ELECTRONIC PUBLISHING, VOL. 8(2 & 3), 167–179 (JUNE & SEPTEMBER 1995)

XTABLE — A tabular editor and formatter

XINXIN WANG
Nortel
P.O. Box 3511, Station C
Ottawa, Ontario, K1Y 4H7, Canada

DERICK WOOD
Department of Computer Science
Hong Kong University of Science & Technology
Clear Water Bay, Kowloon, Hong Kong

e-mail: xinxin@nortel.ca, dwood@cs.ust.hk

SUMMARY

XTABLE is a prototype interactive tabular editor and formatter for the design of high-quality tables and for the exploration of tabular data from different viewpoints. It abstracts the multidimensional logical structure of a table and provides a mechanism to map an abstract table into different two-dimensional presentations. We present XTABLE from four aspects: abstract model, presentational model, system structure, and user interface. We also discuss the merits and limitations of XTABLE.

KEY WORDS Tabular abstraction Tabular editing Tabular formatting

1 INTRODUCTION

A table is a collection of interrelated items, which may be numbers, text, symbols, figures, mathematical equations, or even other tables. We can divide the items into two groups: entries and labels. *Entries* are the basic data a table displays and *labels* are the auxiliary data that are used to locate entries. Labels are grouped into *categories* that are organized hierarchically. For example, Table 1 presents the average marks for the assignments and examinations of a course offered in the three trimesters of 1991 and 1992. The marks are the entries and the strings that denote the years, the terms, and the kinds of marks are the labels. Furthermore, Year is a category that consists of the labels 1991 and 1992. Term is another category that consists of the labels Winter, Spring, and Fall. Mark is a category that consists of the subcategories Assignments and Examinations and the label Final Grade. There are logical associations among the entries and the labels. Each entry is associated with one label sequence from each of the categories. For example, the entry 85 at the top-left corner of Table 1 is associated with the labels 1991, Winter, and A1. The tabular items and their logical relationships define the *logical structure* of a table and the number of categories defines the *logical dimension* of a table. Table 1 has three categories; thus, it is a three-dimensional table.

We usually present tabular data as a row–column structure that has four main regions. The *stub* is the lower left region that contains the row headings. The *boxhead* is the upper right region that contains the column headings. The *stub head* is the upper left region that may either be empty or contain the categories of the row headings and column headings. The *body* is the lower right region that contains the entries. The intersection of a row and a column is called a *cell* and a rectangular collection of cells is called a *block*. We can

CCC 0894–3982/95/020167–13
©1995 by John Wiley & Sons, Ltd.
Received 30 March 1996
Revised 30 June 1996

Table 1. The average marks for 1991–1992

	Assignments			Examinations		Grade
	A1	A2	A3	Midterm	Final	
1991						
Winter	85	80	75	60	75	75
Spring	80	65	75	60	70	70
Fall	80	85	75	55	80	75
1992						
Winter	85	80	70	70	75	75
Spring	80	80	70	70	75	75
Fall	75	70	65	60	80	70

present a multidimensional table as a row–column structure in different topologies and styles. To present multidimensional tables in two dimensions, we need to associate more than one category with the stub or with the boxhead. For example, in Table 1, we assigned the categories Year and Term to the stub and the category Mark to the boxhead. If we move the category Year to the boxhead, we obtain a different topology and presentation (or view) of the data. Different orderings of the labels in a category or different orderings of the categories in the stub and in the boxhead also generate different views of the data.

A table can convey two kinds of information: associations and patterns among entries. *Associations* describe the relationships among entries and *patterns* describe the characteristics of groups of entries. Table 2, from Ehrenberg's text [1], shows the correlations for 10 TV programs based on whether people in a sample of 7,000 UK adults said they "really liked to watch" a range of programs such as World of Sport (WoS), Match of the day (MoD), and Panorama (Pan). TV programs are subcategories of the two TV broadcasting stations: ITV and BBC. In Table 2, the associations and the correlations between pairs of TV programs are easily understood. Table 2, however, does not show any clear pattern among the correlations for the 10 TV programs. After combining the TV programs with the corresponding TV broadcasting stations and reordering them, we obtain Table 3 that shows a cluster for the five Sports programs and another cluster for the five Current Affairs programs. From this presentation, we can clearly see three patterns in the data: correlations of 0.3 to 0.6 among the five Sports programs, correlations of 0.2 to 0.5 among the five Current Affairs programs, and correlations of approximately 0.1 between these two clusters. These two presentations contain the same items and the same logical associations among the items. It is the topological rearrangement of the items that enables a reader to see the clustering and the high correlations immediately. Before reordering the rows in Table 2, we changed the hierarchy of the labels in the stub by combining the TV broadcasting stations with their associated TV programs. In addition, the extra spacing between the Sports programs and the Current Affairs programs in Table 3 helps to highlight the patterns. Thus, to discover patterns in tabular data and to present them clearly with a row–column structure, we need to model tables based on the logical associations among items, to be able to edit category hierarchies, to topologically rearrange items in two dimensions, and to specify styles to emphasize patterns with typographical cues such as fonts, rules, and white space.

Table 2. The initial table of correlations for 10 TV programs

Programs		PrB	Thw	ToD	WoS	GrS	LnU	MoD	Pan	Rgs	24H
ITV	PrB		0.1	0.1	0.5	0.5	0.1	0.5	0.2	0.3	0.1
	Thw	0.1		0.3	0.1	0.1	0.2	0.1	0.4	0.1	0.4
	ToD	0.1	0.3		0.1	0.1	0.2	0.0	0.2	0.1	0.2
	WoS	0.5	0.1	0.1		0.6	0.1	0.6	0.2	0.3	0.1
BBC	GrS	0.5	0.1	0.1	0.6		0.1	0.6	0.2	0.3	0.1
	LnU	0.1	0.2	0.2	0.1	0.1		0.0	0.2	0.1	0.3
	MoD	0.5	0.1	0.0	0.6	0.6	0.0		0.1	0.3	0.1
	Pan	0.2	0.4	0.2	0.2	0.2	0.2	0.1		0.1	0.5
	Rgs	0.3	0.1	0.1	0.3	0.3	0.1	0.3	0.1		0.1
	24H	0.1	0.4	0.2	0.1	0.1	0.3	0.1	0.5	0.1	

Current tabular editing systems focus mainly on the composition of tables based on the associations among the tabular data. Systems such as `tbl` [2], TABLE [3], and Beach's system [4] model tables with a row–column structure. With these systems, we are limited in the presentation of data to reordering rows and columns. It is, however, difficult to obtain more presentations by reordering categories in the stub and the boxhead or by moving a category from the stub to the boxhead and vice versa. Some systems, such as Improv [5] and Vanoirbeek's system [6,7] model tables with a multidimensional logical structure. Improv [5] is an improved version of Lotus 1-2-3. It is an interactive commercial system for the editing and formatting of tabular data for finance and business. Tables are defined by specifying multiple categories in both the horizontal and vertical dimensions of a spreadsheet. The labels of these categories are placed at the top or at the left-hand side of the spreadsheet. Entries are placed in cells that are addressed by the labels of different categories. Vanoirbeek's system [6,7] is based on a table model in which a table is specified as a collection of entries that are semantically connected to multiple labels of different categories. Vanooirbeek's model is the first one to abstract a table from its presentation and the first one to provide a hierarchical model of the tabular abstraction. The logical structure of a table, in Vanoirbeek's system, is modeled by a tree with additional edges. A table consists of a set of logical dimensions (categories) and a set of items (entries). The logical dimensions include rubrics (labels) which may themselves contain subrubrics; additional edges are used in the tree to represent the connections between items and rubrics. As we shall see, in Section 2, our abstract model has much similarity to Vanoirbeek's model, but is sufficiently different to be of independent interest. Both Improv and Vanoirbeek's system are able to specify the multidimensional logical structures of tables. Neither of them, as far as we are aware, provides sufficiently many operations to modify the logical structure of a table. Both systems offer only the basic functions to create a new logical structure interactively. Some changes to an existing structure, such as to combine and split categories and to promote and demote subcategories, may require the user to abandon the old structure and create a new one.

XTABLE is an interactive tabular composition system that runs in a UNIX and X Windows environment. It is not only a tool for the design of high-quality presentations of

Table 3. The modified table of correlations for the 10 TV programs of Table 2

Programs	WoS	MoD	GrS	PrB	Rgs	24H	Pan	Thw	ToD	LnU
ITV WoS		0.6	0.6	0.5	0.3	0.1	0.2	0.1	0.1	0.1
BBC MoD	0.6		0.6	0.5	0.3	0.1	0.1	0.1	0.0	0.0
BBC GrS	0.6	0.6		0.5	0.3	0.1	0.2	0.1	0.1	0.1
ITV PrB	0.5	0.5	0.5		0.3	0.1	0.2	0.1	0.1	0.1
BBC Rgs	0.3	0.3	0.3	0.3		0.1	0.1	0.1	0.1	0.1
BBC 24H	0.1	0.1	0.1	0.1	0.1		0.5	0.4	0.2	0.3
BBC Pan	0.2	0.1	0.2	0.2	0.1	0.5		0.4	0.2	0.2
ITV Thw	0.1	0.1	0.1	0.1	0.1	0.4	0.4		0.3	0.2
ITV ToD	0.1	0.0	0.1	0.1	0.1	0.2	0.2	0.3		0.2
BBC LnU	0.1	0.0	0.1	0.1	0.1	0.3	0.2	0.2	0.2	

tables, but also it is a tool for the exploration of tabular data from different viewpoints. XTABLE abstracts the multidimensional logical structure of a table which it then maps to different presentations according to user-defined topological and style specifications. The formatted concrete tables can be displayed through a Motif interface or transformed into LaTeX [8] source files. XTABLE provides operations to edit the multidimensional logical structure, the topology, and the style of tables.

We first present the underlying abstract model of XTABLE in Section 2 and describe the presentational model used by XTABLE to map the abstract model to a row–column structure in Section 3. Then, we give an overview of the system structure in Section 4 and an introduction to the user interface of XTABLE in Section 5. In Section 6, we discuss the merits and limitations of XTABLE.

2 ABSTRACT MODEL

When one designs a table, one normally has a logical structure in mind before one selects a presentational form. Thus, we should deal with the logical structure and the presentation separately. There are at least two advantages with the separation the logical structure and the presentation. First, tables can be manipulated independently of their presentations. For example, to remove a label from a category, we no longer have to determine which rows or columns should be removed from the presentation. Second, by associating different topologies and styles with a logical structure, we can easily obtain various presentations for a table.

The abstract model for XTABLE is based on the multidimensional logical structure and is independent of any characteristics that are related to the presentation of a table. We specify the logical structure of a table as an *abstract table*, which describes the hierarchical label structure of categories and the logical associations among labels and entries. A *label* can be any string of characters and symbols. A *labeled set* is a set together with a label which we specify as an ordered pair $(label,set)$. A *category* is defined inductively as either a labeled empty set $(L,\emptyset)$ or a labeled set of categories such that the labels of the

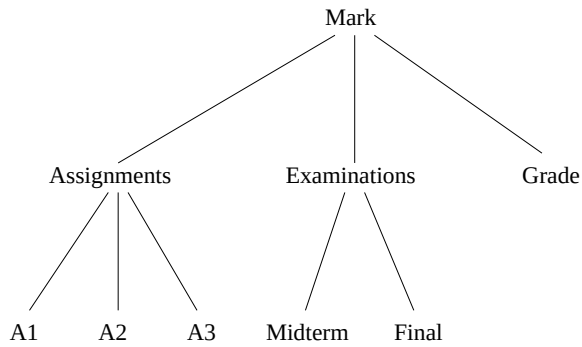

Figure 1. The tree of the category Mark in Table 1

categories are pairwise distinct. Clearly a category can be viewed as a labeled tree; for example, the category Mark in Table 1 can be viewed as the tree of Figure 2. Indeed, Raymond [9] describes categories as partial orders and defines tables with partial orders. Our approach is similar in that a labeled tree defines a partial order. A *label sequence* is either a label or a label sequence s followed by a label l that we denote by $s.l$. We use label sequences to uniquely identify the subcategories and labels in a category. Since categories have a hierarchical structure, label sequences determine paths from the root to a node in the hierarchy. Thus, for the category Mark of Table 1, the label sequence Mark.Assignments determines the subcategory Assignments, and Mark.Assignments.A1 determines the empty-set subcategory $(A1,\emptyset)$; see Figure 2. A *terminal label sequence* is a label sequence that determines an empty-set subcategory and these subcategories are used to index entries in a table. (Terminal label sequences also determine the terminal nodes in the labeled tree of a category.) We use the notation $tls(C)$ to denote the set of terminal label sequences of a category C; $tls(C)$ is the set of all subcategories of C that can label an entry.

An *abstract table* consists of a set of categories and a map that associates values with table entries. It is specificied by an ordered pair (C,δ), where C is a set $\{C_1,C_2,\ldots,C_n\}$ of categories and δ is a map from $\otimes tls(C)$ to the universe of possible values. (The notation $\otimes tls(C)$ is the unordered Cartesian product of $tls(C_1),tls(C_2),\ldots,tls(C_n)$.) We use $\otimes tls(C)$ to model the entry set of a table. Each entry is identified or indexed by a set in $\otimes tls(C)$ and is assigned a value by δ. For example, we can specify the logical structure of Table 1 with the abstract table $T = (C,\delta)$ in which C has following three categories:

$(Year,\{(1991,\emptyset),(1992,\emptyset)\})$;
$(Term,\{(Winter,\emptyset),(Spring,\emptyset),(Fall,\emptyset)\})$;
$(Mark, \{(Assignments, \{(A1,\emptyset), (A2,\emptyset), (A3,\emptyset)\}),$
$\quad (Examinations, \{(Midterm,\emptyset), (Final,\emptyset)\}),$
$\quad (Grade, \emptyset)\})$.

The corresponding map δ is defined by:

$\delta(\{Year.1991, Term.Winter, Mark.Assignments.A1\}) = 85$;
$\delta(\{Year.1991, Term.Winter, Mark.Assignments.A2\}) = 80$;
$\delta(\{Year.1991, Term.Winter, Mark.Assignments.A3\}) = 75$;
$\delta(\{Year.1991, Term.Winter, Mark.Examinations.Midterm\}) = 60$;
$\delta(\{Year.1991, Term.Winter, Mark.Examinations.Final\}) = 75$;
$\delta(\{Year.1991, Term.Winter, Mark.Grade\}) = 75$;

$\vdots$

$\delta(\{Year.1992, Term.Fall, Mark.Assignments.A1\}) = 75;$
$\delta(\{Year.1992, Term.Fall, Mark.Assignments.A2\}) = 70;$
$\delta(\{Year.1992, Term.Fall, Mark.Assignments.A3\}) = 65;$
$\delta(\{Year.1992, Term.Fall, Mark.Examinations.Midterm\}) = 60;$
$\delta(\{Year.1992, Term.Fall, Mark.Examinations.Final\}) = 80;$
$\delta(\{Year.1992, Term.Fall, Mark.Grade\}) = 70.$

Since we use sets to specify categories, the categories are unordered and the labels in a category or a subcategory are also unordered. Ordering is an issue of topology, and we do not include it in the abstract model. We will deal with category ordering and label ordering in Section 3.

The second author well remembers listening to Vanoirbeek's presentation at EP 92 [7] with a mixture of admiration, enjoyment, and horror! We had developed a preliminary abstract model in late 1991 and early 1992 without any knowledge of Vanoirbeek's work. The model had the elements described here but the formalism was not as complete. Vanoirbeek's talk gave us the motivation to describe our model in its preliminary form [10]. A natural question is: How similar are our models and how different are they? The similarity is clear, we each abstract a table from its presentational form and describe its categories as hierarchies. In our work, we specify from the beginning that categories are not ordered in any way, other than by the hierarchy of subcategories. Thus, given two subcategories of a category neither one is first. For example, in Table 1, 1991 is not automatically before 1992. Similarly, we do not impose any order among the categories in the abstract model. For example, in Table 1, Year, Term, and Mark do not have any inherent order. We also specify the asssociation between categories and entries functionally, whereas Vanoirbeek chose a semantic association represented by additional edges in the hierarchical model. Lastly, Vanoirbeek models a table as a single hierarchy (with additional edges for entries) in which it is unclear whether there is an implicit ordering of categories and an implicit ordering within categories. Since Vanoirbeek's model has been incorporated in Grif, there is an implicit ordering in that framework. Clearly, Vanoirbeek's model, while an important stepping stone, is incomplete; it is completed by its implementational environment. Our model, on the other hand, is intended to be a complete, stand-alone model for tables and to be independent of any implementational environment.

3 PRESENTATIONAL MODEL

To present multidimensional tables in two dimensions, we may need to associate more than one category with the stub or with the boxhead. In this case, some labels appear more than once. For example, in the stub of Table 1, the labels of the category Term appear twice to clarify the association between Year and Term. The arrangement of labels determines the arrangement of entries. Each entry usually appears to the right of its associated labels in the stub and beneath its associated labels in the boxhead. Different types of typographic cues can be used to help readers search for information and highlight data patterns. We can use rules or white space to separate tabular items, distinct type faces or point sizes to distinguish different types of items, and background colors or patterns to highlight important information. The presentational model for XTABLE consists of two parts: topological specification and style specification.

3.1 Topological specification

A *topological specification* defines the relative arrangement of tabular items in two dimensions. We use two topological rules to specify the category orderings, one for the stub and the other for the boxhead:

STUB: $C_1^s, C_2^s, \ldots, C_m^s$
BOXHEAD: $C_1^b, C_2^b, \ldots, C_n^b$,

where C_i^s is the ith category in the stub and C_j^b is the jth category in the boxhead. For example, the category orderings of Table 1 are specified by

STUB: Year, Term
BOXHEAD: Mark.

The label ordering within a category or subcategory is another attribute that affects the topological arrangement. In Table 1, the labels in category Term are arranged in the order Winter, Spring, Fall. If we reverse the order to give Fall, Spring, Winter, we get a different arrangement. Therefore, we use another topological rule to specify the label ordering within a category:

ORDER C: $L_1, L_2, \ldots, L_k$,

where C is a category or subcategory and L_i is the ith label of C in the ordering. Sometimes, we do not explicitly specify the label ordering for a category; instead, we implicitly specify the ordering using a standard ordering such as numerical or lexicographic. Thus, another possible topological rule for label ordering is

ORDER C: ⟨ordering option⟩,

where ⟨ordering option⟩ includes numerical order, reverse numerical order, lexicographic order, and reverse lexicographic order. For example, the label orderings for the categories in Table 1 are specified by:

ORDER Year:	lexicographic order
ORDER Term:	Winter, Spring, Fall
ORDER Mark:	Assignments, Examinations, Grade
ORDER Mark.Assignments:	lexicographic order
ORDER Mark.Examinations:	Midterm, Final.

Once we are given a topological specification, we can determine the topological positions of the labels and the entries of a table. The geometric positions, however, cannot be determined without a style specification.

3.2 Style specification

A *style specification* determines the final appearance and the physical dimensions of a table. It consists of style rules for different components of a table. A style rule consists of a scope

and a set of formatting attributes that are associated with the scope. For example, tables (scope) are displayed in Roman (formatting attribute) with only horizontal rules (formatting attribute). The style rules for tables fall into three classes: presentation-oriented style rules, content-oriented style rules, and layout-oriented style rules. A *presentation-oriented style rule* has a scope that is a major region of a table: the table itself, the stub, the boxhead, the stub head, and the body. It affects the cells and separations (rules and spacing) in the major regions. A *content-oriented style rule* has a scope that is a logical object or a set of logical objects of an abstract table, including a category, a subcategory, a label, an entry, an entry value, and an entry set. It affects only the cells in which the logical objects are located and the separations of these cells. A *layout-oriented style rule* has a scope that is a layout component of a concrete table, including a row, a column, and a block. It always affects the cells and separations in the layout component no matter what objects are put into it.

XTABLE provides eight types of formatting attributes for style rules. *Cell style* allows us to control the appearance and the background of the items in cells by specifying type faces and sizes, background colors, line spacing, leading spacing, horizontal and vertical alignment options, and so on. *Separation style* specifies white space or different types of horizontal and vertical rules to separate different kinds of items. *Frame style* enables us to select white space or different types of rules to surround a rectangular area. *Arrangement style* enables us to control the arrangement of labels in the stub, the boxhead, and the stub head. For example, we can arrange the labels hierarchically as in the boxhead of Table 1 or indent the labels as in the stub. *Spanning style* allows us to span the entries that have the same value within a rectangular block. We can span entries in one dimension without spanning them in the other dimension, or span the entries in two dimensions by giving priority to one dimension. *Grouping style* groups items into blocks of a given number of rows by the use of either white space or rules. *Category heading style* allows us to either display the category heading above its labels or hide the category heading. *Size constraints* enables us to restrict the sizes of columns, rows, and tables.

The appearance of a table can be governed by many style rules. Some style rules are given by a publisher or an editor of a book to achieve a uniform appearance for all tables in the same book. Some style rules are given by a table designer for the specific presentation of one table. XTABLE classifies the style rules for a table into two classes: collective style rules and specific style rules. A *collective style rule* is a style rule for the presentation of a collection of tables and a *specific style rules* is a style rule for a particular table.

We do not have to specify style rules for all components of a table. A component can inherit the style rules of one of its super-components or the default style rules. For example, a cell that holds a label can inherit the style rules of the label's category and the cell's region (stub, boxhead, or stub head). Thus, we need to find approaches to solve style inheritance. If we were able to use a tree structure to describe the relationships among the tabular components, we would define a priority order for style inheritance based on single inheritance. There are, however, multiple inheritances in a table. For example, a cell belongs to its row and column, which do not contain each other completely; thus, it can inherit style rules from both the row and the column. Therefore, the approaches for style inheritance should handle multiple inheritance. XTABLE handles multiple inheritance in two steps. It first tries to combine the style rules of all super-objects. For instance, italic Roman is the result of combining Roman and italic. Whenever there is no satisfactory combination, for example, combining Roman and Courier, XTABLE uses the style rules of the super-object with a priority ordering determined by the system and the user. The priority

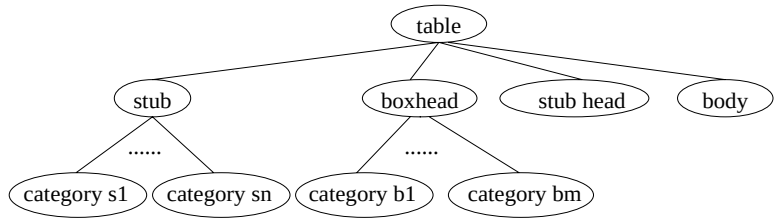

Figure 2. The genealogical relationships of some scopes

for some scopes, including the whole table, the stub, the boxhead the stub head, the body, and the categories, is predetermined. We use a genealogical tree, shown in Figure 2, to describe the relationships between these scopes. Thus, we can predetermine the priority for these scopes using single inheritance. Since the remaining scopes, including the rows, the columns, the blocks, the labels, the subcategories, the entries, the entry set, and the entry values, are specified infrequently and may cause multiple inheritance, the priority for these scopes is determined by users according to their requirements. For the style rules with these scopes, the last specified style rule has the highest priority. Moreover, these style rules have higher priority than the style rules in the preceding single-inheritance ordering.

4 OVERALL SYSTEM STRUCTURE

We specify, in XTABLE, the logical structure of a table using the abstract model given in Section 2 and the presentation using the topological and style rules described in Section 3. Given an abstract table, a topological specification, and a style specification, we generate a concrete table using two processes. First, the *arrangement process* generates a grid structure and a set of size constraints for the columns and rows in the grid structure. Then, the *formatting process* determines the physical dimensions of the columns and rows for the grid structure according to the size constraints. The formatting algorithm and its complexity analysis are described elsewhere [11,12].

XTABLE accepts three kinds of input: table files, collective style files, and user instructions. A table file has three parts: an abstract table, a topological specification, and a specific style specification. A collective style file contains only collective style rules. XTABLE maintains four major data structures for the abstract table, the topological specification, the specific style specification, and the collective style specification. Their initial values are given by a table file and a collective style file or assume defaults if neither table file nor collective style file is provided. During the interactive editing process, these data structures are updated according to user commands. We use Motif as the interface between a user and the system. We generate three intermediate data structures whenever XTABLE displays a table or compiles a table specification into a LaTeX file. The arrangement process generates a grid structure and a set of size constraints, and then the formatting process generates a concrete table. A concrete table can either be displayed through the Motif interface or be transformed into a source file for LaTeX. Due to limitations of the LaTeX table environment, we transform a concrete table to the LaTeX picture environment in which all tabular items and rules are treated as graphical objects.

5 USER INTERFACE

We adopt an object-oriented technology in XTABLE to provide an interactive environment for the manipulation of the logical structure, the topology, and the styles of tables. Tabular components are classified into object classes and editing operations are associated with them.

There are three kinds of object classes: presentational objects, logical objects, and layout objects. The *presentational objects* include the entire table and the four major regions: the stub, the boxhead, the stub head, and the body. The *logical objects* are the logical components of an abstract table including category, subcategory, label, entry, entry set, and entry value. The *layout objects* are the layout components of a concrete table including block, row, and column. There are also three kinds of operations for the object classes: logical, topological, and style. A *logical operation* changes the logical structure of a table. We can change the logical dimensions of tables by adding, removing, combining, and splitting categories, and update the label hierarchy of a category by moving, copying, promoting, and demoting subcategories. We can also edit the labels and entries. A *topological operation* changes only the topological specification of a table, for example, transposing a table, moving a category from the stub to the boxhead, or changing the ordering for a category. A *style operation* changes only the style specification of a table, for example, changing the cell style, the separation style, or the arrangement style for different objects.

XTABLE's user interface enables users to select editing objects by using mouse and to indicate the operations by the menu, tool-box, and dialog-box techniques. Figure 5 shows the main window of XTABLE in which a table is displayed. There are three editing areas in the main window: stub, boxhead, and table. The categories that are assigned to the stub (the boxhead) appear in the stub area (the boxhead area) and the concrete table is presented in the table area. A menu bar and a set of tool boxes are created for users to use for editing. Once users have selected an object and indicated an operation and its arguments, a new presentation of the table is generated in the table area after applying the operation to the object.

The most frequently used operations (Add, Remove, Copy, Move, Combine, Split, and Text) are provided as tool boxes. Once the user clicks on a tool box, the corresponding operation is active until the user clicks on another tool box. When a tool box is active, the user needs to indicate to which object the operation is applied and to specify the required arguments by pointing and dragging in the three editing areas. The tool box Select is used to indicate the editing objects for menu operations. The content of the current active object is displayed in the subwindow at the bottom of the main window.

Most topological operations, style operations, and system commands are listed in the menu bar. The menu File consists of input and output commands, such as reading a table file or a collective style file, and generating a LaTeX source files; the menu Edit consists of the other logical and topological operations that are not available as tool boxes; the menu Style consists of the style operations for specific style specification that can be applied only to the current edited table; menu Collective-Style consists of the style operations for collective style specification that can be applied to a collection of tables; the menu Calculation consists of the operations average, total, minimum, and maximum that are used to compute entry values; and the menu Setting consists of the commands for the selection of the system parameters. When a style operation is selected, a dialog box pops up to assist users to edit the formatting attributes of the style rules for the selected object.

Figure 3. The main window of XTABLE

6 CONCLUSIONS

XTABLE is a tool that helps users to design high-quality tables in two dimensions. It provides an interactive environment for editing the logical structure, topology, and style of a table and for easily presenting a table with different topologies and styles. XTABLE is also a tool that helps users to explore the data from different viewpoints. By arranging table items flexibly in two dimensions, users are able to discover relationships among or patterns in the data. This ability helps users to analyze and understand tabular data in an efficient way.

Since XTABLE is a prototype for validating our tabular model [11], it does not provide the functionality that a production system provides. For example, we have made the simplifying assumption that we do not model footnotes in the abstract model. Clearly, footnotes play an important role in tables. Although we do not model footnotes, a user can still use footnotes with any tabular entry. The limitation is that they are dealt with by the target typesetting system, they are not manipulable as abstract objects within our model. Second, the abstract model does not capture all tables even when we ignore footnotes. The model can be used to specify tables that have only a multidimensional logical structure. Not all tables have such a nice structure however; some tables are a combination of several tables [11].

The topological rules in the presentational model specify only the arrangement of labels in the stub and in the boxhead. This approach forces users to place the most frequently referenced items to the left or to the top of a table. Experiments [13] have proved that readers tend to ignore the labels that are put in the body and consider them as entries. Also, the presentational model cannot specify all styles observed in all tables. For example, we do not handle oblique lines; thus, we are unable to specify a table in which the headings of the categories in both dimensions are put in the stub head, separated by an oblique line. We also allow only horizontal typesetting of text, vertical typesetting is not provided.

We carried out experiments to measure how well our abstract model and presentational model specifies tables in the real world. We collected tables from books in the fields of statistics, sociology, science, and business. The results of the experiment [11] reveal that the abstract model can be used to specify 56 percent of the tables if we consider footnotes, or 97 percent of the tables if we ignore footnotes. The presentational model can be used to specify the topology of 94 percent of the tables and to specify the style of 97 percent of the tables. From this experiment, the majority of the tables in traditional printed documents can be specified with a multidimensional logical structure and our presentational model matches real-world tables.

ACKNOWLEDGEMENTS

This work has benefited from the interactions with Anne Brüggemann-Klein and Darrell R. Raymond over many years. David Levy also helped us to formulate our ideas about our editing, topological, and style models by his arguments against mathematical models in a train from Lausanne and EP 92.

We are also grateful for financial support from the Natural Sciences and Engineering Research Council of Canada and from the Information Technology Research Centre of Ontario.

REFERENCES

1. A. S. C. Ehrenberg, 'Rudiments of numeracy', *Journal of the Royal Statistical Society*, **A. 140, part 3**, 277–297, (1977).
2. M. E. Lesk, 'Tbl — a program to format tables', in *UNIX Programmer's Manual*, volume 2A, Bell Telephone Laboratories, Murray Hill, NJ, 7th edition, (January 1979).
3. T. J. Biggerstaff, D. M. Endres, and I. R. Forman, 'TABLE: Object oriented editing of complex structures', in *Proceeding of the 7th International Conference on Software Engineering*, pp. 334–345, (1984).
4. R. J. Beach, *Setting Tables and Illustrations with Style*, Ph.D. dissertation, Department of Computer Science, University of Waterloo, Canada, 1985. Also issued as Technical Report CSL-85-3, Xerox Palo Alto Research Center, Palo Alto, CA.
5. *Improv Handbook*, Lotus Development Corporation, Cambridge, MA, 1991.
6. Christine Vanoirbeek, *Une Modelisation de Documents pour le Formatage*, Ph.D. dissertation, Department d'Informatique, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, 1988.
7. Christine Vanoirbeek, 'Formatting structured tables', in *EP92(Proceedings of Electronic Publishing, 1992)*, ed., C. Vanoirbeek & G. Coray, pp. 291–309, Cambridge, UK, (1992). Cambridge University Press.
8. L. Lamport, *LaTeX: A Document Preparation System*, Addison-Wesley, Reading, MA, 1985.
9. D. R. Raymond, *Partial Order Databases*, Ph.D. dissertation, Dept. of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 1996.
10. X. Wang and D. Wood, 'An abstract model for tables', TUGBOAT, *The Communications of the TeX Users Group*, **14(3)**, 231–237, (October 1993).
11. Xinxin Wang, *Tabular Abstraction, Editing, and Formatting*, Ph.D. dissertation, Department of Computer Science, University of Waterloo, Canada, 1996. Available as Research Report CS-96-09, Department of Computer Science, University of Waterloo.
12. X. Wang and D. Wood, 'Tabular formatting problems'. To appear in the *Third International Workshop on Principles of Document Processing (PODP 96)*, 1996.
13. P. Wright, 'Using tabulated information', *Ergonomics*, **11(4)**, 331–343, (1968).