

Source Attribution for Querying Against Semi-structured Documents

T. Lee[†], S. Bressan[§], and S. Madnick[†]

[†]Sloan School of Management
Massachusetts Institute of Technology
{tlee,smadnick}@mit.edu

[§]School of Computing
National University of Singapore
steph@nus.edu.sg

Abstract

Mediation architectures like the Context Interchange research project, from which this work stems, integrate disparate information sources, hiding distribution and reconciling heterogeneity. As a result of transparent access, the notion of distinct sources often disappears from queries and results. However, there are situations where users or applications do need to know the sources from which a particular datum is drawn: for example, enforcement of intellectual property, evaluation of data quality or measurement of the timeliness of data. In this paper, we define *attribution* as the association of a value in the result of a query with the sources either from which the data was extracted or which contributed to the selection. Motivated by multi-source querying, attribution has particular applicability in the context of the World Wide Web. After beginning with an example that both describes and motivates the need for attribution within the semi-structured environment of the Web, we offer a general attribution framework and algebra and discuss the implementation of a prototype.

1 Introduction

Mediation networks integrate disparate information sources, hiding distribution and reconciling heterogeneity.[24] In addition to the Context Interchange research project, from which this work stems, several other projects [3, 10, 22] are developing mediation architectures.

The Context Interchange (COIN) Project [12], in particular, is developing a model [11, 7], a prototype [5, 6], and tools [8, 4], for the semantic integration of disparate (distributed and heterogeneous) information sources ranging from on-line databases to semi-structured Web services. From the perspective of a given data source, end-user, or intermediate application, context knowledge constitutes a declarative specification for how data is interpreted. By representing and reasoning about contexts, COIN's automated resolution of semantic conflicts enables transparent access to heterogeneous information sources.

As a result of transparent access, the notion of distinct sources often disappears from queries and results. However, there are situations where users or applications do need to know the sources from which a particular datum is drawn: for example, enforcement of intellectual property, evaluation of data quality or measurement of the timeliness of data.

Consider, in particular, the case of semi-structured data. When the data source is a semi-structured document, the mediation data model is often unable to convey all of the meaningful contents (e.g. images, audio, free text). In such a case, linking the results of a query to its respective sources facilitates the combination of database querying and document browsing in a single, multimodal paradigm.

In this paper, we define *attribution* as the association of a value in the result of a query with the sources either from which the data was extracted or which contributed to the selection. For example, the simplest case of this would be in a relational query where the query itself specifies the relation (the source) from which a particular result derives.

Motivated by multisource querying, attribution finds particular relevance in the World Wide Web. Within the semi-structured environment of the Web, assessing ownership, number, and quality of data sources is even more challenging than in traditional, federated systems. We revisit *Polygen*, a proposal for data-source tagging in a federated, relational database [23], in the context of Web wrapping [4, 8], a strategy for extracting information from semi-structured documents which stems from the COIN information integration research [12].

In the next section, we present an example query, which both describes and motivates the need for attribution. We revisit the example throughout the remainder of the paper to help illustrate central concepts. In Section 3, we offer a general framework for attribution and summarize key elements of the [23] attribution model. An algebra is included as an Appendix. Section 4 describes our architecture and implementation as an instantiation of the general model. We conclude by discussing related work and commenting on implications of our work for future research.

2 Example

One particularly promising application of the evolving, open information infrastructure is financial planning and financial analysis. Not only do companies themselves host public Web pages, but the exchanges and a number of financial

services companies post prices, earnings, industry-sector indices, portfolio management recommendations, and more. Consider a case where you are gathering financial recommendations for making stock purchases on the New York Stock Exchange (NYSE). Given the continuing Internet explosion, you might be particularly interested in viewing the current prices for recommended “strong buys” in the NYSE market segment of “data processing, software”. In pseudo-SQL, the prospective query might ask:

```
select ticker, company_name, recommen-
dation, and price
from 'the Web' where industry_sector =
"data processing, software"
and recommendation = "buy".
```

Unfortunately, the Web, as we know it today, does not lend itself to a rich, structured, query language. Moreover, the diversity of available information is paralleled only by the formats in which data is stored and the means by which that data is retrieved. Beneath the overarching World Wide Web lies a myriad of flat files, file systems, and commercial database products that mediation architectures [24] are designed to integrate.

Web wrapping, one component of a mediation architecture, constructs a relational interface to a variety of pre-selected, structured and semi-structured Web sites. For users seeking financial recommendations on NYSE securities, the mediation architecture might present the following *virtual relation* (call this relation *nyse*) with schema:

```
Ticker, Company_name, Industry_sector,
Share_Out, Recommendation, Last_price
```

Our original, prospective query may now be written as:

```
select Ticker, Company_name, Recommen-
dation, Last_price
from nyse where Industry_sector = "data
processing, software"
and Recommendation ≤ 2
```

This example query results in Table 1:

Ticker	Company_name	Recommen- dation	Last_price
SSW	Sterling Software, Inc.	1.7	32 7/16
CSC	Computer Sciences Corporation	1.8	77 7/8

Table 1: Database query result set¹

However, while the result table may answer the query, it simultaneously introduces the need for additional information. How current is the last price? Upon what basis did who make the recommendation? Is any other information of likely interest available? Whether to acknowledge the source of a particular piece of information, serve as a heuristic for assessing data quality, or uncover additional information, such questions point to the need for attribution.

An attribution system might, for each data cell in Table 1, report the document(s) from which the respective data value stems. Table 2 represents the query results attributed to a list of sources.

¹The table contains only excerpts from the actual query results.

Ticker	Recommendation
SSW url:nyse.com/symbol.csv url:zacks.com?t=SSW url:cnnfn.com/quote?sym=SSW	1.7 url:zacks.com?t=SSW
CSC url:nyse.com/symbol.csv url:zacks.com?t=CSC url:cnnfn.com/quote?sym=CSC	1.8 url:zacks.com?t=CSC

Table 2: Query result with attribution of direct sources²

Each cell associates a value with a list. The list contains a URL for each Web document that was accessed during the processing of the query and from which the corresponding cell value was extracted. For example, from Table 2, the Ticker symbol “CSC” could be read from:

- url:nyse.com/symbol.csv, the complete NYSE listing;
- url:zacks.com?t=CSC, a page reporting analysts recommendation for the CSC security;
- and url:cnnfn.com/quote?sym=CSC, a page reporting the latest performance figures of the CSC security in the NYSE.

whereas the Recommendation is found only at:

- url:zacks.com?t=CSC

In a hyper-text environment, a user could visit one of the listed references to conclude that *Zacks* should receive attribution, from an intellectual property perspective, for publishing the buy recommendation for CSC. If one wanted greater detail about that recommendation, reviewing *Zacks*’ report on CSC would reveal a break down of recommendations from the Wall Street Brokers who were polled by *Zacks* at the close of the market on the prior business day. Likewise, a visit to the associated CNN page would reveal additional detail such as the 52 week high, the 52 week low, and the number of shares of CSC currently being traded.

3 Attribution framework

This attribution framework derives from earlier work on the *Polygen* model for controlling data quality in federated relational databases [23]. [23] define a source as the name of the database from which a particular datum is extracted. In this section, we define *attribution* more generally as an association between a value and the source from which the value is extracted, leaving a more formal definition of a *source* for the appendix.

The attribution framework consists of a data model and an algebra which parallel the relational data model and algebra. The notion of a relation is extended to associate two types of sources, referred to here as *S* for direct sources and *I* for intermediate sources, with each data element in a tuple.

Intuitively, *S* is the set of sources from which a particular data is drawn. *S* will likely include more than one element as a data value in a tuple is often the result of combining several contributing tuples such as through joins, projections,

²The table contains only excerpts from the actual query results. The URLs in the table are heavily abbreviated for brevity.

R_1					
a	{s1}	{s1,s3}	b	{s3}	{s1}
a	{s1}	{s1,s3}	b	{s3}	{s3}
c	{s1}	{s1}	d	{s4}	{s2}

Table 3: Example of a weak duplicate

or unions. Consider, for example, our query of Section 2 and the direct sources from the result depicted in Table 2. For every cell in the tuple corresponding to $Ticker=CSC$, the set of indirect sources includes the source $Zacks$ which is used to verify the $Recommendation$ constraint in the query and $CNNfn$ which is used to verify the $Industry_sector$ constraint.

Similarly, I is the set of sources which provided data for or otherwise contributed to the selection of the result. A source might contribute to the selection of a result in the case of a join or a selection operation. In our example from Section 2, the tuple containing the ticker symbol CSC is included in the result set because the corresponding recommendation from the recommendation service falls below the specified query threshold and the NYSE listing verifies that the company’s industry sector satisfies the industry sector condition.

The attribution algebra comprises the basic set of operators for duplicate elimination, union, projection, selection, and coalesce. For equivalent query plans, produced under standard transformations in the relational algebra, the corresponding plans constructed for evaluating attribution should also be equivalent. Furthermore, the values in a standard relation should correspond to the cell values of the attributed relation. As elaborated upon in the Appendix, the attribution data model does not consider, as duplicates, tuples with equivalent values but different sets S or I . Hence, we introduce the notion of *weak duplicates* into the algebra. From Table 3, for relation R_1 , the tuples $\langle a, b \rangle$ are weak duplicates while $\langle c, d \rangle$ is not.

4 Implementation

Wrappers, as defined in [24], provide the physical connectivity and the first level of logical connectivity at the back end of a mediation architecture. At the logical level, wrappers abstract away the heterogeneity of the paradigms and models of information sources.³ In this section, we describe the integration of our attribution model into wrappers for accessing semi-structured Web sites developed as part of the Context Interchange mediation architecture [8, 4]. A prototype, including a query and navigation interface, is implemented using HTML, JavaScript, and Dynamic HTML. The system offers a practical environment for combining structured queries and casual browsing. After summarizing COIN wrappers, we detail the integration of the attribution model into the wrappers.

4.1 COIN Wrappers

COIN wrappers extract values from Web documents by means of patterns matching with regular expressions. Several other wrappers (eg. TSIMMIS, DISCO) employ similar strategies [4]. A specification language describes the patterns with

³A representative set of contributions about wrappers can be found in [21]

which data is extracted from individual Web pages into individual tables, and a strategy for combining intermediary results into a relational export scheme.

To the COIN Web wrappers, a *page*, is the association of a pattern for a Hyper Text Transfer Protocol (HTTP) message (the method, the Universal Resource Locator -URL-, and the object body) with patterns to be matched in the document source. A relation is defined over a set of one or more pages.

In cases where the message contains one or more variables, the associated page corresponds to a set of documents: one for each possible message generated by a substitution of the variables. Typically, this feature is used to describe documents that are dynamically generated via the common gateway interface (cgi) or collections of similar documents (for instance varying only in the file name). A message with a variable looks like:

```
GET http://quote.yahoo.com/q?s=##Ticker##
&d=##Format##
```

Instantiating the variables defines an evaluable message.

```
GET http://quote.yahoo.com/q?s=CSC&d=basic
```

Since a pattern can receive multiple instantiations within a page, it virtually defines a table of values, each column of which can be associated with an attribute of the relation. The co-occurrence of values in an instantiation of several variables in a page defines the rows of the table. Constructing a single relation over multiple pages is defined as the natural join of the distinct tables corresponding to each individual page. The relation is designed under the Universal Relation concept: attributes of the same name are the same. The window function is the natural join.⁴

Given a declarative definition of the exported relations, the wrapper takes an input query in SQL over the exported scheme, considers the patterns for extracting the corresponding values, and creates and executes a query execution plan. The query execution plan describes the various steps for accessing the documents, collecting data from the documents into individual tables, and combining the intermediary results into the query answer. The query plan is a tree of relational algebra operators (Theta-join, natural-join, select, project) and access operators (fetch and scan of a document). Under the semantics imposed by the choice of the window function (in our case the natural join), the wrapper attempts to minimize the number of documents accessed and the volume of intermediary information generated.

Illustrated in Figure 1, the general wrapper architecture consists of a set of front-end interfaces, a set of compilation and planning modules, and a set of execution modules. The SQL compiler and Specification file compiler provide the planner/optimizer with a goal and an initial search space from which it formulates a query plan. The Planner implements standard strategies for the optimization of queries in multi-databases. However, it accounts for capability restrictions. The leaves of the plan can be data from the query itself, or references to documents in the form of messages, which can either be instantiated or contain variables to be instantiated at run-time.

⁴Alternative window functions could be considered in conjunction with integrity knowledge; and they could offer greater potential for optimization and efficient evaluation. Opportunities for optimization and other wrapper details are reserved for a subsequent paper.

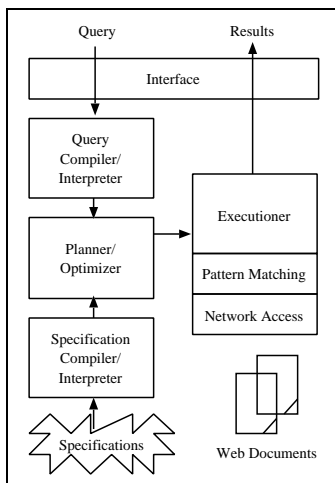


Figure 1: COIN-wrapper architecture

4.2 Wrappers with attribution

We now describe the integration of the attribution model from Section 3 into a prototype implementation of the Web-wrappers. Because COIN Web wrapping takes a document-centric approach, our attribution implementation defines a source as a Web document. More precisely, as specified in Section 4.1, a source is the message which was evaluated by the wrapper at query execution to retrieve the string from which the data values were extracted.

This distinction is important because a URL may not correspond to a single, physical document. Instead, a URL, method, and message body may invoke a script to dynamically generate a response such as our prototype does. More formally, for a tuple T with value $T_V[j]$ corresponding to attribute $j \in J$, the set of sources $T_S[j]$ is initialized to the fully instantiated HTTP message which returns the string from which the value $T_V[j]$ was extracted. The set of intermediates $T_I[j]$ is initialized to the empty set.

One implication of employing HTTP messages for attribution in the COIN wrappers is that the sets S and I must be constructed at run-time rather than derived solely as a function of the query plan. Recall from Section 4.1 that COIN wrappers may instantiate variables in a partial URL with values from some other page. The system must therefore wait until execution to construct complete messages for attribution.

At every node of the query plan, the wrapper's algebraic machine, which executes the plan and constructs the query results, evaluates both a standard, relational operation against values from the source and a parallel operation against the sets S and I associated with each value.

While the implementation of most operations on S and I is straightforward, a few notes are worth mentioning. The natural join is equated with a sequence of operators from the attribution algebra: a Cartesian product, a coalesce, and a project. Every constant declared in the θ -function of a select (σ) is attributed to the string "query" introduced to the algebraic machine as the Cartesian product of a single-valued relation⁵.

⁵From the algebra in the Appendix, the reader may note that an alternative implementation would incorporate the constant into the θ -function.

5 Related Work

Our implementation of the attribution model is integrated into a tool for combining structured queries on data extracted from Web pages with casual browsing of the same pages. More generally we may regard this attribution model as a step towards the integration of data base technologies with information retrieval techniques.

Some of the earliest strategies for integrating databases and information retrieval systems assumed that data was already structured within a document. These strategies focused on representing, querying, and retrieving from documents. Schek and Pistor introduced the NF² relational model to capture the nested structure of document attributes in a bibliographic database (e.g. sets of authors, sets of keywords, etc). [20] Fuhr drew from the keyword and boolean query languages of the information retrieval domain to propose probabilistic extensions to the relational model.[9] Uncertainty (eg. relevancy rankings) is injected into a structured data model and query language.

Uncertainty, however, stems not only from integration. Because of the potential for conflicts, multi-source querying inherently invokes uncertainty. [19, 18] uses a vector notation and an extended relational algebra to associate data sources with tuples in the relation. A probabilistic interpretation of the combination of source vectors produces an assessment of each associated tuple's reliability. Though the intuition behind attribution is similar, [19] raises the level of granularity from individual values to the tuple-level. The boolean simplification of source vectors is compact and aimed towards assessing data quality. It does not target the potential for enhanced browsing and searching highlighted by the proliferation of semi-structured sources like the Web.

Other authors have focused on the structure of the document itself to propose models that integrate databases and document retrieval systems. In [16], the authors distinguish between the layout structure which defines document presentation, the logical structure which defines document components (eg. title, author, etc.), and the conceptual structure which associates data with an abstraction of the real world. A general survey of document models and retrieval systems for text databases is presented in [14]. From the perspective of our attribution framework, these different models help us define, for a given application, the appropriate definition of a document and a source.

A number of recent proposals that focus on *semi-structured data* extend or specialize data models and query languages for NF² and complex object models [1]. Many of these models and languages [15, 17, 13] are powerful enough for representing and expressing our attribution model and algebra. However, in order to capture our attribution framework, most of these customized languages and models would require users to explicitly represent and manipulate the sources within their data structures and queries.

6 Conclusion

Building from prior work, we have presented a theory and an algebra as well as a number of conceptual issues relating to the nature of attribution. We also describe an application and implementation of attribution principles to Web querying.

Our original intuition defined attribution as an association between a value and a source. Instead of focusing on individual values, we can extend our theory by updating the granularity of attribution (e.g. what do we attribute to). We

might attribute a row, a column, or even a table. Likewise, we may extend our concept of source (e.g. what we attribute with). A source need not be a database, a document, or a reference to either. We might instead think about features of sources. By modifying our notion of sources (S) to encompass properties like time stamps, or information about authors, the attribution model can be adapted to a variety of applications.

The implementation detailed above makes a number of commitments regarding an instantiation of the framework from Section 3. Some choices are partially dictated by the choice of COIN-wrappers as an implementation environment (e.g. constructing the attribution sets S and I at run-time), while others (e.g. what constitutes a source as an element of S or I) are not.

For example, a different implementation might commit to a different conceptualization of a source. Our current use of an HTTP message as the attribution reference is limiting because the content associated with a message can change over time. Consider, for example, the case of a page that returns stock price quotations. The same message will return a different value for "Last_ price" every time the stock price is updated. In a hyper-text environment, moreover, a document may itself be defined in several ways. Consider the images, applets, scripts and other attachments that are visible through a browser. Are these part of the document source associated with a message? One could go so far as to recursively include all hyper-links, ultimately linking the entire Web!

Once committed to a particular conceptualization of a source, we might turn to the infrastructure for managing sources. First, rather than maintaining a parallel attribution data structure, we might factorize S and I for rows, columns, or even arbitrary slices of the results table. Consider how a well-placed wild-card or variable reference could substitute a single message in lieu of repeating the same string for every value in a column or row. Second, we might revisit how attribution is presented, whether through the query interface or a separate application. While hyperlinks match the context of the Web for browsing, how else might users further query or compose attribution sets, for example as a query execution trace.

In this paper, we have both a pragmatic model and a practical implementation that address the issue of source attribution. Applications of research on mediated systems [2, 22] point to the growing need for systems that integrate structured querying with data retrieval. We present attribution as a component for relating browsing to structured querying. In future work, we will attempt to preserve the general nature of the attribution framework while extending the model to encompass the complexity of temporal data models and richer attribution structures. At the same time, we are seeking opportunities to apply the model in other domains and application systems.

References

- [1] S. Abiteboul and N. Bidoit. Non first normal form relations to represent hierarchical organized data. *Proc. of the Intl. Conf. on Principles of Database Systems*, 1984.
- [2] J.M. Andreoli, U. Borghoff, and R. Pareschi. The constraint-based knowledge broker model: Semantics, implementation, and analysis. *J. of Symbolic Computation*, 1996.
- [3] Y. Arens and C. Knobloch. Planning and reformulating queries for semantically modelled multidatabase. In *Proc. of the Intl. Conf. on Information and Knowledge Management*, 1992.
- [4] Ph. Bonnet and S. Bressan. Extraction and integration of data from semi-structured documents into business applications. In *Proc. of the Intl. Conf. on Industrial Applications of Prolog*, 1997.
- [5] S. Bressan, K. Fynn, S. Madnick, T. Pena, and M. Siegel. Overview of the prolog implementation of the context interchange mediator. In *Proc. of the Intl. Conf. on Practical Applications of Prolog*, 1997.
- [6] S. Bressan, C. Goh, K. Fynn, M. Jakobisiak, K. Hussein, H. Kon, T. Lee, S. Madnick, T. Pena, J. Qu, A. Shum, and M. Siegel. The context interchange mediator prototype. In *Proc. of the ACM SIGMOD Intl. Conf. on the Management of Data*, 1997. (demo track).
- [7] S. Bressan, C. Goh, T. Lee, S. Madnick, and M. Siegel. A procedure for mediation of queries to sources in disparate contexts. In *Proc of the Intl. Logic Programming Symposium*, 1997. (To be published).
- [8] S. Bressan and T. Lee. Information brokering on the world wide web. In *Proc. of the Webnet World Conference*, 1997. (To be published).
- [9] Norbert Fuhr. Models for integrated information retrieval and database systems. *IEEE Data Engineering Bulletin*, 19(1):3-13, 1996.
- [10] H. Garcia-Molina. The TSIMMIS approach to mediation: Data models and languages. In *Proc. of the Conf. on Next Generation Information Technologies and Systems*, 1995.
- [11] C. Goh, S. Bressan, S. Madnick, and M. Siegel. Context interchange: New features and formalisms for the intelligent integration of information. Technical Report CISL WP97-03 to appear in ACM TOIS, MIT Sloan School of Management, 1997.
- [12] C. Goh, S. Madnick, and M. Siegel. Context interchange: Overcoming the challenges of large-scale interoperable database systems in a dynamic environment. In *Proc of the Intl. Conf. on Information and Knowledge Management*, 1994.
- [13] Laks V.S. Lakshmanan, Fereidoon Sadri, and Iyer N. Subramanian. A declarative language for querying and restructuring the web. In *Proceedings of the 6th International Workshop on Research Issues in Data Engineering (RIDE'96)*, New Orleans, February 1996.
- [14] Arjan Loeffen. Text databases: A survey of text models and systems. *SIGMOD Record*, 23(1):97-106, March 1994.
- [15] J. McHugh, S. Abiteboul, R. Goldman, D. Quass, and J. Widom. Lore: A database management system for semi-structured data. *SIGMOD Record*, 26(3), 1997.
- [16] Carlo Meghini, Fausto Rabitti, and Costantino Thanos. Conceptual modeling of multimedia documents. *IEEE Computer*, pages 23-30, October 1991.

- [17] Alberto O. Mendelzon and Tova Milo. Formal models of web queries. In *Proceedings of the Sixteenth ACM Symposium on Principles of Database Systems*, May 1997.
- [18] Fereidoon Sadri. Modeling uncertainty in databases. In *Proc of the 7th Intl Conf on Data Eng*, pages 122–131, 1991.
- [19] Fereidoon Sadri. Reliability of answers to queries in relational databases. *IEEE Transactions on Knowledge and Data Engineering*, 3(2):245–251, June 1991.
- [20] H. J. Schek and P. Pistor. Data structures for an integrated data base management and information retrieval system. In *Proceedings of the Eighth International Conference on Very Large Data Bases VLDB'82*, Mexico City, Mexico, September 1982.
- [21] D. Suciu, editor. *Proceedings of the Workshop on Management of Semi-structured Data*, Tucson, Arizona, 1997. <http://www.research.att.com/suciu/workshop-papers.html>.
- [22] A. Tomasic, L. Raschid, and P. Valduriez. Scaling heterogeneous database and the design of DISCO. In *Proc. of the 16th Intl. Conf. on Distributed Computing Systems*, 1996.
- [23] R. Wang and S. Madnick. A polygen model for heterogeneous database systems: The source-tagging perspective. In *Proc. of the Intl. Conf. on Very Large Databases*, 1990.
- [24] G. Wiederhold. Mediation in the architecture of future information systems. *Computer*, 23(3), 1992.

Appendix A. Attribution algebra

We call a *domain* a set of scalars. Let \mathcal{D} be a set of disjoint domains over which all relations are defined. In this subsection we define a *source* as a scalar. Let \mathcal{S} be the set of all sources. $2^{\mathcal{S}}$ is the power set of \mathcal{S} .

A relation is generally defined over a scheme which is itself defined by a set of associations between domains and attribute names. For notational convenience, we define a *scheme* as an index and a function where the function maps every element in the index to a domain. In other words, we identify attributes by their position.

Definition 1 (Scheme) A scheme is a pair (J, D) where J is an index (a set of integers from 1 to $\max(J)$) and D a function, $D : J \rightarrow \mathcal{D}$

In the standard relational model, a relation is defined over a scheme as a finite subset of the Cartesian product of the domains in the scheme.

Traditionally, each element of a relation is a tuple of scalars. In the attribution algebra, however, every scalar is associated with two sets of sources. A relation element is therefore defined as a tuple of triples. Each triple comprises a scalar and the set of direct and indirect sources. For a scheme (J, D) and $j \in J$ we call E_j the Cartesian product $D(j) \times 2^{\mathcal{S}} \times 2^{\mathcal{S}}$.

Definition 2 (Relation) a relation R , over a scheme (J, D) is a finite subset of the Cartesian product $\prod_{j \in J} E_j$ ⁶

⁶there should be a clear distinction between our use of \prod as the standard Cartesian Product set operation and our use of \times as the Cartesian Product in our attribution algebra which is defined below.

An element t of a relation R is called a *tuple*. The j^{th} element of tuple t is denoted by $t[j]$ and is called a *cell*. Each cell, $t[j]$, is composed of a scalar, which represents the cell's value, and sets of direct and indirect sources. We reference these elements as $t_V[j]$, $t_S[j]$, and $t_I[j]$ respectively.

The two relations illustrated in Table 3 are well formed. Two or more tuples with identical values but different source sets are said to be *weak duplicates*. The first relation, R_1 , does not contain any weak duplicates. R_2 contains several weak duplicates.

Definition 3 (Weak Duplicates) Given two tuples t_1 and t_2 in a relation R defined over the scheme (J, D) , we say that t_1 and t_2 are weak duplicates if and only if $\forall j \in J, t_{1V}[j] = t_{2V}[j]$

The first operator we define eliminates the weak duplicates in a relation. The respective sets of sources and intermediates for each cell of each weak duplicate are simply merged by taking their union. We call $Dup(t)$ the set of weak duplicates of the tuple t in a relation R .

Definition 4 ((δ) Weak Duplicate Elimination) Given a relation R_1 over the scheme (J, D) , the elimination of weak duplicates in R_1 is a relation $R_2 = \delta(R_1)$ over the scheme (J, D) such that: $R_2 = \delta(R_1) = \{t_2 | \exists t_1 \in R_1 \text{ and } t_{2V}[j] = t_{1V}[j], t_{2S}[j] = \bigcup_{t \in Dup(t_1)} t_S[j], t_{2I}[j] = \bigcup_{t \in Dup(t_1)} t_I[j]\}$

The next operator: the Cartesian product, does not affect the sets S and I associated with each cell.

Definition 5 ((\times) Cartesian Product) Given two relations R_1 , defined over the scheme (J_1, D_1) , and R_2 , defined over the scheme (J_2, D_2) , the Cartesian product of R_1 and R_2 is a relation $R_3 = R_1 \times R_2$ over the scheme J_3, D_3 such that, for $M_1 = \max(J_1)$ and $M_2 = \max(J_2)$:

J_3 is an index ranging from 1 to $M_1 + M_2$, and
 $\forall j \in J_3$ if $j \leq M_1$ then $D_3(j) = D_1(j)$, else $D_3(j) = D_2(j - M_1)$, and
 $R_3 = R_1 \times R_2 = \{t_3 | \exists t_1 \in R_1 \text{ and } \exists t_2 \in R_2, \forall j \in J_3, \text{ if } j \leq M_1 \text{ then } t_3[j] = t_1[j] \text{ else } t_3[j] = t_2[j - M_1]\}$

In the next two operators: the union and the project, incorporate weak duplicate elimination to preserve their similarity to their relational counterparts. For these operations, the sets S and I are affected only by merge operations on weak duplicates.

Definition 6 ((\vee) Union) Given two relations R_1 and R_2 defined over the scheme (J, D) , the union of R_1 and R_2 is a relation $R_3 = R_1 \vee R_2$ over the scheme (J, D) such that $R_3 = R_1 \vee R_2 = \delta(R_1 \cup R_2)$

Definition 7 ((π) Project) Given a relation R_1 over the scheme (J_1, D_1) , an index J_2 , and a function p from J_2 to J_1 , the projection of R_1 with respect to p is $R_2 = \pi(R_1)$ over the scheme (J_2, D_2) such that:

$\forall j \in J_2, D_2(j) = D_1(p(j))$, and
 $R_2 = \pi(R_1) = \delta(\{t_2 | \exists t_1 \in R_1 \text{ and } \forall j \in J_2, t_2[j] = t_{1[p(j)]}\})$

A selection involves the evaluation of a boolean selection condition, θ , over some subset of cells in the tuple. The set S , from every cell used in evaluation of the boolean selection condition, is added to the set I for every cell in the tuple. We say that the select operator defines a tuple level operation on the set I .

Definition 8 ((σ) Select) Given a relation R_1 over the scheme (J, D) , a set $K \subseteq J$, and a boolean function θ over the domain $\prod_{k \in K} D(k)$, the selection of the relation R_1 on the condition θ for the attributes $k \in K$ is the relation $R_2 = \sigma(R_1, \theta, L)$ over the domain (J, D) such that:

(we take $\theta(t)$ to mean the application of the boolean function θ on the values $t_v[k]$ of a tuple t for all $k \in K$)

$$R_2 = \sigma(R_1, \theta, K) = \{t_2 | \exists t_1 \in R_1 \text{ such that } \theta(t_1) \text{ and } \forall j \in J \ t_{2V}[j] = t_{1V}[j] \ t_{2S}[j] = t_{1S}[j] \ t_{2I}[j] = t_{1I}[j] \cup (\bigcup_{k \in K} t_S[k])\}$$

θ is defined generically as an n-ary boolean operator to account for compound or complex operations. The θ -join operator is now easily defined as a Cartesian Product and a Select. We omit from this paper the definitions of other operators such as intersection and difference because their definitions either follow in the same manner as the given definitions or may be expressed as compositions of those operators which have already been defined.

One final operator of note, however, is *coalesce*. Coalesce was introduced by Date (see [23]) to represent the merging of attributes with the same name when two relations with non-disjoint schemas are joined. As a part of this merge, coalesce would fill nulls from the overlapping attribute in one relation with values from the second relation. While our definition does not address null values, we do define a specific coalesce operator to differentiate between the behavior of S and I for selection on an equality condition versus the explicit merging of identical attributes. Where the equi-join is a Cartesian product and a select, the natural join is a Cartesian product, a Coalesce, and a Project.

To represent the possible conceptual difference between an equi-join and a natural join, we define *coalesce* such that

it only updates the I sets for those cells involved in the condition. For example, when data from the NYSE listing is joined with that from the Recommendations page on a ticker symbol, both NYSE and the Recommendations are valid sources for the attribute “Ticker_symbol”. The NYSE listing is not an intermediate source for the attribute “Recommendation”.

Definition 9 ((κ) Coalesce) Given a relation R_1 over the scheme (J, D) , a set $K \subseteq J$, the coalesce of the relation R_1 for the attributes in K is the relation $R_2 = \kappa(R_1, L)$ over the domain (J, D) such that:

(if we note $eq(t)$ the application of the boolean function verifying the equality of all its parameters on the values $t_v[k]$ of a tuple t for $k \in K$)

$$R_2 = \kappa(R_1, K) = \{t_2 | \exists t_1 \in R_1 \text{ such that } eq(t_1) \text{ and } \forall j \in J - K \ t_{2J}[j] = t_{1J}[j] \text{ and } \forall j \in K \ t_{2V}[j] = t_{1V}[j] \ t_{2S}[j] = t_{1S}[j] \ t_{2I}[j] = t_{1I}[j] \cup (\bigcup_{k \in K} t_S[k])\}$$

The model and algebra we have proposed is clearly based on a Non-First-Normal-Form (NF²) relational model [20, 1]. Except for the propagation of sources in I for the select operator, we can express the framework using an NF² algebra in a straightforward manner. In particular, we can express weak duplicate elimination using a sequence of applications of the nest and un-nest operators. Because the relations do not require more than one level of nesting, it is believed that we can express all plans with First Normal Form relational operators that are preceded and followed by an un-nest and a nest operation, respectively.⁷

⁷Use of First Normal Form relational operators assumes that the empty set is actually a singleton set containing a special symbol.