Seamless Integration of Diverse Data Types into Exploratory Visualization Systems

Anilkumar Patro, Matthew O. Ward and Elke A. Rundensteiner
Computer Science Department
Worcester Polytechnic Institute
Worcester, MA 01609
{anil,matt,rundenst}@cs.wpi.edu [*]

Abstract

Visual data exploration involves presenting data in some graphical form and allowing the human to interactively gain insight into the data. Visual data exploration techniques have proven to be very useful for exploratory data analysis, especially for mining large databases. However, the lack of explicit content semantics within the framework of the visualization tool makes interpretation of the visualizations difficult at times. The aim of our research is the development of a comprehensive framework for integrating semantics of data into an existing visualization tool in a flexible yet minimally intrusive manner. The first stage of the framework involves a transformation process that enables the mapping between a wide range of data types to a form the visualization system can process. This stage exploits the strengths of XML for handling semantics extraction, flexible data modelling of domains and managing the data mapping rules to conform the data to the requirements of the visualization tool. The second stage of the framework enables the visualization tool to access and present the semantic information separate from the data display. We demonstrate and validate this framework by illustrating the extension of XmdvTool, a multivariate data visualization system, to handle varied data types such as categorical fields and unstructured text files. We believe the framework is sufficiently powerful to incorporate other forms of data, as well as to be used to extend other visualization tools.

Keywords: Visual data exploration, Data mapping, Data semantics, Metadata, Nominal data visualization, Text visualization

1 Introduction

Visual data exploration is concerned with exploring data and information in such a way as to promote a deeper level of understanding and foster new insight into the data, relying on the humans' powerful ability to perform visual information processing [25]. In a simple sense, visualization is essentially a mapping process, taking data attributes and mapping them to representations that can be perceived. Humans can often easily detect patterns, trends and outliers in the underlying data when presented with visual depictions of the data, which may be more difficult to identify with automatic techniques.

The data is the key component of the visualization and it plays a large role in determining the effectiveness of the visualization tool. The usefulness of a data visualization tool might be limited by:

- The amount of data that can be handled by the tool.
- The preprocessing of the data that is required to convert the data from its original format to a form the visualization tool can process.
- The different types of data that can be handled by the tool.
- The limited presentation of data semantics within the visualization.

Even though visualization systems have tried to tackle the problem of displaying large datasets [2] (for example, pixel-oriented displays [18] and hierarchical parallel coordinates [9]) effectively, the other issues have been given less attention. This paper tries to address the last two issues in an existing visualization system:

Most of the current software systems for graphically presenting data are not built to handle a wide range of data types and hence they tend to be special purpose. For example, Parallel Coordinates [13] for numeric variables, Mosaic Displays [7] for nominal data, ThemeView and Galaxies [36] for text corpus are few exemplars of many such techniques. The reason of the tools being special purpose could be that the data that the visualization system uses to generate graphics is primarily numeric. This is because numeric data can be easily mapped to graphical attributes, which are usually themselves ordinal in nature (e.g. size, position, intensity). However, a great deal of data can be of other types such as nominal / categorical data. The problem with visualizing this in traditional ways is that the ordinal nature of the graphical attribute may introduce errors in the interpretation of the visualization.

Data sets have structure, both in terms of the means of representation (syntax) and the types of interrelationships within a given record as well as between records (semantics). Semantics of the data may be as important as the data for particular domains. Most visualization systems can pictorially represent the structure, but fail to capture the semantics of the data. For example, the most common way to represent nominal variables is by way of including legends for the variables. Although this can be effective in some situations, it cannot be generalized to other data types and might not scale to the amount of data in the dataset. Inclusion of semantics might be important in situations where the user who provides the data (data provider) is different from the user who analyzes the data (data analyst). This scenario can make the process all the more difficult because the analyst may not know the true meaning of the data. And since there is no means of passing semantics to the visualization tool, the knowledge within the data is completely lost in the visual mapping process.

In this paper, we propose a framework for including heterogeneous data types and embedding data semantics into the structure of an existing visualization tool while attempting to minimize the modification of the tool itself. This would surely increase the utility value of the tool for use in various application domains. To

[*]This work is supported under NSF grant IIS-0119276.

realize this, one basically needs an embedding of the data semantics within the framework. This can be achieved by (a) capturing the description of the data, (b) storing the mapping between data attributes to numerical values and (c) then driving the display of the particular visualization with this knowledge.

To capture the description of the data, we associate the metadata with the original data. This metadata can provide information such as the format of the individual fields within the data records, which helps in its interpretation. It may also contain the base reference point from which some of the data fields are measured, the units used in the measurements, the symbol or number used to indicate a missing value, and the resolution at which measurements were acquired. Each of these may be important in selecting appropriate preprocessing operations and setting their parameters.

Storing the mapping information with the metadata itself can make the integration of different data types seamless within the system. This allows the framework to work with an existing visualization tool.The flexibility of having different mapping types will make the process more powerful to extend the tool to many application domains. One such domain is the area of text visualization, an emerging area, where a mapping from structured or unstructured text files to the visualization tool-compatible format would enable the same tool to handle this kind of data.

The data semantics can be displayed within the existing visualization by accommodating the metadata into the display. This would require minimal recoding of the input and display modules to integrate the proposed framework.

The remainder of the paper presents the framework as follows. Section 2 presents work by others that is related to ours. Section 3 presents the proposed framework. Section 4 provides details about the data mapping process that is one of the main component in the transformation process. Details about the other components of the framework are discussed in Section 5. The different encodings for the mappings are discussed in Section 6. We validate our ideas using two case studies in Section 7. Section 8 contains our conclusions and the future work.

2 Related Works

The related work to our framework can divided into two categories:

- Mention some of the existing visualization tools and how they deal with different data types and the purpose of this is to show that little attention has been given to the issue of handling data management and data semantics.

- Mention some of the related research for visualizations that can support abstract data collections, i.e., data having varied data types.

2.1 Earlier Approaches

Visualization systems such as OpenDX [33], AVS [1], and XmdvTool [34], by default, make use of single precision floating-point values for the visualization of multivariate data sets. These are some of the systems which would surely benefit by the addition of data semantics within the tools.

Database oriented visualization systems such as TreeViz [14], the VisDB system [17], the DeVise tool [22], Polaris [30] and SGI's MineSet [26] do handle numeric and nominal types. But these systems just assign some order to the nominal variables and treat them as ordinal types. Statistical Data Analysis packages including SAS [5], S Plus [20], XGobi [31] and Data Desk [32] do allow one to specify and manipulate the mappings from nominal to numeric, but the mappings are completely embedded within the tool and not extensible to allow for other data types.

2.2 Related Research

Research on providing general visualizations for abstract data have been underway for some time. The TRIP [16] system is based on the process of translation from application's data representation into pictorial representations. In this scheme, the data is represented by abstract objects and relations, which are mapped to graphical objects and graphical relations. Proximity Visualization [3] is another system that visualizes the similarity between elements of abstract data collections. We have extended the ideas put forth by these systems and applied it for the creation of the proposed framework to facilitate the inclusion of various data types in general purpose visualization systems.

3 Proposed Approach

There are four basic stages in the pipeline for a typical visualization tool (the visual mapping process), together with a number of feedback loops [35]. They consist of:

- The collection and storage of data itself.

- The preprocessing designed to transform the data into a form that can be visualized.

- The graphics algorithms that produce an image on the screen.

- The human perceptual and cognitive system (the perceiver).

Figure 1 gives an overview of the framework proposed in this paper. Our framework introduces another mapping process, parallel to the visual mapping process, to include the semantics of the data within this pipeline of the visualization tool. The basic idea is that many visualization tools have a *semantically independent data display* and a *semantically dependent data display*. In current visualization systems, the semantically independent display is driven by the visual mapping process and the semantically dependent display is normally ad hoc (indicate the numeric values instead of the actual values). Our framework hinges on this mapping process, that we term *data mapping*, that tries to capture the semantics of the data and then drive the semantically dependent display through this process.

The data mapping process outlined here involves two stages:

- Mapping Stage

  This stage of the framework involves a transformation process that enables the mapping between a wide range of data types to a form the visualization system can process. This transformation process involves the creation, manipulation and storage of the mappings. The subcomponents of this process are:

  – Data Mapping Generator

    This helps in the creation of mappings through the use of automatic / semi-automatic tools and processing helper components. These are discussed in detail in Section 5.1 and Section 5.2.

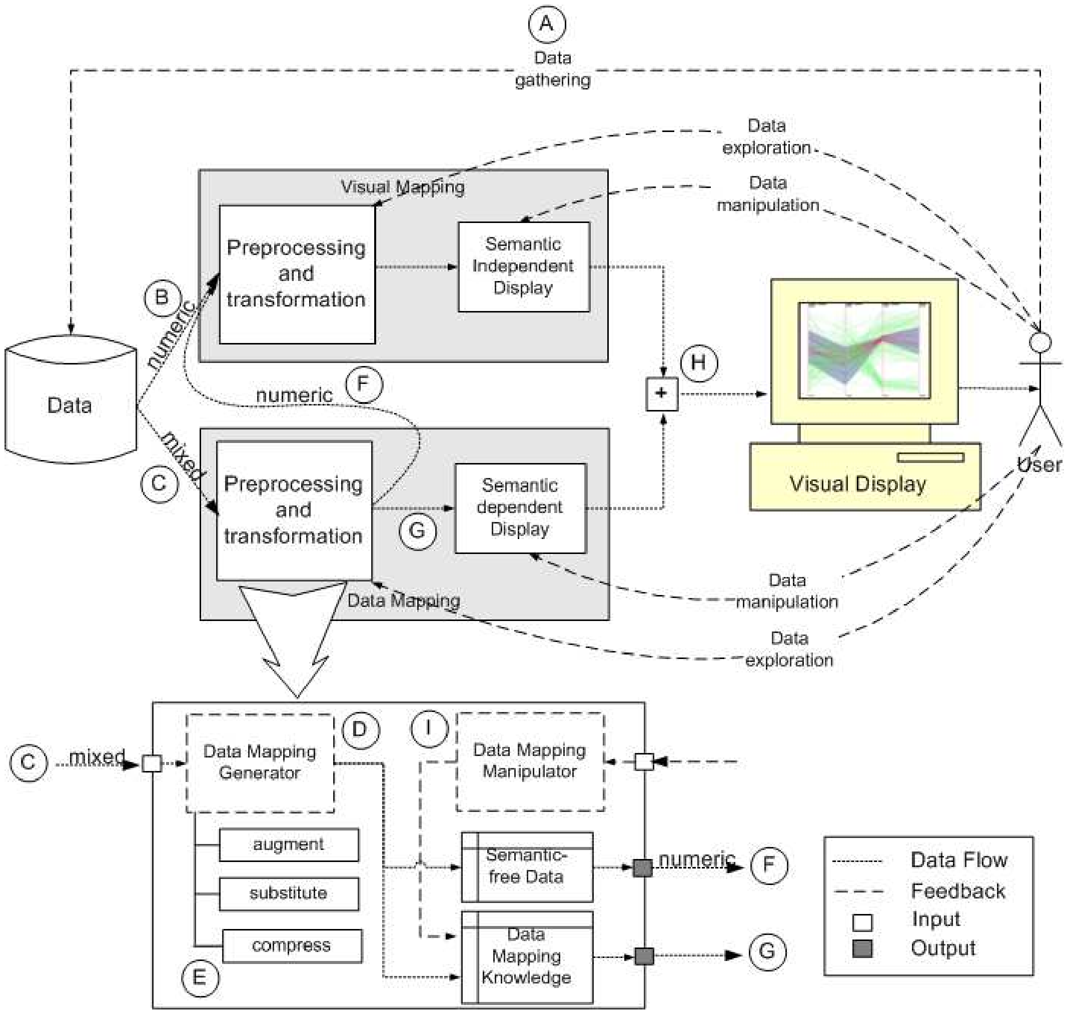


Figure 1: Block Diagram of the proposed framework.

– Data Mapping Manipulator

  The data exploration process will also involve the manipulation of these mappings to gain better insight into the data. These subcomponents help in the manipulation of the mappings within the framework of the visualization system. Section 5.3 presents the Data Mapping Manipulators in detail.

– Data Mapping Encodings

  Efficient and extensible storage of the mappings is an issue within the process. Section 6 provides details on the various encodings that are possible for storing the mappings.

- Display Stage

  Semantic information associated with the data is captured by the first stage of the process. But this would be of no use if this information were not to be displayed on the screen. The integration of the mapping knowledge with the display of the tool would enable the presentation of this semantic information, either separate from or integrated with the actual display. This should involve minimal recoding of the display of the visualization tool so as to incorporate the different types of data that can be provided by the user.

The flow of data within the entire framework now can be conceptualized (alphabets in parentheses refer to the corresponding circles in Figure 1) as follows:

- Data is acquired by the user through some source (A).
- If the data is numeric in nature, then it can be directly fed to the visual mapping process (B).
- Otherwise the data is directed to the data mapping process (C) to convert it to a form that the visual mapping process can utilize.
  – A Data Mapping Generator (D) creates the set of mappings and the numeric representation of the data. A set of processing helper components (E) assist in further refining the mappings. The numeric data (semantic-free data) (F) created by the Generator is directed to the visual mapping process and the mapping information is stored within the data mapping process in the data mapping knowledge repository (G).

– The visual mapping process and the data mapping process can both drive the entire display (H) to provide an effective visualization of the data. If the user is not satisfied with the visualization, he may want to explore the possibilities of manipulating the the numeric data or the mappings. The mappings can be modified through the Data Mapping Manipulator (I).

The idea of data mapping can be best understood with the help of a simple example. Consider we want to represent the grades of students for a particular class. In presenting the data to the visualization system in numeric form, the data would have to be converted to values such as 4.0 for ’A’, 3.0 for ’B’, and so on. This converted data can be easily visualized by the system, but this mapping cannot be represented within the tool. Our data mapping process can store these mappings and as part of the framework drive the display to show the mapped values. Section 4.3 elaborates on this small example presented here.

4 Data Mapping

Techniques for the visualization of data are divided into two distinct groups [27]:

- **Non-transformational techniques:** These techniques simply map the data directly onto graphical primitives. The location of the graphical primitives on the screen is determined by some other numeric data. For example, we can use colors or the shapes of the points to represent nominal values. However, this may not work for many numeric displays such as ScatterPlot Matrices and Parallel Coordinates if the colors or shapes already have pre-defined purposes.

- **Transformational techniques:** These techniques transform the data into numeric values. For example, the most natural transformation we might apply to a set of nominal values is a frequency-based analysis. Given n nominal data points, each with m possible discrete values, we create a new table that counts the frequency of each value. In this example, we’ve transformed our nominal data into numeric values, and now tt’s a straightforward job to map these numeric values to the appropriate kinds of graphical entities (e.g. columns of a histogram).

The Data Mapping process that we outline here is a transformational technique that is minimally intrusive within the framework of an existing Visualization system. The need for such a mapping process was pointed out in the introduction. This block is the main system component responsible for storing the knowledge of the mappings.

4.1 Definition of Data Mapping

The main idea of the data mapping process is to include many diverse data types into an existing visualization system through association of metadata with the data to drive the visualization. This is a critical step of the knowledge discovery process within a semantics-based visualization system. It involves transforming data into a suitable format for a particular analysis tool. A good data transformation would facilitate better accuracy of the results of exploratory data analysis. Thus, Data Mapping can be defined *as a way to specify the binding of semantics of the data to a representation of the data that the visualization system can process*.

A rich and flexible data mapping process is the central component of the visualization framework. This process will serve to organize the information found in the complex, heterogeneous data sets obtained from different sources. The need is for a process that describes the semantic content of the data rather than just reducing it to a collection of numerical data structures.

Since the framework that we are proposing needs to work with an existing visualization tool, the data mapping process has to work in tandem with the existing environment of the tool. The semantics and the mappings must serve as add-ons to the already existing numerical to graphical attribute mapping within the tool.

4.2 Data Mapping Knowledge

The data mapping knowledge or the metadata carries two categories of meta knowledge:

1. **Domain Description:** This is the description which characterizes the data types within the data. The inclusion of a large variety of data types within the framework will allow the visualization tool to process data gathered from many sources. Data types that would immediately benefit the visualization system if incorporated within the tool are:

   - Simple data types: These data types are common in most data sets. Simple data types include types such as real and integer for interval data, strings for categorical data and the uniform resource identifiers (URI) for document collections. Other simple data types such as date and time can also be included for time-varying data. Our framework also allows the specification of domain constraints for the simple data types to validate the data that is being fed to the visualization system. A simple example of this is an enumeration of strings, where the value of a particular data entry can only be one of the values in the enumeration. This allows for maintaining the integrity of the data set and in some cases better efficiency for the storage of these simple data types.
   - Complex data types: These data types are essentially a collection of simple data types. One example of such a data type would be an address composed of several simple data types such as string for the street address, state and country information and integer for the zip code.

2. **Mapping Description:** This is the description that characterizes the mapping of the original data to assigned numerical values. The semantically dependent display uses these mappings to show the semantics. Mappings that could be specified are:

   - Enumerated Mappings: This simply lists out all the mapping pairs between the data and its numerical representation within the metadata. This represents a finite number of enumerable domains possible for a dataset.
   - Function-driven Mappings: This presents a function within the data mapping knowledge to map a particular data value to its numerical representation on demand. This is a much more flexible representation of the mapping because this does not depend on the number of data values within the dataset.

4.3 Example

Consider the example of a purchase transaction dataset from Amazon.com that could have many variables such as `product code` (numerical values), `product type` (book, CD, etc), and `addresses`. This example incorporates the use of simple data types such as integer for the `code`, strings or enumerations for the `product type` and complex data types such as `addresses`. A Data Mapping Generator can then preprocess this data set and generate numeric mappings for the product type and addresses. The mappings, for example, can specify that the `product type` - book - will have a value of 0.3456 and that the `product type` - CD - may have a value of 0.6782. The metadata associated with the data will specify the data type of each dimension and the mappings associated with the values. The numeric data will contain these values to specify book or CD for the product type dimension. The visualization tool is capable of presenting the actual data using the numeric representation of the raw data and identifying the actual values using the stored mappings. The display section of the tool can be modified to make use of these mappings in order to present the visualization in a much more user-friendly way, as will be shown below in the case studies (Section 7). Figure 2 gives a better view of this example. Note that the product code does not have any mappings associated with it since it's already in numeric form.

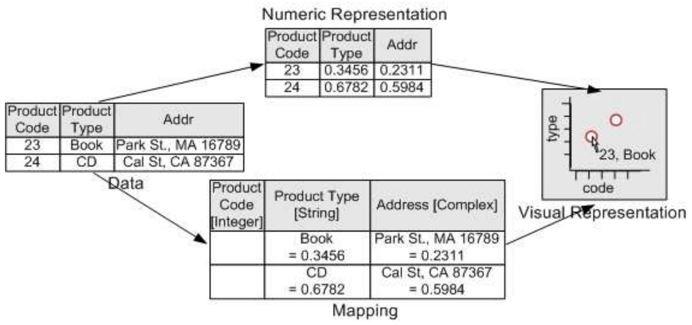


Figure 2: Data Mapping Example.

5 Data Mapping Helpers

The data mapping process is a simple process and as such can be handled manually. But, it would become cumbersome to generate these mappings for a large amount of data. Also, the feedback from the visualization might be used to modify the mappings from the original in an interactive manner. Thus, we propose a system of mapping helpers that assist the user to create and manipulate the mappings in this process. These may be automatic or semi-automatic tools that execute within the system or as stand-alone processes.

5.1 Data Mapping Generator

Converting from one data representation to another is both time-consuming and labor-intensive. The user is already involved getting the raw data into a form the visualization tool can interpret. Adding another conversion process for generating the mappings would not be acceptable for large amounts of data. Generators are a set of analytic tools that aid in the creation of the metadata needed for the data mapping process, either automatically or semi-automatically (i.e., with user input). The data mapping generator is not only responsible for generating the numeric form of the data for the visualization tool, but also for creating the mappings from the particular data type to the numeric values.

For example, to display nominal variables using numeric displays, one way is to map the nominal values to numbers, i.e. assign order and spacing to the nominal values. This could be driven by a data mapping generator that deals with nominal values. One such example of a data mapping generator could be an implementation of [23].

5.2 Data Processing Helpers

Processing Helpers are analytic tools that facilitate the importation of heterogeneous data types into a visualization system. To date we have identified three classes of Helpers:

1. **Substitution:** This type of helper does not change the number of records or dimensions, but substitutes values of one type with values that are interpretable by the visualization tool. Examples of this type of helper is converting nominal variables into numeric types, applying scaling operators, and transforming data from one coordinate system into another [23, 29].

2. **Augmentation:** This helper adds to the dimensionality of a data set by performing some analytic process that generates derived values or classifications. Examples include extracting the first three principal components and adding the values in as new dimensions or performing clustering [15] on the data and labelling each data record according to the cluster to which it belongs.

3. **Compression:** this type of helper can reduce the number of records, the number of dimensions, or both. For example, we can implement helpers that eliminate outliers, remove redundant dimensions, or replace each record with its equivalent position in a lower dimension space using a process such as multidimensional scaling [15].

It is critical that each Helper augment the metadata associated with a data set to properly reflect to the user the nature of the mappings. Without this additional information it would be relatively easy for the user to misinterpret what is seen in the visualization.

5.3 Data Mapping Manipulator

After the metadata has been derived from the mapping process, it can be interpreted from the visualization much more efficiently by the analyst. The analyst may further want to modify the mappings in order to derive a possible improvement. The manual modification of the metadata might require expert knowledge about the data and metadata. The framework includes a set of manipulator helper components that free the user from manually modifying the associations. The manipulators could include expert system tools that generate suggestions on how to enhance the mappings.

In order to create suggestions for the improvement of the mappings, a base for the decisions leading to these suggestions has to be provided by gathering statistical data about the mapping process. For example, given a nominal to numeric mapping, a manipulator might be used to reorder and re-quantify the nominal mappings and statistics can be displayed to show the ”goodness” factor [23] of the new ordering.

A persistent mapping scheme would also be beneficial in the interchange of semantics for the same set of data. This would mean that the user can try various semantic associations with the dataset in order to get an effective visualization. For example, the user could manipulate the data mappings through the Data Mapping Manipulator and store this new set of mappings in a separate file. The user can then associate the newly created metadata with the data and can then study the effect of the manipulations.

6 Data Mapping Encodings

There is a need for the mapping knowledge to be persistent within the process because we do not want to modify the visualization tool in order to maintain these mappings. The mappings can be encoded in a variety of ways, including

- ASCII Encoding

  A simple and fast solution would be to define a simple ascii file format with attribute-value pairs for the purpose of storing the mappings and to specify the type association. Such a method works for small to medium size volumes of data, and can be extremely efficient when coded correctly. However, it does require newcomers to learn the syntax of the data encoding format and is also difficult to validate. Basically, the problem of ascii encoding is the tight integration of the format with the code, making it hard to comprehend, extend and validate.

- Binary Encoding

  Binary encoding of the mappings is possible and one could also associate a binary value for each type possible within the system. Binary encoding will, in most cases, result in compact files but has the same problems as of ASCII encodings along with the problem of being platform dependent.

- XML Encoding

  XML is the eXtensible Markup Language, a World Wide Web Consortium (W3C) standard for the representation of information as structured documents, and is an emerging standard for digital information markup. It allows information (data, metadata, documentation, resources of any kind) to be expressed in a structured, flexible, and easily parsable manner. XML allows for contents-based tagging of any information resource, and consequently it allows for powerful, focused, and efficient contents-based search and retrieval of information. For these reasons, XML Encoding provides a more elegant solution to the above specified problems in other encoding systems. The advantages are:

  - XML is self-describing, i.e., its tags are meaningful allowing us to semantically markup the document and making it user-friendly.
  - XML is extensible in many ways: extensible tag library, extensible document structure, and extensible document elements.
  - XML is very suitable for creation and manipulation by a programming language due to the easy availability of parsers.
  - XML can be easily validated.

Thus, we chose the Data Mapping encodings to be in XML format due to the above mentioned advantages.

7 Case Studies

The development of the framework presented in this article was driven by our goal of adding support for embedding semantics within XmdvTool. Although we show the ideas can be applied to a single visualization package, we believe they are general enough to be applied to other visualization systems.

XmdvTool is a public-domain software package designed for the interactive visual exploration of multivariate data [34]. The tool provides four methods for displaying both flat (non-hierarchical) form data and hierarchically clustered data, namely scatterplots, star glyphs, parallel coordinates, and dimensional stacking. XmdvTool also supports a variety of interaction modes and tools, including brushing in screen, data, and structure spaces, zooming, panning, and distortion techniques, and the masking and reordering of dimensions. More details can be found at http://davis.wpi.edu/ xmdv/.

7.1 Nominal Data Analysis in XmdvTool

There are several approaches to visualizing nominal variables. One can use displays that are specifically designed for nominal variables: sieve diagrams [8], mosaic displays [8, 11], maps from Correspondence Analysis [10], fourfold displays [8], treemaps [19], dimensional stacking [21] and CatTrees [19]. Unfortunately, several of these approaches are either special-purpose, not readily available in common data analysis software [8], or cannot handle high cardinality nominal variables well. So, the goal was to include the capability to analyze categorical data within XmdvTool.

While to date, XmdvTool has addressed visualization of numeric data, the inclusion of our framework enables the same tool to visualize categorical data. Unlike numeric data, categorical values do not have an order. So, this is problematic for typical visualization techniques, since categorical values need to be mapped to the numeric values to specify the graphical attributes. Since the semantics of the data has changed, it would require special coding to provide the special displays required for categorical data within XmdvTool.

```
<okcmeta>
  <dimensions value="10" />
  <datapoints value="205" />

  <dimension name="make" type="word">
    <value scale="-0.62750">honda</value>
    <value scale="-0.53840">mitsubishi</value>
    <value scale="-0.42860">plymouth</value>
      .
      .
      .
  </dimension>

  <dimension name="fueltype" type="word">
    <value scale="-0.14135">gas</value>
    <value scale="1.30745">diesel</value>
  </dimension>
    .
    .
    .
</okcmeta>
```

Figure 4: XML Snippet for the Automobile DataSet

For the following discussion, we are using the Automobile Data Set, which has the following nominal variables - make, fuel type, aspiration, number of doors, body type, wheels, engine location, engine type, number of cylinders and fuel system. A Data Mapping Generator for nominal data is used to generate the ordering

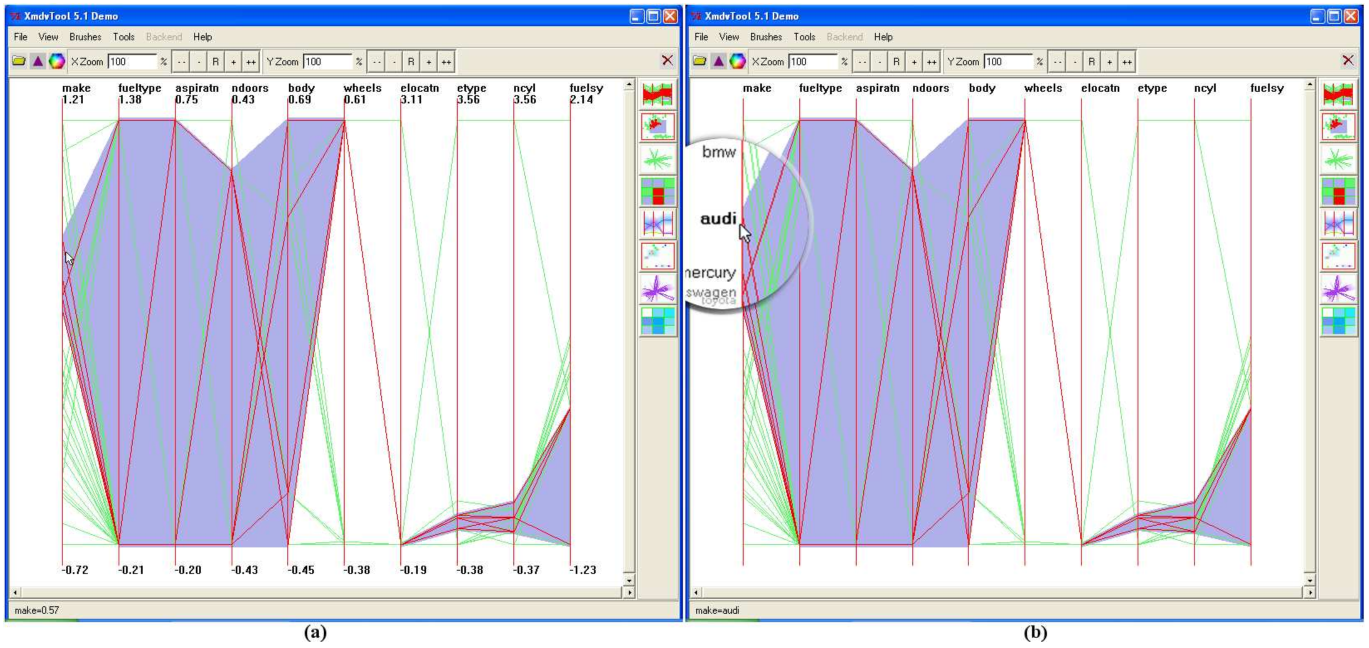


Figure 3: Nominal Data within XmdvTool. (a) Without annotation (b) With annotation (The labels have been highlighted by zooming into the region of the mouse cursor).

and spacing attributes through the use of a data-driven, multivariate, scalable and distance-preserving method [29]. This generates the numerical data required by XmdvTool as well as the data mapping knowledge required by the framework.

Figure 3 (a) shows how the Automobile Data Set would be represented if the data mapping process is not used. The difference is clearly visible in Figure 3 (b) where the semantically dependent part of the display has been recoded. The display now makes use of the data mapping knowledge and presents the visualization in a more user-friendly way. The statusbar and the annotated labels indicate the nominal values instead of the numerical values when the mouse hovers over a particular data entry. Figure 4 shows a snippet of the XML mapping produced by the Data Mapping Generator and stored by the system. This gives an example of how the semantics can be attached for the purpose of nominal data analysis within XmdvTool.

7.2 Text Visualization in XmdvTool

Text Visualization is primarily concerned with the presentation of a corpus of unstructured textual information so as to aid in the process of finding interesting or useful patterns in the document collection. Establishing a set of meaningful concepts for a text document collection allows one to look at hierarchical ordering of the concepts, and then mine for relationships in between documents and between concepts.

There are many text visualization tools available. Bead [4] presents similarity relationships between documents by 3-dimensional particle clouds. Using a similar visualization approach, Starlight [28] aims to visualize the content of multimedia databases. Closely related to these approaches is Galaxies [36] which yields a simple 2D scatter plot. VxInsight [6] visualizes the document distribution density by means of a mountain terrain metaphor. These approaches are focused on the display of similarity between individual documents. The scaling techniques used try to optimize the distances between documents with respect to the given proximity measure. However, directly using a 2D or 3D space results in a relatively high information loss and can only very coarsely represent the original documents similarity.

Our approach is to use a multivariate data visualization tool for the purpose of unstructured text mining. The hope is that the visualization of the document collection using a large number of dimensions would help gain better insight into the collection.

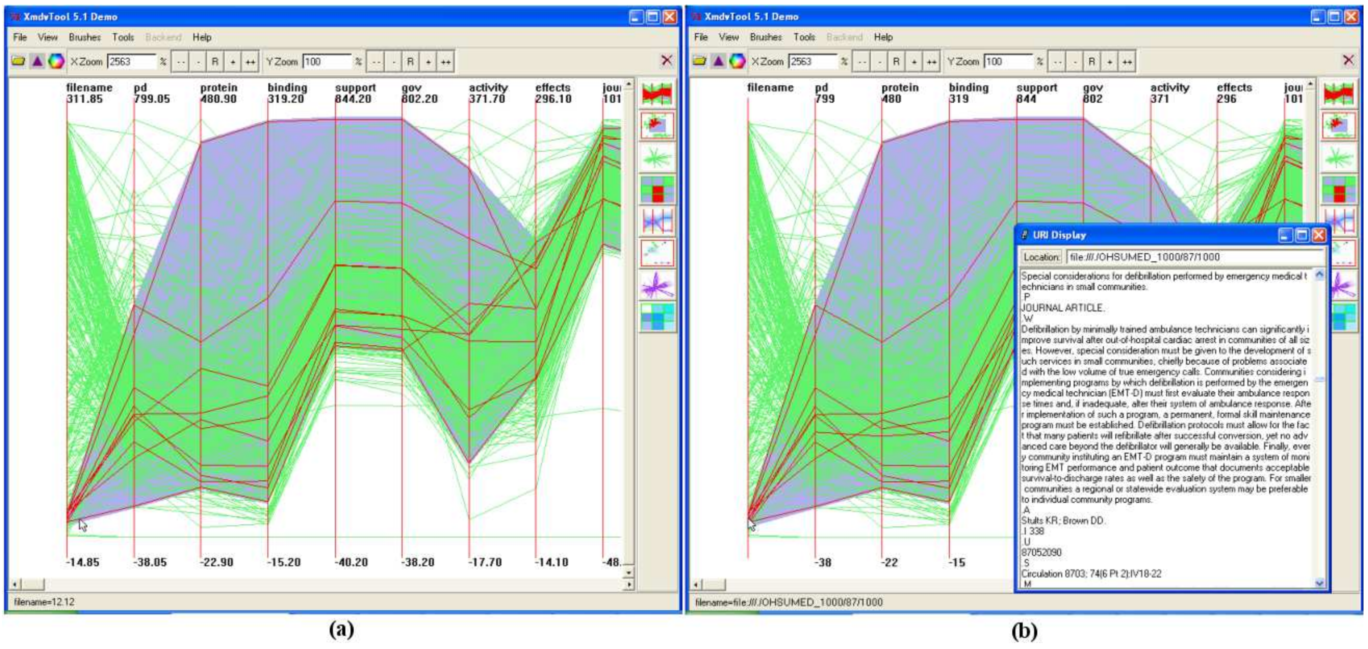
```
<okcmeta>
  <dimensions value="216" />
  <datapoints value="298" />

  <dimension name="filename" type="uri">
    <value>
      file:///./OHSUMED_1000/87/1000
    </value>
    <value>
      file:///./OHSUMED_1000/87/1001
    </value>
      .
      .
      .
  </dimension>

  <dimension name="pd" type="integer" />
  <dimension name="protein" type="integer" />
  <dimension name="binding" type="integer" />
  .
  .
  .
</okcmeta>
```

Figure 6: XML Snippet for the OHSU Medical Abstracts DataSet

The Data Mapping Generator for the text visualization system is responsible for text extraction, pruning and dimension reduction. [12] An available public domain tool, rainbow [24], was em-



Location: file:///OHSUMED_1000/87/1000

```
Special considerations for defibrillation performed by emergency medical t
echnicians in small communities.
.P
JOURNAL ARTICLE.
.W
Defibrillation by minimally trained ambulance technicians can significantly i
mprove survival after out-of-hospital cardiac arrest in communities of all siz
es. However, special consideration must be given to the development of s
uch services in small communities, chiefly because of problems associate
d with the low volume of true emergency calls. Communities considering i
mplementing programs by which defibrillation is performed by the emergen
cy medical technician (EMT-D) must first evaluate their ambulance respon
se times and, if inadequate, alter their system of ambulance response. Afte
r implementation of such a program, a permanent, formal skill maintenance
program must be established. Defibrillation protocols must allow for the fac
t that many patients will refibrillate after successful conversion, yet no adv
anced care beyond the defibrillator will generally be available. Finally, ever
y community instituting an EMT-D program must maintain a system of moni
toring EMT performance and patient outcome that documents acceptable
survival-to-discharge rates as well as the safety of the program. For smaller
 communities a regional or statewide evaluation system may be preferable
to individual community programs.
.A
Stults KR; Brown DD.
.I 338
.U
87052090
.S
Circulation 8703; 74(6 Pt 2):IV18-22
.M
```

Figure 5: Text Visualization within XmdvTool (a) Without annotation (b) With annotation (along with a uri display window showing the complete text for the selected document)

ployed for text extraction. The text was tokenized by the most used tokenization options; the words from the SMART stoplist (524 common words), such as ”the” and ”of”, are neglected before tokenization; the Porter stemming algorithm was applied for all words before they are counted. After tokenization, a document-term matrix was acquired and was processed by dimension reduction algorithms such as Multidimensional Scaling, Self-Organizing Maps and Principal Component Analysis.

Figure 5 (a) shows the visualization of document collection of medical abstracts (in 298 files, and taking 215 terms into account) without the data mapping framework. This figure shows the importance of including semantic meaning within the visualization system so that the data analyst can analyze the data more effectively. For example, an analyst would be completely lost in numerical data when he wants the association between the terms that are being visualized to the set of documents he has selected. Figure 5 (b) changes the same visualization in (a) to a more comprehensible visualization by presenting the document when the user clicks over a particular data entry. Figure 6 shows the XML encoding produced by the Data Generation helper component.

7.3 Implementation

The implementation of the data mapping process is in a separate static library from the XmdvTool. This framework has been implemented in C++ using object oriented concepts so that it is easy to extend the framework for other data types. Once the framework was ready, it just took 2-3 man weeks time to integrate each of the nominal and text visualization capabilities to XmdvTool. The data input module and the data display modules for each of the visualization techniques needed to be modified to integrate loading and displaying of type information within the tool.

We believe that it would take more or less the same amount of time for integrating the implemented framework within other visualization tools depending on the complexity of the implementation of the system.

8 Conclusion and Future Work

The key contribution of this work is the development of a general framework to enable heterogeneous data types to be included within the environment of an existing visual data exploration tool and to enable capturing and integration of data semantics into the system. The main component of this framework consists of a data mapping process, which enables the mapping between varied data types to a form that the visualization tool can interpret. This enables the framework to present the semantic information either separate from or integrated with the display. We have a working prototype implementation for a publicly available multivariate visualization tool which validates the applicability of the framework.

Future research plans include incorporating this framework within other visualization tools. This effort will help validate the generality of the framework and demonstrate the impact that this idea can have in the development of future visualization systems. Even though the framework includes many of the data types found in most data sets, the extension to other data types will be explored to verify that the system is sufficiently flexible. The inclusion of function-driven mappings in addition to the currently implemented enumerated mappings within the data mapping knowledge will certainly reduce the storage requirements of the metadata. Currently, the mapping helper tools - data mapping generators and manipulators - are a separate system of tools and utilities from the system. It would be beneficial to the user of the system if the entire data mapping process could be executed through a graphical user interface - a mapping editor.

9 Acknowledgments

We gratefully acknowledge our colleagues in the XmdvTool group at WPI for their contributions to this research. The research funds from NSF and NSA are gratefully appreciated.

References

[1] Advanced Visual Systems Inc. Avs/express visualization edition. http://www.avs.com/software/soft_t/avsxps.html, 2003.

[2] D. Andrews. Plots of high dimensional data. In *Biometrics, Vol. 28, p. 125-36*, 1972.

[3] W. Basalaj. Proximity visualization of abstract data. http://www.pavis.org/, 2001.

[4] M. Chalmers and P. Chitson. Bead: Explorations in information visualization. In *Research and Development in Information Retrieval*, pages 330–337, 1992.

[5] R. P. Cody, R. Cody, and J. Smith. *Applied Statistics and the SAS Programming Language*. Prentice Hall, 4th edition, 1997.

[6] G. S. Davidson, B. Hendrickson, D. K. Johnson, C. E. Meyers, and B. N. Wylie. Knowledge mining with vxinside: Discovery through interaction. In *Journal of Intelligent Information Systems, Vol. 11, No. 3*, pages 259–285, 1998.

[7] M. Friendly. Mosaic displays for multi-way contingency tables. 89:190–200, 1994.

[8] M. Friendly. Visualizing categorical data. In M. G. Sirken, D. J. Herrmann, S. Schechter, N. Schwarz, J. M. Tanur, and R. Tourangeau, editors, *Cognition and Survey Research*, pages 319–348. John Wiley & Sons, Inc., New York, 1999.

[9] Y. Fua, M. Ward, and E. Rundensteiner. Hierarchical parallel coordinates for exploration of large datasets. In *Proc. of Visualization '99, p. 43-50*, San Francisco, California, USA, Oct. 1999.

[10] M. J. Greenacre. *Correspondence Analysis in Practice*. Academic Press, London, 1993.

[11] J. Hartigan and B. Kleiner. Mosaics for contingency tables. In W. F. Eddy, editor, *Computer Science and Statistics: Proceedings of the 13th Symposium on the Interface*, pages 268–273, New York, 1981. Springer-Verlag.

[12] S. Huang, M. O. Ward, and E. A. Rundensteiner. Exploration of dimensionality reduction for text visualization. Technical Report TR-03-14, Worcester Polytechnic Institute, Computer Science Department, 2003.

[13] A. Inselberg and B. Dimsdale. Parallel coordinates: A tool for visualizing multidimensional geometry. In *Proc. of Visualization '90, p. 361-78*, 1990.

[14] B. Johnson. Treeviz: treemap visualization of hierarchically structured information. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 369–370. ACM Press, 1992.

[15] R. A. Johnson and D. W. Wichern. *Applied Multivariate Statistical Analysis*. Prentice Hall International, Inc., second edition, 1988.

[16] T. Kamada and S. Kawai. A general framework for visualizing abstract objects and relations. In *ACM Trans. Gr.*, volume 10, pages 1–39, 1991.

[17] D. Keim and H. Driegel. Visdb: Database exploration using multidimensional visualization. In *Computer Graphics & Applications, Sept. 1994, p. 40-49*, 1994.

[18] D. Keim, H. Kriegel, and M. Ankerst. Recursive pattern: a technique for visualizing very large amounts of data. In *Proc. of Visualization '95, p. 279-86*, San Francisco, CA, USA; 27 Oct.-1 Nov. 1995, 1995. ACM; New York, NY, USA.

[19] E. Kolatch and B. Weinstein. Cattrees: Dynamic visualization of categorical data using treemaps. http://www.cs.umd.edu/class/spring2001 /cmsc838b/Project/Kolatch_Weinstein, 2001.

[20] A. Krause and M. Olson. *The Basics of S-PLUS*. Springer Verlag, 3rd edition, 2002.

[21] J. LeBlanc, M. Ward, and N. Wittels. Exploring n-dimensional databases. In *Proc. of Visualization '90, p. 230-7*, San Francisco, CA, USA; 23-26 Oct. 1990, 1990. IEEE Comput. Soc. Press; Los Alamitos, CA, USA.

[22] M. Livny, R. Ramakrishnan, K. S. Beyer, G. Chen, D. Donjerkovic, S. Lawande, J. Myllymaki, and R. K. Wenger. DEVise: Integrated querying and visualization of large datasets. In *Proc ACM SIGMOD Intl Conf on Management of Data, May 13-15, 1997, Tucson, Arizona, USA*, pages 301–312. ACM Press, 1997.

[23] S. Ma and J. Hellerstein. Ordering categorical data to improve visualization, 1999.

[24] A. K. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/~mccallum/bow, 1996.

[25] G. S. Owen, G. Domik, T.-M. Rhyne, K. W. Brodlie, and B. S. Santos. Hypervis - teaching scientific visualization using hypermedia. http://www.siggraph.org/education/materials/HyperVis/hypervis.htm, 1999.

[26] D. Rathjens. *MineSet User's Guide*. Silicon Graphics, Inc., 1996.

[27] R. Resnick and M. O. Ward. Visualizing nominal data. http://www.cs.wpi.edu/~matt/courses/cs563/talks/nominal_data/index.html, 1997.

[28] J. S. Risch, R. A. May, S. T. Dowson, and J. J. Thomas. A virtual environment for multimedia intelligence data analysis. pages 33–41, 1996.

[29] G. E. Rosario, E. A. Rundensteiner, D. C. Brown, and M. O. Ward. Mapping nominal values to numbers for effective visualization. Technical Report TR-03-11, Worcester Polytechnic Institute, Computer Science Department, 2003.

[30] C. Stolte and P. Hanrahan. Polaris: A system for query, analysis and visualization of multi-dimensional relational databases. *Information Visualization 2000*, 2000.

[31] D. F. Swayne, D. Cook, and A. Buja. Xgobi: Interactive dynamic data visualization in the x window system. *Journal of Computational and Graphical Statistics*, 7(1), 1998.

[32] P. F. Velleman. *Learning Data Analysis With Data Desk*. W H Freeman and Co., 1993.

[33] Visualization and Imagery Solutions, Inc. Opendx: The open source software project based on ibm's visualization data explorer. http://www.opendx.org/index2.php, 2003.

[34] M. Ward. Xmdvtool: Integrating multiple methods for visualizing multivariate data. In *Proc. of Visualization '94*, pages 326–33, Washington, DC, USA; 17-21 Oct. 1994, 1994. IEEE Comput. Soc. Press; Los Alamitos, CA, USA.

[35] C. Ware. *Information Visualization: Perception for Design*. Harcourt Publishers Ltd, 2000.

[36] J. A. Wise, J. J. Thomas, K. Pennock, D. Lantrip, M. Pottier, A. Schur, and V. Crow. Visualizing the non-visual: Spatial analysis and interaction with information from text documents. In *Proc. of Information Visualization '1995, p. 51-58*, 1995.