

Wang Notation Tool: A Layout Independent Representation of Tables

by

Piyushee Jha

A Thesis Submitted to the Graduate
Faculty of Rensselaer Polytechnic Institute
in Partial Fulfillment of the
Requirements for the degree of
MASTER OF SCIENCE
Major Subject: ELECTRICAL ENGINEERING

Approved:

Dr. George Nagy, Thesis Adviser

Rensselaer Polytechnic Institute
Troy, New York

May, 2008

CONTENTS

Wang Notation Tool: A Layout Independent Representation of Tables	i
LIST OF TABLES.....	v
LIST OF FIGURES	vii
ACKNOWLEDGMENT	ix
ABSTRACT	x
1. Introduction.....	1
1.1 TANGO.....	1
1.2 Wang Notation	2
1.3 Tables	3
1.3.1 Categories as Trees	4
1.3.2 Well-formed Tables	5
1.3.3 Virtual Headers	5
1.3.4 Unique Categories.....	6
1.3.5 Factors Affecting Table Processing Time.....	8
1.3.6 Foreign Tables.....	9
1.4 Organization of Thesis	10
2. Literature Review	11
2.1 Detection, Extraction, Interpretation and Understanding	11
2.2 Perspectives on Table Processing	12
2.3 Wang's Table Model.....	19
2.4 Techniques of Table Processing.....	20
3. Description of Interactive System	30
3.1 Overview of System.....	30
3.2 Early Versions of System.....	31
3.3 Detecting Tables in HTML pages	31

3.4	Generating Category Notation	32
3.4.1	Interactive Category Construction	33
3.4.2	Intermediate Category Processing.....	34
3.4.3	Error-Correction by User	36
3.4.4	Determining Final Category Notation.....	37
3.5	Generating Delta Notation	39
3.6	Generating XML Representation	39
3.6.1	Ontology to Describe Tables.....	40
3.6.2	XML Schemas.....	40
3.6.3	XML Generation in Matlab.....	41
3.7	Verifying Results with a GUI	42
3.8	System Logging	44
3.9	Matlab and WNT.....	44
3.10	Summary	45
4.	Evaluation of WNT Methodology	47
4.1	Preliminary Testing.....	47
4.2	Training.....	48
4.3	Evaluation	48
5.	Evaluation of WNT.....	50
6.	Future Work.....	58
6.1	Aggregations and Annotations.....	58
6.2	Automation and Learning	58
6.3	Improved Training	59
7.	Conclusion	60
	References.....	62

Appendix.....	64
A. Training Tables	64
B. Test Tables	66
C. Wang Notation	75
D. XML Representation.....	76
E. Log	82
F. Training PowerPoint	83
G. Quantitative Results	93

LIST OF TABLES

Table 1: Wang Table	2
Table 2: Number of Females with Degrees in Canada	6
Table 3: Original Population & Area Table	10
Table 4: Foreign Population & Area Table	10
Table 5: Distribution of Processing Time for T09, Average Over All Subjects	50
Table 6: Success Rate by Table, Average Over Subjects	51
Table 7: Average Times by Table	56
Table 8: University Degrees for Males (TRN1)	64
Table 9: Divorces by Province (TRN2).....	64
Table 10: Economy of Mali (TRN3)	65
Table 11: Food Services for Nunavut (TRN4)	65
Table 12: Wang Table (TRN5).....	65
Table 13: Induced Abortions by Province (T01)	66
Table 14: Deaths and death rate, by province (T02)	66
Table 15: Deaths and death rate, by province (T03)	67
Table 16: University Degrees (Females) by province (T04)	67
Table 17: Food and Drink for Alberta (T05)	67
Table 18: Food and drink for Newfoundland and Labrador (T06).....	68
Table 19: Food and drink for Prince Edward Island (T07)	68
Table 20: Infant mortality rates by province (T08)	68
Table 21: Lakes of Canada (T09)	69
Table 22: Mountains of Canada (T10).....	70
Table 23: Administrative units of Utah (T11)	71
Table 24: American Indian/Alaska Native Populations (T12)	72
Table 25: General info for Angola (T13)	72
Table 26: Bodies of Water (T14).....	73
Table 27: Economy of Albania (T15).....	73
Table 28: Economy of New Zealand (T16).....	73
Table 29: World population (T17).....	74

Table 30: Example Log	82
Table 31: Distribution of Processing Time for T01, Average Over All Subjects	93
Table 32: Distribution of Processing Time for T02, Average Over All Subjects	93
Table 33: Distribution of Processing Time for T03, Average Over All Subjects	93
Table 34: Distribution of Processing Time for T04, Average Over All Subjects	93
Table 35: Distribution of Processing Time for T05, Average Over All Subjects	94
Table 36: Distribution of Processing Time for T06, Average Over All Subjects	94
Table 37: Distribution of Processing Time for T07, Average Over All Subjects	94
Table 38: Distribution of Processing Time for T08, Average Over All Subjects	94
Table 39: Distribution of Processing Time for T10, Average Over All Subjects	95
Table 40: Distribution of Processing Time for T11, Average Over All Subjects	95
Table 41: Distribution of Processing Time for T12, Average Over All Subjects	95
Table 42: Distribution of Processing Time for T13, Average Over All Subjects	95
Table 43: Distribution of Processing Time for T14, Average Over All Subjects	96
Table 44: Distribution of Processing Time for T15, Average Over All Subjects	96
Table 45: Distribution of Processing Time for T16, Average Over All Subjects	96
Table 46: Distribution of Processing Time for T17, Average Over All Subjects	96

LIST OF FIGURES

Figure 1: Tables as Trees	4
Figure 2: Virtual Header	5
Figure 3: Two Unique Categories (Example 1).....	7
Figure 4: Four Unique Categories (Example 2)	7
Figure 5: Category Trees for Example 1	7
Figure 6: Category Trees for Example 2	7
Figure 7: Merged Cells	10
Figure 8: Split Cells	10
Figure 9: Models of Table Structure.....	13
Figure 10: Observations in Table Recognition Literature	14
Figure 11: Transformations in Table Recognition Literature.....	15
Figure 12: Classifiers	16
Figure 13: Segmenters and Parsers.....	16
Figure 14: Regions Within a Table.....	19
Figure 15: Flowchart of the design of Silvia et al.	21
Figure 16: Table Topologies.....	23
Figure 17: Results from [8].....	23
Figure 18: Example of a Tree Model.....	24
Figure 19: Example of a DAG.....	24
Figure 20: Architecture for TINTIN system.....	25
Figure 21: Star Table	25
Figure 22: Layout Graphs	26
Figure 23: System of Green et al.	28
Figure 24: ASCII Version of Wang Table.....	32
Figure 25: Wang Table as Displayed in Matlab	33
Figure 26: Wang Table with Marked Categories	33
Figure 27: Wang Table After User Marks Categories.....	34
Figure 28: Control GUI for Selection of Categories	34
Figure 29: Category Tree.....	35

Figure 30: Indented Notation.....	35
Figure 31: Table of Contents Representation	35
Figure 32: Error Correction GUI.....	37
Figure 33: Indented Notation as seen in Matlab.....	37
Figure 34: General Nonsense Tree	38
Figure 35: Equivalent Binary Tree with Pointers	38
Figure 36: Ontology of a General Table.....	40
Figure 37: Matlab Structure for XML	41
Figure 38: XML for Table Schema	42
Figure 39: Verifying Delta Cell (1)	43
Figure 40: Verifying Delta Cell (2)	43
Figure 41: Verifying Category Cell (1)	43
Figure 42: Verifying Category Cell (2)	44
Figure 43: WNT Flowchart	46
Figure 44: GUI to control tables.....	47
Figure 45: WNT Results by Subject.....	51
Figure 46: WNT Results by Table.....	52
Figure 47: Average Total Time by Table	56
Figure 48: Average Total Time by Subject	57

ACKNOWLEDGMENT

I would like to express my gratitude for my adviser, Dr. George Nagy, for his valuable assistance, guidance, and encouragement in all my endeavors. I want to offer thanks to Dr. David Embley from Brigham Young University for his support and the hospitality he showed me when I visited. I also want to thank my fellow graduate students for all their help, in particular: Raghav Padmanabhan, Stephen Lynn, and Cui Tao. The Electrical, Computer, and Systems Engineering department of Rensselaer Polytechnic Institute was generous enough to support me through a teaching assistantship, one of the best experiences of my collegiate career, and for that, I am deeply indebted. This research was also funded by the National Science Foundation (Grant# 044114854). In addition, this research would not have been possible without the subjects whom I used for evaluation, thank you for taking so much time out of your day to help me.

My deepest gratitude goes to my family for their unflagging support and encouragement. To my father, who has advised me at every stage of life, you are my role model. He has inspired me to enter a technical field despite my former misgivings and much of my enthusiasm for learning can be attributed to him. To my mother, whose love and attention are responsible for the person I am today. She has always encouraged me to believe in myself, enjoy all aspects of life, and be strong and independent. To my brother, who is following my path into electrical engineering, I couldn't be more proud of you. This section would not be complete without an acknowledgement of my friends, who have been a valuable part of my life. In particular, I'd like to thank Nadeeka Yapa, Sayuri Yapa, and Matthew Flack.

ABSTRACT

The Wang Notation Tool (WNT) is a semi-automatic, interactive tool that converts tables from HTML pages to Wang notation and corresponding XML representation. Both are layout independent representations of tables where all relationships between cells are recorded in an abstract form that does not rely on the physical structure of tables. WNT requires minimal interaction to delineate the categories in a table, from which an intermediate category tree describing the relationships within each category is determined. The category trees are shown to the user for correction and/or approval. User correction at this step makes WNT robust because the user can modify the automatically generated category tree in almost any way. The approved category trees are used to generate a description of the relationship between each delta (content) cell and the categories as well as an XML representation of tables based on an ontology describing general trees. With current training methods, layout independent representations were generated for 98% of all tables, and were generated correctly for 71% of all tables. Evaluation indicates that with further training, most users will be able to rapidly and correctly generate a layout independent representation of tables using WNT.

1. Introduction

The Semantic Web combines various technologies to supplement or replace the content of web documents with descriptive data that will assist the user (human or automated agent) in decision making and will address their specific needs and wants. This can only be accomplished with an abundance of ontologically annotated data. However, creating ontologies is a difficult process. The first step of TANGO [1], a project that creates ontologies from the data found in tables, is to convert all the information in any given table into a standard form for easy comparison and manipulation. This thesis describes the creation of a semi-automatic tool – the Wang Notation Tool – that converts tables from HTML pages to Wang notation [2] and XML representation.

1.1 TANGO

Table Analysis for Generating Ontologies (TANGO) is an interdisciplinary project that aims to use conceptual modeling extraction techniques to convert structured data, such as tables, into ontologies by “understanding” the tables. TANGO operates in four steps:

1. Recognize and normalize table information
2. Construct mini-ontologies from normalized tables
3. Discover inter-ontology mappings
4. Merge mini-ontologies into a growing application ontology

To implement TANGO, information from several tables in any given domain (i.e., geopolitical information) must be assembled. The first step of TANGO, and the work reported in this thesis, is to recognize and normalize the tables. This is important because the same concepts and relations can be presented within a table in different ways. To create ontologies, it is necessary to separate the concepts and relations from the physical structure of a table. This is done by converting physical tables (tables with a visual structure) to Wang notation (and XML representation), which is consistent for all tables with the same content.

The next steps are to construct ontologies for every table and discover the matching concepts between mini-ontologies. These steps rely largely on lexical information. The final step is to merge the mini-ontologies using the mappings discovered between them

and resolving conflicts between ontologies. These steps are implemented at Brigham Young University in Provo, Utah.

1.2 Wang Notation

Wang proposed a layout-invariant representation of tables [2]. A table without physical structure is called an *abstract table*. As per Wang [2], an abstract table is specified by an ordered pair (C, δ) where C is a finite set of labeled domains (header, sub headers of tables, etc) and δ is a mapping from C to the universe of possible values. In other words, a table has two types of cells: category cells and delta cells. *Category cells* are the headers and sub headers in a table that describe the content of the table. *Delta cells* contain the content of the table. Wang Notation consists of two parts: category notation and delta notation. Table 1 shows the table from Wang's PhD thesis that was used as a point of reference during the creation of the Wang Notation Tool (WNT).

Table 1: Wang Table

Year	Term	Mark					
		Assignments			Examinations		Grade
		Ass1	Ass2	Ass3	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

The Wang table has three dimensions and therefore, three categories, which are shown below. Year is the first category with 1991 and 1992 as its subcategories. Term is the next category with winter, spring, and fall as its subcategories. Mark is the most complicated category with three subcategories (Assignments, Examinations, and Grade) among which Assignments and Examinations have their own subcategories (Ass1, Ass2, Ass3, and Midterm, Final, respectively).

$(Year, \{(1991, \phi), (1992, \phi)\})$

$(Term, \{(Winter, \phi), (Spring, \phi), (Fall, \phi)\})$

$(Mark, \{(Assignments, \{(Ass1, \phi), (Ass2, \phi), (Ass3, \phi)\}), (Examinations, \{(Midterm, \phi), (Final, \phi)\}), (Grade, \phi)\})$

The delta notation shows which category cells are related to each of the individual values within the table. The delta notation for the first two rows of the Wang table are below. Every delta cell must be related to every category in the table, therefore, delta notation is an aggregation of paths defining some content.

$$\delta\{Year.1991, Term.Winter, Mark.Assignments.Ass1\} = 85$$

$$\delta\{Year.1991, Term.Winter, Mark.Assignments.Ass2\} = 80$$

$$\delta\{Year.1991, Term.Winter, Mark.Assignments.Ass3\} = 75$$

$$\delta\{Year.1991, Term.Winter, Mark.Examinations.Midterm\} = 60$$

$$\delta\{Year.1991, Term.Winter, Mark.Examinations.Final\} = 75$$

$$\delta\{Year.1991, Term.Winter, Mark.Grade\} = 75$$

$$\delta\{Year.1991, Term.Spring, Mark.Assignments.Ass1\} = 80$$

$$\delta\{Year.1991, Term.Spring, Mark.Assignments.Ass2\} = 65$$

$$\delta\{Year.1991, Term.Spring, Mark.Assignments.Ass3\} = 75$$

$$\delta\{Year.1991, Term.Spring, Mark.Examinations.Midterm\} = 60$$

$$\delta\{Year.1991, Term.Spring, Mark.Examinations.Final\} = 70$$

$$\delta\{Year.1991, Term.Spring, Mark.Grade\} = 70$$

1.3 Tables

The Wang Notation Tool (WNT) consists of three main interactive tasks, all dependent on understanding tables, categories, and the relationships between all the cells. The first involves choosing categories within a table. The second corrects the categories. The third step verifies whether the final processing was done correctly. It is important to understand the nature of the relationships between cells of a tables because there is no set way to make a table; tables are different from one author to the next.

1.3.1 Categories as Trees

A category consists of a set of cells that are related to each other. Those relations can be represented as a tree. Figure 1 shows the tree representation of each category in Table 1. It is beneficial to describe categories using tree notation and tree operations as explained below.

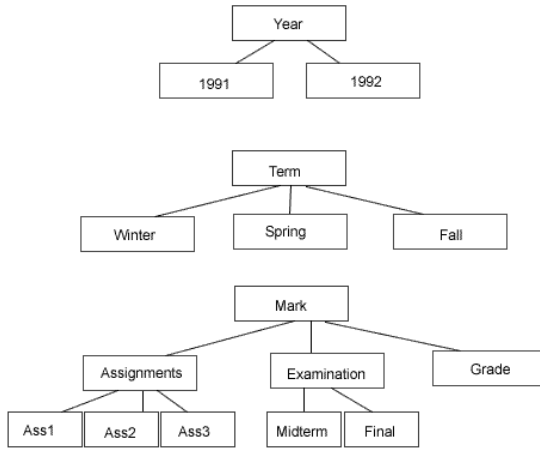


Figure 1: Tables as Trees

- Forest/Table: $F(T_1, T_2, \dots, T_i, \dots, T_n)$
A table can be described as a forest with n trees where n is the number of categories. Each tree represents a category.
- Tree/Category: $T_i(s, d)$
A category can be described as tree T_i with s levels, d nodes and root F .
- Root: $N_i(s_1, c_1, d_1)$
The root node of tree i is the only node

located on the top level which contains only one subtree.

- Level: $S_j(c)$
 c is the index of the subtree in level S_j ; sibling cells are distinguished from cousin cells below because they are associated with different subtrees.
- Node/Cell: $N_i(s_j, c_t, d_k)$
A node is located in tree T_i on the S_j level of that tree. c_t tells us which subtree of that level (determined from left to right) the node is located in and d_k (also determined from left to right) is the node number within that subtree.
- Leaf Node: $L_i(s_j, c_t, d_k)$
A leaf node can be located anywhere in the tree; even the root node can be a leaf node if a category consists of only a single cell.

1.3.2 Well-formed Tables

A well-formed table or category is one that WNT can convert perfectly to Wang notation. It is only when a table or category is not well-formed that user corrections are needed. The following are requirements for well-formed tables.

- 1) Every table must have n categories, where $n \geq 2$.
- 2) Every category must have a root (sometimes requiring the addition of virtual headers, Section 1.3.3).
- 3) Every delta cell must be specified by n paths, one through each category tree.
- 4) Category trees cannot contain subcategory trees that are identical (discussed further in Section 1.3.4).
- 5) Category cells only exist in the top-most rows and left-most columns of a table.

1.3.3 Virtual Headers

To correctly convert a category to Wang notation, the tree describing a category must be complete. Often, this is not the case. Table 2 shows a table with two categories. The first is the leftmost column and the second is found in the topmost rows. Neither category has a root, making them “rootless” trees. When a category is “rootless”, a virtual header must be added. Figure 2 shows one of the completed categories after the addition of a virtual header.

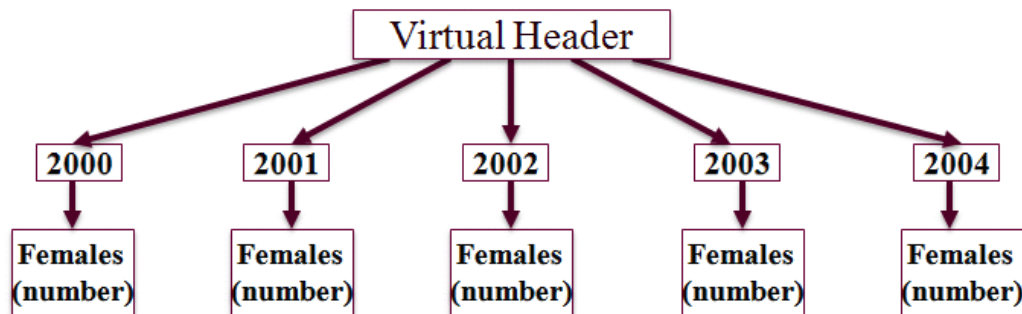


Figure 2: Virtual Header

Table 2: Number of Females with Degrees in Canada

	<u>2000</u>	<u>2001</u>	<u>2002</u>	<u>2003</u>	<u>2004</u>
	Females number				
Canada	103,326	105,207	111,027	118,467	124,830
Newfoundland and Labrador	1,773	1,755	1,749	1,830	1,989
Prince Edward Island	345	411	363	429	453
Nova Scotia	4,542	4,677	4,857	5,304	5,769
New Brunswick	2,424	2,460	2,670	2,811	3,000
Quebec	29,706	30,144	32,358	34,161	36,384
Ontario	39,297	39,972	41,982	45,042	47,862
Manitoba	3,114	3,183	3,366	3,612	3,897
Saskatchewan	3,408	3,378	3,393	3,486	3,528
Alberta	8,106	8,961	9,483	10,269	10,758
British Columbia	10,614	10,263	10,803	11,520	11,196

1.3.4 Unique Categories

To preserve consistency, every table should have unique categories determined by the same guidelines. In general, differentiating between category cells and delta cells is straightforward to a user with lexical knowledge. The category cells have to be picked such that every delta cell can be uniquely designated by category cells. However, to divide the category cells into separate categories is tricky. The categories should be picked according to the guidelines detailed in Section 1.3.2. A combination of any path from every category should lead to exactly one delta cell.

Figure 3 and Figure 4 show two tables that are the same structurally, but have different categories. Figure 3 has two categories. One is a single cell category *Pop.* and the other consists of the three leftmost columns (State, County, and Town, see Figure 5). If the three leftmost columns were each a separate category, the guidelines discussed in Section 1.3.2 will not be satisfied. It would not be possible to take a random path from each category and expect them to lead to a delta cell. For example, if the random paths were as follows: *State* > *New York*, *County* > *San Diego County*, and *Town* > *Troy*, there is no delta cell that meets those criteria. The only possible solution is for the three leftmost columns to be a single category. By contrast, if the three leftmost columns of Figure 4 was one category, the identical subcategories mean that it is possible to break

up the category further. Therefore, Figure 4 has four categories (Figure 6). Each of the three leftmost columns is a category on its own and *Pop.* is a single cell category. Any combination of random paths will lead to a delta cell.

STATE	COUNTY	TOWN	POP.
New York	Rensselaer	Troy	
		Brunswick	
	St. Lawrence	Potsdam	
		Canton	
California	San Diego County	Coronado	
		Del Mar	
	Los Angeles County	Malibu	
		Compton	

Figure 3: Two Unique Categories (Example 1)

STATE	YEAR	M/F	POP.
New York	2000	M	
		F	
	2001	M	
		F	
California	2000	M	
		F	
	2001	M	
		F	

Figure 4: Four Unique Categories (Example 2)

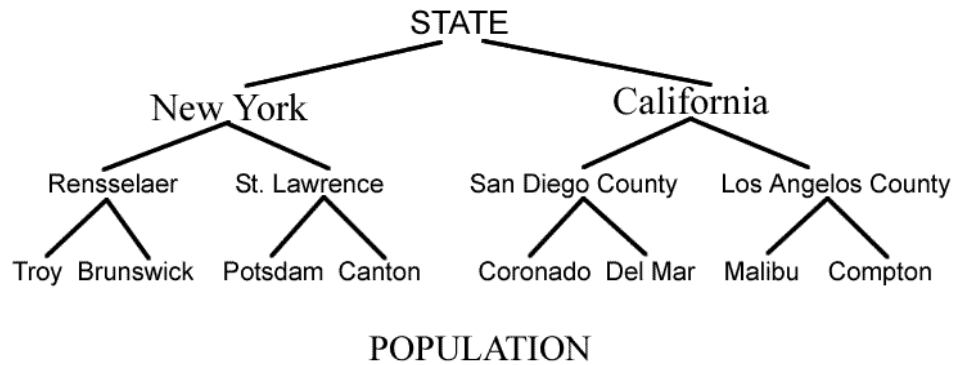


Figure 5: Category Trees for Example 1

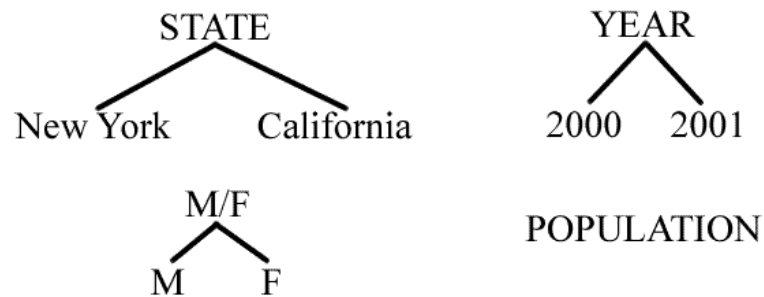


Figure 6: Category Trees for Example 2

1.3.5 Factors Affecting Table Processing Time

There are many factors that affect table processing time: some have a large impact and some not so much. All the factors that affect table processing time are user dependent, because the computer time for WNT is a small fraction of total processing time. Therefore, generating the category notation of a table is time consuming, whereas generating the delta notation and XML representation are virtually instantaneous because they require no user intervention. Considering all the factors together yields a prediction of how long it will take to process a table.

- 1) Confusion factor: The confusion factor of a table is inversely proportional to how well-formed the table is. It is a rough measure of the amount of time it takes a user to decide what the categories are, how they would delineate the categories, and what corrections need to be made. The confusion factor has the largest impact on the processing time of a table; a simple table with a low confusion factor can be processed quickly, but a complicated table that needs thought will take much longer.
- 2) Number of Categories: The number of categories plays a bigger role in the time taken to process a category than the size of categories. The user has to correct and approve each category individually, so with more categories, the user has to check, approve, and correct more categories.
- 3) Number of levels within categories: The number of levels is a factor in table processing time because the probability of WNT interpreting a category incorrectly increases with the number of levels, which results in more corrections and time invested by the user.
- 4) Category Size: Category size (number of category cells) is a relatively minor factor, because all it means is that a user spends a bit more time scrolling through and looking at a larger category. A large category does not mean a complicated category (which would be accounted for by the confusion factor) and in a significant number of tables, large categories are simple.

1.3.6 Foreign Tables

Foreign tables are tables where the words are nonsense words and the relations within the table must be determined using structural information only. This is an important topic to discuss because WNT does not rely on lexical information and if WNT were to be made fully automatic, it would have to be based on structural information only. To explore the possibility of a fully automatic WNT, the following question has to be answered: Is it possible for a user to differentiate between category and delta cells in a foreign table?

To answer this question a table (Table 3) with its corresponding foreign table (Table 4) is shown. We assume that in all tables, the left-most column always consists of category cells and topmost row always consists of category cells. The minimum dimension of a table is two and conventionally, those two dimensions are located on the left and top of the table. In Table 3, the left-most column headed by Country is the first category and *area sq. km.*, *population*, *yearly growth*, and *today* with a virtual header is the second category.

The key concept that gives insight into which cells are category cells and which cells are delta cells are merged cells. Figure 7 shows a table with two merged cells: *State* and *Information*. Figure 8 shows the same table, but the merged cells are split so that the table has m cells in every row and n cells in every columns. Splitting the cells results in repeated cells. In Table 3, *country*, *area sq. km.*, and *population* are merged cells. If Table 3 was to be represented by n cells in every column and m cells in every row (mxn), *country*, *area sq. km.*, and *population* would have to be repeated. A merged cell is always a category cell, because merged cells are a structural way of indicating that there is a connection between the merged cell and the cells directly adjacent to it.

Therefore, going from the bottom right hand corner to the top left hand corner, a category cell is encountered when it is either in the leftmost row or is adjacent to a merged cell. This is a intuitive and fairly accurate way to differentiate between category and delta cells in foreign tables. To implement a fully automatic WNT, many other factors and exceptions would have to be explored and accounted for.

STATE	INFORMATION		
	CAPITAL	AREA	POP.
NY			
CA			
AZ			
FL			

Figure 7: Merged Cells

STATE	INFO.	INFO.	INFO.
STATE	CAPITAL	AREA	POP.
NY			
CA			
AZ			
FL			

Figure 8: Split Cells

Table 3: Original Population & Area Table

country	area sq.km.	population	
		yearly growth	today
World	510,072,000	1.14%	6,563,077,034
China	9,596,960	0.59%	1,317,924,274
India	3,287,590	1.38%	1,103,054,870
United States of America	9,631,418	0.91%	299,828,179
Indonesia	1,919,440	1.41%	247,216,367

Table 4: Foreign Population & Area Table

country	area sq.km.	population	
		yearly growth	today
World	510,072,000	1.14%	6,563,077,034
China	9,596,960	0.59%	1,317,924,274
India	3,287,590	1.38%	1,103,054,870
United States of America	9,631,418	0.91%	299,828,179
Indonesia	1,919,440	1.41%	247,216,367

1.4 Organization of Thesis

Section 2 defines common terms in table processing literature and then discusses relevant literature. Section 3 is a detailed description of the Wang Notation Tool: overview, extraction of tables from HTML pages, the process of choosing categories, the corrections done by the user, the generation of the Wang category and delta notation, the generation of the XML representation, and the logging done in the background. Section 4 is a description of testing, including training the subject, preliminary testing, and final testing. Section 5 presents the results quantitatively and then discusses the reasons for those results. Section 6 discusses possible future work and Section 7 contains concluding remarks. References can be found after Section 7 and finally, there is an Appendix containing the tables used for evaluation and training as well as an example of Wang notation, XML representation, and the log. Also in the Appendix is the PowerPoint used for subject training.

2. Literature Review

There is a vast amount of literature concerning tables, most of which addresses how to find and parse a table within a scanned document. The Wang Notation Tool (WNT) interprets and attempts to understand web tables rather than merely detecting scanned tables. Some literature groups table and form processing together. However, tables and forms are inherently different. Tables have one author and are read by many people. Forms, on the other hand, have several authors and are read by one person. The sections below begin by defining the commonly used terms in table processing, then refer to some surveys detailing perspective on table processing, discuss the work of Wang, and finally discuss past literature to relate what has been done in table processing.

2.1 Detection, Extraction, Interpretation and Understanding

The words detection, extraction [3],[4],[5],[6],[7],[8], interpretation [7], and understanding [9] often come up in table processing literature. However, there is no set definition for these words; rather different people use these words in different senses. This section will attempt to solidify the definitions of the words detection, extraction, interpretation, and understanding with regard to table processing.

Detecting a table means locating the table and its cells and determining the size of tables and its cells in a given document. Detection is not relevant to WNT since WNT uses web tables. Web tables, written in HyperText Markup Language (HTML), have well defined <table> tags that require a simple search of the source code to detect. However, table detection is an important step when working with scanned image or ASCII tables with no markup language [5],[7],[10],[11]. Detection requires layout analysis to find the grid structure that is common to tables and further processing on rulings and white spaces to detect the locations of cells within a table.

Extracting a table is done after detection. Extraction goes beyond simple detection and separates the table from the rest of the document or image. Information is stored in a separate form: a separate file, separate image, or even a separate interpretation. In addition, extraction can mean separating and storing just the table's layout and structure or separating and storing both table structure and table content. The latter usually

involves OCR unless the source is initially electronic. The word *recognition* [8],[10],[11],[12],[13] is also widely used. Recognition consists of both detection and extraction; it is the input needed for interpretation.

Interpretation is the next level of table processing. Interpreting a table means obtaining the original information from a table and presenting the information within the table in a different way [3]. Interpretations of tables usually are layout independent. WNT interprets tables in multiple ways: as trees, XML, and Wang notation. Interpretation can also mean creating table models, among which Wang’s table model [2] is the most complete.

Understanding is the final step of table processing, one that has been explored much less than detection, extraction, and interpretation. Understanding a table means going beyond detection, extraction, interpretation and putting the information from one table into a greater context. Humans understand by gathering information from within a table and connecting that information to all the other information they know subconsciously. Understanding is a difficult task to accomplish with computers; however projects like TANGO [1] are ongoing attempts at understanding tables. Information from all tables processed within TANGO will be conglomerated into a comprehensive ontology that describes the relations within and between each table, thus enabling a computer to “understand” the tables.

2.2 Perspectives on Table Processing

To briefly summarize the perspectives on table processing and familiarize the reader with a high-level overview of past work in table processing, perspectives from some research surveys [9],[14] are cited.

Tables have physical and logical structure [14]. Physical structure allows table detection and describes the regions where parts of a table are located within a file or image. Logical structure is the goal of extracting, interpreting and understanding tables. Logical structure defines the types of regions within a table and their relationship to each other at several levels. The highest level of logical structure describes how the header cells are related to the body cells. The lowest level of logical structure consists of a

single cell. In the middle there are levels of logical structure describing groups of cells (cells in the same row or column) and arrangements of cells, such as cell topology, which are often described by a table grid (allowing easy indexing of tables).

Most table processing papers use table models [14] that statically or adaptively define the physical and logical structure of tables. Complete table models, such as Wang's [2], can be used to generate physical tables from given logical structure. Wang's model separates table structures into three parts. The first is an abstract indexing scheme that relates the header and the body cells, the second is a topology defining the placement and ordering of dimensions in various cells, and the third is formatting attributes such as fonts and separators. To be useful for the extraction and interpretation of tables, table models should be able to detect tables and then separate them into regions. Figure 9 from [14] details the different types of structures that have been used in the past.

Primitive Structures	Table-Specific Structures
Run lengths [15, 16, 50, 54, 95]	Table grid [5, 22, 27, 33, 45, 54, 86, 98]
Connected components [1, 33, 34, 47, 50, 54, 80, 95, 98]	Cells
Separators	Multi-line cells [27, 38, 44, 65]
Lines [15, 16, 46, 50, 72]	Spanning cells [24, 69, 86]
White space [22, 27, 36, 41, 49, 65, 86, 92, 98]	Cell Topology (usually rows and columns of cells [21, 27, 33, 38, 45, 65, 78, 85, 98])
Intersections	Table regions: boxhead, stub, and body [38, 43, 73, 96]
Of separators [4, 5, 15, 50, 78, 84, 87, 94]	Captions, titles, sources, footnotes and other text associated with tables [20, 69, 73, 86]
Of lines and text [6, 34, 97]	Tables (for table detection [14, 24, 36, 46, 48, 49, 54, 65, 78, 84, 91, 93])
Characters	Indexing structure
Provided in text or markup files [21, 24, 36, 47, 65, 73, 85]	Indexing relation for tables [21, 24, 85, 96]
From Optical Character Recognition [7, 49, 57, 69, 80, 86]	Entry structure in tables of contents [7, 9, 81, 82]
Text lines [46, 67, 79, 80]	
Other Symbols	
Arrow heads (to repeat cell values [6])	
X's (to cancel cells [4])	

Figure 9: Models of Table Structure

There are three sets of procedures used to make table models: observations, transformations, and inferences. *Observations* gather the information needed to recognize a table. Observations can be taken from physical structure (images, ASCII files), logical structure (description of tables), descriptive statistics (set of existing observations), or parameters associated with tables. Figure 10 from [14] lists the types of observations found in table recognition literature.

Physical Structure

Geometry
Area, height, & width
Aspect ratio (height: width)
Angle (e.g. of a line)
Skew estimation
 From skew of detected lines[53]
 From bounding boxes[46]
Docstrum: angle, distance between connected components[67]
Overlap of regions (e.g. table regions[14])
Region perimeter[34, 78]
Representative point
 Centroid[67]
 Top-left corner[94]
 Text baseline y-position[33]
Distance between points (e.g. between centroids[67])
Histograms (Projection Profiles)
 Projected image pixels[16, 63, 79]
 Projected bounding boxes[25, 32, 45]
 Boxes projected as symmetric triangles [98]
 Boxes projected as 'M' shapes of constant area[49]
 Weighted projections (e.g. by height[27, 98])
Texture
 Value transition count (cross-counts)[50, 86]
 Pixel density[13]
 Character density[21]

Parameters

Static or Adaptive
 Probability (e.g. for table detection[93])
 Thresholds (adaptive examples: [32, 46, 98])
 Tolerances (e.g. used to tolerate noise in X-Y cutting[13])
 Weights (e.g. for linear combinations[48])
Adaptive
 Line grammar (e.g. for table of contents[7])
 Regular expressions for cell contents[69, 78]
Encoded Domain Knowledge (Static)
 Word bigrams[41]
 Ontologies[85]

Logical Structure

Table structures (see Figure 7)
Edit distance[10]
 Deriving reg. expressions for strings[69, 78]
Cell block cohesion measures[43, 85, 91]
Graphs
 Line intersections[87]
 Form structure[12]
 Table indexing structure[22, 39]
Table Syntax (as grammars; see Figure 12)

Descriptive Statistics

Cardinality (counting)
Probability (e.g. computed from a sample)
Weighted Linear Combinations of Observations
 'Columnness' of a region[48]
 'Tableness' of a region[93]
Comparisons
 Difference (e.g. between heights[85])
 Derivative (e.g. of histograms[16, 79])
 Inner ('dot') product and cosine of vectors [80, 91, 96]
 Correlation (e.g. of text line spacings [49])
 Word uniqueness[91, 96]
Summary Statistics
 Range, median, mean
 Variance/standard deviation[46]
 Periodicity
 In histograms[79]
 In column, row structure[91]
 Line/string periodicity[73]

Figure 10: Observations in Table Recognition Literature

Transformations restructure observations to emphasize features of a data set to make the next set of observations easier or more reliable. WNT uses a tree transformation to represent categories and splits merged cells. Examples of the types of transformations used in past literature can be seen in Figure 11 from [14].

Physical Structure	Logical Structure
Image Binarization (e.g.[27, 79])	Merging/splitting of regions
Image compression	Cells[43]
Run-length encoding[95]	Tables[78, 96]
Block adjacency graph[97]	Splitting region at detected separators[54]
Image resampling	Graph/tree transformations
Subsampling[15, 34]	To correct structural errors[7]
Supersampling[67]	Join regions into a table region[76]
Quadtree[79]	Filtering
Hough transform[30] (e.g. for locating lines)	Small regions for noise reduction[52, 66, 80]
Affine transformations: rotation, shearing, translation	Textures, images and half-tones[80]
and scaling[77], (e.g. used for deskewing	Insertion of table lines[33, 54]
an image[46])	Produce boxes from line intersections [3, 87]
Interpolation to recover parts of characters intersected	Sorting and Indexing
by lines[97]	Sorting (e.g. boxes by geometric attributes[8])
Mathematical Morphology[30]	Indexing (e.g. of cells[22, 45])
RLSA (Run-length smoothing algorithm[95])	Translation
Dilations and closings	HTML to character matrix[24, 91]
In images[66, 6]	Map strings to regular expressions[69]
In text files[47]	Transform tokens of a single class to a uniform
For joining lines[97]	representation ([65, 69])
Thinning[45]	Encoding recognized form data[12, 97]
Edge detection[87, 94]	Indexing relation of a table[22]
Descriptive Statistics	
Histogram smoothing[79]	
Histogram thresholding	

Figure 11: Transformations in Table Recognition Literature

Lastly, *inferences* decide whether a document contains the physical and logical structure of a table model by generating and testing a hypothesis using one of three different techniques. Classifiers assign structure and determine relations between table models and data. Segmenters determine if it is possible for a type of table model structure to exist in the data. Parsers return graphs on structures according to table syntax (which are defined in table models). Figure 12 and Figure 13 from [14] show the classifiers, segmenters, and parsers that were used in the past.

Decision Tree

Single Dimension

- Thresholding (e.g. threshold a 'columnness' feature to locate columns[48])
- Priority of separators (e.g. table lines by thickness[22])
- Using area to classify noise vs. signal[15, 52, 66, 72]
- Character class (e.g. alphabetic, nonalphabetic, other[65])

Multiple Dimensions

- Connected components
- Defining[30]
- Classifying[33, 45, 46, 50]
- Document region classification[46, 50, 90]
- C4.5 decision tree induction[74] (for table detection[65])
- Word token sets[82]
- Table/non-table classification[91]
- Text orientation (vertical vs. horizontal[50])
- Chain code line segment type[53]

Nearest Neighbour

- k-nn (e.g. for defining clusters[67])
- Weighted k-nn (e.g. for table detection[91])

Neural Network

- Optical character recognition[80]
- Logo recognition[12]

Syntactic

- String matching (e.g. HTML cell types[91])
- Regular expressions (e.g. assigning types to text lines[36, 65, 86])
- Part of speech tagging (e.g. to classify roles of words in tables of contents[7])

Statistical

- Bayesian Classifier ('Naive Bayes')
- Table detection[91]
- Functional class of text block (e.g. author, title for table of contents [81])
- Bayesian network (e.g. assigning labels to regions in tables of contents[81])
- Probabilistic relaxation[77] (assigning labels to words in tables of contents[9])

Figure 12: Classifiers

SEGMENTERS

Clustering

- Connected components
 - Creation (e.g. for adjacent pixels[30], for adjacent word boxes[47])
 - Clustering connected components[46]
- Tables by content[96]
- K-means clustering (of projection histogram groups[98])
- Hierarchical clustering of regions by distance[36]
- Transitive closure (e.g. of a proximity relation [47])

Partitioning

- Using breadth first search (e.g. to segment columns[37])
- Using best-first search (e.g. to recover body, stub, and boxhead[43])
- Table detection
 - Using dynamic programming[36]
 - Using best-first search[93]
 - Using simplex[64] algorithm[14]
 - Using iterative decreasing step[14]

Recursive Partitioning

- X-Y cut[63]: alternating horizontal and vertical partitions at projection histogram minima[22]
- Modified X-Y cuts, using histogram minima and lines[13]
- Recursive line partitioning (e.g. by 'best' separating line[50], by line separator priority[22])

Exact Matching

- Splitting text columns into rows at blank lines[47, 65]

PARSERS

Hidden Markov Models

- Maximizing region adjacency probabilities[90]

Attributed Context-Free Grammars

- Tables in images (with parse control-flow directives: [18])
- Table form box structure[3, 8, 94]
- Form structure[19]
- Using input tokens with multiple types to parse tables of contents[82]
- For tables in HTML files[91]
- For page segmentation[52, 88]

Graph Grammars

- Table form cell composition[2]
- Table structure from word boxes[76]

Figure 13: Segmenters and Parsers

To determine if a table has been processed accurately, most table processing papers include a performance evaluation section [14] which details how fast a system is, what kinds of errors the system makes, and how a particular system compares to other systems. Performance evaluation is generally done by establishing a ground truth where the physical and logical structures of tables (determined by another table model) are encoded into a file. Documents with ground truth are separated into training and testing sets using one of three methods [14]. All documents are used to train and test a system in the *resubstitution* method. In the *leave-one-out* method, each document is tested once with all other documents used to train the system and finally, training and testing sets can be assigned *randomly*.

Some generalized paradigms for table processing can be found in [9]. Tables can be found everywhere, however the formal definition of “tabularity” is elusive because some forms of data share similarities with tables but are not actually tables. Past research has mainly been on the extraction of low-level geometric information from scanned images of tables with growing research on electronic tables. Recently, research has been done on table analysis and composition, which has furthered understanding of different aspects of tables.

To briefly describe a table: a table consists of a finite set of labels and corresponding to each label is a set called the domain of the label. A complete table is a set of functions from the labels to their corresponding domains where each function is an element of the table and each label is an element of its domain [9]. Forming an ontology (“a formal, explicit specification of a shared conceptualization”) of a table is a way to automatically understand a table. As per [9] understanding a table means having the ability to recover the label-value pairs from the representation of a given table. Formal definitions of tables also leads to generalizations.

There are various table models in literature, but the majority of the table models do not demonstrate table interpretation or understanding. Most table models are aimed at detecting and extracting a table. Low-level models make use of the rulings, white space, grids, and characters to find and extract a table [9]. Some models describe specific tables and some models are geared towards sets of similar tables. High-level models are

more useful for editing tables and describe both the physical and logical aspects of a table. Wang's table model is the most complete [9],[14]. Some applications of table processing are: converting similar large tables from an old form (usually typed) to a usable form (electronic) [3],[4],[7], mining data from large tables of different types [5],[8], making a database of individual data, interactively obtaining information from large tables, rendering a text table into an audio format, manipulating existing tables, and modifying tables to fit different displays. Commercially, low-end OCR systems, such as ScanSoft's Omni-Page, find table location and segmentation features for tables which have explicit grids. Companies like XML Cities capture table data, try to index data properly, and include features for validation and correction by humans, as does WNT.

There are three broad types of inputs into table processing systems: *ASCII files* (text, HTML and XML) [3],[5],[6],[7],[8],[11],[13] which consist solely of linguistic content and character-level spacing, *page-descriptor files* (Word, PDF, Latex, Postscript) with linguistic content and formatting, and *bitmap files* (images, scanned tables) [4],[12]. Tables in ASCII format are represented only by characters, white space, and carriage returns. In WNT the derived ASCII file includes various delimiters to separate rows and columns and to account for tables with merged cells. Mark-up language such as HTML can be misused and abused because of their flexibility, which is something we noticed during the work on WNT. It was necessary to separate <table> tags into tags that represent tables from tags that were used for layout purposes.

Different types of table require different table processing paradigms. Steps for all these paradigms can be found in [9]. The simplest paradigm is for simple tables, then compound tables with blank lines, compound tables without blank lines, tables with rulings, tables with simple headers, tables with nested headers, nested tables with row and column headers, and finally n-dimensional tables. Most of the tables in the dataset used for testing WNT have nested headers, oftentimes in the rows and columns, and tables that exceed two dimensions.

2.3 Wang's Table Model

Wang notation [2], (Section 1.2), ontologies and the semantic web [15] are very relevant to our work on WNT, which is an integral part of TANGO [1], (Section 1.1). Wang's table model [2], discussed in several table-processing papers [8],[11],[14], is a means to describe tables or to help a particular table processing system. Figure 14, from Wang's thesis, delineates the different parts of a row-column structured table.

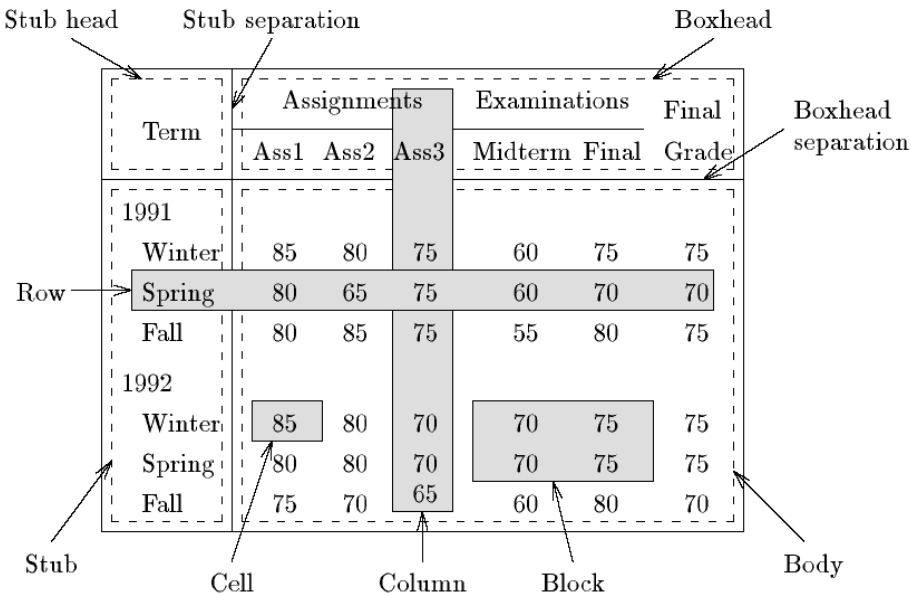


Figure 14: Regions Within a Table

Wang provides a set of guidelines for creating a table such that its underlying logical structure is obvious and tabular items are located and interpreted easily. The following three guidelines should be followed while deciding the content of a table: 1) The table should only contain necessary information, 2) Table should be presented as an explicit structure, and 3) The number of categories and subcategories should be reduced whenever possible. Once the content is decided, the following guidelines can be used to clearly show the logical structure of the table: 1) Place related items close together, 2) Avoid using two dimensions (using both column and row headings) whenever possible, 3) Place the most frequently referenced items at the top or left, 4) Vertically arrange items to be compared, and 5) Arrange items in a meaningful order. Finally, Wang offers sets of guidelines for the presentation of the table, which include separating and aligning

related parts of the table, spanning items, rounding numbers, and using appropriately sized fonts.

2.4 Techniques of Table Processing

The following papers, even if they don't seem directly related to this thesis, are nevertheless relevant because these papers discuss the structure of tables and the arrangement of columns and rows, which in addition to providing valuable insights, will be useful for automatically, rather than semi-automatically, determining the Wang notation for a table.

Silva et Al. [7] design, but not fully implement, an end-to-end system to automatically extract information from financial statements of companies to be used by various software agents. An extensive section on table-related research argues that table processing can be separated into five parts: location (detecting tables), segmentation (physical description of tables), functional analysis (classifying different tables areas), structural analysis (connecting category and content cells), and interpretation (understanding tables in context with each other). These five steps are followed in the design presented by Silva et al. (illustrated in Figure 15).

A document of any type is first converted to an ASCII document. The steps listed above are then implemented non-linearly (see Figure 15) to increase the confidence of all decisions made and allow the system to correct errors in the light of new information. Once an ASCII file is generated, the tables within that ASCII file are located and segmented into cells. These cells are then separated into two different types of cells (called content and data cells, equivalent to category and delta cells in WNT) and a relation between them is identified. Lastly, the results are interpreted and extracted to a database. Since WNT begins with a HTML table, WNT simply scans the source code to find tags defining tables and cells and then converts those tables to an ASCII file with specific delimiters. The segmentation of cells in WNT is assisted by the user, but the interpretation is done by the computer.

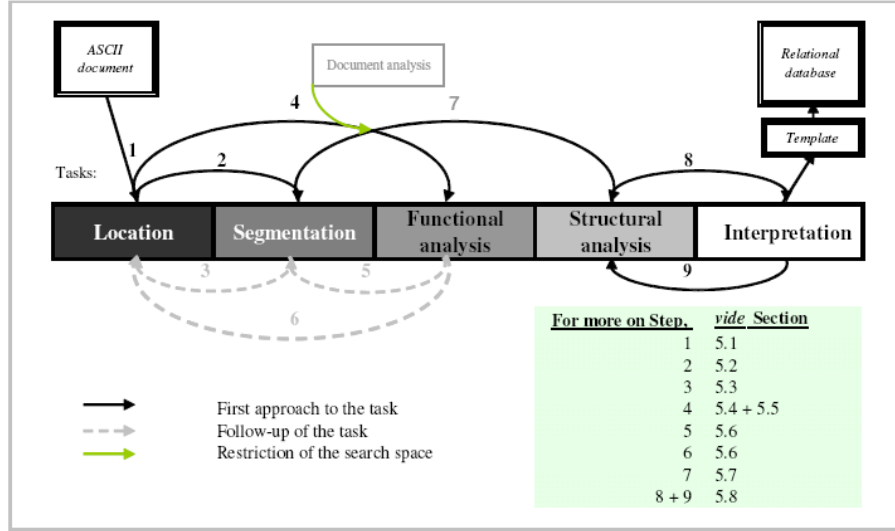


Figure 15: Flowchart of the design of Silvia et al.

Watanabe et al. [12] propose a system to recognize the layout structures of many kinds of well-formed (containing horizontal and vertical rulings) table-form document images. The recognition system does not detect a table within a document; rather the different parts of an already isolated table are extracted and used to build a knowledge-system of table structures. Classification trees are used to manage the relationships among different classes of layout systems. The recognition system has two modes: *layout knowledge acquisition* (table-form document images are distinguished according to classification tree and description trees are generated automatically) and *layout structure recognition* (individual fields are extracted and are classified by searching the classification tree and interpreting the structure description tree).

A knowledge-based method that uses binary trees represents the logical information within a table, including layout structures of table-form documents. A table is represented by two types of trees: global structure trees and local structure trees. The global structure tree describes relationships between blocks (sets of related cells) in the table while the local structure tree describes the relationships within the blocks. Classification trees contain information about sets of table-form documents that are physically similar and can, therefore, be identified by the same layout knowledge. Trees are used very differently in the recognition system of Watanabe et al. and WNT. Watanabe et al. use trees to describe the possible structures of a table whereas WNT uses

trees to describe the specific structure of a specific table. WNT also uses binary trees to represent the relationship between category cells.

Gatterbauer et al. [8], similar to WNT, focus their attention on web tables. A modification of the 2-D visual box used by browsers (visual box representation) is used to display pages rather than <table> tags and a tree-based representation (DOM trees) of web pages. The problem of extracting information from large-scale, domain independent sources is tackled by moving away from linguistic techniques to a “2-D pattern recognition problem using a variation of the CSS2 visual box model”. Previously, natural language processing techniques were used to extract information from web tables.

The information in web pages can be represented either by DOM trees or visual box representations. There are three types of nodes in a DOM tree: text nodes, element nodes, and edge nodes which define the relationship between text and element nodes. WNT describes the tables within web pages similarly to DOM trees (with category cells, delta cells, and a path describing the relationship between category and delta cells). A visual box representation makes use of Cascading Style Sheets (CSS), which govern the style or layout on all web pages associated with it.

Most web pages topologically form a frame in the visual box model. *Gatterbauer et al.* divide web tables into multiples types of topologies (Figure 16). The first task is to find the table location (identifying tables and their cells), second to recognize the table (identify spatial relationships between cells) and third to interpret the table (extract and save information in a format that retains relevant table information). Table extraction is done by finding all the frames (areas containing tables) in a given web page, then matching these frames with pre-defined tables and determining which 2D grids are semantically significant. The table is then transferred, following a set of rules, into a topological grid. Finally, an interpretation of the table using Wang’s table model is provided.

WNT aims to determine the number of categories (dimension of table) based on the structure of the table. *Gatterbauer et al.*, on the other hand, use lexical information to determine the dimensions which do not correspond with WNT dimensions. The

difficulty in using lexical information to identify categories arises when the lexical information is unknown or in a different language. The results from the method of Gatterbauer et al. are shown in Figure 17. 57% of tables are interpreted correctly, compared to 68% for WNT.

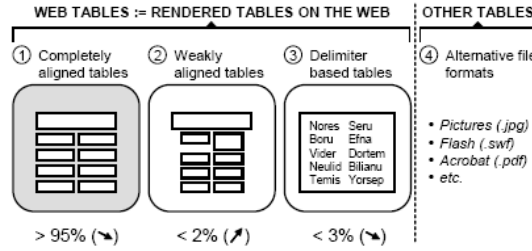


Figure 16: Table Topologies

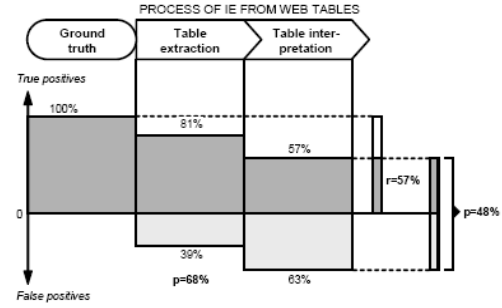


Figure 17: Results from [8]

Hu et al. [11] describe a way to recognize the structure of a table within a region that is already known to contain a table. Column segmentation is a key component of recognizing table structure. *Hu et al.* attempt improvements on previous work that segment columns by creating white-space profiles, or histograms, of each column of pixels or characters. The peaks and valleys in the histogram roughly indicate where a column began and ended.

Hu et al. apply hierarchical clustering to all the words in the detected table region to identify groupings (columns). These groupings are represented by binary trees (constructed bottom-up) where the root is the entire body, leafs are the words, and intermediate nodes are groupings at different levels. The binary trees describe the entire table and make no distinction between category and data cells. The binary trees in WNT, on the other hand, are used solely to describe the relationships between category cells. *Hu et al.* process tables with simple categories and thus, eliminate the need to ascertain in detail the relationship between the cells belonging to a category; oftentimes, there are a minimal number of category cells. WNT, on the other hand, primarily processes tables with numerous category cells and complicated structure.

After the columns are segmented, spatial and lexical information is used to differentiate category cells (also called headers) from content cells using the following information: 1) the header for each column is roughly aligned with the column and 2) the

hierarchical headers are centered over the columns they describe. Using these and other assumptions (row headers are always in the left-most column) headers are classified both on the left and on top. Finally, row segmentation is done using more heuristics.

A graph model (Directed Acyclic Graph, Figure 19), rather than a tree model (Figure 18), is utilized to describe tables. DAGs are more general than tree models because several parents can share the same children. DAGs can be split into two types of nodes: *leaf nodes* have no children and *composite nodes* have children. A tool called Daffy was developed to browse and edit table DAGs. Daffy can display and edit graphical mark-up, define new mark-up types, examine hierarchical structure, print and save PS page images, and run algorithm animation scripts to visualize the results of document analysis. Inputs can be images or text. WNT also has user interaction, but with trees instead of graphs.

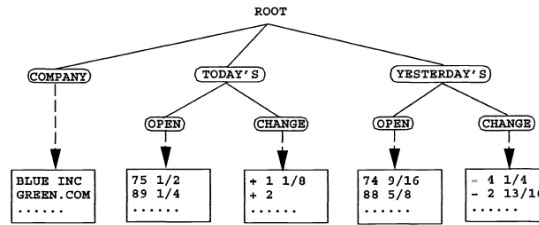


Figure 18: Example of a Tree Model

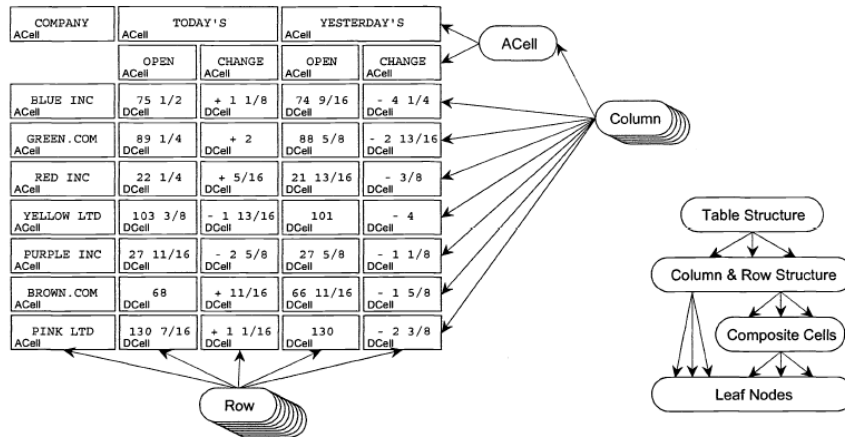


Figure 19: Example of a DAG

Pyreddy *et al.* [6] developed a system called TINTIN (Table INformation-based Text INquiry) that identifies tables and their component fields using structural information and then lets users query the fields. TINTIN uses heuristic methods to

extract structural elements and separate tables from text. The results are indexed and users can query the new database of indexed documents. Figure 20 shows the basic architecture for TINTIN.

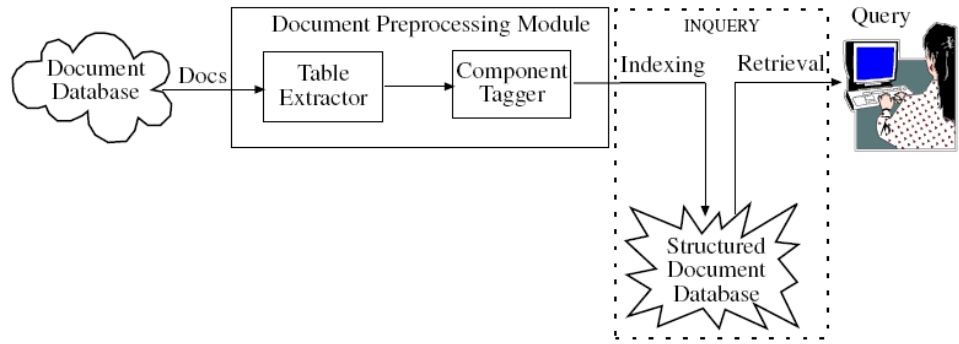


Figure 20: Architecture for TINTIN system

During pre-processing TINTIN extracts table data from plain text documents and tags the components of a table. Table extraction is done by looking for aligned white spaces. However, since tables are not always uniform, TINTIN makes use of a data structure called the Character Alignment Graph (CAG). The CAG is a histogram of the number of characters that appear at a certain location. The table structure is extracted from the CAG in the form of a text table. WNT also extracts a text table from HTML source code, but in a different manner.

*****	***	*****	
*****	*****	*****	*****
*****	*****	*****	*****
*****	*****	*****	*****
*****	***	*****	***
*****	**	*****	**
*****	**	*****	**
*****	**	*****	**
***	*****	*****	**

Figure 21: Star Table

The component tagger was difficult to implement because different people make tables differently. Primarily, syntactic heuristics were used (i.e. <table>, <caption> tags). Each character in the table was replaced with a star to

make a corresponding star table (Figure 21), which clearly shows which segments of stars belong to the header and which segments contain content (similar to the discussion of foreign tables presented in the Introduction). Using star tables, sets of heuristics were developed to classify each component. After the pre-processing is completed, the resulting table is indexed.

For retrieval, a system called INQUERY (“a probabilistic retrieval engine”) is used to obtain tables from structured documents. The user can type in a query to get a table

that hopefully answers their query. A matching compares what was typed with words present in the table; words that appear in captions of the structured documents get more weight. One of the goals of TANGO (Query By Table) is similar to the INQUERY system.

Rahgozar *et al.* [10] describe a bottom-up method of detecting table structures in documents by converting all documents to layout graphs (Figure 22) where boundary regions enclose the separate parts of a document and the arrows between parts of the document

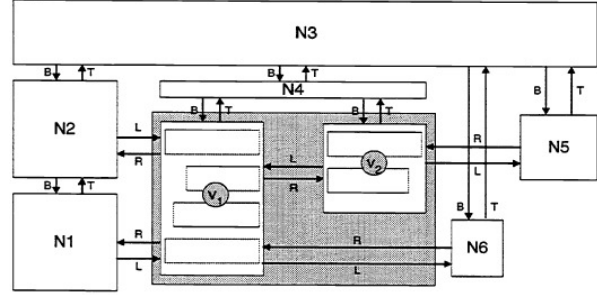


Figure 22: Layout Graphs

show how those boundary regions are related. Once this graph is obtained, it is rewritten using a set of rules that are based on apriori knowledge of documents. The rewritten graph gives a logical view of the documents and can be parsed to extract tables. Graphs represent complex multidimensional information, but are usually computationally taxing.

Rahgozar *et al.* propose a computationally efficient four-step method of graph rewriting to recognize table structures. *Segmentation* divides the documents into non-overlapping regions of text, images, line-drawings, and halftones. *Graph construction* transforms the segmented document into a graph with relations between the different types of regions. *Entity recognition* is used to label each section of the graph by its contents (C, W, L, TR, IR for character, word, line, text region, and image region respectively). Finally *graph rewriting* extracts the logical structure of the document from its layout graph.

Pinto *et al.* [5] use Conditional Random Fields (CRFs) to detect and extract tables from plain text government statistical reports with a 92% success rate. The CRF method uses both layout and content information to locate tables in plain-text documents and label each of the documents' constituent lines with tags (i.e., header, sub-header, data, separator).

Table extraction is broken into six overlapping problems: locate the table, identify the row positions and types, identify the column positions and types, segment the table into cells, tag the cells as data or headers, associate data cells with their corresponding headers. The method presented by Pinto et al. focuses on locating the table and identifying the row positions and types by employing a conditional probability Markov model to label lines and thus determine whether the lines are part of a table.

There are four major types of line labels. *Non-extraction labels* are lines where no information about table cells is found (nontable, blankline, separator). *Header labels* contain metadata for table cells and are related to lines below (title, superheader, tableheader, subheader, sectionheader). *Data row labels* mark rows containing content information (datarow, sectiondatarow). *Caption labels* mark rows that are found below or above the proper table but are still related to the table (tablefootnote, tablecaption).

The CRF and Hidden Markov Models (HMM) are compared. CRFs and HMMs are configured using the same set of features and are trained the same way on the same set of inputs. The feature sets associated with the CRFs and HMMs are white-space features, text features, and separator features. Each feature is represented by a binary value and a threshold is set to determine what each feature means in terms of table location and extraction. The CRF model has a higher rate of success than the HMMs.

Chandran et al. [4] present a simple method to convert paper tables into electronic tables. A table is extracted from a scanned document by performing binarization and de-skewing operations. The image is then scanned for horizontal and vertical lines and white streams. The table must contain the minimum number of lines needed to determine the boundaries of the table and one perfectly horizontal line before any skewed lines. Pre-processing is done by de-skewing the image using an affine transformation. Horizontal lines are detected by brute force, while intersection points with vertical lines are simultaneously identified. Missing lines are then inferred using white-stream profiles and finally, the cells in the table are labeled based on some very simple assumptions.

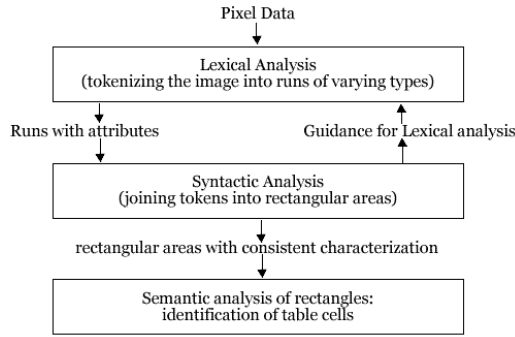


Figure 23: System of Green et al.

Green et al. [13] recognize the cells of a table in a two-dimensional binary document image by extending the methods of one-dimensional parsing. Grammars (production rules) are used, however since grammars are inherently one-dimensional, they have to be modified to account for the two-dimensional

nature of tables. One-dimensional grammars are modified by scanning both horizontal and vertical directions within the same production rule. Figure 23 shows the system diagram along with the purpose of each type of grammar (lexical, syntactic, semantic). Similar to the method of Chandran et. al. [4], the input consists of binary images or table that are horizontal, not skewed, and contain vertical rulings. Lexical and syntactic analysis further defines the different portions of a table. WNT also uses “grammars” or sets of rules to determine various characteristics of tables.

Kornfield et al. [3] detect and extract tabular data from ASCII files, in particular financial tables, using a modified version of the LR(k) parsing algorithm [16]. Since table construction is often sloppy, users are allowed to quickly correct defects in the source document (similar to WNT). Kornfield et al. optimized their system for commercial application, specifically EDGAR, which is an electronic means of filing financial reports in ASCII. The plain format increases distributability but hampers readability. The ASCII files are parsed to obtain the implicit hierarchical structure from which several derivative data streams are generated and put into readable templates, creating a basic interpretation of the table. The original ASCII file is very hard for financial experts to understand but once the information is put into a template file – the output – it is much easier to comprehend.

A parse tree shows the hierarchical structure of a table containing financial information. The parse tree is displayed with indentations, similar to the indented notation for each category used in WNT. Each node on the tree is called a unit and can either be a primitive unit (terminal node) or a compound unit (non-terminal node).

Parsing is done by “a single–stack non-backtrack parser analogous to an LR(k) parser” which is described by Kornfield et al. The parser processes 85% of the tables correctly; for the rest manual intervention is needed. The algorithm is constructed such that when an error is discovered it’s shown to the user in a human–readable way to ease the correction process. Most errors occur in the form of typos and arithmetic mistakes.

WNT detects, extracts, interprets, and readies tables for understanding. Detection is simpler in WNT than in most of the methods discussed above. The input to WNT consists of HTML pages containing tables. HTML pages are easily parsed to discover the location of tables. Extraction is also simple with HTML pages; HTML tags specify the types of cells within the table and lead to easy extraction. The majority of the methods discussed above had images of tables for their inputs; images make the detection and extraction problem much more challenging. Interpretation is not commonly investigated in table processing paradigms. The main goal of WNT is the complete interpretation of tables into a layout-independent form, which also leads to the understanding of tables.

3. Description of Interactive System

The Wang Notation Tool (WNT) was developed to convert a variety of physical web tables to abstract tables. The primary advantages of having a tool generate Wang notation rather than manually writing it are: speed and a lower propensity for error. It would take a person much longer to type the Wang notation for a table (particularly the delta notation) and their chances of error would be relatively high due to typos. WNT, on the other hand, generates notation relatively fast and there is no typing involved. WNT was also made to be robust, able to handle a variety of tables, both in shape and size. The end result is a tool that is mostly automatic and able to handle numerous types of tables.

3.1 Overview of System

There were many early versions of WNT; each successive version was more automatic and robust. The current version of WNT goes through many steps to determine the Wang notation and XML representation of a table. The first step is to acquire the table from an HTML page. This is done via a short program written in Java that searches for tables in HTML pages. The rest of WNT is executed in Matlab.

After the output of the Java program is recognized by Matlab, the table is displayed as a Graphical User Interface (GUI). Each cell in the table is clickable and the user can click the cells they believe to be category cells. Some intermediate category processing follows, where WNT tries to determine the correct category trees. The user then has a chance to either correct or approve those category trees. Processing the categories determines the Wang category notation, from which the Wang delta notation and XML representation are derived automatically.

The user has a chance to check if the relations within the table were determined correctly with the aid of another clickable GUI that changes the colors of related to the cell that was clicked. If the users finds the results to be incorrect, the GUI offers an option to process the table again. Throughout the entire process, a log is maintained in the background that records every button click by the user and the time between each step.

3.2 Early Versions of System

WNT evolved over a period of many months and as such, there were many earlier versions of the tool [17]. The first version of the tool was very limited and involved user intervention. It asked the user questions about the category and delta cells and had them type in responses. There was a limit on the number of levels within categories and no provision for user correction. For Table 1, 46 interventions were required to enter the categories and 36 interventions were required to enter the delta cells. All of these interventions were typed by the user.

The second version of WNT had clickable GUIs, which eliminated the need to type and thus, reduced the chance of typos. Version 2 did not have any provision for correction either. The second version generated delta notation automatically after asking the user some questions to convert a table to its symmetric form (a symmetric table is a table where all the category cells pertaining to a delta cell are in either the same row or same column as that delta cell [17]). For Table 1, roughly 50 interventions were needed to generate the category notation and none to generate the delta notation. All of these interventions were button clicks.

WNT, as it is now, is built upon version three. The number of interventions for choosing categories was reduced from 50 to 7 for Table 1, with the generation of delta notation remaining automatic. The XML representation, log, user correction, and verification were added to this version, making the program much more robust. User correction increased the number of interventions, but the number of interventions still stayed significantly below 50. In addition, prior methods of determining delta notation were simplified.

3.3 Detecting Tables in HTML pages

HTML has specific tags to denote tables, rows, and columns. Anything between the `<table>` and `</table>` tags is within a table. The `<tr>` and `</tr>` tags denote rows and the `<td>` and `</td>` tags denote columns. The words *colspan* and *rowspan* within the `<tr>` or `<td>` tags indicate merged columns or rows. Using this information, a Java program was written to find tables within an HTML page by parsing an HTML file and

looking for the <table>, <td>, and <tr> tags. The Java program is interactive because some HTML pages use the table tags for layout purposes, rather than to display a table. There is also provision to enter the table title, caption, and citation of the table. This information is later recorded in the XML representation.

Once the Java program finds a table, the table is saved as an ASCII file with specific delimiters, that will later guide the Matlab routines of WNT to recreate the original table as a Matlab array. Figure 24 shows the ASCII representation of Table 1. The long row of stars indicates the beginning and end of a table. Five stars are placed at the end of each row and two stars are placed between each column. The words *rowspan* and *colspan* indicate how many rows or columns are merged.

```
*****
Year [rowspan=3] ** Term [rowspan=3] ** Mark [colspan=6] *****
Assignments [colspan=3] ** Examinations [colspan=2] ** Grade [rowspan=2] *****
Ass1 ** Ass2 ** Ass3 ** Midterm ** Final *****
1991 [rowspan=3] ** Winter ** 85 ** 80 ** 75 ** 60 ** 75 ** 75 *****
Spring ** 80 ** 65 ** 75 ** 60 ** 70 ** 70 *****
Fall ** 80 ** 85 ** 75 ** 55 ** 80 ** 75 *****
1992 [rowspan=3] ** Winter ** 85 ** 80 ** 70 ** 70 ** 75 ** 75 *****
Spring ** 80 ** 80 ** 70 ** 70 ** 75 ** 75 *****
Fall ** 75 ** 70 ** 65 ** 60 ** 80 ** 70 *****
*****
*****
*****
```

Figure 24: ASCII Version of Wang Table

3.4 Generating Category Notation

Generating the category notation is the most significant part of WNT because it requires user intervention and the delta notation and XML representation stem directly from the category notation. The category notation records all the cells within a table that are category cells and the relationships between those cells. It is not necessary for category cells to be related lexically; instead, they must be related structurally. For example, in Table 2, one of the categories consists of the years and the words 'Female' and 'number'. The years are not related to 'female' and 'number' lexically, but in Table 2, they are related structurally and therefore, part of the same category.

3.4.1 Interactive Category Construction

The first step for generating category notation is to display the original table as an interactive GUI in Matlab, using the ASCII representation of the table. Matlab displays the original table as a $m \times n$ table where every column has m rows and every row has n columns. This means that all merged cells are split and repeated (Figure 25).

Year	Term	Mark	Mark	Mark	Mark	Mark	Mark
Year	Term	Assignments	Assignments	Assignments	Examinations	Examinations	Grade
Year	Term	Ass1	Ass2	Ass3	Midterm	Final	Grade
1991	Winter	85	80	75	60	75	75
1991	Spring	80	65	75	60	70	70
1991	Fall	80	85	75	55	80	75
1992	Winter	85	80	70	70	75	75
1992	Spring	80	80	70	70	75	75
1992	Spring	80	80	70	70	75	75
1992	Fall	75	70	65	60	80	70

Figure 25: Wang Table as Displayed in Matlab

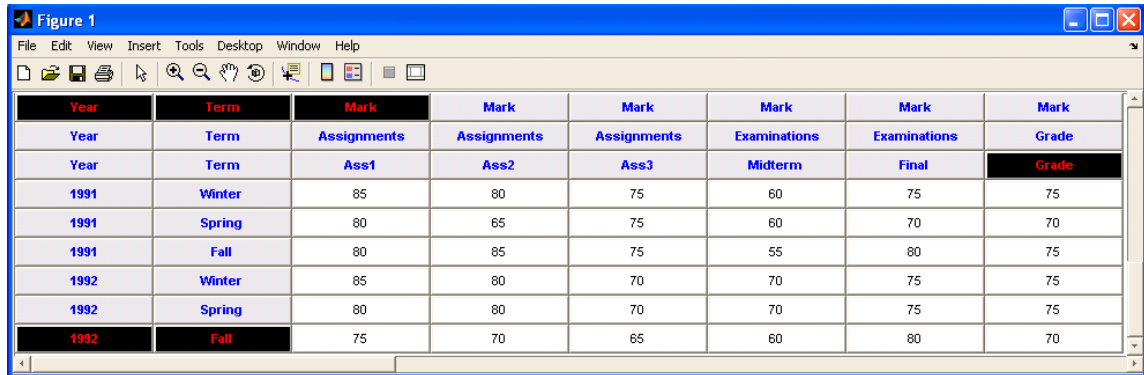
Next, the user indicates which cells are category cells and which categories they belong to. To reduce the number of interventions, it is assumed that all the cells pertaining to any one category falls within a specific rectangle within the table, as illustrated in Figure 26. This assumption has held for every table tested thus far.

Year	Term	Mark					
		Assignments			Examinations		Grade
		Ass1	Ass2	Ass3	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

Figure 26: Wang Table with Marked Categories

For rectangular categories, it is only necessary to click the top leftmost cell and the bottom rightmost cell to mark a category (Figure 27). The cells clicked are marked in black and the gray/blue cells in between are interpolated by WNT. If a cells is selected by mistake, it can be unselected by clicking on it again. While selecting cells, the user

has to keep in mind the points discussed in Section 1.3.4, for correct category construction. If a category consists of a single cell (a rare case), the GUI shown in Figure 28 can be used to enter a single-cell category. Once the selection of categories is deemed correct and completed, ‘DONE entering categories’ is clicked in the GUI shown in Figure 28 to move on to the next step.



Year	Term	Mark	Mark	Mark	Mark	Mark	Mark
Year	Term	Assignments	Assignments	Assignments	Examinations	Examinations	Grade
Year	Term	Ass1	Ass2	Ass3	Midterm	Final	Grade
1991	Winter	85	80	75	60	75	75
1991	Spring	80	65	75	60	70	70
1991	Fall	80	85	75	55	80	75
1992	Winter	85	80	70	70	75	75
1992	Spring	80	80	70	70	75	75
1992	Fall	75	70	65	60	80	70

Figure 27: Wang Table After User Marks Categories

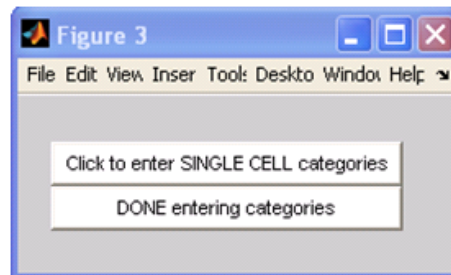


Figure 28: Control GUI for Selection of Categories

3.4.2 Intermediate Category Processing

At this point, WNT only knows which cells belong to which categories; the relationships between those cells are unknown. Therefore, WNT does some intermediate category processing to determine the relationships between the cells of each category. These intermediate relationships are displayed in the next step to be corrected and approved.

The intermediate processing cleans up each of the categories by deleting repeated values, blank cells, and nonsense cells (with a few minor exceptions determined by extensive testing of tables). WNT then creates trees describing the relationships within each category. These trees (Figure 29) are represented as indented notation (Figure 30).

In addition to the indented notation, a corresponding Table of Contents representation (Figure 31) is also determined.



Figure 29: Category Tree

'Mark'	"	"	Mark	1	0	0
"	'Assignments'	"	Assignments	1	1	0
"	"	'Ass1'	Ass1	1	1	1
"	"	'Ass2'	Ass2	1	1	2
"	"	'Ass3'	Ass3	1	1	3
"	'Examinations'	"	Examinations	1	2	0
"	"	'Midterm'	Midterm	1	2	1
"	"	'Final'	Final	1	2	2
"	'Grade'	"	Grade	1	3	0

Figure 30: Indented Notation

Figure 31: Table of Contents Representation

Included in the intermediate processing is the special case of *virtual headers*. There is no way to conclusively say that a category needs a virtual header by looking at the category trees. Section 1.3.3 discusses virtual headers using Table 2 as an example. The intermediate processing output for both the categories of Table 2 are shown below (both categories require correction). It is obvious that the category on the left requires a virtual header, but it is not obvious that the category on the right also requires a virtual header.

2000			Canada	
	Female			Newfoundland and Labrador
		Number		Prince Edward Island
2001				Nova Scotia
		Number		New Brunswick
2002				Quebec
		Number		Ontario
2003				Manitoba
		Number		Saskatchewan
2004				Alberta
		number		British Columbia

The category on the right requires a virtual header because there are delta cells associated directly with the entry *Canada*. In section 1.3.2, it was established that delta cells are specified by *paths* through category trees. Canada, as shown above, is a root and therefore, not a path. For this reason, the category on the right requires a virtual header with Canada and the provinces as its children. However, if after the indented notation is determined, the first column has more than one entry (as in the category on the left), a virtual header must be added. This is done automatically by WNT to save time.

3.4.3 Error-Correction by User

Due to the variety of tables found on the web, there is no guarantee that the intermediate processing by WNT will be correct. To make WNT more robust, error-correction must be an integral part of the process. If the categories were chosen correctly, it is almost always possible to correct the Wang notation with the error correction GUI. Notation is not generated when the user creates an invalid indented table (i.e., more than one entry per row).

The error-correction GUI appears on-screen after the intermediate indented table for each category is determined (Figure 32). To correct the relationships within a category, the incorrect indented notation (tree) for that category is corrected. The user corrects and approves each category separately (Figure 33). The error-correction GUI has enough options for every possible change to be executed, although some changes require multiple actions. A list of the functions of the buttons in the error-correction GUI follows:

- *Undo Last*: Reverts to the indented notation before the last correction was made.
- *Add Row*: A blank row is added above the row clicked.
- *Add Column*: A blank column is added to the left of the column clicked.
- *Delete Row*: Entire row containing the cell clicked is deleted.
- *Delete Column*: Entire column containing the cell clicked is deleted.
- *Clear Cell*: Clicked cell is cleared.

- *Rename cell*: Clicked cell can be renamed (pushbutton becomes a textbox to type a new name).
- *Add Virtual Header*: A root is added to the indented notation with a textbox in the root spot to be renamed.
- *Notation is Correct*: Clicked when the indented notation is deemed correct.

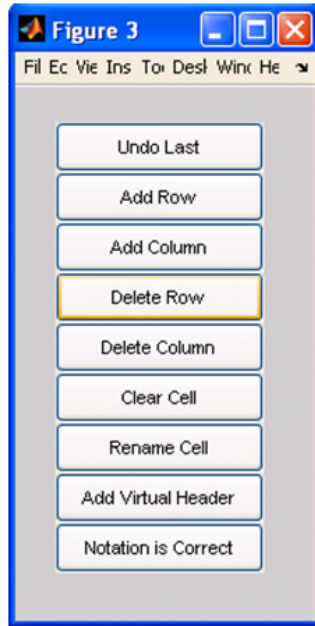


Figure 32: Error Correction GUI

Mark		
	Assignments	
		Ass1
		Ass2
		Ass3
	Examinations	
		Midterm
		Final
	Grade	

Figure 33: Indented Notation as seen in Matlab

3.4.4 Determining Final Category Notation

Wang category notation has a specific order for the keywords and different types of parenthesis that show the relationships within trees. The order of the keywords and the parentheses can be determined by a pre-order traversal (or depth-first traversal) of the category trees. To simplify the implementation of pre-order traversal of trees, the general trees are converted to binary trees. Converting general trees to binary trees preserves all relations, but the nodes in binary trees have, at most, two children, which makes the pre-order traversal algorithm simpler [18].

The leftson in a binary tree is the firstson of the father in the general tree. The rightson in a binary tree is the rightsibling of the preceding son in the general tree. The order of the nodes does not matter, only the pointers. Each node has three pointers. The

first pointer is to the father of the current node, the second pointer is to the leftson of the current node, and the third pointer is to the rightson of the current node. The binary tree is represented in a structure array with fields *nodename* and *pointers*. A function was written to convert a table of contents representation (obtained directly from indented notation) to a binary tree. Figure 34 is an example of a general tree. Figure 35 is the equivalent binary tree with the node numbers, pointers, and Wang symbols shown.

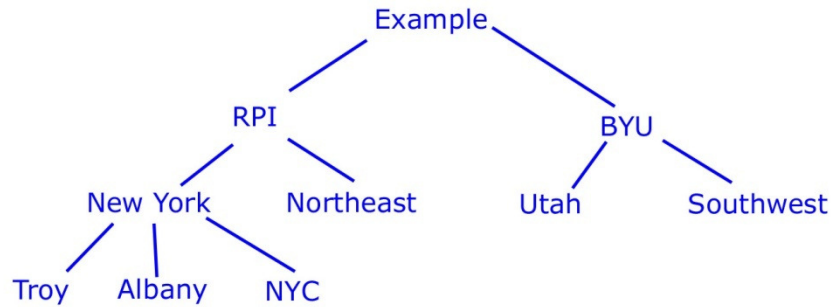


Figure 34: General Nonsense Tree

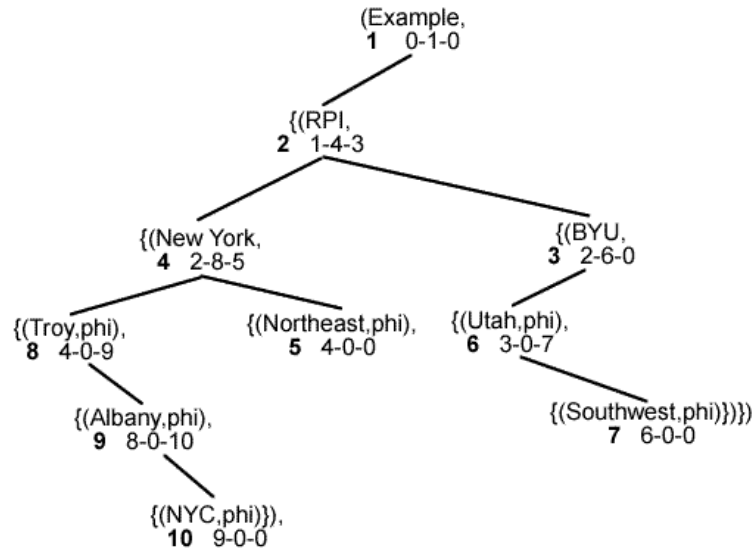


Figure 35: Equivalent Binary Tree with Pointers

Once the binary tree for each category is determined, a recursive function traverses the trees depth-first to determine the order of the keywords for the Wang category notation. In addition to the depth-first traversal of the keywords, the category notation contains delimiters such as parentheses, curly parentheses, and commas that define the

relation between the cells of a category. Rules were developed for correct insertion of all delimiters.

3.5 Generating Delta Notation

Delta notation describes how a particular cell is related to the category cells by listing the delta cells along with a path from every category describing those cells. In a well-formed table, there is exactly one delta cell associated with every possible combination of paths from all the category trees. Generating the delta notation starts by fusing all categories into a single tree describing the entire table. Complete indented notation and a corresponding table of contents for the whole table are also generated.

For each delta cell, the program searches the original table for all the leaf cells in the same row and column as the delta cell. For example, if a table has three categories, there should be three leaf cells that are in the same row or column as every delta cell in that table. Tables with multiple leaf cells of the same name are accommodated. The paths that correspond to every delta cell are determined by working backwards in the fused table of contents – starting with the leaf cells and working up to the root. Finally, all the paths are associated with the right delimiters to generate delta notation (Section 1.2).

3.6 Generating XML Representation

In the next steps of TANGO (creating mini-ontologies, discovering inter-ontology mappings, and merging ontologies), tables are represented in XML, a legible mark-up language. Therefore, it was necessary to generate an XML representation of every table in addition to Wang notation. The XML representation of tables captures all the information contained in Wang notation (category cells, delta cells and their relations), but also has provisions for recording the table title, caption, citation, and an identifying number. More importantly, it has provisions to record annotations such as footnotes and augmentations.

3.6.1 Ontology to Describe Tables

The XML representation stems from an ontology developed by the contributors to TANGO. This ontology describes the structure of any table (Figure 36). It has four schemas (table, categoryparentnode, datacell, and augmentation) which are discussed in Section 3.6.2.

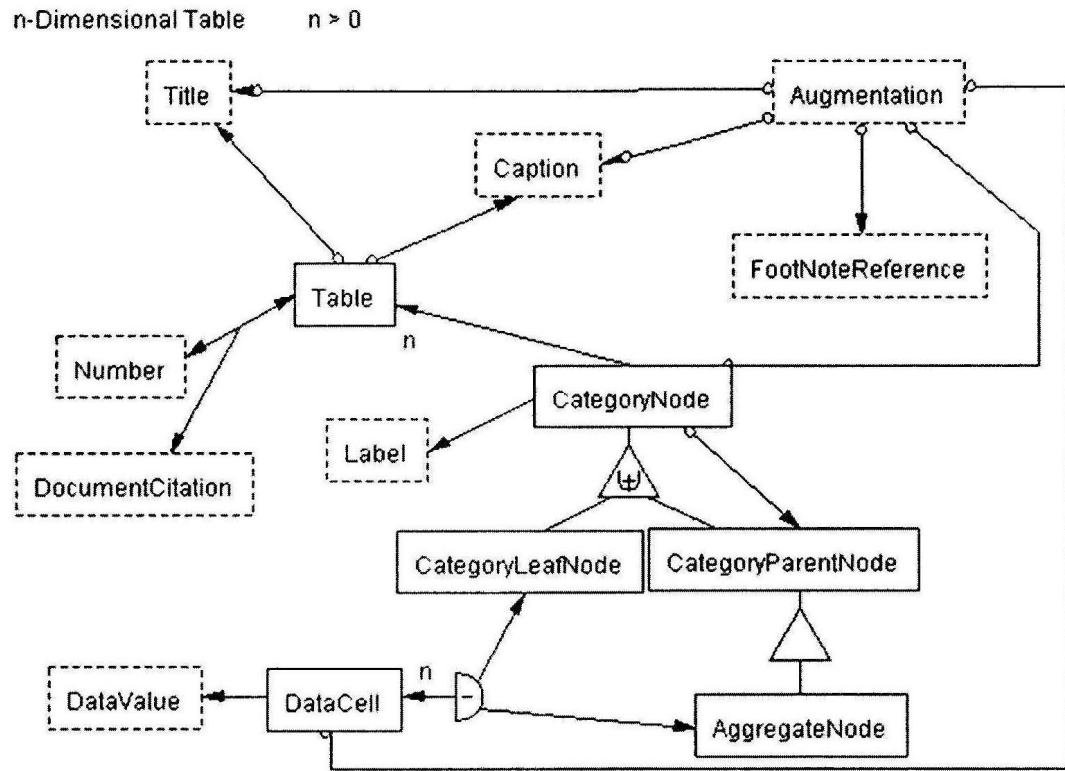


Figure 36: Ontology of a General Table

3.6.2 XML Schemas

There are four XML schemas with their own scheme trees, that set the guidelines for the XML representation of tables. All the schemas are derived directly from the ontology that describes a general table (Figure 36). The first XML scheme tree, called Table, includes the Table, Number, Document Citation, Title, Caption, and CategoryNode boxes. This schema provides basic information about the table (title, caption, citation) and lists the category nodes with their labels. Each non-lexical element (solid box) is given an OID (Object Identifier). Label is a lexical element (dashed box) connected to each category node.

The second XML scheme tree is CategoryParentNodes. This contains the CategoryNode, CategoryLeafNode, and CategoryParentNode boxes. This schema describes the tree structure of the table, similar to Wang's category notation. The CategoryParentNode's are treated as fathers and all their children are recorded. Some category nodes are both children and fathers. The category leaf nodes are never fathers, only children.

The third XML scheme tree is DataCells. This schema contains the DataCell, DataValue, CategoryLeafNode, CategoryParentNode, CategoryNode and Aggregate Node boxes. This schema describes how each data cell is related to the table by stating the leaf nodes that correspond to a data cell, similar to Wang's delta notation. It also has a provision for distinguishing aggregate nodes. The final scheme tree is for Augmentations. This includes the Augmentation, FootNoteReference, and the box being augmented. An augmentation with no FootNoteReference means that the augmentation is an annotation. Currently, the fourth schema is not represented in WNT.

3.6.3 XML Generation in Matlab

With all the schemas acting as guidelines for the XML representation, generating the XML is straightforward and automatic with the Matlab XMLToolbox. Using the table of contents for the entire table, every category and delta cell was assigned an OID. For each schema, a corresponding structure was generated in Matlab and then passed through the XML toolbox to convert it to XML. An example of the Table schema for Table 1 is shown below.

```
x.Table.ATTRIBUTE.Title = "Wang Table"
x.Table.ATTRIBUTE.Caption = "Students Grades"
x.Table.ATTRIBUTE.TableOID = 'Table2';
x.Table.ATTRIBUTE.Number = '2';
x.Table.ATTRIBUTE.Citation = 'Wang's PhD Thesis';
x.Table.CategoryNodes.CategoryNode(1).ATTRIBUTE.CategoryNodeOID = 'C1';
x.Table.CategoryNodes.CategoryNode(1).ATTRIBUTE.Label = 'Year';
x.Table.CategoryNodes.CategoryNode(2).ATTRIBUTE.CategoryNodeOID = 'C11';
x.Table.CategoryNodes.CategoryNode(2).ATTRIBUTE.Label = '1991';
x.Table.CategoryNodes.CategoryNode(3).ATTRIBUTE.CategoryNodeOID = 'C12';
x.Table.CategoryNodes.CategoryNode(3).ATTRIBUTE.Label = '1992';
xmlstr_Table = xml_formatany(x)
```

Figure 37: Matlab Structure for XML

```

<root xml_tb_version="3.1">
  <Table Title="Wang Table" Caption="Students Grades" TableOID="Table2"
Number="2" Citation="Wang's PhD Thesis">
    <CategoryNodes>
      <CategoryNode CategoryNodeOID="C1" Label="Year">
      </CategoryNode>
      <CategoryNode CategoryNodeOID="C11" Label="1991">
      </CategoryNode>
      <CategoryNode CategoryNodeOID="C12" Label="1992">
      </CategoryNode>
    </CategoryNodes>
  </Table>
</root>

```

Figure 38: XML for Table Schema

To generate the above automatically, especially for the CategoryParentNode schema that determined the trees for the entire table, some guidelines were developed to determine when a node is a parent node and what its children are. In the table of contents, a node (x,y) is a parent node if and only if the node(x,y) $\sim = 0$ AND if node(x,y+1) = 0 AND node(x+1,y+1) $\sim = 0$. The children of this parent node are nodes for which length(parentnode)+1 AND Child(1,1:length(parentnode)) = parentnode.

3.7 Verifying Results with a GUI

A method for verifying the output of WNT was devised to make WNT more robust. Directly verifying the Wang notation or XML representation is time-consuming and difficult, therefore, a visual method of verification was implemented. Complete verification requires access to the original table, therefore a GUI containing the original table pops up after all the relationships between cells are established and all the processing is done. The user can then click on any cell any number of times to verify any cell-to-cell relationship.

The cell clicked by the user turns blue. If the cell clicked was a delta cell, all the category cells corresponding to it turn red (Figure 39 & Figure 40). If the cell clicked was a category cell, all the delta cells corresponding to that category cell turn green and all the other category cells in the same category as the cell clicked, turn red (Figure 41 & Figure 42).

The table of contents for the entire table is used to decide the color of each cell depending on what was clicked. Therefore, if there is a mistake in the table of contents, the verifying GUI will “light-up” cells incorrectly, in which case the user can process the table again.

Year	Term	Mark	Mark	Mark	Mark	Mark	Mark
Year	Term	Assignments	Assignments	Assignments	Examinations	Examinations	Grade
Year	Term	Ass1	Ass2	Ass3	Midterm	Final	Grade
1991	Winter	85	80	75	60	75	75
1991	Spring	80	65	75	60	70	70
1991	Fall	80	85	75	55	80	75
1992	Winter	85	80	70	70	75	75
1992	Spring	80	80	70	70	75	75
1992	Fall	75	70	65	60	80	70

Figure 39: Verifying Delta Cell (1)

Year	Term	Mark	Mark	Mark	Mark	Mark	Mark
Year	Term	Assignments	Assignments	Assignments	Examinations	Examinations	Grade
Year	Term	Ass1	Ass2	Ass3	Midterm	Final	Grade
1991	Winter	85	80	75	60	75	75
1991	Spring	80	65	75	60	70	70
1991	Fall	80	85	75	55	80	75
1992	Winter	85	80	70	70	75	75
1992	Spring	80	80	70	70	75	75
1992	Fall	75	70	65	60	80	70

Figure 40: Verifying Delta Cell (2)

Year	Term	Mark	Mark	Mark	Mark	Mark	Mark
Year	Term	Assignments	Assignments	Assignments	Examinations	Examinations	Grade
Year	Term	Ass1	Ass2	Ass3	Midterm	Final	Grade
1991	Winter	85	80	75	60	75	75
1991	Spring	80	65	75	60	70	70
1991	Fall	80	85	75	55	80	75
1992	Winter	85	80	70	70	75	75
1992	Spring	80	80	70	70	75	75
1992	Fall	75	70	65	60	80	70

Figure 41: Verifying Category Cell (1)

Year	Term	Mark	Mark	Mark	Mark	Mark	Mark
Year	Term	Assignments	Assignments	Assignments	Examinations	Examinations	Grade
Year	Term	Ass1	Ass2	Ass3	Midterm	Final	Grade
1991	Winter	85	80	75	60	75	75
1991	Spring	80	65	75	60	70	70
1991	Fall	80	85	75	55	80	75
1992	Winter	85	80	70	70	75	75
1992	Spring	80	80	70	70	75	75
1992	Fall	75	70	65	60	80	70

Figure 42: Verifying Category Cell (2)

3.8 System Logging

Evaluating WNT requires recreation and times of user attempts. Such an evaluation (Section 5) requires a detailed log that records time and button clicks. There are two types of time: user time and computer time. User time is the time a user spends selecting categories, correcting them, and verifying them. Computer time is the time taken by the computer for processing. User time accounts for most of the table processing time. Every button click while selecting or unselecting categories and making corrections is recorded. An example of the log is shown in Appendix E.

3.9 Matlab and WNT

Matlab was chosen because it was the language the author of this thesis is most familiar with. Matlab was well-suited to WNT because tables can be easily represented as arrays (all tables are $m \times n$ arrays). Creating interactive GUIs is simpler with Matlab than with other programs and Matlab can interact easily with HTML pages to display the original tables, text files for the ASCII representation, and saving Wang notation and XML, and Excel to save logs for evaluation. Matlab, however, is not ideal for WNT because it has limited facilities for handling strings.

To compensate, a corresponding array of numbers, instead of strings, was generated for every table, indented notation, and table of contents. Every empty cell was given the value of 0 and every non-empty cell was assigned an integer. For the indented notation, the integer was always 1. For the table of contents, the integers reflected the level and

position of the cell. For a table, every delta cell was given the value of 0 and every category cell was labeled by number. WNT derives its layout independent representations based on the structure of the tables, rather than the content, of the tables, so an array of numbers was enough to determine relationships. It was also much easier to search through arrays of numbers than through arrays of strings. Overall, the advantages of Matlab outweighed the disadvantages and WNT was successfully created.

3.10 Summary

WNT is a complete system for converting web tables to layout-independent form. User interaction allows WNT to accommodate several types of tables successfully. However, there are still improvements to be made. The process of converting an HTML page to an ASCII table is not thorough and errors in this process have to be fixed manually. Further user interaction can be implemented to rectify ASCII tables. WNT fails completely in some instances because the indented table sent to the final processing stage is incorrect. This occurs often if the user makes many corrections. The indented table could be tested after the error-correction process and a warning issued if the indented table is invalid. Figure 43 shows a flowchart illustrating the system.

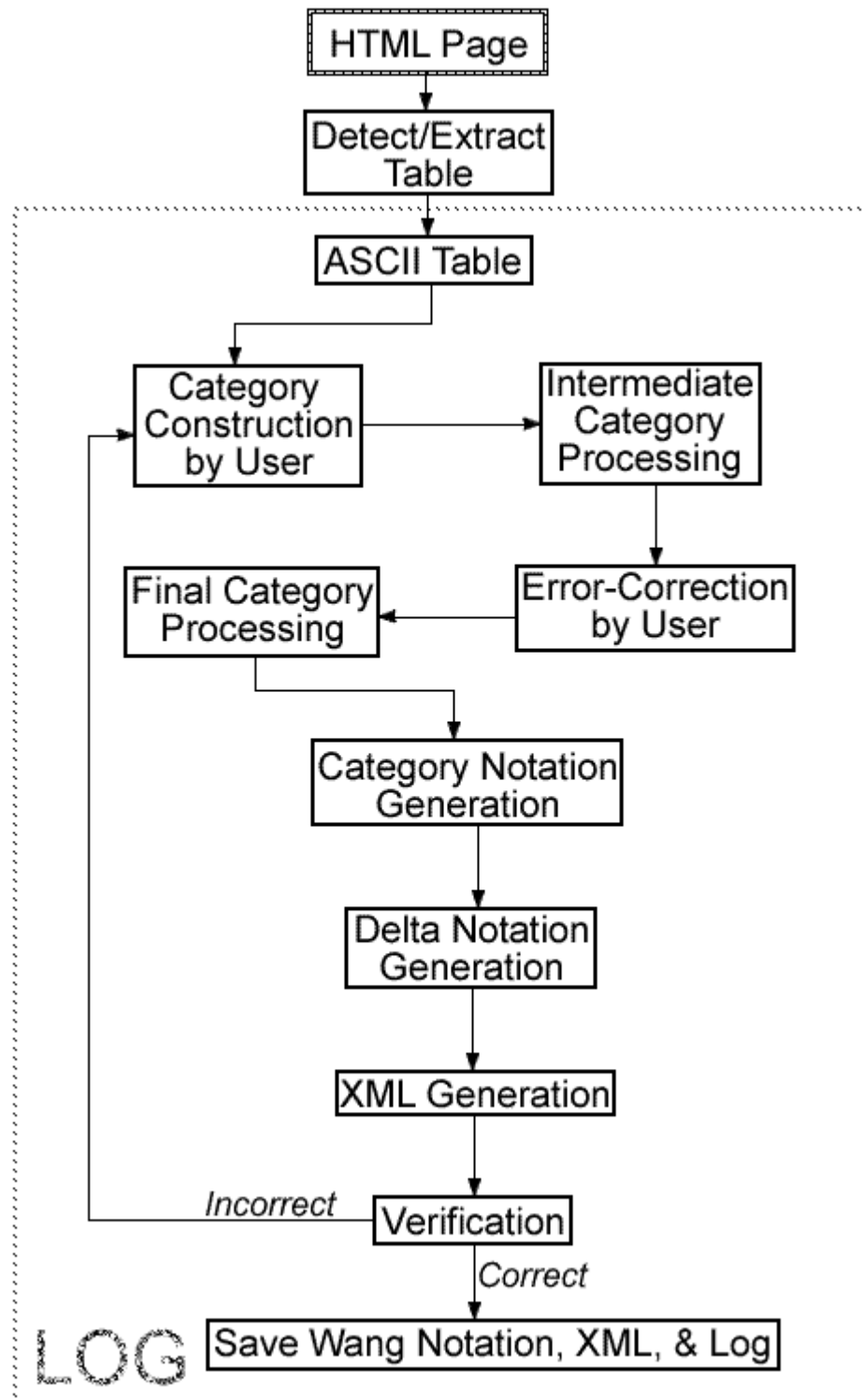


Figure 43: WNT Flowchart

4. Evaluation of WNT Methodology

WNT can be used on any computer with Matlab 7.0 or higher. Preliminary testing was done to check WNT's user friendliness and devise a training scheme. Final testing was done to evaluate WNT.

4.1 Preliminary Testing

Preliminary testing was conducted on two users as a means to determine the usability of WNT. Preliminary testing revealed that numerous cosmetic changes had to be made: the program was not user friendly. For example, there were occasions where several windows popped open at the same time on top of each other and were not sized appropriately. This resulted in the user having to spend time re-sizing and moving around the windows. It was also easy to ignore some portions of the Matlab table because the original HTML file was not displayed for reference.

To rectify these problems, all the GUIs in WNT now have scrollbars and are assigned a location and size on the screen corresponding to the contents of the GUIs. The users no longer have to move the GUIs around or resize them. If a GUI is too big for the screen, the scrollbars can be used to view the entire GUI. In addition, the original HTML file is displayed at all times and an additional button ('Undo Last') was added to the error correction GUI. Implementing these improvements required exploration of some arcane aspects of Matlab.



Figure 44: GUI to control tables

In the earlier version, each table was processed individually; the user had to select the table they wanted to process by typing in the name of the table into Matlab and running one table at a time. To improve usability, a GUI was designed to control the tables to be tested. This GUI is shown in Figure 44. As a result of all these changes, WNT is now more intuitive for the naïve user.

4.2 Training

The subjects were trained in how to use WNT by the author of this thesis and a PowerPoint presentation, found in Appendix F. The PowerPoint covers table concepts, such as trees, virtual headers, category cells, and delta cells. It does not detail the criteria for selecting unique categories (Section 1.3.4) because a naïve user who has not given tables much thought would be confused by the specific criteria. Users learned how to pick categories correctly by example later in the training session.

The PowerPoint also illustrated, step-by-step, how to use WNT. These illustrations consisted of screenshots taken during interactive category construction, error-correction, and verification. The last stage of training consisted of the author processing five tables in front of the subject. This portion of training was interactive; each step for every table and the reasons behind the selection of categories, error-correction, and verification were explained. Subjects were free to ask questions. Subjects familiar with computer science were faster to train, taking about half an hour each. Subjects that were unfamiliar with computer science took significantly longer, upwards of 45 minutes, to train. The tables used to train subjects can be found in Appendix A.

4.3 Evaluation

Final evaluation was conducted on 12 subjects with 17 tables each (presented in Appendix B). The tables used for evaluation were picked such that there were 5-6 different kinds of tables; some well-formed and some badly-formed. The tables in the beginning of the session were fairly simple and similar to some of the training tables. The tables in the middle of the session had the highest confusion factor and troubled almost all subjects. The tables towards the end of the session were neither simple, nor difficult, to see how well the subjects had learned. All the tables required corrections, most often the addition of virtual headers. However, none of the tables required extensive corrections if the categories were picked correctly.

All subjects were trained in the same manner and none of the subjects were given any input by the author while processing tables. The tables were presented to all the subjects in the same order in continuous sessions. The subjects were from diverse age

groups (18-51) and backgrounds (electrical engineering graduate students, undergraduate electrical engineers, communications graduate students, chemistry graduate students, aeronautical engineering professors, figure skating coaches, aspiring actors, and accountants). The trainer, remained present to cope with any error by Matlab, which required restarting the Matlab program. Every subject could choose to process a table as many times as they wished and each attempt, whether partial or complete, was recorded in the log. Section 5 will discuss the results from the evaluation.

5. Evaluation of WNT

Every attempt by every subject was recorded in detail during evaluation. An example of the complete log for one attempt appears in Appendix E. The log recorded times and button clicks made by the user, specifying whether the button click was to undo a mistake or not. A subject's interaction with WNT can be re-created with the logs. Appendix G shows summaries of times for every table.

An example summary table for T09 is shown in Table 5. All values (time in seconds) are averages over subjects. *# of attempts* is the average number of attempts made by all subjects on a table. *Time for Pre-Processing* (computer time) is the time taken to display the original HTML table, convert the ASCII file to a Matlab array, and display a corresponding GUI to the subject. *Time to Construct Categories* (subject time) is the time taken by subjects to think about and click the cells designating categories. This time indicates the confusion factor (Section 0) of a table; subjects spend more time constructing categories when a table is confusing.

Time for Category Correction (subject time) is the time subjects took to correct all categories in the table using the error-correction GUI. This time is higher for confusing and badly-formed tables and lower when subjects have seen similar tables before. *Time for Final Processing* (computer time) is the time taken to perform final category processing, generate category notation, generate delta notation, and generate the XML representation. *Total Time* is the addition of all time and *% Subject Time* is the percent of total time that is subject time.

Table 5: Distribution of Processing Time for T09, Average Over All Subjects

	AVERAGE	STD. DEVIATION
# of attempts	1.67	0.65
Time for Pre-Processing	0.52	0.10
Time to Construct Categories	80.68	67.63
Time for Category Correction	103.21	126.45
Time for Final Processing	0.42	0.19
Total Time	184.81	192.11
Percent Table is Completed	77.78	35.06
% Subject Time	0.99	0.01

Wang notation was generated in 82.75% of all attempts and was generated *correctly* in 57.25% of all attempts (Table 6). Figure 45 shows the results of the evaluation by

subject and Figure 46 shows the results of the evaluation by table. The dark gray bars represent the percent of all attempts where Wang notation was generated correctly. The light gray bars represent the percent of all attempts where Wang notation was generated incorrectly.

Table 6: Success Rate by Table, Average Over Subjects

	# of attempts	% correct	% generated
T01	12	100.00	100.00
T02	12	100.00	100.00
T03	12	100.00	100.00
T04	12	58.33	75.00
T05	16	75.00	87.50
T06	12	100.00	100.00
T07	12	100.00	100.00
T08	12	91.67	100.00
T09	20	20.00	60.00
T10	18	38.89	66.67
T11	22	18.18	63.64
T12	15	53.33	80.00
T13	14	50.00	100.00
T14	15	26.67	86.67
T15	16	43.75	75.00
T16	18	55.56	83.33
T17	17	29.41	70.59
TOTAL	255	57.25	82.75

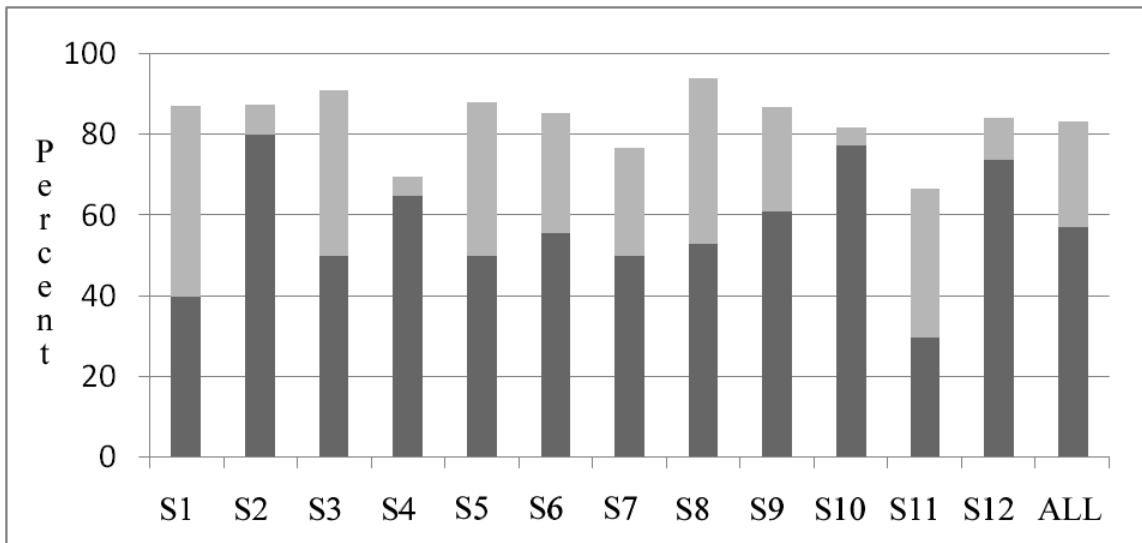


Figure 45: WNT Results by Subject

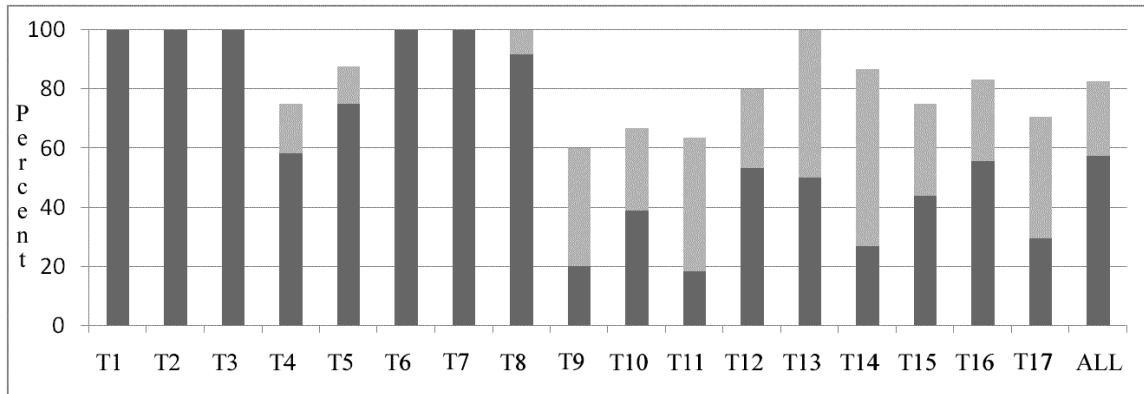


Figure 46: WNT Results by Table

Each subject processed 17 tables, but there were more than 17 attempts per subject (Table 6); most subjects used the verification tool to validate their responses, and if incorrect, frequently started over. The numbers above are percentages of *all* attempts. If averages are taken over the set of tables, Wang notation was generated for 98% of all tables, and was generated correctly for 71% of all tables. Wang notation could not be generated when the subject made corrections that produced invalid trees. This usually occurred when a large number of corrections were made and the integrity of the indented notation was overlooked.

Wang notation was generated incorrectly by subjects who did not understand the concept of virtual headers (Section 1.3.3). WNT automatically adds virtual headers to some category configurations, therefore, when WNT does not automatically add virtual headers, some subjects either forgot or didn't realize that they had to add virtual headers. For example, one of the categories in T15 consists of: *Gross Domestic Product*, *GDP at Purchasing Power Parity*, and *Inflation Index (2000=100)*, found in the top row of T15. This category is rootless and needs a virtual header. WNT displays the indented notation with a root, albeit incorrect:

Gross Domestic Product	
	GDP at Purchasing Power Parity
	Inflation Index (2000=100)

Four subjects assumed WNT was correct and failed to modify the indented notation. The correct indented notation is shown on the next page.

Albanian Economy (VH)	
	Gross Domestic Product
	GDP at Purchasing Power Parity
	Inflation Index (2000=100)

One of the most challenging tasks for the naïve user was how to choose unique categories (Section 1.3.4). This task confused several subjects, particularly for T09 and T10. In both T09 and T10, the subject had to realize that the two leftmost columns, and the three leftmost columns, respectively, constituted a single category. They cannot be split, because there are no repeated subcategories. As a result, only four subjects generated correct notation for T09, and only seven subjects generated correct notation for T10 (largely because the subjects learned from T09).

Subjects also had trouble picking unique categories because they frequently over-defined tables by defining redundant categories. This could be seen in T11, T13, T14, and T17. For example, in T14, a significant fraction of subjects picked the first column as one category, the second column as the second category, and *Area* and *Maximum Depth* as the third category with a virtual header. This construction of categories is incorrect, because both columns have one root with 25 children describing the same set of delta cells. The correct category construction is to assign the first column to one category, and *Body of Water*, *Area*, and *Maximum Depth* to be the second category with a virtual header.

T04 was not generated three times because it is a badly formed table: the top row consists of years, the second and third row consist of the words *Female* and *number* respectively. If T04 was well-formed, *Female* and *number* would appear above the years, thus giving that category a root. The original T04 is quite confusing and corrections are a challenge. The next pages shows the output from WNT after intermediate processing.

RENAME			
	2000		
		Female	
			Number
	2001		
			Number
	2002		
			Number
	2003		
			number
	2004		
			number

The word *Female* was removed in four paths because WNT deemed it redundant. Below are the two possible fixes. Every subject, but one, attempted the fix on the left, which involved numerous corrections and often resulted in invalid indented notation. The fix on the right is straightforward but requires a thorough grasp of the concepts of trees and virtual headers and was only implemented by one subject, S02.

Year			
	2000		
		Female	
			Number
	2001		
		Female	
			Number
	2002		
		Female	
			number
	2003		
		Female	
			Number
	2004		
		Female	
			number

Female		
	Number	
		2000
		2001
		2002
		2003
		2004

Virtual headers are difficult because subjects have to recognize the instances where a virtual header is needed and the lack of virtual headers complicates the table enough that subjects have trouble choosing unique categories. Tables would be much less confusing if virtual headers were not needed, because the presence of a root (virtual header) removes ambiguity about its children. Without all roots present it is difficult for subjects to ascertain unique categories, but the absence of virtual headers does not always result in incorrect selecting of categories. Tables with obvious virtual headers

(T02, T03, T04, among others) were almost always correctly demarked. It is when the need to add virtual headers is not obvious that subjects have trouble (T11, T12, T13, among others).

There are several ways in which WNT could warn subjects when they choose incorrect categories. To prevent over-defining categories (Section 5), WNT could issue a warning if the subject chooses two categories that are of the same shape ($n \times 1$ or $1 \times n$), are completely adjacent to each other (every part of the two category rectangles are adjacent), have only one level, and have the same number of children. Another indication that a category is picked incorrectly would be if the number of nodes decreases with each level. A warning could be given if any delta cell is associated directly with any non-leaf node (Section 3.4.2). WNT could determine if a subject chooses a category where the subcategory trees are repeated and therefore need to be split, or if the subject chooses two or more categories that should really be one category.

The personality and background of the subjects made a difference in the results. In general, subjects with knowledge of computer science (S01, S02, S10, and S12) picked up the concepts quickly. S01 had more knowledge of computer science than anyone else, but was also the most careless subject, and therefore, generated most tables incorrectly. S02, S10, and S12 were thorough and had previous knowledge and were the best at using WNT. S04 had no background in computer science and did not understand the concepts of trees readily, but was very thorough. Therefore, S04 did not generate notation for many tables (due to invalid indented notation), but when notation was generated, it was usually correct. The other subjects were a mix of people with limited exposure to computer science and some with no exposure to computer science. The more thorough the subject was, the better they performed, and all subjects learned from their mistakes.

Table 7 shows the average total time by table, the average percentage of user time, the average user time, and the average computer time. Figure 47 shows the average total time by table and Figure 48 shows the average total time by subject. User time accounts for 98% of total processing time. The total amount of time required to process a table was directly related to how well-formed the table was. The total time taken by subjects

was not a good indication of their performance on WNT. The correlation coefficient of average total time and percent of generated tables is -0.76 and the correlation coefficient of average total time and percent of correctly generated tables is -0.66. S04 had the highest average total time because S04 was thorough, but did not generate many tables. S02, on the other hand, generated a large portion of the tables, but with a rather low average total time. S11 had a high average total time and the worst performance, but S01, S03, and S08 had low average total times and similarly mediocre performances.

Table 7: Average Times by Table

Table	Total Time	% User Time	User Time	Computer Time
T01	73.20	0.98	71.74	1.46
T02	70.80	0.98	69.38	1.42
T03	62.06	0.98	60.82	1.24
T04	143.38	0.99	141.95	1.43
T05	66.79	0.98	65.45	1.34
T06	41.73	0.98	40.90	0.83
T07	48.56	0.98	47.59	0.97
T08	42.76	0.98	41.90	0.86
T09	184.81	0.99	182.96	1.85
T10	185.11	0.99	183.26	1.85
T11	112.27	0.98	110.02	2.25
T12	96.16	0.98	94.24	1.92
T13	78.75	0.98	77.18	1.58
T14	87.34	0.98	85.59	1.75
T15	53.79	0.98	52.71	1.08
T16	48.99	0.98	48.01	0.98
T17	214.46	0.98	210.17	4.29
T01-T17	94.76	0.98	93.17	1.59

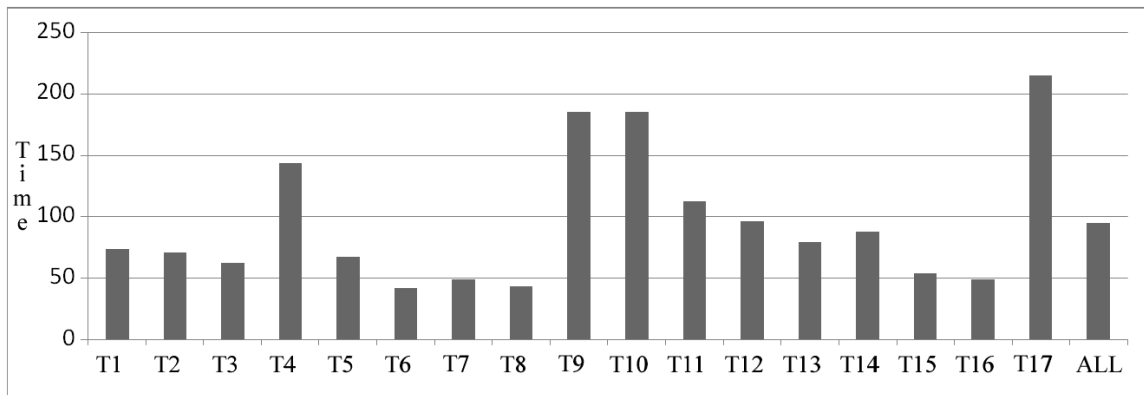


Figure 47: Average Total Time by Table

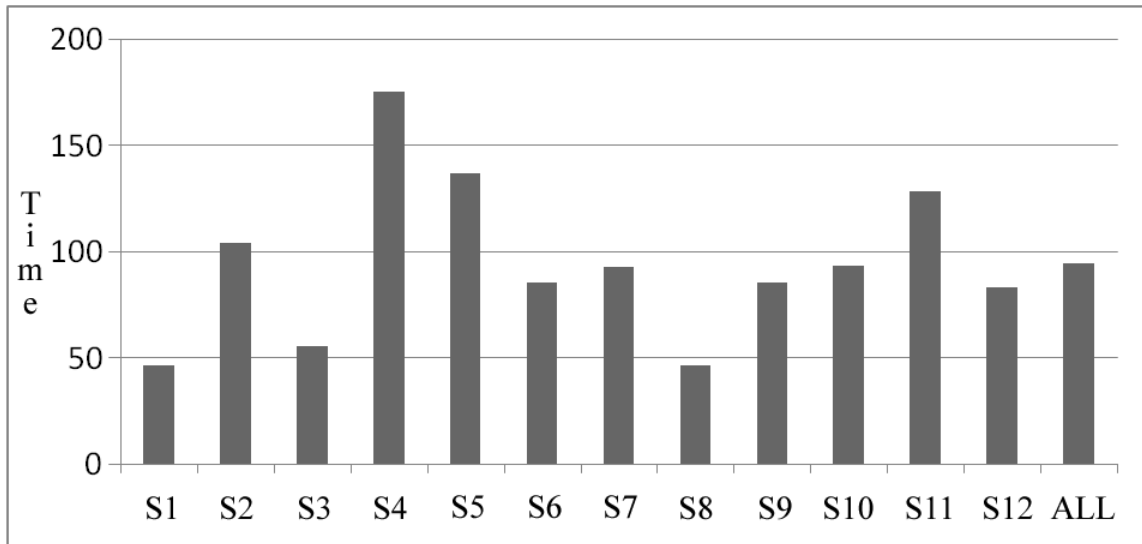


Figure 48: Average Total Time by Subject

This evaluation of WNT shows that with experience, subjects could generate notation correctly. More experience could be gained through improved training (Section 6.3), a large database of training tables, and feedback during the evaluation process. Every subject stated that with more experience, she or he could generate notation correctly. The error-correction and verification tools are essential for generating correct notation. Evaluation also showed that WNT was able to process a variety of tables whose layout-independent representations could be generated correctly by an experienced user.

6. Future Work

6.1 Aggregations and Annotations

Wang notation captures only the direct relations between and within category and delta cells in a hierarchical or tree format. The subtleties of tables and additional information, such as the title, caption, author, citation, and number in database cannot be captured by Wang notation. The XML representation is generated to capture additional table information, as well as acting as a medium to relay layout independent tables to other applications. Currently, a user can enter additional information to be added to the XML representation, but there is no provision for capturing aggregations and annotations.

As discussed in Section 3.6, the ontology that represents general tables (Figure 36), has a schema for capturing aggregations and annotations, but is not yet part of WNT. Some tables include *aggregates*, such as averages, in the original table, and in other tables, it would be useful if the user could add columns or rows containing aggregates. Either way, to create an ontology, it is useful to know which cells are aggregates and which cells are not. It is also useful to know which cells contain *annotations*. Annotations include *augmentations*, such as units, and *footnotes*.

It will be difficult to capture aggregations and annotations automatically, because they are inherently lexical. It might be possible to have WNT guess the locations of cells containing aggregations and annotations and then have a user correct or approve the guesses. Extensive research into where aggregations and annotations generally appear would be required to modify WNT to make initial guesses. The addition of aggregations and annotations would increase the quality of ontologies constructed with information from WNT.

6.2 Automation and Learning

Further automating and learning would constitute a large improvement in the speed and robustness of WNT. Automation and learning go hand in hand because further automation is only possible if adaptive learning is employed. There are two instances where automation and adaptive learning can be implemented.

First, it is possible to streamline the category construction step. Instead of the user delineating categories, WNT could determine which cells are category cells and which cells are delta cells based on structural patterns within the table. Structural patterns in tables can be explored using foreign tables (Section 1.3.6). A much more challenging problem for WNT is to separate all category cells into separate categories. This challenge can be overcome with the use of adaptive learning; WNT could “learn” to make guesses on the locations of categories based on past tables. All guesses would have to be approved or corrected by the user.

Second, the error correction step can be simplified if WNT can “learn” to make corrections based on past responses. A detailed log would be instrumental in implementing a more streamlined error-correction process. WNT could compare the current indented notation to past indented notations and make corrections based on similarities between indented notations. Also using the log, the error-correction GUI (Figure 32) could have a few dynamic buttons that change depending on what types of corrections are performed most often. Ideally, WNT should not make the same mistake twice.

6.3 Improved Training

Improved training would greatly increase the fraction of tables that are processed correctly. The current training method consists of the subject watching the author use WNT and a PowerPoint presentation. A more thorough and useful process would be to have the subject interact during training by using a mock WNT before actual testing. The mock WNT could be in the form of a website that looks like the real WNT. Subjects can process 2-3 tables on the website and every time they make a mistake, the website will prompt them with corrections and suggestions, thus teaching them the nuances of tables firsthand.

7. Conclusion

The Wang Notation Tool (WNT) was developed as part of the project known as TANGO (Table ANalysis for Generating Ontologies) [1]. TANGO aims to create an ontology by “understanding” a multitude of tables. The first step of TANGO is to fully interpret a table’s structure and conceptual content by converting it to a layout independent, or *canonicalized*, form with guidance from a user. Few attempts at complete interpretation, like that performed by WNT, appear in the literature, and none that convert HTML tables to Wang notation.

WNT is an interactive tool for converting HTML tables to two layout-independent representations. The first layout independent representation generated is Wang notation [2]; the second, an extension of Wang notation, is XML representation corresponding to an ontology that represents general tables. Both representations delineate the tree structure of the category cells and relate delta cells to branches of the category trees. The XML representation includes additional information about the table (title, caption, citation) and cells (aggregates, annotations).

The input to WNT is an ASCII file resulting from parsing an HTML table with a JAVA program that extracts the content and layout information necessary for complete interpretation. WNT, written in Matlab, interacts with the user to determine the relationships within a table and generate Wang notation and XML representation. The XML representation is sent to researchers at BYU to generate mini-ontologies, discover inter ontology mappings and merge all information into growing ontologies [19]. WNT is also being used for ontology-related applications, such as Query By Table [20].

The average total time for an experienced user (the author of this thesis) was 48 seconds. This time was faster than that of every subject except S01, who was very careless and did not make many corrections. S01 did not generate many tables correctly, but the experienced user generated 100% of the tables correctly. The average time over 12 subjects for 17 tables was 95 seconds. Overall, 71% of tables were correctly converted to Wang notation.

The subjects tested were naïve, but upon detailed feedback after the evaluation session, all of them understood how WNT worked. The average time for training was

about 30 minutes and the average time for evaluation was about 90 minutes. A longer, more interactive training session may improve results and speed up evaluation. In addition, adaptation to the current spectrum of tables would increase the speed and robustness of WNT. However, even before implementing an adaptive WNT, several changes can be made as a result of the evaluation. The two most difficult aspects of WNT for subjects were virtual headers (Section 1.3.3) and choosing unique categories (Section 1.3.4). Additions to WNT that could alleviate these difficulties are described in Section 5.

About 85% of the development time for WNT was spent writing Matlab code. WNT consists of over 1700 lines of code and 54 functions. Aspects of the Matlab program that required considerable thought were: developing the GUI for interaction, pre-order traversal of category trees, determining delta notation, adding error-correction, and adding scrollbars to all figures. WNT is a fast and robust tool for generating Wang notation, especially as a user gains more experience.

References

- [1] Y.A. Tijerino, D.W. Embley, D.W. Lonsdale, Y. Ding, and G. Nagy, "Toward Ontology Generation from Tables," *World Wide Web*, vol. 8, no. 3, pp 261 – 285, Sept. 2005.
- [2] X. Wang, "Tabular Abstraction, Editing, and Formatting," Ph.D Dissertation, University of Waterloo, Waterloo, ON, Canada, 1996.
- [3] W. Kornfield and J. Wattecamps, "Automatically Locating, Extracting and Analyzing Tabular Data," *Proceedings of 26th ACM SIGIR*, pp.347-348, Melbourne, Australia, 1998.
- [4] S. Chandran and R. Kasturi, "Structural Recognition of Tabulated Data," *Proceedings of 2nd International Conference on Document Analysis and Recognition*, pp. 516-519, Tsukuba Science City, Japan, Oct. 1993.
- [5] D. Pinto, A. McCallum, X. Wei, and W. B. Crott, "Table Extraction Using Conditional Random Fields," *Proceedings of 26th ACM SIGIR*, pp. 235-242, Toronto, Canada, 2003.
- [6] P. Pyreddy and W. B. Croft, "TINTIN: A System for Retrieval in Text Tables," *International Conference On Digital Libraries*, pp. 193-200, Philadelphia, Pennsylvania, July 1997.
- [7] A.C. e Silva, A. M. Jorge, and L. Torgo, "Design of an end-to-end method to extract information from tables," *International Journal on Document Analysis and Recognition*, vol. 8, no. 2, pp. 144-171, June 2006.
- [8] W. Gatterbauer, P. Bohunsky, M. Herzog, B. Krupl, and B. Pollak, "Towards Domain-Independent Information Extraction from Web Tables," *World Wide Web*, pp. 71-80, Banff, Canada, May 2007.
- [9] D.W. Embley, M. Hurst, D. Lopresti, and G. Nagy, "Table-Processing Paradigms: a Research Survey," *International Journal on Document Analysis and Recognition*, vol. 8, no. 2, pp. 66-86, June 2006.
- [10] M. A. Rahgozar and R. Cooperman, "A Graph-Based Table Recognition System," *Document Recognition III, SPIE Proceedings Series*, vol. 2660, pp. 192-203, San Jose, California, Jan. 1996.
- [11] J. Hu, R. Kashi, D. Lopresti, and G. Wilfong, "Table Structure Recognition and Its Evaluation," *Document Recognition and Retrieval VIII, SPIE Proceedings Series*, vol. 4307, pp. 44-55, San Jose, California, Jan. 2001.

- [12] T. Watanabe, Q. Luo, and N. Sugie, "Layout Recognition of Multi-Kinds of Table-Form Documents," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 4, pp. 432-445, April 1995.
- [13] E. Green and M. Krishnamoorthy, "Recognition of Tables Using Table Grammers," *Proceedings of 4th Symposium on Document Analysis and Information Retrieval*, pp. 261-277, Las Vegas, Nevada, April 1995.
- [14] R. Zanibbi, D. Bolstein, and J. R. Cordy, "A Survey of Table Recognition: Models, Observations, Transformations, and Inferences," *International Journal on Document Analysis and Recognition*, vol. 7, no. 1, pp. 1-16, April 2004.
- [15] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific American*, May 2001.
- [16] A.V. Aho, R. Sethi and J. D. Ullman, *Compilers: Principles, Techniques and Tools*, Addison-Wesley, 1986.
- [17] P. Jha, "Interactive Wang Notation Tool for Web Pages," DOC Lab, Rensselaer Polytechnic Institute, Troy, NY, Technical Report, May 2007.
- [18] E.M. Reingold and W.J. Hansen, *Data Structures*, Boston: Little, Brown and Company, 1983.
- [19] S. Lynn, and D.W. Embley, "Automatic Generation of Ontologies from Canonicalized Web Tables", *submitted manuscript*, March 2008, <http://tango.byu.edu/>.
- [20] R. Padmanabhan, and G. Nagy, "Query By Table", *submitted to ICPR*, 2008.
- [21] A. Laurentini, and P. Viada, "Identifying and Understanding Tabular Material in Compound Documents," *Proceedings of the International Conference on Pattern Recognition*, pp. 405-409, The Hague, Netherlands, Sept. 1992.
- [22] T. Hu, "Recognizing Table Entries In A Scanned Document," M.S. Thesis, Rensselaer Polytechnic University, Troy, NY, October 1993.
- [23] D.W. Embley, D.P. Lopresti, and G. Nagy, "Notes on Contemporary Table Recognition," *Document Analysis Systems 2006*, pp. 164-175, 2006.

Appendix

A. Training Tables

The following tables were used to train the user in WNT. Most tables have a title on top that is not part of Wang notation, but is part of the XML representation.

Table 8: University Degrees for Males (TRN1)

University degrees, diplomas and certificates granted by sex, by province
(Males)

	<u>2000</u>	<u>2001</u>	<u>2002</u>	<u>2003</u>	<u>2004</u>
	Males				
	number				
Canada	73,233	72,852	75,111	80,049	84,216
Newfoundland and Labrador	1,158	1,101	1,149	1,143	1,173
Prince Edward Island	186	195	195	198	222
Nova Scotia	3,096	2,967	3,009	3,465	3,792
New Brunswick	1,608	1,644	1,725	1,743	1,947
Quebec	21,144	21,009	21,651	23,625	24,828
Ontario	27,927	28,314	28,764	30,822	32,574
Manitoba	2,229	2,211	2,211	2,256	2,412
Saskatchewan	2,385	2,319	2,346	2,382	2,310
Alberta	5,946	6,123	6,864	6,927	7,254
British Columbia	7,560	6,975	7,194	7,494	7,704

Table 9: Divorces by Province (TRN2)

Divorces by province and territory

	<u>1999</u>	<u>2000</u>	<u>2001</u>	<u>2002</u>	<u>2003</u>
	number of divorces				
Canada	70,910	71,144	71,110	70,155	70,828
Newfoundland and Labrador	892	913	755	842	662
Prince Edward Island	291	272	246	258	281
Nova Scotia	1,954	2,054	1,945	1,990	1,907
New Brunswick	1,671	1,717	1,570	1,461	1,450
Quebec	17,144	17,054	17,094	16,499	16,738
Ontario	26,088	26,148	26,516	26,170	27,513
Manitoba	2,572	2,430	2,480	2,396	2,352
Saskatchewan	2,237	2,194	1,955	1,959	1,992
Alberta	7,931	8,176	8,252	8,291	7,960
British Columbia	9,935	10,017	10,115	10,125	9,820
Yukon Territory	112	68	91	90	87
Northwest Territories including Nunavut	83
Northwest Territories	..	94	83	68	62
Nunavut	..	7	8	6	4

Table 10: Economy of Mali (TRN3)

Year	Gross Domestic Product	US Dollar Exchange	Inflation Index (2000=100)
1980	356,026	211.29 CFA Francs	48
1985	551,381	449.37 CFA Francs	75
1990	749,122	272.21 CFA Francs	70
1995	1,405,870	499.06 CFA Francs	92
2000	1,899,186	710.24 CFA Francs	100
2005	2,760,689	525.34 CFA Francs	111

Table 11: Food Services for Nunavut (TRN4)

Food services and drinking places — Full-service restaurants, by province and territory
(Nunavut)

	2001	2002	2003	2004	2005
	\$ millions				
Nvt.					
Operating revenue	F	F	x	3.3	x
Operating expenses	F	F	x	3.1	x
Salaries, wages and benefits	F	F	x	1.1	x
			%		
Operating profit margin	F	F	x	7.6	x

Table 12: Wang Table (TRN5)

Year	Term	Mark					
		Assignments			Examinations		Grade
		Ass1	Ass2	Ass3	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

B. Test Tables

The following tables were processed by each subject in testing. The results of this testing are discussed in detail in Section 5.

Table 13: Induced Abortions by Province (T01)

Pregnancy outcomes by province or territory of residence
(Induced abortions)

	2003	
	Induced abortions	
	<u>number of events</u>	<u>rate per 1,000 women</u>
Canada	103,768	12.8
Newfoundland and Labrador	895	6.6
Prince Edward Island	137	4.0
Nova Scotia	1,925	8.1
New Brunswick	944	5.0
Quebec	30,802	16.4
Ontario	36,666	11.6
Manitoba	3,670	12.9
Saskatchewan	1,846	7.6
Alberta	10,814	12.9
British Columbia	15,499	14.5
Yukon Territory	129	14.7
Northwest Territories	255	21.5
Nunavut

Table 14: Deaths and death rate, by province (T02)

Deaths and death rate, by province and territory
(Death rate)

	<u>2001/2002</u>	<u>2002/2003</u>	<u>2003/2004</u>	<u>2004/2005</u>	<u>2005/2006</u>
	death rate per 1,000 population				
Canada	7.1	7.1	7.2	7.3	7.3
Newfoundland and Labrador	7.9	8.2	8.3	8.5	8.7
Prince Edward Island	8.8	8.9	8.6	8.8	8.9
Nova Scotia	8.5	8.5	8.7	8.9	9.0
New Brunswick	8.1	8.2	8.4	8.6	8.8
Quebec	7.4	7.3	7.5	7.4	7.0
Ontario	6.7	6.8	6.9	7.1	7.2
Manitoba	8.4	8.5	8.5	8.6	8.7
Saskatchewan	8.7	8.9	9.1	9.2	9.3
Alberta	5.8	5.8	5.9	6.0	6.2
British Columbia	7.0	6.9	7.1	7.1	7.1
Yukon Territory	5.0	4.8	4.4	4.5	4.8
Northwest Territories	4.0	4.4	4.8	5.0	5.0
Nunavut	4.6	4.5	4.5	4.5	4.5

Table 15: Deaths and death rate, by province (T03)Deaths and death rate, by province and territory
(Number of deaths)

	<u>2001/2002</u>	<u>2002/2003</u>	<u>2003/2004</u>	<u>2004/2005</u>	<u>2005/2006</u>
	number of deaths				
Canada	220,494	223,905	230,092	233,749	234,914
Newfoundland and Labrador	4,126	4,276	4,318	4,405	4,494
Prince Edward Island	1,205	1,217	1,190	1,208	1,231
Nova Scotia	7,922	7,944	8,146	8,305	8,446
New Brunswick	6,065	6,181	6,325	6,461	6,585
Quebec	54,735	54,896	56,475	55,800	52,900
Ontario	80,993	83,410	85,524	88,196	90,945
Manitoba	9,720	9,852	9,940	10,096	10,226
Saskatchewan	8,650	8,880	9,061	9,172	9,250
Alberta	17,937	18,098	18,888	19,517	20,310
British Columbia	28,697	28,694	29,752	30,103	30,028
Yukon Territory	150	145	136	141	149
Northwest Territories	164	183	205	212	214
Nunavut	130	129	132	133	136

Table 16: University Degrees (Females) by province (T04)University degrees, diplomas and certificates granted by sex, by province
(Females)

	<u>2000</u>	<u>2001</u>	<u>2002</u>	<u>2003</u>	<u>2004</u>
	Females number				
Canada	103,326	105,207	111,027	118,467	124,830
Newfoundland and Labrador	1,773	1,755	1,749	1,830	1,989
Prince Edward Island	345	411	363	429	453
Nova Scotia	4,542	4,677	4,857	5,304	5,769
New Brunswick	2,424	2,460	2,670	2,811	3,000
Quebec	29,706	30,144	32,358	34,161	36,384
Ontario	39,297	39,972	41,982	45,042	47,862
Manitoba	3,114	3,183	3,366	3,612	3,897
Saskatchewan	3,408	3,378	3,393	3,486	3,528
Alberta	8,106	8,961	9,483	10,269	10,758
British Columbia	10,614	10,263	10,803	11,520	11,196

Table 17: Food and Drink for Alberta (T05)Food services and drinking places — Full-service restaurants, by province and territory
(Alberta)

	2001	2002	2003	2004	2005
	\$ millions				
Alta.					
Operating revenue	1,876.5	2,119.3	2,224.9	2,301.9	2,459.9
Operating expenses	1,753.8	2,001.3	2,160.7	2,189.2	2,333.6
Salaries, wages and benefits	549.3	650.4	695.2	705.1	760.8
	%				
Operating profit margin	6.5	5.6	2.9	4.9	5.1

Table 18: Food and drink for Newfoundland and Labrador (T06)Food services and drinking places — Full-service restaurants, by province and territory
(Newfoundland and Labrador)

	2001	2002	2003	2004	2005
	\$ millions				
N.L.					
Operating revenue	83.0	87.0	122.0	98.9	104.9
Operating expenses	76.8	83.3	119.3	97.2	104.4
Salaries, wages and benefits	23.1	26.2	38.3	30.1	30.7
	%				
Operating profit margin	7.5	4.2	2.2	1.8	0.5

Table 19: Food and drink for Prince Edward Island (T07)Food services and drinking places — Full-service restaurants, by province and territory
(Prince Edward Island)

	2001	2002	2003	2004	2005
	\$ millions				
P.E.I.					
Operating revenue	57.8	F	x	52.8	48.9
Operating expenses	53.4	F	x	51.5	47.3
Salaries, wages and benefits	16.7	F	x	16.8	15.7
	%				
Operating profit margin	7.6	F	x	2.5	3.3

Table 20: Infant mortality rates by province (T08)Infant mortality rates, by province and territory
(Both sexes)

	<u>2000</u>	<u>2001</u>	<u>2002</u>	<u>2003</u>	<u>2004</u>
	Both sexes				
Canada	5.3	5.2	5.4	5.3	5.3
Newfoundland and Labrador	4.9	4.9	4.5	5.0	5.1
Prince Edward Island	3.5	7.2	1.5	4.9	4.3
Nova Scotia	4.9	5.6	4.2	5.7	4.6
New Brunswick	3.5	4.3	3.8	4.1	4.3
Quebec	4.7	4.7	4.8	4.4	4.6
Ontario	5.6	5.4	5.3	5.3	5.5
Manitoba	6.5	7.0	7.1	8.0	7.0
Saskatchewan	6.8	5.5	5.7	6.3	6.2
Alberta	6.6	5.6	7.3	6.6	5.8
British Columbia	3.7	4.1	4.6	4.2	4.3
Yukon Territory	2.7	8.7	8.8	6.0	11.0
Northwest Territories	8.9	4.9	11.0	5.7	0.0
Nunavut	12.3	16.9	11.0	19.8	16.1

Table 21: Lakes of Canada (T09)

Province	Principal Lakes	Elevation (m)	Area (km ²)
Newfoundland and Labrador	Smallwood Reservoir	471	6527
	Melville Lake	tidal	3069
Quebec	Lac Mistassini	372	2335
	Reservoir Manicougan	360	1942
	Reservoir Gouin	404	1570
	Lac à l'Eau-Claire	241	1383
	Lac Bienville	426	1249
	Lac Saint-Jean	98	1003
	Reservoir Pipmuacan	396	978
	Lac Minto	168	761
	Reservoir Cabonga	361	677
Manitoba	Lake Winnipeg	217	24387
	Lake Winnipegosis	254	5374
	Lake Manitoba	248	4624
	Southern Indian Lake	254	2247
	Cedar Lake	253	1353
	Island Lake	227	1223
	Gods Lake	178	1151
	Cross Lake	207	755
	Playgreen Lake	217	657
Alberta	Lake Clair	213	1436
	Lesser Slave Lake	577	1168
British Columbia	Williston Lake	671	1761
	Atlin Lake	668	775

Table 22: Mountains of Canada (T10)

Province	Range/Region	Principal Heights	Elevation (m)
New Foundland and Labrador	Torngat mountains	Mount Caubvik (highest point in Nfld. lab.) (on Nfld. Lab. - Que. boundary)	1652
		Cisque Mountian	1568
		Mount Cladonia	1453
		Mount Eliot	1356
		Mount Tetragona	1356
		Quartzite Mountain	1186
		Blow Me down Mountian	1183
	Mealy Mountains	Unnamed peak (46°37' 58°33')	1176
	Kaumajet Mountains	Bishops Mitre	1113
	Long Range Mountains	Lewis Hills	814
		Gros Morne	806
Manitoba		Baldy Mountain	817
		Riding Mountain	785
Quebec	Monts Torngat	Mount D'Iberville (highest point in Que.) (on Nfld. lab. - Que. boundary)	1652
	Les Appalaches	Mont Jacques Cartier	1268
		Mont Gosford	1192
		Mont Richardson	1185
		Mount Me'gantic	1105
	Les Laurentides	Unnamed peak (47°19' 70° 50')	1166
		Mont Tremblant	968
		Mount Sainte-Anne	800
		Mont Sir-Wilfrid	783
	Monts Otish	Unnamed peak (52°19' 71° 27')	1135
	Collines Monterigiennes	Mont Brome	533

Table 23: Administrative units of Utah (T11)

administrative units	capital	area (sq.km.)	population 2000-04-01 census	population 2006-07-01 estimate
Beaver County	Beaver	6,713.97	6,005	6,294
Box Elder County	Brigham City	17,428.11	42,745	47,197
Cache County	Logan	3,038.24	91,391	98,662
Carbon County	Price	3,845.02	20,422	19,469
Daggett County	Manila	1,872.72	921	947
Davis County	Farmington	1,641.28	238,994	276,259
Duchesne County	Duchesne	8,432.92	14,371	15,701
Emery County	Castle Dale	11,555.34	10,860	10,698
Garfield County	Panguitch	13,489.18	4,735	4,534
Grand County	Moab	9,567.62	8,485	8,999
Iron County	Parowan	8,551.73	33,779	40,544
Juab County	Nephi	8,822.22	8,238	9,420
Kane County	Kanab	10,640.76	6,046	6,532
Millard County	Fillmore	17,684.46	12,405	12,390
Morgan County	Morgan	1,581.94	7,129	8,134
Piute County	Junction	1,983.31	1,435	1,347
Rich County	Randolph	2,813.48	1,961	2,040
Salt Lake County	Salt Lake City	2,092.14	898,387	978,701
San Juan County	Monticello	20,546.61	14,413	14,265
Sanpete County	Manti	4,150.87	22,763	24,196
Sevier County	Richfield	4,968.32	18,842	19,640
Summit County	Coalville	4,874.49	29,736	35,469
Tooele County	Tooele	18,873.55	40,735	53,552
Uintah County	Vernal	11,652.30	25,224	27,955
Utah County	Provo	5,545.11	368,536	464,760
Wasatch County	Heber City	3,131.74	15,215	20,255
Washington County	Saint George	6,293.44	90,354	126,312
Wayne County	Loa	6,388.13	2,509	2,544
Weber County	Ogden	1,707.99	196,533	213,247
total		219,886.96	2,233,169	2,550,063

Table 24: American Indian/Alaska Native Populations (T12)

State	American Indian/Alaska Native alone or in combination with one or more other races			American Indian/Alaska Native alone		
	Number	Percent of total population	Percent of American Indian/Alaska Native total population	Number	Percent of total population	Percent of American Indian/Alaska Native alone population
United States	4,119,301	1.5	100.0	2,475,956	0.9	100.0
California	627,562	1.9	15.2	333,346	1.0	13.5
Oklahoma	391,949	11.4	9.5	273,230	7.9	11.0
Arizona	292,552	5.7	7.1	255,879	5.0	10.3
Texas	215,599	1.0	5.2	118,362	0.6	4.8
New Mexico	191,475	10.5	4.6	173,483	9.5	7.0
New York	171,581	0.9	4.2	82,461	0.4	3.3
Washington	158,940	2.7	3.9	93,301	1.6	3.8
North Carolina	131,736	1.6	3.2	99,551	1.2	4.0
Alaska	119,241	19.0	2.9	98,043	15.6	4.0
Oregon	85,667	2.5	2.1	45,211	1.3	1.8
Minnesota	81,074	1.6	2.0	54,967	1.1	2.2
Colorado	79,689	1.9	1.9	44,241	1.0	1.8
South Dakota	68,281	9.0	1.7	62,283	8.3	2.5
Montana	66,320	7.4	1.6	56,068	6.2	2.3
Nevada	42,222	2.1	1.0	26,240	1.3	1.1
Utah	40,445	1.8	1.0	29,684	1.3	1.2
North Dakota	35,228	5.5	0.9	31,329	4.9	1.3
Idaho	27,237	2.1	0.7	17,645	1.4	0.7
Wyoming	15,012	3.0	0.4	11,133	2.3	0.4
All other states	1,277,491	0.8	31.0	569,499	0.4	23.0

Table 25: General info for Angola (T13)

provincia	capital	area (sq.km.)	population 1991-07 estimate	population 1996-07 estimate
Provincia do Bengo	Caxito	41,000	166,000	190,000
Provincia de Benguela	Benguela	39,827	644,000	718,000
Provincia do Bié	Kuito	70,314	1,125,000	1,280,000
Provincia de Cabinda	Cabinda	7,270	163,000	199,000
Provincia do Cunene	Ondjiva	77,213	232,000	255,000
Provincia do Huambo	Huambo	35,771	1,524,000	1,730,000
Provincia da Huila	Lubango	79,022	869,000	954,000
Provincia do Kuando Kubango	Menongue	199,335	130,000	139,000
Provincia do Kuanza Norte	NDalatando	24,110	378,000	440,000
Provincia do Kuanza Sul	Sumbe	58,698	651,000	710,000
Provincia de Luanda	Luanda	2,257	1,629,000	2,022,000
Provincia da Lunda Norte	Lucapa	103,760	292,000	320,000
Provincia da Lunda Sul	Saurimo	77,637	155,000	165,000
Provincia de Malanje	Malanje	93,302	892,000	1,020,000
Provincia do Moxico	Luenha	223,023	316,000	360,000
Provincia do Namibe	Namibe	57,091	115,000	154,000
Provincia do Uíge	Uíge	58,698	837,000	985,000
Provincia do Zaire	Mbanza Congo	40,130	192,000	262,000
total		1,246,600	10,310,000	11,903,000

Table 26: Bodies of Water (T14)

rank	body of water	area (sq.km.)	maximum depth
1.	Pacific Ocean	179,650,000	11,022 m
2.	Atlantic Ocean	106,100,000	9,212 m
3.	Indian Ocean	74,900,000	7,450 m
4.	Arctic Ocean	13,230,000	5,449 m
5.	South China Sea	2,974,600	5,559 m
6.	Caribbean Sea	2,753,000	7,680 m
7.	Mediterranean Sea	2,510,000	5,150 m
8.	Bering Sea	2,261,000	4,079 m
9.	Gulf of Mexico	1,542,985	4,023 m
10.	Sea of Okhotsk	1,527,570	5,210 m
11.	East China Sea	1,249,150	2,719 m
12.	Sea of Japan / East Sea	1,012,945	4,225 m
13.	Andaman Sea	797,700	3,113 m
14.	Hudson Bay	730,380	218 m
15.	North Sea	575,300	725 m
16.	Red Sea	437,700	2,604 m
17.	Black Sea	436,400	2,244 m
18.	Baltic Sea	414,400	459 m
19.	Caspian Sea	371,800	1,025 m
20.	Yellow Sea	294,000	93 m
21.	Persian Gulf	238,790	75 m
22.	Gulf of California	162,000	1,293 m
23.	Irish Sea	103,600	245 m
24.	English Channel	89,900	72 m
25.	Lake Superior	82,350	405 m

Table 27: Economy of Albania (T15)

Year	Gross Domestic Product	GDP at Purchasing Power Parity	Inflation Index (2000=100)
1980	17,411	4.836	5.10
1985	18,896	6.891	5.10
1990	18,840	8.233	5.10
1995	251,843	8.108	55
2000	530,906	17.483	100
2005	986,833	21.944	121

Table 28: Economy of New Zealand (T16)

Year	Gross Domestic Product	US Dollar Exchange	Inflation Index (2000=100)
1980	22,976	1.02 New Zealand Dollars	30
1985	45,003	2.00 New Zealand Dollars	53
1990	73,745	1.67 New Zealand Dollars	84
1995	91,881	1.52 New Zealand Dollars	93
2000	114,563	2.18 New Zealand Dollars	100
2005	154,108	1.41 New Zealand Dollars	113

Table 29: World population (T17)

rank	country	area sq.km.	population (all estimates)				
			2007-07-01	yearly growth	yesterday	daily increase	today
	World	510,072,000	6,602,224,175	1.17%	6,597,158,008	211,090	6,597,369,098
1.	China	9,596,960	1,321,851,888	0.61%	1,321,325,175	21,946	1,321,347,122
2.	India	3,287,590	1,129,866,154	1.61%	1,128,673,015	49,714	1,128,722,729
3.	United States of America	9,631,418	301,139,947	0.89%	300,962,926	7,376	300,970,302
4.	Indonesia	1,919,440	234,693,997	1.21%	234,506,808	7,800	234,514,607
5.	Brazil	8,511,965	190,010,647	1.01%	189,884,709	5,247	189,889,956
6.	Pakistan	803,940	164,741,924	1.83%	164,543,909	8,251	164,552,159
7.	Bangladesh	144,000	150,448,339	2.06%	150,244,949	8,475	150,253,424
8.	Russia	17,075,200	141,377,752	-0.48%	141,422,745	-1,875	141,420,870
9.	Nigeria	923,768	135,031,164	2.38%	134,819,938	8,801	134,828,739
10.	Japan	377,835	127,433,494	-0.09%	127,440,868	-307	127,440,560
11.	Mexico	1,972,550	108,700,891	1.15%	108,618,481	3,434	108,621,915
12.	Philippines	300,000	91,077,287	1.76%	90,971,647	4,402	90,976,049
13.	Vietnam	329,560	85,262,356	1.00%	85,206,069	2,345	85,208,414
14.	Germany	357,021	82,400,996	-0.03%	82,402,784	-74	82,402,709
15.	Egypt	1,001,450	80,335,036	1.72%	80,244,128	3,788	80,247,915
16.	Ethiopia	1,127,127	76,511,887	2.27%	76,397,585	4,763	76,402,347
17.	Turkey	780,580	71,158,647	1.04%	71,109,986	2,028	71,112,014
18.	Congo, Dem. Rep. of the	2,345,410	65,751,512	3.39%	65,604,949	6,107	65,611,056
19.	Iran	1,648,000	65,397,521	0.66%	65,369,011	1,188	65,370,199
20.	Thailand	514,000	65,068,149	0.66%	65,039,783	1,182	65,040,965
21.	France	547,030	63,713,926	0.59%	63,689,292	1,026	63,690,319
22.	United Kingdom	244,820	60,776,238	0.28%	60,765,248	458	60,765,706
23.	Italy	301,230	58,147,733	0.01%	58,147,351	16	58,147,367
24.	Korea, South	98,480	49,044,790	0.39%	49,032,084	529	49,032,613
25.	Myanmar	678,500	47,373,958	0.82%	47,348,571	1,058	47,349,629
26.	Ukraine	603,700	46,299,862	-0.68%	46,320,412	-856	46,319,555
27.	Colombia	1,138,910	44,379,598	1.43%	44,337,781	1,742	44,339,524
28.	South Africa	1,219,912	43,997,828	-0.46%	44,011,136	-554	44,010,581
29.	Spain	504,782	40,448,191	0.12%	40,445,106	129	40,445,234
30.	Argentina	2,766,890	40,301,927	0.94%	40,277,070	1,036	40,278,106
31.	Tanzania	945,087	39,384,223	2.09%	39,330,073	2,256	39,332,330
32.	Sudan	2,505,810	39,379,358	2.08%	39,325,448	2,246	39,327,694
33.	Poland	312,685	38,518,241	-0.05%	38,519,406	-49	38,519,358
34.	Kenya	582,650	36,913,721	2.80%	36,845,784	2,831	36,848,614
35.	Morocco	446,550	33,757,175	1.53%	33,723,259	1,413	33,724,672
36.	Canada	9,984,670	33,390,141	0.87%	33,371,062	795	33,371,857
37.	Algeria	2,381,740	33,333,216	1.22%	33,306,564	1,110	33,307,675
38.	Afghanistan	647,500	31,889,923	2.63%	31,834,880	2,293	31,837,174
39.	Uganda	236,040	30,262,610	3.57%	30,191,532	2,962	30,194,493
40.	Nepal	140,800	28,901,790	2.13%	28,861,274	1,688	28,862,962
41.	Peru	1,285,220	28,674,757	1.29%	28,650,453	1,013	28,651,466
42.	Uzbekistan	447,400	27,780,059	1.73%	27,748,422	1,318	27,749,740
43.	Saudi Arabia	1,960,582	27,601,038	2.06%	27,563,652	1,558	27,565,210
44.	Iraq	437,072	27,499,638	2.62%	27,452,299	1,972	27,454,272
45.	Venezuela	912,050	26,023,528	1.49%	25,998,101	1,059	25,999,160
46.	Malaysia	329,750	24,821,286	1.76%	24,792,578	1,196	24,793,774
47.	Korea, North	120,540	23,301,725	0.79%	23,289,697	501	23,290,199
48.	Ghana	239,460	22,931,299	1.97%	22,901,565	1,239	22,902,804
49.	Taiwan	35,980	22,858,872	0.30%	22,854,303	190	22,854,493
50.	Romania	237,500	22,276,056	-0.13%	22,277,916	-78	22,277,839

C. Wang Notation

Wang notation for T09 is presented below.

(Province (v),{(Newfoundland and Labrador, {(Smallwood Reservoir,phi), (Melville Lake,phi)}),
(Quebec,{(Lac Mistassini,phi), (Reservoir Manicougan,phi), (Reservoir Gouin,phi), (Lac a`l'Eau-
Claire,phi), (Lac Bienville,phi), (Lac Saint-Jean,phi), (Reservoir Pipmuacan,phi), (Lac Minto,phi),
(Reservoir Cabonga,phi)}), (Manitoba,{(Lake Winnipeg,phi), (Lake Winnipegosis,phi), (Lake
Manitoba,phi), (Southern Indian Lake,phi), (Cedar Lake,phi), (Island Lake,phi), (Gods Lake,phi), (Cross
Lake,phi), (Playgreen Lake,phi)}), (Alberta,{(Lake Clair,phi), (Lesser Slave Lake,phi)}), (British
Columbia,{(Williston Lake,phi), (Atlin Lake,phi)}))

(Info (v),{(Elevation (m),phi),(Area (km),phi)}))

delta({Info (v).Elevation (m),Province (v).Newfoundland and Labrador.Smallwood Reservoir})=471
delta({Info (v).Area (km),Province (v).Newfoundland and Labrador.Smallwood Reservoir})=6527
delta({Info (v).Elevation (m),Province (v).Newfoundland and Labrador.Melville Lake})=tidal
delta({Info (v).Area (km),Province (v).Newfoundland and Labrador.Melville Lake})=3069
delta({Info (v).Elevation (m),Province (v).Quebec.Lac Mistassini})=372
delta({Info (v).Area (km),Province (v).Quebec.Lac Mistassini})=2335
delta({Info (v).Elevation (m),Province (v).Quebec.Reservoir Manicougan})=360
delta({Info (v).Area (km),Province (v).Quebec.Reservoir Manicougan})=1942
delta({Info (v).Elevation (m),Province (v).Quebec.Reservoir Gouin})=404
delta({Info (v).Area (km),Province (v).Quebec.Reservoir Gouin})=1570
delta({Info (v).Elevation (m),Province (v).Quebec.Lac a`l'Eau-Claire})=241
delta({Info (v).Area (km),Province (v).Quebec.Lac a`l'Eau-Claire})=1383
delta({Info (v).Elevation (m),Province (v).Quebec.Lac Bienville})=426
delta({Info (v).Area (km),Province (v).Quebec.Lac Bienville})=1249
delta({Info (v).Elevation (m),Province (v).Quebec.Lac Saint-Jean})=98
delta({Info (v).Area (km),Province (v).Quebec.Lac Saint-Jean})=1003
delta({Info (v).Elevation (m),Province (v).Quebec.Reservoir Pipmuacan})=396
delta({Info (v).Area (km),Province (v).Quebec.Reservoir Pipmuacan})=978
delta({Info (v).Elevation (m),Province (v).Quebec.Lac Minto})=168
delta({Info (v).Area (km),Province (v).Quebec.Lac Minto})=761
delta({Info (v).Elevation (m),Province (v).Quebec.Reservoir Cabonga})=361
delta({Info (v).Area (km),Province (v).Quebec.Reservoir Cabonga})=677
delta({Info (v).Elevation (m),Province (v).Manitoba.Lake Winnipeg})=217
delta({Info (v).Area (km),Province (v).Manitoba.Lake Winnipeg})=24387
delta({Info (v).Elevation (m),Province (v).Manitoba.Lake Winnipegosis})=254
delta({Info (v).Area (km),Province (v).Manitoba.Lake Winnipegosis})=5374
delta({Info (v).Elevation (m),Province (v).Manitoba.Lake Manitoba})=248
delta({Info (v).Area (km),Province (v).Manitoba.Lake Manitoba})=4624
delta({Info (v).Elevation (m),Province (v).Manitoba.Southern Indian Lake})=254
delta({Info (v).Area (km),Province (v).Manitoba.Southern Indian Lake})=2247
delta({Info (v).Elevation (m),Province (v).Manitoba.Cedar Lake})=253
delta({Info (v).Area (km),Province (v).Manitoba.Cedar Lake})=1353
delta({Info (v).Elevation (m),Province (v).Manitoba.Island Lake})=227
delta({Info (v).Area (km),Province (v).Manitoba.Island Lake})=1223
delta({Info (v).Elevation (m),Province (v).Manitoba.Gods Lake})=178
delta({Info (v).Area (km),Province (v).Manitoba.Gods Lake})=1151
delta({Info (v).Elevation (m),Province (v).Manitoba.Cross Lake})=207
delta({Info (v).Area (km),Province (v).Manitoba.Cross Lake})=755

delta({Info (v).Elevation (m),Province (v).Manitoba.Playgreen Lake})=217
 delta({Info (v).Area (km),Province (v).Manitoba.Playgreen Lake})=657
 delta({Info (v).Elevation (m),Province (v).Alberta.Lake Clair})=213
 delta({Info (v).Area (km),Province (v).Alberta.Lake Clair})=1436
 delta({Info (v).Elevation (m),Province (v).Alberta.Lesser Slave Lake})=577
 delta({Info (v).Area (km),Province (v).Alberta.Lesser Slave Lake})=1168
 delta({Info (v).Elevation (m),Province (v).British Columbia.Williston Lake})=671
 delta({Info (v).Area (km),Province (v).British Columbia.Williston Lake})=1761
 delta({Info (v).Elevation (m),Province (v).British Columbia.Atlin Lake})=668
 delta({Info (v).Area (km),Province (v).British Columbia.Atlin Lake})=775

D. XML Representation

XML representation for T09 is presented below.

```

<InterpretedTable xsi:noNamespaceSchemaLocation="G:\RPI\XML\02_TableInterface.XS.070803.xml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Table TableOID="tableOID" Number="1" DocumentCitation="Canada Statistics" Title="Lakes-
  Simulated-Table 0-Ascii" Caption="CAPTIONHERE">
    <CategoryNodes>
      <CategoryNode CategoryNodeOID="C1" Label="Province (v)"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.1" Label="Newfoundland and Labrador"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.1.1" Label="Smallwood Reservoir"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.1.2" Label="Melville Lake"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.2" Label="Quebec"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.2.1" Label="Lac Mistassini"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.2.2" Label="Reservoir Manicougan"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.2.3" Label="Reservoir Gouin"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.2.4" Label="Lac à l'Eau-Claire"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.2.5" Label="Lac Bienville"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.2.6" Label="Lac Saint-Jean"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.2.7" Label="Reservoir Pipmuacan"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.2.8" Label="Lac Minto"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.2.9" Label="Reservoir Cabonga"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.3" Label="Manitoba"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.3.1" Label="Lake Winnipeg"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.3.2" Label="Lake Winnipegosis"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.3.3" Label="Lake Manitoba"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.3.4" Label="Southern Indian Lake"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.3.5" Label="Cedar Lake"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.3.6" Label="Island Lake"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.3.7" Label="Gods Lake"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.3.8" Label="Cross Lake"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.3.9" Label="Playgreen Lake"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.4" Label="Alberta"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.4.1" Label="Lake Clair"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.4.2" Label="Lesser Slave Lake"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.5" Label="British Columbia"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.5.1" Label="Williston Lake"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.5.2" Label="Atlin Lake"></CategoryNode>
      <CategoryNode CategoryNodeOID="C2" Label="Info (v)"></CategoryNode>
      <CategoryNode CategoryNodeOID="C2.1" Label="Elevation (m)"></CategoryNode>
      <CategoryNode CategoryNodeOID="C2.2" Label="Area (km)"></CategoryNode>
    </CategoryNodes>
  </Table>
</InterpretedTable>
  
```



```

</CategoryNodes>
</CategoryParentNode>
<CategoryParentNode CategoryParentNodeOID="C1.5">
  <CategoryNodes>
    <CategoryNode CategoryNodeOID="C1.5.1"></CategoryNode>
    <CategoryNode CategoryNodeOID="C1.5.2"></CategoryNode>
  </CategoryNodes>
</CategoryParentNode>
</CategoryParentNodes>
<DataCells>
  <DataCell DataCellOID="D1" DataValue="471">
    <CategoryLeafNode CategoryLeafNodeOID="C1.1.1"></CategoryLeafNode>
    <CategoryLeafNode CategoryLeafNodeOID="C2.1"></CategoryLeafNode>
  </DataCell>
  <DataCell DataCellOID="D2" DataValue="6527">
    <CategoryLeafNode CategoryLeafNodeOID="C1.1.1"></CategoryLeafNode>
    <CategoryLeafNode CategoryLeafNodeOID="C2.2"></CategoryLeafNode>
  </DataCell>
  <DataCell DataCellOID="D3" DataValue="tidal">
    <CategoryLeafNode CategoryLeafNodeOID="C1.1.2"></CategoryLeafNode>
    <CategoryLeafNode CategoryLeafNodeOID="C2.1"></CategoryLeafNode>
  </DataCell>
  <DataCell DataCellOID="D4" DataValue="3069">
    <CategoryLeafNode CategoryLeafNodeOID="C1.1.2"></CategoryLeafNode>
    <CategoryLeafNode CategoryLeafNodeOID="C2.2"></CategoryLeafNode>
  </DataCell>
  <DataCell DataCellOID="D5" DataValue="372">
    <CategoryLeafNode CategoryLeafNodeOID="C1.2.1"></CategoryLeafNode>
    <CategoryLeafNode CategoryLeafNodeOID="C2.1"></CategoryLeafNode>
  </DataCell>
  <DataCell DataCellOID="D6" DataValue="2335">
    <CategoryLeafNode CategoryLeafNodeOID="C1.2.1"></CategoryLeafNode>
    <CategoryLeafNode CategoryLeafNodeOID="C2.2"></CategoryLeafNode>
  </DataCell>
  <DataCell DataCellOID="D7" DataValue="360">
    <CategoryLeafNode CategoryLeafNodeOID="C1.2.2"></CategoryLeafNode>
    <CategoryLeafNode CategoryLeafNodeOID="C2.1"></CategoryLeafNode>
  </DataCell>
  <DataCell DataCellOID="D8" DataValue="1942">
    <CategoryLeafNode CategoryLeafNodeOID="C1.2.2"></CategoryLeafNode>
    <CategoryLeafNode CategoryLeafNodeOID="C2.2"></CategoryLeafNode>
  </DataCell>
  <DataCell DataCellOID="D9" DataValue="404">
    <CategoryLeafNode CategoryLeafNodeOID="C1.2.3"></CategoryLeafNode>
    <CategoryLeafNode CategoryLeafNodeOID="C2.1"></CategoryLeafNode>
  </DataCell>
  <DataCell DataCellOID="D10" DataValue="1570">
    <CategoryLeafNode CategoryLeafNodeOID="C1.2.3"></CategoryLeafNode>
    <CategoryLeafNode CategoryLeafNodeOID="C2.2"></CategoryLeafNode>
  </DataCell>
  <DataCell DataCellOID="D11" DataValue="241">
    <CategoryLeafNode CategoryLeafNodeOID="C1.2.4"></CategoryLeafNode>
    <CategoryLeafNode CategoryLeafNodeOID="C2.1"></CategoryLeafNode>
  </DataCell>

```


[illegible]

[illegible]

```

<DataCell DataCellOID="D39" DataValue="217">
  <CategoryLeafNode CategoryLeafNodeOID="C1.3.9"></CategoryLeafNode>
  <CategoryLeafNode CategoryLeafNodeOID="C2.1"></CategoryLeafNode>
</DataCell>
<DataCell DataCellOID="D40" DataValue="657">
  <CategoryLeafNode CategoryLeafNodeOID="C1.3.9"></CategoryLeafNode>
  <CategoryLeafNode CategoryLeafNodeOID="C2.2"></CategoryLeafNode>
</DataCell>
<DataCell DataCellOID="D41" DataValue="213">
  <CategoryLeafNode CategoryLeafNodeOID="C1.4.1"></CategoryLeafNode>
  <CategoryLeafNode CategoryLeafNodeOID="C2.1"></CategoryLeafNode>
</DataCell>
<DataCell DataCellOID="D42" DataValue="1436">
  <CategoryLeafNode CategoryLeafNodeOID="C1.4.1"></CategoryLeafNode>
  <CategoryLeafNode CategoryLeafNodeOID="C2.2"></CategoryLeafNode>
</DataCell>
<DataCell DataCellOID="D43" DataValue="577">
  <CategoryLeafNode CategoryLeafNodeOID="C1.4.2"></CategoryLeafNode>
  <CategoryLeafNode CategoryLeafNodeOID="C2.1"></CategoryLeafNode>
</DataCell>
<DataCell DataCellOID="D44" DataValue="1168">
  <CategoryLeafNode CategoryLeafNodeOID="C1.4.2"></CategoryLeafNode>
  <CategoryLeafNode CategoryLeafNodeOID="C2.2"></CategoryLeafNode>
</DataCell>
<DataCell DataCellOID="D45" DataValue="671">
  <CategoryLeafNode CategoryLeafNodeOID="C1.5.1"></CategoryLeafNode>
  <CategoryLeafNode CategoryLeafNodeOID="C2.1"></CategoryLeafNode>
</DataCell>
<DataCell DataCellOID="D46" DataValue="1761">
  <CategoryLeafNode CategoryLeafNodeOID="C1.5.1"></CategoryLeafNode>
  <CategoryLeafNode CategoryLeafNodeOID="C2.2"></CategoryLeafNode>
</DataCell>
<DataCell DataCellOID="D47" DataValue="668">
  <CategoryLeafNode CategoryLeafNodeOID="C1.5.2"></CategoryLeafNode>
  <CategoryLeafNode CategoryLeafNodeOID="C2.1"></CategoryLeafNode>
</DataCell>
<DataCell DataCellOID="D48" DataValue="775">
  <CategoryLeafNode CategoryLeafNodeOID="C1.5.2"></CategoryLeafNode>
  <CategoryLeafNode CategoryLeafNodeOID="C2.2"></CategoryLeafNode>
</DataCell>
</DataCells>
</InterpretedTable>

```

E. Log

An example log for T09 is presented below.

Table 30: Example Log

Event	Time			Time Elapsed		
	hr	min	sec	hr	min	sec
Start Time	17	35	1.515	0	0	0
Acquire ASCII	17	35	1.562	0	0	0.047
Display Original HTML file	17	35	1.625	0	0	0.063
GUI Generated	17	35	2.25	0	0	0.625
Province (click)	17	35	5.484	0	0	3.234
Atlin Lake (click)	17	35	6.703	0	0	1.219
Range/Region (click)	17	35	6.891	0	0	0.188
Range/Region (unclick)	17	35	7.231	0	0	0.34
Elevation (m) (click)	17	35	7.875	0	0	0.644
Area (km) (click)	17	35	8.469	0	0	0.594
All Categories Clicked	17	35	10.547	0	0	2.078
Category Displayed	17	35	11.328	0	0	0.781
Delete Row	17	35	17.265	0	0	5.937
Delete Row	17	35	20.562	0	0	3.297
Rename Cell	17	35	22.797	0	0	2.235
Notation is Correct	17	35	31.344	0	0	8.547
Category Notation Determined	17	35	31.437	0	0	0.093
Category Displayed	17	35	31.844	0	0	0.407
Rename Cell	17	35	35.109	0	0	3.265
Delete Column	17	35	44.656	0	0	9.547
Add Virtual Header	17	35	46.453	0	0	1.797
Notation is Correct	17	35	54.937	0	0	8.484
Category Notation Determined	17	35	54.969	0	0	0.032
CATEGORY NOTATION	17	35	54.969	0	0	0
DELTA NOTATION	17	35	55.172	0	0	0.203
User Entered Table Info	17	35	55.187	0	0	0.015
XML REPRESENTATION	17	35	55.719	0	0	0.532

F. Training PowerPoint

WNT TRAINING

Wang Notation Tool

Developed by
Piyushee Jha

Acknowledgments:

National Science Foundation
Rensselaer Polytechnic Institute
Brigham Young University

What is WNT?

- Purpose: convert tables to layout independent form so several tables can be merged to create ontologies
 - Semantic Web
- Input: tables from the web (HTML)
- Output: Wang notation, XML schema

2

Wang Notation & XML

Year	Term	Mark					
		Assignments			Examinations		Grade
		Ass1	Ass2	Ass3	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

- Wang Category Notation

```
(Year, {(1991,φ), (1992,φ)})
(Term, {(Winter,φ), (Spring,φ), (Fall,φ)})
(Mark, {(Assignments, {(Ass1,φ), (Ass2,φ), (Ass3,φ)}),
        (Examinations, {(Midterm,φ), (Final,φ)}), (Grade,φ)})
```

- Wang Delta Notation (1st row)

```
δ{Year.1991, Term.Winter, Mark.Assignments.Ass1} = 85
δ{Year.1991, Term.Winter, Mark.Assignments.Ass2} = 80
δ{Year.1991, Term.Winter, Mark.Assignments.Ass3} = 75
δ{Year.1991, Term.Winter, Mark.Examinations.Midterm} = 60
δ{Year.1991, Term.Winter, Mark.Examinations.Final} = 75
δ{Year.1991, Term.Winter, Mark.Grade} = 75
```

- XML Document (very small portion shown)

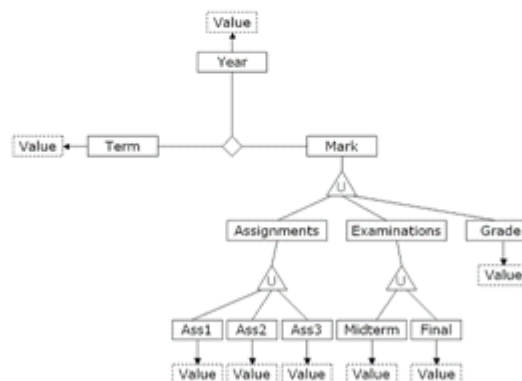
```
<CategoryParentNode>
  <CategoryParentNode CategoryParentNodeOID="C1">
    <CategoryNode>
      <CategoryNode CategoryNodeOID="C1.1"></CategoryNode>
      <CategoryNode CategoryNodeOID="C1.2"></CategoryNode>
    </CategoryNode>
    <CategoryParentNode CategoryParentNodeOID="C1">
      <CategoryNode>
        <CategoryNode CategoryNodeOID="C1.1"></CategoryNode>
        <CategoryNode CategoryNodeOID="C1.2"></CategoryNode>
        <CategoryNode CategoryNodeOID="C1.3"></CategoryNode>
      </CategoryNode>
    </CategoryParentNode>
```

...

3

Ontology

- Data model that contains a set of concepts and the relation between those concepts



4

Why WNT?

- Difficult to automatically determine Wang notation and XML schema for several tables
- Tables are different depending on who creates them so the interactive Wang Notation Tool (WNT) was developed
- WNT is semi-automatic, requires user input and an understanding of tables to make corrections

5

Table Concepts – Category Cells

- Cells that show what the information in the table represents
- Categories are Year, Term, and Mark.
- Subcategories are 1991 & 1992 for Year, Winter, Spring, Fall for Term and so on.
- Each category is contained within a RECTANGLE in the table

Year	Term	Mark					
		Assignments			Examinations		Grade
		Ass1	Ass2	Ass3	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

6

Categories as Trees

- Every category can be thought of as a tree



7

Table Concepts – Delta Cells

- Cells that have information (all cells that are not category cells)
- Marked in green below
- Users do not have to tell WNT which cells are delta cell; users are only responsible for category cells

Year	Term	Mark					
		Assignments			Examinations		Grade
		Ass1	Ass2	Ass3	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

8

Using WNT

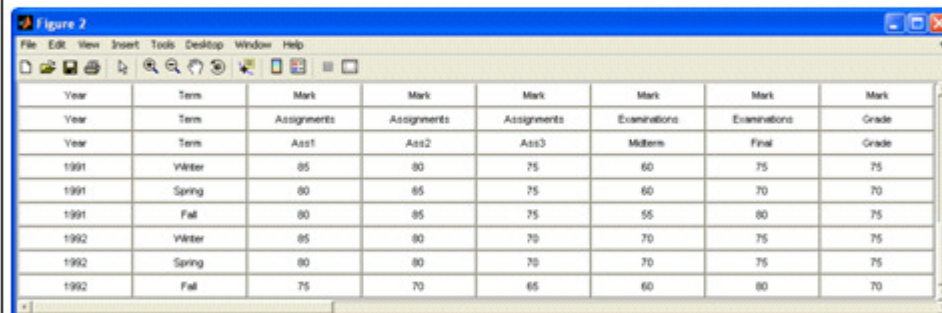
- WNT is programmed in Matlab.
- First image to pop-up is shown below
- User must enter their name in Matlab command window
- Original table will be displayed while a table is being processed



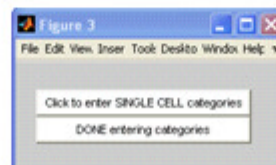
9

Using WNT (2)

- Next the original clickable table appears



Year	Term	Mark	Mark	Mark	Mark	Mark	Mark
Year	Term	Assignments	Assignments	Assignments	Examinations	Examinations	Grade
Year	Term	Ass1	Ass2	Ass3	Midterm	Final	Grade
1991	Winter	85	80	75	60	75	75
1991	Spring	80	85	75	60	70	70
1991	Fall	80	85	75	55	60	75
1992	Winter	85	80	70	70	75	75
1992	Spring	80	80	70	70	75	75
1992	Fall	75	70	65	60	80	70



10

Choosing Categories

- User will select categories by clicking on the top left and bottom right corners of the rectangle containing a category (black cells)
- User can undo mistakes by clicking a cell again but ONLY in reverse order of cells clicked

Figure 1

Year	Term	Mark	Mark	Mark	Mark	Mark	Mark
Year	Term	Assignments	Assignments	Assignments	Examinations	Examinations	Grade
Year	Term	Ass1	Ass2	Ass3	Midterm	Final	Grade
1991	Winter	85	80	75	60	75	75
1991	Spring	80	85	75	60	70	70
1991	Fall	80	85	75	55	80	75
1992	Winter	85	80	70	70	75	75
1992	Spring	80	80	70	70	75	75
1992	Fall	75	70	65	60	80	70

Trees & Indented Notation

- Each category is displayed for approval/correction
- Correct categories using Figure 3 to reflect what you think the correct "tree" for the category is

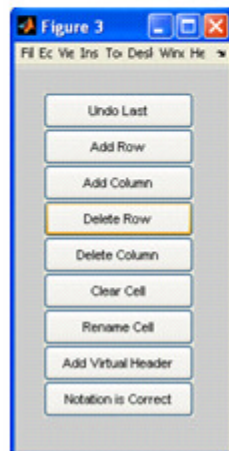


Figure 2

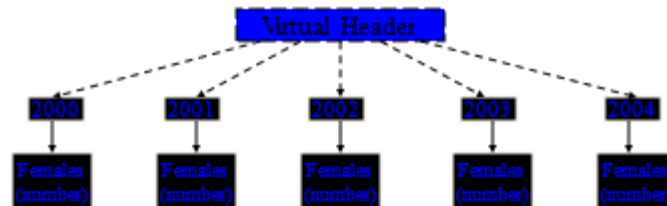
Mark	Assignments	Examinations	Grade
	Ass1		
	Ass2		
	Ass3		
	Midterm		
	Final		



Virtual Headers

- A category without a distinct header needs a virtual header
- It is a “rootless” tree

	2000	2001	2002	2003	2004
	Females number				
Canada	103,326	105,207	111,027	118,467	124,830
Newfoundland and Labrador	1,773	1,755	1,749	1,830	1,989
Prince Edward Island	345	411	363	429	453
Nova Scotia	4,542	4,677	4,857	5,304	5,769
New Brunswick	2,424	2,460	2,670	2,811	3,000
Quebec	29,706	30,144	32,358	34,161	36,384
Ontario	39,297	39,972	41,982	45,042	47,862
Manitoba	3,114	3,183	3,366	3,612	3,897
Saskatchewan	3,408	3,378	3,393	3,486	3,528
Alberta	8,106	8,961	9,483	10,269	10,758
British Columbia	10,614	10,263	10,803	11,520	11,196



13

Virtual Headers (2)

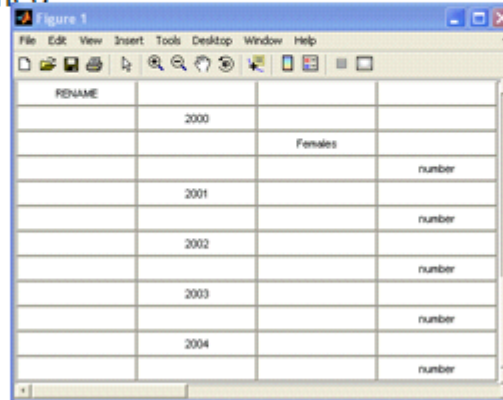
- Both categories below (yellow and green) require virtual headers

?	2000	2001	2002	2003	2004
?	Females	Females	Females	Females	Females
?	number	number	number	number	number
Canada	103,326	105,207	111,027	118,467	124,830
Newfoundland and La.	1,773	1,755	1,749	1,830	1,989
Prince Edward Island	345	411	363	429	453
Nova Scotia	4,542	4,677	4,857	5,304	5,769
New Brunswick	2,424	2,460	2,670	2,811	3,000
Quebec	29,706	30,144	32,358	34,161	36,384
Ontario	39,297	39,972	41,982	45,042	47,862
Manitoba	3,114	3,183	3,366	3,612	3,897
Saskatchewan	3,408	3,378	3,393	3,486	3,528
Alberta	8,106	8,961	9,483	10,269	10,758
British Columbia	10,614	10,263	10,803	11,520	11,196

14

Virtual Header (3)

- Sometimes virtual headers are added automatically by WNT and just need to be renamed

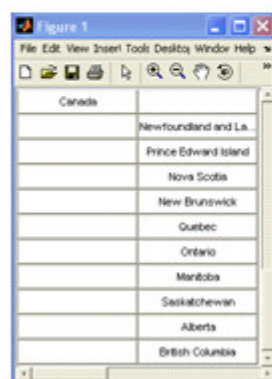


RENAME	2000	Females	number
	2001		number
	2002		number
	2003		number
	2004		number

15

Virtual Headers (4)

- Other times, they need to be added manually using the Error Correction GUI (Figure 2)



Canada	Newfoundland and La...
	Prince Edward Island
	Nova Scotia
	New Brunswick
	Quebec
	Ontario
	Manitoba
	Saskatchewan
	Alberta
	British Columbia

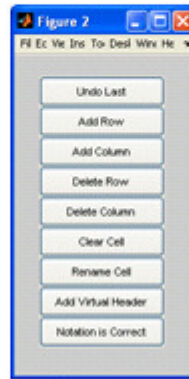


Figure 2

File Edit View Insert Tools Desktop Window Help

Undo Last

Add Row

Add Column

Delete Row

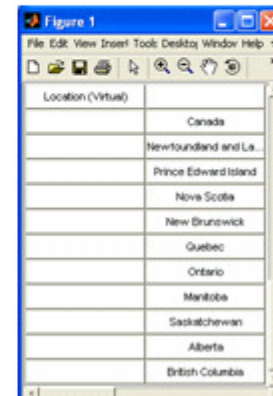
Delete Column

Clear Cell

Rename Cell

Add Virtual Header

Notation is Correct



Location (Virtual)	Canada
	Newfoundland and La...
	Prince Edward Island
	Nova Scotia
	New Brunswick
	Quebec
	Ontario
	Manitoba
	Saskatchewan
	Alberta
	British Columbia

16

Verification

- Before moving on to the next table, the user can verify that the table was interpreted correctly with the aid of a final GUI (shown on the next 2 slides)
- After user is satisfied, they can move onto the next table
- If they are not satisfied, they can start over

17

Verification (2)

- Any delta cell that is clicked (blue) will “light” up its corresponding category cells.

Figure 2

Year	Term	Mark	Mark	Mark	Mark	Mark	Mark
Year	Term	Assignments	Assignments	Assignments	Examinations	Examinations	Grade
Year	Term	Ass1	Ass2	Ass3	Midterm	Final	Grade
1991	Winter	85	80	75	60	75	75
1991	Spring	80	65	75	60	70	70
1991	Fall	80	85	75	55	80	75
1992	Winter	85	80	75	75	75	75
1992	Spring	80	80	70	70	75	75
1992	Fall	75	70	65	60	80	70

Figure 2

Year	Term	Mark	Mark	Mark	Mark	Mark	Mark
Year	Term	Assignments	Assignments	Assignments	Examinations	Examinations	Grade
Year	Term	Ass1	Ass2	Ass3	Midterm	Final	Grade
1991	Winter	85	80	75	60	75	75
1991	Spring	80	65	75	60	70	70
1991	Fall	80	85	75	55	80	75
1992	Winter	85	80	75	75	75	75
1992	Spring	80	80	70	70	75	75
1992	Fall	75	70	65	60	80	70

18

Verification (3)

- Any category cell that is clicked (blue) will “light” up its corresponding category and delta cells.

Figure 2

Year	Term	Mark	Mark	Mark	Mark	Mark	Mark
Year	Term	Ass1	Ass2	Ass3	Midterm	Final	Grade
1991	Winter	85	88	75	68	75	75
1991	Spring	80	85	75	60	70	70
1991	Fall	80	85	75	55	80	75
1992	Winter	85	88	75	75	75	75
1992	Spring	80	85	70	70	75	75
1992	Fall	75	70	65	60	80	70

Figure 2

Year	Term	Mark	Mark	Mark	Mark	Mark	Mark
Year	Term	Ass1	Ass2	Ass3	Midterm	Final	Grade
1991	Winter	85	88	75	68	75	75
1991	Spring	80	85	75	60	70	70
1991	Fall	80	85	75	55	80	75
1992	Winter	85	88	75	75	75	75
1992	Spring	80	85	70	70	75	75
1992	Fall	75	70	65	60	80	70

19

End of Interaction

No further interaction is required of the user.

Now, I will demo a few tables using WNT.

20

G. Quantitative Results

Table 31: Distribution of Processing Time for T01, Average Over All Subjects

	AVERAGE	STD. DEVIATION
# of attempts	1.00	0.00
Time for Pre-Processing	0.61	0.97
Time to Construct Categories	36.54	14.66
Time for Category Correction	35.68	16.91
Time for Final Processing	0.36	0.12
Total Time	73.20	27.94
Percent Table is Completed	100.00	0.00
% Subject Time	0.98	0.02

Table 32: Distribution of Processing Time for T02, Average Over All Subjects

	AVERAGE	STD. DEVIATION
# of attempts	1.00	0.00
Time for Pre-Processing	0.52	0.17
Time to Construct Categories	28.70	13.61
Time for Category Correction	40.98	15.72
Time for Final Processing	0.59	0.11
Total Time	70.80	22.09
Percent Table is Completed	100.00	0.00
% Subject Time	0.98	0.01

Table 33: Distribution of Processing Time for T03, Average Over All Subjects

	AVERAGE	STD. DEVIATION
# of attempts	1.00	0.00
Time for Pre-Processing	0.54	0.14
Time to Construct Categories	26.86	14.54
Time for Category Correction	34.09	13.37
Time for Final Processing	0.57	0.06
Total Time	62.06	24.81
Percent Table is Completed	100.00	0.00
% Subject Time	0.98	0.01

Table 34: Distribution of Processing Time for T04, Average Over All Subjects

	AVERAGE	STD. DEVIATION
# of attempts	1.00	0.00
Time for Pre-Processing	0.51	0.14
Time to Construct Categories	24.95	15.81
Time for Category Correction	117.64	57.01
Time for Final Processing	0.27	0.10
Total Time	143.38	64.44
Percent Table is Completed	66.67	49.24
% Subject Time	0.99	0.01

Table 35: Distribution of Processing Time for T05, Average Over All Subjects

	AVERAGE	STD. DEVIATION
# of attempts	1.33	0.49
Time for Pre-Processing	0.56	0.12
Time to Construct Categories	26.72	10.27
Time for Category Correction	39.17	28.44
Time for Final Processing	0.34	0.07
Total Time	66.79	36.14
Percent Table is Completed	95.83	14.43
% Subject Time	0.98	0.01

Table 36: Distribution of Processing Time for T06, Average Over All Subjects

	AVERAGE	STD. DEVIATION
# of attempts	1.00	0.00
Time for Pre-Processing	0.40	0.11
Time to Construct Categories	13.95	5.82
Time for Category Correction	27.05	16.15
Time for Final Processing	0.33	0.06
Total Time	41.73	20.53
Percent Table is Completed	100.00	0.00
% Subject Time	0.98	0.01

Table 37: Distribution of Processing Time for T07, Average Over All Subjects

	AVERAGE	STD. DEVIATION
# of attempts	1.00	0.00
Time for Pre-Processing	0.41	0.10
Time to Construct Categories	17.37	10.98
Time for Category Correction	30.45	21.09
Time for Final Processing	0.32	0.12
Total Time	48.56	29.91
Percent Table is Completed	91.67	28.87
% Subject Time	0.98	0.01

Table 38: Distribution of Processing Time for T08, Average Over All Subjects

	AVERAGE	STD. DEVIATION
# of attempts	1.00	0.00
Time for Pre-Processing	0.47	0.10
Time to Construct Categories	17.78	4.85
Time for Category Correction	23.99	8.74
Time for Final Processing	0.53	0.08
Total Time	42.76	9.04
Percent Table is Completed	100.00	0.00
% Subject Time	0.98	0.01

Table 39: Distribution of Processing Time for T10, Average Over All Subjects

	AVERAGE	STD. DEVIATION
# of attempts	1.50	0.52
Time for Pre-Processing	0.50	0.13
Time to Construct Categories	53.35	21.44
Time for Category Correction	130.82	241.12
Time for Final Processing	0.45	0.10
Total Time	185.11	240.50
Percent Table is Completed	83.33	32.57
% Subject Time	0.99	0.01

Table 40: Distribution of Processing Time for T11, Average Over All Subjects

	AVERAGE	STD. DEVIATION
# of attempts	1.83	0.58
Time for Pre-Processing	0.55	0.15
Time to Construct Categories	55.12	19.53
Time for Category Correction	55.60	23.45
Time for Final Processing	1.04	0.35
Total Time	112.27	40.83
Percent Table is Completed	90.28	22.98
% Subject Time	0.98	0.01

Table 41: Distribution of Processing Time for T12, Average Over All Subjects

	AVERAGE	STD. DEVIATION
# of attempts	1.36	0.50
Time for Pre-Processing	0.63	0.24
Time to Construct Categories	48.99	28.94
Time for Category Correction	45.79	14.47
Time for Final Processing	0.74	0.21
Total Time	96.16	33.57
Percent Table is Completed	90.91	30.15
% Subject Time	0.98	0.01

Table 42: Distribution of Processing Time for T13, Average Over All Subjects

	AVERAGE	STD. DEVIATION
# of attempts	1.17	0.39
Time for Pre-Processing	0.53	0.15
Time to Construct Categories	31.28	16.13
Time for Category Correction	46.23	19.75
Time for Final Processing	0.72	0.14
Total Time	78.75	34.10
Percent Table is Completed	100.00	0.00
% Subject Time	0.98	0.01

Table 43: Distribution of Processing Time for T14, Average Over All Subjects

	AVERAGE	STD. DEVIATION
# of attempts	1.25	0.45
Time for Pre-Processing	0.50	0.14
Time to Construct Categories	28.57	13.43
Time for Category Correction	57.80	54.98
Time for Final Processing	0.47	0.26
Total Time	87.34	59.85
Percent Table is Completed	91.67	28.87
% Subject Time	0.98	0.03

Table 44: Distribution of Processing Time for T15, Average Over All Subjects

	AVERAGE	STD. DEVIATION
# of attempts	1.33	0.49
Time for Pre-Processing	0.43	0.16
Time to Construct Categories	18.79	7.83
Time for Category Correction	34.38	15.71
Time for Final Processing	0.19	0.04
Total Time	53.79	22.48
Percent Table is Completed	95.83	14.43
% Subject Time	0.98	0.01

Table 45: Distribution of Processing Time for T16, Average Over All Subjects

	AVERAGE	STD. DEVIATION
# of attempts	1.50	0.52
Time for Pre-Processing	0.29	0.10
Time to Construct Categories	18.59	10.85
Time for Category Correction	29.91	11.11
Time for Final Processing	0.21	0.04
Total Time	48.99	17.82
Percent Table is Completed	100.00	0.00
% Subject Time	0.99	0.01

Table 46: Distribution of Processing Time for T17, Average Over All Subjects

	AVERAGE	STD. DEVIATION
# of attempts	1.42	0.51
Time for Pre-Processing	1.35	0.20
Time to Construct Categories	75.42	17.47
Time for Category Correction	134.74	81.22
Time for Final Processing	2.97	1.93
Total Time	214.46	92.22
Percent Table is Completed	87.50	31.08
% Subject Time	0.98	0.01