical considerations. For example, a table of populations of nations in 2000 might list nations (rows) in alphabetical order rather than in an order based on the similarity of the countries to each other. Similarly, a table of populations of France for each year from 1950 to the present might list the years (rows) in calendar or reverse-calendar order. We will discuss an economic argument for why rows of a table might best be placed in what first appears to be a non-topical order.

Ordering of data along one dimension of a table may be optimized given certain assumptions. While published tables are most frequently presented as 2 dimensional, we are treating the columns and rows as separate 1 dimensional vectors. Some of what we are considering here can be modeled in 2 dimensions, but treating the dimensions separately provides a clearer and simpler exposition of the basic issue of providing an organizing principle for tables that brings related material together.

This model of column organization will allow for efficient browsing of material. In fact, many tables are designed explicitly for browsing; others may be optimized for searching. Other methods may be more effective for data presentation if the user wishes to retrieve a single data element, such as might be retrieved by a DBMS given a query designed to retrieve a single record.

Users retrieve information consistent with one of two paradigms. The first, searching an information retrieval system, uses a query to represent the user's information needs. The user enters a query into a system and then the system responds, presenting relevant information to the user. The browsing paradigm, on the other hand, assumes the data is organized to support browsing, usually based on a notion of topical similarity. The user selects a starting point within the informational space and then moves (or jumps) to neighboring or distant information, based upon what was already found.

Users may retrieve information from a table of data by browsing through it, in a manner similar to how they browse library collections (Boyce, Douglass, Rabalais, Shiflett, & Wallace, 1990; Losee, 1993b, 2002; Morse, 1970; Rice, McCreadie, & Chang, 2001). Reflecting similar concerns, we may arrange the columns or rows on a displayed table so that each row or column is similar to the row or column adjacent to it. This will increase user searching and browsing efficiency. We address the issue of placing similar columns near each other, as well as columns whose separations are most costly. This arrangement has similarities to those proposed on qualitative grounds by Ranganathan for use in indexing systems, including chain indexing (Foskett, 1996).

We optimize data organization in a table given constraints and assumptions of the model. The development of a formal model for the placement of columns and rows in a table and the analysis of the assumptions are essential to a science of table development. While most user-centered research involves studying users directly, many aspects of user-centered questions concerning *optimization* best may be addressed formally, while others best may be addressed through empirical studies (Schoemaker, 1991). While it is clear that mathematical models may be elegant but ignore certain complexities (Kahneman, Slovic, & Tversky, 1982), all inferential work runs the risk of being inaccurate because all cases aren't considered and the inference is based on incomplete information.

2 Tables and the Display of Data

Tables display data in an organized manner (Borowick, 1996; Schriver, 1997), presenting information in rows and columns that allows for the easy understanding of the information. Compared to graphs, the other primary form of data presentation, tabular data is usually presented in a form that is more accurate than graphical displays but tabular data is usually weaker at showing relationships (Smith, Best, Stubbs, Archibald, & Roberson-Nay, 2002). In some cases, additional information, such as metadata, may be available to users of tabular data to assist in the understanding of the source or features of the data (Marchionini, Hert, Shneiderman, & Liddy, 2001).

Printed or displayed tables need to restrict the amount of information present. Borowick notes that "when tables have more than five columns and five rows.... [the author should] separate the data or facts into subsets to encourage reader attention" (Borowick, 1996, p. 113). Tables must be organized so as to be relatively easy to understand. Our goal below is to provide an organizing principle that will allow us to automatically group columns to increase the degree to which we might "encourage reader attention."

The organization of tabular data is often consistent with a particular organizing principle. The simplest principle is randomness, with data beng displayed in any order. In this case, users searching for information might best use a linear or random search technique. A great deal of the organization used for tabular data is consistent with human notions of what is a suitable taxonomy. While such a taxonomy may be appealing, some users will find this taxonomy more natural than others do. Each taxonomy represents a particular cultural viewpoint; using a particular arbitrary taxonomy makes browsing more difficult for those who have viewpoints inconsistent with these knowledge structures. The techniques below will provide a set of objective criteria for organizing data in tables, given whatever features the user or system chooses, as well as a set of adaptive techniques so that a table may be tailored for a particular user or group of users.

3 The Gray Code

The binary Gray code provides an ordering for each possible item's representation such that there is only 1 character difference between the representation for an item and the representation for the next item (Hamming, 1986). The *Hamming distance* between two individual binary representations for items is the number of features by which they differ (Losee, 1990). For example, the Hamming distance between 10101 and 11111 is 2 because the two representations differ in 2 posi-

Decimal	*Binary*	*Gray Code*
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

Table 1: Decimal, traditional binary, and Gray code numbers.

tions, the second and the fourth. The Hamming distance is always 1 between the Gray code representations of two numbers when one number is the successor or predecessor of the other.

The path defined by the Gray code through the topical space may be viewed geometrically as moving over the surface of a unit cube in *n*-space connecting each vertex where each connection in the path is of length 1, i.e., on an edge of the cube. A unit cube is a cube where the allowable locations on each axis allow for the two values, 0 and 1. Every binary number of n bits serves to represent a vertex on this *n*-cube, and conversely, each vertex on the n dimensional unit cube represents one of the *n*-bit binary numbers.

The most commonly considered form of the Gray code is the *reflected Gray code* (Conway, Sloane, & Wilks, 1989; Flores, 1956; Gilbert, 1958). Consider Table 1 which shows the numbers from 0 to 7 in decimal, traditional binary, and reflected Gray codes. The reflected Gray code is developed according to a pattern that is best seen visually. Note that if one drew a line across Table 1 between the decimal 3 and decimal 4, one would see that the right two bits (columns) for the Gray code seem to be reflected around the line between 3 and 4. The top half of the Gray code numbers (decimal 0 to 3) have a 0 as the leftmost bit and the bottom half (decimal 4 to 7) having a 1 as the leftmost bit (Flores, 1956). A reflection can similarly be seen if one drew a line between decimal 1 and decimal 2, with decimal 1 being similar (on the rightmost bit) to decimal 2 and the decimal 0 being similar (on the rightmost bit) to the decimal 3.

A Gray code is transformationally equivalent to another code if one code can be changed to another by shuffling the bit positions. Taking an example from the decimal number system, one might arbitrarily swap the ten's and the one's columns and still count and do math, although it would certainly be awkward at first and would require that arithmetic algorithms be modified. One may modify the reflected Gray code by moving positions around and still retain many of the features of the original reflected Gray code.

One can similarly make a Gray code that is non-reflective (Losee, 2002), although non-reflected codes are far less tractable. The non-reflected code can be produced by noting that at some positions in the reflection, one can remove a group of consecutive numbers so that the edges of the remaining "hole" have a Hamming distance from each other of 1 (and thus a Gray code exists locally "over" where the section was removed, and there is a place where the removed set of numbers can be inserted so that the requirements of a Hamming distance of 1 between adjacent code words is met.

The differences in orderings provided by the Gray code and "true binary" may be seen by considering a square, where the Gray code values moving around the four corners of the square would be $(0, 0)$, $(0, 1)$, $(1, 1)$, and $(1, 0)$. Note that each corner differs from its adjacent corners by only one position or bit. If one were to provide an ordering consistent with true binary ordering, one would obtain (with decimal values in square brackets, e.g. [and]): $(0, 0)[0], (0, 1)[1], (1, 0)[2], (1, 1)[3]$. Clearly, the difference between the decimal 1 and the decimal 2 as represented by true binary is 2, as is the difference between the end value and the first value (3 and 0) when one wraps around from the end to the beginning. Differences of 2 never occur in the Gray code, where the distance is always 1; the Gray code brings items closer together than does true binary. Packing items together and the relationships with information theory of such packings is an interesting problem (Thompson, 1983).

While each digit in a Dewey Decimal Number and each bit in either a true binary number or a Gray code binary number can be used to represent a position in a hierarchy, and all these codes are consistent with a hierarchical indexing or feature system, the Gray code places items closer to neighbors with which they have similarities (assuming each "corner" is actually instantiated as a feature combination in a table or a document.) Using the Gray code produces smaller gaps at transition points than does true binary, where the number after 0111 is 1000, a number that has literally nothing in common with its predecessor. When viewed in an indexing context, Ranganathan referred to these gaps as *jerks* and realized that they should be minimized (Foskett, 1996). The Gray code uniquely meets this requirement.

When tables are synthesized from retrieved data, such as the result of SQL queries, the columns may be arranged so that they are in Gray code order, providing a degree of similarity between adjacent columns. Each table column contains a subset of the features from the universe of features for the dataset being used.

These features represent what the column is about, and when the features are brought together within each column to represent the column, the columns may be arranged by using the Gray code ordering for columns based on the list of presence and absence of features (*feature vectors*). The features may be natural language terms or arbitrary symbol strings representing concepts. Metadata may be seen as an example of the latter (Greenberg, 2002). Clearly, placing similar columns near each other organizes columns in a cognitively smoother form than would be experienced with a random ordering of columns.

4 Searching and Browsing in a Topical Space

The process of searching or browsing through a set of entities, such as table columns, may be viewed as consistent with a geometric model in which a path leads through a concept space and where each entity or column represents a point in the space.

A topical space provides a location for every possible combination of the topics which it represents (Newby, 2001). Different spaces will allow for different sets of combinations; essentially, a particular space is consistent with one set of possible combinations of the topics. Most humans accept that kittens are biologically male or female, while few native speakers of English would attribute such inherent gender features to a teapot, television, or a rug. Different combinations of variables are allowed in different spaces; some cultures view spirits as having gender, while others do not. Each topical space is consistent with a particular world view, and a user's conceptual world can be represented by a particular space. For our purposes, we assume that concepts n the topical space are independent, or that concepts can be developed which are statistically independent e.g., through Latent Semantic Indexing (Deerwester et al., 1990), and all (or a very large number of) concepts are included in our feature space.

A path exists between two points in a space when there is a set of points linking (or more formally a continuous function exists between) the two points. A path may be a straight line, or it may curve or be zigzagged. The nature of paths in a space determines how distances may be measured. If one looked at an aerial view of a large city, for example, the distance between two intersections may be measured "as the crow flies," that is, as though there were no intervening objects between the intersection. This is the Euclidean distance between two points. When one measures the distance that a taxi would travel between the two intersections, however, one is limited to traveling by street. Measuring distance this way uses the "taxi-cab" or "city block" metric.

A Hamiltonian path is a path through the space that passes through each point exactly once. A Hamiltonian circuit is a Hamiltonian path that begins and ends at the same point. The Gray code that will be used in our work provides such a Hamiltonian, and the table columns can be understood as being consistent with a Hamiltonian circuit.

In a topical space, *features* are variables representing a topic, feature, or metatag (Greenberg, 2002). Given one variable per topic, we may speak of an *n*-space as consisting of all the possible topic combinations in a particular world or world view. A simple way of determining the feature vector associated with a table column might be to include the feature associated with each term, assuming a one-to-one relationship between terms and feature variables. Features may also be assigned values based on human subject analysis (Foskett, 1996) or through the statistical extraction of features (Deerwester et al., 1990). Features which are relatively independent are widely referred to as *facets* in the indexing literature. The author believes that using relatively independent features is very important to the success of feature-based automatic organizing systems.

The classification systems suggested below provides a path through the topical space that is consistent with a Gray code. Note that the Gray code *does not* address indexing or how the feature vectors are derived: the Gray code as used here provides an ordering or enumerating process that can be used to prescribe an arrangement for these feature vectors. When we use such a code to describe a path through a dataset, it is unnecessary to compute distances between all of the columns. If distance measures are constantly used when trying to organize table columns, one must compute $n(n-1)/2$ distances to compute a similarity matrix when there are n possible nodes or table columns. If there are m binary features and there are nodes representing all the possible feature combinations, then there are 2^m nodes, and the number of distance measures that must be computed becomes $2^m(2^m-1)/2 = (4^m - 2^m)/2$.

The process of *searching* through the concept space may be described as *following a path through the topical space*. One can define information retrieval as the determination of a path through the space as well as a starting point, often the first item to be examined. Interestingly, the output from an information retrieval system is provided in an order designed to place documents in roughly decreasing order of probability of relevance. This probabilistic order provides a path through space: the reader might note that as one progresses through this article, considerations are incorporated into the Gray code model that are consistent with the concerns of decision-theoretic and economic models, providing a Gray code based model that is optimal in terms of placing similar document near each other while at the same time probabilistically or economically weighting features and possibly adapting weights as searching progresses.

Browsing similarly begins at a point and then moves through a path. It may be very directed toward answering a specific question, or it may be more exploratory, where the act of browsing helps one to develop the sense of information need (Marchionini, 1995). Browsing usually takes place on a path that is determined for a group. Libraries, for example, order documents on shelves by using a single classification system, such as the Dewey Decimal system, that is designed for all users of the library. If we have an adaptive classification system (Losee, 1997) that can be automatically customized for each user, the path may be developed in such a way that the individual has to travel the shortest distance possible to find those topics of interest. This differs from non-adaptive systems to which the user must adapt. Bates (1986), for example, describes how users adapt to the ordering in reference texts by quickly inferring the organizing principle and then searching based on the properties of the inferred arrangement. The proposed adaptive Gray code system brings together related material in a manner that non-adaptive models fail to do.

When counting using a code such as the traditional binary counting system as well as the binary Gray code, the rightmost bits change or cycle more frequently than do the bits to the left. For a Gray code based classification, arranging features so that the most probable bits (features) are on the left and the least probable bits are on the right results in the lowest expected dissimilarity between adjacent entities than would be the case if the bit order were reversed. This arrangement of features is similar to the feature ordering in *chain indexing* as proposed by Ranganathan (Foskett, 1996). This form of indexing, used by some search engines, including Yahoo!, orders features from the most general (the highest probability) to the most specific (the lowest probability). For example, one such hierarchy used by Yahoo! is:

> Computers and Internet > Internet > World Wide Web > Searching the Web > Indices to Web Documents > Dewey Decimal Classification.

This feature ordering will be considered more extensively below.

We should note that there is a growing literature studying *carousel systems* (Litvak & Adan, 2001) that will likely contribute to the browsing literature in the near future. Assume that inventory is stored on a large wheel or carousel, and that someone filling orders spins the carousel (or moves around it), selecting items to fill the order. What will be the person's expected travel time (or the wheel's spin time)? This is very similar to browsing through randomly organized collections, determining the expected browsing distance. We believe that this carousel literature may be found to provide tools useful for describing organization and its

Table 2: Data concerning column headers and features of a five column table. Here *Y* represent *yes* or the feature being present and *N* represents *no* or the feature's absence. This order of column headers (the rows in this table) is what is obtained with standard decimal or "true" binary order.

Column Name	*NameQ*	*OfficeQ*	*AddressQ*
Home Phone Number	N	N	N
Home Address	N	N	Y
Office Phone Number	N	Y	N
Office Address	N	Y	Y
Person's Name	Y	N	N

relationship to browsing time. In return, studies of browsing, and more specifically those studies addressing the optimal organization of information using a Gray code, provide a solution to questions about how to organize products placed on such a carousel.

5 A Sample Ordering of Table Columns

As an example of the use of this technique, let us assume that we have five variables and column headers, shown in Table 2. We use three binary features to capture the different aspects of our column names. Note that there are some combinations of features that don't represent column headers, for example, there is nothing that is both a *NameQ*, an *OfficeQ*, and an *AddressQ*.

If a table's columns are arranged using traditional numbering systems (true binary or decimal), the column order would be as (the rows) in Table 2. If, however, we wish to use the Gray code, with the features used for the Gray code in the order in Table 2 e.g., *NameQ, OfficeQ,* and then *AddressQ*, we would find the column order to be

- (0 0 0) = (N N N) [Home Phone Number],
- (0 0 1) = (N N Y) [Home Address],
- (0 1 1) = (N Y Y) [Office Address],

- (0 1 0) = (N Y N) [Office Phone Number],
- (1 1 0) = (Y Y N) [],
- (1 1 1) = (Y Y Y) [],
- (1 0 1) = (Y N Y) [],
- (1 0 0) = (Y N N) [Name].

Here the first number in parentheses represents whether the *NameQ* feature is present or not, the second number whether the *OfficeQ* feature is present or not, and the third number whether the *AddressQ* feature is present or not. Note that the Gray code ordering does not determine the indexing, which in this case is simply whether a feature is present or absent, a feature that is objectively determined. The Gray code, instead provides an *ordering* of the feature vectors, e.g., the ordering of the eight lines above based on their numeric value. This ordering has the effect of placing both addresses (office and home) together. If we consider *wrap-around* among the first four lines, it is also clear that phone numbers are placed next to each other.

If, on the other hand, we placed the third (rightmost) column in Table 2 between the first and the second, so that the feature order is *NameQ, AddressQ*, and then *OfficeQ*:

- (0 0 0) [Home Phone Number],
- (0 0 1) [Office Phone Number],
- (0 1 1) [Office Address],
- (0 1 0) [Home Address],
- (1 1 0) [],
- (1 1 1) [],
- (1 0 1) [],
- (1 0 0) [Name].

In this feature arrangement, columns with the feature *OfficeQ* are brought together. Notice that if there were not the third (leftmost) feature, columns with the two *HomeQ* features would "wrap-around" and be adjacent, and if there were a fourth or a fifth feature, each column with a *HomeQ* feature would be adjacent to another column with a *HomeQ* feature (assuming that all possible columns exist.) We believe that if all columns don't exist, this arrangement is still the best in most cases.

Clearly, the nature of feature and column arrangement determines the level of adjacency or subject distance between columns. In one instance in our example, we were able to have columns addressing work or home next to each other, while with a different ordering of features, the columns addressing phones or addresses were placed adjacent to each other.

6 Table Browsing Performance

The performance of a user browsing a table may be measured or estimated as the distance along the path in topical space. While there are other measures of browsing task performance, such as clock time, eye movements, and so forth, using the distance in concept space to measure browsing performance can serve as a surrogate for the cognitive distance and complexity experienced by the user. The length of the path the user travels in the topical space thus serves as a performance indicator. The path can be adjusted to minimize the distance the user travels in the topical space; this will allow us to provide tables with column organization adapted to users or groups of users.

Measures may be descriptive, or we may try to predict their values. A descriptive measure of the browsing distance may calculate the distance based upon the sum of the observed distances between individual entities through which the user browses. Psychologically, a Gray Code based path through the set of columns or rows in a table may approximate what happens at the level of eye movement, as the eye scans across a table, or it may approximate the cognitive process underlying such actions (Salvucci & Anderson, 1998). It is clear that the eye doesn't follow an exact linear path when it reads or scans, but the linear path is a reasonable approximation of what happens over relatively large distances or within brief scans of chunks of text.

Predictive measures estimate the distance, based upon the algorithm used to produce the links between the entities and knowledge of the locations of the entities. As a simple example of this distinction, we can measure how far an auto travels in an hour using an odometer, or we can predict how far it will go by multiplying the one hour by the average rate of speed, yielding the expected distance covered in an hour.

7 Information Theoretic Ordering of Column Features

The ordering of table columns may be adapted to a particular set of circumstances, and may be changed as the circumstances change. One ordering principle based on probabilistic considerations is information theory. While the use of Shannon's information theory (Shannon & Weaver, 1949) to model phenomena outside of electronic communication may be stretching the basic model some (Ritchie, 1991), the information content of features or columns in tables may be approximated by this model. Related to this is the excellent work by Lee that addresses information-theoretic models of databases and tables (Lee, 1987).

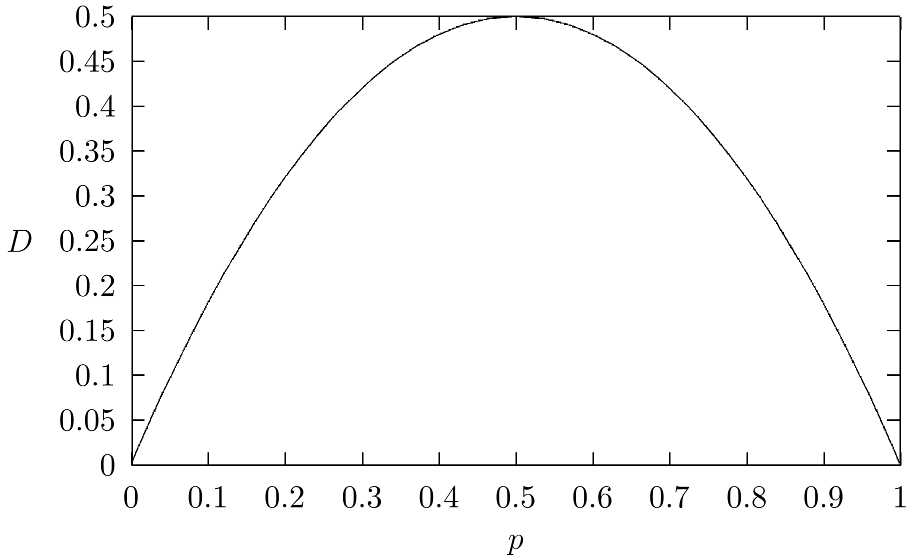
When using the Gray code as a classifying and ordering principle, being able to measure the subject distance between entities in the code and thus columns in tables allows one to evaluate different codes (Losee, 1992, 1993a). In predictive situations, a probabilistic description of the distance will allow us to consider formally when one system for organizing table columns will be superior to another. To simplify our argument, we assume below that all features are binary (although this applies equally well to non-binary features), and thus each table column or row *is* or *is not* about each one of the features in the feature set. We will allow for the weighting of features, which provides added flexibility in feature ordering.

We begin examining the dissimilarity between columns by considering the case where all features are considered to have the same weight or information. Assume that distance or difference between two identical features is 0 while the difference between two different features is 1. One can then compute the dissimilarity between two columns i and j for a single feature k as the sum of the probabilities that the feature has different values given random column ordering. The chance that there is a difference between two features is the chance that the first is a 1 multipled by the chance the second is a $0,$ with the same occurring for the inverse values. The value of the dissimilarity for feature k is thus

$$D(c_{i,k}, c_{j,k}) = p_k(1 - p_k) + (1 - p_k)p_k, \qquad (1)$$

where p_k is the probability a characteristic feature $c_{i,k}$ will occur in a column or data class i. Such probabilities may be estimated using a variety of methods, ranging from methods of moments to Bayesian methods to more elaborate methods from the machine learning community (Fisher, 1925; Heckerman, Geiger, & Chickering, 1995; Cherkassky & Mulier, 1998). The dissimilarity D reaches its maximum when $p_k = .5,$ as can be seen in Figure 1. The dissimilarity between adjacent table columns is computed here as sum of the dissimilarity between all

Figure 1: Dissimilarity (D) between a one feature document with different feature frequency.



the features in the feature set:

$$D(c_i, c_j) = \sum_k p_k(1 - p_k) + (1 - p_k)p_k. \tag{2}$$

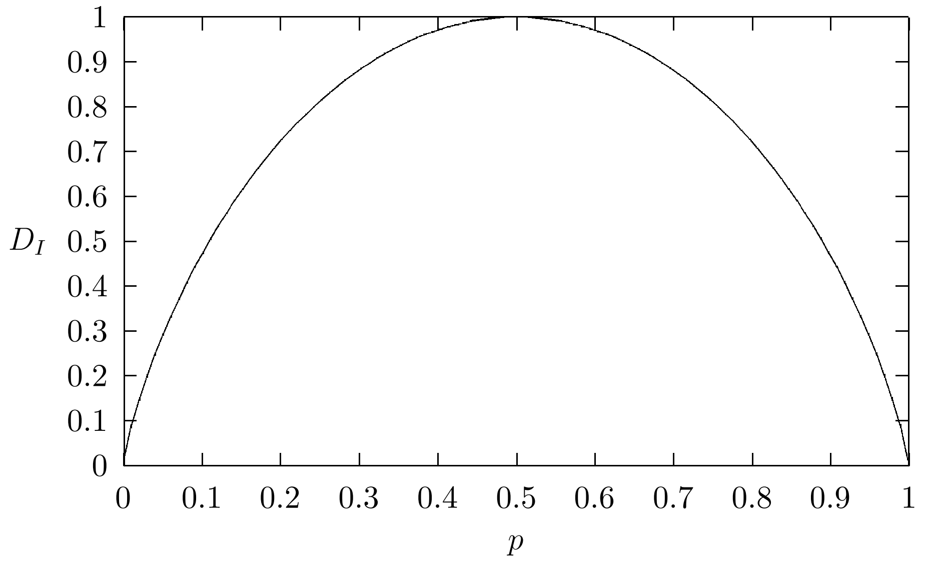
The difference in information content of different columns may be computed using Shannon's notions of information, where the self information of an event x is $-\log(\Pr(x))$. The expected information dissimilarity, D_I, may be computed as the average information associated with a feature (the entropy of the feature) multipled by an indicator variable showing the presence or absence of a difference between features:

$$D_I(c_i, c_j) = -\sum_k |c_{i,k} - c_{j,k}| \left(p_k \log p_k + (1 - p_k)\log(1 - p_k)\right), \tag{3}$$

where $c_{i,k}$ is the value for feature k in table column c_i, and p_k is the probability of feature k occurring. The trend in values for D_I is shown in Figure 2.

Using these information-theoretic measures of topical distance between table columns, features may be ordered within each column's feature vector so as to help minimize the expected distance between columns (Losee, 1992). Features may be placed in any order; however, the expected dissimilarity D_I, computed over the set of columns, is minimized when *those features with the lowest ex-*

Figure 2: Difference in information content (D_I) using Shannon's measure of information.



pected dissimilarity are placed furthest to the right in the feature vector. The features on the right change most frequently, as can be seen informally by counting from 0 to 100 in the decimal system and noting how often changes occur in the one's column and how often changes occur in the ten's column. Placing the features with the lowest expected dissimilarity on the right results in a lower average dissimilarity over the set of vectors as the features with the greatest expected similarity (as changes occur) are changed most frequently, rather than the features whose changes would cause an overall decreased expected similarity to occur. Assuming that all the features occur with $p < .5,$ then $D_I(c_{i,k}, c_{j,k})$ will be lower for neighboring columns i and j when the rarer terms (with the lower dissimilarity values) are on the right side of the vector and it is most likely that the difference between the neighboring columns will be due to a smaller difference on the right rather than a larger difference on the left.

Consider the use of features in a library classification system. In most libraries, the code used on the spine labels for books that represents a topic like *ocean* will be to the left of the symbols used for *lighthouse*. The topic *ocean* is a broad one and somewhat general. It surely has a higher probability of occurrence in most libraries than does the much less common feature *lighthouse*. The probabilistic and information-theoretic arguments provide a justification for this ordering. It similarly provides a justification for placing *ocean* as a feature on the left hand side of the numbering system used in representing features, as opposed to the placement of *lighthouse* which should be on the right hand side of the vector.

Using our earlier example and the table columns with features described in Table 2, we can see that if the dissimilarity D (derived from probabilities p and $1-p$) associated with the *OfficeQ* feature occurring is greater than the dissimilarity associated with the *AddressQ* feature, then the *AddressQ* feature being placed on the right will result in a lower average dissimilarity between columns than if the columns were reversed.

8 Adaptive Organizing: Economic Principles

Economic considerations may be used to weight features to improve feature ordering in a vector beyond the ordering consistent with information-theoretic concerns. While information-theoretic considerations may be used to organize features to produce a better coding system that is consistent with the Gray code, economic weightings may be expected to further improve performance.

A negative utility or loss, denoted by L, is associated with features having different values in columns that are located less than an arbitrarily chosen distance apart. For a given feature, the economic loss of having a 0 for that feature value in the first table column and a 1 for that feature in an adjacent, second fragment, is denoted as $L_{0,1}$, with a similar loss, $L_{1,0}$, for the first table column having a value of 1 and the second table column having a value of 0.

The loss associated with the dissimilarity of feature k, having a different value in one randomly selected table column than in a second randomly selected adjacent table column, may be computed as

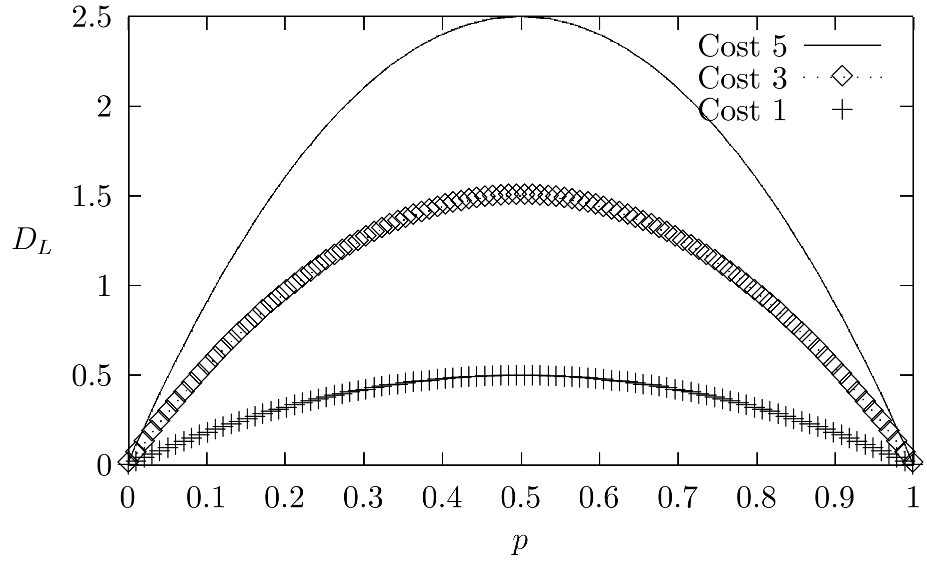
$$\begin{aligned} D_L &= L_{1,0}p_k(1-p_k) + L_{0,1}(1-p_k)p_k \\ &= (L_{1,0}+L_{0,1})p_k(1-p_k), \end{aligned} \tag{4}$$

the product of the expected dissimilarity and the loss associated with that expected dissimilarity, yielding the expected loss. As before, p_k is the probability that feature k has the value 1 in the database of table columns. We assume that $L_{0,0}$ and $L_{1,1}$ equal 0. For notational simplicity, we denote the sum of loss $L_{1,0}+L_{0,1}$ as L_k for feature k.

The loss associated with placing table columns i and j immediately adjacent to each other, each with n binary features, denoted as $c_{i,1}, c_{i,2}, \ldots, c_{i,n}$ for table column i, with similar notation for table column j, is

$$D_L(c_i, c_j) = \sum_{k=1}^{n} |c_{i,k} - c_{j,k}| \, L_k, \tag{5}$$

Figure 3: Expected loss associated with a one feature document being adjacent to a document with a different feature frequency, with the cost difference being 1, 3, or 5.



assuming that features may be treated independently. This is the sum of losses for features which differ in value between table column i and j. The effect of different costs and probabilities is shown in Figure 3.

Using this, features used in the Gray code are ordered so that the expected loss (D_L) associated with an ordering of table columns is minimized. We do this by placing the lowest D_L values furthest to the right in each column's vector, with the highest values to the left.

The parameters of the model may be learned so that the organizing system adapts to an individual's preferences and needs. The feedback may be elicited directly from the end user, with costs associated with viewing columns with specific features being used to determine the average cost associated with each feature.

The system may also gather information non-reactively. For example, the system may keep track of data elements requested over a period of time, and then treat the features associated with these data elements as having greater benefit and positive utility than other features.

As an application of this economic technique, consider our earlier example of the features *ocean* and *lighthouse*. In most libraries, the feature *ocean* is considered to be of greater economic value to general users, providing a justification for its placement to the left of the feature *lighthouse* in the classification numbers placed on book spines. This greater economic value implies a greater loss when this feature changes. A user who focuses on lighthouses would undoubtedly place this as an economically more important feature than the general feature *ocean*. The ordering for this individual would probably be reversed from that found best for the general public.

Using the address example with data in Table 2, we can similarly see that by assigning varying costs to specific features, such as *OfficeQ* or *AddressQ*, we can modify the ordering of features and thus economically adapt our tables to a user's personal cost structures. It is this sort of adaptability of the classification and organizing systems that allows us to produce tables that are designed to minimize the subject distance (and associated cognitive dissonance) between columns.

9 Discussion

There are several ways these models might be modified or that their application might be expanded.

When a column is displayed, one might consider the option of displaying unrequested but highly related columns of data when there is sufficient room for its display and when there will not be significant degradation of the viewing experience. Columns might be added only when the degradation is relatively small, such as might be found when the number of columns does not exceed a fixed small size (e.g., a maximum of 6 columns) and the table will not become significantly more complex for the end user.

Another issue is which column of data should be displayed on the far left side of the table in cultures that read from left to right, with a similar concern about what is displayed on the right in cultures that scan from right to left. The Gray code places the columns into an order that can be visualized as a ring. If the ring is cut at one place and laid out flat, we can imagine each point in the order being a column header.

Where the cut should take place is an open question. One method is to find the largest subject gap between adjacent columns. By "cutting" the ring at that point and stretching the cut ring out from left to right on the table, we have placed more similar columns near each other, minimizing the total subject distance between one side of the table and the other side.

A second method is to take a given query and place the column nearest the query vector (or exactly representing it) at the center of the table. One then builds up the sets of columns consistent with the Gray code moving out from this centerpoint. Using the "ring" metaphor, we might view this as placing the point of the ring representing the query on the center of the table, cutting the ring on the diametrically oppositive point, and then flattening out the ring. Clearly we may need to limit the number of columns included and may need to effectively truncate the ring on either the right or the left or both.

Columns on the table may be placed so that a key column is placed on the left margin (which readers often see first if they read and visually scan from left to right.) One could also question whether the most interesting material should be placed in the center of the table This may be determined based on empirical studies of how people use tables.

While in a modelling sense there are strong similarities between rows and columns, and the Gray code can be used to order either rows or columns, or both rows and columns, social conventions sometimes suggest differing uses for rows than for columns. For example, people often place text row labels in alphabetical order, such as in a table of countries and their populations. Similarly, a table of the richest 100 executives might place the executives in decreasing order of their net worth. These orderings are economically rational because the cost of jumps between adjacent rows is minimal. This can be consistent with our economic model of feature ordering. The effectiveness of these orderings allows us to emphasize that the Gray code based ordering is largely topical. Additionally, we see that economics can seemingly override the non-economic Gray code in cases where the cost of following the Gray code is high and the cost of following another (conventional) method is very low. In most circumstances, there is a higher economic and cognitive cost in the sequential order *Afghanistan–Liechtenstein–Albania* than in the order *Afghanistan–Albania–Liechtenstein*.

One may also treat features as non-binary within a Gray code. These features may then provide their own intrinsic ordering within the ordering provided by the Gray code. For example, a 3 valued Gray code (e.g. 0, 1, 2) might have a set of entities with feature sets and ordering as shown (each feature set is in one set of parentheses): $(0,0,0)$, $(0,0,1)$, $(0,0,2)$, $(0,1,2)$, $(0,1,1)$, $(0,1,0)$, $(0,2,0)$, $(0,2,1)$, $(0,2,2)$, $(1,2,2)$, and so forth. Note that 2 is adjacent to 0 because when counting $0,1,2$, the number after 2 is again 0. Such non-binary features may be represented alphabetically and may be consistent with alphabetical ordering (e.g. 26-valued for a 26 letter alphabet) or ordering by decimal value, for example.

10 Conclusion

When data is requested from a store of structured information, data may be displayed in tabular form. We have suggested how this data may be arranged to best present the information to users, and, more specifically, how can it be made adaptive so as to be optimized for a particular user group or a specific information need. The justification for this arrangement provides mathematical support for the use of facets and chain indexing, as well as an understanding about why they are effective.

We have provided a method by which tables can be organized so that the columns are placed in a manner that decreases the average subject distance between the columns. Using the Gray code, we are able to automatically order all the feature combinations so that the columns have this desirable organization. A variety of Gray codes may be used for this application. Features in the Gray code may be ordered so that the coding system used is one that has a lower inter-column subject distance than with many other codes. Means were described by which user preferences may be incorporated. In addition, criteria are developed by which the system displays columns of data that are not requested but that are related to requested data.

References

Bates, M. J. (1986). What is a reference book: A theoretical and empirical analysis. *RQ*, *26*, 35–57.

Borko, H. (1985). Research in computer based classification systems. In *Theory of Subject Analysis: A Sourcebook*, pp. 287–305. Libraries Unlimited, Littleton, Colo.

Borowick, J. N. (1996). *Technical Communication and Its Applications*. Prentice-Hall, Englewood Cliffs, NJ.

Boyce, B. R., Douglass, J. S., Rabalais, J., Shiflett, L., & Wallace, D. P. (1990). Measurement of subject scatter in the Superintendent of Documents classification. *Government Publications Review*, *17*, 333–339.

Cherkassky, V., & Mulier, F. (1998). *Learning from Data*. Wiley InterScience, New York.

Conway, J. H., Sloane, N. J. A., & Wilks, A. R. (1989). Gray codes for reflection groups. *Graphs and Combinatorics*, *5*, 315–325.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, *41*(6), 391–407.

Fisher, R. A. (1925). Theory of statistical estimation. *Proceedings of the Cambridge Philosophical Society*, *22*(5), 700–725.

Flores, I. (1956). Reflected number systems. *IRE Transactions on Electronic Computers*, *EC-5*(2), 79–82.

Foskett, A. C. (1996). *The Subject Approach to Information* (Fifth edition). Library Association Pub., London.

Gilbert, E. N. (1958). Gray codes and paths on the n-cube. *Bell System Technical Journal*, *37*, 815–826.

Greenberg, J. (2002). Metadata and the world wide web. In Kent, A. (Ed.), *Encyclopedia of Library and Information Science*, Vol. 72. Marcel Dekker, New York.

Hamming, R. (1986). *Coding and Information Theory* (Second edition). Prentice-Hall, Englewood Cliffs, N.J.

Heckerman, D., Geiger, D., & Chickering, D. M. (1995). Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, *20*(3), 197–243.

Kahneman, D., Slovic, P., & Tversky, A. (1982). *Judgment under Uncertainty: Heuristics and Biases*. Cambridge University Press, Cambridge, England.

Lee, T. T. (1987). An information theoretic analysis of relational databases, parts I and II. *IEEE Transactions on Software Engineering*, *SE-13*(10), 1049–1072.

Litvak, N., & Adan, I. (2001). The travel time in carousel systems under the nearest item heuristic. *Journal of Applied Probability*, *38*, 45–54.

Losee, R. M. (1990). *The Science of Information: Measurement and Applications*. Academic Press, San Diego.

Losee, R. M. (1992). A Gray code based ordering for documents on shelves: Classification for browsing and retrieval. *Journal of the American Society for Information Science*, *43*(4), 312–322.

Losee, R. M. (1993a). The relative shelf location of circulated books: A study of classification, users, and browsing. *Library Resources & Technical Services*, *37*(2), 197–209.

Losee, R. M. (1993b). Seven fundamental questions for the science of library classification. *Knowledge Organization*, *20*(2), 65–70.

Losee, R. M. (1997). Browsing document collections: Automatically organizing digital libraries and hypermedia using the Gray code. *Information Processing and Management*, *33*(2), 175–192.

Losee, R. M. (2002). Optimal user-centered knowledge organization and classification systems: Using non-reflected Gray codes. *Journal of Digital Information*, *2*(3). http://jodi.ecs.soton.ac.uk/Articles/v02/i03/Losee.

Marchionini, G. (1995). *Information Seeking in Electronic Environments*. Cambridge University Press, New York.

Marchionini, G., Hert, C., Shneiderman, B., & Liddy, E. (2001). E-tables: Non-specialist use and understanding of statistical data. In *National Conference on Digital Government Research: Proceedings of dg.02001*, pp. 114–119.

Morse, P. M. (1970). *On Browsing: the Use of Search Theory in the Search for Information*. MIT Press, Cambridge, Mass.

Newby, G. B. (2001). Cognitive space and information space. *Journal of the American Society for Information Science and Technology*, *52*(12), 1026–1048.

Rice, R. E., McCreadie, M., & Chang, S.-J. L. (2001). *Accessing and Browsing Information and Communication*. MIT Press, Cambridge, Mass.

Ritchie, D. L. (1991). *Information*. Sage, Newbury Park, CA.

Salvucci, D. D., & Anderson, J. R. (1998). Tracing eye movement protocols with cognitive process models. In *Proceedings of the twelfth Annual Conference of the Cognitive Science Society*, pp. 923–928 Hillsdale, NJ. Lawrence Erlbaum Associates.

Schoemaker, P. (1991). The quest for optimality. *Behavioral and Brain Sciences*, *14*, 205–245.

Schriver, K. A. (1997). *Dynamics in Document Design*. John Wiley & Sons, New York, NY.

Shannon, C. E., & Weaver, W. (1949). *The Mathematical Theory of Communication*. University of Illinois Press, Urbana, Ill.

Smith, L. D., Best, L. A., Stubbs, D. A., Archibald, A. B., & Roberson-Nay, R. (2002). Constructing knowledge: The role of graphs and tables in hard and soft psychology. *American Psychologist*, *57*(10), 749–761.

Thompson, T. M. (1983). *From Error-Correcting Codes Through Sphere Packings to Simple Groups*. Mathematical Association of America, Washington, D.C.