

Applicability of Universal Relation to Data Integration *

Wisconsin Data Integration Kit

Shreepadma Venugopalan
Computer Science Department
University of Wisconsin-Madison
Madison, WI, 53706
vshree@cs.wisc.edu

Krishna Pradeep R Tamma
Computer Science Department
University of Wisconsin-Madison
Madison, WI, 53706
pradeep@cs.wisc.edu

ABSTRACT

Data integration is the problem of combining data residing at different sources, and providing the user with a unified view of these data. Data Integration Systems studied in literature are characterized by an architecture based on the mediated schema and a set of base schemas. The relation between the mediated schema and base schemas is specified as a set of mappings. Designing such systems involves a schema generation phase, a query translation phase and a result integration phase. In schema generation phase, a mediated schema is defined as a set of relations possibly with integrity constraints, and each relation in the mediated schema is mapped to one or more relations in source schemas. Heterogeneity in nature and autonomy in design of real world systems hinder the creation of mediated schema and mappings. Further the updating of schemas and mappings becomes difficult in an environment that involves dynamic addition and deletion of sources. In the query translation phase, the queries that are posed on the mediated schema are translated on to sources based on the mappings. Queries posed are based on the integrity constraints that appear between the relations in the mediated schema. One of the main problems with such an approach is the distribution of attributes among the relations in the sources is not the same as in the mediated schema. In such a scenario the query writer can't correctly decide the joins between the relations. We propose a model where the user is presented with a universal relation as the mediated schema. The join decisions are no longer made by the query writer, but by the Query Translator. Attributes are added dynamically based on the sources. We design and build a system, Windik, based on our model. In literature, datalog, a subset of first order calculus is studied as the language for data integration systems. Integration involves transformation of values found in different databases. These transformations can be represented by function symbols in first order calculus. Windik

*Subject to change on finding an apt title

is designed to support first order calculus as the translation language.

Categories and Subject Descriptors

H.2.1 [Database Management]: Logical Design—*Data models*; H.2.5 [Database Management]: Heterogeneous Databases—*Data translation*

1. INTRODUCTION

Data integration is the problem of combining data residing at different sources, and providing the user with a unified view of these data [8]. Queries are specified over the provided unified view, and needs to be translated on to the source schemas¹. The results obtained after executing the queries should be integrated. An ideal data integration system once built should provide a language to create the unified view, and to integrate new sources when required. It should also provide interfaces for seamless querying by a manual user and programs.

Most of the Data Integration Systems today are characterized by an architecture based on the mediated schema and a set of base schemas. [8] presents a high level perspective of a data integration system and analyzes the different modeling approaches and mentions the problems involved in these approaches. The bases contain the real data, while the mediated schema provides a reconciled, integrated and a virtual view of the underlying bases. Modeling the relations between the mediated and the base schemas is therefore a crucial aspect. Two basic approaches have been proposed for this. The first one, Global as View (GAV)[8] defines the mediated schema in terms of the bases. The second approach, Local as View (LAV)[8] defines the mediated schema independently of the bases. The relationship between the mediated and the bases is established by defining the bases as views over the mediated schema.

Given such an architecture, queries posed on the mediated schema have to be reformulated into a set of queries in terms of the sources. A number of algorithms have been proposed for the reformulation, under both GAV and LAV mappings techniques. But real world data sources are autonomous, and hence the problem of incomplete and inconsistent data sources arise. To deal with the inconsistent and incomplete

¹base schema, source schema refer to the schema of source databases

sources, some Cleaning and Transformation techniques [5, 3] have been proposed. But we believe that this problem is more deep rooted and is fundamental to a Data Integration System, because of the limited knowledge of the query writer. Query Containment Check [4] is a fundamental problem in database theory and query containment check has to be done for the reformulated queries in a Data Integration System. The need for Query Containment Check arises in the LAV system because the sources are defined as views over the mediated schema. The query that is posed on the mediated schema is based on the semantics at the global level. So upon translation to the sources, the reformulated queries have to be checked for containment. We talk about these problems in greater depth later in the report.

Users in a Data Integration System typically specify the joins based on the knowledge of the mediated schema. But the distribution of attributes between the relations in the mediated schema is not same as in the base schema i.e., attributes in the same relation in the mediated schema may be present in different relations in the base schema. Similarly attributes in different relations in the mediated schema may be present in the same relation in the base. Hence the query translator, rather than the query writer, is in a better position to decide the joins between relations. So we present the user a schema that hides the integrity constraints. We give the details of such an architecture and some of the problems it solves in Section 1 of this report. This report is organized as follows. Section 2 presents some of the problems with the existing Data Integration Systems. Section 3 discusses Universal Relations and its applicability to our system. In Section 4 we describe the architecture of our system along with the solutions it provides. Finally Section 5 concludes the report by talking about some open problems and research issues that are yet to be addressed.

2. EXISTING DATA INTEGRATION SYSTEMS

Data Integration Systems, as discussed before, are characterized by the mediated schema - base schema architecture. One of the most important aspects in the design of these systems is the specification of correspondence between the mediated and the base schemas. Such a correspondence is modeled through the notion of mapping. As discussed in the previous section, the two basic approaches are GAV and LAV. Query Processing in the GAV approach is simpler, but changing the source warrants a change in the mediated schema. In LAV, changes to the source can be made independent of the mediated schema, but query processing is more complex and less accurate than in GAV. We introduced some of the problems with these systems in the previous section. We now discuss each of the problems in detail in the following sub sections. We also discuss two example data integration systems.

2.1 Inconsistency between Sources

Real world data sources are autonomous and hence may be inconsistent and incomplete. If the data obtained from these sources don't satisfy the integrity constraints of the mediated schema, query reformulation becomes meaningless. To deal with this problem, cleaning and transformation procedures [5, 3] have to be applied to the data obtained from

these sources.

In a Data Integration System, there may be a case that the data retrieved from the sources cannot be reconciled in the mediated schema in such a way that both the constraints of the mediated schema, and the mappings are satisfied. For example, in a LAV system, this happens when a key constraint specified for the relation r in the mediated schema is violated by the tuples retrieved by the views associated with r . A possible solution proposed [8] is to consider the integrity constraints of the mediated schema strong and the mapping between the schemas soft.

The problem of inconsistent sources in data integration [7] is addressed for particular cases:

- the global schema is relational schema with key-foreign key constraints
- mapping is of type GAV
- the query language is the language of union of conjunctive queries

In such a scenario, the technique specified above [8] checks whether a tuple is a certain answer to a query with respect to a given database in coNP complexity. Hence the problem of computing the certain answer has been shown to be co-NP complete.

But as discussed in the previous section, we believe that this problem is very fundamental to a data integration system. This problem arises, because the query writer has knowledge of the attribute distribution and integrity constraints only in the mediated schema. Hence the tuples retrieved from the sources may violate the integrity constraints of the mediated schema. We feel that presenting the user with a universal relation [10], instead of a mediated schema addresses the fundamental problem better. Since no integrity constraints are present in the universal relation, the query writer has to only specify the value of the "known attributes" and ask for the "unknown attributes". Our scheme is a lot like Query-By-Example. The tuple retrieved from the sources do not have to satisfy any integrity constraints at the mediated schema. We believe that our approach tackles the fundamental problem in a data integration system better and hence can provide solutions to a lot of other problems that arise because of this fundamental problem.

2.2 Query Containment Check

The basic form of reasoning on queries in Query Containment Check i.e., whether one query returns the subset of the result computed by the other query in all databases. A Query is said to be contained in another query relative to a set of sources modeled as views [9], if, for each extension of the views, the certain answer to the former query are a subset of the certain answer to the later. Most of the results on query containment concern conjunctive queries and their extensions. Query containment check is known to be NP-complete [4]. Query containment check has to be done for the existing LAV Systems upon query translation. In LAV systems, the sources are defined as views over the

global schema, and hence query is based on the semantics of the global schema. So upon query translation, containment check has to be performed to check whether the original query is contained in the translated queries.

In our system, the user does not pose queries on the universal relation. Instead, the user specifies only the known attributes and the unknown attributes. The universal relation has no referential integrity constraints. Based on the attributes given by the user and mappings maintained by the system, Windik formulates queries on the source schemas. Hence the issue of query containment check does not arise in our system. Windik formulates queries such that all tuples matching a given "filter"² are returned.

2.3 Relation Level Mappings

Mediated schema is defined as a set of relations in GAV and LAV architectures. In GAV, each relation in mediated schema is defined as union of views over relations in source schemas. Figures (1, 2, 3, 4, 5) represents the course listing schemas of different universities. If we consider Time(start_time, end_time) exists in mediated schema it can be represented as

$$\begin{aligned} \text{Time}(\text{start_time}, \text{end_time}) \equiv & \\ \text{Rice_Time}(\text{start_time}, \text{end_time}) \cap & \\ \text{Reed_Time}(\text{start_time}, \text{end_time}) \cap & \\ \text{UWash_Time}(\text{start_time}, \text{end_time}) \cap & \\ \text{UWisc_Hours}(\text{start}, \text{end}) \cap & \\ \text{WSU_Time}(\text{start}, \text{end}) \cap & \end{aligned}$$

A query on Time just has to unravel over the definition of time.

In LAV each relation in local schema is defined as a view over global schema. For example, if we again consider the Time relation in mediated schema, the Hours relation in UWisc can be specified as

$$\begin{aligned} \text{UWisc_Hours}(\text{start}, \text{end}) \equiv & \\ \text{Time}(\text{start_time}, \text{end_time}) & \end{aligned}$$

But the problem is when we try to create mapping over other relations like section, they don't come up as a proper union in case of GAV approach and a view over mediated schema in case of LAV approach.

2.4 Tsimmis

The Stanford-IBM Manager of Multiple Information Sources (Tsimmis)[2] is a Data Integration System developed at Stanford. Tsimmis uses Object-Exchange-Model(OEM) for communication between different components and Mediator Specification Language(MSL) as a query language. The systems follows different approach in a way that the system generates wrappers and mediator automatically for a particular query when they are specified in MSL. The systems seems to lack a general purpose approach.

²By filter, we mean the known attributes whose value is specified by the user

2.5 Information Manifold

Information Manifold[6] is a LAV Data Integration System. In this systems, a query Q is expressed in terms of Extensible Database Predicates(EDB). The problem in this system is to find a "solution" S for the query Q. A solution is an expression in terms of the views. In order to be a valid solution, the views in S are replaced by their definitions, and an expansion query E is obtained, which must be equivalent to the original query Q. For example, let us suppose there is a single EDB predicate p(X,Y) which we interpret to mean that Y is a parent of X. Let there be two views, defined as follows:

$$\begin{aligned} \text{v1}(\text{Y}, \text{Z}) & :- \text{p}(\text{X}, \text{Y}) \ \& \ \text{p}(\text{Y}, \text{Z}) \\ \text{v2}(\text{X}, \text{Z}) & :- \text{p}(\text{X}, \text{Y}) \ \& \ \text{p}(\text{Y}, \text{Z}) \end{aligned}$$

Suppose we want to query this information system for the great grandparents of a particular individual, whom we denote by constant 0. This query is expressed in terms of the EDB predicate p by

$$\text{q}(\text{C}) :- \text{p}(0, \text{A}) \ \& \ \text{p}(\text{A}, \text{B}) \ \& \ \text{p}(\text{B}, \text{C})$$

The problem is to find a conjunctive query whose subgoals use only the predicates v1 and v2 and whose equivalent to the query above. One of the solutions is,

$$\text{s1}(\text{C}) :- \text{v2}(0, \text{D}) \ \& \ \text{v1}(\text{D}, \text{C})$$

That is, if we replace each of the subgoals of s1 by the definition of the views, we get the expansion:

$$\text{e1}(\text{C}) :- \text{p}(0, \text{E}) \ \& \ \text{p}(\text{E}, \text{D}) \ \& \ \text{p}(\text{F}, \text{D}) \ \& \ \text{p}(\text{D}, \text{C})$$

We can use the conjunctive query containment test in both directions to prove that e1 \equiv q.

3. APPLICABILITY OF UNIVERSAL RELATION

A "universal relation" is an imaginary relation that represents all of the data in the database. A query language that lets us refer to the universal relation rather than to the actual database scheme, can be much simpler than typical relational query languages, because we need to mention only attributes than attribute relation pairs. The concept of universal relation, and various approaches to translating queries posed on a universal relation are described in [10]. In our approach, we expose the attributes of different relations of the mediated schema, as a universal relation. The attributes are dynamically created based on the base schemas. Addition of new source involves specifying the mapping file between the existing universal relation and the source schema. If a particular attribute fails to be mapped to the existing source relation, it is then added to the universal relation. The next source considers this new attribute while adding its schema.

3.1 Running Example

We explain our approach based on a running example. Figures(1, 2, 3, 4, 5) presents the course offering schemas of Rice University, Reed college, University of Washington, University of Wisconsin and Washington State University respectively. The schemas are taken from [1]. In each figure, bold letters specify the relations, while plain letters specify

the attributes. The plain lines specify attributes of a relation, while arrows represent key - foreign key relation. For example, in Figure ?? **Course**, **Section**, **Place** and **Time** are relations and remaining attributes. There exists a key - foreign key relation between Course-Section, Section-Place and Section-Time.

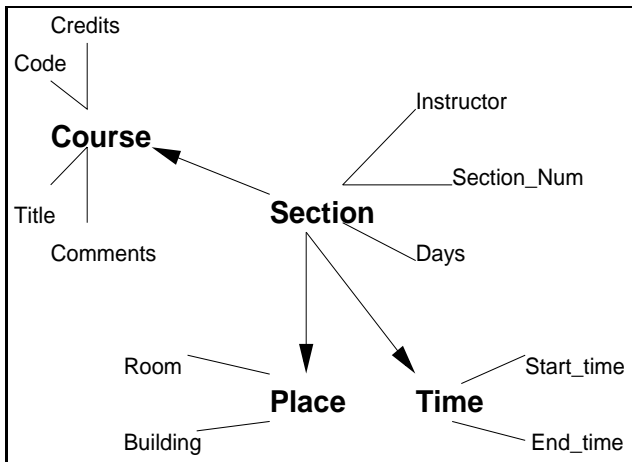


Figure 1: Rice University

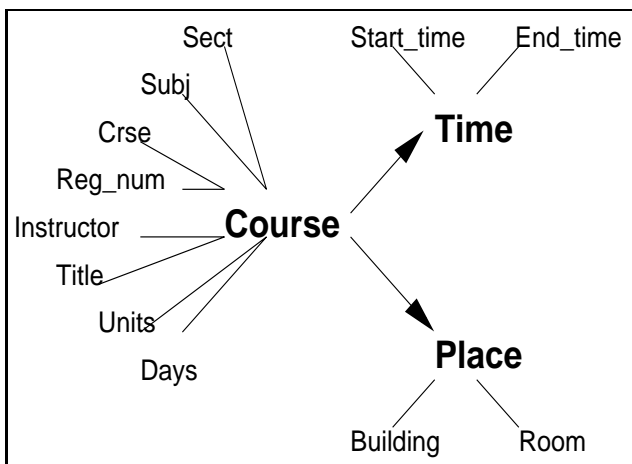


Figure 2: Reed College

3.2 Mappings

Our approach differs from existing approaches in creation of the mediated schema and the mapping scheme. While existing sources consider relation level mappings, we go for attribute level mappings. The decision to go for attribute level mappings is because no reasonable relation mapping is possible. This is due to distribution of attributes between relations, non-existence of some of the attributes and semantic differences of the attributes. [1] also gives data files for each of the schemas.

Attribute level mappings provide the best next-fit approach. Although there exists no one-to-one correspondence, most of the attributes in a relation are mapped to attributes in same or different relations. Table 1 shows the mappings between different schemas. The Universal relation is the union of attributes from different relations in the mediated schema.

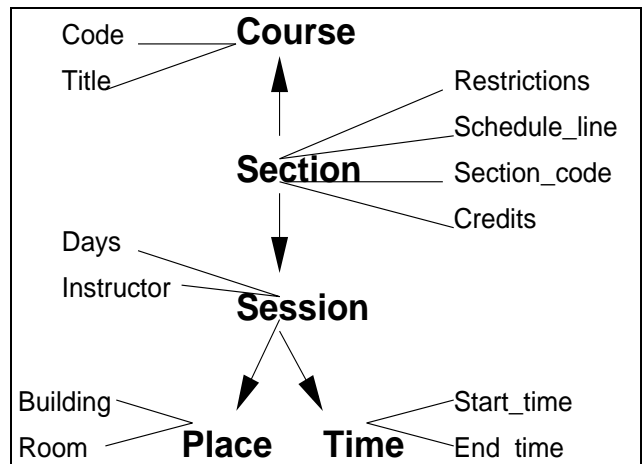


Figure 3: University of Washington

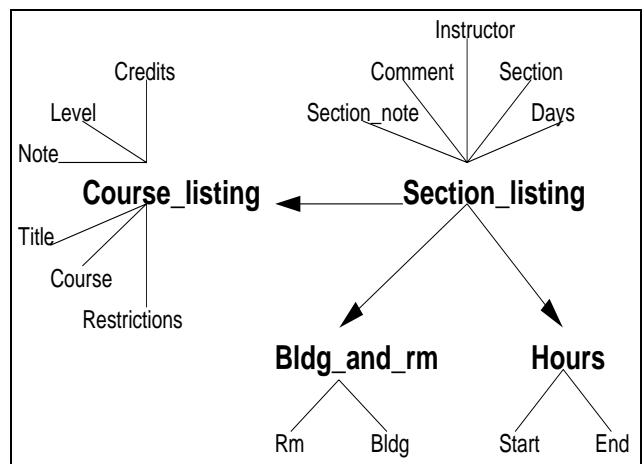


Figure 4: University of Wisconsin-Milwaukee

3.3 Foreign Keys

”Flattening” the relations in the XML schemas, creates foreign keys. For example in Figure 1, relation **Section** obtains keys of **Course**, **Place** and **Time** as foreign keys. The foreign keys are not exposed in the universal relation as they are dependent on the structure of schema, and typically do not have any semantics. Due to this, our approach to query processing differs from the existing universal relation approaches. Existing approaches expose all the attributes in each of relation, in the universal relation. Also they demand each attribute representing a particular ”entity” in the real world have the same name in all the relations. The existing schemes do not specify handling of relations with foreign key and primary key in the same relation.

For example, in the following Employee relation

```
Employee(SSN, ENAME, SuperSSN)
```

SuperSSN is the foreign key while SSN is the primary but both represents the same thing in real world. The universal relation approach does not specify how to handle the above case. In our approach, we do not specify foreign keys in the universal relation. An example for the self join is presented

Table 1: Mappings between different schemas

Universal	Reed	Rice	Uwash	WSU	UW-Milwaukee
title	course.title	course.title	course.title	course.title	course_listing.title
level					course_listing.level
course_credits	course.units	course.credits		course.credit	course_listing.credits
course_code_id	course.reg_num	course.code	course.code	course.sln	
subject	course.subj			course.prefix	course_listing.course
course_number					
schedule_line_number		section.section_num	section.schedule_line		
section_code	course.sect		section.section_code	course.sect	
section_credits			section.credits		
comment		course.comments	section.restrictions	course.footnote	section.comments
enrollment_limit				course.limit	
current_enrollment				course.enrolled	
date	course.days	section.days	session.days	course.days	section_listing.days
lecturer	course.instructor	section.instructor	session.instructor	course.instructor	section_listing.instructor
start_time	time.start_time	time.start_time	time.start_time	times.start	hours.start
end_time	time.end_time	time.end_time	time.end_time	times.end	hours.end
building	place.building	place.building	place.building	place.bldg	bldg_and_rm.bldg
room	place.room	place.room	place.room	place.room	bldg_and_rm.rm

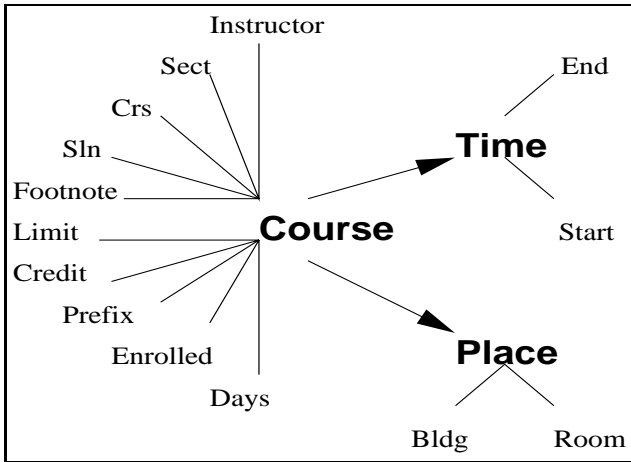


Figure 5: Washington State University

in the Architecture section 4

3.4 Query Translation

We present a graph based approach in mapping the query from the universal relation to the source schemas. An adjacency matrix is created based on the relations in the schema. If N is the number of relations in a schema, the matrix is of $N \times N$ dimensions. The cells of the matrix are filled with one if there is a foreignkey-key relationship between the relations that occur in the rows and columns of the matrix. Table 2 shows the adjacency matrix of the Rice schema.

Table 2: Adjacency Matrix of Rice Schema

Relations	Course	Section	Place	Time
Course	0	0	0	0
Section	1	0	1	1
Place	0	0	0	0
Time	0	0	0	0

In the query translation phase, attributes specified on the universal relation are matched with those of the source schema. All the matched relations with the matched attributes are isolated. Then for every two relation the graph is traversed for any existing path. If there exists a path the key and foreign key attributes of the two relations are selected. The following example specifies the translation approach.

Let the user requests the course title, instructor and time for each course. The query in the universal relation is specified as

```
Query(title, lecturer, start_time, end_time) :-
universal(title, lecturer, start_time, end_time)
```

Matching with the rice schema selects the following relations

```
Query(title, lecturer, start_time, end_time) :-
Course(title),
Section(instructor),
Time(start_time, end_time)
```

Between Section and course there exists a key - foreignkey relation such that course_id is selected in Course and Section relations. Also there exists a key-foreignkey relation between Section and Time relations. Hence time_id is selected between Section and Time relations. The final query looks as below,

```
Query(title, lecturer, start_time, end_time) :-
Course(course_id, title),
Section(instructor, course_id, time_id),
Time(time_id, start_time, end_time)
```

Finally the above datalog query is converted to SQL using a Datalog2SQL translator. The SQL query is as follows,

```
select title, instructor as lecturer, start_time end_time
from Course, Section, Time
where Course.course_id = Section_id and
Section.time_id = Time.time_id;
```

In the above example we did not consider comparison predicates. For example, the user can specify all 4-credit level courses by specifying 'credits=4' in the query over universal relation. The corresponding attribute in the source schema is matched and compared against the specified value. The final datalog query with a comparison predicate looks like

```
Query(title, lecturer, start_time, end_time) :-
Course(course_id, title),
Section(instructor, course_id, time_id),
Time(time_id, start_time, end_time), credits=4
```

The results obtained from each source is integrated and presented to the user or returned to the application program.

4. WINDIK ARCHITECTURE

Wisconsin data integration kit, Windik, is built to facilitate the integration of data sources. Windik makes use of attribute level mappings. Windik allows dynamic addition and deletion of sources. The attributes in the universal relation is updated on a "new" attribute in a source. An attribute is "new" if it cannot be mapped to any of the existing attributes in the universal relation. GAV scheme has the disadvantage that global schema has to be re-modeled when a source is added or deleted. In the LAV approach, the mediated schema is predefined and it is assumed not to change with the addition and deletion of sources. Windik gives the flexibility to modify the universal relation and does it without human intervention.

Our system provides a graphical user interface for an interactive session and class library for the application programs. The architecture of the kit is shown in Figure 6. The main components of the tool are User Interface, Source Wrapper, Result Integrator and Query translator. Windik currently supports SQL and Datalog as the language for application programs. It is designed to support the full power of first order calculus to allow transformations between the universal and local schemas. Concatenation of values of multiple

attributes, scaling values of an attribute, currency conversion are some examples of transformation operations. The different components are as follows:

4.1 User Interface

Windik exposes a graphical user interface(GUI) to the user for an interactive session and a class library for application programs. The graphical user interface exposes a Universal relation constructed from the union of attributes from all the sources. User can dynamically add, delete sources and universal relation gets updated accordingly. The user can select the attributes of interest and filter the attributes by specifying the values for some of the attributes. 7 shows the GUI of the Windik tool with few attributes selected and few attributes filtered over values. The class library presented to the user, can be used by an application programs. Applications programs can specify a query in datalog or SQL over universal relation. Only SELECT, FROM and WHERE clauses of SQL are currently supported.

An example query in datalog can be specified as

```
Query(title, lecturer):- Universal(title, lecturer) and
course_credits=3 and
enrollment_limit=10;
```

The same query in SQL is specified as

```
SELECT title, lecturer
FROM universal
WHERE course_credits=3 and
enrollment_limit=10;
```

Currently the GUI does not expose any interface for self joins, but the class library can be used for the same by specifying the universal relation twice in the datalog. Consider the following query: Consider the Employee relation shown below,

```
Employee(ENO, ENAME, MENO)
```

The corresponding universal relation is

```
universal(ENAME)
```

The following datalog query gives the employees and their managers.

```
Query(ENAME, MNAME):- universal(ENAME) and
universal(ENAME);
```

4.2 Result Integrator

The integrator performs two functions, it invokes each source wrapper with the user specified attributes and filters. Once the source returns the results, it performs a union over the results and presents it to the user. The complexity of inte-

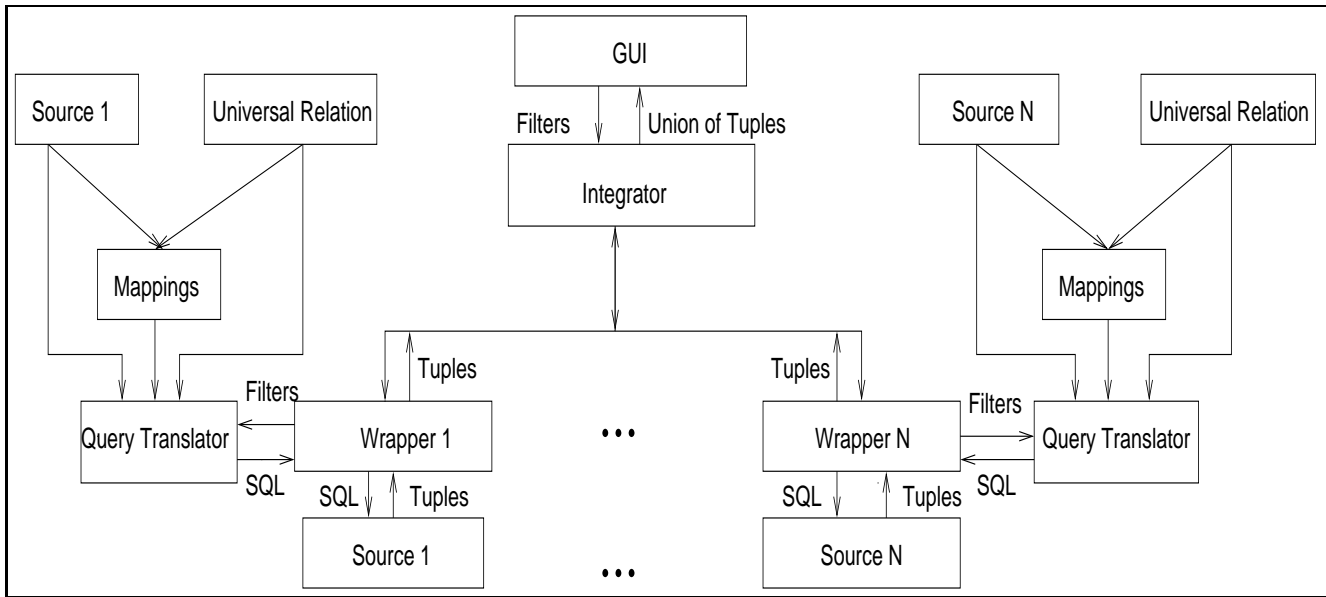


Figure 6: Windik Architecture

gration is reduced by having the wrapper return the results in a common format.

4.3 Source Wrapper

The wrapper program is the interface between integrator and the source database. The integrator invokes the wrapper program with the selected attributes and filters specified over these attributes. The wrapper invokes the Query translator to formulate the SQL query over the source schema and executes the query over the source database. It processes the results before passing it to the Integrator. Translator fills in NULL values if a particular attribute is not obtained in the result.

4.4 Query Translator

Query translator has knowledge of the Source schema, the Universal relation and the mappings. The translator obtains the query over universal relation as the input. It follows the approach described in 3.4 and passes an SQL query to the Wrapper program.

5. CONCLUSION AND FUTURE WORK

We present the application of Universal relation to the data integration framework. Application of universal relation solves the problem of integrity constraint mismatch between different schemas. We present a graph based algorithm to translate queries from the Universal relation to the source schemas. The graph based approach is polynomial in the number of relations. Further it solves the self join problem. We present the architecture of Windik, a system designed and built on the above approach.

5.1 Multiple relationships

Consider the following relation scheme

```
Employee(SSN, ENAME, DNO)
Department(DNO, DNAME, MENO)
```

The corresponding universal relation scheme will be

```
universal(SSN, ENAME, DNAME)
```

The query for Employees and their corresponding departments will be

```
Query(ENAME, DNAME) :-
universal(ENAME, DNAME)
```

and the query for Departments and their managers will be

```
Query(DNAME, ENAME) :-
universal(DNAME, ENAME)
```

Both queries are syntactically the same. So the Universal relation has no way of distinguishing between queries, if they involve different relationships between the same relation. The Universal relation concept should be extended to incorporate this.

5.2 Function Symbols

Data integration involves transformations between values of attributes. For example, a schema may store courses as Graduate and Undergraduate, while another schema may actually give varying numbers. In one of the schemas, attribute name may be stored as FirstName, Middle Initial and LastName, while in another it may be stored as a single name attribute. Based on application, one can define the Universal relation. The values in source schemas have to be transformed to the format of Universal relation. This require the translation language to include function symbols. Existing systems are use Datalog as the notation. Adding function systems to datalog gives first order calculus. Hence data integration should be extended to involved FOL.

An example query on universal relation can be

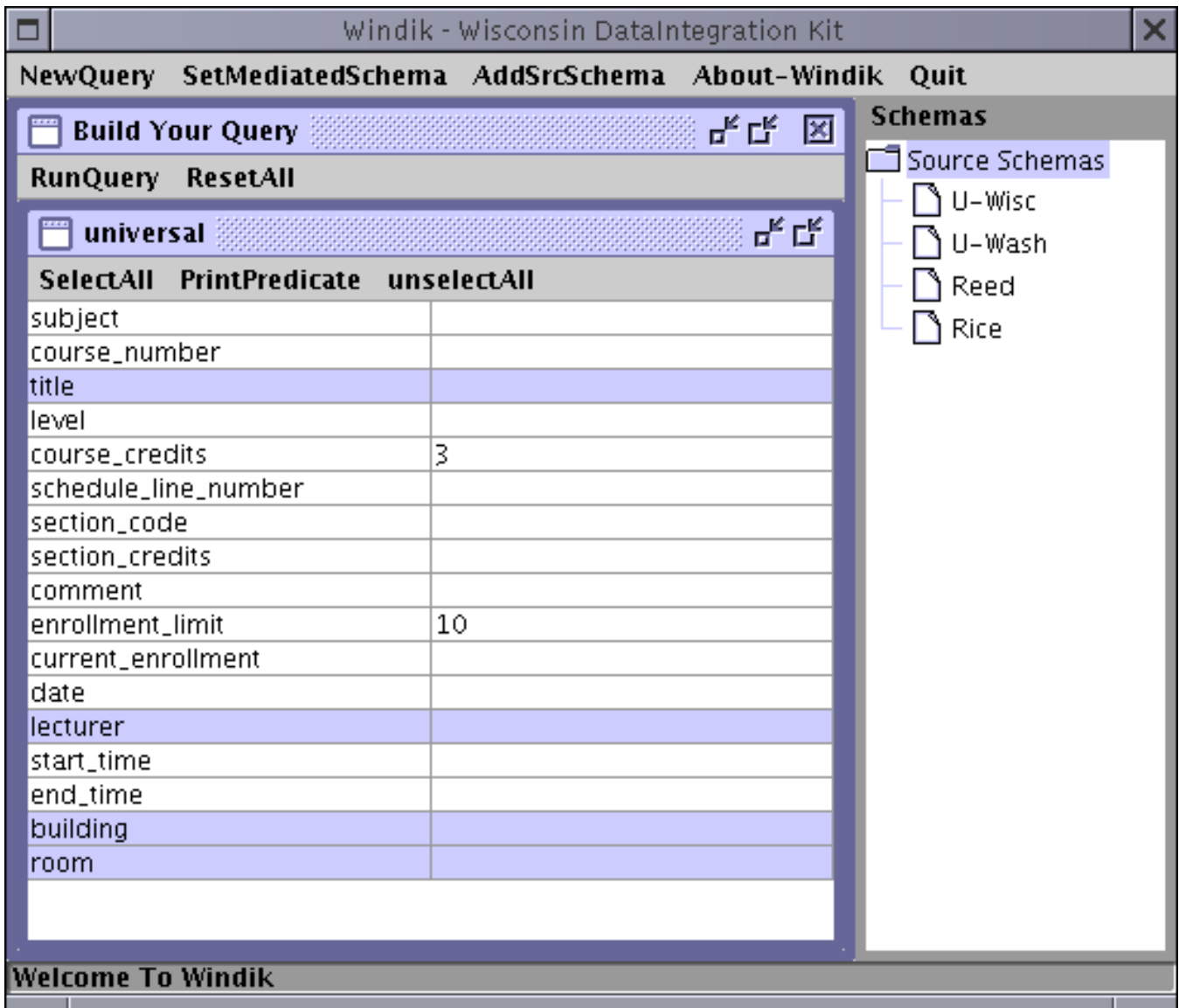


Figure 7: Windik Graphical User Interface

Query(name) :- universal(name)

Then the translation language should provide opportunity to specify query like

Query(name) :-
Person(concatenate(FNAME, MNAME, LNAME))

Limitations of the datalog and applicability of FOL should be thoroughly studied.

6. ACKNOWLEDGEMENTS

We thank Jeffrey F Naughton for the insightful discussions and valuable suggestions. We thank Anhai Doan for the schemas and the data set.

7. REFERENCES

- [1] Illinois semantic integration archive. <http://anhai.cs.uiuc.edu/archive/>.
- [2] The stanford-ibm manager of multiple information sources. <http://www-db.stanford.edu/tsimmis>.
- [3] M. Bouzeghoub and M. Lenzerini. Introduction to the special issue on data extraction, cleaning, and reconciliation. *Information Systems*, 26(8):535-536, 2001.
- [4] A. K. Chandra and P. M. Merlin. Optimal implementation of conjunctive queries in relational databases. In *Proceedings of the ninth annual ACM symposium on Theory of computing(STOC'77)*, pages 77-90, 1977.
- [5] H. Galhardas, D. Florescu, D. Shasha, and E. Simon.

An extensible framework for data cleaning. *Technical Report*, 3742, 1999.

- [6] T. Kirk, A. Y. Levy, Y. Sagiv, and D. Srivastava. The Information Manifold. In *Information Gathering from Heterogeneous, Distributed Environments*, 1995.
- [7] D. Lembo, M. Lenzerini, and R. Rosati. Source inconsistency and incompleteness in data integration. 2002.
- [8] M. Lenzerini. Data integration: a theoretical perspective. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 233–246, 2002.
- [9] T. Millstein, A. Y. Levy, and M. Friedman. Query containment for data integration systems. pages 67–75, 2000.
- [10] J. D. Ullman. *Principles of Database and Knowledge-based systems Volume II: The new technologies*. 1988.