CPSC504 project proposal: A Datalog to SQL converter

Jian Xu (xujian@cs.ubc.ca)

August 19, 2010

Abstract

This project proposal for CPSC 504 describes the task of building a Datalog to SQL parser. Datalog is commonly used in semantic integration to express queries and mappings and SQL is the language for RDBMS systems such as MySQL. Building of a Datalog to SQL parser will enable the execution of queries generated/translated by semantic integration algorithms on existing RDBMSs (e.g., a running MySQL database) and has good impact on data integration research and evaluation.

1 Introduction

This project aims to build a language interpreter to translate a Datalog query to an SQL query. Doing this enables Datalog queries be processed in existing databases, e.g., a MySQL database. The side-products of this project are (1) a language interpreter that convert Datalog sentences from/to grammar trees and (2) data structures that manage schema information.

We only need a 1-way interpreter that takes a Datalog sentence as input and output an equivalent SQL query. Doing so in a reverse direction is not necessary for this project.

This project only requires a "simple" interpreter that supports SPJ and aggregation queries in Datalog. Complicated query types such as recursive queries are not necessary. However, the interpreter should be able to detect if an input is in its supported class.

The following are two examples for the anticipated interpreter.

Example 1 *Datalog query* $Q(x, y) : -R_1(x, 3, y), R_2(x, t), t > 5$ *with schema* $R_1(a_1, a_2, a_3)$, $R_2(b_1, b_2)$ *where* a_i, b_i *are attribute names and* x, y, t *are variables.*

Query Q *is translated to SQL:*

```
SELECT a_1, a_3 FROM R_1, R_2
WHERE a_1 = b_1 AND
a_2 = 3 AND
b_2 > 5
```

And here is an example with aggregation with the same schema as above.

Example 2 *Datalog query* $Q(sum(x), y) : -R_1(x, 3, y), R_2(x, t), t > 5, y > 2$ *is translated to SQL*

```
SELECT sum(a_1), a_3 FROM R_1, R_2
WHERE a_1 = b_1 AND
a_2 = 3 AND
b_2 >5
GROUP BY a_3
HAVING a_3 > 2
```

The above example showed conceptually how aggregations, group by predicates and having predicates are expressed in Datalog.

2 Prerequisite

We need a interpreter to both work as a stand-alone program and as a class library. Library implementation should be done with C++. Knowledge with C++ STL is required.

We use GNU Bison-flex tool chain for lexical and grammar interpreter. Project member should learn/get familiar with these tools.

The project requires some coding to query a MySQL database. Interfacing with MySQL using C++ is recommended. As the converter outputs SQL queries, it's also possible to use other language (e.g., Java) for a frontend.

The project requires basic knowledge on relational databases and how schema is managed in MySQL databases.

3 Task breakdown

Tasks in this project can be broken down as following :

1. Check the syntax of a Datalog query, verify the correctness with database schema.
2. Translate a Datalog query into a Datalog grammar tree.
3. Translate a Datalog grammar tree into SQL query.
4. Send the SQL query to a MySQL server and retrieve answers.

The project is estimated for a group of 3 students to finish as a one semester course project. A final report is required for the project to cover the design, implementation and testing of the converter.

4 Code to start with

See files in LangSpec for the definition of a datalog grammar tree. File dlogParserSpec.pdf described the data structure for the grammar tree. datalog.l and datalog.ypp are the Flex lexical and bison grammar spec. for the required datalog support (The Bison and Flex files still require debugging).

5 Expected help from TA

Your TA could help in the design process of the converter, help with using the MySQL database and provide some test cases.