An MAS Negotiation Support Tool for Schema Matching (Demonstration)

Nguyen Quoc Viet Hung*, Xuan Hoai Luong*, Zoltán Miklós†, Tho Quan Thanh‡, Karl Aberer*
*École Polytechnique Fédérale de Lausanne
{quocviethung.nguyen,hoai.luong,karl.aberer}@epfl.ch
†Université de Rennes 1
zoltan.miklos@univ-rennes1.fr
‡Ho Chi Minh City University of Technology
qttho@cse.hcmut.edu.vn

ABSTRACT

Database schema matching is the process of establishing correspondences between attributes of schemas for data integration purpose. Though various commercial tools have been developed, their results are inherently uncertain. In practice, to obtain correct attribute correspondences, there is a need for collecting human input, after the use of automatic matching tools, to reconcile erroneous mappings. We present a negotiation support tool that enables not a single expert but an expert team, whose members might have conflicting views, can work collaboratively to reconcile the output of the automatic tools. In an attempt to facilitate and support cooperation in team integration, our tool sets the goal to compute all possible decisions from expert inputs as well as explanations for each decision. Moreover, it also shows the foreseeable consequences of choosing a particular decision. Technically, this tool is developed on top of an argumentation framework.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

MAS Applications, Decision Support Systems

Keywords

Argumentation, Data Integration, Team Integration, Schema Matching

1. INTRODUCTION

Schema matching is a major concern in most enterprise applications to integrate data sources. Consider a business scenario (provided by SAP [1]) in which similar partner information is stored in three different database schemas (Figure 1). The first schema (labeled SRM) belongs to the purchasing department, the second one (labeled CRM) stems from the support department, while the third schema (labeled MDM) is located at an external service provider. Since the data are distributed among different tables, we need to establish the correspondences between the schemas of these tables. Figure 2 shows the mappings generated by automatic matchers [1]. According to common properties of this application (e.g. transitive closure of attributes), there are some problematic correspondences that need to be reconciled. In case of larger schemas, the problem is even more significant. We attempt to deal with this problem by employing a team integration approach.

Team integration is performed by a (small) group of experts who might have different opinions about the correctness of certain attribute correspondences. Their inputs (feedbacks) thus inevitably involve conflicts. Because of such conflicts, aggregating these inputs generates many alternative decisions. As a result, the experts need to negotiate to agree on a unique decision. For the purpose of facilitating this process, we develop an *argumentation-based negotiation support tool* (ArgSM) in which each expert plays the role of an agent. Our tool provides the following functionalities:

- *Aggregate and visualize expert inputs.* The inputs come from various experts. Hence, there is a need for a unified view with which experts compare their inputs with those of the others.
- *Generate and explain possible decisions.* The inputs might involve conflicts that make available various decisions. To support negotiation, our tool automatically generates all possible decisions along with explanations as the insights.
- *Compute the consequences of decisions.* When examining a particular decision, the experts naturally look ahead for consequences. ArgSM computes the consequences to assist them by computing the consequences in advance.

ArgSM leverages the theoretical advances and multiagent nature of *argumentation* [2, 5]—in which decisions and explanations are modelled as *claims* and *arguments*, respectively. We expect that, by showing explanations, the expert would trust more his own decisions and those of the others, resulting in a rapid negotiation process. Next, we first describe the system overview in Section 2 and then provide the implementation details in Section 3. Next, Section 4 presents some demonstrations. Finally, Section 5 concludes the paper with some discussions. A runnable JAR of ArgSM is available at [3] while the demo video is on YouTube at [4].

2. SYSTEM OVERVIEW

We show a simplified architecture of our system in Figure 1. The *Input Modeling* receives and models expert inputs with argumentation. Then, it detects the conflicts in these inputs and shows all possible decisions together with their explanations. Finally, the *Input Reviewing* evaluates the strength of explanations, ranks the associated decisions, and shows the entailment of each decision.

[1] http://www.sap.com

[3] https://code.google.com/p/argsm/wiki/ArgSM

[4] http://www.youtube.com/watch?v=rlCxZT5HsMg

Appears in: *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2013)*, Ito, Jonker, Gini, and Shehory (eds.), May, 6–10, 2013, Saint Paul, Minnesota, USA.

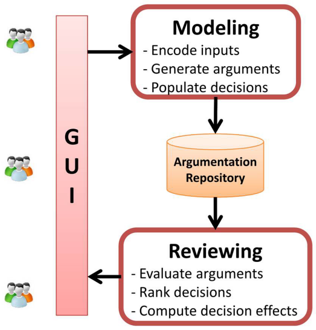

Figure 1: Architecture of ArgSM

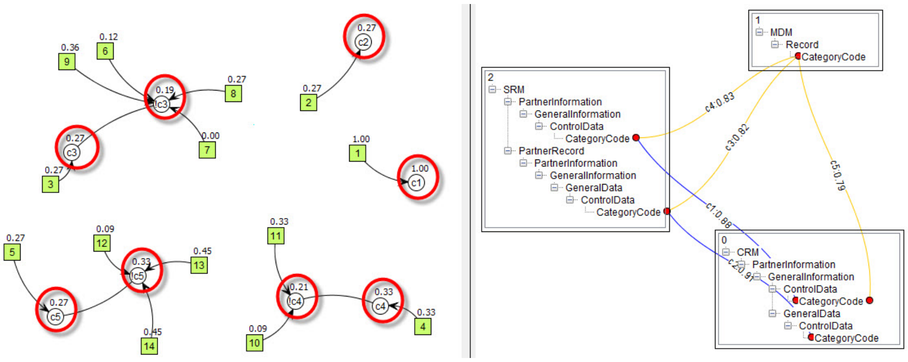

Figure 2: Argumentation view (left) and schema view (right) for the running scenario

Input Modeling. This component collects expert inputs. A typical input has three elements: (1) *object* (a particular correspondence), (2) *value* (approved or disapproved), and (3) *provenance* (identity of the expert who gives this input). After that, the inputs are encoded as formulae in propositional logic. Then, a set of arguments are generated from those formulae. We gather arguments sharing the same claim into disjoint groups. In each group, a claim is a decision whose explanations are the arguments.

Input Reviewing. As there are many possible decisions, the experts need to negotiate to reach an agreement. However, the number of inputs might easily overwhelm them and hinder the negotiation process. This component aims to support the negotiation process by evaluating arguments and ranking possible decisions. The metrics to evaluate arguments rely on *acceptability semantics* [4].

In our system, we also build an argument repository to store the generated arguments and decisions.

3. IMPLEMENTATION

ArgSM is developed by using the Java programming language and the JUNG[5] library (for the visualization). To generate arguments and compute acceptability semantics, we use Vispartix[6] and ASPARTIX[4] respectively. Both rely on Answer Set Programming.

When implementing this tool we had to cope with a number of scalability issues. The schemas are usually too large, leading to high response time (i.e. computation time) for each human interaction and overwhelming control for the experts. To overcome these challenges, we apply the following techniques:

- **Partitioning:** we divide the correspondences into small disjoint and independent subsets such that any two correspondences in the same subset share a common attribute.
- **Caching:** we apply the view maintenance technique [3] in which a repository is used to store intermediate results along the whole process. The rationale behind this is that team integration is an incremental process where a modification (insertion or removal) only affects a few arguments. Thus, it is imprudent to recompute all arguments per modification.

4. DEMONSTRATION

ArgSM provides two views giving the insights of expert inputs:

- **Schema view:** shows all the inputs and the related correspondences.
- **Argumentation view:** shows all possible decisions (aggregated from the inputs) and the associated witnesses (i.e. arguments) that explain the reason for making a decision.

These two views are displayed alongside each other in the unified GUI. Upon clicking on a correspondence in the schema view, the experts can see all generated decisions (circle shapes) and witnesses (square shapes) in the argumentation view. Moreover, we also show the number (outside the shape) that indicates the strength of each decision and witness. In contrast, when choosing a witness in the argumentation view, the experts will see all the involved correspondences in the schema view. We believe that these views together will help the experts to review the inputs and make the final decisions more effectively.

For a better understanding and stronger feelings of trust, ArgSM not only generates explanations but also provides the foreseeable effects of each decision. Technically, we keep the strength of arguments and the possible decisions up-to-date during negotiation.

5. DISCUSSION

We have developed a MAS negotiation support tool that facilitates the team integration process for schema matching problem. This tool focuses on providing the explanations and the effects of possible decisions. We believe that showing the explanations would improve the expert's trust in his own decisions and those of the others. This makes the negotiation of the experts more transparent.

Acknowledgement

This research has received funding from the NisB project (FP7-ICT-256955), and the European Commission in the Planet-Data NoE (contract nr. 257641). Many thanks to SAP [1]and MOM [7] for providing usecases and datasets as well as evaluating this tool.

6. REFERENCES

[1] Philip A. Bernstein, Jayant Madhavan, and Erhard Rahm, *Generic schema matching, ten years later*, PVLDB **4** (2011), no. 11, 695–701.

[2] Philippe Besnard and Anthony Hunter, *Elements of argumentation*, The MIT Press, 2008.

[3] Jose A. Blakeley, Per-Ake Larson, and Frank Wm Tompa, *Efficiently updating materialized views*, SIGMOD Rec. (1986), 61–71.

[4] Martin Caminada and Gabriella Pigozzi, *On judgment aggregation in abstract argumentation*, AAMAS **22** (2011), no. 1, 64–102.

[5] Phan Minh Dung, *On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games*, Artif. Intell. (1995), 321–358.

[5] JUNG library - http://jung.sourceforge.net/applet/index.html

[6] http://www.dbai.tuwien.ac.at/proj/argumentation/

[7] http://www.momentumni.org