2015 IEEE 2015 International Conference on Computer, Communication, and Control Technology (I4CT 2015), April 21 - 23 in Imperial Kuching Hotel, Kuching, Sarawak, Malaysia

The Method and Algorithms to Find Essential Attributes and Objects of Subject Domains

Vitaliy Mezhuyev
Faculty of Computer Systems and Software Engineering
Uneversiti Malaysia Pahang
Gambang, Malaysia
mejuev@ukr.net

Eugene Malakhov, Denys Shchelkonogov
Department of Mathematical Support of Computer Systems
Odessa I.I. Mechnikov National University
Odessa, Ukraine
opmev@mail.ru, delis91@gmail.com

***Abstract*—The paper proposes a new model, method and algorithms for processing a Subject Domain (SD). It considers an SD as a tuple (E,V,T), where E is a set of all objects of this domain, V is a set of all connections between these objects, T is a set of mass problems that can be solved in this domain. Each object is represented by its essential attributes. This allows us to describe SD, to solve most of the mass problems from T, and to control a state of the domain. The goal of the paper is to develop a method and algorithms that allow us to find essential attributes of domains, described as relational databases or production systems. For a production system, we build a dependency graph with attributes as vertexes, and calculate for each of the vertexes a value that indicates how many other attributes can be found using a particular vertex. High-ranked vertexes are considered as essential. The method is based on considering as essential of the most frequently accessed attributes in a relational database. Having essential attributes of objects of a domain, it allows us to define essential attributes of the domain itself.**

***Index Terms*—Subject domain, essential attributes, mass problems, relational databases.**

I. Introduction

Subject domains (SD) can be described in a variety of ways (database schemes, UML-diagrams, etc.), but existing models do not allow us to define the mass problems that should be solved in this domain. To contribute in this area, we propose a model of an SD as a tuple (E,V,T), where E is a set of all objects of this domain, V is a set of all connections between objects, T is a set of mass problems that can be solved in this domain. We classify objects as active, passive, and objects that are a result of an intellectual activity [1]. Each object of an SD is described by a set of its attributes.

II. The Problem of a Subject Domain Identification

An identification, a description, an extraction, and a comparison of SDs pose a problem of a large number of attributes of an SD as well as its objects. In spite of the fact that all attributes of every object are significant for its description, an expert in the SD can allocate the attributes that are the most important for this subject domain. E.g. if a car is considered as an SD, then an attribute "engine" is more important than an attribute "a colour of an interior" for solving most of the mass problems (computation of speed, calculation of a cost of transportation, of price for repair; comparison of cars etc.). Nevertheless, a colour of an interior may also be necessary (e.g. for solving a problem of a choice of a car for a specific person depending on the taste). Furthermore, such important attributes allow us to distinct different SDs. Analyses and manipulating such attributes give us possibility to control an SD in order to achieve desirable goals. Let us call such attributes "essential".

Taken together, essential attributes are subset of all attributes of objects of an SD, which defines the SD in general, and allows solving most important mass problems in the SD, identifying the SD, comparing the SD with other SDs, and controlling the SD.

Although using an expert to allocate essential attributes is a common practice, it is not always possible. First, the size of an SD may be so large that an expert or a group of experts will not be able to allocate essential attributes from the general set of attributes in short terms. Second, there may not be a sufficient number of experts in a new SD, which has just appeared. Third, a number of SDs needs to be processed and a time for handling these SDs may exceed human possibilities.

Consequently, the problem of the automatic extraction of essential attributes of SDs arises. To solve this problem a method and corresponding algorithms for extraction of essential attributes have to be developed. Note, that considering a number of essential attributes, approaches and algorithms for their extraction may differ depending on a goal that motivates this extraction.

III. Methods for Describing Models of Subject Domains

The problems of describing, identifying SDs, solving problems in an SD are widely described in a literature. A reason for this is a complexity and a generality of the studied object (i.e., an SD). Although a concept of an essential attribute is not introduced directly in these works, quite similar problems are considered.

At present, there is a great amount of methods for analysis of SDs [2]. Most of these approaches use information about an SD in a textual format and extract knowledge, needed to describe an SD and to solve concrete problems in different ways. However, these methods focus on an extraction of knowledge about an SD in general, not emphasizing on importance of its parts.

Many papers [3-5] are focused on the development of methods for the definition of SDs with a set of keywords extracted from texts describing these SDs. Those keywords may be used either for comparing SDs or for further search of information. Yet, keywords obtained this way are not always attributes of an SD – they may be just expressions, characteristic of this SD, for example, some specific actions. It

978-1-4799-7952-3/15/$31.00 ©2015 IEEE

is extremely difficult to judge importance of extracted attributes because quantitative and structural analysis may be insufficient to solve this problem. E.g., an attribute mentioned in the text only once might be more important than other attributes that specified more frequently. Detection of such the attributes may require semantic analysis of set of linked texts.

Formal representation of SDs is more convenient for automatic processing than a representation in a form of a set of texts. The problems of transformation of a text describing an SD into its structured representation are considered, in particular, in [6]. Its authors suggest a way of constructing a semantic network for a representation of an SD from its textual description. Such representation combines both flexibility of a description and its formality, opening a fair number of possibilities for analysis of the given SD.

Processing of a formal representation of SDs is described in the papers [1], [7], [8]. Their topics are operations on SDs, problems of separation of objects for managing SDs, isolation of object kernels of SDs. Introduction of operations on SDs allows expressing an SD in terms of other SDs, essential dependencies between them, and describe development of an SD. The problem of separation of objects for managing an SD is the neighboring to the problem taken up in this paper. Objects that allow managing SD are, in some sense, essential for this SD, because they allow defining a state of the SD. Extraction of a container and elementary objects allows describing SD at different levels of abstraction and gives a possibility of simplifying solution of mass problems in it.

The book [9] discusses a way of modelling an SD with a set of realizations of Abstracts Data Types (ATD). At the same time, ATD are particularly described by set of operations that they provide. Operations, in turn, are divided into constructors, which create objects of a given type, queries, which provide information about an object of a given type, and commands, which modify an initial object.

The relational data model [10] supposes modelling SD with a set of relations. Each relation is described with a set of attributes. Each attribute may either be an information attribute, which describes an entity, or may belong to a potential key, which allows unambiguously distinguish a tuple from other tuples, or may belong to a foreign key, which allows to define connections between tuples.

The problems of development of information systems for subject domains were also considered in [11] and further elaborated in [12].

To summarize, in [9] authors classified attributes of objects of SDs without emphasizing importance of particular classes of attributes for a description of an SD.

IV. THE AIM AND THE OBJECTIVES OF THE RESEARCH

The aim of the research is to increase effectiveness of solving mass problems in the different SDs. The objectives of this paper is to define the method and algorithms for extraction of essential attributes of SDs of two particular types. To do that, SDs described by relational databases and SDs described by production systems are considered. Based on the knowledge of a structure of an SD given by a way of representation of the SD, the method and algorithms for extraction of essential attributes of such SDs are developed.

V. A CHOICE OF SUBJECT DOMAINS

The two groups of SDs are chosen due to their significant importance for solving applied problems. Studying SDs, described by production systems, is important because they are frequently used for definition of knowledge bases in, for example, recommendation systems. Extraction of essential attributes in such the systems, in particular, will allow us to simplify analysis of these systems. SDs, described by relational databases, include a wide range of SDs of production and services. Such a way of describing an SD shows its structure, logic of problems being solved in an SD; also, this type is used in most systems that automate information processes of production, business, etc. As result, an extraction of essential attributes of such SDs may help raise productivity of corresponding business.

A. Groups of essential attributes

In the context of solving mass problems, attributes of objects of SDs and attributes of SDs are to be divided into following groups:

- external;
- internal;
- constant.

External attributes are the attributes which values may be **directly** changed from outside while solving mass problems in an SD, e.g.: a current gear of a car, a price of a good, a doze of a drug for a treatment etc. These attributes also may allow managing other attributes of the SD and its objects indirectly by changing their state. Therefore, this group of attributes may help solve a problem of managing an SD, since these attributes are the only way to influence an SD.

Internal attributes are the attributes, values of which may be changed during functioning of an SD without direct external influence on them. This group of attributes displays a current state of an SD in general and its objects in particular, e.g.: current speed, an income from sales of a product, temperature of a patient etc. This group of attributes together with the previous group allows us defining different states of the same SD and the same objects during functioning of an SD. Possible applications include: description of a state of an SD and the optimal state in terms of these attributes, comparison of states of an SD, identification of an SD.

Constant attributes are the attributes, values of which are non-changed for an object of a given SD or for the given SD. E.g.: a required number of wheels of the car, a title of a product, a name of a patient etc. This group of attributes together with the previous group may help solve problems of an identification of an SD and a comparison of SDs.

Let us note that an attribute may simultaneously be both external and internal.

B. Algorithm for extracting essential attributes of production systems

Let us consider essential attributes in the context of solving problems in an SD. Essential attributes will be those attributes

of objects of an SD that will allow solving as many problems of the SD as possible, given that its objects are described only with their essential attributes.

Let us assume that problems in an SD are described as essential attributes of its objects or attributes of the SD itself with the help of other attributes of its objects. For the moment, the way in which these attributes can be found (it may be a formula or an algorithm) is not important. Only existence of such a way is significant. Then a set of problems, which is solved in the SD, may be described in a form of productions:

$$a_i^1 \wedge a_i^2 \wedge \ldots \wedge a_i^{n_i} \to b_i, \quad (1)$$

where $a_i^j \in A$ – attributes of objects of the SD, that are needed for solving the problem defined by i-th production, i.e. to find the attribute b_i.

Let us note that attributes that are in the left parts of the productions and cannot be found in the right parts may be placed to the external group. All other attributes may be placed to the internal group.

Let us also assume that the constructed set of productions does not contain cycle dependencies, i.e. it is impossible to obtain a rule

$$a_i^1 \wedge \ldots \wedge a_i^m \wedge b_i \wedge a_i^{m+1} \wedge \ldots \wedge a_i^{n_i} \to b_i \quad (2)$$

by substituting an attribute in the left part of a rule with attributes that it can be found with.

We assume that for each attribute there is only one production in the set of all productions where this attribute is in the right part.

Let us introduce a function k that shows how many attributes can be found with each attribute of all objects of an SD and let us build an algorithm of its computation. To do this, let us build a graph with vertexes corresponding to attributes of objects of an SD that are present in productions. For each pair of vertexes (a_i, a_j) we add a corresponding directed edge between them if there is a production in which there is a_i in its left part and a_j in its right part. Let us denote this graph G.

Let us introduce auxiliary functions:

$To(a_i)$ – a punctured neighbourhood of the vertex a_i;

$in(a_j)$ – an indegree of the vertex a_j;

$to(a_i)$ – an attribute that is adjacent to a_i (if there is only one such an attribute)

Algorithm 1.

Input: a graph G.

Execution:

For each attribute a_i calculate the function k for every attribute that is adjacent to a_j, i.e. for those attributes for which there is an edge (a_i, a_j) in the graph G;

Let

$$k(a_i) = 1 + \sum_{a_j \in To(a_i)} \frac{k(a_j)}{in(a_j)}. \quad (3)$$

Output: values of the function k for all vertexes of the graph.

Thus, function k shows for an attribute its contribution in finding another attributes of an SD taking into account its share in left parts of productions. We can prove that the sum of values of this function calculated on the vertexes with zero indegree is equal to the number of vertexes in the graph.

After calculating values of the function k, we sort attributes in descending order of their values and normalize them. The most essential attributes will be at the top of the list. By fixing different thresholds, it is possible to have a different number of attributes in the set of essential attributes, giving a possibility of solving a number of problems of initial SD in a truncated SD.

Let us consider now an SD with a set of productions where the same attribute may be obtained with different productions. We add to set of vertexes of an initially empty graph G all attributes from all productions. Next, we group productions by the attributes b_i in their right parts. For each group G_i let us add to the graph the vertex D_i corresponding to a disjunction of all rules of the group and an edge (D_i, b_i). For each rule P_j^i of a group we add a vertex C_j^i corresponding to conjunction of attributes in the left part of this rule and also edges (a_j^n, C_j^i) for each attribute $a_j^n \in L(P_j^i)$ ($L(P_j^i)$ is a set of attributes in the left part of the rule P_j^i) which reflect this conjunction of attributes in the graph. In addition, we add edges (C_j^i, D_i) showing belonging of these rules to the group.

Let A be a set of all attributes of objects of an SD, let C be a set of all added conjunction vertexes, let D be a set of all added disjunction vertexes. Thus, a set of all vertexes of the graph

$$V(G) = A \bigcup C \bigcup D. \quad (4)$$

For the obtained graph let us apply the following algorithm to calculate values of the function k.

Algorithm 2.

Input: a graph G.

Execution:

For each vertex v calculate a value of the function k for all its adjacent vertexes;

Calculate a value $k(v)$

a) If $v \in C \cup D$, let

$$k(v) = k(to(v)); \quad (5)$$

b) If $v \in A$, let

$$k(v) = 1 + \sum_{w \in To(v)} \frac{k(w)}{in(w)} . \quad (6)$$

Output: values of the function k for all vertexes of the graph.

Having these values, essential attributes can be extracted as in the previous case.

Extraction of essential attributes in such SDs may also allow, apart from decreasing the size of an SD with the least functional reduction, finding SDs with a similar set of the mass problems (set *T*).

C. Extracting essential attributes from relational databases

Let us consider SDs, described by relational databases (RDB). Let us solve a problem of extraction of essential attributes and their classification.

Let us consider a set of all queries $Q = \{q_i\}$ of an Information System (IS) to the RDB. Let $z : Q \to N_0$ be a function that associates each query with a number of its executions at a DB server.

Let $R = \{R_i\}$ be a set of all relations (tables) of the RDB, and X be a set of all attributes of the RDB. Let $f : (R \cup X) \times \mathrm{Q} \to \mathrm{N}_0$ be a function that shows a number of occurrences of an attribute or a relation in the given query.

In that case:

$$\forall R_i \in R \; \forall q \in Q \; f(R_i, q) = \sum_{x \in R_i} f(x, q) . \quad (7)$$

Let us define a function that shows a number of queries to an attribute or a relation by a given query:

$$g : (R \cup X) \times Q \to N_0 ; \; g(x, q) = z(q) f(x, q) . \quad (8)$$

Let us define a total number of queries to an attribute or a relation over all queries to the DB as a function

$$h : R \cup X \to N_0 , \; h(x) = \sum_{q \in Q} g(x, q) = \sum_{q \in Q} z(q) f(x, q) . \quad (9)$$

Let

$$h'(x) = \frac{h(x)}{\sum_{y \in (X \cup R)} h(y)} \quad (10)$$

and

$$h''(x) = \begin{cases} \dfrac{h(x)}{\sum_{y \in R} h(y)}, \forall x \in R \\ \dfrac{h(x)}{\sum_{y \in X} h(y)}, \forall x \in X \end{cases} . \quad (11)$$

Let us order attributes and entities by descending of $h'(x)$ or $h''(x)$. Having fixed a threshold $p \in [0,1]$, let us consider to be essential those attributes which satisfy $h'(x) \geq p$ (or $h''(x) \geq p$).

Having extracted essential attributes, let us determine a group that they belong to. If an attribute appears in a SET part of an UPDATE query and assigned values are calculated directly from values passed to this query from outside (e.g. as a parameters of a query), then let us consider this attribute as an external one. If an attribute appears in a SET part of an UPDATE query, but assigned values are calculated based on values of attributes of an RDB, then let us consider this attribute as an internal one. If an attribute does not belong to a SET part of any UPDATE query but only to an INSERT query, then let us consider this attribute as a constant one.

D. Extraction of essential attributes of a subject domain based on essential attributes of its objects

We assume that for a given SD all essential attributes of its objects have been extracted. Having that, let us consider a way of constructing a set of essential attributes of the SD itself for different SDs.

Let (E, V, T) be an SD, let $E = \{e_i\}$, where $\{a_{i,j}\}$ is a set of attributes of an object $e_i \in E$, $a_{i,j} \in A$. Having that, for all $a_{i,j}$ values of a function $p : A \to [0,1]$ on those values $a_{i,j}$ are known, where p shows a degree of belonging of an attribute to the set of essential attributes of the corresponding object. Let U be a set of essential attributes of the SD.

The first approach for construction of a set of essential attributes of an SD is to unite essential attributes of all objects of the SD. Thus:

$$U = \{a_{i,j} \in A \mid p(a_{i,j}) > 0\} . \quad (12)$$

This approach works assuming that an SD can be totally defined by attributes of objects that it consists of. An advantage of this approach is a great amount of information given by essential attributes of an SD. Disadvantages are not taking into account connections between objects, a large number of essential attributes of an SD, a potentially high level of unnecessary attributes in the list of essential attributes. This approach is good for comparing states of SDs, but is poor for analysis of an SD based on its essential attributes. The problems can be partially reduced by analysis of representations that are used for data presentation for a user while solving mass problems in this SD. Let $w : A \to \{0,1\}$ be a function that shows whether a given attribute is used in displaying results of solutions of mass problems in the given SD. Having that, a set of essential attributes of an SD is defined as:

$$U = \{a_{i,j} \in A \mid p(a_{i,j}) w(a_{i,j}) > 0\} . \quad (13)$$

The second approach for construction of a set of essential attributes of an SD is to build a set of attributes that are aggregations of values of essential attributes of an SD. E.g.: let $A = A_1 \cup A_2$, where A_1 – quantitative attributes, A_2 – not quantitative attributes. Let *value* be a function defined for

attributes from the set A_1 that returns a set of all values of an attribute that are present in the current state of an SD. Let *freq* be a function that for a given attribute returns a vector of pairs (v_i, n_i), where v_i runs through all met values of the attribute, and n_i is equal to the number of such occurrences in the current state of an SD. Let us define a set of essential attributes as follows:

$$U = \{\min(a), \max(a), E(a) \mid a \in A_1\} \cup \{\text{freq}_a \mid a \in A_2\}, \quad (14)$$

where values of given attributes for an SD will be calculated in the following way:

$$\max(a) = \max(value(a)), \quad (15)$$

$$\min(a) = \min(value(a)), \quad (16)$$

$$\text{E}(a) = \text{avg}[value(a)], \quad (17)$$

$$\text{freq}_a = freq(a), \quad (18)$$

where $\text{avg}[value(a)]$ – the mean value of numbers from the set $value(a)$. This approach fits better for analysis of an SD, but displays its state worse because of a loss of concrete data.

The next possible approach is to order essential attributes of objects of an SD by descending their importance (based on a characteristics calculated at the stage of their extraction), with further cutting off a part of attributes. A set of essential attributes of an SD will be:

$$U(t) = \{a_{i,j} \in A \mid p(a_{i,j}) > t\}, \quad (19)$$

where $t \in (0,1)$ – a threshold. This approach is similar to the first one, but a number of essential attributes varies depending on a value of the parameter t. This allows managing level of abstraction in a description of an SD by its essential attributes.

Furthermore, essential attributes of an SD may not only be attributes of objects of the SD, but objects themselves. We propose to extract such attributes by comparing objects, for example, by a sum of estimations of their attributes that shows whether an attribute is essential or not. It is suggested to associate with each object $e_i \in E$ an estimate

$$s_i = \sum_{a_{i,j} \in e_i} p(a_{i,j}). \quad (20)$$

Having objects in descending order of such the sums, let us choose some start elements of a list as essential attributes.

While extracting objects as essential attributes of an SD, it is also possible to take into account a structure of an SD. Let us assume that based on connections between objects, cluster analysis of a graph of objects have been conducted. Let $X_i \subset E$ ($X_i \cap X_j = \varnothing$, $i \neq j$, $\cup X_i = E$) be clusters of objects of an SD. As essential attributes of an SD let us choose a set of objects

$$\{\hat{e}_i \mid \hat{e}_i = \arg\max_{e_k \in X_i} \sum_{a_{k,j} \in e_k} p(a_{k,j})\}, \quad (21)$$

i.e. from each cluster the most weighty objects are taken. This set of essential attributes reflects a set of main component parts of an SD near which other objects are situated.

CONCLUSION

The concept of essential attributes is introduced in the paper. Methods and algorithms to find such the attributes and their possible applications are proposed. Our further research will be focused on detailing and classification of essential attributes, development of specific algorithms for their detection as well as on development of algorithms for solution of mathematical optimization problems in different SDs.

REFERENCES

[1] Malakhov E.V. (2010). Expansion of operations on subject domains metamodels taking into account the mass problems. *Eastern-European journal of enterprise technologies*, Vol. 5/2 (47), 20-24. – (in Russian).

[2] Hjorland, B. (2002). Domain analysis in information science. Eleven approaches - traditional as well as innovative. *Journal of Documentation, 58*(4), 422-462.

[3] Chen, N.-S, Kinshuk, Wei, C.-W., Chen, H.-J. (2008). Mining e-Learning domain concept map from academic articles. *Journal of Computational Information Systems*, 50, 1009-1021.

[4] Jiang, W., Hui, X. (2008) Automatic keyword extraction based on imbalanced classification methods. *Journal of Computational Information Systems*, 9, 8483-8490.

[5] Yang, Z., Lei, J., Fan, K., Lai, Y. (2013). Keyword extraction by entropy difference between the intrinsic and extrinsic mode. *Physica A,* Vol. 392 №19, 4523-4531.3

[6] Naihanova, L.V., Aiusheeva, N.B., Khaptakhaeva, N.B. (2007). Building semantic network of a subject domain basing on the extraction of data from a scientific text. *University proceedings. Volga region. Technical sciences,* 4, 51-61. – (in Russian)

[7] Malakhov E.V., Stanovskiy, A.L. (2010). The objects definition for subject domains management. *Eastern-European journal of enterprise technologies*, Vol. 1/2 (49), 47-50. – (in Russian).

[8] Malakhov E.V., Tonkonogiy, V.M. (2010). Elementary objects as a basis of object cores of subject domains. *Eastern-European journal of enterprise technologies*, Vol. 1/2, 128-130. – (in Russian).

[9] Meyer, B. (1997). Object-oriented software construction (2nd ed.). New Jersey: Prentice hall PTR.

[10] Connolly, T. M., & Begg, C. E. (2001). Database Systems: a Practical Approach to Design, Implementation, and Management (3rd ed.). Boston: Addison-Wesley.

[11] Vitaliy Mezhuyev, George Vostrov. Problems of development of information systems on subject domains. Artificial Intelligence. - № 4. - 2008. - Pp. 736-746.

[12] Vitaliy Mezhuyev. Information technology for domain-specific mathematical modelling. - Berdyansk: BSPU, 2013. – 310 p.