A Survey on Automated Schema Matching Tools and Techniques for Heterogeneous Database Schemas

Dr. Dhaval Joshi[1]
[1]*Department of ICT, Veer Narmad South Gujarat University, Surat, Gujarat, India*
(E-mail: joshi.dhaval@hotmail.com)

***Abstract*—**Schema matching is now an essential task for wide variety of domains to fulfill various industrial needs. It can be used for data warehousing, schema integration, data synchronization between two applications, etc. It is always seems as a time consuming and tedious task for large scale schemas of different types as they contains heterogeneity at different levels. In this paper, some of those known tools and techniques are discussed which are designed or developed for database oriented schema matching. From the review of those tools and techniques, observations are described in research gap analysis which shows lot of work still needed to automate schema matching process for heterogeneous databases.

Keywords*—Schema Matching; Heterogeneous Schemas*

INTRODUCTION

Schema matching is a process of defining mapping or identifying correspondence between two schemas. It is a basic requirement for data oriented applications and widely used for different operations of data warehousing, schema integration, data synchronization between two applications, etc. In many cases, it seems always a tedious and time consuming task for large applications when schemas are heterogeneous in nature. Schema heterogeneity can be defined at various levels like type of data, values of data, name of storage element, structure of data storage, constraints on data and dependency of data elements.

To store data, there are various types or formats exist in real world such as XML, SQL and OWL. There are also several types of SQL based databases exist in market like MSSQL, MySQL, DB2, Oracle, etc. All such types of database have different storage structure and storage types to store data. Furthermore, these databases are widely used for application development for large scale data storage. In order to match large schemas with all such types of heterogeneity, automated schema matching is better than manually mapping all schema elements to improve quality of mapping with less human efforts.

Since long back, schema matching is performed to fulfill all such requirements with the help of automated and semi-automated tools and techniques designed or developed by different people. In this paper, some of existing automated or semi-automated schema matching tools and techniques are discussed in the literature review section.

LITERATURE REVIEW

There have been several tools and techniques exist for schema matching of heterogeneous data storages. In this paper, special focus is given to some of those tools and techniques which are designed and/or developed for relational databases.

Harmony is an innovative, open source semi automated schema matching tool, available both as a standalone and as part of the OpenII information integration tool suite [1,2]. It works with a wide variety of data models, including those expressed as XML Schema (XSD files), SQL data definition language, OWL, spreadsheets with column headings, and others. In addition to matching across schemas, it has been used successfully to speed matching across separately developed code lists. It uses match voter strategy to identify expected matching within given schemas which include techniques like Bag of words, Edit distance, Thesaurus, Exact structure matcher and user defined customized matcher. It can perform automatic matching of schema elements based on the selected technique(s) by the user and it allows user to accept or reject the expected matching pair.

CUPID [3] represents a sophisticated hybrid match approach combining a name matcher with a structural match algorithm, which derives the similarity of elements based on the similarity of their components hereby emphasizing the name and data type similarities present at the finest level of granularity (leaf level). To address the problem of shared elements, the schema graph is converted to a tree, in which additional nodes are added to resolve the multiple relationships between a shared node and its parent nodes. CUPID returns element-level correspondences of 1:1 local and n:1 global cardinality.

COMA3 is a schema and ontology matching tool. It extends previous prototypes COMA and COMA++ by an enhanced workflow management and additional features like ontology merging. Furthermore, it offers a comprehensive infrastructure to solve large real-world match problems. The COMA project was first released in 2002, and after that within a decade got gradually extended and improved for schema matching [4]. It follows a composite approach, which provides an extensible library of different matchers and supports various ways for combining match results. The matchers exploit schema information such as element and structural properties. Furthermore, a special matcher is provided to reuse the results from previous match operations. The combination strategies address different aspects of match processing, such as, aggregation of matcher-specific results and match candidate selection. Schemas are transformed to rooted directed acyclic

graphs, on which all match algorithms operate. Each schema element is uniquely identified by its complete path from the root of the schema graph to the corresponding node [5], [6]. It produces element-level matches of 1:1 local and m:n global cardinality. COMA/COMA++ have used rule based techniques for schema matching.

Aumueller et. el. [7] presented rule based techniques for schema matching as COMA/COMA++ as generic schema and ontology matching systems where simple, hybrid and reuse oriented matchers are used. In the systems, schemas are internally encoded as DAGs (Directed Acyclic Graphs) and are analyzed using string matching algorithms. Different aggregation functions such as average, minimum, maximum and weighted sum along with rule based techniques are used in the systems for obtaining combined match results. However, in COMA/COMA++, determining best combination of matcher is not easy.

LSD [8] and its extension GLUE [9] use a composite approach to combining different matchers. While LSD matches new data sources to a previously determined global schema, GLUE performs matching directly between the data sources. Both use machine-learning techniques for individual matchers and an automatic combination of match results. In addition to a name matcher, they use several instance-level matchers, which discover during the learning phase different characteristic instance patterns and matching rules for single elements of the target schema. The predictions of individual matchers are combined by a so-called meta-learner, which weights the predictions from a matcher according to its accuracy shown during the training phase. The match result consists of element-level correspondences with 1:1 local and n:1 global cardinality. Both systems use machine learning techniques like Multi-strategy learning approach as base learner, Naïve Bayes for classifying text, and Meta learner for finding matching among a set of instances.

SF - Similarity Flooding [10] converts schemas (SQL DDL, RDF, XML) into labeled graphs and uses fix-point computation to determine correspondences of 1:1 local and m:n global cardinality between corresponding nodes of the graphs. The algorithm has been employed in a hybrid combination with a simple name matcher, which suggests an initial element-level mapping to be fed to the structural SF matcher. Unlike other schema-based match approaches, It does not exploit terminological relationships in an external dictionary, but entirely relies on string similarity between element names. In the last step, various filters can be specified to select relevant subsets of match results produced by the structural matcher.

Clio [11], the IBM Research system for expressing declarative schema mappings, has progressed in the past few years from a research prototype into a technology that is behind some of IBM's mapping technology. It provides a declarative way of specifying schema mappings within either XML or relational schemas. Mappings are compiled into an abstract query graph representation that captures the transformation semantics of the mappings. The query graph can then be serialized into different query languages, depending on the kind of schemas and systems involved in the mapping. It produces XQuery, XSLT, SQL, and SQL/XML queries.

ACM - Auto Mapping Core [12], a framework that supports fast construction and tuning of schema matching approaches for specific domains such as ontology alignment, model matching or database-schema matching. Distinctive features of the framework are new visualization techniques for modeling matching processes, stepwise tuning of parameters, intermediate result analysis and performance oriented rewrites. Furthermore, existing matchers can be plugged into the framework to comparatively evaluate them in a common environment. This allows deeper analysis of behavior and shortcomings in existing complex matching systems.

Wan et. el. [13] presented a schema matching algorithm based on partial functional dependencies using genetic algorithm. In this approach, partial functional dependencies are identified from relational databases and it is mixed with the cupid and similarity flooding concept to improve results of schema matching.

Nikovski et al. [14] presented Bayesian networks based automatic schema matching method. This method creates composition of matcher model based on statistical correlation between similarity values produced by individual matcher which uses same or similar information.

Zahra et al. [15] presented a hybrid semantic schema matching algorithm by exploiting WordNet lexicon database which semi-automatically finds matching between two schemas. Their algorithm tried to find best quality matches and overcomes to semantic ambiguity over other existing algorithms.

Gillani et al. [16] defined taxonomy of all possible semantic similarity measures and also proposed an approach that exploits semantic relations stored in the DBpedia dataset while utilizing a hybrid ranking system to dig-out the similarity between nodes of two graphs.

Chenlu et al. [17] proposed a multilayer schema matching approach. In this approach, first layer finds out semantic similarity using lexicographic similarity measure, second layer uses functional dependency to form structural information of schemas and last layer finds matching using probabilistic factor of matching.

Embley et al. [18] develop an approach based on learning rules of decision trees for discovering hidden mapping among entities. In this approach, the rules are used for matching terms in WordNet. However, the decision trees are not used for choosing the best match algorithms.

Duchateau et. al.[19] presented a decision tree based approach for schema matching to combine the best suitable match algorithms. In this approach, a set of schemas and a decision tree are passed as input for schema matching and after processing on given schemas using defined decision tree, it can generated list of mappings as output. The received output (mapping between schemas) should be validated by experts to find out it significance of correctness. Expert feedback can be feed into another decision tree for learning. They have defined the rule based approach as a better solution compared to the machine learning technique for schema matching as rule based approach doesn't require manual generation of refined training dataset.

YAM [20] is a schema matching factory using machine learning. It considers users' requirement like preference for recall or precision for learning phase with expert correspondences. It uses Knowledge Base to match unknown schemas in the matching phase. The Knowledge Base consists of a set of similarity measures, a set of classifiers and repository of already matched pairs of schemas. It allows users to select appropriate classifier and if no classifier is defined by user, it selects default classifier as per Knowledge Base. The hybrid approach by combining rule based technique and decision tree is used to design matching model.

KSMS [21] uses Hybrid-RDR [22] approach that combines both machine learning and incremental knowledge engineering approaches for matching entities at the element level using ontology features. KSMS combines decision tree, Censor Production Rules (CPR) based Ripple Down Rules (RDR), J48 and incremental knowledge engineering approach. It allows users to correct and validate the matching results automatically. For structure level matching it uses Similarity Flooding to match the hierarchical structure of a full graph. The final mapping result is produced by applying aggregation function such as Harmony-mean on the results of both levels.

Khalid et. el. [23] introduced a technique for large scale schema matching using tree mining. They investigated scalability with respect to time performance in the context of approximate mapping where tokenization, abbreviations and synonyms were used for the linguistic matching of node labels. The matching strategy was hybrid and optimized for schemas in tree format. In their technique, they labeled each node and assign values to it, which is complex formation of tree and also need more computation for schema matching.

Anan et. el. [24], proposed a machine learning approach SMB that uses the Boosting algorithm to classify the similarity measures. The Boosting algorithm converts weak classifiers to strong one. By iterating weak classifiers over the training set, the boosting algorithm composes a strong classifier while re-adjusting the importance of elements in this training set. Thus, SMB automatically selects a pair of similarity measures as a matcher by focusing on harder training data. An advantage of this algorithm is the important weight given to misclassified pairs during the training. Although this approach makes use of several similarity measures, it mainly combines a similarity measure (first-line matcher) with a decision maker (second-line matcher). Their empirical results show that the selection of the pair does not depend on their individual performance.

Feng et. el. [25] proposed a new approach of instance based schema matching based on the hypothesis that corresponding attributes are relatively equally important. The main components of their three-part framework: attribute ranking, attribute classification and matching phase. In contrast to traditional approaches, which consider all attributes with the same importance, they employ machine learning methods in prioritizing all schema attributes according to rank and class. When matching, they have constructed an optimal objective function to determine all equivalent attributes. However, their approach is suitable only for numeric instances, as the result of precision (P) dropped to 66% when string instances are considered.

RESEARCH GAP

After reviewing various existing tools and techniques for schema matching of heterogeneous databases, their summary is listed in Table-1. Here, they are reviewed with five different characteristics like type of tool or technique, algorithm selection process, data models supported and designed for source Vs target data model type. Type of tool or technique column contains values like semi-automated (SA) or automated (A). Algorithm selection process column contains values like manual (M) or automated (A). Data model column contains values like multiple data models (m) or single data model (S). Matcher type column contains values like hybrid (H), composite (C) or customized (CT). Designed for source Vs target data model type column contains values like same type of data model (Sm) or different type of data model (D). As per the review contained here for above maintained tools and techniques, if any column value in Table-1for specific tools or technique is not observed then it is defined with value undefined (U).

TABLE-1 : SCHEMA MATCHING TOOLS AND TECHNIQUES

Tools / Techniques	Type	Algo. Selection	Data Model	Matcher Type	Source Vs Target
Harmony – OpenII [1,2]	SA	M	m	CT/C	Sm
Cupid [3]	SA	M	m	H	Sm
COMA [4,5,6]	SA	M	m	H/C	Sm
LSD [8] GLUE [9]	SA	A	S	C	Sm
SF [10]	SA	M/A	m	H	Sm
Clio (IBM) [11]	SA	A	m	U	Sm
AMC [12]	A	A	m	U	U
Wan [13]	SA	U	S	H	Sm
Nikovski [14]	A	A	U	C	U
Zahra [15]	SA	U	U	H	U
Gillani [16]	SA	A	S	U	Sm
Chenlu [17]	SA	U	U	H	Sm
Embley [18]	SA	U	S	C	Sm
Duchateau [19]	SA	A	m	U	U
YAM [20]	SA	A	m	H	U
KSMS [21]	SA	M/A	U	H	U
RDR [22]	SA	M/A	m	H	U
Khalid [23]	SA	U	S	H	Sm
Anan [24]	A	A	S	C	Sm
Feng [25]	A	U	S	U	Sm

SA – Semi-Automated, A – Automated, M – Manual, H – Hybrid, C – Composite, CT – Customized, m – Multiple, S – Single, D – Different. U – Undefined

It can be seen from the review of above mentioned tools and techniques for schema matching for heterogeneous databases that they have one or more issues from the following:

- User needs to provide one or more input of information like select elements of schemas, mapping datatype of element(s), etc.
- Mapping algorithm(s) selection is manual.
- Schema matching (or mapping) accuracy is not high.
- Perform over mapping or wrong mapping.
- Designed for same type of data models only.

CONCLUSION

In this paper, several tools and techniques for schema matching of heterogeneous databases are reviewed and their characteristics are listed. It shows that most of them are semi-automated tools which need user input at various stages. Some of them are also automated tools but they are not capable of mapping schemas of different types of databases. Furthermore, as per review of various tools and techniques for schema matching, derived knowledge states that none of the solution is complete for automated schema matching of heterogeneous databases. So, there is lot of scope to design automated solutions for large scale schema matching of heterogeneous databases.

REFERENCES

[1] OpenII : Open Information Integraion. (2013, November). openii.sourceforge.net. [Online]. http://openii.sourceforge.net/index.php?act=tools&page=harmony.

[2] L. Seligman, P. Mork, A. Halevy, K. Smith, M. J. Carey, K. Chen, C. Wolf, J. Madhavan, and A. Kannan, "OpenII: An Open Source Information Integration Toolkit," in ACM SIGMOD'10 International Conference on Management of data, Indianapolis, Indiana, USA, 2010, pp. 1057-1060.

[3] J. Madhavan, P. A. Bernstein, and E. Rahm, "Generic Schema Matching with Cupid," in Proceedings of the 27th International Conference on Very Large Data Bases, San Francisco, CA, USA, 2001, pp. 49-58.

[4] E. Rahm, P. Arnold, H. Do, and D. Aumüller. (2014, January). Schema and Ontology Matching with COMA 3.0. dbs.uni-leipzig.de. [Online]. https://dbs.uni-leipzig.de/en/Research/coma.html

[5] H. Do and E. Rahm, "COMA – A System for Flexible Combination of Schema Matching Approach," in 28th Intl. Conference on Very Large Databases (VLDB), Hongkong, 2002.

[6] H. Do, S. Melnik, and E. Rahm, "Comparison of Schema Matching Evaluations," in Proceedings of the workshop on Web and Databases (2002), Verlag Berlin Heidelberg, 2003, pp. 221-237.

[7] D. Aumueller, H. Do, S. Massmann, and E. Rahm, "Schema and ontology matching with COMA++," in ACM SIGMOD international conference on Management of data, 2005, pp. 906-908.

[8] A. Doan, P Domingos, and H. Alon, "Reconciling Schemas of Disparate Data Sources: A Machine-Learning Approach," in ACM SIGMOD 2001, Santa Barbara, California, USA, 2001.

[9] A. Doan, J. Madhavan, P. Domingos, and A. Halevy, "Learning to Map between Ontologies on the Semantic Web," in WWW-2002, Honolulu, Hawaii, USA, 2002.

[10] S. Melnik, H. Garcia-Molina, and E. Rahm, "Similarity Flooding: A Versatile Graph Matching Algorithm and its Application to Schema Matching," in Proceedings 18th International Conference on Data Engineering, 2002.

[11] L. M. Haas, M. A. Hernandez, H. Ho, L. Popa, and M. Roth, "Clio Grows Up: From Research Prototype to Industrial Tool," in SIGMOD 2005, Baltimore, Maryland, USA., 2005.

[12] E. Peukert, J. Eberius, and E. Rahm, "AMC - A framework for modelling and comparing matching systems as matching processes," in Proceeding of 27th International Conference on Data Engineering (ICDE), 2011, pp. 1304-1307.

[13] J. Wan and D. Xiaokun, "A Genetic Schema Matching Algorithm based on Partial Functional Dependencies," Journal of Computational Information Systems, pp. 4803-4811, 2013.

[14] D. Nikovski, A. Esenther, X. Ye, M. Shiba, and S. Takayama, "Bayesian Networks for Matcher Composition in Automatic Schema Matching," in International Conference on Enterprise Information Systems (ICEIS), Broadway, Cambridge, Massachusetts, 2012, pp. 48-55.

[15] S. Zahra, M. Mohsenzadeh, and M. A. Dezfuli, "An Improved Semantic Schema Matching Approach," Journal of Advances in Computer Engineering and Technology, pp. 29-36, 2015.

[16] S. Gillani, N. Muhammad, H. Raja, and A. Qayyum, "Semantic Schema Matching Using DBpedia," International Journal of Intelligent Systems and Applications, pp. 72-80, 2013.

[17] Z. Chenlu, S. Derong, and K. Yue, "A Multilayer Method of Schema Matching Based on Semantic and Functional Dependencies," in 9th Web Information Systems and Applications Conference (WISA), Haikou, 2012, pp. 223-228.

[18] D. W. Embley, L. Xu, and Y. Ding, "Automatic direct and indirect schema mapping: experiences and lessons learned," in ACM SIGMOD, New York, NY, USA, 2004, pp. 14-19.

[19] F. Duchateau, Z. Bellahsene, and R. Coletta, "A flexible approach for planning schema matching algorithms," in On the Move to Meaningful Internet Systems: OTM 2008, Berlin, Heidelberg, 2008, pp. 249-264.

[20] F. Duchateau, R. Coletta, Z. Bellahsene, and R. J. Miller, "Yam: a schema matcher factory," in 18th ACM conference on Information and knowledge management, 2009, pp. 2079-2080.

[21] S. Anam, Y.S. Kim, B.H. Kang, and Q. Liu, " Designing a Knowledge-based Schema Matching System for Schema Mapping," in Thirteenth Australasian Data Mining Conference, AusDM 2015, Sydney, Australia, 2015.

[22] S. Anam, Y.S. Kim, B.H. Kang, and Q. Liu, "Schema Mapping Using Hybrid Ripple-Down Rules," in The Thirty-Eighth Australasian Computer Science Conference, ACSC 2015, Sydney, Australia, 2015, pp. 17-26.

[23] S. Khalid, B. Zohra, and H. Ela, "Performance Oriented Schema Matching," in 18th International Conference on Database and Expert Systems Applications, 2007, pp. 844-853.

[24] M. Anan and G. Avigdor, "Boosting schema matchers," in OTM 2008 confederated international conferences, CoopIS, DOA, GADA, IS, and ODBASE, Heidelberg, 2008, pp. 283–300.

[25] J. Feng, X. Hong, and Y. Qu, "An InstanceBased Schema Matching Method with Attributes Ranking and Classification," in 6th International Conference on Fuzzy Systems and Knowledge Discovery, NJ, USA, 2009, pp. 522-526.