Richi Nayak
Mohammed J. Zaki (Eds.)

LNCS 3915

Knowledge Discovery from XML Documents

First International Workshop, KDXD 2006
Singapore, April 2006
Proceedings

Springer

Richi Nayak Mohammed J. Zaki (Eds.)

Knowledge Discovery from XML Documents

First International Workshop, KDXD 2006
Singapore, April 9, 2006
Proceedings

Springer

Volume Editors

Richi Nayak
Queensland University of Technology
Faculty of Information Technology, GP, School of Information Systems
GPO Box 2434, Brisbane, QLD 4001, Australia
E-mail: r.nayak@qut.edu.au

Mohammed J. Zaki
Rensselaer Polytechnic Institute, Computer Science Department
Troy, NY 12180-3590, USA
E-mail: zaki@cs.rpi.edu

Library of Congress Control Number: 2006922467

CR Subject Classification (1998): H.2.8, I.2, H.3, F.1

LNCS Sublibrary: SL 3 – Information Systems and Application, incl. Internet/Web and HCI

ISSN 0302-9743
ISBN-10 3-540-33180-8 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-33180-3 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media

springer.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11730262 06/3142 5 4 3 2 1 0

Lecture Notes in Computer Science 3915

Preface

The KDXD 2006 (Knowledge Discovery from XML Documents) workshop is the first international workshop running this year in conjunction with the 10th Pacific-Asia Conference on Knowledge Discovery and Data Mining, PAKDD 2006. The workshop provided an important forum for the dissemination and exchange of new ideas and research related to XML data discovery and retrieval.

The eXtensible Markup Language (XML) has become a standard language for data representation and exchange. With the continuous growth in XML data sources, the ability to manage collections of XML documents and discover knowledge from them for decision support becomes increasingly important. Due to the inherent flexibility of XML, in both structure and semantics, inferring important knowledge from XML data is faced with new challenges as well as benefits. The objective of the workshop was to bring together researchers and practitioners to discuss all aspects of the emerging XML data management challenges. Thus, the topics of interest included, but were not limited to: XML data mining methods; XML data mining applications; XML data management emerging issues and challenges; XML in improving knowledge discovery process; and Benchmarks and mining performance using XML databases.

The workshop received 26 submissions. We would like to thank all those who submitted their work to the workshop under relatively pressuring time deadlines. We selected ten high-quality full papers for discussion and presentation in the workshop and for inclusion in the proceedings after being peer-reviewed by at least three members of the Program Committee. Accepted papers were grouped in three sessions and allocated equal presentation time slots. The first session was on XML data mining methods of classification, clustering and association. The second session focused on the XML data reasoning and querying methods and query optimization. The last session was on XML data applications of transportation and security. The workshop also included two invited talks from leading researchers in this area. We would sincerely like to thank Tok wang Ling and Stephane Bressan for presenting valuable talks in the workshop program.

Special thanks go to the Program Committee members, who shared their expertise and time to make KDXD 2006 a success. The final quality of selected papers reflects their efforts.

Finally, we would like to thank Queensland University of Technology for providing us with the resources and time and the Indian Institute of Technology, Roorkee India for providing us with the resources to undertake this task. Last but least, we would like to thank the organizers of PAKDD 2006 for hosting KDXD 2006. We trust that you will enjoy the papers in this volume.

January 2006

Richi Nayak
Mohammad Zaki

Organization

KDXD 2006 was organized by the School of Information System, Queensland University of Technology, Brisbane, Australia, in cooperation with PAKDD 2006.

Workshop Chairs

Richi Nayak	Queensland University of Technology, Australia
Mohammad Zaki	Rensselaer Polytechnic Institute, USA

Program Committee

Hiroki Arimura (Japan)
Giovanna Guerrini (Italy)
Jung-Won Lee (Korea)
Xue Li (Australia)
Yuefeng Li (Australia)
Chengfei Liu (Australia)
Marco Mesiti(Italy)
Ankush Mittal (India)
Shi Nansi (Australia)
Siegfried Nijssen (Netherlands)
Maria Orlowska (Australia)
Seung-Soo Park (Korea)
Wenny Rahayu (Australia)
Michael Schrefl (Austria)
David Tanier (Australia)
Takeaki Uno (Japan)
Yue Xu (Australia)

Lecture Notes in Computer Science

For information about Vols. 1–3817

please contact your bookseller or Springer

Vol. 3925: A. Valmari (Ed.), Model Checking Software. X, 307 pages. 2006.

Vol. 3924: P. Sestoft (Ed.), Programming Languages and Systems. XII, 343 pages. 2006.

Vol. 3923: A. Mycroft, A. Zeller (Eds.), Compiler Construction. XV, 277 pages. 2006.

Vol. 3922: L. Baresi, R. Heckel (Eds.), Fundamental Approaches to Software Engineering. XIII, 427 pages. 2006.

Vol. 3921: L. Aceto, A. Ingólfsdóttir (Eds.), Foundations of Software Science and Computational Structures. XV, 447 pages. 2006.

Vol. 3920: H. Hermanns, J. Palsberg (Eds.), Tools and Algorithms for the Construction and Analysis of Systems. XIV, 506 pages. 2006.

Vol. 3916: J. Li, Q. Yang, A.-H. Tan (Eds.), Data Mining for Biomedical Applications. VIII, 155 pages. 2006. (Sublibrary LNBI).

Vol. 3915: R. Nayak, M.J. Zaki (Eds.), Knowledge Discovery from XML Documents. VIII, 105 pages. 2006.

Vol. 3905: P. Collet, M. Tomassini, M. Ebner, S. Gustafson, A. Ekárt (Eds.), Genetic Programming. XI, 361 pages. 2006.

Vol. 3904: M. Baldoni, U. Endriss, A. Omicini, P. Torroni (Eds.), Declarative Agent Languages and Technologies III. XII, 245 pages. 2006. (Sublibrary LNAI).

Vol. 3903: K. Chen, R. Deng, X. Lai, J. Zhou (Eds.), Information Security Practice and Experience. XIV, 392 pages. 2006.

Vol. 3901: P.M. Hill (Ed.), Logic Based Program Synthesis and Transformation. X, 179 pages. 2006.

Vol. 3899: S. Frintrop, VOCUS: A Visual Attention System for Object Detection and Goal-Directed Search. XIV, 216 pages. 2006. (Sublibrary LNAI).

Vol. 3896: Y. Ioannidis, M.H. Scholl, J.W. Schmidt, F. Matthes, M. Hatzopoulos, K. Boehm, A. Kemper, T. Grust, C. Boehm (Eds.), Advances in Database Technology - EDBT 2006. XIV, 1208 pages. 2006.

Vol. 3895: O. Goldreich, A.L. Rosenberg, A.L. Selman (Eds.), Theoretical Computer Science. XII, 399 pages. 2006.

Vol. 3894: W. Grass, B. Sick, K. Waldschmidt (Eds.), Architecture of Computing Systems - ARCS 2006. XII, 496 pages. 2006.

Vol. 3890: S.G. Thompson, R. Ghanea-Hercock (Eds.), Defence Applications of Multi-Agent Systems. XII, 141 pages. 2006. (Sublibrary LNAI).

Vol. 3889: J. Rosca, D. Erdogmus, J.C. Príncipe, S. Haykin (Eds.), Independent Component Analysis and Blind Signal Separation. XXI, 980 pages. 2006.

Vol. 3888: D. Draheim, G. Weber (Eds.), Trends in Enterprise Application Architecture. IX, 145 pages. 2006.

Vol. 3887: J.R. Correa, A. Hevia, M. Kiwi (Eds.), LATIN 2006: Theoretical Informatics. XVI, 814 pages. 2006.

Vol. 3886: E.G. Bremer, J. Hakenberg, E.-H.(S.) Han, D. Berrar, W. Dubitzky (Eds.), Knowledge Discovery in Life Science Literature. XIV, 147 pages. 2006. (Sublibrary LNBI).

Vol. 3885: V. Torra, Y. Narukawa, A. Valls, J. Domingo-Ferrer (Eds.), Modeling Decisions for Artificial Intelligence. XII, 374 pages. 2006. (Sublibrary LNAI).

Vol. 3884: B. Durand, W. Thomas (Eds.), STACS 2006. XIV, 714 pages. 2006.

Vol. 3881: S. Gibet, N. Courty, J.-F. Kamp (Eds.), Gesture in Human-Computer Interaction and Simulation. XIII, 344 pages. 2006. (Sublibrary LNAI).

Vol. 3880: A. Rashid, M. Aksit (Eds.), Transactions on Aspect-Oriented Software Development I. IX, 335 pages. 2006.

Vol. 3879: T. Erlebach, G. Persinao (Eds.), Approximation and Online Algorithms. X, 349 pages. 2006.

Vol. 3878: A. Gelbukh (Ed.), Computational Linguistics and Intelligent Text Processing. XVII, 589 pages. 2006.

Vol. 3877: M. Detyniecki, J.M. Jose, A. Nürnberger, C. J. '. van Rijsbergen (Eds.), Adaptive Multimedia Retrieval: User, Context, and Feedback. XI, 279 pages. 2006.

Vol. 3876: S. Halevi, T. Rabin (Eds.), Theory of Cryptography. XI, 617 pages. 2006.

Vol. 3875: S. Ur, E. Bin, Y. Wolfsthal (Eds.), Haifa Verification Conference. X, 265 pages. 2006.

Vol. 3874: R. Missaoui, J. Schmidt (Eds.), Formal Concept Analysis. X, 309 pages. 2006. (Sublibrary LNAI).

Vol. 3873: L. Maicher, J. Park (Eds.), Charting the Topic Maps Research and Applications Landscape. VIII, 281 pages. 2006. (Sublibrary LNAI).

Vol. 3872: H. Bunke, A. L. Spitz (Eds.), Document Analysis Systems VII. XIII, 630 pages. 2006.

Vol. 3870: S. Spaccapietra, P. Atzeni, W.W. Chu, T. Catarci, K.P. Sycara (Eds.), Journal on Data Semantics V. XIII, 237 pages. 2006.

Vol. 3869: S. Renals, S. Bengio (Eds.), Machine Learning for Multimodal Interaction. XIII, 490 pages. 2006.

Vol. 3868: K. Römer, H. Karl, F. Mattern (Eds.), Wireless Sensor Networks. XI, 342 pages. 2006.

Vol. 3866: T. Dimitrakos, F. Martinelli, P.Y.A. Ryan, S. Schneider (Eds.), Formal Aspects in Security and Trust. X, 259 pages. 2006.

Vol. 3865: W. Shen, K.-M. Chao, Z. Lin, J.-P.A. Barthès (Eds.), Computer Supported Cooperative Work in Design II. XII, 359 pages. 2006.

Vol. 3863: M. Kohlhase (Ed.), Mathematical Knowledge Management. XI, 405 pages. 2006. (Sublibrary LNAI).

Vol. 3862: R.H. Bordini, M. Dastani, J. Dix, A.E.F. Seghrouchni (Eds.), Programming Multi-Agent Systems. XIV, 267 pages. 2006. (Sublibrary LNAI).

Vol. 3861: J. Dix, S.J. Hegner (Eds.), Foundations of Information and Knowledge Systems. X, 331 pages. 2006.

Vol. 3860: D. Pointcheval (Ed.), Topics in Cryptology – CT-RSA 2006. XI, 365 pages. 2006.

Vol. 3858: A. Valdes, D. Zamboni (Eds.), Recent Advances in Intrusion Detection. X, 351 pages. 2006.

Vol. 3857: M.P.C. Fossorier, H. Imai, S. Lin, A. Poli (Eds.), Applied Algebra, Algebraic Algorithms and Error-Correcting Codes. XI, 350 pages. 2006.

Vol. 3855: E. A. Emerson, K.S. Namjoshi (Eds.), Verification, Model Checking, and Abstract Interpretation. XI, 443 pages. 2005.

Vol. 3854: I. Stavrakakis, M. Smirnov (Eds.), Autonomic Communication. XIII, 303 pages. 2006.

Vol. 3853: A.J. Ijspeert, T. Masuzawa, S. Kusumoto (Eds.), Biologically Inspired Approaches to Advanced Information Technology. XIV, 388 pages. 2006.

Vol. 3852: P.J. Narayanan, S.K. Nayar, H.-Y. Shum (Eds.), Computer Vision – ACCV 2006, Part II. XXXI, 977 pages. 2006.

Vol. 3851: P.J. Narayanan, S.K. Nayar, H.-Y. Shum (Eds.), Computer Vision – ACCV 2006, Part I. XXXI, 973 pages. 2006.

Vol. 3850: R. Freund, G. Păun, G. Rozenberg, A. Salomaa (Eds.), Membrane Computing. IX, 371 pages. 2006.

Vol. 3849: I. Bloch, A. Petrosino, A.G.B. Tettamanzi (Eds.), Fuzzy Logic and Applications. XIV, 438 pages. 2006. (Sublibrary LNAI).

Vol. 3848: J.-F. Boulicaut, L. De Raedt, H. Mannila (Eds.), Constraint-Based Mining and Inductive Databases. X, 401 pages. 2006. (Sublibrary LNAI).

Vol. 3847: K.P. Jantke, A. Lunzer, N. Spyratos, Y. Tanaka (Eds.), Federation over the Web. X, 215 pages. 2006. (Sublibrary LNAI).

Vol. 3846: H. J. van den Herik, Y. Björnsson, N.S. Netanyahu (Eds.), Computers and Games. XIV, 333 pages. 2006.

Vol. 3845: J. Farré, I. Litovsky, S. Schmitz (Eds.), Implementation and Application of Automata. XIII, 360 pages. 2006.

Vol. 3844: J.-M. Bruel (Ed.), Satellite Events at the MoDELS 2005 Conference. XIII, 360 pages. 2006.

Vol. 3843: P. Healy, N.S. Nikolov (Eds.), Graph Drawing. XVII, 536 pages. 2006.

Vol. 3842: H.T. Shen, J. Li, M. Li, J. Ni, W. Wang (Eds.), Advanced Web and Network Technologies, and Applications. XXVII, 1057 pages. 2006.

Vol. 3841: X. Zhou, J. Li, H.T. Shen, M. Kitsuregawa, Y. Zhang (Eds.), Frontiers of WWW Research and Development - APWeb 2006. XXIV, 1223 pages. 2006.

Vol. 3840: M. Li, B. Boehm, L.J. Osterweil (Eds.), Unifying the Software Process Spectrum. XVI, 522 pages. 2006.

Vol. 3839: J.-C. Filliâtre, C. Paulin-Mohring, B. Werner (Eds.), Types for Proofs and Programs. VIII, 275 pages. 2006.

Vol. 3838: A. Middeldorp, V. van Oostrom, F. van Raamsdonk, R. de Vrijer (Eds.), Processes, Terms and Cycles: Steps on the Road to Infinity. XVIII, 639 pages. 2005.

Vol. 3837: K. Cho, P. Jacquet (Eds.), Technologies for Advanced Heterogeneous Networks. IX, 307 pages. 2005.

Vol. 3836: J.-M. Pierson (Ed.), Data Management in Grids. X, 143 pages. 2006.

Vol. 3835: G. Sutcliffe, A. Voronkov (Eds.), Logic for Programming, Artificial Intelligence, and Reasoning. XIV, 744 pages. 2005. (Sublibrary LNAI).

Vol. 3834: D.G. Feitelson, E. Frachtenberg, L. Rudolph, U. Schwiegelshohn (Eds.), Job Scheduling Strategies for Parallel Processing. VIII, 283 pages. 2005.

Vol. 3833: K.-J. Li, C. Vangenot (Eds.), Web and Wireless Geographical Information Systems. XI, 309 pages. 2005.

Vol. 3832: D. Zhang, A.K. Jain (Eds.), Advances in Biometrics. XX, 796 pages. 2005.

Vol. 3831: J. Wiedermann, G. Tel, J. Pokorný, M. Bieliková, J. Štuller (Eds.), SOFSEM 2006: Theory and Practice of Computer Science. XV, 576 pages. 2006.

Vol. 3830: D. Weyns, H. V.D. Parunak, F. Michel (Eds.), Environments for Multi-Agent Systems II. VIII, 291 pages. 2006. (Sublibrary LNAI).

Vol. 3829: P. Pettersson, W. Yi (Eds.), Formal Modeling and Analysis of Timed Systems. IX, 305 pages. 2005.

Vol. 3828: X. Deng, Y. Ye (Eds.), Internet and Network Economics. XVII, 1106 pages. 2005.

Vol. 3827: X. Deng, D.-Z. Du (Eds.), Algorithms and Computation. XX, 1190 pages. 2005.

Vol. 3826: B. Benatallah, F. Casati, P. Traverso (Eds.), Service-Oriented Computing - ICSOC 2005. XVIII, 597 pages. 2005.

Vol. 3824: L.T. Yang, M. Amamiya, Z. Liu, M. Guo, F.J. Rammig (Eds.), Embedded and Ubiquitous Computing – EUC 2005. XXIII, 1204 pages. 2005.

Vol. 3823: T. Enokido, L. Yan, B. Xiao, D. Kim, Y. Dai, L.T. Yang (Eds.), Embedded and Ubiquitous Computing – EUC 2005 Workshops. XXXII, 1317 pages. 2005.

Vol. 3822: D. Feng, D. Lin, M. Yung (Eds.), Information Security and Cryptology. XII, 420 pages. 2005.

Vol. 3821: R. Ramanujam, S. Sen (Eds.), FSTTCS 2005: Foundations of Software Technology and Theoretical Computer Science. XIV, 566 pages. 2005.

Vol. 3820: L.T. Yang, X.-s. Zhou, W. Zhao, Z. Wu, Y. Zhu, M. Lin (Eds.), Embedded Software and Systems. XXVIII, 779 pages. 2005.

Vol. 3819: P. Van Hentenryck (Ed.), Practical Aspects of Declarative Languages. X, 231 pages. 2005.

Vol. 3818: S. Grumbach, L. Sui, V. Vianu (Eds.), Advances in Computer Science – ASIAN 2005. XIII, 294 pages. 2005.

Table of Contents

Keynote Papers

XML Data Mining Methods

XML Data Reasoning and Querying Methods

Opportunities for XML Data Mining in Modern Applications, or XML Data Mining: Where Is the Ore?

Stephane Bressan, Anthony Tung, and Yang Rui

Department of Computer Science, School of Computing,
National University of Singapore
steph@nus.edu.sg

Abstract. We attempt to identify the opportunities for XML data mining in modern applications. We will try and match requirements of modern application managing XML data with the capabilities of the existing XML mining tools and techniques.

R. Nayak and M.J. Zaki (Eds.): KDXD 2006, LNCS 3915, p. 1, 2006.

Capturing Semantics in XML Documents

Tok Wang Ling

Department of Computer Science, School of Computing,
National University of Singapore
lingtw@comp.nus.edu.sg

Abstract. Traditional semantic data models, such as the Entity Relationship (ER) data model, are used to represent real world semantics that are crucial for the effective management of structured data. The semantics that can be expressed in the ER data model include the representation of entity types together with their identifiers and attributes, n-ary relationship types together with their participating entity types and attributes, and functional dependencies among the participating entity types of relationship types and their attributes, etc.

Today, semistructured data has become more prevalent on the Web, and XML has become the de facto standard for semi-structured data. A DTD and an XML Schema of an XML document only reflect the hierarchical structure of the semistructured data stored in the XML document. The hierarchical structures of XML documents are captured by the relationships between an element and its attributes, and between an element and its subelements. Element-attribute relationships do not have clear semantics, and the relationships between elements and their subelements are binary. The semantics of n-ary relationships with n > 2 cannot be represented or captured correctly and precisely in DTD and XML Schema. Many of the crucial semantics captured by the ER model for structured data are not captured by either DTD or XML Schema. We present the problems encountered in order to correctly and efficiently store, query, and transform (view) XML documents without knowing these important semantics. We solve these problems by using a semantic-rich data model called the ***O***bject, ***R***elationship, ***A***ttribute data model for ***S***emi***S***tructured Data (ORA-SS). We briefly describe how to mine such important semantics from given XML documents.

R. Nayak and M.J. Zaki (Eds.): KDXD 2006, LNCS 3915, p. 2, 2006.

Mining Changes from Versions of Dynamic XML Documents

Laura Irina Rusu[1], Wenny Rahayu[2], and David Taniar[3]

[1,2] LaTrobe University, Department of Computer Science & Computer Eng, Australia
lirusu@students.latrobe.edu.au
wenny@cs.latrobe.edu.au
[3] Monash University, School of Business Systems, Clayton, VIC 3800, Australia
David.Taniar@infotech.monash.edu.au

Abstract. The ability to store information contained in XML documents for future reference becomes a very important issue these days, as the number of applications which use and exchange data in XML format is growing continuously. Moreover, the contents of XML documents are dynamic and they change across time, so researchers are looking to efficient solutions to store the documents' versions and eventually extract interesting information out of them. This paper proposes a novel approach for mining association rules from changes between versions of dynamic XML documents, in a simple manner, by using the information contained in the consolidated delta. We argue that by applying our proposed algorithm, important information about the behaviour of the changed XML document in time could be extracted and then used to make predictions about its future performance.

1 Introduction

The increasing interest from various applications in storing and manipulating their data in XML format has determined, during the last few years, a growing amount of research work, in order to find the most effective and usable solutions in this respect. One main focus area was XML warehousing [9, 10], but a large volume of work have been also concentrating on the issue of mining XML documents [7, 8, 11]. The later one evolved in a quite sensitive issue, because the users became interested not only in storing the XML documents in a very efficient way and accessing them at any point in time, but also in getting the most of the interesting information behind the data.

In addressing first part of the problem, i.e. XML warehousing, we have identified at least two types of documents which could be included in a presumptive XML data warehouse: *static XML documents*, which do not change their contents and structures in time (e.g. an XML document containing the papers published in a proceedings book) and *dynamic XML documents*, which change their structures or contents based on certain business processes (e.g. the content of an on-line store might change hourly, daily or weekly, depending on the customer behavior). While the first category of XML documents was the subject of intense research during the recent years, with various methods for storing and mining them being developed, there is still work to be done in finding efficient ways to store and mine dynamic XML documents [1].

R. Nayak and M.J. Zaki (Eds.): KDXD 2006, LNCS 3915, pp. 3 – 12, 2006.

The work in this paper continues the proposal made in [1], visually grouped in the general framework presented in Figure 1. In this framework, we focused on both warehousing and mining dynamic XML documents, in three main steps, i.e. (*i*) storing multiple versions of dynamic XML documents (Fig. 1A), (*ii*) extracting historic changes for a certain period of time (Fig.1B) and (*iii*) mining the extracted changes (Fig.1C) to obtain interesting information (i.e. association rules) from them.

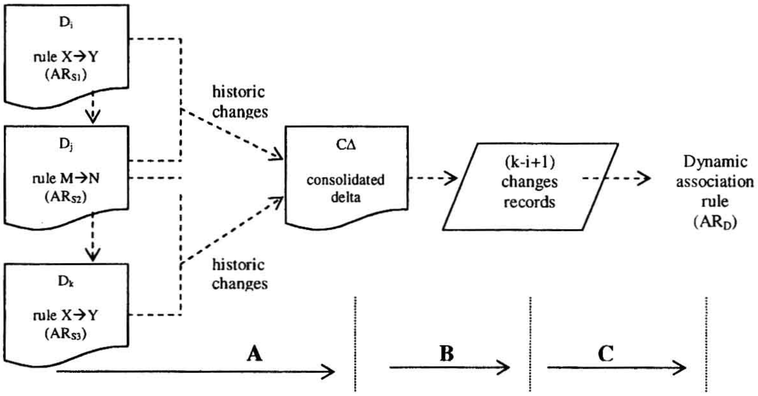


Fig. 1. A visual representation of the mining historic changes process, using consolidated delta

In this paper, we are focusing on the part C of the above mentioned framework, i.e. extracting association rules from changes affecting dynamic XML documents. We believe this knowledge would be very useful in determining if there are any relationships between changes affecting different parts of the documents and making predictions about the future behaviour of the document.

2 Related work

To our knowledge, there is no much work done in the area of mining changes between versions of dynamic XML documents. The existing work is more focused on determining interesting knowledge (e.g. frequently changing structures, discovering association rules or pattern-based dynamic structures) from the multiple versions of the document themselves, not from the actual changes happened in the specified interval of time. We detail below some of this work, noting in the same time that the list of examples is nor complete or exhaustive.

In [2], the authors focus on extracting the FCSs (Frequently Changing Structures). They propose an H-DOM model to represent and store the XML structural data, where the history of structural data is preserved and compressed. Based on the H-DOM model, they present two algorithms to discover the FCSs.

X-Diff algorithm is proposed in [3] and it deals with unordered trees, defined as trees where only the ancestor relationship is important, but not the order of the siblings. This approach is considered to be better and more efficient for the purpose of database applications of the XML. In [3], changes in a XML document over the time are determined by calculating the *minimum-cost edit script*, which is a specific sequence of operations which can transform the XML tree from the initial to the final phase, with the lowest possible cost. In introduces the notion of *node signature* and a new type of matching between two trees, corresponding to the versions of a document, utilized to find the minimum cost matching and cost edit script, able to transform one tree into another.

Another algorithm, proposed by [4], deals with the unordered tree as well, but it goes further and does not distinguish between elements and attributes, both of them being mapped to a set of labeled nodes.

In [5], the authors focus on discovering the pattern-based dynamic structures from versions of unordered XML documents. They present the definitions of *dynamic metrics* and *pattern-based dynamic structure mining* from versions of XML documents. They focus especially on two types of pattern-based dynamic structures, i.e. *increasing dynamic structure* and *decreasing dynamic structure*, which are defined with respect to dynamic metrics and used to build the pattern-based dynamic structures mining algorithm.

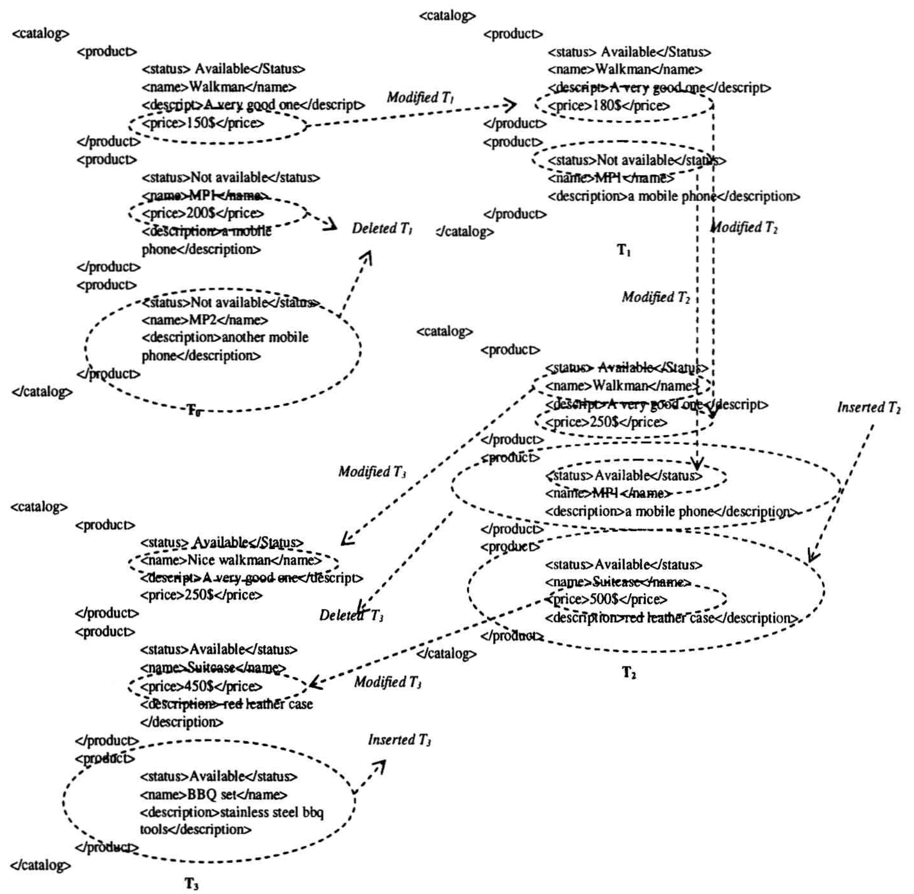
3 Problem Specification

To exemplify the problem, in Figure 2 we present one XML document, at the time T_0 (the initial document), followed by three versions, at three consecutive moments of time, i.e. T_1, T_2 and T_3. Each version brings some changes to the previous one, visually represented by the dotted lines.

One technique for storing the changes of a dynamic XML document (i.e. which changes its context in time) was proposed in [1]. Three main features of this technique are: (*i*) the resulting XML document is much smaller in size than the sum of all versions' sizes; (*ii*) it allows running a simple algorithm to extract any historic version of the document and (*iii*) the degree of redundancy of the stored data is very small, only the necessary information for quick versioning being included.

By running the consolidated delta algorithm [1], we obtain a single XML document containing the historic changes on top of the initial document. We resume here the main steps in building the consolidated delta and few important concepts, for a document changing from the version D_i (at time T_i) to version D_j (at time D_j):

- unique identifiers are assigned for the new inserted elements in the D_i version;
- version D_j is compared with the previous one D_i and for each changed element in the D_j version, a new child element is inserted in the consolidated delta, namely <stamp>, with two attributes: (*a*) "time" , which contain the T_i value (e.g. month, year etc) and (*b*) "delta" , which contain one of *modified*, *added*, *deleted* or *unchanged* values, depending on the change detected at the time T_i; there are some rules to be observed when adding the <stamp> elements [1];
- the D_i version is removed from the data warehouse, as it can be anytime recreated using the consolidated delta. The D_j version is kept until a new version arrives or until a decision to stop the versioning process is taken; Dj will be removed after the last run of the consolidated delta algorithm;
- at the end of the process, the consolidated delta will contain enough historical information to allow for versioning.



```
<catalog>
    <product>
        <status> Available</Status>
        <name>Walkman</name>
        <descript>A very good one</descript>
        <price>150$</price>
    </product>
    <product>
        <status>Not available</status>
        ~~<name>MP1</name>~~
        <price>200$</price>
        ~~<description>a mobile~~
        phone</description>
    </product>
    <product>
        <status>Not available</status>
        <name>MP2</name>
        <description>another mobile
        phone</description>
    </product>
</catalog>
```

T0

```
<catalog>
    <product>
        <status> Available</Status>
        <name>Walkman</name>
        ~~<descript>A very good one</descript>~~
        <price>180$</price>
    </product>
    <product>
        <status>Not available</status>
        ~~<name>MP1</name>~~
        <description>a mobile phone</description>
    </product>
</catalog>
```

T1

```
<catalog>
    <product>
        ~~<status> Available</Status>~~
        <name>Walkman</name>
        ~~<descript>A very good one</descript>~~
        <price>250$</price>
    </product>
    <product>
        <status>Available</status>
        ~~<name>MP1</name>~~
        <description>a mobile phone</description>
    </product>
    <product>
        <status>Available</status>
        ~~<name>Suitcase</name>~~
        <price>500$</price>
        ~~<description>red leather case~~</description>
    </product>
</catalog>
```

T2

```
<catalog>
    <product>
        <status> Available</Status>
        <name>Nice walkman</name>
        ~~<descript>A very good one</descript>~~
        <price>250$</price>
    </product>
    <product>
        <status>Available</status>
        ~~<name>Suitcase</name>~~
        <price>450$</price>
        ~~<description>red leather~~ case
        </description>
    </product>
    <product>
        <status>Available</status>
        <name>BBQ set</name>
        <description>stainless steel bbq
        tools</description>
    </product>
</catalog>
```

T3

Fig. 2. The "catalog.xml" document in four consecutive versions

We need to mention that the D_0 version of the XML document (i.e. the initial one) will be included in the initial consolidated delta; from that point, only the changes for any subsequent version will be recorded in the consolidated delta as described above.

After running the consolidated delta algorithm [1] to capture all the changes affecting the running example document in period $T_0 - T_3$, we will obtain an XML document where each initial element from D_0 has attached a history of its changes. Note that, if an element was either deleted at updated at a time Ti, $0<i<3$, its children do not have attached any stamp elements for that specific time and this helps in limiting as much as possible the degree of redundancy of the data stored in the consolidated delta.

In our working example, the changes affecting the initial XML document during the consecutive transformations from a version to another are presented in Table 1.

Table 1. The list of changes for the working example, for three consecutive versions of an XML document

$T_0 \rightarrow T_1$	Price – modified; Product – deleted; Price – deleted
$T_1 \rightarrow T_2$	Product – inserted; Price – modified; Status – modified
$T_2 \rightarrow T_3$	Name – modified; Product – deleted; Price – modified; Product – inserted

If, in our working example, we consider the sets of changes in periods $T_0 \rightarrow T_1$, $T_1 \rightarrow T_2$ and $T_2 \rightarrow T_3$ (see Table 1) as transactions, it can be noticed that the pairs "Price-modified" and "Product-inserted" appear in 2 of the 3 (66%) of the transactions. If the minimum support required is set at a level lower than 66%, the association rule extracted would be: "when a *price* element is modified, a *product* is inserted as well, and this happens in 66% of the sets of changes appearing from a version to another".

This paper proposes to build a generic algorithm for extracting association rules from the changes which affect dynamic XML documents, i.e. to discover if there are any relationships between modifications, deletions or insertions of some elements or another. As exemplified above, the resulting rules could be very informative about how parts of the document are changing together so the user can make predictions about the future behaviour of the dynamic XML document.

4 Mining Changes – The Proposed Algorithm

The algorithm for mining changes from historic versions of dynamic XML documents is an improved Apriori one, redesigned to be applicable to XML documents; it has a preparation step and four main working steps, as follows:

```
For each Ti, 0<i<n
  Get all nodes with timestamp Ti
    For each node with timestamp Ti and delta not "unchanged"
      If the delta is "modified" or "inserted"
        If the node has no other children elements except stamp
           Record timestamp, value and delta
        End if
      Else   ' i.e. delta is "deleted"
        Record timestamp and delta
      End if
    Next
Next
```

Fig. 3. Generic algorithm for extracting historic changes (ECD) from consolidated delta document