Information Fusion 27 (2016) 1–18

Contents lists available at ScienceDirect

Information Fusion

journal homepage: www.elsevier.com/locate/inffus

Dynamical order construction in data fusion

Antoon Bronselaer [a,*], Marcin Szymczak [b,a], Sławomir Zadrożny [b], Guy De Tré [a]

[a] Department of Telecommunications and Information Processing, Ghent University, Sint-Pietersnieuwstraat 41, Ghent, Belgium
[b] Systems Research Institute, Polish Academy of Sciences, Newelska 6, Warsaw, Poland

ARTICLE INFO

Article history:
Received 21 November 2014
Received in revised form 19 February 2015
Accepted 8 May 2015
Available online 14 May 2015

Keywords:
Data fusion
Order relation
Relational databases
Knowledge base construction

ABSTRACT

A crucial operation in the maintenance of data quality in relational databases is to remove tuples that mutually describe the same entity (i.e., duplicate tuples) and to replace them with a tuple that minimizes information loss. A function that combines multiple tuples into one is called a fusion function. In this paper, we investigate fusion functions for attributes of which the values can be sorted by means of an order relation that reflects a notion of generality. It is shown that providing such an order relation a priori, let alone keeping it up-to-date, is a costly operation. Therefore, the Dynamical Order Construction (DOC) algorithm is proposed that constructs an order relation in an automated fashion upon inspecting the data that need to be fused. Such order relations can be immediately deployed in a framework of selectional fusion functions, which are fusion functions that adopt the sort-and-select principle. These fusion functions are investigated closely in terms of their selection strategies. An experimental evaluation of our method shows the influence of the parameters and the benefit with respect to using a fixed and predefined taxonomy.

1. Introduction

In the past decades, the handling of duplicate data has gained a lot of attention in the literature [1,14,16,48,38]. Informally, the problem of duplicate data (also known as record linkage or entity resolution) stems from the fact that multiple descriptions of the same real world entity within one (or more) database(s) may exist. Usually, a solution to the problem of duplicate data comprises two distinct steps: (i) detection and (ii) fusion. The first step deals with the question "Are two descriptions referring to the same entity?" and has been the topic of much research throughout the past decades [18,24]. The second step deals with the question "How can duplicate descriptions be combined?" Hereby, there are several possibilities in how the result of such a combination must look like [6]: it could be a single description or a set of descriptions that contains as little redundant information as possible. In addition, there are several ways in how the result of fusion is used: it could be used to replace duplicate information in the database (possible using data lineage to link back to original value) or it could be stored in a table/view where it can be retrieved later on, for example to support duplicate free query results. Within the scope of this paper, a contribution is made in solving the fusion step by proposing a novel fusion function that combines multiple descriptions into a single description.

1.1. Problem illustration

As a running example throughout this paper, we consider a relational database of Points Of Interest (POIs). POIs are annotations of a geographical map that pinpoint locations of specific interest. Table 1 shows a sample of the central table that stores the main information about the POIs: name, longitude, latitude and type. The tuples shown in Table 1 between horizontal lines are mutual duplicates: they all refer to the same location.

Provided that the tuples between horizontal lines shown in Table 1 have been detected as duplicates, the next step in data cleansing is to combine them into one tuple that best represents the information about the referred location. As we will formalize in the following, this is usually done by projecting the tuples over their attributes and treating the attributes separately. For the POI name, a possible solution is to choose the most frequent name. For longitude and latitude, a summarizing function like the median could be considered.

However, for the POI type, finding a representative value is less trivial. In order to explain this, let us begin with noting that the type of a POI is an ordinal attribute [43] of which the values are (partially) ordered by means of a generalization/specialization

* Corresponding author.
E-mail addresses: antoon.bronselaer@ugent.be (A. Bronselaer), marcin.szymczak@ibspan.waw.pl (M. Szymczak), slawomir.zadrozny@ibspan.waw.pl (S. Zadrożny), guy.detre@ugent.be (G. De Tré).

http://dx.doi.org/10.1016/j.inffus.2015.05.001
1566-2535/

Table 1
Example of duplicate POIs. All tuples between two horizontal lines describe the same POI.

Name	Lon.	Lat.	Type
Belfry	3.725098	51.053552	Tower
Belfry	3.724837	51.053555	POI
Belfrey	3.724911	51.053653	Monument
Korenlei 2	3.720472	51.055569	Hist. Bld.
Korenlei 2	3.720472	51.055568	Restaurant
Ghent	3.715449	51.025529	City
Gent	3.715449	51.025529	POI
Castle	3.721106	51.056984	Monument
Castle	3.721100	51.056981	POI

relation. Next, an important question is: "What causes variations in the attribute values for duplicate POIs?" In case of the POI name, variations can be attributed to spelling errors, abbreviations, etc. In case of the POI type, variations are caused by *subjectivity*. Whereas one agent decides that the Belfry of Ghent is a monument, another might decide that monument is not an adequate type and instead annotates it with the general type "POI". In this setting, the taxonomical connection between the values of POI type can be used as a basis for fusing them. However, in order to deploy such a strategy, a number of problems occur that deserve attention:

- How do we cope with the fact that, in many cases, the taxonomical structure that connects the values of an attribute is not known to the fusion function?
- How do we induce a fusion function from a generated taxonomical structure?
- Should the fusion aim at more specific or more general information?

Within the remainder of this paper, these questions shall be answered by developing a framework for fusion functions, in which order relations can be generated automatically.

1.2. Contributions

With respect to the problems given above, the following important contributions are made by this paper. An algorithm is proposed for the construction of an order relation over the domain of an attribute in an *automated* fashion. The input for this algorithm is a list of observed duplicates in a dataset. The generated order relation can be used to support fusion of duplicate values for that attribute. Our approach has the advantages that there is no need for a priori taxonomical knowledge on the attribute domain and that the order relation automatically adapts to the values in the dataset. The paper also studies how an order relation can best be used in the context of fusion. A new strategy for selection is proposed called *balanced* selection. The behaviour and (dis) advantages of our methods are investigated experimentally on real life datasets.

1.3. Paper outline

The remainder of this paper is organized as follows. In Section 2, an extensive overview of work related to the topic of this paper is provided. Next, in Section 3, some preliminary concepts are introduced that serve as a theoretical foundation of this paper. In Section 4, a framework of fusion functions is introduced in order to formalize the problem that is studied here. The basic solution to this problem is provided in Section 5, where an algorithm for Dynamical Order Construction (DOC) is introduced. The important features of this algorithm are pointed out and the parameters are discussed. In Section 6, the usage of the DOC method in the context of fusion functions is investigated and selection strategies are evaluated. In Section 7, an experimental study of the proposed methods and techniques is reported and supported by careful statistical analysis. In Section 8, some possibilities for future work are discussed. Finally, Section 9 summarizes the most important contributions of this paper.

2. Related work

In the past decades, the problem of fusing duplicate data in (relational) databases has been studied by many authors. In the first place, a number of authors consider the usage of relational operators in order to remove redundant data. Galindo-Legaria et al. [22,23] use the union operator followed by removal of subsumed tuples. Yan and Özsu [55] propose the match join operator which is a combination of union and join. Greco et al. [25] have investigated a similar approach. Bleiholder and Naumann [6] have proposed the FUSE BY-operator as an extension of the SQL syntax to support redundancy removal operations. Further development of that approach is reported in [8,4,5].

Apart from the usage of relational operators, some authors have been investigating stand-alone integration systems. Bilke et al. [3] and Naumann et al. [39] have proposed an integrated system (HumMer) that allows the semi-automated integration of heterogeneous data sources. It uses three steps of data integration: schema matching, duplicate detection and data fusion. In their work, they mention several functions for resolving inconsistencies that can be interpreted in terms of fusion functions discussed later in this paper. Motro and Anokhin [35] have approached the problem of data fusion as a multi-dimensional optimization problem. In their framework called *FusionPlex*, Motro et al. propose to use a utility function that is a linear combination of six metadata dimensions in order to tackle the problem of data fusion. Whereas the approach by Motro et al. assumes the data to be stored in a relational database, other approaches weaken this assumption and also consider semi-structured data by providing a data transformation layer (wrappers) [41].

Some authors have investigated the use of taxonomies and/or ontologies in the scope of data integration. In [37], a classification of possible semantical conflicts in (heterogeneous) databases is presented. Lu et al. [33] have investigated the automated construction of arithmetic-based conversion functions for numerical data. In their work, Lu et al. adopt correlation analysis for conflict detection and linear regression for conflict resolution. The resulting conversion functions are shown to allow for a mapping between different monetary rating systems. The initial framework of conversion functions is further developed in [19], where context awareness is taken into account in order to enhance the conversion functions. Ram and Park [42] have developed SCROL: a standard ontology to facilitate semantic translations between heterogeneous databases. Such an ontology allows to detect both data level conflicts (e.g., representation, precision...) and schema level conflicts (naming conflicts, entity identifier conflicts...). A similar approach has been investigated by Liu et al. [31]. In [27], the usage of ontologies is studied to support conflict resolution in query languages for heterogeneous databases. Bleiholder and Naumann [6] consider the selection of the most general and the most specific value as two of their conflict resolution strategies. They mention explicitly the usage of an ontology to infer a ranking of values to be fused. Dong and Naumann [15,13] and Berti Equille et al. [2] have investigated the impact of source dependence on data fusion. Their work focusses on truth discovery among multiple sources that need to be integrated. For further readings on the usage of ontologies in data integration, the reader is referred to the overview paper by Wache et al. [26].