Summary of ’Data Fusion in Three Steps: Resolving Schema, Tuple and Value Inconsistencies’

Grishma Jena

Last Updated: May 23, 2014
Contents

1 Introduction 1

2 Motivation 1

3 Data Integration 1
   3.1 Schema Matching and Data Transformation 2
      3.1.1 Schema Matching 2
      3.1.2 Data Transformation 2
   3.2 Duplicate Detection 3
      3.2.1 Description Selection 3
      3.2.2 Duplicate Detection 3
   3.3 Data Fusion 4

4 Conclusion 5

5 References 5
1 Introduction

This work is a summary of the research paper 'Data Fusion in Three Steps: Resolving Schema, Tuple and Value Inconsistencies' which has been authored by Felix Naumann, Alexander Bilke, Jens Bleiholder and Melanie Weis. The research paper is a part of the Bulletin of the IEEE Computer Society Technical Committee on Data Engineering.

2 Motivation

Data is often stored in heterogeneous formats i.e. in different schemata. Identical real-world objects have multiple representations which may cause duplicates and conflicts of values.

The research paper proposes an array of methods which have been combined in a single tool known as the Humboldt Merger or HumMer. This tool has been created to fuse data from heterogeneous sources. It also provides the option of visualizing every intermediate step and users may interfere where they deem necessary. The results obtained from schema matching can be adjusted, tuples which have been found to be border-line duplicates can be removed and data conflicts can be resolved manually.

3 Data Integration

The paper puts forth the following three steps for integration of data:

Figure 1: The three steps of data fusion

Figure 1: Source:'Data Fusion in Three Steps: Resolving Schema, Tuple and Value Inconsistencies'
3.1 Schema Matching and Data Transformation

Rarely does it happen that the data sources to be integrated belong to the same schema. The first step for data integration is the resolution of schematic heterogeneity which consists of two sub-steps: schema matching and data transformation.

3.1.1 Schema Matching

This involves detection of correspondences between attributes of two heterogeneous schemata. The DUMAS schema matching algorithm has been proposed for this. A few duplicates in unaligned databases are found. Based on similar attribute values of the duplicates, a schema matching is derived. The challenge faced in this step is that it is unknown as to which attribute values are to be compared. But the goal of this phase is not to detect all the duplicates but only those needed to match the schemata. Detection of all duplicates is done in later stages.

DUMAS considers a tuple to be a string and finds the most similar tuple pairs by using the TF-IDF similarity measure. Empirical records show that the most similar tuples are truly duplicates.

However, it is possible that two non-corresponding attributes may have similar values. Hence, several duplicates are used. The fields of two duplicates are compared with the Soft TF-IDF similarity metric. The result is a matrix that contains similarity scores for each pair of attributes. Correspondences below the threshold similarity score are removed. The matrices are averaged, maximum weight matching is done and a set of 1:1 correspondences is produced.

3.1.2 Data Transformation

Data is transformed so that it appears under a common schema. One schema is assumed to be the preferred schema which decides the names of the attributes that appear in multiple sources. The attributes of the non-preferred schema that take part in a correspondence are renamed accordingly. An additional attribute of ‘sourceID’ is added to every table. The full outer union of all the tables is computed.

Figure 2: Schema matching using duplicates

Figure 2: Source: ‘Data Fusion in Three Steps: Resolving Schema, Tuple and Value Inconsistencies’
3.2 Duplicate Detection

Duplicates are detected using not only the attribute values but also interesting attributes called 'descriptions' that are from relations which have some relationship to the table under consideration.

3.2.1 Description Selection

The attributes that are considered interesting for duplicate detection are usually:

- related to the object under consideration
- usable by the similarity metric
- likely to differentiate duplicates from non-duplicates

The children tables are also considered for selecting descriptions. The children tables are matched pairwise and the final result is a transitive closure over the pairwise matches. For example, let tables T1 and T2 be the two matched tables, and let T1,1, . . . , T1,k and T2,1, . . . , T2,m be their respective children tables. Then, matching is done for every pair of tables (T1,i, T2,j) where, 1 ≤ i ≤ k, i ≤ j ≤ m.

![Diagram](Image)

Figure 3: Matching children tables to improve duplicate detection

Figure 3: Source: ‘Data Fusion in Three Steps: Resolving Schema, Tuple and Value Inconsistencies’

3.2.2 Duplicate Detection

A similarity metric sim(t1, t2) is chosen and the tuples are compared. The proposed similarity measure takes into account:

- matched vs. unmatched attributes
- data similarity between matched attributes using edit distance and numerical distance functions
A filter can be used to reduce the number of comparisons. An upper bound to the similarity measure is specified. Hence, if the similarity for a pair is more than the threshold, it is taken to be a possible duplicate.

Pairs identified as possible duplicates are presented to the user by the HumMer tool. Users may choose to manually classify the given pairs.

A transitive closure over the sure duplicate pairs is created and clusters of objects representing a single real-world entity are obtained. An objectID column is added for identification. Conflicts among duplicates are resolved in the next step i.e. conflict resolution.

### 3.3 Data Fusion

This is the last step in the process of data integration. Data fusion tries to resolve any existent conflicts and combines various representations of a real-world object to give one single representation.

Talking specifically about the HumMer system, it takes a number of tables containing multiple representations of an object as the input and outputs one table with exactly one representation for each real world object. This is done by grouping and aggregation. For a group, conflicts may arise in the columns not used for grouping. They can be resolved by the use of a conflict resolution function such as standard SQL aggregation functions (min, max, sum, etc) or other functions that use data from other attributes. An example of the functions that can be used are:

- **Max / Min**: Returns the maximum/minimum value of the conflicting data values.
- **Group**: Returns a set of all conflicting values and leaves resolution to the user.
- **Shortest / Longest**: Chooses the value of minimum/maximum length.
- **Vote**: Returns the value that appears most often among the present values. Ties could be broken by a variety of strategies, e.g., choosing randomly.
- **First / Last**: Takes the first/last value of all values, even if it is a null value.
- **Coalesce**: Takes the first non-null value appearing.
- **Choose(source)**: Returns the value supplied by the specific source.
- **Most Recent**: Recency is evaluated with the help of another attribute or other metadata.

An example of what an SQL-like fusion operation would like:
SELECT Name, RESOLVE(Age, max), RESOLVE(Address, choose(EE_Students)) FUSE FROM EE_Students, CS_Students FUSE BY (Name)

This fuses two student tables. Suppose, a conflict arises in the age of the student. The max criterion can be applied to resolve this conflict as a student’s age only increase and not decreases. Incase a conflict arises in the address, the address given in the source database EE_Students is considered.

4 Conclusion

The research paper gives an outline of a few approaches to data integration. The afore-mentioned steps have been aggregated into a research tool HumMer which is a part of the Merging Autonomous Content project at Humboldt Universitat, Berlin.

5 References


4. The Humboldt-Merger http://www.hpi.uni-potsdam.de/naumann/hummer/ (Last accessed on 23 May, 2014)