

CZECH TECHNICAL UNIVERSITY IN PRAGUE



DOCTORAL THESIS STATEMENT

Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Cybernetics

Martin Řimnáč

**DATA STRUCTURE ESTIMATION
AND ITS APPLICATIONS**

Ph.D. Programme:

Electrical Engineering and Information Technology

Branch of study:

Artificial Intelligence and Biocybernetics

Doctoral thesis statement for obtaining the academic title of “Doctor”,
abbreviated to “Ph.D.”

Prague, February 2011

The doctoral thesis was produced in *full-time* manner Ph.D. study at the *Department of Cybernetics* of the *Faculty of Electrical Engineering* of the *CTU in Prague*.

Candidate: Ing. Martin Řimnáč
Department of Cybernetics
Faculty of Electrical Engineering; CTU in Prague
Technická 2
166 27 Praha 6

Supervisor: Ing. Július Štuller
Institute of Computer Science, p.r.i,
Academy of Sciences of the Czech Republic,
Pod Vodárenskou věží 2,
182 07 Praha 8

Supervisor Specialist: *(none)*

Opponents:

The doctoral thesis statement was distributed on

The defence of the doctoral thesis will be held on at before the Board for the Defence of the Doctoral Thesis in the branch of study Artificial Intelligence and Biocybernetics in the meeting room No. of the Faculty of Electrical Engineering of the CTU in Prague.

Those interested may get acquainted with the doctoral thesis concerned at the Dean Office of the Faculty of Electrical Engineering of the CTU in Prague, at the Department for Science and Research, Technická 2, Praha 6.

Prof. Ing. Vladimír Mařík, DrSc.

Chairman of the Board for the Defence of the Doctoral Thesis
in the branch of study Artificial Intelligence and Biocybernetics
Faculty of Electrical Engineering of the CTU in Prague
Technická 2, 166 27 Prague 6.

Contents

1	Introduction	1
2	Thesis Aims	2
3	Working Methods	3
3.1	Extensional Functional Dependencies	4
3.2	Functional Dependency Discovery	5
4	Results	6
4.1	Incremental Functional Dependency Discovery	6
4.1.1	Basic Version of Incremental Algorithm	6
4.1.2	Optimised Version of Incremental Algorithm	7
4.1.3	Complex Attribute Support	8
4.1.4	Method Usage	9
4.2	Supporting Data Integration	9
4.2.1	Naïve Element Mapping	9
4.2.2	Attribute Correspondence	10
4.2.3	Instance Mapping	10
4.2.4	Weights	10
4.3	Reputation System	11
4.4	Building a Semantic Web Portal	11
5	Conclusion	12
6	Summary	16
7	Resumé	17
8	Author Publications	18

1 Introduction

The amount of data presented on the web has been increasing almost exponentially in the past 20 years. The web data is usually provided in independent documents, very often in a redundant way. Searching web data effectively is in general rather problematic. Current search engines can return thousands of links to a given query. As no postprocessing is systematically used, obtaining a complete, well arranged answer to a given query, navigating and aggregating data from all the (relevant) sources, is far beyond the actual possibility of current search engines. This is the reason why new advanced technologies for data searching on the web are so much studied.

One of such approaches is based on the semantic web vision, where search engines may build inverse indices over semantic web triples. The vision inspires a **linked data** approach, interpreting the whole web as a crosslinked environment carrying a resource description, where connected sources share, redefine or extend the description of a (subject) resource. The question the thesis is aimed to answer is whether data from current web sources can be (semi)automatically transformed into any structured data format.

Since a relational data model is frequently used for manipulating and storing data, the thesis modifies conventional approaches to **functional dependency discovery** (they were developed for a (once applied) semiautomatic creation of relational modes for adhoc structured data, when the relational database approach has been well established) into an incremental version for web sources providing often actualised data without any additional description of their structure. The proposed methods incrementally estimate a data structure described by (active) attribute domains and a functional dependency system and store data into a repository in an effective way according to these estimated structural relationships.

Similarly, the structural relationships can be estimated also over the sources in the environment in order to support a cooperation between them. Moreover, a reputation of particular sources is evaluated; honoured sources in the environment presenting sound and actual data should be benefited.

All the estimated structural relationships are concentrated into a proposed framework, which allows data extracted from conventional web pages without available data structure to be processed as data with available data structure description, and consequently to provide cooperation between both types of data. Finally, the framework can be materialised into a modern semantic web portal to manage, to aggregate and to search data from all data sources in a way returning a complete, well arranged answer to a given query.

2 Thesis Aims

The thesis focuses on the estimation of various structural relationships of data coming without their description. For this purpose, the thesis presents

- an *incremental algorithm for data structure estimation*, using a new formalism handling structural relationship and data storage,
- *special measures* for evaluating quality of the estimated structural relationships (partly inspired by semiautomatic approaches to data integration),
- a *reputation system* as a support for possible data inconsistency resolution,
- a *semantic web portal* presenting data from the built repository.

3 Working Methods

The data are often stored and manipulated in **relational databases** proposed by Codd in 1970 [4]. For the corresponding relational data model [14]:

*Let \mathcal{A} be a finite set of attributes $\{A; A \in \mathcal{A}\}$ and $\mathcal{D}(A)$ the corresponding attribute domains. A **Relational Schema** S is a couple*

$$S = (\mathcal{A}, \{\mathcal{D}(A), A \in \mathcal{A}\})$$

Many approaches further extends a schema S by so called **integrity constraints** precisely defining allowed tuples. These constraints can be in general expressed by restrictions on (complex) attribute domains. Such an extended schema is often called a **header**. Then [7]

- A **Relation** $R(H, \mathcal{T})$ is a couple of a header H and a set of tuples \mathcal{T} describing the objects.
- A **Relational Database** is a set of relations $\mathcal{R} = \{R_i\}$.
- A **Relational Database Schema** of the relational database \mathcal{R} is a pair $(\mathcal{H}, \mathcal{C})$, where \mathcal{H} is a set of headers H_i of relations $R_i(H_i, \mathcal{T}_i) \in \mathcal{R}$ and \mathcal{C} is a set of constraints.

A special kind of constraints is a **functional dependency** [6]. Let be

- S a relational schema over attributes \mathcal{A} , and
- $R(S, \mathcal{T})$ be any relation over the schema S .

A **Functional Dependency** $f = \mathcal{A}_i \rightarrow \mathcal{A}_j$ over a schema S is a constraint between attributes $\mathcal{A}_i \subseteq \mathcal{A}$ and $\mathcal{A}_j \subseteq \mathcal{A}$, which assures that a value of the attribute \mathcal{A}_j has to be uniquely defined by a value of the attribute \mathcal{A}_i for each tuple $t \in \mathcal{T}$ from any relation over the schema S , e.g.

$$(\mathcal{A}_i \rightarrow \mathcal{A}_j) \Leftrightarrow \forall R(S, \mathcal{T}) \forall t_1, t_2 \in \mathcal{T} : t_1(\mathcal{A}_i) = t_2(\mathcal{A}_i) \rightsquigarrow t_1(\mathcal{A}_j) = t_2(\mathcal{A}_j)$$

All functional dependencies over the schema S will be called the **Functional Dependency System** over S and will be denoted \mathcal{F}_∞ .

The functional dependencies just defined are called **intensional**; they hold for any relation having the same schema S .

*The functional dependency is **trivial**, if it can be inferred by a rest of functional dependencies in the system using Armstrong axioms.*

The intensional functional dependencies can be analysed for

- the *model verification* on a set of tuples. If the constraints disallow insertion of any tuple from the testing set, the verification fails and the model has to be redesigned. The often verified characteristics are a functional dependency system and other integrity constraints (i.e. attribute domains).

- the *model decomposition*. These tasks analyse all valid functional dependencies and verify a given model to meet additional restrictions. The functional dependency system is verified to satisfy conditions given by database normal forms [11] or may be used to propose the model changes in order to satisfy higher normal form [2].
- the *operations over the model*. The similar techniques are useful for automated redesign of the model for special purposes, as for example, the multiseure level model, materialised or non-materialised optimisation issue, etc [21].

3.1 Extensional Functional Dependencies

The issue focused in the thesis deals with a situation, when only tuples \mathcal{T} are available from any relation $R(H, \mathcal{T})$, i.e. the data from the relation are available only as a (sub)set of tuples $\mathcal{T}' \subseteq \mathcal{T}$ without the corresponding intensional header H (a relational schema, including integrity constraints, i.e. functional dependencies), see Figure 1. Since these constraints, especially functional dependencies, are important for advanced data processing, the **functional dependency discovery** methods [1, 9, 17] try to estimate the extensional header H' directly from the tuples \mathcal{T}' .

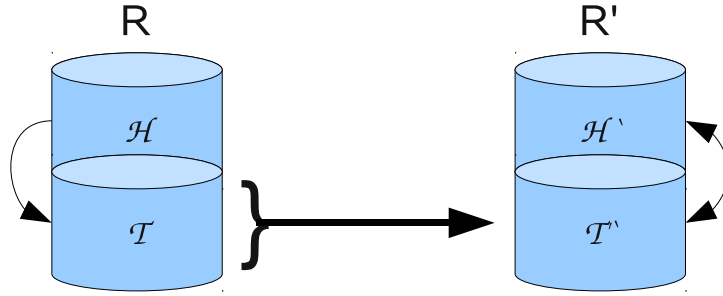


Figure 1: Motivation Scenario

The **extensional headers** are obtained by an analysis of a particular subset of tuples; the methods observe features valid over a given set of tuples \mathcal{T}' and try to estimate the constraints valid over the particular data set. The extensional (estimated) header may include

- *The Attribute Active Domain $\mathcal{D}_\alpha^\mathcal{T}(A)$ of the attribute $A \in \mathcal{A}$ with respect to the given set of tuples \mathcal{T} is the part of an attribute domain $\mathcal{D}(A)$, which is formed by elements from the tuples \mathcal{T} using:*

$$\mathcal{D}_\alpha^\mathcal{T}(A) = \{v; \exists t \in \mathcal{T} : (A, v) \in t\}$$

- Let \mathcal{T} be a given set of tuples over attributes \mathcal{A} . An **Extensional Functional Dependency** is a relationship between attributes $\mathcal{A}_i, \mathcal{A}_j \subseteq \mathcal{A}$ holding if a value of an attribute \mathcal{A}_j is uniquely defined by a value of an attribute \mathcal{A}_i for each tuple $t \in \mathcal{T}$. The set of extensional functional dependencies over tuples \mathcal{T} will be called the **extensional functional dependency system** as will be denoted $\mathcal{F}_{\mathcal{T}}$.

$$\mathcal{A}_i \rightarrow \mathcal{A}_j \in \mathcal{F}_{\mathcal{T}} \Leftrightarrow \forall t_1, t_2 \in \mathcal{T} : t_1(\mathcal{A}_i) = t_2(\mathcal{A}_i) \rightsquigarrow t_1(\mathcal{A}_j) = t_2(\mathcal{A}_j)$$

It is known

$$\forall \mathcal{T} : \mathcal{F}_{\mathcal{T}} \supseteq \mathcal{F}_{\infty}$$

the equivalence holds for tuples corresponding to so called Armstrong relations [1].

3.2 Functional Dependency Discovery

The basic algorithm for discovering functional dependencies is called *naïve*, and was proposed in [15, 16, 17]. The algorithm input is a set of tuples \mathcal{T} and a set of attributes \mathcal{A} . The algorithm output is an extensional functional dependency system $\mathcal{F}_{\mathcal{T}}$.

Algorithm 1 Naïve Algorithm

Require: Tuples \mathcal{T} , Attributes \mathcal{A}

Ensure: Extensional functional dependency system $\mathcal{F}_{\mathcal{T}}$

- 1: $\mathcal{F}_{\mathcal{T}} = \emptyset$;
 - 2: **for** $\forall A_R \in \mathcal{A}$ **do**
 - 3: **for** $\forall \mathcal{A}_L \subseteq \mathcal{A} - \{A_R\}$ **do**
 - 4: **if** $\forall t_1 \in \mathcal{T} \forall t_2 \in \mathcal{T}, \mathcal{A}_L(t_1) = \mathcal{A}_L(t_2) \rightsquigarrow A_R(t_1) = A_R(t_2)$ **then**
 - 5: $\mathcal{F}_{\mathcal{T}} = \mathcal{F}_{\mathcal{T}} \cup \{\mathcal{A}_L \rightarrow A_R\}$;
 - 6: **end if**
 - 7: **end for**
 - 8: **end for**
 - 9: **return** $\mathcal{F}_{\mathcal{T}}$;
-

Since the complexity of test at line 4 is $\mathcal{O}(|\mathcal{A}_L| |\mathcal{T}| \log |\mathcal{T}|)$ and the test is $(|\mathcal{A}| \cdot 2^{|\mathcal{A}|-1})$ times repeated, a whole algorithm complexity is

$$\mathcal{O}(2^{|\mathcal{A}|-1} \cdot |\mathcal{A}|^2 \cdot |\mathcal{T}| \cdot \log |\mathcal{T}|)$$

All functional dependency discovery methods are based on the naïve algorithm, but they eliminate an amount of functional dependencies to be tested, for example by removing trivial functional dependencies, which can be derived by Armstrong axioms from already detected functional dependencies, from a candidate set.

4 Results

The thesis proposes to use a formalism of elements defined as attribute value pairs. The special matrix notation is used, the data are stored as instances - associative rules between elements - in the (binary) repository matrix, which has to respect the estimated extensional functional dependency system.

4.1 Incremental Functional Dependency Discovery

The thesis states that the functional dependency discovery can be handled in an incremental way, if all the input tuples are step by step stored into a repository (e.g. a repository is an archive without removing instances). Under this condition, the functional dependency discovery can incrementally test only a newly inserted tuple instead of testing all tuple pairs.

Firstly, the thesis restricts a whole set of all possible functional dependencies (a $\mathcal{P}(\mathcal{A}) \times \mathcal{P}(\mathcal{A})$ space) into the relationships between simple attributes (a $\mathcal{A} \times \mathcal{A}$ space) only. Using such a restriction and an incremental version, the complexity of testing these $\mathcal{O}(|\mathcal{A}|^2)$ functional dependencies is significantly reduced; while a whole data set analysis complexity is $\mathcal{O}(|\mathcal{A}|^2 \cdot |\mathcal{A}| |\mathcal{T}| \log |\mathcal{T}|)$, the incremental approach complexity is $\mathcal{O}(|\mathcal{A}|^2 \cdot |\mathcal{A}| |\mathcal{T}|)$. The thesis proposed several versions of the algorithm for the incremental functional dependency discovery:

- a basic version analysing so called *full form functional dependency system* gathering all $\mathcal{O}(|\mathcal{A}|^2)$ functional dependencies
- an advanced version selecting only $\mathcal{O}(|\mathcal{A}|)$ functional dependencies into the *skeleton* (a transitive reduction of the full form functional dependency system), which have to be tested. Only in the case the test of a functional dependency in the skeleton fails, next at most $\mathcal{O}(|\mathcal{A}|)$ functional dependencies should be additionally tested. Since a transitive reduction is generally a task returning ambiguous solutions, the skeleton is optimised¹ to cover a minimal amount of instances required to be stored. Therefore, using the skeleton functional dependencies leads to an effective data storage.
- based on skeleton algorithms, the bottom up algorithm decomposing input tuples is proposed.

4.1.1 Basic Version of Incremental Algorithm

The incremental algorithm is based on the fact, that the repository can be initialised by the first tuple described by a set of elements \mathcal{E}^1 . Since there is

¹As the second side effect of applying such a criterion minimising amount of instances, the relational models based on an estimated functional dependency system satisfy a third database normal form

no crossexample², the functional dependency system is initialised by a whole space $\mathcal{A} \times \mathcal{A}$ and all instances $\mathcal{E}^I \times \mathcal{E}^I$ are stored into the repository matrix. During insertion of other tuples, the (extensional) functional dependency condition may fail, such a functional dependency is corrupted, and all its related instances are removed from the repository matrix.

4.1.2 Optimised Version of Incremental Algorithm

Usage of the skeleton of the functional dependency system is based on transitivity, which holds in each functional dependency system (e.g. functional dependencies to be inferred by rest ones are removed from the skeleton). Several basic configurations of the skeleton are a **star skeleton**, a **chain skeleton** and a **cycle skeleton**.

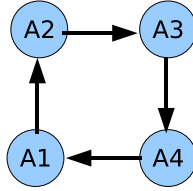


Figure 2: Cycle Skeleton

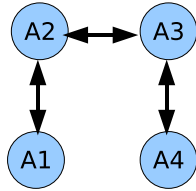


Figure 3: Chain Skeleton

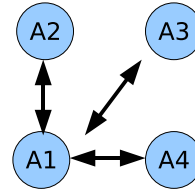


Figure 4: Star Skeleton

The incremental algorithm using the skeletons is based on analysing the functional dependencies in the skeleton only, whether a new inserted tuple is a crossexample to any already stored tuple. In a positive case, corresponding functional dependencies are corrupted, and therefore cannot be further used for storing corresponding instances (as in the previous algorithm variant), but also for inferring other (valid) functional dependencies outside the skeleton. As proposed in the thesis, the algorithm then tries to update the skeleton in order not to cover any corrupted functional dependency under keeping all valid functional dependencies to be transitively derivable from functional dependencies in the skeleton. Such a situation is illustrated on Figure 5, where keeping the functional dependencies outside the skeleton is

²A tuple, which together with a tuple to be inserted causes a corruption of a functional dependency.

warranted by inserting functional dependencies, which can be derived from the corrupted functional dependency and a rest of functional dependencies in the skeleton. Such an insertion creates 'a bridge' over the corrupted functional dependency as illustrated on Figure 6.

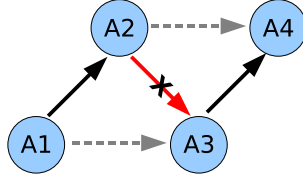


Figure 5: Skeleton Failure

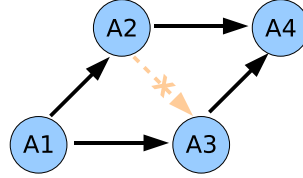


Figure 6: Solving the Failure

As illustrated on Figure 7, the functional dependency to be inserted as an 'arm of the bridge' may be also corrupted. In such a case, the (arm corresponding) side of bridging the functional dependency is moved in the skeleton in order to reach a functional dependency, which is valid. In special cases, such movement leads to a functional dependency between the same attributes $A \rightarrow A$, which is trivial - therefore the arm of the bridge is destroyed.

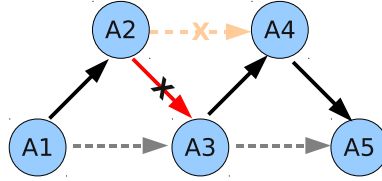


Figure 7: Skeleton Before Update

All the changes in the skeleton should keep no trivial functional dependency is included in the skeleton. After the skeleton is updated for all corrupted functional dependencies, the change corresponding instances are evaluated and consequently the instances related to corrupted functional dependencies are removed from the repository.

4.1.3 Complex Attribute Support

Under the condition all the input tuples include also an element of simple attribute to be a primary key, the functional dependencies with a complex attribute on the left side can be discovered as additional metadata only, e.g. without a need to store them corresponding instances. Unfortunately, the amount of such functional dependencies is given by a combinatorial number, e.g. it is not polynomial. The proposed extension tries to minimise the amount such functional dependencies as much as possible using the same restrictions as other relevant approaches in this area (keeping an incremental test).

4.1.4 Method Usage

The resulted functional dependency system can be used for creating data structure description by a relational model. An enabled direct connection of the estimated functional dependency system with a repository matrix enables to store in time data effectively. On the other hand, such a direct connection disallows to precisely compare an effective complexity of the proposed incremental variants with conventional offline approaches.

A part of the estimated functional dependency system including functional dependencies between simple attributes and stored instances can be used for defining a (semantic web) ontology. The instances can be seen as associative rules, which can be further extended by metadata describing data structure. For example, an estimated functional dependency $State \rightarrow Capitol$ with an instance $(State, CZ) \rightarrow (Capitol, Prague)$ can be transformed into a definition of the Czech Republic (CZ): The *Czech Republic* is a *State*, which *Capitol* is *Prague*.

Since there is a strong correspondence between the relational data model and the first order logic [8, 10], the (incremental) functional dependency discovery can be used to solve **inductive logic programming** tasks.

4.2 Supporting Data Integration

The conventional data sources are constructed independently on another ones, their headers are designed by different designers, for different purposes or from a different point of view. This is the reason, why these headers are at general heterogeneous. On a natural need of advanced co-processing between data sources, correspondences between the headers have to be established. The problem of finding such correspondence is called **schema matching** [19, 20]. The same issue is also solved in data warehousing [3], data exchange area [13], information retrieval, ontology alignment [18] etc.

The correspondences can be established at the element or attribute level, it can be defined by humans or using a (semi)automatic design driven by measures analysing data over instance, term or structure level. The data integration can be provided by a **mediator system**, which can be centralised (conventional **global as view** and **local as view** approaches) or decentralised (more common for web technologies). Using the same formalism (handling instances related to general relationships over the attributes), the following mappings and measures are proposed.

4.2.1 Naïve Element Mapping

The naïve element mapping is based on the assumption, that an element $(A_i, v) \in \mathcal{E}_{S_k}$ from a source S_k corresponds to an element $(A_j, v') \in \mathcal{E}_{S_l}$ of a source S_l , if the elements are related to the same value $v = v'$. Such a correspondence is stored into the element correspondence matrix. Using

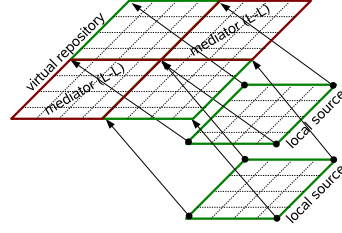
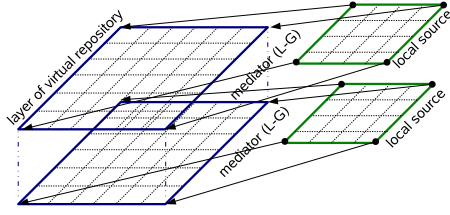


Figure 8: Centralised Approach Figure 9: Decentralised Approach

the naïve element correspondence may lead to invalid establishment of the correspondences.

4.2.2 Attribute Correspondence

The naïve element correspondence can be also used to estimate correspondences between the attributes. Since the naïve mapping correspondences can be propagated to the attribute, each attribute correspondence can be evaluated by an amount of values overlapping the particular attribute domains. Only attribute correspondences having enough overlap should be considered, and only element correspondences of these selected attribute correspondences are allowed. When the (active) attributes domains are disjunctive, the obtained attribute correspondences are unique; alternatively, ambiguous attribute correspondences having a maximal support only are allowed.

4.2.3 Instance Mapping

When the attribute domains are the same, the previous attribute correspondence obtained from naïve element mapping fails (all the correspondences have the same overlap). This is the reason, why searching for correspondences between instances $(A_i, v_i) \rightarrow (A_j, v_j)$ from a source S_k and $(A_{i'}, v_{i'}) \rightarrow (A_j, v_{j'})$ from a source is proposed: The instances can be the same, if $v_i = v_{i'} \wedge v_j = v_{j'}$. These instances can be propagated into an attribute level and the final mapping can be established on getting only the most supported attribute correspondences as in the previous case.

4.2.4 Weights

The estimated correspondences can be extended by a weight, which express a support of the particular correspondence. Weights may play an important role in accessing the distributed repository - when a result is obtained from several sources, each returned element is extended by a weight expressing its relevance to a query over all the sources; the weights can be used for arranging inconsistent parts of results, e.g. when one source presents, for

example, *Prague* is a town in *Czech Republic* and the second *Prague* is a village in *Slovak Republic*. Since the first case may have a bigger support over all the sources, the answer *Czech Republic* is preferred to *Slovak Republic*. Moreover in the decentralised approach, a selection of the source to be queried influences the weights of returned elements, for example when the source providing ZIP codes in Slovak Republic is asked, the answer *Slovak Republic* is preferred to *Czech Republic*.

4.3 Reputation System

Using weights, which affects query results, during accessing the repository opens a new question - how the particular sources may influence a part of a result returned by external sources. One way to be proposed in the thesis is to extend each source by its reputation, a weight expressing a quality and confidence of the particular source. The proposed source reputation is based on an instance overlap, on ability of the source its instances to be confirmed by other (relevant) sources and on an amount of inconsistent instances presented by a source. These mentioned criteria are proposed as input parameters for a complex `reputation system` [5] managing the reputations in the environment of sources.

4.4 Building a Semantic Web Portal

The results from previous theoretical sections can be applied into a semantic web portal. Firstly, the rules transforming repositories (having intensional as well as extensional models) into triples of a semantic web document - an ontology - are proposed. Such an ontology can be processed by advanced semantic web tools (using reasoners) and may be combined with other ontologies. Such a transformation allows to reach a practical thesis motivation, to build a framework in order to process data from conventional web sources by advanced tools for the semantic web. The triple weights established in repositories can be also propagated into resulted ontologies using a fuzzy approach to the semantic web. Including the attribute or element correspondences into the ontology allows to build a cooperating environment similarly to modern approaches in a semantic web area - to a linked data.

Secondary, the triples from various sources can be aggregated together by an aggregated index. Such an index can be searched for resources by a query - a list of features, which resources have to satisfy. Further, this index can collect all triples from all known sources in order to provide and navigate a complete description of requested resources - the resource features can be sorted according to their support - a sum of supports over the sources providing them; the inconsistent parts of the results can be navigated in a sense of preferences in advance.

5 Conclusion

The theoretical framework proposed in the thesis concentrates

- the incremental version of the functional dependency discovery to be used for an estimation of structural relationships from data annotated by names of features in order to enable the data extracted from web pages to be processed with similar tools as data from relational databases.
- an effective data storage using an online estimated data structure
- an estimation of element and attribute correspondences in order to support a cooperation between the sources
- an input parameters for a reputation system evaluating a quality of the sources.

Firstly, it is shown, that an estimated functional dependency system is the same as the system designed by humans, if tuples used for an estimation are representative, e.g. meet the condition to be Armstrong relation. The proposed methods estimating a (extensional) functional dependency system are incremental, they analyse only changes caused by new tuple insertion into the repository (instead of conventional batch approaches, which need to test all tuple pairs). The incremental approach is significantly needed for web sources, which often update the data they provide.

Under the condition each tuple is uniquely identified by a key element, the functional dependencies with complex attributes on the left side can be estimated only as metadata, the corresponding instances can be derived from instances of functional dependencies between simple attributes; e.g. instances of functional dependencies with complex attributes are not required to be stored. A part obtaining such metadata has nonpolynomial complexity, a nonpolynomial amount of tested functional dependencies can be reduced using the same mechanisms as in the case of batch variants. Moreover, the rules transforming instances and the functional dependency system into an ontology are proposed.

Secondary, the problematic of a cooperation between the sources is presented, the methods establishing the mappings between the sources are proposed, and its usage discussed respecting the features of mapped attribute domains. The thesis recommends to weight all the obtained mappings. Finally in order to manage querying the sources, the (weighted) mapping is extended by assuming a quality of the particular sources providing data.

The designed theoretical framework is materialised into an application combining a (intensional) ontologies with data having estimated data structure, which may enable a close cooperation between the conventional web

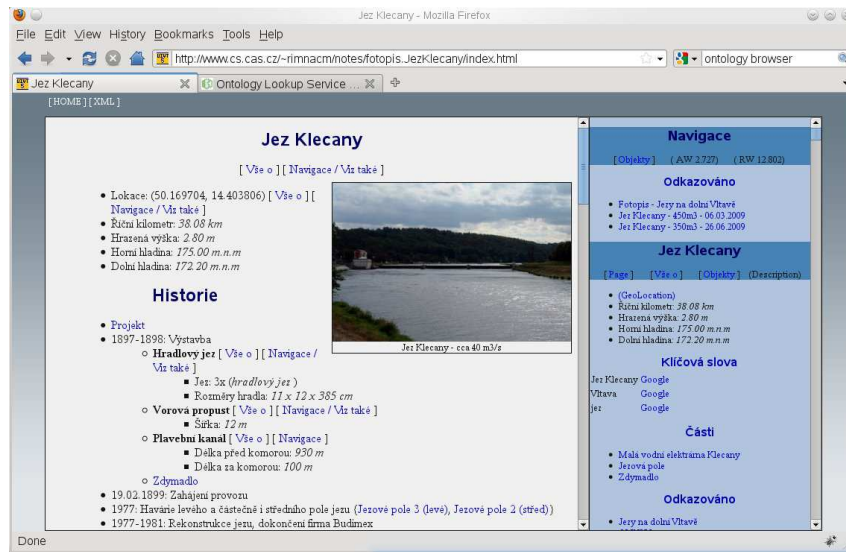


Figure 10: Experimental Semantic Web Portal

sources and semantic web ones, as well as conventional fulltext search engines and semantic web ones (for example based on an aggregated index). Since RDF documents are frequently encoded to an XML format, such RDF documents can be easily transformed into XHTML documents, for example by an XSLT transformation. These transformations, for example [12], enable end users to browse RDF documents via XHTML web pages as well as fulltext engines to index XHTML web page content.

Processing triples from RDF documents allows to aggregate triples over the sources and to build a new modern semantic web search engine with ability to operate also over the conventional web sources. Using such a semantic search enables to find a complete (but may be inconsistent) answer on a given query, which is beyond any possibility of nowadays (fulltext) search engines.

References

- [1] C. Beeri, M. Dowd, R. Fagin, and R. Statman. On the structure of armstrong relations for functional dependencies. *J. ACM*, 31(1):30–46, 1984.
- [2] J. Biskup, U. Dayal, and P. A. Bernstein. Synthesizing independent database schemas. In *Proceedings of the 1979 ACM SIGMOD international conference on Management of data*, SIGMOD '79, pages 143–151, New York, NY, USA, 1979. ACM.

- [3] D. Calvanese, G. De Giacomo, M. Lenzerini, D. Nardi, and R. Rosati. Data integration in data warehousing. *Int. J. of Cooperative Information Systems*, 10(3):237–271, 2001.
- [4] E. F. Codd. A relational model of data for large shared data banks. *Commun. ACM*, 13:377–387, June 1970.
- [5] E. Damiani, S. D. C. di Vimercati, S. Paraboschi, P. Samarati, and F. Violante. A reputation-based approach for choosing reliable resources in peer-to-peer networks. In *Proceedings of ACM Conference on Computer and Communications Security*, pages 202–216, 2002.
- [6] C. Date. *Introduction to Database Systems, An (8th Edition)*. Addison Wesley, 8 edition, 2003.
- [7] C. J. Date. *Database in Depth: Relational Theory for Practitioners*. O’Reilly Media, Inc., 2005.
- [8] R. Fagin. Horn clauses and database dependencies. *J. ACM*, 29:952–985, October 1982.
- [9] P. A. Flach and I. Savnik. Database dependency discovery: A machine learning approach. *AI Communications*, 12/3:139–160, 1999.
- [10] J. A. Gonzalez, L. B. Holder, and D. J. Cook. Experimental comparison of graph-based relational concept learning with inductive logic programming systems. In *Proceedings of the 12th international conference on Inductive logic programming, ILP’02*, pages 84–100, Berlin, Heidelberg, 2003. Springer-Verlag.
- [11] G. Grahme and K. Rähkä. Database decomposition into fourth normal form. In *Conference on Very Large Databases*, pages 186–196, 1983.
- [12] E. Hyvönen, A. Valo, K. Viljanen, and M. Holi. A logic-based semantic web html generator - a poor man’s publishing approach. In *WWW Alt. ’04: Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, pages 294–295, New York, NY, USA, 2004. ACM.
- [13] P. G. Kolaitis. Schema mappings, data exchange, and metadata management. In *PODS ’05: Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 61–75, New York, NY, USA, 2005. ACM Press.
- [14] D. Maier. *The Theory of Relational Databases*. Computer Science Press, 1983.
- [15] H. Mannila and K. Rähkä. Dependency inference. *Proc. of VLDB*, pages 155–158, 1987.

- [16] H. Mannila and K. R  ith  . Design by example: An applications of armstrong relations. *Journal of computer and system sciences*, 33:129–141, 1986.
- [17] H. Mannila and K. R  ith  . Algorithms for inferring functional dependencies from relations. *Data & Knowledge Engineering*, 12:83–99, 1994.
- [18] N. Noy and M. Musen. The prompt suite: Interactive tools for ontology merging and mapping, 2002.
- [19] E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal: Very Large Data Bases*, 10(4):334–350, 2001.
- [20] P. Shvaiko and J. Euzenat. A survey of schema-based matching approaches. 3730:146–171, 2005.
- [21] K. Yazdanian and F. Cuppens. A natural decomposition of multi-level relations, 1992.

6 Summary

The thesis presents a novel approach to data analysis, storage and browsing. It is focused on the estimation of the structural relationships of data missing their structural description, especially of the web data. The thesis proposes to use the well known approach to functional dependency discovery adopted for web data to estimate their structure. As data on the web are usually less or more frequently updated, the thesis innovatively presents this approach in an incremental way.

In comparison to the existing functional dependency discovery methods (estimating data structure from a given, already stored, data set), the incremental variant analyses a stream of continually incoming data, which step by step evolves the estimated structure. The obtained structure is used online for an effective data storage into a dynamically built repository. Moreover the process of data insertion into such a repository is analysed and the description how the inserted data affect the whole repository is presented. Since the obtained structure can be seen as just the best available estimation of the (missing) data description, the correctness and the usefulness of the methodology are discussed as well as the measures expressing impreciseness of the obtained structure. The same principle is further used for semiautomatic data integration approaches, where the appropriate measures play an important role in browsing and navigating trusted data in a general environment of divers web sources.

The application of the obtained theoretical results is materialised in a complex framework, which extracts data from various web sources, stores them effectively (according to their estimated structure), tries to integrate them together, and, finally, enables them to be browsed by end users facing data inconsistency and respecting different trustiness of web sources.

7 Resumé

Disertační práce se zabývá problematikou analyzování, ukládání a prohlížení dat. Specializuje se na webové zdroje, které běžně poskytují data pouze bez popisu jejich struktury. Jelikož tento popis je důležitý pro další zpracování dat, práce navrhuje použít známé metody pro odhad funkčních závislostí, modifikuje je však podle specifických potřeb webových dat, která jsou často aktualizována. Z tohoto důvodu práce inovativně navrhuje inkrementální varianty těchto metod.

Oproti klasickým metodám pro odhad funkčních závislostí (na základě již někde uložených dat) inkrementální varianta analyzuje vstupní proud dat, která postupně zpřesňují odhadovaný model. Odhadnutá struktura dat je ihned využita pro uložení dat. Práce popisuje jednotlivé dílčí kroky vedoucí jak ke zpřesňování modelu, tak i k efektivnímu ukládání dat na základě tohoto odhadnutého modelu. Jelikož na získanou strukturu dat je možné pohlížet pouze jako na odhad, práce uvádí podmínky, při jejichž splnění bude odhadnutý model identický s modelem, který by pro taková data vytvořil člověk. Stejný přístup je posléze aplikován na problematiku návrhu integračních pravidel. Práce doporučuje vážít získaná pravidla odhadem podpory dané korespondence v datech, což ve svém důsledku při dotazování umožňuje poskytované výsledky obohatit o aspekty důvěryhodnosti dat. Ta se odráží v problematice prohlížení a navigace v datech z pocházejících z různých zdrojů.

Předložené teoretické části práce jsou shrnuty do návrhu aplikace, která umožňuje extrahovat data z webových stránek, efektivně je uložit s ohledem na z nich odhadnutou strukturu, pokusit se je zintegrovat s daty ostatních zdrojů a finálně takto zkompletovaná data nabídnout koncovým uživatelům k prohlížení a to včetně nekonzistentních částí. Ty by měly být zobrazeny s ohledem na odhadnutou důvěryhodnost zdrojů, které tato nekonzistentní data poskytují.

8 Author Publications

Recent Publications

- Řimnáč, Martin (50%) ; Špánek, Roman. Automated Incremental Building of Weighted Semantic Web Repository. In Foundations of Computational Intelligence, Volume 6. Berlin : Springer, 2009. S. 265-296. ISBN 978-3-642-01090-3.
- Řimnáč, Martin. Data Structure Estimation for RDF Oriented Repository Building. In Complex, Intelligent and Software Intensive Systems. Los Alamitos : IEEE Computer Society, 2007. S. 147-154. ISBN 0-7695-2823-6.
- Špánek, Roman ; Řimnáč, Martin(50%). The Reputation System for Distributed Data Source Environment. In Applications of Digital Information and Web Technologies. Piscataway : IEEE, 2008. S. 488-493. ISBN 978-1-4244-2623-2.
- Řimnáč, Martin (50%); Špánek, Roman. Hodnocení datových zdrojů pomocí reputačního systému. In Datakon 2009. Praha : Oeconomica, 2009. S. 117-126. ISBN 978-80-245-1568-7.
- Řimnáč, Martin. Minimalising Binary Predicate Knowledge Base using Transitivity Rule in Incremental Algorithm. Invited talk. In Proceedings of the 22nd European Conference on Operational Research. Prague : University of Economics, 2007. S. 171-171.

Next Publications

- Řimnáč, Martin. Semantic Web Portal Using Remarks as RDF Data. In iiWAS2010. New York : ACM, 2010. S. 831-834. ISBN 978-1-4503-0421-4.
- Řimnáč, Martin (33%); Linková, Zdeňka ; Špánek, Roman. Semantic Web: Vision of Distributed and Trusted Data Environment?. In Digital Information Management. Los Alamitos : IEEE, 2007. 627-634. ISBN 1-4244-1476-8.
- Špánek, Roman ; Řimnáč, Martin (33%); Linková, Zdeňka. On Creating a Trusted and Distributed Data Source Environment. In SOFSEM 2008: Theory and Practice of Computer Science. Student Research Forum. 2.. Košice : P.J. Šafárik University, 2007. S. 112-123. ISBN 978-80-7097-697-5.
- Řimnáč, Martin. Data Structure Estimation for RDF Oriented Repository Building. In Frontiers in Mobile and Web Computing. Wien : Österreichische Computer Gessellschaft, 2006. s. 681-685. ISBN 3-85403-216-1.
- Řimnáč, Martin. Transforming Current Web Sources for Semantic Web Usage. In SOFSEM 2006. Theory and Practice of Computer Science. Prague : Institute of Computer Science AS CR, 2006. s. 155-165. ISBN 80-903298-4-5.

- Římnáč, Martin. Návrh portálu pro sdílení RDF poznámek. In *Datakon 2010*. Ostrava : Ostravská univerzita, 2010. S. 187-190. ISBN 978-80-7368-424-2.
- Římnáč, Martin (50%); Špánek, Roman. Hodnocení datových zdrojů pomocí reputačního systému. In *Datakon 2009*. Praha : Oeconomica, 2009. S. 117-126. ISBN 978-80-245-1568-7.
- Římnáč, Martin. Web Information Integration Tool: Data Structure Modelling. In *DMIN'05*. Las Vegas : CSREA Press, 2005. S. 37-40. ISBN 1-932415-79-3.
- Linková, Zdeňka ; Římnáč, Martin (50%). Automatizovaný návrh pravidel pro integraci dat a sémantický web. In *Znalosti 2008*. Bratislava : Vydavateľstvo STU, 2008. S. 124-135. ISBN 978-80-227-2827-0.
- Římnáč, Martin. Advanced Features of Attribute Annotated Data Sets. In *Evolutionary Techniques in Data-processing*. Ostrava : VŠB Technická univerzita, 2007. s. 54-59. ISBN 978-80-248-1332-5.
- Římnáč, Martin (33%); Špánek, Roman ; Linková, Zdeňka. Sémantický web: vize globálního úložiště dat?. In *DATAKON 2007*. Brno : Masaryk University, 2007. S. 176-186. ISBN 978-80-7355-076-9.
- Římnáč, Martin. Portál poznámek aneb polidštění sémantického webu. In *Doktorandské dny '10*. Praha : Ústav informatiky AV ČR, v. v. i. & MATFYZPRESS, 2010. S. 87-88. ISBN 978-80-7378-133-0.
- Římnáč, Martin. Experimenty s RDF úložištěm dat a reputacemi zdrojů. In *Doktorandské dny '09*. Praha : Ústav informatiky AV ČR, v. v. i. & MATFYZPRESS, 2009. 102-102. ISBN 978-80-7378-087-6.
- Římnáč, Martin. Nevyužité možnosti sémantického webu. In *Doktorandské dny '08*. Praha : Ústav informatiky AV ČR, v. v. i. & MATFYZPRESS, 2008. S. 106-111. ISBN 978-80-7378-054-8.
- Špánek, Roman ; Římnáč, Martin (50%). Security and Trust in (Semantic) Web. In *Inteligentní modely, algoritmy, metody a nástroje pro vytváření sémantického webu*. Praha : Ústav informatiky AV ČR, 2008. S. 164-174. ISBN 978-80-87136-03-4.
- Římnáč, Martin. Redukce datových modelů. In *Doktorandské dny '07*. Praha : Ústav informatiky AV ČR, v. v. i. & MATFYZPRESS, 2007. S. 80-86. ISBN 978-80-7378-019-7.
- Tyl, Pavel ; Římnáč, Martin (50%). Kombinace metod pro srovnání ontologií. In *Informačné technológie - Aplikácie a teória*. Košice : Prírodovedecká fakulta, Univerzita P.J. Šafárika, 2008. S. 113-117. ISBN 978-80-969184-8-5.
- Římnáč, Martin. Asociativní úložiště dat v prostředí sémantického webu. In *Inteligentní modely, algoritmy, metody a nástroje pro vytváření sémantického webu*. Praha : Ústav informatiky AV ČR, 2006. s. 102-109. ISBN 80-903298-7-X.
- Římnáč, Martin. Odhadování struktury a asociativní úložiště dat. In *Doktorandský den '06*. Praha : Ústav informatiky AV ČR & MATFYZPRESS, 2006. s. 135-142. ISBN 80-86732-87-8.

- Římnáč, Martin. Odhad struktury dat a induktivní logické programování. In Information Technologies - Applications and Theory. Košice : Přírodovědecká fakulta Univerzity Pavla Jozefa Šafárika, 2005. s. 263-272. ISBN 80-7097-609-8.
- Římnáč, Martin. Odhadování struktury dat pomocí pravidlových systémů. In Doktorandský den '05. Praha : MATFYZPRESS, 2005. s. 124-133. ISBN 80-86732-56-8.
- Římnáč, Martin. Rekonstrukce databázového modelu na základě dat (studie proveditelnosti). In Doktorandský den '04. Praha : MATFYZPRESS, 2004. s. 113-120. ISBN 80-86732-30-4.

Other Publications

- Římnáčová, Daniela ; Ždímal, Vladimír; Schwarz, Jaroslav; Smolík, Jiří; Římnáč, Martin (20%). Atmospheric aerosols in suburb of Prague: The dynamics of particle size distributions, Atmospheric Research, In Press, Corrected Proof, Available online 9 November 2010, ISSN 0169-8095. IF:1.811.
- Římnáč, Martin (33%); Šusta, Richard ; Živnůstka, Jiří. Automatické prohlédávání webových stránek I. Automatizace, 2004, Roč. 47, č. 4, s. 256-258. ISSN 0005-125X.
- Římnáč, Martin (33%); Šusta, Richard ; Živnůstka, Jiří. Automatické prohlédávání webových stránek II. Automatizace, 2004, Roč. 47, č. 5, s. 326-328. ISSN 0005-125X.
- Římnáč, Martin (33%); Šusta, Richard ; Živnůstka, Jiří. Problematika inteligentního automatického mapování webových stránek. In Process Control 2004. Proceedings of the Technical Conference. Pardubice : Univesity of Pardubice, 2004. ISBN 80-7194-662-1.
- Tyl, Pavel ; Římnáč, Martin (12%) ; Špánek, Roman ; Štuller, Július ; Linková, Zdeňka ; Kozler, J. ; Antošová, B. ; Váňa, V. Tvorba ontologie huminových látek. Prague : ICS AS CR, 2010. 32 s. (Technical Report : V-1098).
- Linková, Zdeňka ; Nedbal, Radim ; Římnáč, Martin (33%). Building Ontologies for GIS. Prague : ICS AS CR, 2005. 9 s. (Technical Report. : V-932).

Responses

No responses to my publications are known to me so far.