Journal of Theoretical and Applied Information Technology
31st October 2016. Vol.92. No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

XML INTEGRITY CONSTRAINTS, WHAT'S NEXT?

[1]MOHAMMED HAKAWATI, [1]PUTEH SAAD, [1]NASEER SABRI, [1]YASMIN YACOB,
[1]R. B. AHMAD, M. [2]S. SALIM

[1]School of Computer and Communication Engineering
University Malaysia Perlis, Campus Pauh Putra, 02600 Arau, Perlis, Malaysia

[2]Ministry of Higher Education, AlNahrain University, Iraq

E-mail: [1]mshakawati@hotmail.com

ABSTRACT

Without any doubt, XML data model considered the most dominant document type over the web with more than 60% of the total; nevertheless, their quality is not as expected. Data cleaning is equipped to overcome database's quality issues. Integrity Constraint is a very important criterion for keeping data in a consistent manner, almost all previous XML dependencies are introduced to improve the schema and normalization, with a small effort toward improving data instance. This paper summarizes the most important XML integrity constraint notations and data cleaning approaches. In addition, to highlight the shortcoming of these constraints and proved it is limitation for increasing data quality. Finally, introduce the next generation of conditional integrity constraints, which will be held mainly for data cleaning issues.

Keywords: *XML, Data Quality, Data Cleaning, Integrity Constraints.*

1. INTRODUCTION

With the increasing significance of XML as the main data model for data transfer and data integration, data quality becomes a critical issue to make these applications success. Data cleaning, which refers to a set of processes used to improve data quality, has been used extensively in relational databases with less concern in XML [1].

According to studies presented from Data Mentor's blog in 2015, the expense of bad data may be even higher than that 12% lost revenue, 28% of those who have had problems delivering email say that customer service has suffered as a result while 21% experienced reputation damage. Most of the companies, 86%, admitted that their data might be inaccurate in some way. Whereas 44% of businesses said, missing or imperfect data is the most frequent problem with outdated information [2].

Data cleaning is an important process for organizations that seek to extract valuable information from raw data. Many raw datasets typically contain erroneous information such as misspellings, missing values [3].

Cleaning XML databases pose new challenges and problems not faced in cleaning relational databases, The first challenge is the semi-structure tree model which more difficult to handle than relational one, the second challenge is that there are no unified notations for XML integrity constraints, which is, in turn, played an important role in data cleaning approaches [4].

In light of these, there has been increasing demand for data quality tools, for effectively detecting and repairing errors in the XML data. Here, we introduce model approache for handling XML dirty data using enhanced and conditional copy of integrity constraints in order to enhancing data quality.

This work covers the most important aspects of data quality attributes with more concentrate on data consistency. In addition, to covers earlier dependency based cleaning approaches and see that former XML integrity constraints cannot fit for increasing the quality. Furthermore, we argue that classical constraints often need to be revised or extended to capture more errors in real-life data, and to match, repair and query inconsistent data.

This paper organized as follows: Section 2 provides a basic definition of data quality and list

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

most important quality attribute. Section 3 overviews XML integrity constraints (XFD, XIND) and highlight its significant in future dependencies. Section 4 will cover briefly, Data cleaning approaches using traditional dependencies, Section 5 argues about what is next in XML constraints, and finally, Section 6 concludes and outlines future work.

2. DATA QUALITY: AN OVERVIEW

Nowadays, high-quality data or at least non-dirty data is indispensable for all businesses and organizations to run their data-analysis applications efficiently (e.g. customer relationship management, decision support systems, and data mining). Initially, to understand the value of data quality, we need to define the quality of the data and discover common data quality dimensions used and highlight the methods offered to enrich poor data, for a survey see [5].

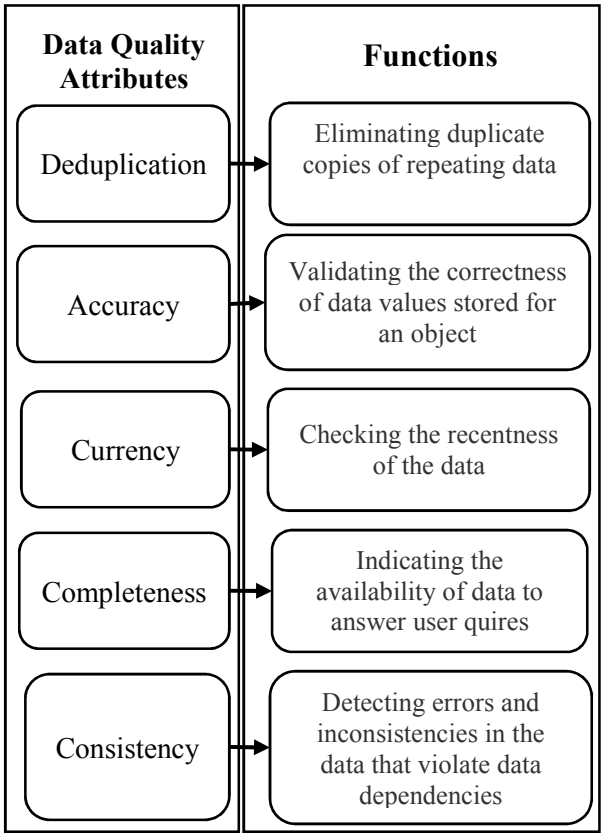
In general, high-quality data is definitely not data that is without errors, incorrect data is also a part of the data quality equation. However, data quality can be defined as "fitness for use", or the ability of data to meet the user's requirement and business attributes. In other words, the problem of data quality is fundamentally intertwined in how our system fits into the practical and with how users use the data in the system. Exactly the right data in exactly the perfect place at the correct time and in the right format to complete a certain procedure, serve a customer, make a decision, or set and execute strategy [6].

The impact of poor data quality can be classified into three types: *operational* (causing customer and employee disappointment and expanded costs), *tactical* (affecting decision making and causing mistrust), and *strategic impacts* (affecting the overall organization's strategy). Overall, any system or enterprise that heavily relies on data is inclined to experience problems if the data being handled does not have the normal quality characteristics [7].

2.1 Data Quality Attributes

In order to improve data quality, a set of attributes or criterions needed, see Fig .1. [8], these attributes are common with different data models: *structured* like relational databases and flat tables or *semi-structured* like XML and UML.

Data Accuracy is to answer the question, how much do our data close to real-world values? In other words, the percentage of objects without data errors such as misspellings, out-of-range values [9]. **Data Completeness** concerns about the question, does our database has a complete information to answer user queries or not [10]. **Data Deduplication** is the problem of identifying tuples from one or more (possibly unreliable) relations or document that refer to the same real-world entity, simply, are there two elements refer to identical real object? [11]. **Data Currency** (timeliness) aims to identify the latest (newest) values of entities represented by possibly outdated database [12]. Finally, **Data Consistency** refers to the validity and integrity of data representing real-world entities. It aims to detect errors (inconsistencies) in the data that recognized as violations of data dependencies (integrity constraints), it is also indispensable for data repair by fixing the errors. There are at least two questions related with data consistency. What data dependencies should we use to detect errors? What repair model do we adopt to fix the errors?[13].



ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

3. XML INTEGRITY CONSTRAINTS, AN OVERVIEW.

Obviously, XML integrity constraints have attracted much interest in recent years, to ensure data consistency, integrity constraints [14] are vital and indispensable. The expression "integrity constraint" in XML used to mean extensions of relational integrity constraints, such as, *functional dependencies, Inclusion Dependencies* and so on, which depend principally on the equality of data values within single or multi-relations.

Many integrity constraints types have been proposed, the hierarchical nature of XML data calls for not just absolute constraints that hold on the entire document, but also relative constraints that hold on sub-documents. However, both absolute and relative integrity constraints for XML documents are defined through XML schemas: DTD or XML Schema.

Integrity constraints can be defined as a set of properties that should be satisfied by every data instance (XML data path) of a database [15]. Data, which violate these constraints considered as a dirty data and requires cleaning. Since there is more than one way of making these changes, methods have been proposed to clean data using criteria such as cost and distance functions [16], The goal of such approaches is to make the cleaned database as close as possible to original one.

3.1 Functional Dependencies.

A considerable amount of literature has been published on XML functional dependency, in this paper; we will revise three main approaches: tree tuple-based, Path-based, and generalized tree tuple.

Arenas and Libkin adopted a tree tuple-based approach (see Fig 2.A), it was the first to define XML FD as an expression of the form: $(S_1 \rightarrow S_2)$, where S_1, S_2 are finite non-empty subsets of paths. In addition, to provide an idea of XML normal form (XNF) to avoid update anomalies and redundant values, it considers tree T confirms FD $(T \vDash (S_1 \rightarrow S_2))$, *iff* two tree tuple t_1 and t_2 on tree, such that $t_1.S_1 = t_2.S_1$ and $t_1.S_1$ is well defined, then $t_1.S_2 = t_2.S_2$.[17]. XNF is a type of schema cleaning, which used for more than 50 years in relational databases, and as a result, it becomes necessary for all data model.

Path-based approach or closest-node (Fig 2.B) is an XML FD of the form: $(\{P_{x1} \ldots P_{xn}\} \rightarrow P_y)$ where P_{xi} called Left-Hand-Side (LHS) and present the paths specifying the condition elements, and P_y, is the Right-Hand-Side (RHD) and it is also a path but specifying the implication elements and the target elements are implicitly specified as a set of elements pointed to be P_y [18].

Both previous approaches concerned about Absolute Keys (keys that hold overall XML document), but none of them cares about Relative Keys, which in turn hold on sub-document. Moreover, neither notion can effectively capture constraints with set elements. The different between the aforementioned approaches is the way of dealing with Target Elements of FD; the former specifies it independent of each FD, while the latter encodes the target element inside the FD.

Similar to a tree tuple [17], generalized tree tuple is a data tree estimated from the original data tree. However, rather than isolating sibling nodes with the same path at all hierarchy levels, this notation has an extra parameter called *pivot node*, and the division is done only at subtrees rooted at the pivot node. Require type of XML FDs deal with Null values as with [18], which required each FD to be Strongly satisfied.

Generalized tree tuple (Fig 2.C) is of the form <Cp, LHS, RHS>, where Cp denotes a tuple class, LHS is a set of paths relative to p, and RHS is a single path relative to p. The satisfaction of XFD over T, $(T \vDash X)$ is for any two GTTs $t_1, t_2 \in Cp$, $\exists p \in LHS$, $t_1.p$ or $t_2.p$ = Null, or if $\forall p \in LHS$, $t_1.p = t_2.p$, then $t_1.RHS$ and $t_2.RHS \neq$ Null, and $t_1.RHS = t_2.RHS$. [19].

The perception is that neither a 'tree tuple', nor a 'closest-node', can express integrity constraints including sets of elements rather than just individual values, and a generalization of a 'tree tuple' XFD was presented to overcome these problems and give more flexibility, however, the main limitation is the path must reach the leaves of XML document tree.

Aforementioned XFD approaches conducted mainly for schema design, and their ability to improve data instance is limited, also these dependencies hold on the entire relation, where sometimes a small part of data become considered dirty.

3.2 Inclusion Dependencies

Inclusion dependencies are the type of constraints, which connect two set of attributes, between two different or even same relation, for instance, R_1 [A,B] $\subseteq$ R_2 [A,B] means that all values of the dependent attribute A, B of R_1, are contained in the value set of the referenced attribute A, B of R_2. XML inclusion dependencies are important in many fields like XML publishing,

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

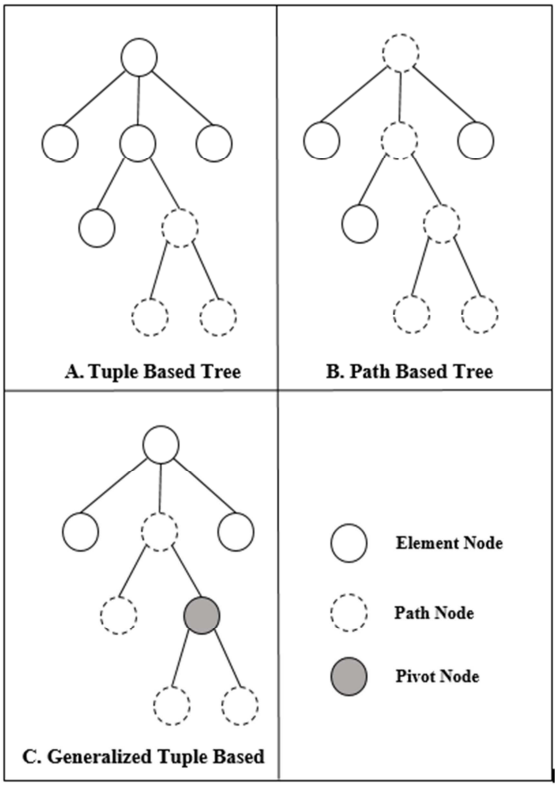


Figure 2: Conceptual Representation for Different Tree Approaches.

where the relational database has to map to a single predefined XML Schema.

Karlinger, Vincent, & Schrefl, presented XIND as a dependency preservation when mapping relation database to XML dataset, they used the same notation offered by their old work for XFDs [21], the most notable different in this notations is not using any XML Schema (DTD or Schema) but using the closest-node .

Their notations for XIND has a form of $((P, [P_1, \dots, P_n]) \subseteq (P', [P_1', \dots, P_n']))$, where P and P' paths are LHS and RHD selectors respectively, and $P_1, \dots, P_n$ and $P_1', \dots, P_n'$ are a non-empty set of paths called LHS, RHS fields respectively. This notation different slightly from the previous notation in that it considers only simple paths for both selectors and filed paths , in addition, to allow attribute/ text node rather than element node types .

Many other notations for XIND presented, we did not mention it for lack of space. However, XIND does not present any effort toward data cleaning and data quality, even though they played an important role in improving schema and instance in the relational database.

4. INTEGRITY-CONSTRAINTS DATA CLEANING

The majority of XML cleaning solutions can be classified into two lines, duplicate detection, and data integration; indeed, they are two important research topics for web technologies. Duplicate or record linkage concerns of finding the same entities refer to same real-world objects, where integration is vital in the new database application to ease merging data documents, from many resources and reconciles between their schema and instance.

Actually, a few number of XML cleaning approaches based on IC presented, for same reasons mentioned early. The first real move toward clean XML tree back to 2003, Flesca and others [22] propose a technique for making minimum changes to turn XML data into consistency situation, and prepare it to answer user's queries. The repair phases described in this approach is even by changing the value of an attribute or the content of an element, or by marking some of the attributes or elements of the document as "unreliable".

Repair XML tree is a triplet (T, δ, ϱ) where (T, δ) is an XML tree and ϱ is a reliability function from set of nodes (N_T) to {true, false}, such that for each pair of nodes n_1, $n_2 \in NT$ with n_2 descendant of n_1, it holds that $(n_1) = \text{false} \Rightarrow (n_2) = \text{false}$. To be able to create a repair, an R-XML tree must not satisfy FD according to definition of *weak satisfiability*.

The problem with the previous approach is that the complexity time was "undecidable"; however, it becomes decidable when particular classes of constraints are considered. The existence of repairs is proved to be decidable and, in particular, NP-complete, if inconsistent data are interpreted as "dirty" data (so that repairs are data cleaning operations consisting in only deletions)[23].

The combination of data integration and data quality for dealing with inconsistencies during merging data from various resources conducted in [24], three probabilistic answers used for resolving a set of inconsistencies violating XML FDs defined in the target schema, by-peer, by sequence, and by-subtree. Mainly this approach concerns about repair inconsistencies for integration issues rather than cleaning in addition to use probabilistic dataset.

Shahriar and Anam [25] consider XML constraints as a quality measurement in XML data and they believe that XML constraints can play an important role for future data quality applications, they employee XFD and XMVD for finding some interesting patterns and association rules in the

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

XML documents then utilize it in data mining not data cleaning .

Improving XML data quality with XFD [26] was the first demonstrated experimentally approaches conducted to highlight the importance of XML integrity constraints in data quality , using the cost-based model to modified violated data, in addition, to providing an efficient two-step heuristic method to repair XFD violations.

The advantage of this approach is the complexity time ; by defining upper and lower bounds for optimum repair , their algorithm is still in NP-complete in the size of the data even though when using fixed DTD and a set of FDs . Also , the notation of functional dependencies invoked was the one introduced by [27], which is the most common notation presented both XFD and Keys in the same manner.

Aforementioned models considered no user interaction needed in cleaning process, as we know that user perspective is important as he is the definer of integrity constraints, [28] incorporate user interactions during cleaning phase, in addition, to introducing the concept of *repair groups*, this improvement overcome the limitations of [22]. The advantage of this approach is making a repair even with a few modifications; also, the user has the ability to change the modified values as it suits him.

The impact of retroactive updates on the consistency of the database is mentioned by [29] they propose an efficient approach that preserves such a consistency, and show how to repair automatically and safely data inconsistencies which result from a transaction that includes some operations acting on past data.

All pervious approaches for cleaning XML hired traditional dependencies, which in turn introduced mainly for schema design not cleaning issues. Moreover, no one of them concerns about dependency preserving during the mapping phase between relational and XML databases.

5. NEXT VISION

Bohannon et al.[30] developed a new extension to traditional dependencies for improving data instance, the idea was the first big step toward new comprehensive data cleaning approaches and hundred papers published to study how CFD improve data quality and it's siblings like data mining and data integration .

XML database is not less important of relational one as mentioned early , this motivation leads [31] to conduct a conditional copy of XFD called XCFD , and changed the notation of XFD to convey the new rules .

XCFD notation is an expression of the form: $\psi = pv : \{e\}, \{X\} \rightarrow \{Y\}$, where pv : is the context path, $\{e\}$ is the conditional part, and $\{X\} \rightarrow \{Y\}$ is a standard XFD. The conformation between tree and XCFD $T \models \psi$ achieved if two paths agree Conditionally on their LHS then their RHS should matches as well, this type of functional dependency is important for data cleaning issues rather than schema design issues and holds on a subset rather than entire document.

Even though XFD played a tiny role in instance cleaning, XIND had nothing to remember. Bohannon and others[32] recommended that not only *functional dependencies* but also *inclusion dependencies* required for data cleaning as well as schema design. This motivates database theory scientist to introduce a massive number of approaches for improving relational database using both types' of dependencies.

At the same line, XML used at early stages as view for relational tables and currently for data transfer, so dependency preserving for XML becomes indispensable, and cleaning approaches with complete components become urgent.

The need for building an XML data cleaning approach using both type of enhanced dependencies is the main objective behind this work.

6. CONCLUSION

XML data quality becomes premonition for web application developer and database creators. Increasing data consistency depends mainly on integrity constraints, which in turn becomes too old to help us achieve the required quality peak. In this work, we revised the short comes of old fashion integrity constraints and the application rely on it. Future XML cleaning application should cover all quality attributes with more focus on data consistency.

To achieve our objective of improving the quality of XML dataset and discover more inconsistencies and faults, we propose comprehensive XML cleaning approach using multi-integrity constraints. The proposed approach has the ability to discover the inconsistences in multi-level XML tree and clean it when required, in addition to keep eyes on the complexity and validity of conducted algorithm in order to produce correct results in a manageable time.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

REFERENCES:

[1] S. Grijzenhout and M. Marx, "The quality of the XML Web," *J. Web Semant.*, vol. 19, pp. 59–68, 2013.

[2] Larisa Bedgood, "How Much is Dirty Data Costing You? | DataMentors," 2015. [Online]. Available: http://www.datamentors.com/blog/how-much-dirty-data-costing-you. [Accessed: 16-Jan-2016].

[3] J. Den Broeck and L. T. Fadnes, "Data Cleaning," *Epidemiol. Princ. Pract. Guidel.*, pp. 389–399, 2013.

[4] Z. Tan and L. Zhang, "Repairing XML functional dependency violations," *Inf. Sci. (Ny).*, vol. 181, no. 23, pp. 5304–5320, 2011.

[5] M. Ge, M. Helfert, D. Mcgilvray, and S. Sadiq, *Handbook of Data Quality*. 2013.

[6] M. Chen, M. Song, J. Han, and E. Haihong, "Survey on data quality," *Proc. 2012 World Congr. Inf. Commun. Technol. WICT 2012*, pp. 1009–1013, 2012.

[7] N. Laranjeiro, S. N. Soydemir, and J. Bernardino, "A Survey on Data Quality: Classifying Poor Data," in *2015 IEEE 21st Pacific Rim International Symposium on Dependable Computing (PRDC)*, 2015, vol. 10, no. January, pp. 179–188.

[8] W. Fan and F. Geerts, "Foundations of Data Quality Management," *Synth. Lect. Data Manag.*, vol. 4, no. 5, pp. 1–217, 2012.

[9] Y. Cao, W. Fan, and W. Yu, "Determining the relative accuracy of attributes," *Proc. 2013 Int. Conf. Manag. data - SIGMOD '13*, p. 565, 2013.

[10] S. Razniewski and W. Nutt, "Completeness of queries over incomplete databases," *Proc. VLDB Endow*, vol. 4, no. 11, pp. 749–760, 2011.

[11] N. A. Pande and N. D. Ghuse, "Record Deduplication Approaches and Algorithm for Removing Duplicate Data," vol. 4, no. 3, 2015.

[12] H. Zhangn, Y. Diao, and N. Immerman, "Recognizing patterns in streams with imprecise timestamps," *Inf. Syst.*, vol. 38, no. 8, pp. 1187–1211, 2013.

[13] W. Fan, "Edinburgh Research Explorer Data Quality : Theory and Practice Data Quality : Theory and Practice," vol. 7418, 2012.

[14] L. Caruccio, V. Deufemia, and G. Polese, "Relaxed Functional Dependencies - A Survey of Approaches," *IEEE Trans. Knowl. Data Eng.*, vol. 0, no. 0, pp. 1–1, 2016.

[15] R. Elmasri, *Fundamentals of database systems*. Pearson Education India, 2008.

[16] W. Fan, F. Geerts, and X. Jia, "A revival of integrity constraints for data cleaning," *Proc. VLDB Endow.*, vol. 1, no. 2, pp. 1522–1523, 2008.

[17] M. Arenas and L. Libkin, "A normal form for XML documents," *ACM Trans. Database Syst.*, vol. 29, no. 1, pp. 195–232, 2004.

[18] M. W. Vincent, J. Liu, and C. Liu, "Strong functional dependencies and their application to normal forms in XML," *ACM Trans. Database Syst.*, vol. 29, no. 3, pp. 445–462, 2004.

[19] C. Yu and H. V Jagadish, "Efficient discovery of XML data redundancies," in *Proceedings of the 32nd international conference on Very large data bases*, 2006, pp. 103–114.

[20] M. Karlinger, M. Vincent, and M. Schrefl, "Inclusion Dependencies in XML: Extending Relational Semantics," in *Proceedings of the 20th International Conference on Database and Expert Systems Applications*, 2009, pp. 23–37.

[21] M. W. Vincent, J. Liu, and M. Mohania, "On the equivalence between FDs in XML and FDs in relations," *Acta Inform.*, vol. 44, no. 3–4, pp. 207–247, 2007.

[22] S. Flesca, F. Furfaro, S. Greco, and E. Zumpano, "Repairs and consistent answers for XML data with functional dependencies," in *Database and XML Technologies*, Springer, 2003, pp. 238–253.

[23] S. Flesca, F. Furfaro, S. Greco, and E. Zumpano, "Querying and repairing inconsistent XML data," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3806 LNCS, pp. 175–188, 2005.

[24] T. Pankowski, "Reconciling inconsistent data in probabilistic XML data integration," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5071 LNCS, no. 1, pp. 75–86, 2008.

[25] M. S. Shahriar and S. Anam, "Towards Data Quality and Data Mining Using Constraints in XML," *Int. J. Database Theory Appl.*, vol. 2, no. 1, pp. 23–30, 2009.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[26] Z. Tan and L. Zhang, "Improving XML Data Quality with Functional Dependencies," no. 60603043, pp. 450–465, 2011.

[27] W. Fan and J. Simeon, "Integrity constraints for XML," *J. Comput. Syst. Sci.*, vol. 66, no. 4082003, pp. 254–291, 2003.

[28] M. Švirec and I. Mlýnková, "Efficient Detection of XML Integrity Constraints Violation," in *Communications in Computer and Information Science*, vol. 293 PART 1, 2012, pp. 259–273.

[29] H. Hamrouni, Z. Brahmia, and R. Bouaziz, "An Efficient Approach for Detecting and Repairing Data Inconsistencies Resulting from Retroactive Updates in Multi-Temporal and Multi-version XML Databases," in *New Trends in Database and Information Systems II*, vol. 312, Cham: Springer, 2015, pp. 135–146.

[30] P. Bohannon, W. Fan, F. Geerts, X. Jia, and A. Kementsietsidis, "Conditional functional dependencies for data cleaning," *Proc. - Int. Conf. Data Eng.*, pp. 746–755, 2007.

[31] L. Vo, J. Cao, and W. Rahayu, "Discovering conditional functional dependencies in XML data," in *Proceedings of the Twenty-Second Australasian Database Conference-Volume 115*, 2011, vol. 115, no. 5, pp. 143–152.

[32] P. Bohannon, W. Fan, M. Flaster, and R. Rastogi, "A cost-based model and effective heuristic for repairing constraints by value modification," *Proc. 2005 ACM SIGMOD Int. Conf. Manag. data - SIGMOD '05*, p. 143, 2005.