An optimization of the structure of databases

VLADIMIR RYBINKIN, ROMAN LUKATSKY
Bureau of Internet Technologies BIT Ltd
Novoposelkovaja str., 6/7, Moscow, Russia, 123459
RUSSIA

Abstract: - This paper is devoted to the technology of optimization of the structure of some database. The criteria of optimality of structure of a database are stated. A number of algorithms of extraction of a metadata from poorly structured data and subsequent modification of the scheme of data on their basis are offered and tested. The iterative change of structure of a real database by means of described technology is executed. The result of this alteration was transformation of initial array of semistructured data into set of relations.

Key-Words: - Data Mining, Software Tools for AI, DBMS, Metadata, Graph

1 Introduction

The problem of analytical processing of non-uniform data, including semistructured ones, presented by means of a web sites, of text files, of data of relational DB, etc. is a one of the most popular themes of scientific researches. These data can be presented in the diversified formats of various information systems, can be described by different metadata, and these distinctions create serious difficulties for data analysis. Numerous attempts to use for the analysis a semantic conformity, particular algorithms or specialized DBMS [1] [2] [5] are for the present far from success. Hereinafter we'll tell about processing of the data of databases, which can be presented in textual format (numerical fields of relational DB, etc.).

2 Problem Formulation

The purpose of this paper is to reveal an optimal structure of arbitrary database and to modify the scheme of data to this state. As the data of DB can have in general an arbitrary or unknown structure, we plan determine information about the structure of original data in an automatic mode exclusively. This information, extracted from data themselves and their links, and also from the metadata already presented in structure of a DB, was used for search of optimal structure of a concrete database. The offered method is founded on repeated re-structuring of a database. The data should be presented in graph-oriented data model, as it has flexible means of representation of arbitrary data structures. This opportunity is provided by graph DBMS "Sindbad", which allows using for designing not only nodes and arcs, but also separate fields and tags of nodes [6].

Each node of the graph can have in general an arbitrary number of directed arcs and nondirectional (bidirectional) edges. Besides, each node can have information fields of various types, which, in turn, can be connected with each other by different ways. Quantity and type of information fields, kind of their links are determined by contents of tag fields of the node (fig.1).

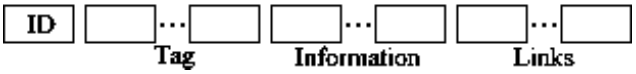

Fig.1. The structure of the node.

Differently, a node of the graph in format of DBMS "Sindbad" is considered as a subgraph and, on the contrary, some subgraph or even whole graph of a database can be considered as integral unit of the graph type. Thus, the set of elementary operations in this DBMS includes addition and removal of nodes and edges, an assembling and disassembling of nodes and a moving of data between fields of the node.

Optimization of DB structure includes in 3 steps:

- Preparation of data and metadata for the analysis.
- Extraction of metadata, their comparison with original ones and selection of scheme of a database.
- Synthesis of optimal structures for the concrete DB.

On the first step the data were modified to a kind, suitable for the further analysis. Identification of metadata is simultaneously made. The method of identification of metadata depends on a format of representation of data (names of columns of a relational DB, of XML-tags, etc.) and, certainly, assumes presence of the corresponding utility of processing of these data in DBMS. The identified metadata were recorded in the database as usual data. Such approach allow to solve some important tasks:

- Unification of data types, i.e. reduction of their original diversity to the limited set of basic types.
- Unification of models of data presentation, irrespective from their initial models.
- Unification of representation and methods of processing of data and metadata.

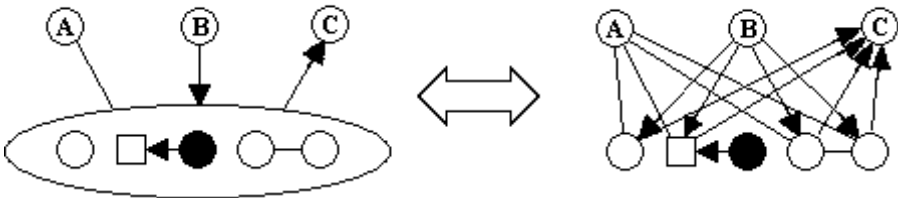

Fig.2. Disassembling of a node.

On the second step a possible regularity was found out in the structure of the graph by special algorithms. On this basis, on the third step, the assembling of new data structures, which easily can be converted into a set of relations, was made. The optimization of structure of data was controlled out by means of minimization of **operational volume** of a DB (its total volume without taking into account contents of information fields). We use this value as criterion of choice of optimal variant from several ones re-structuring of a database. Each step of the offered mechanism will be considered in the course of optimization of the structure of real database.

3 Optimization of structure of a DB

Testing and practical realization of an offered technology was made on the subset "Kids and Teens" of the catalogue of Internet-resources DMOZ [3]. Original data are received from archive, dated 22.05.2006. Data are presented in RDF format, total volume of archive is 23 MB in 4 files. Computer Pentium-4, 2.4 GHz (512 Mb RAM), Windows-2000, the programming language C were used. Compilation of utilities of batch-mode processing was carried out by means of compiler BC++ v3.1. A web-browser MS IE 5.00 as a client's part was used. All operations for optimization of structure of a DB were carried out by graph's DBMS "Sindbad".

3.1 Preparation of data for analysis

We suppose that a database always contains two special nodes: ENTRANCE (the main source) and METADATA (the main sink). Any node of the graph necessarily is connected with both these poles, directly or through the intermediate nodes. Operation of a binding is carried out during the import of data. Even the isolated nodes of the graph become accessible through automatically formed node ERRORS (DBMS by default consider such nodes as errors in data). Similarly, nodes of an unknown format become attached to the node of metadata through node UNKNOWN. Metadata for our DBMS is any node having the outgoing arc to node METADATA, so metadata can easily transformed to usual data and vice versa in contrast to of schemas with standardized set of metadata [4].

After creation of an empty database, it contains two special nodes only. The first of them have operator-entered name of a database (DMOZ, Kids and Teens, data type DATABASE). Four original files have been imported to new nodes connected with it. Each of these nodes contains filename and the data of the corresponding file as a continuous binary array of data type RDF. After import the DB consists of 9 nodes and 16 arcs, the total volume is 22.3 MB (table 1, step 0). All further processing was carried out with the created graph only.

Assembling and disassembling of nodes are the basic operations of re-structuring. Disassembling is a process of formation of new nodes, which contains some number of fields from the original node of the graph. Edges and arcs are created between the new nodes and they form a subgraph, congruous to links between original fields of the node (fig.2). Besides, new nodes inherit all edges (A), incoming (B) and outgoing (C) arcs of the original node. Exception is made for non-terminal nodes of a tree if hierarchical links between the fields are exist (black).

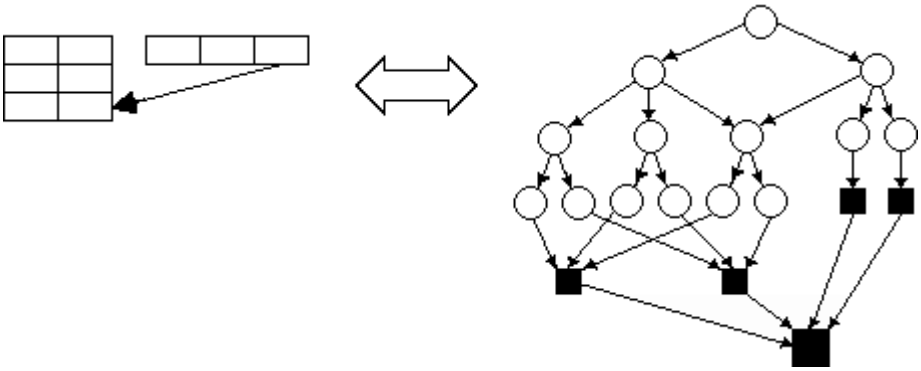

Fig.3. Splitting of the relational database.

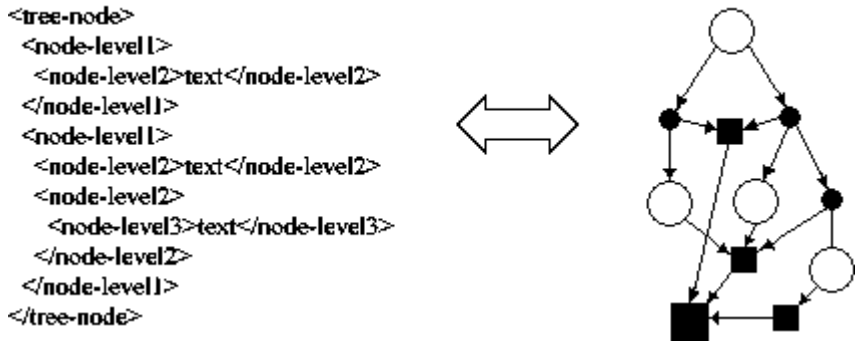

Fig.4. Splitting of the XML-document.

If, for example, the original node contains data of relational DB, the utility of a fragmentation of relations will be called. This utility allocates each relation in separate nodes, and then splits them into corteges and their fields (fig.3). Before splitting of the relation, DBMS puts into separate nodes all names of their columns and forms the arcs from these nodes to node METADATA (are represented as squares). Later, at splitting of each cortege, DBMS forms the arcs to corresponding nodes with metadata. The data format of these nodes (STRING, INTEGER, FLOAT, etc.), after checking of conformity of contents to the type declared in metadata, will be transformed into arcs to the corresponding nodes of data type. Reference fields of relations do not form personal nodes, but transform into arcs of the graph.

The splitting, for example, of nodes with data of the XML-document, will be executed by utility of processing of XML-data. This utility allocates contents of XML-tags in separate nodes and recursively opens enclosed tags if necessary (fig.4). The tags, according to recommendations W3C [8], contain the information about the arcs of the graph, presented by means of XML-document. Therefore the structure of the data after fragmentation should be similar to format of DBMS for semistructured data LORE [1]. However, we still do not have a mechanism for transformation of an edge to a node, which would allow edges to have their own arbitrary information. Therefore metadata of edges were placed in separate nodes, incidental to each of nodes of a corresponding arc (are represented as squares), which also were connected with node METADATA.

As embedded tags of RDF-documents are specified on spaces of parent's tags only, processing of RDF-documents has some specificity. The utility creates the separate node with a new tag's name, united through a hierarchical separator with the name of parent's tag, and forms an arc from this node to the node of parent's tag. By the described way the utility of DBMS "Sindbad" for splitting XML/RDF data has divided four initial nodes to 310413 new (table 1, step 1). Then an identification of standard formats of data (STRING, DATE, URL, etc.) has been made.

Merging of nodes with equal values of information fields was carried out in the assumption, that the coalescing nodes are the same node at a logic level. At that all nodes with double value of information fields were deleted, their arcs were united and possible multiple arcs and loops were removed. There is a danger of occurrence of errors caused by a homonymy of values of data. For instance, a merging of nodes with the same name of RDF-tag, of nodes of type URL, is quite reasonable, since such nodes cannot represent different information units. On the contrary, a merging of rubrics of the hierarchical catalogue can leads to logic errors. For example, a merging of nodes with information value Museums will lead to erroneous association of resources of rubrics Madrid->Museums and London->Museums. However even if that's the case it is important to reveal undoubted logic relationship between this nodes, what, probably, is lacked an original data.

Thus, after ending of this step, original structure of a DB was seriously changed. Each node of a DB contains one information field only. All initial data are disjoined on separate nodes and do not depend on their original formats and data model. All metadata keeps their status in graph's representation only, by having an arc to node METADATA. All other nodes are incidental, at least, to one of these nodes. This transformation allows extracting the metadata from the structure of graph itself and allows modifying structure of data and metadata. After ending of this step each node has a unique value of its single information field and has from 2 to 26 outgoing arcs to nodes of metadata (table 1, step 2).

3.2 Optimization of metadata

For each node the quantity of incoming and outgoing arcs has been counted up. If this value exceeded a threshold of the statistical importance, the node was considered as metadata, and it receives an arc to node METADATA (9 nodes altogether). Threshold was specified as 1% of volume of an analyzed database. The further analysis of structure consists in revealing any regularity in an analyzed database or some subset

of its elements. The method of revealing is based on comparison of these sets. We shall use concepts: **group** (vertex set) and **channel** (edges set) if all its elements possess some commonality. After operation "preparation of data" such commonality can be the graph's characteristics only: the group of sinks of some channel, the internal channel of some group, etc. Separate nodes can be considered as group. The same node can enter into any number of groups.

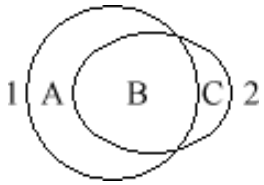

Fig.5. Comparison of groups.

Identification of groups was made on initial set of nodes of a DB, since METADATA node. The channel of this node has 51 arcs and, as it is specified above, identifies nodes of metadata. Channels of all nodes of metadata were consistently used for identification of new groups as the set of opposite nodes of respective channel. An information field of node of metadata determined a name of the formed group. Each of the named groups was checked on presence of the **internal channel** (set of arcs between nodes of the same group). By this sign the separate groups can form **sinks**, **sources**, **isolators** (not having arcs of this channel) or **conductors** (having both outgoing and incoming arcs). As a matter of fact, a process of identification can be much more complex, say, by allocation of groups on connectivity of subgraph, by external channel, by repeated analysis of already detected groups, etc. However we limited for the time being this operation to described and single way of identification only.

If an internal channel is absent, the new groups cannot be identified. Otherwise, if it was be possible to find out a group of isolators, it allocates to a separate group (the code 1 was added to original name of group). Groups of sinks (code 2) or sources (code 3) also forms a new groups, if a quantity of their channel's arcs is no less than a half of quantity of investigated group's nodes. If these methods were unable to detect new groups, an attempt to divide the explored group by bidirectional edges was made. A new group was identified (code 4), if an amount of edges was no less than a half of quantity of nodes and moving off edges conduct to forming of isolators. The rest of analyzed group after allocation from it everyone new group (**complement**) also forms a group (code 0). It has been as a result identified 24 groups with the internal channel and 60 groups of isolators.

To find out an optimal structure, all the detected groups have been compared in pairs with each other in a few iterations. The result of comparison roughly is shown on fig.5. On the first iteration, if fully identical groups came to light, one of theirs was deleted from the list of groups, and the second got the compound name, including a name of removed group and got a **rank** (number of elements within group's name). On the second iteration all groups, which are entirely composed from two or more of rank groups were split, and each rank's group added inherited name. On the third iteration were considered conflict situations, when non-overlapping sets are below and overlapping sets (C) is above a threshold of the statistical importance. Conflicts were always resolved in favour of the group having more a high rank. In case of equality of ranks the incorporated group was made of all three areas A, B, C.

Let's put, that the group 1 on fig.5 has a higher rank. If a set B is below a threshold of the statistical importance, it was separated to a personal group and was marked like possible errors in data and a residue of the group 2 keeps its name. If a set of a higher rank A is below a threshold of the statistical importance, it also is marked like possible errors and supplemented with missing name up to name of group C. Besides, tiny groups (less than 1% from threshold) were marked like possible errors too.

On a fourth iteration all groups that are a component part of group with equal or higher rank and do not marked as errors, were removed. In this way it has been revealed 10 statistically significant groups (in the further GROUPS) from which only one has the internal channel, 10 SMALLGROUPS from which 7 have the status POSSIBLE_ERROR and 8 TINYGROUPS with the status POSSIBLE_ERROR, - 28 groups altogether.

On the end of analysis, the reforming a DB has been executed: all initial metadata (except of data types) have lost their arcs. Instead of it was created nodes with names of all generated groups; they have formed incoming arcs from each node of corresponding group and an outgoing arc to node METADATA. Just listed nodes of metadata have simultaneously been created and received incoming arcs from nodes with names of the generated groups. As a result, at full preservation of available links between the data, a sharp reduction of number of arcs to metadata there was: any node has from 2 to 3 arcs.

3.3 Optimization of data structure

On this step the analysis of available internal and external channels of the revealed groups is made. In a general view, process of the analysis of channels has shown on fig.6. For simplicity sake we suppose that all channels are presented by nondirectional edges only. As the edge can be represented in any of incidental nodes [6], edges of the channel can be transformed to information fields of one of analyzed pairs groups. Each of nodes of intergroup channel BC has strictly

one edge of the channel (for example, the channel of groups catid/2 and Topic/30). Hence, edges of the channel can form a reference field in any of analyzed groups pair. Unfortunately, such situation meets seldom, and at formation of structural units usually it is necessary to resolve arising conflicts.

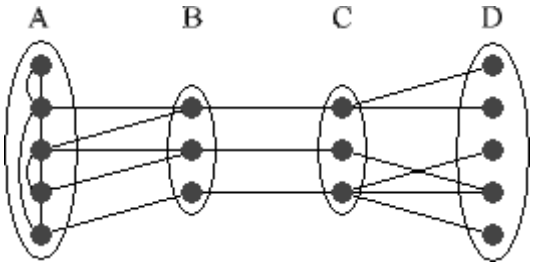

Fig.6. Channel analysis.

Edges of the channel CD also belong to all elements of their groups without exception. However, as group D has more elements, assembly of the channel to nodes of this group absorbs greater number of edges and reduces operational volume of a DB more (for example, the channel of groups Page/2 and Topic/30). The rest of the channel keeps edges that have not received places in link's fields of nodes.

Channel AB not include all nodes of group A, therefore assembling of edges of the channel to its nodes will lead to occurrence in the generated structure of empty fields. On the other hand, assembly of the channel in nodes of group B can absorb smaller number of edges of the channel. Criterion of a choice of a method of representation of edges, as well as in the previous case, the operational volume is. At last, the assembling of the internal channel (group A, for example, Topic/30+narrow) is made to absorb the most of edges in the structure of group's nodes.

It is obvious, that an assembling of nodes of one group with another one and removal of the last together with all of its arcs provide the greatest reduction of operational volume (fig.2). At that an absorbed should be a group with smaller number of structured intergroup channels. In our case only groups Page/2 and Topic/30 have more than one such channel, therefore all conflicts were resolved in their favours. Two groups of nodes with regular structure of information fields and with duplication of values of absorbed nodes or with creation of empty fields if necessary have been created by such way. Edges of the intergroup channel of these groups and of the internal channel of group Topic/30 have remained links. Other groups, and also the nodes that have not entered into regular structures of the absorbed groups, structured theirs own arcs in one or several fields. Metadata of initial group ages/2 have been reformed into tags of incidental nodes [6]. In relational view of this operation each group is represented as relation, and group Page/2 in addition is broken to 8 relations, and the column ages/2 is deleted from each relation.

It is necessary to note, that at the described assembling of nodes the information about links between elements of the DB, detected on a stage of merging of nodes, can be lost. For its preservation, all the nodes, having a double in fields of created structures, have been left in a DB with their metadata. It has allowed giving the additional information for the end user as an opportunity of navigation on the revealed links. The operational volume of DB, after this structuring, has sharply decreased (table 1, step 3).

3.4 Results

Expert evaluation has shown satisfactory quality of the structuring. The DB, suitable for practical application, was created, although there wasn't any information neither about structure of initial data, nor about the software for data processing. Significant part of data that were identified as errors, were errors in reality. Even without using of special methods of revision [7] some types of logic errors in data have been detected. For example, RDF-tag charset have one value only (UTF-8), but exactly this coding used by default. Marked like errors categories of catalogue do not belong to subset Kids and Teens in reality (for example, Top/Regional/Europe/United_Kingdom).

Table 1. Re-structuring of the graph of database.

Step	Nodes	Edges	Bytes	Description
0	9	16	23416258	Import of data
1	310418	931141	20246071	Splitting of nodes
2	148494	652260	13705432	Merging, removing of multiply arcs and loops
3	55976	81352	9388056	Assembling of data structures
4	166041	866741	1500463	Splitting of nodes
5	138765	806295	12095018	Merging, removing of multiply arcs and loops
6	46275	55834	8613510	Assembling of data structures

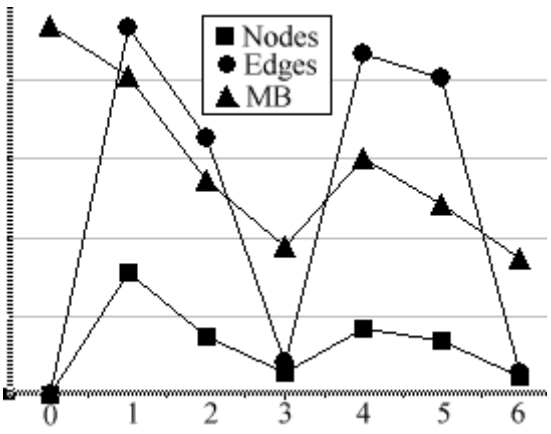

Fig.7. Results.

New data structure allows ending user to navigate on a DB without of search inquiries, including abilities don't stipulated by composers of the initial catalogue. So, it has appeared an opportunity of navigation on age groups (RDF-tag ages), on keywords (term), on editors of rubrics (editor), etc. As a whole, the navigating opportunities of the created DB and the initial catalogue of resources appreciably coincide, except for the data presented by RDF-tag altlang (this category in other languages). Besides the link, there is presented data about the language of a reference rubric on language of current rubric. Such structure was not been identified by DBMS as link. But this fact is not a lack of algorithm at all as we artificially divide into a few stages the process of structuring for demonstration of flexibility of mechanisms of splitting of data, also for simulation of process of change of the scheme of data during an operation time of database.

It is obvious, that process of splitting of fields of nodes can be regulated by introduction of various restrictions. Restrictions can be set by the operator or automatically be defined by algorithms of splitting by data's types. For instance, nodes of type DATIME could be split to nodes of type DATE and TIME. Or it was possible to execute splitting of information fields of nodes of type URL to domains, sites, pages, etc.

Original data has been again split, however this time each group in addition were checked by the special utility on an opportunity of presence of several elements in one data field, implicitly established by means of separators. It was found out, that in information fields of nodes with metadata (RDF-tags) Alias, altlang, symbolic, etc., a symbol ":" meets exactly once and, hence, can be a separator of two different fields. All these nodes have been split with formation of an edge between the revealed fields. Besides, there have been split by a separator "/" all nodes of group ages (table 1, step 4). Other opportunities of splitting, including search of hierarchical separators, have not realized. The process of re-structuring of a database can be repeated if necessary. A modification of structure of database can be demanded because of changing of its contents, tools set or information needs of users.

Repeated application of the described methods had led a DB to final state (table 1, steps 5, 6). Graphically the results are shown on fig.7. Expert appraisal of repeated optimization has shown, that the created DB by the opportunities does not concede, but for number of parameters surpasses a manual catalogue DMOZ. The latest transformation of a database was converted to a set of relations and it is presented on our site (http://www.2bit.ru/dmoz.zip). Its reliability was checked by means of import to format of MS Access.

Thus, the offered technology based on repeated re-structuring of DB, has allowed optimizing structure of an initial array of semistructured data. This technology at some extension of toolkit may be interesting for the decision of other practical problems.

4 Conclusion

The opportunity of changing of scheme of DB can be demanded at designing, updating of the scheme and for data verification of real commercial databases. The offered approaches can be used for the decision of problems in structuring of arbitrary streams of external data, including semistructured one. Information implicitly presented in a database can be extracted and structured, however a high-grade semantic analysis of arbitrary textual data at present time is complicated because of unfinished mechanism of representation of data are placed in edges.

References:

[1] S.Abiteboul, D.Quass, R.GoldMan, J.McHugh and J.Widom, Lore: A DBMS for Semistructured Data, *Technical report, Stanford University*, 1997.

[2] T.Berners-Lee, J.Hendler, O.Lassila, The Semantic Web, *Scientific American*, May 2001.

[3] DMOZ Open Directory Project, *http://www.dmoz.org*

[4] J.Hunter and F.Nack, Combining RDF and XML Schemas to enhance interoperability between metadata application profiles, *Intern. WWW Conference*, 2001.

[5] S.Madnick, Integrating Information from Global Systems: Knowledge Representation and Reasoning in the Context Interchange System, *Moscow ACM SIGMOD*, Chapter 2005.

[6] V.Rybinkin, Realization of some methods of processing of poorly structured data in digital libraries, *RCDL*, 2005, pp 182-190.

[7] V.Rybinkin, Revelation and correction of error of textual data in relational databases, *Moscow ACM SIGMOD*, Chapter, 2004.

[8] W3C Recommendation, 2004, *http://www.w3.org/TR/rdf-primer*