ESHMAWI, ALA A., M.S. Integrating Uncertain XML Data from Different Sources. (2009)
Directed by Dr. Fereidoon Sadri. 32 pp.

Data Integration has become increasingly important with today's rapid growth of information available on the web and in electronic form. In the past several years, extensive work has been done to make use of the available data from different sources, particularly, in the scientific and medical fields. In our work, we are interested in integrating data from different uncertain sources where data are stored in semistructured databases, markedly XML-based data. This interest in XML-based databases came from the flexibility it provides for storing and exchanging data. Furthermore, we are concerned with reliability of different query answers from various sources and on specifying the source where the data came from (the provenance). In essence, our work lies among three areas of interest, data integration, uncertain databases and lineage or provenance in databases. This thesis extends previous work on information integration to accommodate integration of uncertain data from multiple sources.

INTEGRATING UNCERTAIN XML DATA FROM DIFFERENT SOURCES

by

Ala Eshmawi

A Thesis Submitted to
the Faculty of The Graduate School at
The University of North Carolina at Greensboro
in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Greensboro
2009

Approved by

Committee Chair

To my father who was a self-made man of extraordinary magnitude and who always inspired me and pushed me to be what I am today.
Your spirit will always be alive in the hearts of all of us whose lives you have touched.
May you rest in peace.

To my mother who continues to be my best friend and my guidance in each step.
You have and will always be my role model in your strength and dedication.
I wish you all the happiness in the world.

To my loving husband who stands by me in rest and hardship.
Who his support, encouragement and constant love sustains me throughout my life.
Who always believes in me.
Thank you from all of my heart.

To my beloved kids Shaimaa, Baraa and Owais
Who are the light of my life.
Who fill each day of my life with so much love and laughter.
May God bless you.

APPROVAL PAGE

This thesis has been approved by the following committee of the Faculty of The Graduate School at The University of North Carolina at Greensboro.

Committee Chair_____

Committee Members_____

_____

_____
Date of Acceptance by Committee

_____
Date of Final Oral Examination

ACKNOWLEDGMENTS

This research thesis would not have been possible without the support of my supervisor Dr. Fereidoon Sadri who was abundantly helpful and offered invaluable assistance, support and guidance.

Deepest gratitude to the Ministry Of Higher Education of Saudi Arabia for giving me this opportunity to continue my studies and for their financial support.

TABLE OF CONTENTS

Page

CHAPTER I

INTRODUCTION

Data Integration has become increasingly important with today's rapid growth of information available on the web and in electronic form. In the past several years, extensive work has been done to make use of the available data from different sources. Particularly, in the scientific and medical fields. In our work, we are interested in integrating data from different uncertain sources where data are stored in semistructured databases, markedly XML-based data. This interest in XML-based databases came from the flexibility it provides for storing data and exchanging it through out the web. Furthermore, we are concerned with reliability of different query answers emerged from various sources and on specifying the source where the data came from (the provenance). In essence, our work lies among three areas of interest, data integration, uncertain databases and lineage or provenance in databases. We based our work on [8]. We extend their first two algorithms for data integration ,to handle probabilities and lineage.

In the Materialization approach we add extra information to the semantic model to represent the paths to help computing the confidences afterwards. Then we add extensions to the basic algebra to manipulate paths expressions to produce the correct confidence for the user query. In the subquery approach we introduce the flat representation for keeping track of the lineage and global mutual exclusion. We present an algorithm to calculate confidences of the answers to a user query. Our extensions were inspired from the work done by [12] and [10] in which computing confidences in the relational context depend on the lineage of the resulted answer.

Computing confidences are managed also in semistructured databases as shown in [2]. In the next section, we highlight some of the previous work done in lineage, uncertain databases and in data integration. Then, we will present our work in extending data integration algorithms in [8] to handle probabilities and lineage. Finally, we conclude with some of the open problems and future work.

CHAPTER II

PRELIMINARIES AND RELATED WORK

Our work lies among three fields: provenance, uncertain databases and data integration. We will highlight in the following three sections some of the work done in each of these areas of interest.

2.1 Provenance

The provenance, or the source of the data, is essential in many real life applications such as scientific and data integration applications. In many cases, users of such applications need to know where the answers to their queries came from to give them some indication of the degree of trust with which they can rely on this result. In [16], a survey of provenance in databases was presented. The writer discussed different types of provenance in databases, the *workflow* and the *data provenance*. In the *workflow*, details of the programs deriving the result are hidden and are treated as blackboxes. Whereas, in the *data provenance*, details of how a piece of data in the result that was produced are given. According to [16], most of the work done to keep track of the provenance adopted one of two approaches. *Annotated approach*, in which an extra piece of information is added to the original data to help in computing the provenance. Conversely in *Unannotated approach* queries are executed as they are and the provenance is computed by analyzing the underlying definition of the query. In [7], extensions to the relational model were proposed to represent and manipulate the provenance of data from different

sources for the goal of deciding the reliability of the user query answers. In [13], the authors were the first to differentiate between *where* and *why* provenance. The *why* provenance refers to the parts of the original database that contributed to the produced result. While, the *where* provenance specifies which part of the original database was copied to the result. Moreover, the authors presented their approach to compute the provenance for the result of a user query that can be applied to both relational and semistructured data models. Now, consider that we have the source where our result emerged from, what amount of trust can we put on this result? That depends on how reliable our source is, and how probable the result is to satisfy our query. This is where the importance of probabilities, confidences and hence, uncertain databases arise.

2.2 Uncertain databases and Probabilities

a vast amount of work has been done towards the goal of representing probabilities for uncertain data in both relational and semistructured models. We will briefly outline some of this work.

2.2.1 In The Relational Context

In [12], ULDB's were introduced to be the building blocks for the Trio system developed at Stanford University. See [10] for more information on the Trio project. ULDB's were the first to use lineage tracking approach to compute confidence, and from ULDBs' our idea of keeping the paths emerged, but we do this for the semistructured data models instead of the relational model. In [3], the authors discussed complete and incomplete models. In addition, they established a strict hierarchy based on the expressive powers of different incomplete models, and a full description of the closure properties of each of these models. Complete models can

be complicated and unintuitive in contrast with incomplete models. Incomplete models are more simple and are easily understood, hence these models are more attractive for many applications; the authors suggested a model of two layers that benefits from both complete and incomplete models.

2.2.2 In The Semistructured Context

In [1], SPO's (Semistructured Probabilistic Objects) were introduced, and the "probability distribution" of one or more random variables are the objects that need to be stored. This object is different from "real" objects in the probabilistic object models, in which objects are real and some of their properties are uncertain. For example, an SPO can store the probability distribution of a student's performance in one course. More complicated situations can be stored also using the joint probability distribution, contexts and conditions. According to the authors, these latter models along with probabilistic relational models are not flexible enough to store uncertain data in straight forward manners. They also originated a new semistructured algebra to manipulate probabilities in their model. In [17], they discussed implementation issues of SPO's framework based on XML and how to efficiently map XML data into SPO's. In [5], Probabilistic Interval XML data model was introduced for incorporating probabilities in XML databases in which they use two semantics; The global intuitive model which is not computationally efficient, and the local more efficient model.

In [15] a more recent work, the authors presented the Fuzzy Tree model based on probabilistic events variables. They proved that their model is as expressive as the impractical possible world model, and yet more concise. In [14] a full complexity analysis and mathematical foundation for the Fuzzy Trees were proposed. Fuzzy Trees can capture more dependencies than simple models, but it is much more

complex, especially for applications where "the updating part" is not needed.

Finally, we chose to work with [2], because this model was (from our point of view), the simplest and more intuitive than SPO's, Fuzzy trees and other models. Moreover, this model can capture dependencies that are natural within most of today's applications. In [2], they present the probabilistic tree database to manage probabilities in semistructured data such as XML. They address in their model several challenges of XML data such as probabilities in multiple granularities, and missing data (incompleteness). They introduce three different constructs to accommodate probabilities and to capture different dependencies. The probability attribute Prob that takes a value between 0 and 1 (inclusive). If an element does not contain the Prob attribute, then its probability is assumed to be equal to 1. Dist construct that may have multiple Val elements as children, each with an associated probability. This Dist construct can record dependencies between its values. Possible distribution types include mutually-exclusive and independent. In ProTDB they define the probability of an element to be the probability that the state of the world includes this element and the subtree rooted at it.

Consider a chain $A \rightarrow B \rightarrow C$ from the root node A. the source XML document will contain the probabilities $Pro(C|B)$, $Prob(B|A)$, and $Prob(A)$, associated to the nodes C, B, And A, respectively. We have the formulas :

$$P(AandB) = P(A) * P(B|A)$$

$$P(AandB) = P(B) * P(A|B)$$

$$P(B) * P(A|B) = P(A) * P(B|A)$$

Now we want to calculate the probability of B (for example) in response to a query. We have that the probability of the parent is 1.0 if the child exist so, $P(A|B) = 1$, so we get that:

$$P(B) = P(A) * P(B|A)$$

In general, the probability of an element e can be found by multiplying the conditional probabilities found in source XML, along the path from e to the root.

2.3 Data Integration

Many researchers are interested in integrating data from different sources either to make use of the abundant amount of data available on the web, or for other application specific needs. Here we will consider [9]and [8]to base our work for extensions with probabilities and lineages. In [9], a new approach for integrating XML data from different sources was introduced. This approach uses an application specific semantic model to allow interoperability between sources using common applications. In [8], three algorithms for data integration were presented. The materialization, subquery-based, and the wrapper approach. Furthermore, query optimization issues were discussed and techniques presented to enhance the performance of these algorithms. In our work, we will extend the first two approaches to handle probabilities and paths (provenance) from different sources.

CHAPTER III

THE MATERIALIZATION APPROACH

In this section we are adding confidences to the semantic model presented in [8] using the idea of nodes probabilities in [2]. In the materialization Approach, binary tables using the mapping rules are precomputed in the mediator. In order to be able to compute confidence for different query results against the materialized tables, we added globally unique ID for each node in the original XML documents as shown in figure 3.2. When Binary relations are computed in the mediator, this unique ID will help identify the paths for different nodes up to the root. The difference between the binary relations in the basic materialization approach (without confidence) and our relations 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, is the extra paths columns specifying the path of each XML node in the original document. In addition, we need an extra paths table that contains each node's Id, it's probability, it's parent's Id, and its type, see table 3.10. The type field is basically for capturing different types of distributions: Mutual Exclusive and Independent. When a node's type is specified as DistME, that means it's children (either single nodes or whole tree branches)are mutually exclusive and this helps in computing conjunctive queries as we will see later. Moreover, when querying our model, we need to extend the basic algebra operations to manipulate paths that are required to compute the final confidence for query results. The Idea of keeping track of the paths (lineage) came from [12] that uses the lineage or the provenance for the confidence post computations. Through out this chapter we will use the Example from [2], see figure 3.1.

3.1 The Semantic Model View for Countries Example

One possible set of predicates (relations) of the semantic model view for the countries example presented in [2], see figure 3.1, are: country-countryName, country-latitude, country-longitude, country-independenceDay, country-chiefOfState, chiefOfState-title, chiefOfState-name, chiefOfState-age, chiefOfState-spouse. In these predicates, country and chiefOfState are URIs (IDs). The rest of attributes are values.

3.1.1 Mapping Rules for Probabilistic XML Database

The mapping rules should be modified to accommodate Dist and Val constructs. For each mapping rule such as

$$country - ChiefOfState(uri(\$C), \$S) \leftarrow countries/country\$C, \$C/government/chiefOfstate\$S$$

we need to add Dist and Val to the path. So the mapping rule will be :

$$country - ChiefOfState(uri(\$C), \$S) \leftarrow countries/country\$C, \$C/government/chiefOfstate/Dist/Val\$S$$

we can have both mapping rules, to handle both cases in the original document. The case where an instance do have Dist and Val as subelements, and the case where don't. In our model we will assume uniformity. In other words, if an element has Dist/Val subelements, all instances of that element must have Dist/Val subelements. This way, only one mapping rule is needed, and can be obtained automatically from the mapping rule that assumes no uncertainty.

Table 3.1: Country-CountryName R

Country	Path	CountryName	Path
2	1/2	United States	1/2/3
53	1/54	Uruguay	1/53/54

```
<countries>
  <country Prob='0.9'>
    <countryName>United States</countryName>
    <coordinates Prob='.9'>
      <latitude>
        <direction>North</direction>
        <degrees Prob='.8'>38</degrees>
        <minutes>00</minutes>
      </latitude>
      <longitude>
        <direction>West</direction>
        <degrees>97</degrees>
      </longitude>
    </coordinates>
    <government>
      <independenceDay Prob='.85'>07/04/1776
      </independenceDay>
      <chiefOfState>
        <Dist type="mutually-exclusive">
          <Val Prob='.5'>
            <title Prob='0.75'>President</title>
            <name>
              <Dist>
                <Val Prob='.4'>George W. Bush</Val>
                <Val Prob='.7'>George Bush</Val>
              </Dist>
            </name>
            <age>
              <Dist type="mutually-exclusive">
                <Val Prob='.2'>54</Val>
                <Val Prob='.35'>55</Val>
                <Val Prob='.1'>56</Val>
                <Val Prob='.15'>77</Val>
              </Dist>
            </age>
            <spouse>
              <Dist type="mutually-exclusive">
                <Val Prob='.5'>Laura Welch</Val>
                <Val Prob='.2'>Barbara Pierce</Val>
              </Dist>
            </spouse>
          </Val>
          <Val Prob='.2'>
            <title Prob='0.65'>President</title>
            <name>Bill Clinton</name>
            <age Prob='.3'>55</age>
          </Val>
        </Dist>
      </chiefOfState>
    </government>
  </country>

  <country>
    <countryName>Uruguay</countryName>
      . . .
  </country>
</countries>
```

Figure 3.1: Countries Example

Table 3.2: Country-ChiefOfState C

Country	Path	ChiefOfState	Path
2	1/2	20	1/2/15/18/19/20
2	1/2	43	1/2/15/18/43

Table 3.3: Country-Latitude L

Country	Path	Latitude	Path
2	1/2	5	1/2/4/5

3.2 Completeness

In this section we will discuss the completeness of our model depending on the mapping rules and the way they are reflecting the actual data in the original XML document. If we assumed that the mapping rules only encodes the existing data and skip missing data, then there is the possibility of loosing data when querying the binary encoded tables. Consider the following example, assume that we have Parts-Supplier XML document as in figure 3.3 , and consider having the following mapping rule : $p-s(\$p, \$s) \leftarrow parts/part\$G, \$G/pno\$P, \$G/supplier/sno\$S$ that only encodes parts that have suppliers, then the binary table will be as in table 3.11, where only parts that have suppliers are projected in it. In this case if we have the following query : $\Pi_{pno}(P-S)$, then the answer will contain only $pno = 1$, when the result should also contain $pno = 2$. In this case we have lost part of the actual information presented in the original XML document. Consequently, we have decided to make the mapping rules encode missing data in the original document to avoid loosing information. In this case the result for the previous query will be as in table 3.12. Although, NULL can cause some computational complexities, but having NULL in this case is necessary to avoid loosing information. When performing projection in this case, we won't be loosing any information. On the other hand, if we want to project pno that have suppliers from P-S, the project operation should check for the existence of such information and the answer should be computed as follows: $answer(\$X) \leftarrow parts/part[supplier/sno]/pno$, in which [supplier/sno] will check for the existence, and here it will project only pno=1 and not pno=2.

Table 3.4: Country-Longitude G

Country	Path	Longitude	Path
2	1/2	10	1/2/4/10

Table 3.5: Country-IndependenceDay D

Country	Path	IndependenceDay	Path
2	1/2	07-04-1776	1/2/4/5

3.3 The Algebra

In this section we will discuss our extensions to the basic algebra operations SJP (Select σ - Join $\bowtie$ - Project Π)to handle probabilities in our model and how can we manipulate paths' expressions for each operation to give us the correct final confidences for the result of different queries, including queries that involve conjunction and disjunction events. For that purpose, we will work with the same query examples in [2], to compare our results with their's.

3.3.1 Single node query

Consider the following query:

$$\sigma_{IndependenceDay=07-04-1776}(D)$$

the result of this query will be as in table 3.13.

Here we need an extra field to specify the resulting paths of such query, and we will call it Tuple Path. Where in this case the tuple path is the path of the tuple satisfying the query predicate. See table 3.14, where $p_1 = 1/2$ and $p_2 = 1/2/15/16$.

To compute the final confidence of the query answer we will multiply the probabilities associated to each node in the tuple path to get: $Pr(1) * Pr(2) * Pr(15) * Pr(16) = 1.0 * 0.9 * 1.0 * 0.85 = 0.765$, where $Pr(n)$ is the probability of node n. This confidence is the same as the confidence computed by ProTDB System for the same query in [2].

Table 3.6: ChiefOfState-Name N

ChiefOfState	Path	Name	Path
20	1/2/15/18/19/20	George W.Bush	1/2/15/18/19/20/22/23/24
20	1/2/15/18/19/20	George Bush	1/2/15/18/19/20/22/23/25
43	1/2/15/18/19/43	Bill Clinton	1/2/15/18/19/43/45

Table 3.7: ChiefOfState-Title T

ChiefOfState	Path	Title	Path
20	1/2/15/18/19/20	President	1/2/15/18/19/20/21
43	1/2/15/18/19/43	President	1/2/15/18/19/43/44

3.3.2 Conjunctive Queries

Consider the following query: "ChiefsOfStates with an age=55"

$$\Pi_{ChiefOfState}(\sigma_{Age=55}(A))$$

The result of this query with Tuple Paths is shown in table 3.15.

To project over ChiefOfState we perform the operation $\wedge$ between the path of ChiefOfState and the tuple Path to get the table 3.16.

Finally, to calculate the final confidence for each tuple we will get: for the first tuple path we have

$$p_1 \wedge p_2$$

where

$$p_1=1/2/15/18/19/20 \text{ and } p_2=1/2/15/18/19/20/28/29/31$$

so we need to factor out the common nodes before we compute the confidence to get:

$$Pr(1) * Pr(2) * Pr(15) * Pr(18) * Pr(19) * Pr(20) * Pr(28) * Pr(29) * Pr(31) =$$
$$1.0 * 0.9 * 1.0 * 1.0 * 1.0 * 0.5 * 1.0 * 1.0 * 0.35 = 0.1575$$

and for the second tuple path we have

$$p_3 \wedge p_4$$

Table 3.8: ChiefOfState-Age A

ChiefOfState	Path	Age	Path
20	1/2/15/18/19/20	54	1/2/15/18/19/20/28/29/30
20	1/2/15/18/19/20	55	1/2/15/18/19/20/28/29/31
20	1/2/15/18/19/20	56	1/2/15/18/19/20/28/29/32
20	1/2/15/18/19/20	77	1/2/15/18/19/20/28/29/33
43	1/2/15/18/19/43	55	1/2/15/18/19/43/46

Table 3.9: ChiefOfState-Spouse S

ChiefOfState	Path	Spouse	Path
20	1/2/15/18/19/20	Laura Welch	1/2/15/18/19/20/36/37/38
20	1/2/15/18/19/20	Barbara Pierce	1/2/15/18/19/20/36/37/39

where

p_3=1/2/15/18/19/43 and p_4=1/2/15/18/19/43/46

and as previously we need to factor out the common nodes before we compute the confidence to get:

$Pr(1)*Pr(2)*Pr(15)*Pr(18)*Pr(19)*Pr(43)*Pr(46) = 1.0*0.9*1.0*1.0*0.2*0.3 = 0.054$

See table 3.17

We can see that the results are the same as in [2], where each tuple conforms to a subtree with the same confidence.

Now Consider the following query that has the three SJP operations and a conjunctive predicate: "ChiefsOfStates with name=Goerge Bush and Age=55"

$$\Pi_{ChiefOfState}(\sigma_{Name=GoergeBush\ AND\ Age=55}(N \bowtie A))$$

The result of joining the two tables N and A will be as in table 3.18.

Where

$$p_1 = 1/2/15/18/19/20$$
$$p_2 = 1/2/15/18/19/20/23/24$$
$$p_3 = 1/2/15/18/19/20/28/29/30$$

Table 3.10: The Path Table

Node	Probability	Parent	Type
1	1.0		
2	0.9	1	
3	1.0	2	
.	.	.	.
.	.	.	.
.	.	.	.
.	.	.	.
18	1.0	15	
19	1	18	Dist ME
20	0.5	19	Val
.	.	.	.
.	.	.	.

Table 3.11: Pno-Supplier P-S

Pno	Path	Supplier	Path
1	p_1	100	p_2

$$p_4 = 1/2/15/18/19/20/28/29/31$$

$$p_5 = 1/2/15/18/19/20/28/29/32$$

$$p_6 = 1/2/15/18/19/20/28/29/33$$

$$p_7 = 1/2/15/18/19/20/23/25$$

$$p_8 = 1/2/15/18/19/43$$

$$p_9 = 1/2/15/18/19/43/45$$

$$p_{10} = 1/2/15/18/19/43/46$$

In the join Result we are performing the operation $\wedge$ between both paths of the join attribute from the two tables (ChiefOfState) (in this case it gave us the same path since both tables are in the same source and the join attribute has the same path [$p_1 \wedge p_1 = p_1$]). The result of the select operation with tuple path is as in table 3.19. As before we choose the tuple path to be the path of the attribute or attributes satisfying the predicate. In this case we had p_1 in the tuple path which was the result of applying the join. Here we perform $\wedge$ between the old tuple path and the new selected one.

Table 3.12: Pno-Supplier P-S

Pno	Path	Supplier	Path
1	p_1	100	p_2
2	p_3	NULL	NULL

Table 3.13: The result of $\sigma_{IndependenceDay=07-04-1776}(D)$

Country	Path	IndependenceDay	Path
2	1/2	07-04-1776	1/2/15/16

Then, to project over ChiefOfState we need to perform $\wedge$ operation between the tuple path and the ChiefOfState path as previously done, to get the table 3.20.

To compute the final confidence we will do as previously. We have the tuple path:

$$p_1 \wedge p_1 \wedge p_7 \wedge p_4$$

and after factoring out the common nodes we get: $0.9 * 0.5 * 0.35 * 0.7 = 0.11025$. The final confidence result of this query will be as in table 3.21.

3.3.3 Handling Disjunctions

We will see how we can handle disjunctions in our model by presenting the $\vee$ operation.

Consider the following query "ChiefOfState with name=Goerge Bush or Age=55"

$$\Pi_{ChiefOfState}(\sigma_{Name=GoergeBush\ OR\ Age=55}(N \bowtie A)$$

The result of the join is in table 3.18. Then the result of the selection with Tuple Path will be as in table 3.22.

Very important: Confidence computations (multiplications) should be performed last, since we need to factor out the common nodes from all path expressions before multiplying any nodes' probabilities.

As shown from the table, we have different tuple path associated to each tuple depending on the part of the tuple that satisfies the predicate. For example, the first tuple's tuple path is $p_1 \wedge p_4$ because p_4 is the path of (age=55). In the same way, the tuple path of the third tuple is $p_1 \wedge [(p_7) \vee (p_4)]$ because both parts of that tuple do satisfy the predicate of the selection.

Table 3.14: The result of $\sigma_{IndependenceDay="07-04-1776"}(D)$ with the Tuple Path

Tuple Path	Country	Path	IndependenceDay	Path
p_2	2	p_1	07-04-1776	p_2

Table 3.15: The result of $\sigma_{Age=55}(A)$ with Tuple Path

Tuple Path	ChiefOfState	Path	Age	Path
p_2	20	p_1	55	p_2
p_4	43	p_3	55	p_4

From this example, we can define the selection operation as follows: The tuples will be selected as normal and the tuple path will depend on the part of the tuple that satisfies the predicate.

Now, to project over ChiefOfState, we get the following: for each tuple we need to perform $\wedge$ between the path of the attribute we want to project over (in this case ChiefOfState) and the tuple path. We will obtain table 3.23 as a result of the project operation. All the tuples with ChiefOfState=20 will be merged to one tuple with one tuple path that is the result of the disjunction of all the tuple paths.

Finally, to calculate the final confidence for ChiefOfState = 20 we have:

$$
\begin{gathered}
p_1 \wedge p_4 \wedge p_1 \\
\vee \\
p_1 \wedge p_7 \wedge p_1 \\
\vee \\
p_1 \wedge (p_1 \wedge [(p_7) \vee (p_4)]) \\
\vee \\
p_1 \wedge p_7 \wedge p_1 \\
\vee \\
p_1 \wedge p_7 \wedge p_1
\end{gathered}
$$

after simplifications we will get:

$$p_7 \vee p_4$$

this will be calculated using the following formula:

$$Pr(p_7) + Pr(p_4) - Pr[(p_7 \wedge p_4)]$$

Table 3.16: The result of $\Pi_{ChiefOfState}(\sigma_{Age=55}(A))$ with Tuple Path

Tuple Path	ChiefOfState	Path
$p_1 \wedge p_2$	20	p_1
$p_3 \wedge p_4$	43	p_3

Table 3.17: The Final Confidence for The result of $\Pi_{ChiefOfState}(\sigma_{Age=55}(\mathrm{A}))$

ChiefOfState	Final Confidence
20	0.1575
43	0.054

$$= (0.9 * 0.5 * 0.35) + (0.9 * 0.5 * 0.7) - (0.9 * 0.5 * 0.7 * 0.35)$$
$$= 0.36225$$

and for ChiefOfState=43 we have

$$p_8 \wedge p_{10} \wedge p_8$$

after simplifications we get:

$$p_9 \wedge p_{10}$$
$$= Pr(1) * Pr(2) * Pr(15) * Pr(18) * Pr(19) * Pr(43) * Pr(46)$$
$$= 0.9 * 0.2 * 0.3$$
$$= 0.54$$

and as can be seen these results matches the results from [2].

See table 3.24.

3.4 Node Probability Update

In [2], they update the nodes probabilities in the resulted subtrees of a query. This can be obtained from our model as follows: we divide the tuple path for the node by the final confidence of the query result. For Example: to find out the probability of the node George Bush in the result of the disjunction query of section 3.3.3, we divide $Pr(p_7)$ by the final confidence of the query which is 0.36225. $Pr(p_7) = (0.9 * 0.5 * 0.7) = 0.315$.

The node probability of George Bush

$$= \frac{0.315}{0.36225} = 0.8695$$

Table 3.18: The Result of N⋈A with Tuple Path

Tuple Path	ChiefOfState	Path	Name	Path	Age	Path
p_1	20	p_1	G W Bush	p_2	54	p_3
p_1	20	p_1	G W Bush	p_2	55	p_4
p_1	20	p_1	G W Bush	p_2	56	p_5
p_1	20	p_1	G W Bush	p_2	77	p_6
p_1	20	p_1	G Bush	p_7	54	p_3
p_1	20	p_1	G Bush	p_7	55	p_4
p_1	20	p_1	G Bush	p_7	56	p_5
p_1	20	p_1	G Bush	p_7	77	p_6
p_8	43	p_8	Bill Clinton	p_9	55	p_{10}

Table 3.19: The Result of $\sigma_{Name=\text{"}GoergeBush\text{"}\ AND\ Age=55}(N \bowtie A)$

Tuple Path	ChiefOfState	Path	Name	Path	Age	Path
$p_1 \wedge p_7 \wedge p_4$	20	p_1	G Bush	p_7	55	p_4

which is the same as in [2]. From this observation, we can define the relationship between the ProTDB XML model and our model as follows: each tuple in the final result of a query in our model conforms to a subtree of the final result in the ProTDB model as shown previously. The tuple that contains 43 and 0.54 in table 3.24 conforms to the second subtree of the fourth tree pattern in [2] and so on.

Table 3.20: The Result of $\Pi_{ChiefOfState}(\sigma_{Name="GoergeBush"\ AND\ Age=55}(N \bowtie A))$

Tuple Path	ChiefOfState	Path
$p_1 \wedge p_1 \wedge p_7 \wedge p_4$	20	p_1

Table 3.21: The final result of $\Pi_{ChiefOfState}(\sigma_{Name=GoergeBush\ AND\ Age=55}(N \bowtie A))$

ChiefOfState	Final Confidence
20	0.11025

Table 3.22: The Result $\sigma_{Name=GoergeBush\ OR\ Age=55}(N \bowtie A)$ with Tuple Path

Tuple Path	ChiefOfState	Path	Name	Path	Age	Path
$p_1 \wedge p_4$	20	p_1	G W Bush	p_2	55	p_4
$p_1 \wedge p_7$	20	p_1	G Bush	p_7	54	p_3
$p_1 \wedge [(p_7) \vee (p_4)]$	20	p_1	G Bush	p_7	55	p_4
$p_1 \wedge p_7$	20	p_1	G Bush	p_7	56	p_5
$p_1 \wedge p_7$	20	p_1	G Bush	p_7	77	p_6
$p_8 \wedge p_{10}$	43	p_8	Bill Clinton	p_9	55	p_{10}

Table 3.23: The Result $\Pi_{ChiefOfState}(\sigma_{Name="GoergeBush"\ OR\ Age=55}(N \bowtie A))$ with Tuple Path

Tuple Path	ChiefOfState	Path
$p_1 \wedge p_4 \wedge p_1$	20	p_1
$p_1 \wedge p_7 \wedge p_1$	20	p_1
$p_1 \wedge (p_1 \wedge [(p_7) \vee (p_4)])$	20	p_1
$p_1 \wedge p_7 \wedge p_1$	20	p_1
$p_1 \wedge p_7 \wedge p_1$	20	p_1
$p_8 \wedge p_{10} \wedge p_8$	43	p_8

Table 3.24: The final confidence $\Pi(\sigma_{Name="GoergeBush"\ OR\ Age=55}(\mathrm{N} \bowtie \mathrm{A}))$ with Tuple Path

ChiefOfState	Final Confidence
20	0.36225
43	0.54

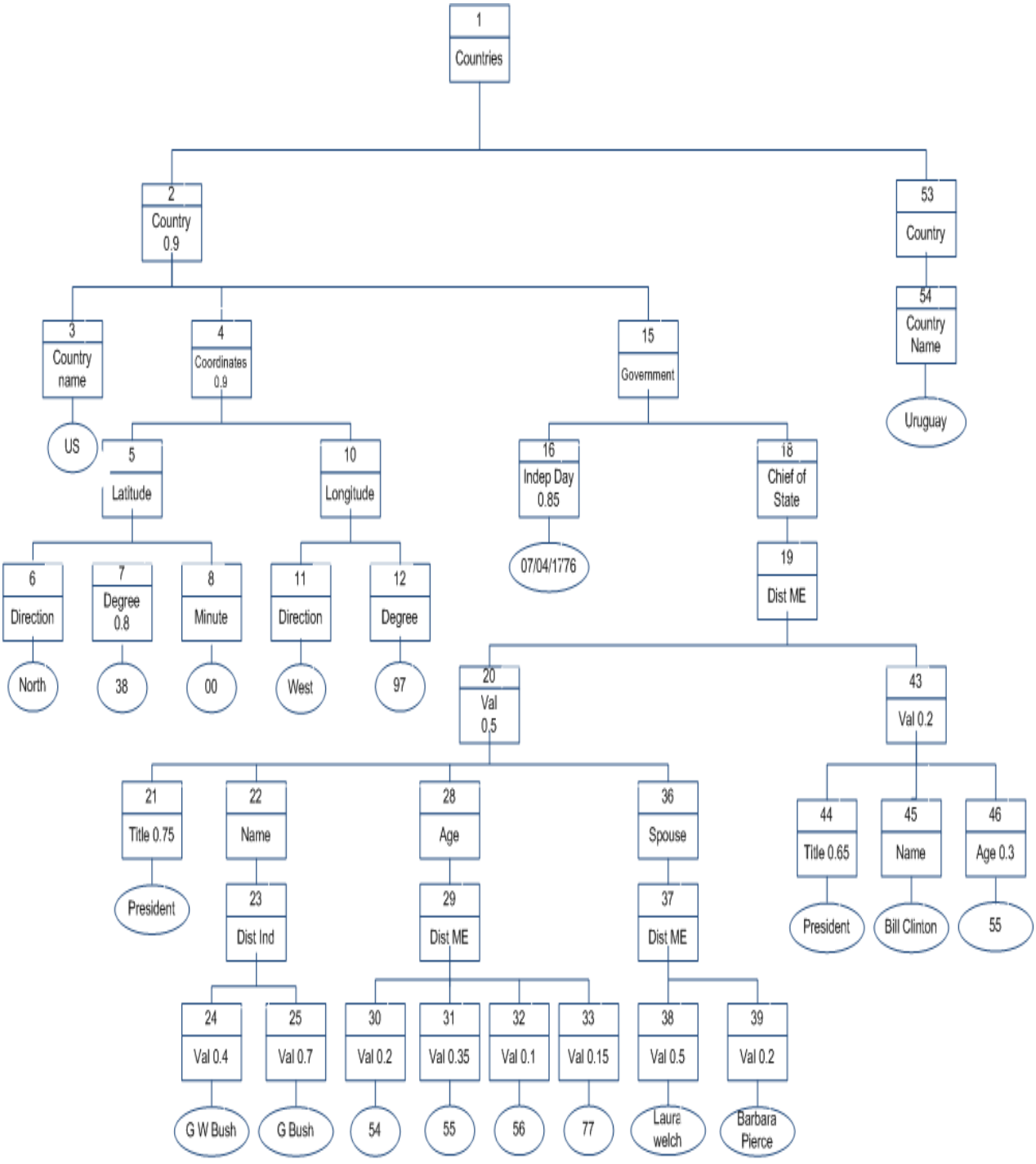

Figure 3.2: Countries Example Tree Representation

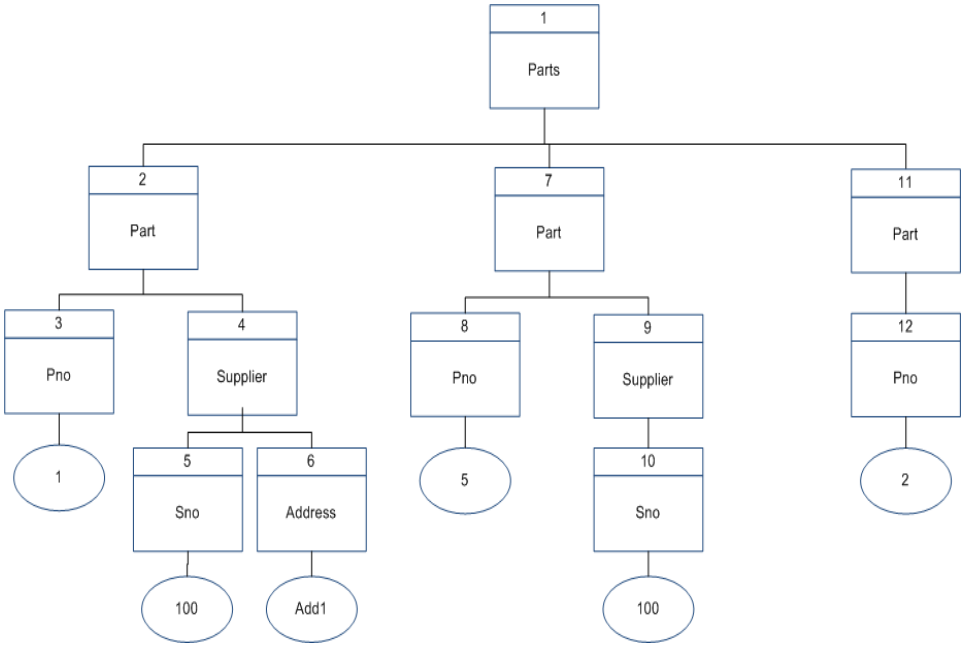

Figure 3.3: Parts-Supplier Tree Representation

CHAPTER IV
THE UNCERTAIN SUBQUERY APPROACH

In this chapter we will consider the subquery approach for processing user queries and getting the answer by generating and executing local subqueries from different sources. To accommodate probabilities, we will assume that ProTDB system is implemented in each site. In this approach, the user query is on the sematic model (relational). These relational queries will be then translated into queries against each uncertain XML source. After processing the query in each source,the results composed of one subtree or several subtrees are sent to the mediator along with their flat representation, which contains the resulted nodes with their paths. This path will help in computing the final confidence of the query depending on the query predicate. We will not consider intersource subqueries for now (queries that involve uncertain data from different sources). This assumption can be removed in the future if we could extend our work to merge uncertain XML data from different sources. In [4], they introduced a context-dependent logic-based approach to merge uncertain structured reports, where they use a Prolog knowledgebase to help in recognizing relationships between these reports. While in [11], a much simpler approach for merging uncertain XML data was introduced. Both approaches are not applicable to us because of the differences between their model and the ProTDB. In general, merging uncertain data still a complicated problem because of the nature of the resulted subtrees containing probabilities in multiple granularities and each node is probabilistically conditioned on it's parent. Through out this chapter, we will work with Parts-and-Suppliers example and try to investigate the uncertain subquery approach. The main idea behind the uncertain subquery approach, is to get the results from different sources and represent it in a flat form then find the final confidence of the user query by computing the disjunction of each tuple path of the results as we will show in the rest of this section. Also, we will show how to handle globally mutual exclusive distribution, when we have a user query that involves mutually exclusive conjunctive events from different sources.

4.1 The Parts and Suppliers example

We will assume an application that manages information about suppliers who supply parts. Each supplier have a unique ID (sno) and an address. Also, each part has a unique ID (pno) and a supplier or suppliers who supply this particular part. Then we will assume having two sources with two different schemes, See 4.1, 4.2 for full data in both sources.

4.2 Computing the final confidence of a user query

Consider the following single node query "Sno=200". The result using ProTDB system mechanisms in source 1 will be the two subtrees in figure 4.3. The flat representation of both subtrees will be as in table 4.1, where $p_1 = 1/2/4/5/7/11$ and $p_2 = 1/18/20/21/22/24$.

Table 4.1: Flat Representation of the result of "Sno=200" from source 1

Tuple Path	Sno	Path
p_1	200	p_1
p_2	200	p_2

The resulted subtree of this query from source 2 will be as in figure 4.4 and the flat representation as in table 4.2, where $p_3 = 32/44/45$.

Table 4.2: Flat Representation of the result of "Sno=200" from source 2

Tuple Path	Sno	Path
p_3	200	p_3

These subtrees along with their flat representations will be sent to the mediator. The probabilities associated to each node in each paths are also sent to the mediator to compute the final confidence. Now, to compute the final confidence of the previous user query in the mediator, we need to calculate the disjunction of p_1, p_2 and p_3. $p_1 \vee p_2 \vee p_3$ $= Pr(p_1) + Pr(p_2) + Pr(p_3) - Pr[(p_1 \wedge p_2)] - Pr[(p_2 \wedge p_3)] - Pr[(p_1 \wedge p_3)] + Pr[(p_1 \wedge p_2 \wedge p_3)]$ where $Pr(p_n)$ is the probability computed by multiplying nodes' probabilities in path n.

$= 0.25 + 0.63 + 0.9 - (0.25 * 0.63) - (0.63 * 0.9) - (0.25 * 0.9) + (0.25 * 0.63 * 0.9)$
$= 1.78 - 0.1575 - 0.567 - 0.225 + 0.14175 = 0.97225$ and as can be seen, the need for the flat representation is necessary to compute the correct probabilities because we need to keep track of the node's ancestors to factor out any common nodes before starting any multiplications. This previous method can be expensive because we are sending both the subtrees and their flat representations. One possible way to reduce the cost is to send only the flat representations of the resulted subtrees along with the node's probabilities without sending the actual subtrees.

4.3 Handling more complex queries

If we considered more complex queries in which the paths from each source is composed of more complex expressions, we need to convert the disjunction expressions into their canonical forms then compute the confidence as shown previously. For example if we have to compute final confidence of the following expression $(p_1 \wedge p_2) \vee p_3$ then we need to transform it to the following canonical form:

$$(p_1 \wedge p_2) \vee p_3$$
$$= [p_1 \wedge p_2 \wedge p_3] \vee [p_1 \wedge p_2 \wedge \neg p_3] \vee [p_1 \wedge \neg p_2 \wedge p_3] \vee [\neg p_1 \wedge p_2 \wedge p_3] \vee [\neg p_1 \wedge \neg p_2 \wedge p_3]$$

Confidences for each separate part of this expression can be easily calculated, where we assume just like in the ProTDB that $\neg A = 1 - A$.

4.4 Handling Mutual Exclusive

Assume that we have in the mediator a list of mutually exclusive events and one of these events is the address. So, each supplier can have only one address. Then, consider the query "The address of supplier with sno=200". The answer of this query from source 1 will be as in figure 4.5, and the flat representation of this answer as in table 4.3, where $p_1 = 1/2/4/5/7/12$ and $p_2 = 1/18/20/21/22/25$.

The answer from source 2 will be as in figure 4.6. And the flat representation as in table 4.4, where $p_3 = 32/44/46$.

Table 4.3: Flat Representation of the query "The address of supplier with sno=200" from source 1

Tuple Path	Address	Path
p_1	Add3	p_1
p_2	Add3	p_2

Table 4.4: Flat Representation of the result of "The address of supplier with sno=200"

Tuple Path	Address	Path
p_3	Add6	p_3

In this case, the probabilities for these events will be computed as previously and the subtrees will be sent to the mediator along with their flat representations. We also can send flat representations only without the subtrees. Then, when we want to calculate confidences for the previous example, the two confidences from the first source will be (0.5*0.5=0.25) and (0.7*0.9=0.63) and the probability of $p_1 \vee p_2 = 0.25 + 0.63 - (0.25 * 0.63) = 0.7225$. Moreover, the probability from the second source will be 0.9. These results are contradictory since for any mutually exclusive events e_1 and e_2, we have $Pr(e1) + Pr(e2) \leq 1$. In [6], they presented a solution to such situation which we will adopt. In this solution, probabilities Pr_1 and Pr_2 for two mutual exclusive events e1 and e2, will be modified to Pr_3 and Pr_4 (only if they are contradictory) with respect to two criterion.

- Preservation of ratio:

$$\frac{Pr_1}{Pr_2} = \frac{Pr_3}{Pr_4} \qquad (1)$$

- Preservation of all-fail probability:

$$p(\neg e_1 \wedge \neg e_2) = 1 - Pr_3 - Pr_4 = (1 - Pr_1) * (1 - Pr_2) \qquad (2)$$

If we considered the probability from source 1 is $Pr_1 = 0.7225$ and the probability from source 2 is $Pr_2 = 0.9$, then solving the two equations (1) and (2) for Pr_3 and Pr_4, we get $Pr_3 = 0.4329$ and $Pr_4 = 0.5393$. Having these modified probabilities, we can

conclude that the probability of having add3 as the address for the supplier with sno=200 is 0.4329 and the probability of having add6 instead of add3 is 0.5393.

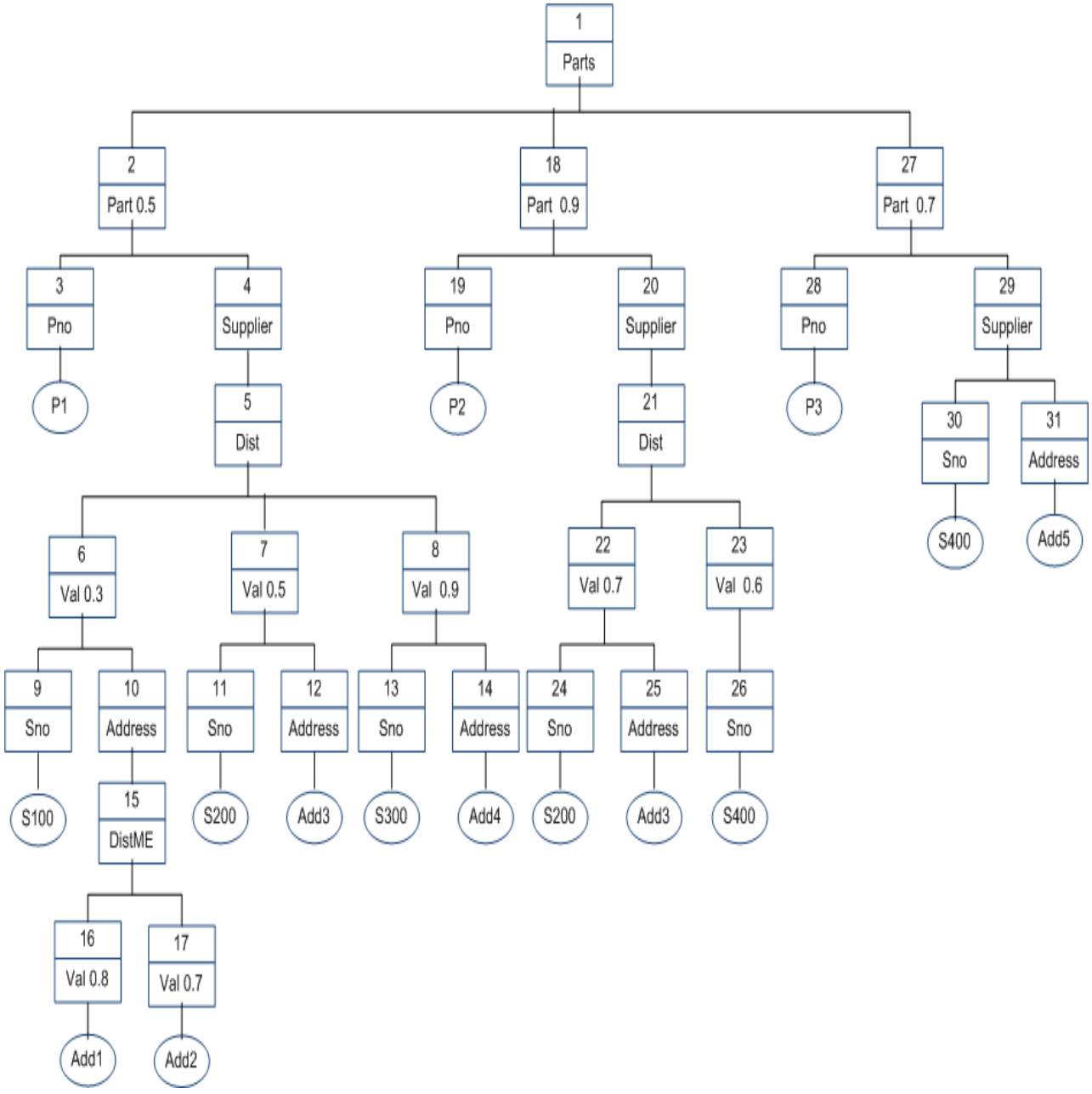

Figure 4.1: Source1 Parts and Suppliers data with probabilities

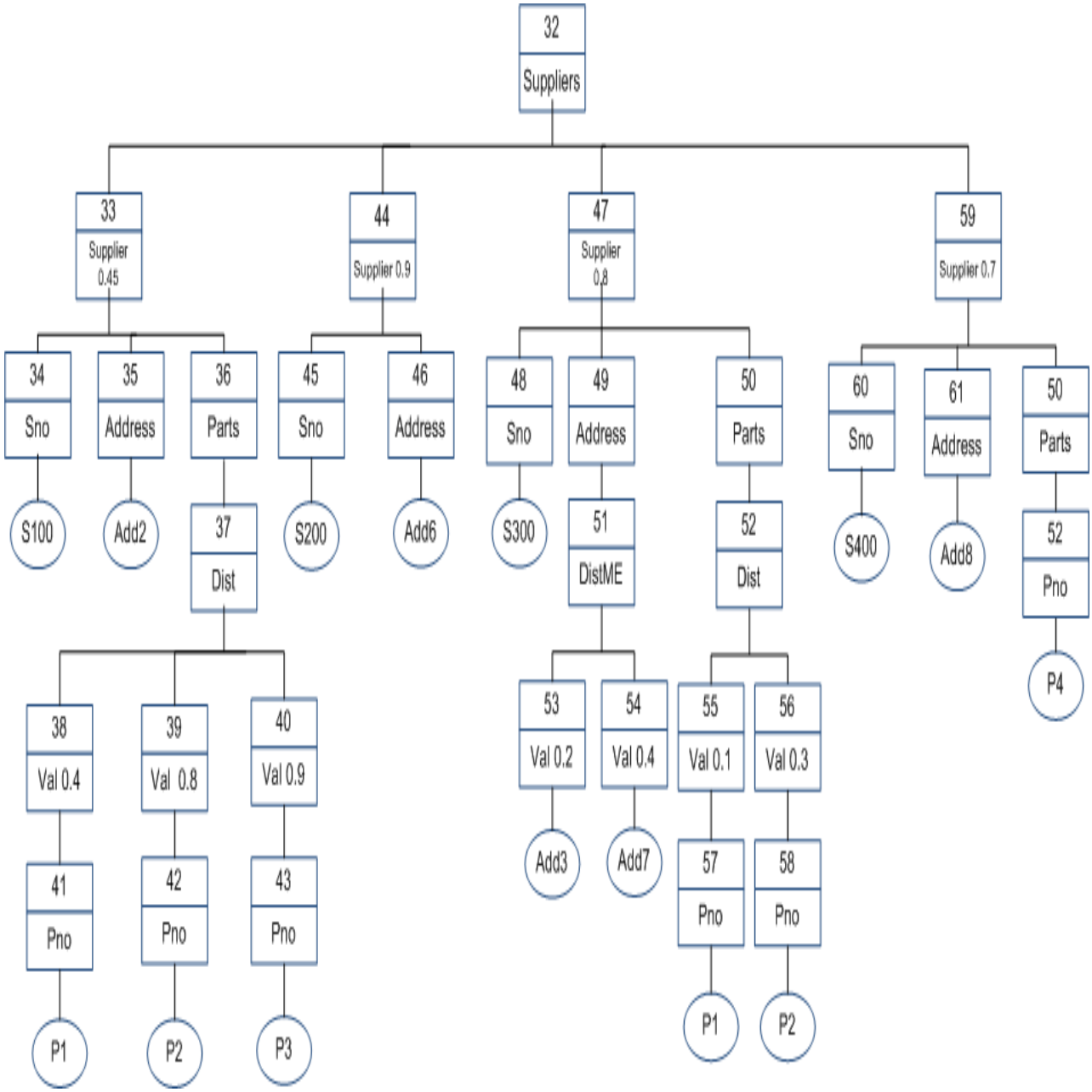

Figure 4.2: Source2 Parts and Suppliers data with probabilities

Figure 4.3: The resulted two subtrees from source 1 of the query sno=200

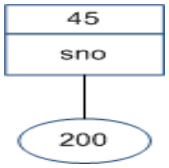

Figure 4.4: The resulted subtree from source 2 of the query sno=200

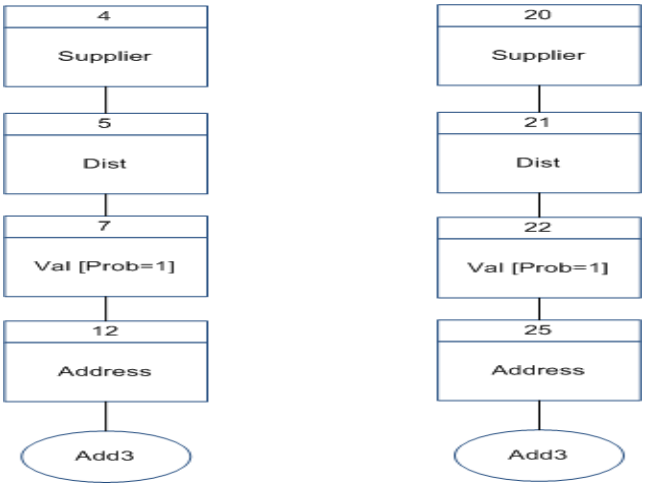

Figure 4.5: The resulted subtrees from source 1 of the query "The address of supplier with sno=200"

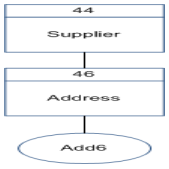

Figure 4.6: The resulted subtree from source 2 of the query "The address of supplier with sno=200"

CHAPTER V
CONCLUSIONS AND FUTURE WORK

We have presented in our work extensions to data integration algorithms in [8] to handle probabilities from different uncertain sources. These extensions were inspired from two different systems, the ProTDB and the Trio systems. In the Materialization approach, the first approach for data integration, we extended the binary relations in the semantic model with attributes to handle probabilities. In these attributes we represent different paths for different mapped information from the original document. These paths' major purpose is to help produce the correct confidence for the user query against the semantic model. We found that each tuple in the final result of a query in our model conforms to a subtree of the final result in the ProTDB model which helped us confirm our results when compared to results presented in [2]. We discussed briefly the completeness of our model depending on the mapping rules and what data they should encode. We also made extensions to the subquery approach, the second approach for data integration, to handle probabilities. We had to represent resulted subtrees from each uncertain source in a flat representation to help us compute the final confidence in the mediator by adding the probabilities disjunctions and to handle globally mutual exclusive. In our work we did not consider intersource subqueries in which case we need to join data from different sources to answer a single query. This is because of the nature of the probabilistic model we based our work in which each node's probability is dependant on the existence of its parent and the complications that this fact implies. This can be solved in the future, if we can extend our work to merge uncertain XML data. Another possible extension to our work is to handle probabilities for the wrapper approach, the third approach of data integration based on the semantic model.

BIBLIOGRAPHY

[1] Alex Dekhtyar, Judy Goldsmith and Sean R. Hawkes, Semistructured Probabilistic Databases, *Scientific and Statistical Database Management SSDBM Proceedings.* **13** (2001) 36–45.

[2] Andrew Neirman and H. V Jagadish, ProTDB: Probabilistic Data in XML, *Proceeding of the 28th VLDB Conference* (Hong Kong, China, 2002).

[3] Anish Dash Sarma, Omar Benjelloun, Alon Halevy and Jennifer Widom, Working Models for Uncertain Data, *22nd International Conference on Data Engineering ICDE* (2006).

[4] Anthony Hunter and Weiru Liu, Merging Uncertain Information with Semantic Heterogeneity in XML, *Knowledge and Information Systems* (2005) 230–258.

[5] Edward Hung, Lise Getoor and V.S. Subrahmanian, Probabilistic Interval XML, *ICDT* (2003) 361–377.

[6] Fereidoon Sadri, Integrity Constraints in the Information Source Tracking Method, *IEEE Transactions on Knoledge and Data Engineering* (1995).

[7] Fereidoon Sadri, Reliability of Answers to Queries in Relational Databases, *IEEE* (1991).

[8] Fereidoon Sadri, Dongfeng Chen, Rada Chirkova and Timo J. Salo, Query Optimization in XML-Based Information Integration, *CIKM* (2008).

[9] Feridoon Sadri and Laks V. S. Lakshmanan, XML Interoperability, *DBLP* (2003).

[10] Jennifer Widom et al., Trio A System for Integrated Management of Data, Uncertainty, and Lineage, 2008 (http://infolab.stanford.edu/trio/).

[11] Maurice van Keulen, Ander de Keijzer and Wouter Alink, A probabilistic XML Approach to Data Integration , *ICDE* (2005).

[12] Omar Benjelloun, Anish Das Sarma, Alon Halvey, Martin Theobald and Jennifer Widom, Databases with Uncertanty and Lineage, *The VLDB Journal* **17** (2008) 243–264.

[13] Peter Bueneman, Sanjeev Khanna, and Wang-Chiew Tan, A charecterization of Data Provenance, *International Conference on Database theory ICDT* (2001).

[14] Serge Abiteboul and Pierre Senellart, On the Complexity of Managing Probabilistic XML Data, *PODS* (2007).

[15] Serge Abiteboul and Pierre Senellart, Querying and Updating Probabilistic Information in XML, *DBLP* (2006) 1059–1068.

[16] Wang-Chiew Tan, Provenance in Databases: Past, Current, and Future, *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering* (2007).

[17] Wenzhong Zhao, Alex Dekhtyar and Judy Goldsmith, Representing Probabilistic Information in XML, (2003).