Methods for Big Data Integration in Distributed Computation Environments

© Vladimir V. Sazontev
Lomonosov Moscow State University,
Moscow, Russia
vladimir.sazontyev@gmail.com

Abstract. One of the main challenges in data integration is to create an extensible end-to-end system, which will perform not only extraction, but also schema alignment and entity resolution techniques. This is even more challenging in a world of the big data, when we have to deal with the large number of heterogeneous data sources. In this case the system has to be automatic and less user depended. This paper aims to overview and analyze the modern approaches and systems to successfully perform big data integration. This work is performed as a master thesis, which is aimed to propose an architecture of the system to perform integration of heterogenous sources in a distributed computation environment, implement and apply it to a real-world problem of e-commerce domain as a part of master thesis.

Keywords: Big data integration, Automatic entity resolution

CEUR-WS.org/Vol-2277/paper41.pdf

1 Introduction

The Web contains ever-increasing massive amount of heterogenous data sources which can consist of structured and unstructured data. Data integration systems aim to integrate massive amount of the sources, but they are still limited to the human resource that is needed to perform data source selection, to develop rules to perform schema alignment and to resolve entities. Some applications provide very good results in automation of these processes in a limited scope of the problem, but their design does not allow performing end-to-end data integration and being extensible.

Data integration is a complex problem, that traditionally consists of three steps: schema alignment, record linkage and data fusion. Each of these steps even harder to perform for the big data integration, because of volume, velocity, veracity and variety of the data and heterogenous sources. The schema alignment aspects are addressed by implementing probabilistic schemas and mappings, profiling and scoring the sources. The record linkage problem is addressed by implementing incremental updates, by using blocking function, combinations of blocking functions and meta blocking, and by implementation of smart crowdsourcing system, by considering time in attribute changes of the entity and applying tagging algorithm to extract additional data from unstructured and semi-structured data. The data fusion step is addressed not only by user defined function, but also with algorithms that detect data copies. Traditionally to resolve the ambiguity on each step, involvement of experienced user required, but for big data integration systems the aim is to automate this work using different techniques. Enterprise data integration systems in general provide us with end-to-end solution, they lack automation in the field of entity resolution and data fusion. The research systems have a wide variety of automation attempts, but in general do not provide end-to-end solutions and are good only for some specific problem.

Last but not least important feature of a modern big data integration system is the distributed computation environment, such as Hadoop or Ignite, which is crucial due to the volume and velocity of sources and data. Different computational models implemented over mentioned environments like Spark and MapReduce can be applied.

This paper aims to overview and analyze the modern approaches and systems to successfully perform big data integration. This work is performed as a master thesis, which is aimed to propose an architecture of the system to perform integration of heterogenous sources in a distributed computation environment, implement and apply it to a real-world problem of e-commerce domain as a part of master thesis. In this paper the related work and architecture are analyzed and overviewed and the current progress of implementation is reported. As future work the proposed list of the methods and approaches is planned to be thoroughly implemented.

In section 2 some of modern methods and approaches to address various aspects of big data integration problem are outlined. In section 3 enterprise and research data integration systems in which approaches from section 2 are implemented are compared. In section 4 the architecture and various design choices are described. In section 5 the implementation of the architecture to the real-world problem and the current progress is described.

2 Methods Supporting Different Aspects of Big Data Integration

2.1 Schema Alignment

One of the major steps in data integration is schema alignment. This step traditionally consists of developing a mediated schema, attribute matching and schema

Proceedings of the XX International Conference "Data Analytics and Management in Data Intensive Domains" (DAMDID/RCDL'2018), Moscow, Russia, October 9-12, 2018

mapping. A mediated schema is created to provide a unified view of the heterogeneous sources. Then attributes in each source schema are matched to the corresponding attributes in the mediated schema. A schema mapping is built between each source schema and the mediated schema, it specifies the semantic relationships between the contents of different data sources and is used to reformulate a query on the mediated schema into a set of queries on the underlying data sources.

2.1.1 Probabilistic Schemas and Mappings

The probabilistic mediated schema [1] can be thought of as a "clustering" of source attributes. Similar attributes of sources are grouped into the same cluster - mediated attribute. The probabilistic schema mapping [2] allows us to describe a probability distribution of a set of possible schema mappings between a source schema and a target schema. Both methods work the best in combination in pay-as-you-go approach, which addresses the variety and velocity problem of big data integration at schema alignment step. Given a query, this combination generates best-effort or approximate answers from data sources where perfect schema mappings do not exist and directs which schema is worth integrating manually.

In [1] the combination of probabilistic mediated schema and probabilistic schema mapping is evaluated on web tables crawled in five domains, where each domain contains 50–800 web tables (i.e., data sources). The methods obtained an F-measure over 0.9 in query answering on every domain comparing with an integration system where schema mappings are manually specified.

2.1.2 Source Scoring - Evaluate Sources Before Alignment

In [3] it is shown that using several real-world data sets is not always worth integrating all available sources. For example, integrating new sources may not increase the coverage significantly, while the total cost will increase. Even worse, some low-quality data sources can even affect the accuracy of integrated data negatively, while still adding to the total cost. In [3] the problem of source selection which is performed before real integration is proposed. This approach balances the cost and benefit of integrating the source. It also shows that source selection in the context of data fusion is NP-complete in general, and that a straightforward greedy algorithm can generate an arbitrarily bad solution.

The greedy randomised adaptive search procedure (GRASP) [4] addresses the limitations of the greedy approach in two ways: firstly, instead of making a greedy decision in every step, secondly, in each repetition, after generating the initial solution, it performs local search in a hill-climbing fashion. Both components make it possible to reach a near-optimal selection. GRASP is significantly better than Greedy in selecting the subset of sources with the highest profit and quite scalable, taking less than 1 hour for synthetic data with up to 1 million sources of various accuracy distributions.

2.1.3 Source Profiling

The goal of source profiling is to effectively address the challenging problem of helping users understand the source contents, before they even decide whether integration needs to be performed [5]. In [6] an approach to summarise the contents of a relational source is proposed, so that users can quickly identify the data domains of the source, and the main tables in which each type of information resides.

The source schema summarisation obtains about 70% accuracy on the three pre-classified categories in [6].

2.2 Record Linkage

The goal of record linkage is to decide which records refer to the same entity, and which refer to different entities. Record linkage consists of three main steps: blocking, pairwise matching (compares a pair or records), and clustering.

The pairwise matching step compares a pair of records to find out whether they refer to the same entity, while the clustering step makes a decision whether the results of the pairwise step are globally consistent. Since pairwise matching requires a quadratic number of record pair comparisons, blocking step aims to decrease the number of pairwise comparisons.

It is also worth mentioning that algorithms based on knowledge and ontology exists, which are not considered in this overview due to the lack of space. Moreover, it does not meet the goal as end-to-end system should work with a large variety of heterogenous sources. In this case, it is not expected that any significant number of data sources meet such level of formalization.

2.2.1 Multiple and Meta Blocking Schemas

The records could be partitioned by using a blocking function that is the composition of values of the attributes. The advantage of this approach, instead of performing pairwise comparison of all records, we do this for subsets of records, which significantly lowers the number of comparisons. The disadvantage is that this strategy may produce false negatives, which prevent from comparison of pairs that refer to one entity.

To address this disadvantage in [7] it is shown that using multiple blocking functions could result in high quality record linkage without necessarily incurring a high cost. In [8] meta-blocking is proposed as an alternative approach to this problem, which builds an edge-weighted blocking graph G, for a set of blocks, where the nodes of G are the records that occur in at least one block of B, and (undirected) edges connect pairs of records that co-occur in at least one block. In [8] it is shown experimentally that this method improves blocking efficiency significantly, often by 1-2 order of magnitude, while preserving high recall.

2.2.2 BlockSplit and PairRange

It is shown in [9] even with blocking, record linkage for big data sets can take significant time span. To address this volume issue, in [10] it is proposed to use MapReduce programming model, which is highly effective in parallelising data-intensive computing in cluster environment. The straightforward approach would be to pass every block as a separate job for reducer, but since blocks are not even sized, it is better to balance the pairwise matching. In [10] two methods - BlockSplit and

PairRange are proposed.

BlockSplit aims to split the blocks and balance the load.

PairRange globally enumerates each comparison and splits it into equal sized tasks.

In [10] it is shown that both PairRange and BlockSplit are stable across all data skews, with small advantage of PairRange and is also stated that BlockSplit and PairRange scale and keep being close to the number of reducers.

Table 1 Methods and Systems for Data Integration

Stage	Sub-stage	Approach	Enterprise Data Integration Systems	Research Data Integration Systems
Schema Alignment		Probabilistic schema mapping		Das Sarma et al. [2] DI 2008
		Probabilistic mediated schema		Das Sarma et al. [2] DI 2008
		Pay-as-you-go		Das Sarma et al. [2] DI 2008
		Source scoring		GRASP 2011 [4]
		Source profiling		Summarize RDB [6] 2009, ITBenchmarking [28] 2017
		Rule-based (user defined)	Talend [18], CloverETL [19], Centerprise [20], Attunity [21], Pentaho ETL[22], Jaspersoft ETL[22], jBoss Teiid [23]	SERF [24] 2009, Hyperion [26] 2016
		Other automatic schema alignment approach		Hyperion [26] 2016, BigGorilla [27] 2017, ITBenchmarking [28] 2017
Record Linkage	Blocking	Balanced Mapreduce		CrowdER [12] 2012-2014
		Multiple blocking schemas	Attunity [21]	Herna´ndez and Stolfo DI [7] 1998
		Meta blocking schema		ERBlockingframework [8] 2014-2018
		Other type of optimization	Talend [18], CloverETL [19]	OYSTER [25] 2012-2016, ITBenchmarking [28] 2017
	Pairwise matching	Connected Component algorithm		CrowdER [12] 2012-2014, Gruenheid et al [11] 2014
		Iterative algorithm		Gruenheid et al. [11] 2014
		Greedy Incremental Algorithm		OYSTER [25] 2012-2016, Gruenheid et al. [11] 2014
		Rule-based	Talend [18], CloverETL [19], Centerprise [20], Attunity [21], Pentaho ETL[22], Jaspersoft ETL[22], jBoss Teiid [23]	Herna´ndez and Stolfo DI [7] 1998
		Other type of pairwise optimization	Talend [18], CloverETL [19]	OYSTER 2012-2016, BigGorilla 2017, ITBenchmarking [28] 2017
	Clustering	Crowdsourcing		CrowdER [12] 2012-2014
		Text file parse (tagging)	Bing shopping and product catalog, 2011 [14]	
		Time consideration in data (agreement/disageement decay)		OYSTER [25] 2012-2016
		Other type of domain methods to refine clusters		OYSTER [25] 2012-2016, ITBenchmarking [28] 2017
Data Fusion		Accucopy		Dong et al 2009 [17]
		Temporal data Fusion		Dong et al 2009 [17], TemporalLinkage [15] 2011
		Third dimension considers extractors		Dong et al 2009 [17]
		Data Profiling (for rule based intuition)	Talend [18], CloverETL [19], Centerprise[20], Attunity [21], Pentaho ETL[22], Jaspersoft ETL[22], jBoss Teiid [23]	Centerprise [20], Attunity [21]

2.2.3 Incremental Record Linkage

In record linkage we face the velocity problem, since each data source update makes results of record linkage obsolete, we must perform record linkage again. It is also important to mention, that besides speeding up the record linkage process, our goal is to preserve the quality. The ConnectedComponent algorithm [11] considers only the clusters in the previous record linkage result that are directly or indirectly connected to the nodes in the update.

The Iterative algorithm [11] starts with the clusters in the previous record linkage result that are directly connected to the nodes in the update and expands it only when necessary.

The disadvantage of clustering component is that the graphs considered may be too large, and the disadvantage of the iterative algorithm is that to converge we may need to consider a large number of such sub-graphs. The Greedy Incremental algorithm [11] addresses this problem. The algorithm considers three possible operations - split cluster, merge clusters and move nodes from one cluster to another based on the lowest penalty value

In [11] the benefits of the incremental algorithms over batch linkage are experimentally shown. The efficiency of record linkage significantly improves, often by 1-2 orders of magnitude. Using a synthetic data shows that the Greedy algorithm is the most robust in noisy environments.

2.2.4 Crowdsourcing

A system is a crowdsourcing system if it enlists a crowd of humans to help solve a problem defined by the system owners. A naive approach to crowdsourcing record linkage would result in human intelligent tasks, which for big data integration is not scalable. CrowdER [12] discards all pairs with low likelihood of being matched, using automatic entity resolution techniques, it also exploits the fact that record linkage satisfies the transitive relations. The key contribution was made by labelling the records in the long chains of such relations, they should be label in decreasing order of likelihood.

In [13] sequential and parallel strategies on two real-world public datasets, using simulation and amazon mechanical turk (AMT) are experimentally evaluated. On the Cora data set of research publications, using transitive relations reduces the number of crowdsourced record pairs by 95%. On the Abt-Buy product data set, about 20% crowdsourced record pairs are saved.

2.2.5 Text File Parsing

Many applications see a need to link unstructured text data, while they have structured data. In [14] a supervised learning approach tagging to create a mapping from unstructured data to structured data is presented. It performs tagging of strings in the text snippet with attribute names and predicting the most promising mapping (tagging).

In [14] experimentally shown that this method is scalable especially with the use of blocking function. This system is deployed and used to match all the offers received by Bing Shopping to the Bing product catalog.

2.2.6 Temporal Record Linkage

Temporal record linkage addresses the problem of record evolution over time which belongs to veracity problem of big data integration. A strategy described in [15] is based on assumption that entities evolves smoothly, this process typically is not erratic and attribute values have continuity property (in the small time gaps it is likely that attributes are less likely to change). This strategy invents two coefficients. Disagreement decay - denotes the probability that one entity changes its attribute value within time gap. Agreement decay - denotes the probability that two different entities have the same attribute value within time gap. Then we multiply these coefficients by similarity of attributes.

In [15] experimentally using DBLP data set it is shown that F-measure improved by 43% over traditional record linkage.

2.3 Data Fusion

The data fusion is the third component of data integration. In [16] a data fusion architecture consists of three steps truth discovery (according to the correctness of its values), trustworthiness evaluation, copy detection between data sources. Due to the lack of space we consider only Accucopy family of algorithms.

2.3.1 Basic Accucopy

In [16] end-to-end AccuCopy algorithm based on architecture from section 2.3 is proposed. The algorithm proposed in [16] was evaluated experimentally using the Flight data set, where copying happens a lot between low-quality sources, most models that consider source accuracy obtain even lower precision than naive voting. AccuCopy, on the other hand, significantly improves the precision of the results over naive voting, by 9.1%.

2.3.2 Temporal Accucopy

In the real-world sources are usually non-static and the truth changes over time. The Temporal Accucopy data fusion decides the true value of the data item at each time. In [17] it is suggested to consider source quality in dynamic setting and the lifespan of each data item. In [17] the dynamic data fusion algorithm is experimentally evaluated using a restaurant data set including over 5K restaurants in Manhattan, crawled from 12 web sources weekly in a period of 8 weeks. In this period 467 restaurants were marked by some source as being closed and among them 280 were indeed closed. The proposed method obtains a F-measure of 0.86; Precision = 0.86 (considering all these restaurants as closed yields Precision =0.60) Recall = 0.87(considering restaurants marked by at least two data sources as closed yields Recall=0.34).

2.3.2 Extractors as Third Dimension in Accucopy

It also worth considering the extractors as a third dimension to address the variety problem as it is proposed in [17], which shows that those being extracted by at least 8 extractors have a much higher accuracy (on average 70% higher) than those being extracted by a single extractor.

3 Software Systems for Big Data Integration

The following data integration systems comparison is divided into two sections - enterprise and research since the enterprise systems in general lack of the advanced methods, and research systems are not as user friendly as enterprise ones.

3.1 Enterprise Data Integration Systems

Table (see Table 1) outlines the main distinguishable features. All seven systems share common, they are all good at data extraction, but schema alignment step heavily relying on a rule-based approach. Only Talend[18] and CloverETL[19] provide some basic automatic tools for entity resolution. Centerprise [20] and Attunity [21] provide only rule-based option, while LANSA is not designed for entity resolution within the system. Pentaho ETL[22], Jaspersoft ETL[22], jBoss Teiid[23] are focused on big data ETL processes. The problem is they are not designed to work with a large number of the heterogenous and schema evolving sources. It is worth mentioning, that Pentaho ETL, Jaspersoft ETL, jBoss Teiid have advanced visual data management tools, which help users to manage the data flow and prepare custom rules for big data integration. They have an ability to be expanded by custom user scripts. All of them, except LANSA provide data profiling to assess the quality of the data sources, based on data.

3.2 Research Data Integration Systems

Table (see Table 1) also outlines the features of the research systems. SERF [24] has fixed schema mappings, but it is worth mentioning their entity resolution approach, where each entry merged based on vote of each metric. OYSTER [25] is an entity resolution system, it is not design to be end-to-end data integration system but provides a lot of automation in the entity resolution. It has probabilistic direct matching, transitive linking (when entry A matches B, and B matches C - then A also matches C) and assert linking (when entries A and B have different names but are based on prior knowledge of their equivalence this system will link them together), and also allows users to fix conflicts. Hyperion [26] intensively relies on peer-to-peer approach for data integration, it provides an API to which sources must conform to be in the peer-to-peer network, this approach is a different angle of and ontology-based solution, but still can not be applied to wide range of heterogeneous data sources. BigGorilla [27] follows the approach of a module-based system, while the system itself can not provide rich functionality, it aims to be a part of the bigger system. ITBenchmarking [28] relies on ontology-based solution, that is hardly applicable for the domain with high amount of heterogeneous data sources. It provides automatic schema alignment and entity resolution based on ontology. UFeed [29] refines schema mappings and mediated schema based on user actions over query answers.

4 Architecture

In this section the architecture and user workflow in data integration are presented, which aims to address some of the outlined big data integration problems. An implementation of the architecture is developed on Python. It will be implemented in the distributed computation environment to address the volume and velocity of sources and data. The variety of approaches such as MapReduce or even more high-level ones such as Spark [30] will be considered and one of them will be chosen.

The data flow is depicted in Figure 1. As it is shown in Figure 1, the owner of the system initiates the search of the relevant sources. The sources are filtered by machine learning algorithm that is based on various meta attributes of a particular source, which can be performed with various clustering algorithms. Along with the source the owner provides user defined rules or program to extract the data from the source, which will be refined in using extraction rule generator component, which have to be refined because of the velocity problem of the sources. To implement this component tagging [14] is applied and if it does not provide a reliable rule the component notifies the owner of the system in case similarity between database updates is lower than the threshold. This data is stored in extraction support database. The extractor component initiates the extraction procedure on distributed system according to the time schedule, rules and sources list in extraction support database. It stores the extracted information in the database for extracted entries and then record linkage step is performed, which stores the information in database for resolved entries. At record linkage step the volume problem is addressed by implementing blocking and by balancing the pairwise comparison with PairRange [10] in distributed computational jobs, which flattens equally the load, while

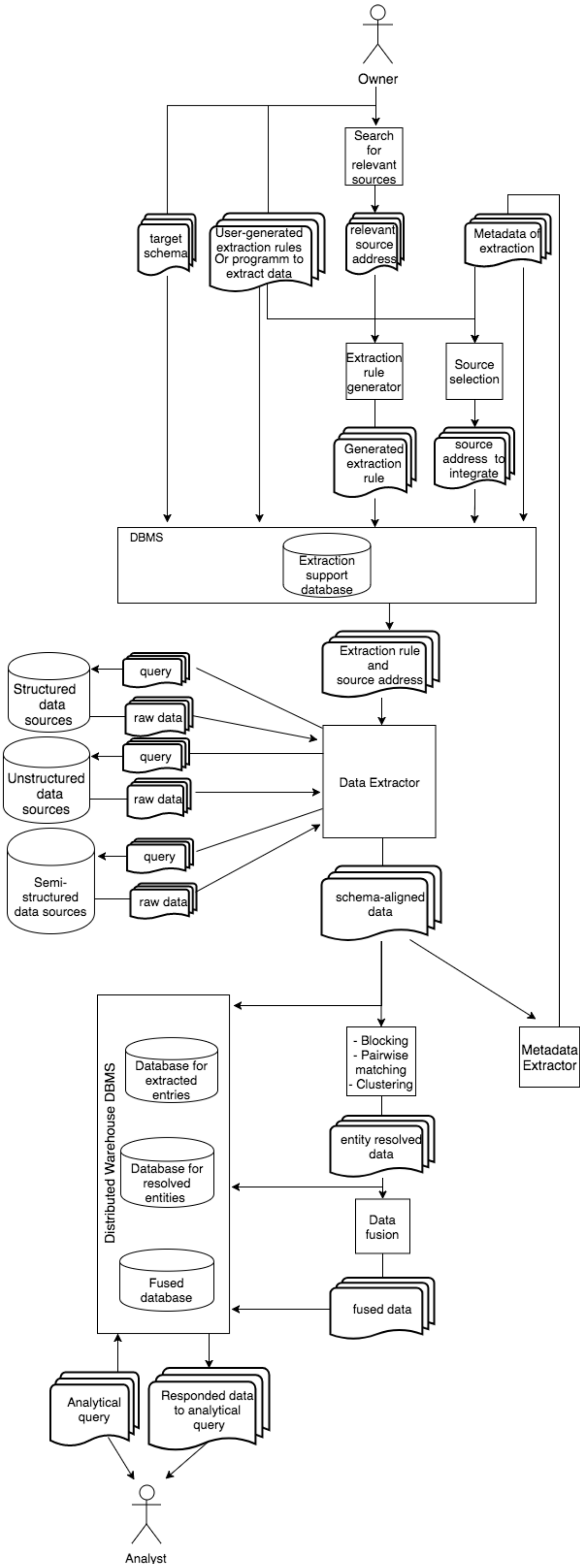
Owner
Search for relevant sources
target schema
User-generated extraction rules Or programm to extract data
relevant source address
Metadata of extraction
Extraction rule generator
Source selection
Generated extraction rule
source address to integrate
DBMS
Extraction support database
Extraction rule and source address
query
Structured data sources
raw data
query
Unstructured data sources
raw data
Data Extractor
query
Semi-structured data sources
raw data
schema-aligned data
Distributed Warehouse DBMS
Database for extracted entries
Database for resolved entities
Fused database
- Blocking
- Pairwise matching
- Clustering
Metadata Extractor
entity resolved data
Data fusion
fused data
Analytical query
Responded data to analytical query
Analyst

Figure 1 Data Flow

the velocity problem is addressed by implementing the Incremental linkage [11]. Finally, the data fusion step is performed on the resolved entries information and the results stored in the fused database. This step implemented by applying k-partite graph to identify the true value of the attribute [31] and user defined functions. On the top of that, an analyst can perform queries on fused database and fetch the response.

5 Conclusions and Future Work

The paper presents an overview of methods and systems for big data integration. A big data architecture is provided, that combines the most promising approaches addressing big data integration problems in a distributed computation environment and covers most of major steps of data integration. The architecture and the big data integration workflow intended to be highly extensible and address various problems are suggested. As an implementation of source selection algorithm, machine learning techniques are chosen to be applied to assess whether the source should be ever integrated. To refine rules, tagging or notifying owner the system is applied in case similarity between database updates is lower than the threshold. For the record linkage step, Blocking with Incremental linkage is applied to address the volume and velocity problem implemented with PairRange to balance the load and solve the volume problem. Also, k-partite graph is applied to identify true value of the attribute and perform user-defined rules.

The initial steps of architecture implementation are performed. At the current state by a given list of the sources the system is capable of extracting information based on rule or program defined by user; perform record linkage without optimisation and fuse the data with user defined and pre-defined functions. As an application area for the implementation e-commerce domain is used.

As a future work it is planned to implement the whole architecture as a stand-alone software solution over a distributed computational environment.

Acknowledgments. This work is supervised by Sergey Stupnikov, Federal Research Center “Computer Science and Control” of Russian Academy of Sciences.

References

[1] Franklin, M.J., Halevy, A.Y., Maier, D.: From databases to dataspaces: a new abstraction for information management. ACM SIGMOD Rec., 34, 27–33 (2005). doi: 10.1145/1107499.1107502. 35

[2] Sarma, A., Dong X., Halevy, A.: Bootstrapping pay-as-you-go data integration systems. Proc. ACM SIGMOD Int. Conf. on Management of Data, 1, 861–874 (2008). doi: 10.1145/1376616.1376702

[3] Dong, X.L., Saha B., Srivastava, D.: Less is more: Selecting sources wisely for integration. Proc. VLDB Endowment, 6, 37-48 (2012). doi: 14778/2535568.2448938

[4] Festa, P.,Resende M.: GRASP: basic components and enhancements. Telecommun. Syst., 46, 253–271 (2011). doi: 10.1007/s11235-010-9289-z

[5] Naumann, F.: Data profiling revisited. ACM SIGMOD Rec., 42, 40–49 (2013). doi: 10.1145/2590989.2590995

[6] Yang, X., Procopiuc, C. M., Srivastava, D.: Summarizing relational databases. Proc. VLDB Endowment, 2, 634–645 (2009). doi: 10.14778/1687627.1687699

[7] Mauricio A. Hernandez, M.A., and Stolfo, S.J.: Real-world data is dirty: Data cleansing and the merge/purge problem. Data Mining and Knowledge Discovery, 2, 9–37 (1998). doi: 10.1023/A:1009761603038

[8] Papadakis, G., Koutrika, G., Palpanas, T.,Nejdl, W.: Meta-blocking: Taking entity resolutionto the next level. IEEE Trans. Knowl. and Data Eng., 26, 1946–1960 (2014). doi: 10.1109/TKDE.2013.54

[9] Ko¨pcke, H., Thor A., Rahm E.: Evaluation of entity resolution approaches on real-world match problems. Proc. VLDB Endowment, 3, 484–493, (2010). doi:10.14778/1920841.1920904

[10] Kolb, L., Thor, A., Rahm E.: Load balancing for mapreduce-based entity resolution. In Proc. 28th Int. Conf. on Data Engineering, 1, 618–629, (2012). doi: 10.1109/ ICDE.2012.22

[11] Gruenheid, A., Dong X.L., Srivastava, D.: Incremental record linkage. Proc. VLDB Endowment, 7, 697–708, (2014). doi: 10.14778/2732939.2732943

[12] Wang, J., Kraska, T., Franklin, M.J., Feng J.: Crowder: Crowdsourcing entity resolution. Proc. VLDB Endowment, 5, 1483–1494 (2012). doi: 10.14778/2350229.2350263

[13] Whang, S.E., Lofgren, P., Garcia-Molina, H.: Question selection for crowd entity resolution. Proc. VLDB Endowment, 6, 349–360 (2013). doi: 0.14778/2536336.2536337

[14] Kannan, A., Givoni I.E., Agrawal R., Fuxman A.: Matching unstructured product offers to structured product specifications. In Proc. 17th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, 1, 404–412, (2011). doi: 10.1145/2020408.2020474

[15] Li P., Dong X.L., Maurino A., Srivastava D.: Linking temporal records. Proc. VLDB Endowment, 4, 956–967, (2011). doi: 10.1007/s11704-012-2002-5

[16] Dong X.L., Berti-Equille L., Srivastava, D.: Integrating conflicting data: The role of source dependence. Proc. VLDB Endowment, 2, 550–561, (2009). doi: 10.1.1.151.4068

[17] Dong, X.L., Berti-Equille, L., Srivastava, D.: Truth discovery and copying detection in a dynamic world. Proc. VLDB Endowment, 2, 562–573, (2009). doi: 10.14778/1687627.1687691

[18] ComparisNew Features & Product Changes for Talend Summer '16. http://info.talend.com/rs/talend/images/TN_EN_TLD_Talend_6_2_Features.pdf

[19] Comparison CloverETL vs. competitors https://www.cloveretl.com/sites/applicationcraft/files/files/case-studies/Comparison_CloverETL_vs_Talend_Pentaho.pdf

[20] Centerprise Data Integrator - Astera Software. http://www.astera.com/media/1391/centerprise-product-brochure.pdf

[21] Attunity Replicate User and Reference Guide https://support.ibt.com.au/helpdesk/File/Get/6215375

[22] Whang, S. E., Marmaros, D., Garcia-Molina, H.: Pay-As-You-Go Entity Resolution. IEEE Transactions on Knowledge and Data Engineering 25, 1111-1124 (2012). doi: 10.1109/TKDE.2012.43

[23] Parra V., Halgamuge M.: Performance Evaluation of Big Data and Business Intelligence Open Source Tools: Pentaho and Jaspersoft. Internet of Things and Big Data Analytics Toward Next-Generation Intelligence, 1, 147-176, (2018). doi: 147-176. 10.1007/978-3-319-60435-0_6

[24] Teiid Designer User Guide http://docs.jboss.org/teiid/designer/10.0/user-guide/en-US/prd/Teiid_Designer_User_Guide.pdf

[25] Nelson, E. D., Talburt, J. R.: Entity resolution for longitudinal studies in education using OYSTER (2011)

[26] Arenas, M., Kantere, V., Kementsietsidis, A., Kiringa, I.,Miller, R. J. , Mylopoulos, J.: The Hyperion project: From data integration to data coordination. Proceeding of the ACM SIGMOD RECORD 32, 53-58 (2003). doi: 10.1145/945721.945733

[27] Golshan, B., Halevy, A., Mihaila, G., Tan, W.: Data Integration: After the Teenage Years. Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems 1, 101-106 (2017). doi: 10.1145/3034786.3056124

[28] Krcmar, H., Pfaff, M.: A web-based system architecture for ontology-based data integration in the domain of IT benchmarking. Enterprise Information Systems 12, 236-258 (2017). doi: 10.1080/17517575.2017.1329552

[29] A., Aboulnaga,. A.: UFeed: Refining Web Data Integration Based on User Feedback. Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 1, 187-196 (2017). doi:10.1145/3132847.3132887

[30] Spark Overview https://spark.apache.org/docs/latest/index.html

[31] Guo, S., Dong, X., Srivastava, D., Zajac, R.: Record linkage with uniqueness constraints and erroneous values. Proc. VLDB Endowment, 3, 417–428 (2010). doi: 10.14778/1920841.1920897